On deep learning tools for scientific discovery in healthcare

by

Mukund Sudarshan

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Computer Science

New York University

September, 2022

Professor Rajesh Ranganath

Professor Oded Regev

Dedication

To my family and friends, with affection.

Acknowledgements

The work presented was supported by many individuals.

I thank my advisors Rajesh Ranganath and Oded Regev. They have been highly influential in the way I approach research. I am grateful they allowed me the flexibility to pursue projects that interest me while still being involved and providing helpful guidance along the way.

I thank my committee members Lakshmi Subramanian, Carlos Fernandez-Granda, and Rob Fergus. Lakshmi's advice early in my PhD helped shape my research interests and career goals.

I thank my research collaborators: Aahlad Manas Puli, Wes Tansey, Raghav Singhal, Neil Jethani, Susan Liao, Yin Aphinyanaphongs, Sriram Sankararaman, Narges Razavian, Vincent Major, Sumit Chopra, and Dan Sodickson. Working with them was an immense pleasure and inspired me to pursue applications of machine learning in healthcare.

New York University and the Department of Computer Science have been invaluable. The department administrators have always been friendly and helpful. Shenglong Wang at NYU HPC has been critical in helping with any and all compute issues.

I thank another mentor of mine, Matcheri Keshavan, whose lab I worked in for a summer. If not for that experience, I would likely have never pursued a PhD.

My graduate experience would not have been the same without the support of my family and friends. For them I am truly grateful.

Abstract

Scientists validate hypotheses by building mathematical models of the real world. They make inferences by checking if their models are supported by data. Often, the models are hand-crafted and do not accurately reflect real processes. This often leads to low power in making scientific discoveries or even false discoveries.

Machine learning can solve these issues in several ways. By allowing data to inform the construction of models, scientists can use machine learning to create more powerful statistical hypothesis testing procedures, or build more realistic models of underlying processes.

This thesis details techniques to address both of these approaches. First we address the creation of machine learning-based statistical discovery procedures for scientific discovery. Specifically, we discuss how machine learning can be used to construct conditional independence tests, which are used to identify causal links in data. We detail how such methods can be used to control the false discovery rate when testing multiple hypotheses. We then apply these techniques to two important problems in healthcare. We solve a timely problem in medical informatics: identifying a small set of variables that are highly informative of whether an ICU patient with Covid will experience an adverse event. At the height of Covid in 2020, NYU doctors used a deployed version of this tool to quickly identify patients to discharge and free up beds in the ICU. We also apply our methods to a problem in cancer genomics, where the goal is to identify a set of gene mutations that are most predictive of tumor metastasis. In the near future, we expect tools like ours to lead to targeted gene therapies that tailor treatments to the mutations present in an individual's

tumor.

Next we detail the construction of an interpretable machine learning model that helps understand an important step in the creation of proteins. Specifically, we build a model to understand RNA splicing. Our model accurately predicts splicing outcomes across a large dataset of sequences, but more importantly leads to several biologically validated insights. We use the interpretable nature of our model to infer that most splicing decisions are a function of a small set of short sequence features. We further learn that certain pre-mRNA secondary structures strongly inhibit the inclusion of an exon in the final mRNA transcript. Finally, we validate these model-driven findings by carefully designing experiments for the wet lab.

Contents

List of Figures

List of Tables

1 Introduction

Progress is made in scientific disciplines by formulating and validating hypothesis with data. The validation procedure often involves building a model of the world, then using this model to report a finding about the data. For example, if geneticists seek to identify genes that are most likely responsible for height, they might collect the genotype of many individuals and test the correlation between the presence of each gene and the heights of these individuals. If any of these genes correlate highly with height, they are deemed important predictors. While researchers in this example may not explicitly construct a model of the world, they are doing so implicitly via assumptions about their data. By using correlation as a measure of dependence, they assume that genes impact height in a linear way.

Sometimes, these assumptions are wrong and lead to false scientific discoveries. To lessen the dependence on strong assumptions, data can be used inform the construction of models. In recent years, data-dependent models have increasingly been adopted.

In this thesis, we cover instances of two broad techniques that use machine learning to build data-driven models for scientific discovery. The first technique is based on causal discovery: when a scientist wishes to find the causal links between a set of predictors and a response. We discuss how and when machine learning can help achieve this goal. The second technique involves constructing an interpretable machine learning model of the real world, then using this model to make inferences about underlying processes. We discuss this technique in the context of a novel model for an important biological process - RNA splicing.

The remainder of this thesis is structured as follows. Chapter 2 discusses several foundational papers for two types of scientific discovery. In the first part of chapter 2, we introduce the problem of conditional independence testing. We discuss the two main classes of conditional independence tests and discuss their advantages and disadvantages. We dive deeper into the latter of these ways to test conditional independence – Model-X methods – and discuss the foundational paper that inspired much subsequent work. In the second part of chapter 2 we introduce interpretable machine learning as a tool for scientific discovery. We discuss why a practitioner might want to use an interpretable model or an interpretability technique to explain a black-box model.

In chapter 3 we introduce deep direct likelihood knockoffs (DDLK), a deep generative model for use in controlled variable selection procedures. Given a set of inputs $\mathbf{x}_1, \ldots, \mathbf{x}_d$ and a response $\mathbf{y}$, scientists often seek to identify the variables in the Markov blanket of $\mathbf{y} \mid \mathbf{x}_1, \ldots, \mathbf{x}_d$. This involves testing the conditional independence: $\mathbf{y} \perp\!\!\!\perp \mathbf{x}_j \mid \mathbf{x}_{-j}$, where $\mathbf{x}_{-j}$ represents all inputs but the jth one. To simultaneously test the conditional independence of each variable $\mathbf{x}_j$ with $\mathbf{y}$ given the remaining variables $\mathbf{x}_{-j}$, Candes et al. [2018] introduce a framework termed Model-X "knockoffs." This framework guarantees that if there exists a set of variables $\widetilde{\mathbf{x}}_1, \ldots, \widetilde{\mathbf{x}}_d$ that is exchangeable with $\mathbf{x}_1, \ldots, \mathbf{x}_d$ and is sampled independent of $\mathbf{y}$, then the rate at which inputs not in the Markov blanket of $\mathbf{y}$ are selected (this is called the false discovery rate (FDR)) can be controlled at a user-specified level. The quality of the knockoff variables $\widetilde{\mathbf{x}}_1, \ldots, \widetilde{\mathbf{x}}_d$ determines the accuracy with which a user can control the FDR. Generating knockoffs, especially in high dimensions, is a challenging problem because the joint distribution of $\mathbf{x}_1, \ldots, \mathbf{x}_d$ must be known or learned from data. We demonstrate that DDLK outperforms several baselines at generating high quality knockoff variables for a variety of data distributions.

In chapter 4 we introduce contrarian randomization test (CONTRA), a method that addresses a common issue in conditional randomization tests (CRTs). CRTs test the independence of $\mathbf{y}$ and $\mathbf{x}$ given $\mathbf{z}$, written as $\mathbf{y} \perp\!\!\!\perp \mathbf{x} \mid \mathbf{z}$, and require modeling $p(\mathbf{x} \mid \mathbf{z})$ well to control the FDR. They

compute a test statistic, some function T of a dataset of $(\mathbf{x}, \mathbf{y}, \mathbf{z})$ samples, then compare the T's value to its empirical null distribution. If the probability that the test statistic came from the null distribution is very low, then the null hypothesis of conditional independence is rejected. Often the $p(\mathbf{x} \mid \mathbf{z})$ distribution is not modeled well, and many popular test statistics will erroneously reject the null hypothesis: resulting in FDR violations. CONTRA is a test statistic designed specifically for such scenarios. The CONTRA test statistic computes a mixture of two functions: one using samples of the real data $(\mathbf{x}, \mathbf{y}, \mathbf{z})$, and the other using samples of $(\widetilde{\mathbf{x}}, \mathbf{y}, \mathbf{z})$ where $\widetilde{\mathbf{x}}$ is drawn from the estimated $p(\mathbf{x} \mid \mathbf{z})$ distribution. We demonstrate that CONTRA yields a noticeable improvement in the ability to control FDR compared to several baselines while still yielding a high probability of correctly rejecting the null hypothesis.

In chapter 5 we introduce decoupled independence test (DIET), a type of CRT that leverages marginal dependence measures to test conditional independence. The power of a CRT, or the probability that the null hypothesis is correctly rejected, depends on how well the test statistic T measures the *additional* information that $\mathbf{x}$ contains about $\mathbf{y}$ having already observed $\mathbf{z}$. This is a challenging problem as test statistics must make tradeoffs between computational efficiency and power. For example, some methods to achieve computational efficiency have to use a subset of the available data to compute T, which reduces their ability to detect conditional dependence. Other methods use all the available data but make restrictive assumptions about the data generating process or rely on heuristics. DIET is similar in spirit to the latter but relies on a different set of assumptions that yield better empirical performance. DIET computes the marginal dependence between two random variables that are independence if and only if $\mathbf{x} \perp\!\!\!\perp \mathbf{y} \mid \mathbf{z}$. We prove that for a class of data distributions DIET is the most powerful conditionally valid CRT.

In chapters 6 and 7 we discuss applications of CRTs to two important problems in healthcare. In chapter 6 we address a timely problem faced by doctors at NYU Langone during the height of the Covid-19 pandemic in March 2020. ICU beds were filling up and doctors needed a tool to help them identify patients that could be safely discharged, making room for the patients most in need

of urgent care. With over 60 different variables in a patient's chart and no official guidance on how to treat patients, doctors sought a data-driven tool that could help them better understand a patient's risk of an adverse event. Using CRTs, we identified a subset of just over a dozen variables that were sufficient to accurately predict whether or not a patient would experience an adverse event in the next 96 hours. Using the subset of important variables, we built and deployed an interpretable machine learning model that helped doctors better understand the underlying mechanism behind a patient's risk of an adverse event. It would output a risk score every time an ICU patient was given a complete blood count. In just a few months of deployment, our model was used to make over a half a million predictions across NYU hospitals.

In chapter 7 we detail an ongoing effort by Memorial Sloan Kettering Cancer Center (MSKCC) to identify cancer patients that are likely to benefit from gene therapy. To achieve this, the first step is to identify which tumor mutations likely cause metastasis. Patients can then be screened using only this subset of mutations. The goal is to identify as many patients as possible while controlling the rate at which they are brought in without any causal mutations. Using a large dataset of tumor mutations across various sites, we outline a proof-of-concept tool that uses CRTs developed in this thesis to identify a set of causal mutations for metastasis. We observe in challenging simulations that this tool outperforms popular baselines at selecting causal mutations. In the near future, we expect to see such a tool be deployed for screening purposes at MSKCC.

In chapter 8 we build an interpretable machine learning model to understand a fundamental process in biology: RNA splicing. Splicing involves the removal of introns from precursor messenger RNA (pre-mRNA) and the joining of exons to form mature messenger RNA (mRNA). mRNA is used further downstream to create useful molecules like proteins. Much like a film editor cuts out irrelevant material, the spliceosome cuts out intronic regions. However, its behavior is currently not well understood. We construct a machine learning model to accurately predict the behavior of the spliceosome given a sequence, and understand the intermediate steps involved in determining a splicing outcome. We use this model to derive several biological in-

sights. For example, splicing decisions are mostly determined by a small set of short sequence features called motifs, and secondary structures formed by the pre-mRNA prevent exonic regions from being identified as such. We validate these findings with carefully constructed biological experiments.

2 Select topics in machine learning for scientific discovery: a brief review

Here we discuss several foundational papers for two types of scientific discovery. First we introduce the problem of conditional independence testing. We discuss the two main classes of conditional independence tests and discuss the advantages and disadvantages of each. We dive deeper into the latter of these ways to test conditional independence – Model-X methods – and discuss the foundational paper that inspired much subsequent work.

In the second part of this chapter we introduce interpretable machine learning as a tool for scientific discovery. We discuss why a practitioner might want to use an interpretable model or an interpretability technique to explain a black-box model. Since this topic is very broad, we limit the scope of the discussion to recent methods most relevant to this thesis.

2.1 Conditional independence testing

Conditional independence testing plays a key role in causal modeling. A causal graph is a graphical way to represent the conditional independences that exist in a data generating distribution. Conditional independence tests are often used to build causal graphs from data [Spirtes et al.

2000; Silverstein et al. 2000; Tsamardinos et al. 2003]. Conditional independence also provides a framework to reason about invariant prediction, which aims to identify a subset $\mathbf{x}_s$ of variables $\mathbf{x}_1, \ldots, \mathbf{x}_d$ for a target $\mathbf{y}$ such that $\mathbf{y}$ is conditionally independent of an environment variable $\mathbf{u}$ given $\mathbf{x}_s$ [Peters et al. 2016; Heinze-Deml et al. 2018].

Conditional independence is often tested using a statistical hypothesis testing framework. Doing so allows scientists to explicitly control the type-1 error rate: the rate at which the null hypothesis of conditional independence is erroneously rejected. If a scientist is testing multiple hypotheses, a hypothesis testing framework also allows them to control the false discovery rate (FDR):

$$\text{FDR} = \frac{\text{\# of false rejections}}{\text{\# number of total rejections}}.$$

While a hypothesis testing framework allows for error control, it does not necessarily guarantee any power to reject the null hypothesis even with a countably infinite dataset. Shah and Peters [2020] prove that without restrictions on the set of possible null distributions or on the set of alternate distributions, power is not guaranteed. Specifically, they show that for a test where the probability of rejecting the null hypothesis using a set of N samples of $(\mathbf{x}, \mathbf{y}, \mathbf{z})$ is α, it is possible to construct a null distribution where samples from the null are arbitrarily close in ℓ_∞-norm to samples from the true distribution. This means the type-1 error rate would also be greater than α.

One way to achieve power then is to restrict the class of null distributions so such constructions are not possible. This restriction is achieved by making assumptions about the data generating process or about the set of null distributions. Next we describe two classes of conditional independence tests and the assumptions they make to control the type-1 error rate and achieve power.

Model-Y methods We term the first class of conditional independence tests "Model-Y" methods. The typical setup is to assume a parametric form of $\mathbf{y} \mid \mathbf{x}, \mathbf{z}$ like a linear model with standard Gaussian noise. This includes procedures that test for edges in Bayesian networks [Koller and Friedman 2009; Spirtes et al. 2000; Cheng et al. 1998; De Campos and Huete 2000]. There are other methods that don't assume a parametric form, but make other assumptions like smoothness or assume a point null distribution [Fukumizu et al. 2007; Zhang et al. 2012; Gretton et al. 2012; Doran et al. 2014; Lee and Honavar 2017]. In practice, these approaches do not strictly require their assumptions to be met to be useful. However there are common instances where an alternate set of assumptions might be better suited for controlling error rates. We discuss these assumptions next.

2.1.1 Model-X methods

Model-X methods [Candes et al. 2018] assume the ability to sample from the covariate distribution $p(\mathbf{x} \mid \mathbf{z})$ to control the type-1 error rate. They do not need to assume anything about the $\mathbf{y} \mid \mathbf{x}, \mathbf{z}$ distribution. This is often advantageous in domains like biology and healthcare for several reasons. First, the covariate distribution is often known well. For example, in genomics, modeling the distribution of single nucleotide polymorphisms (SNPs) is a well studied problem. However, how these SNPs relate to a biologically complex phenotype is not yet well understood. Second, a large amount of unsupervised data is often available. Collecting a patient's genotype is fairly cheap these days but collecting phenotypic information is more expensive as it requires followups or other expensive measurements. Similarly in healthcare, a hospital may have the vital signs and blood test results of many people, but perhaps only a few of these people have a rare type of cancer. In such cases, the large unsupervised data can be leveraged to build a good model for $p(\mathbf{x} \mid \mathbf{z})$ and the quality of the $\mathbf{y} \mid \mathbf{x}, \mathbf{z}$ model doesn't affect the type-1 error rate. Next we discuss two model-X methods: knockoffs, and conditional randomization tests.

2.1.2 Model-X knockoffs

In this section we first provide an overview of the problem model-X methods were designed to solve: controlled variable selection, then describe formally what model-X knockoffs are. Given a set of variables $\mathbf{x}_1, \ldots, \mathbf{x}_d$ and a response $\mathbf{y}$, the goal of controlled variable selection is to test the conditional independence of each variable $\mathbf{x}_j$ with $\mathbf{y}$ having observed the other variables $\mathbf{x}_{-j}$. Formally, it simultaneously tests the following null hypotheses:

$$\begin{aligned} \mathcal{H}_0^{(1)} &: \mathbf{x}_1 \perp\!\!\!\perp \mathbf{y} \mid \mathbf{x}_2, \ldots, \mathbf{x}_d \\ &\vdots \\ \mathcal{H}_0^{(d)} &: \mathbf{x}_1 \perp\!\!\!\perp \mathbf{y} \mid \mathbf{x}_1, \ldots, \mathbf{x}_{d-1} \end{aligned}$$

while controlling the false discovery rate (FDR). Letting $\mathbf{x}_s \subseteq \{\mathbf{x}_1, \ldots, \mathbf{x}_d\}$ be the set of variables for which the corresponding null hypothesis is rejected, the FDR is defined as:

$$\text{FDR} = \mathbb{E}\left[\frac{|\{j : \mathbf{x}_j \in \mathbf{x}_s, \mathbf{x}_j \perp\!\!\!\perp \mathbf{y} \mid \mathbf{x}_{-j}\}|}{\max(1, |\{j : \mathbf{x}_j \in \mathbf{x}_s\}|)}\right].$$

Model-X knockoffs are a set of random variables $\widetilde{\mathbf{x}}_1, \ldots, \widetilde{\mathbf{x}}_d$ that are exchangeable with $\mathbf{x}_1, \ldots, \mathbf{x}_d$ and are conditionally independent with $\mathbf{y}$ given $\mathbf{x}_1, \ldots, \mathbf{x}_d$. This means that given any set of indices $H \subseteq \{1, \ldots, d\}$, they must satisfy the following properties:

$$\begin{aligned} p([\mathbf{x}_1, \ldots, \mathbf{x}_d, \widetilde{\mathbf{x}}_1, \ldots, \widetilde{\mathbf{x}}_d]) &= p([\mathbf{x}_1, \ldots, \mathbf{x}_d, \widetilde{\mathbf{x}}_1, \ldots, \widetilde{\mathbf{x}}_d]_{\text{swap}(H)}) \\ \mathbf{y} &\perp\!\!\!\perp [\widetilde{\mathbf{x}}_1, \ldots, \widetilde{\mathbf{x}}_d] \mid [\mathbf{x}_1, \ldots, \mathbf{x}_d]. \end{aligned}$$

This means that the joint distribution of the covariates and knockoffs is invariant to a swapping of any set of indices H between the covariates and knockoffs.

To perform controlled variable selection with knockoffs, [Candes et al. 2018] compute a scalar

score W_j – a function of the data and knockoffs – for each null hypothesis $\mathcal{H}_0^{(j)}$ called a "feature statistic":

$$W_j = w_j([\mathbf{x}_1, \ldots, \mathbf{x}_d, \widetilde{\mathbf{x}}_1, \ldots, \widetilde{\mathbf{x}}_d], \mathbf{y}).$$

To control the FDR, these feature statistics must obey the following properties. Under the null hypothesis $\mathcal{H}_0^{(j)}$, the distribution of the feature statistic W_j should be symmetric around 0. If the null hypothesis is to be rejected, the distribution of W_j should have a positive skew. Given feature statistics that obey these properties, knockoffs then outline a simple procedure for testing multiple null hypotheses. At a nominal FDR rate α, they return a set of features $\mathbf{x}_s$ such that for all $\mathbf{x}_j \in \mathbf{x}_s$, W_j is greater than a threshold τ chosen as follows:

$$\tau = \min \left\{ t > 0 : \frac{|\{j : W_j \leq -t\}| + 1}{|\{j : W_j \geq t\}|} \leq \alpha \right\}.$$

The intuition behind the threshold τ is as follows. The FDP can be estimated by computing the ratio of falsely rejected null hypotheses in $\mathbf{x}_s$ to the size of $\mathbf{x}_s$. The denominator is exactly the size of $\mathbf{x}_s$. Since we know that null feature statistics are symmetric around 0, the number of null W_j that are greater than or equal to t should be equal in distribution to the number of null W_j that are less than $-t$. This quantity is in the numerator. The $+1$ in the numerator makes the procedure slightly more conservative. Choosing the smallest threshold τ that allows for FDR control at level α yields the most number of null hypothesis rejections at α. This is important because choosing a threshold τ that is too large will result in a procedure that fails to select even important variables.

While testing multiple hypotheses with knockoff feature statistics is very fast, generating knockoffs is a difficult problem. There are many issues with generating high-dimensional random variables. Apart from the computational challenges, if a practitioner is not careful, memorizing the data could trivially satisfy the properties required of knockoffs, but will result in a procedure that fails to select any variables. Further, it is not obvious how to design feature statistics that

yield high power to reject null hypotheses.

To this end, there has been much follow-up work to the knockoffs paper to address both the issue of generating knockoffs and choosing feature statistics [Romano et al. 2020; Sudarshan et al. 2020; Barber et al. 2020; Bates et al. 2021; Bellot and van der Schaar 2019; Jordon et al. 2019; Sudarshan et al. 2021; Sesia et al. 2019; Spector and Janson 2022]. This includes work presented in chapter 3 of this thesis. One issue the knockoff framework cannot solve is testing a small number or a single null hypothesis. In such cases the control a practitioner has over the FDR is very coarse. For example with only a single feature statistic W_j, it is impossible to reject the null hypothesis at any $\alpha < 1$, meaning FDR control is unavailable. To address this problem, Candes et al. [2018] also introduce the conditional randomization test (CRT).

2.1.3 Conditional randomization testing

The CRT is another Model-X method for testing conditional independence. It uses many of the same primitives as knockoffs but differs in a few key ways. In this section we provide a brief overview of the CRT framework. We focus on a single hypothesis test $\mathbf{x} \perp\!\!\!\perp \mathbf{y} \mid \mathbf{z}$ for most of the discussion, then tie it back to the context of controlled variable selection at the end.

The CRT computes a p-value for the null hypothesis, allowing for precise control of the type-1 error rate: the probability of erroneously rejecting the null. To control the type-1 error rate, CRT assumes the practitioner can sample values of $\mathbf{x} \mid \mathbf{z}$. Here is how the CRT p-value is computed.

The CRT computes a quantity called a "test statistic": a function T of the dataset $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$ sampled from $p(\mathbf{x}, \mathbf{y}, \mathbf{z})$:

$$\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}} = \{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}, \mathbf{z}^{(i)})\}_{i=1}^{N}$$

$$T(\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}). \qquad \text{(Test statistic)}$$

It then generates M "null datasets" using $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$:

$$\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(1)}, \mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(2)}, \ldots, \mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(M)}.$$

Each null dataset is identical to $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$ but values for $\mathbf{x}$ are replaced with a resampled value $\widetilde{\mathbf{x}} \sim p(\mathbf{x} \mid \mathbf{z})$. These $\widetilde{\mathbf{x}}$ variables are analogous to knockoff variables. For example, to generate the ith sample of $\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m)}$, the CRT makes a copy of the ith sample of $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$, $(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}, \mathbf{z}^{(i)})$, and replaces $\mathbf{x}^{(i)}$ with a draw from the conditional distribution $p(\mathbf{x} \mid \mathbf{z} = \mathbf{z}^{(i)})$. The CRT then applies the function T to each null dataset:

$$T(\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(1)}), T(\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(2)}), \ldots, T(\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(M)}).$$

Each of the terms in this sequence is called a "null statistic." Finally, to compute a p-value for the hypothesis test, the CRT computes the relative rank of the test statistic among the null statistics:

$$p = \frac{1}{M+1}\left(1 + \sum_{m=1}^{M} \mathbb{1}(T(\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}) \leq T(\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m)}))\right).$$

To test multiple null hypotheses, the CRT procedure can be applied to compute multiple p-values. For FDR control, any multiple testing correction procedure can be applied to the resulting set of p-values [Tukey 1953; Benjamini and Hochberg 1995; Benjamini and Yekutieli 2001].

The CRT allows for precise control of the type-1 error rate for a single hypothesis by varying M. For example, to reject a hypothesis at the level 0.01, the smallest p-value achieved must be less than 0.01. Solving for M, it is clear that the minimum number of null datasets to be sampled is 100. The minimum p-value achieved here is $\frac{1}{1+100} < 0.01$.

While the CRT has the advantage of precise error control over knockoffs, computing T many times can be a computationally intensive process. Therefore, choosing T that balances computational efficiency and power is a challenging problem. Further, the distribution $p(\mathbf{x} \mid \mathbf{z})$ must

often be inferred from data. To address these problems, there have been several recent proposals [Sudarshan et al. 2021; Liu and Janson 2020; Tansey et al. 2018a,b]. This includes work presented in chapters 4 and 5 of this thesis.

2.2 Machine learning interpretability

Broadly speaking, there are two schools of thought regarding interpretability. The first prioritizes predictive accuracy over explainability, and seeks to explain only models that fit the data well. The second believes that models used to understand underlying data generating processes must be human-interpretable by construction. We highlight some common examples of techniques in each school of thought, then discuss more generally when they should be used.

2.2.1 Interpretability methods

ML researchers and practitioners generally agree that understanding models is important. However, there is often a tradeoff between human interpretability and model complexity. For example the most powerful machine learning methods used in practice contain billions of parameters and follow decision logic that is too complicated for a human to understand. As a result, many believe that external tools must be built to explain the predictions of these "black-box" models. Limiting the structure of powerful prediction models for human interpretability may limit their performance.

To this end, there has been a large body of work on interpretability methods that seek to explain the predictions of black boxes. There is much work on interpretability, so in the context of this thesis, we focus on the largest subset of interpretability work: feature importance methods.

Feature importance methods. Feature importance methods rank input features to a model. The following are some examples. Hastie et al. [2009] permutes the inputs to a model and mea-

sure changes in the output. Many recent methods focus on neural networks and inspect the gradient of the output with respect to an input [Baehrens et al. 2010; Zeiler and Fergus 2014]. Other gradient-based methods include guided-backpropagation [Springenberg et al. 2014], which assigns attributes to hidden unit activations within a neural network, and baseline attribution methods [Sundararajan et al. 2017] which employ a baseline to use as a comparison to the inputs.

Beyond gradient-based methods, there are also surrogate model explainers, which are auxiliary models designed to output importance scores for each input feature. Examples include L2X [Chen et al. 2018], INVASE [Yoon et al. 2019], and FastSHAP [Jethani et al. 2021]. There are also many methods that estimate Shapley values [Lundberg and Lee 2017; Ribeiro et al. 2016; Covert and Lee 2021]: a game-theoretic quantity that assigns a score to each input feature that represents the expected difference in model performance with and without the feature over all possible feature subsets.

In general, feature importance methods are most useful to visualize a model's preference for inputs. This can help debug issues like spurious correlations [Geirhos et al. 2020], where a model may be using an artifact in the data rather than a causal feature. No one technique is ideal for all use cases. For example, gradient-based methods fail to correctly rank features in simulation studies conducted by [Jethani et al. 2021]. While surrogate model explainers perform better, they are expensive to train.

2.2.2 Interpretable machine learning models

We previously discussed a tradeoff between explainability and performance in the most powerful machine learning models. In many instances, machine learning models designed to be interpretable achieve sufficiently high predictive performance. Proponents of the interpretable models school of thought believe that as much as possible, the machine learning model must be parameterized such that its decision logic is easily understood by humans.

Examples of interpretable models include linear methods, which model a response $\mathbf{y}$ as an

affine transformation of input features $\mathbf{x}_1, \ldots, \mathbf{x}_d$: $\mathbf{y} = \beta_0 + \sum_{j=1}^{d} \beta_j \mathbf{x}_j$. The weights β_j can be used to rank input features (assuming all inputs are on the same scale), and inform the user about how varying an input feature changes the prediction. Similarly, a generalized additive model extends this concept by replacing linear transformations $\beta_j \mathbf{x}_j$ with more flexible parameterized functions $f_{\beta_j}(\mathbf{x}_j)$. This allows an input feature to have a nonlinear relationship with the prediction. Decision trees provide an alternative route to explaining predictions and work well for categorical data.

Unless there is a large gap between the best interpretable model and black-box model in terms of prediction, using an interpretable model is the preferred tool for scientific discovery. Often this gap is nontrivial and practitioners must choose between explainability and performance. However, with careful model architecture design elements of powerful ML models like convolutional networks or transformers can be used to create an interpretable model. In chapter 8, we discuss the construction of an interpretable convolutional network for an RNA sequence-based prediction task. From this network we derive several biologically relevant insights.

3 | Deep direct likelihood knockoffs

Predictive modeling often uses black box machine learning methods, such as deep neural networks, to achieve state-of-the-art performance. In scientific domains, the scientist often wishes to discover which features are actually important for making the predictions. These discoveries may lead to costly follow-up experiments and as such it is important that the error rate on discoveries is not too high. Model-X knockoffs [Candes et al. 2018] enable important features to be discovered with control of the FDR. However, knockoffs require rich generative models capable of accurately modeling the knockoff features while ensuring they obey the so-called "swap" property. In this chapter, we develop Deep Direct Likelihood Knockoffs (DDLK), which directly minimizes the KL divergence implied by the knockoff swap property. DDLK consists of two stages: it first maximizes the explicit likelihood of the features, then minimizes the KL divergence between the joint distribution of features and knockoffs and any swap between them. To ensure that the generated knockoffs are valid under any possible swap, DDLK uses the Gumbel-Softmax trick to optimize the knockoff generator under the worst-case swap. We find DDLK has higher power than baselines while controlling the false discovery rate on a variety of synthetic and real benchmarks including a task involving a large dataset from one of the epicenters of COVID-19.

3.1 Motivation for DDLK

We motivate DDLK with the following observation. As discussed in section 2.1.2, the swap property

$$[\mathbf{x}, \widetilde{\mathbf{x}}] \overset{d}{=} [\mathbf{x}, \widetilde{\mathbf{x}}]_{\text{swap}(H)} \tag{3.1}$$

is required of any valid knockoff variables $\widetilde{\mathbf{x}}$. That is, the joint distribution of data and knockoffs $[\mathbf{x}, \widetilde{\mathbf{x}}]$ should be invariant to swapping a set of coordinates H between $\mathbf{x}$ $\widetilde{\mathbf{x}}$. One way to check if the swap property is satisfied is to check if the KL divergence between the original and swapped distributions is zero. Formally, let H be a set of indices to swap, and $\mathbf{z} = [\mathbf{x}, \widetilde{\mathbf{x}}]$, $\mathbf{w} = [\mathbf{x}, \widetilde{\mathbf{x}}]_{\text{swap}(H)}$. Then under any such $H \subseteq [d]$:

$$\text{KL}(q_{\mathbf{z}} \parallel q_{\mathbf{w}}) = \mathbb{E}_{q_{\mathbf{z}}(\mathbf{z})}\left[\log \frac{q_{\mathbf{z}}(\mathbf{z})}{q_{\mathbf{w}}(\mathbf{z})}\right] = 0. \tag{3.2}$$

A natural algorithm for generating valid knockoffs might be to parameterize each distribution above and solve for the parameters by minimizing the LHS of eq. (3.2). However, modeling $q_{\mathbf{w}}$ for every possible swap is difficult and computationally infeasible in high dimensions. Theorem 3.1.1 provides a useful solution to this problem.

Theorem 3.1.1. *Let μ be a probability measure defined on a measurable space. Let f_H be a swap function using indices $H \subseteq [d]$. If $\boldsymbol{v}$ is a sample from μ, the probability law of $f_H(\boldsymbol{v})$ is $\mu \circ f_H$.*

Proof. The swap operation f_H on $[\mathbf{x}, \widetilde{\mathbf{x}}]$ swaps coordinates in the following manner: for each $j \in H$, the jth and $(j+d)$th coordinates are swapped. Let $(E, \mathcal{E})$ be a measurable space, where elements of E are $2d$-dimensional vectors, and $\mathcal{E}$ is a σ-algebra on E. Let $(F, \mathcal{F})$ also be a measurable space where each element of F is an element of E but with the jth coordinate swapped with the $(j+d)$th coordinate for each $j \in H$. Similarly, let $\mathcal{F}$ be constructed by applying the

same swap transformations to each element of $\mathcal{E}$. $\mathcal{F}$ is a σ-algebra as swaps are one-to-one transformations, and $\mathcal{E}$ is a σ-algebra.

We first show that f_H is a measurable function with respect to $\mathcal{E}$ and $\mathcal{F}$. This is true by construction of the measurable space $(F, \mathcal{F})$. For every element $B \in \mathcal{F}$, $f_H^{-1}(B) \in \mathcal{E}$. We can now construct a mapping $\mu \circ f_H^{-1}(B)$ for all $B \in \mathcal{F}$. This is the pushforward measure of μ under transformation f_H, and is well defined because f_H is measurable. Using the fact that a swap applied twice is the identity, we get $f_H = f_H^{-1}$. With this, we see that the probability measure on $(F, \mathcal{F})$ is $\mu \circ f_H^{-1} = \mu \circ f_H$. □

Implication of theorem 3.1.1: an example. While theorem 3.1.1 is written generally, it may help to understand its implication for a concrete example. In the continuous case where $q_{\mathbf{z}}$ and $q_{\mathbf{w}}$ are the densities of $\mathbf{z}$ and $\mathbf{w}$ respectively, $q_{\mathbf{w}}$ evaluated at a sample $\boldsymbol{v}$ is simply $q_{\mathbf{z}}$ evaluated at the swap of $\boldsymbol{v}$. To understand why, note that a swap operation on $\mathbf{z}$ is an affine transformation $\mathbf{w} = \mathbf{A}\mathbf{z}$, where $\mathbf{A}$ is a permutation matrix. Using this property, we get:

$$q_{\mathbf{w}}(\mathbf{z}) = \left|\det\left(\frac{\partial \mathbf{A}^{-1}\mathbf{w}}{\partial \mathbf{w}}\right)\right| \cdot q_{\mathbf{z}}(\mathbf{A}^{-1}\mathbf{z}) = q_{\mathbf{z}}(\mathbf{A}^{-1}\mathbf{z}) = q_{\mathbf{z}}(\mathbf{A}\mathbf{z}) = q_{\mathbf{z}}(\mathbf{w}).$$

The first step is achieved by using a change of variables, noting that $\mathbf{A}$ is invertible, and $\mathbf{z} = \mathbf{A}^{-1}\mathbf{w}$. The determinant of the Jacobian here is just the determinant of $\mathbf{A}^{-1}$. $\mathbf{A}^{-1}$ is a permutation matrix whose parity is even, meaning its determinant is 1, and that $\mathbf{A}^{-1} = \mathbf{A}$. I.e. the density of the swapped variables evaluated at $\mathbf{z}$ is equal to the original density evaluated at $\mathbf{w}$.

3.2 Deriving DDLK

A useful consequence of theorem 3.1.1 is that DDLK needs to only model $q_{\mathbf{z}}$, instead of $q_{\mathbf{z}}$ and every possible swap distribution $q_{\mathbf{w}}$. To derive the DDLK algorithm, we first expand eq. (3.2):

$$\mathbb{E}_{q_{\mathbf{z}}(\mathbf{z})}\left[\log \frac{q_{\mathbf{z}}(\mathbf{z})}{q_{\mathbf{w}}(\mathbf{z})}\right] = \mathbb{E}_{q(\mathbf{x})}\mathbb{E}_{q(\widetilde{\mathbf{x}}|\mathbf{x})}\left[\log \frac{q(\mathbf{x})q(\widetilde{\mathbf{x}} \mid \mathbf{x})}{q(\mathbf{u})q(\widetilde{\mathbf{u}} \mid \mathbf{u})}\right], \tag{3.3}$$

where $[\mathbf{u}, \widetilde{\mathbf{u}}] = [\mathbf{x}, \widetilde{\mathbf{x}}]_{\text{swap}(H)}$. DDLK models the RHS by parameterizing $q(\mathbf{x})$ and $q(\widetilde{\mathbf{x}} \mid \mathbf{x})$ with $\hat{q}_{\text{joint}}(\mathbf{x}; \theta)$ and $\hat{q}_{\text{knockoff}}(\widetilde{\mathbf{x}} \mid \mathbf{x}; \phi)$ respectively. The parameters θ and ϕ can be optimized separately in two stages.

Stage 1: Covariate distribution estimation. We model the distribution of $\mathbf{x}$ using $\hat{q}_{\text{joint}}(\mathbf{x}; \theta)$. The parameters of the model θ are learned by maximizing $\mathbb{E}_{\boldsymbol{x} \sim \mathcal{D}_N}\left[\log \hat{q}_{\text{joint}}(\boldsymbol{x}; \theta)\right]$ over a dataset $\mathcal{D}_N := \{\boldsymbol{x}^{(i)}\}_{i=1}^{N}$ of N samples.

Stage 2: Knockoff generation. For any fixed swap H, minimizing the KL divergence between the following distributions ensures the swap property required of knockoffs:

$$\text{KL}(\hat{q}_{\text{joint}}(\mathbf{x}; \theta)\hat{q}_{\text{knockoff}}(\widetilde{\mathbf{x}} \mid \mathbf{x}; \phi) \parallel \hat{q}_{\text{joint}}(\mathbf{u}; \theta)\hat{q}_{\text{knockoff}}(\widetilde{\mathbf{u}} \mid \mathbf{u}; \phi)). \tag{3.4}$$

Fitting the knockoff generator $\hat{q}_{\text{knockoff}}(\widetilde{\mathbf{x}} \mid \mathbf{x}; \phi)$ involves minimizing this KL divergence for all possible swaps H. To make this problem tractable, we use several building blocks that help us (a) sample swaps with the highest values of this KL and (b) prevent $\hat{q}_{\text{knockoff}}$ from memorizing $\mathbf{x}$ to trivially satisfy the swap property.

3.3 Sampling swap subsets

In the previous section we derived an optimization procedure for DDLK, but one problem still remains. The optimization implies the KL-term in eq. (3.4) to be minimized for all possible swap subsets $H \subseteq [d]$. We show next using lemma 3.3.1 that satisfying the swap property for an exponential number of swaps is not necessary. It suffices to satisfy the swap property for a collection of sets where any singleton H can be represented as the symmetric difference of members of the collection.

Lemma 3.3.1. ***Sufficient swaps****: Let $\mathcal{A} = \{A_1, \ldots, A_K\}$ where $A_k \subseteq [d]$. For each $j \in [d]$, if there exists some subset of $\mathcal{A}$ whose symmetric difference is equal to $\{j\}$, then satisfying the swap property for all sets in $\mathcal{A}$ is equivalent to satisfying the swap property for all possible subsets of $[d]$.*

Proof. One approach to check if knockoffs are valid is to verify the swap property for all singleton sets $\{j\} \subset [d]$ [Romano et al. 2018; Jordon et al. 2019]. To check if the swap property eq. (3.1) holds under any $H = \{j_1, \ldots, j_k\}$, it suffices to check if eq. (3.1) holds under each of $\{j_1\}, \ldots, \{j_k\}$.

We can generalize this approach to check the validity of knockoffs under other collections of indices besides singleton sets using the following property. Let $H_1, H_2 \subseteq [d]$ and

$$[\mathbf{x}, \widetilde{\mathbf{x}}] \overset{d}{=} [\mathbf{x}, \widetilde{\mathbf{x}}]_{\text{swap}(H_1)}$$
$$[\mathbf{x}, \widetilde{\mathbf{x}}] \overset{d}{=} [\mathbf{x}, \widetilde{\mathbf{x}}]_{\text{swap}(H_2)}.$$

Then,

$$[\mathbf{x}, \widetilde{\mathbf{x}}]_{\text{swap}(H_1 \Delta H_2)} \overset{d}{=} \left[[\mathbf{x}, \widetilde{\mathbf{x}}]_{\text{swap}(H_1)}\right]_{\text{swap}(H_2)} \overset{d}{=} [\mathbf{x}, \widetilde{\mathbf{x}}]_{\text{swap}(H_2)} \overset{d}{=} [\mathbf{x}, \widetilde{\mathbf{x}}]$$

where $H_1 \Delta H_2$ is the symmetric difference of H_1 and H_2. Swapping the indices in $H_1 \Delta H_2$ is equivalent to swapping the indices in H_1, then the indices in H_2. If $\exists j \in H_1 \wedge j \in H_2$, swapping

j twice will negate the effect of the swap.

We can extend this property to K sets and define sufficient conditions to check if the swap property holds. Let $\{A_k\}_{k=1}^K$ be a sequence of sets where each $A_k \subseteq [d]$. Let

$$A_1^* = A_1$$
$$\forall k \in [K], A_k^* = A_k \Delta A_{k-1}^*$$
$$A_K^* = \{j\}.$$

Checking the swap property eq. (3.1) under a sequence of swaps $\{A_k\}_{k=1}^K$ is equivalent to checking eq. (3.1) under the singleton set $\{j\}$. Therefore, the swap property must also hold under the singleton set $\{j\}$.

If collection of sets of swap indices $\mathcal{A}$ contains a sub-sequence $\{A_m\}_{m=1}^M$ such that their sequential symmetric difference is the singleton $\{j\}$ for each $j \in [d]$, then a set of knockoffs that satisfies the swap property under each $A_k \in \mathcal{A}$, will also satisfy the swap property under each singleton set, which is sufficient to generate valid knockoffs. □

Sampling swaps. Swapping d coordinates can be expensive in high dimensions, so existing methods resort to randomly sampling swaps [Romano et al. 2018; Jordon et al. 2019] during optimization. Rather than sample each coordinate uniformly at random, we propose parameterizing the sampling process for swap indices H so that swaps sampled from this process yields large values of the KL objective in eq. (3.4). We do so because of the following property of swaps.

Lemma 3.3.2. ***Worst case swaps****: Let $q(H;\beta)$ be the worst case swap distribution. That is, the distribution over swap indices that maximizes*

$$\mathbb{E}_{H\sim q(H;\beta)} KL(\hat{q}_{joint}(\mathbf{x};\theta)\hat{q}_{knockoff}(\tilde{\mathbf{x}} \mid \mathbf{x};\phi) \parallel \hat{q}_{joint}(\mathbf{u};\theta)\hat{q}_{knockoff}(\tilde{\mathbf{u}} \mid \mathbf{u};\phi)) \tag{3.5}$$

with respect to β. If eq. (3.5) is minimized with respect to ϕ, knockoffs sampled from $\hat{q}_{knockoff}$ will satisfy the swap property in eq. (3.1) for any swap H in the power set of $[d]$.

Proof. If eq. (3.5) is minimized with respect to ϕ but maximized with respect to β, then for any other distribution $q(H;\beta')$, eq. (3.5) will be lesser. Minimizing eq. (3.5), which is non-negative, with respect to ϕ implies that for any swap H sampled from $q(H;\beta)$ and for any knockoff $\widetilde{\mathbf{x}}$ sampled from $\hat{q}_{\text{knockoff}}$, the swap property will be satisfied. As eq. (3.5) is also maximized with respect to β, swaps H' drawn from all other distributions $q(H';\beta')$ will only result in lower values of eq. (3.5). Therefore, the joint distribution $[\mathbf{x}, \widetilde{\mathbf{x}}]$ will be invariant under any swap H' in the power set of $[d]$. □

Randomly sampling swaps can be thought of as sampling from d Bernoulli random variables $\{\mathbf{b}_j\}_{j=1}^d$ with parameters $\beta = \{\beta_j\}_{j=1}^d$ respectively, where each $\mathbf{b}_j$ indicates whether the jth coordinate is to be swapped. A set of indices H can be generated by letting $H = \{j : \mathbf{b}_j = 1\}$. To learn a sampling process that helps maximize eq. (3.5), we optimize the values of β. However, since score function gradients for the parameters of Bernoulli random variables can have high variance, DDLK uses a continuous relaxation instead. For each coordinate $j \in d$, DDLK learns the parameters for a Gumbel-Softmax [Jang et al. 2016; Maddison et al. 2016] distribution $\hat{q}_{\text{gumbel}}(\beta_j)$.

3.4 Entropy regularization

Minimizing the the KL objective in eq. (3.5) over the worst case swap distribution will generate knockoffs that satisfy the swap property eq. (3.1). However, a potential solution in the optimization of $\hat{q}_{\text{knockoff}}(\widetilde{\mathbf{x}} \mid \mathbf{x})$ is to memorize the covariates $\mathbf{x}$, which reduces the power to select important variables.

To solve this problem, DDLK introduces a regularizer based on the conditional entropy, to push $\widetilde{\mathbf{x}}$ to not be a copy of $\mathbf{x}$. This regularizer takes the form $-\lambda\mathbb{E}[-\log \hat{q}_{\text{knockoff}}(\widetilde{\mathbf{x}} \mid \mathbf{x};\phi)]$, where λ is a hyperparameter.

Algorithm 3.1 DDLK

Input: $\mathcal{D}_N := \{\boldsymbol{x}^{(i)}\}_{i=1}^N$, dataset of covariates; λ, regularization parameter; α_ϕ, learning rate for $\hat{q}_{\text{knockoff}}$; α_β, learning rate for $\hat{q}_{\text{gumbel}}$

Output: θ, parameter for $\hat{q}_{\text{joint}}$, $\hat{q}_{\text{knockoff}}$, parameter for $\hat{q}_{\text{knockoff}}$

$\theta = \arg\max_\theta \frac{1}{N}\sum_{i=1}^N \log \hat{q}_{\text{joint}}(\boldsymbol{x}^{(i)}; \theta)$

while $\hat{q}_{knockoff}$ *not converged* **do**

  Sample $\{\widetilde{\boldsymbol{x}}^{(i)}\}_{i=1}^N$, where $\widetilde{\boldsymbol{x}}^{(i)} \sim \hat{q}_{\text{knockoff}}(\widetilde{\mathbf{x}} \mid \boldsymbol{x}^{(i)}; \phi)$

  Sample swap $H \sim \hat{q}_{\text{gumbel}}(\beta)$

  Create $\{(\boldsymbol{u}^{(i)}, \widetilde{\boldsymbol{u}}^{(i)})\}_{i=1}^N$, where $[\boldsymbol{u}^{(i)}, \widetilde{\boldsymbol{u}}^{(i)}] = [\boldsymbol{x}^{(i)}, \widetilde{\boldsymbol{x}}^{(i)}]_{\text{swap}(H)}$

  Let $\mathcal{A}(\phi) = \frac{1}{N}\sum_{i=1}^N \log \hat{q}_{\text{joint}}(\boldsymbol{x}^{(i)}; \theta) + (1+\lambda)\log \hat{q}_{\text{knockoff}}(\widetilde{\boldsymbol{x}}^{(i)} \mid \boldsymbol{x}^{(i)}; \phi)$

  Let $\mathcal{B}(\phi, \beta) = \frac{1}{N}\sum_{i=1}^N \log \hat{q}_{\text{joint}}(\boldsymbol{u}^{(i)}; \theta) + \log \hat{q}_{\text{knockoff}}(\widetilde{\boldsymbol{u}}^{(i)} \mid \boldsymbol{u}^{(i)}; \phi)$

  $\phi \leftarrow \phi - \alpha_\phi \nabla_\phi (\mathcal{A}(\phi) - \mathcal{B}(\phi, \beta))$

  $\beta \leftarrow \beta + \alpha_\beta \nabla_\beta (\mathcal{A}(\phi) - \mathcal{B}(\phi, \beta))$

end

return θ, ϕ, β

Including the regularizer on conditional entropy, and Gumbel-Softmax sampling of swap indices, the final optimization objective for DDLK is as follows, where $[\boldsymbol{u}, \widetilde{\boldsymbol{u}}] = [\boldsymbol{x}, \widetilde{\boldsymbol{x}}]_{\text{swap}(H)}$:

$$\min_\phi \max_\beta \mathbb{E}_{H\sim\hat{q}_{\text{gumbel}}(\beta)} \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_N} \mathbb{E}_{\widetilde{\boldsymbol{x}}\sim\hat{q}_{\text{knockoff}}(\widetilde{\mathbf{x}}|\boldsymbol{x};\phi)} \log \frac{\hat{q}_{\text{joint}}(\boldsymbol{x}; \theta)\hat{q}_{\text{knockoff}}(\widetilde{\boldsymbol{x}} \mid \boldsymbol{x}; \phi)^{1+\lambda}}{\hat{q}_{\text{joint}}(\boldsymbol{u}; \theta)\hat{q}_{\text{knockoff}}(\widetilde{\boldsymbol{u}} \mid \boldsymbol{u}; \phi)}. \tag{3.6}$$

We show the full DDLK algorithm in algorithm 3.1. DDLK fits $\hat{q}_{\text{joint}}$ by maximizing the likelihood of the data. It then fits $\hat{q}_{\text{knockoff}}$ by optimizing eq. (3.6) with noisy gradients. To do this, DDLK first samples knockoffs conditioned on the covariates and a set of swap coordinates, then computes Monte-Carlo gradients of the DDLK objective in eq. (3.6) with respect to parameters ϕ and β. In practice DDLK can use stochastic gradient estimates like the score function or reparameterization gradients for this step. The $\hat{q}_{\text{joint}}$ and $\hat{q}_{\text{knockoff}}$ models can be implemented with flexible models like MADE [Germain et al. 2015] or mixture density networks [Bishop 1994a].

3.5 Related work

Existing knockoff generation methods can be broadly classified as either model-specific or flexible. Model-specific methods such as hidden markov models (HMMs) [Sesia et al. 2017] or Auto-Encoding Knockoffs [Liu and Zheng 2018] make assumptions about the covariate distribution, which can be problematic if the data does not satisfy these assumptions. HMMs assume the joint distribution of the covariates can be factorized into a markov chain. Auto-Encoding Knockoffs use variational auto-encoders (VAEs) to model $\mathbf{x}$ and sample knockoffs $\widetilde{\mathbf{x}}$. VAEs assume $\mathbf{x}$ lies near a low dimensional manifold, whose dimension is controlled by a latent variable. Covariates that violate this low-dimensional assumption can be better modeled by increasing the dimension of the latent variable, but risk retaining more information about $\mathbf{x}$, which can reduce the power to select important variables.

Flexible methods for generating knockoffs such as KnockoffGAN [Jordon et al. 2019] or Deep Knockoffs [Romano et al. 2018] focus on likelihood-free generative models. KnockoffGAN uses generative adversarial network (GAN)-based generative models, which can be difficult to estimate [Mescheder et al. 2018] and sensitive to hyperparameters [Salimans et al. 2016; Gulrajani et al. 2017; Mescheder et al. 2017]. Deep Knockoffs employ maximum mean discrepancys (MMDs), the effectiveness of which often depends on the choice of a kernel which can involve selecting a bandwidth hyperparameter. Ramdas et al. [2015] show that in several cases, across many choices of bandwidth, MMD approaches 0 as dimensionality increases while KL divergence remains non-zero, suggesting MMDs may not reliably generate high-dimensional knockoffs. Deep Knockoffs also prevent the knockoff generator from memorizing the covariates by explicitly controlling the correlation between the knockoffs and covariates. This is specific to second order moments, and may ignore higher order ones present in the data.

3.6 Experiments

We study the performance of DDLK on several synthetic, semi-synthetic, and real-world datasets. We compare DDLK with several non-Gaussian knockoff generation methods: Auto-Encoding Knockoffs (AEK) [Liu and Zheng 2018], KnockoffGAN [Jordon et al. 2019], and Deep Knockoffs [Romano et al. 2018].

3.6.1 Baseline method implementation

For all comparison methods, we downloaded the publicly available implementations of the code (if available) and used the appropriate configurations and hyperparameters recommended by the authors. For Deep Knockoffs and KnockoffGAN, we use code from each respective repository:

```
https://github.com/msesia/deepknockoffs
https://bitbucket.org/mvdschaar/mlforhealthlabpub/
```

and use the recommended hyperparameter settings.

At the time of writing this thesis, there was no publicly available implementation for Auto-Encoding Knockoffs. We implemented Auto-Encoding Knockoffs with a VAE with a gaussian posterior

$$q(\mathbf{z} \mid \mathbf{x}) \approx \mathcal{N}(\mathbf{z}; \mu_{\mathbf{z}}(\mathbf{x}), \sigma_{\mathbf{z}}(\mathbf{x}))$$

and likelihood

$$p(\mathbf{x} \mid \mathbf{z}) \approx \mathcal{N}(\mathbf{x}; \mu_{\mathbf{x}}(\mathbf{z}), \sigma_{\mathbf{x}}(\mathbf{z})).$$

Each of $\mu_{\mathbf{z}}, \sigma_{\mathbf{z}}, \mu_{\mathbf{x}}.\sigma_{\mathbf{x}}$ is a 2-layer neural network with 400 units in the first hidden layer, 500 units in the second, and ReLU activations. The outputs of networks $\sigma_{\mathbf{z}}, \sigma_{\mathbf{x}}$ are exponentiated to ensure variances are non-negative. The outputs of network $\mu_{\mathbf{z}}$ and $\sigma_{\mathbf{z}}$ are of dimension $d_{\mathbf{z}}$, and the outputs of $\mu_{\mathbf{x}}$ and $\sigma_{\mathbf{x}}$ are of dimension d, the covariate dimension. For each dataset, we choose the dimension $d_{\mathbf{z}}$ of latent variable $\mathbf{z}$ that maximizes the estimate of the ELBO on a validation dataset. In our experiments, we search for $d_{\mathbf{z}}$ over the set $\{d_{\mathbf{z}} : 10 \leq d_{\mathbf{z}} \leq 200, d_{\mathbf{z}} \mod 10 = 0\}$. For each dataset, we use the following $d_{\mathbf{z}}$:

1. `gaussian`: 20
2. `mixture`: 140
3. `gene`: 30
4. `covid-19`: 60.

The neural networks are trained using Adam [Kingma and Ba 2014], with a learning rate of 1×10^{-4} for a maximum of 150 epochs. To avoid very large gradients, we standardize the data using the mean and standard deviation of the training set. To generate knockoffs $\widetilde{\mathbf{x}}$, we use the same approach prescribed by Liu and Zheng [2018]. We first sample the latent variable $\mathbf{z}$ conditioned on the covariates using the posterior distribution:

$$\mathbf{z} \sim \mathcal{N}\left(\mathbf{z}; \mu_{\mathbf{z}}(\mathbf{x}), \sigma_{\mathbf{z}}(\mathbf{x})\right).$$

This sample of $\mathbf{z}$ is then used to sample a knockoff $\widetilde{\mathbf{x}}$ using the likelihood distribution:

$$\widetilde{\mathbf{x}} \sim \mathcal{N}\left(\mathbf{x}; \mu_{\mathbf{x}}(\mathbf{z}), \sigma_{\mathbf{x}}(\mathbf{z})\right).$$

Since these $\widetilde{\mathbf{x}}$ are standardized, we re-scale them by the the training mean and standard deviation.

3.6.2 Experimental setup

Each experiment involves three stages. First, we fit a knockoff generator using a dataset of covariates $\{\boldsymbol{x}^{(i)}\}_{i=1}^{N}$. Next, we fit a response model $\hat{q}_{\text{response}}(\mathbf{y} \mid \mathbf{x}; \gamma)$, and use its performance on a held-out set to create a knockoff statistic w_j for each feature $\mathbf{x}_j$. Finally, we apply a knockoff

filter to the statistics $\{w_j\}_{j=1}^d$ to select features at a nominal FDR level p, and measure the ability of each knockoff method to select relevant features while maintaining FDR at p. For the synthetic and semi-synthetic tasks, we repeat these three stages 30 times to obtain interval estimates of each performance metric.

3.6.2.1 Fitting a knockoff generator

In our experiments, we assume $\mathbf{x}$ to be real-valued with continuous support and decompose the models $\hat{q}_{\text{joint}}$ and $\hat{q}_{\text{knockoff}}$ via the chain rule:

$$q(\mathbf{x} \mid \cdot) = q(\mathbf{x}_1 \mid \cdot) \prod_{j=2}^{d} q(\mathbf{x}_j \mid \cdot, \mathbf{x}_1, \cdots, \mathbf{x}_{j-1}).$$

We model each conditional $q(\mathbf{x}_j \mid \cdot)$ using mixture density networks [Bishop 1994a] which take the form

$$q(\mathbf{x}_j \mid \cdot) = \sum_{k=1}^{K} \pi_k(\cdot; \psi_k) \mathcal{N}(\mu_k(\cdot; \eta_k), \sigma_k^2(\cdot; \omega_k))$$

where functions $\{\pi_k\}_{k=1}^K$, $\{\mu_k\}_{k=1}^K$, and $\{\sigma_k\}_{k=1}^K$ characterize a univariate gaussian mixture. These parameters of these functions are $[\psi_1, \ldots, \psi_K, \nu_1, \ldots, \nu_K, \omega_1, \ldots, \omega_K]$.

Fitting $\hat{q}_{\text{joint}}$. Let θ, the parameters of $\hat{q}_{\text{joint}}$ contain parameters for every conditional $q(\mathbf{x}_j \mid \mathbf{x}_1, \ldots, \mathbf{x}_{j-1})$. The optimization of θ is straightforward:

$$\theta = \arg\max_{\theta} \mathcal{L}(\theta) = \arg\max_{\theta} \frac{1}{N} \sum_{i=1}^{N} \log \hat{q}_{\text{joint}}(\boldsymbol{x}^{(i)}; \theta)$$

only requires taking the derivative of $\mathcal{L}(\theta)$.

FITTING $\hat{q}_{\text{KNOCKOFF}}$. Let ϕ, the parameters of $\hat{q}_{\text{knockoff}}$ contain parameters for every conditional $q(\widetilde{\mathbf{x}}_j \mid \mathbf{x}_1, \ldots, \mathbf{x}_d, \widetilde{\mathbf{x}}_1, \ldots, \widetilde{\mathbf{x}}_{j-1})$. Recall the loss function $\mathcal{L}(\phi, \beta)$

$$\mathcal{L}(\phi, \beta) = \mathbb{E}_{H \sim \hat{q}_{\text{gumbel}}(\beta)} \mathbb{E}_{\boldsymbol{x} \sim \mathcal{D}_N} \mathbb{E}_{\widetilde{\boldsymbol{x}} \sim \hat{q}_{\text{knockoff}}(\widetilde{\mathbf{x}} \mid \boldsymbol{x}; \phi)} \log \frac{\hat{q}_{\text{joint}}(\boldsymbol{x}; \theta) \hat{q}_{\text{knockoff}}(\widetilde{\boldsymbol{x}} \mid \boldsymbol{x}; \phi)^{1+\lambda}}{\hat{q}_{\text{joint}}(\boldsymbol{u}; \theta) \hat{q}_{\text{knockoff}}(\widetilde{\boldsymbol{u}} \mid \boldsymbol{u}; \phi)}.$$

The optimization of ϕ requires $\nabla_\phi \mathcal{L}(\phi, \beta)$, which involves the derivative of an expectation with respect to to $\hat{q}_{\text{knockoff}}(\widetilde{\mathbf{x}} \mid \mathbf{x}; \phi)$. We use implicit reparameterization [Figurnov et al. 2018]. The advantage of implicit reparameterization over explicit reparameterization [Kingma et al. 2015] is that an inverse standardization function S_ϕ^{-1} – which transforms random noise into samples from a distribution parameterized by ϕ – is not needed. Using implicit reparameterization, gradients of some objective $\mathbb{E}_{q(\mathbf{z};\phi)}[f(\mathbf{z})]$ can be rewritten as

$$\begin{aligned}
\mathbb{E}_{q(\mathbf{z};\phi)}[\nabla_\phi f(\mathbf{z})] &= \mathbb{E}_{q(\mathbf{z};\phi)}[\nabla_\mathbf{z} f(\mathbf{z}) \nabla_\phi \mathbf{z}] \\
&= \mathbb{E}_{q(\mathbf{z};\phi)}[-\nabla_\mathbf{z} f(\mathbf{z}) (\nabla_\mathbf{z} S_\phi(\mathbf{z}))^{-1} \nabla_\phi S_\phi(\mathbf{z})].
\end{aligned}$$

We use this useful property to reparameterize gaussian mixture models. Let $q(\mathbf{z}; \phi)$ be a gaussian mixture model:

$$q(\mathbf{z}; \phi) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{z}; \mu_k, \sigma_k^2)$$

where $\phi = [\pi_1, \ldots, \pi_K, \mu_1, \ldots, \mu_K, \sigma_1, \ldots, \sigma_K]$. Let the standardization function S_ϕ be the CDF of $q(\mathbf{z}; \phi)$:

$$S_\phi(\mathbf{z}) = \sum_{k=1}^{K} \pi_k \Phi\left(\frac{\mathbf{z} - \mu_k}{\sigma_k}\right)$$

where Φ is the standard normal gaussian CDF. We use this to compute the gradient of $\mathbf{z}$ with

respect to each parameter:

$$\nabla_{\pi_k} \mathbf{z} = -\frac{\Phi(\frac{\mathbf{z}-\mu_k}{\sigma_k})}{q(\mathbf{z};\phi)}$$
$$\nabla_{\mu_k} \mathbf{z} = \frac{\pi_k \cdot \mathcal{N}(\mathbf{z};\mu_k,\sigma_k^2)}{q(\mathbf{z};\phi)}$$
$$\nabla_{\sigma_k} \mathbf{z} = \frac{\pi_k \cdot \left(\frac{\mathbf{z}-\mu_k}{\sigma_k}\right) \cdot \mathcal{N}(\mathbf{z};\mu_k,\sigma_k^2)}{q(\mathbf{z};\phi)}.$$

Putting it all together, we use the implicit reparameterization trick to implement each conditional distribution in $\hat{q}_{\text{joint}}$ and $\hat{q}_{\text{knockoff}}$.

3.6.2.2 Hyperparameter settings

Each mixture density network is a 3-layer neural network with 50 parameters in each layer and a residual skip connection from the input to the last layer. Each network outputs the parameters for a univariate gaussian mixture with 5 components. We initialize the network such that the modes are evenly spaced within the support of training data.

Using $\hat{q}_{\text{gumbel}}$, we sample binary swap matrices of the same dimension as the data. As we require discrete samples from the Gumbel-Softmax distribution, we implement a straight-through estimator [Jang et al. 2016]. The straight-through estimator facilitates sampling discrete indices, but uses a continuous approximation during backpropagation.

The $\hat{q}_{\text{joint}}$ model is optimized using Adam [Kingma and Ba 2014], with a learning rate of 5×10^{-4} for a maximum of 50 epochs. The $\hat{q}_{\text{knockoff}}$ model is optimized using Adam, with a learning rate of 1×10^{-3} for ϕ and 1×10^{-2} for β for a maximum of 250 epochs. We also implement early stopping using validation loss using the PyTorch Lightning framework [Falcon 2019]. Our code can be found online by installing:

```
pip install -i https://test.pypi.org/simple/ddlk==0.2
```

Across each benchmark involving DDLK, we vary only the λ entropy regularization parameter based on the amount of dependence among covariates. The number of parameters, learning rate, and all other hyperparameters are kept constant. To sample swaps H, we sample using a straight-through Gumbel-Softmax estimator [Jang et al. 2016]. This allows us to sample binary values for each swap, but use gradients of a continuous approximation during optimization.

3.6.2.3 Robust model-based knockoff statistics

The goal of any knockoff method is to help compute test statistics for a conditional independence test. We employ a variant of HRTs [Tansey et al. 2018a] to compute test statistics w_j for each feature $\mathbf{x}_j$. We split dataset $\mathcal{D}_N := \{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})\}_{i=1}^N$ into train and test sets $\mathcal{D}_N^{(\text{tr})}$, and $\mathcal{D}_N^{(\text{te})}$ respectively, then sample knockoff datasets $\widetilde{\mathcal{D}}_N^{(\text{tr})}$ and $\widetilde{\mathcal{D}}_N^{(\text{te})}$ conditioned on each. Next, a model $\hat{q}_{\text{response}}$ is fit with $\mathcal{D}_N^{(\text{tr})}$.

To compute knockoff statistics with $\hat{q}_{\text{response}}$, we use a measure of performance $\mathcal{W}(\hat{q}_{\text{response}}, \mathcal{D}_N^{(\text{te})})$ on the test set. For real-valued $\mathbf{y}$, $\mathcal{W}$ is the mean squared-error, and for categorical $\mathbf{y}$, $\mathcal{W}$ is expected log-probability of $\mathbf{y} \mid \mathbf{x}$. A knockoff statistic $w_j := \mathcal{W}(\hat{q}_{\text{response}}, \mathcal{D}_N^{(\text{te})}) - \mathcal{W}(\hat{q}_{\text{response}}, \widetilde{\mathcal{D}}_{j,N}^{(\text{te})})$ is recorded for each feature $\mathbf{x}_j$, where $\widetilde{\mathcal{D}}_{j,N}^{(\text{te})}$ is $\mathcal{D}_N^{(\text{te})}$ but with the jth feature swapped with $\widetilde{\mathcal{D}}_N^{(\text{te})}$.

In practice, we use use flexible models like neural networks or boosted trees for $\hat{q}_{\text{response}}$. While the model-based statistic above will satisfy the properties detailed in section 2.1.2 and control FDR, its ability to do so is hindered by imperfect knockoffs. In such cases, we observe that knockoff statistics for null features are centered around some $\zeta > 0$, violating a condition required for empirical FDR control. This happens because if the covariates and knockoffs are not equal in distribution, models trained on the covariates will fit the covariates better than the knockoffs and inflate the value of test statistic w_j. This can lead to an increase in the false discovery rate as conditionally independent features may be selected if their statistic is larger than the selection threshold. To combat this, we propose a mixture statistic that trades off power for FDR-control.

The mixture statistic involves fitting a $\hat{q}_{\text{response}}$ model for each feature $\mathbf{x}_j$ using an equal mix-

ture of data in $\mathcal{D}_N^{(\text{tr})}$ and $\widetilde{\mathcal{D}}_{j,N}^{(\text{tr})}$, then computing $\mathcal{W}$ as above. Such a $\hat{q}_{\text{response}}$ achieves lower performance on $\mathcal{D}_N^{(\text{te})}$, but higher performance on $\widetilde{\mathcal{D}}_{j,N}^{(\text{te})}$, yielding values of w_j with modes closer to 0, enabling finite sample FDR control. However, this FDR-control comes at the cost of power as the method's ability to identify conditionally dependent features is reduced. We revisit this issue from a theoretical perspective in chapter 4.

3.6.3 Synthetic benchmarks

Our tests on synthetic data seek to highlight differences in power and FDR between each knockoff generation method. Each dataset in this section consists of $N = 2000$ samples, 100 features, 20 of which are used to generate the response $\mathbf{y}$. Testing the global null (0 important features) can also help understand the performance of a knockoff method, as FDR control equates to control of the family-wise error rate: a stricter notion of false discovery control. However, we found our results in global null experiments to be no more instructive about the differences between each method than with 20 important features. We split the data into a training set (70%) to fit each knockoff method, a validation set (15%) used to tune the hyperparameters of each method, and a test set (15%) for evaluating knockoff statistics.

[`gaussian`]: We first replicate the multivariate normal benchmark of Romano et al. [2018]. We sample $\mathbf{x} \sim \mathcal{N}(0, \Sigma)$, where Σ is a d-dimensional covariance matrix whose entries $\Sigma_{i,j} = \rho^{|i-j|}$. This autoregressive Gaussian data exhibits strong correlations between adjacent features, and lower correlations between features that are further apart. We generate $\mathbf{y} \mid \mathbf{x} \sim \mathcal{N}(\langle \mathbf{x}, \alpha \rangle, 1)$, where coefficients for the important features are drawn as $\alpha_j \sim \frac{100}{\sqrt{N}} \cdot \text{Rademacher}(0.5)$. In our experiments, we set $\rho = 0.6$. We let the DDLK entropy regularization parameter $\lambda = 0.1$. Our model $\hat{q}_{\text{response}}$ for $\mathbf{y} \mid \mathbf{x}$ is a 1-layer neural network with 200 parameters.

[`mixture`]: To compare each method on its ability to generate non-Gaussian knockoffs, we use a mixture of autoregressive Gaussians. This is a more challenging benchmark as each covariate is multi-modal, and highly correlated with others. We sample $\mathbf{x} \sim \sum_{k=1}^{K} \pi_k \mathcal{N}(\mu_k, \Sigma_k)$,

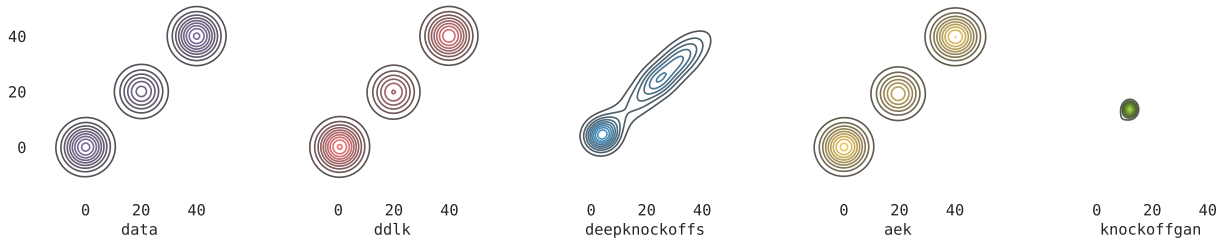


Figure 3.1: DDLK closely models the modes, covariances, and mixture proportions of the `mixture` dataset. Auto-Encoding Knockoffs also capture every mode, but does so by overfitting to the covariates. Deep Knockoffs are able to match the first two moments, but fail to capture every mode. KnockoffGAN suffers from mode collapse and fails to capture every mode.

where each Σ_k is a d-dimensional covariance matrix whose (i, j)th entry is $\rho_k^{|i-j|}$. We generate $\mathbf{y} \mid \mathbf{x} \sim \mathcal{N}(\langle \mathbf{x}, \alpha \rangle, 1)$, where coefficients for the important features are drawn as $\alpha_j \sim \frac{100}{\sqrt{N}} \cdot \text{Rademacher}(0.5)$. In our experiments, we set $K = 3$, and $(\rho_1, \rho_2, \rho_3) = (0.6, 0.4, 0.2)$. Cluster centers are set to $(\mu_1, \mu_2, \mu_3) = (0, 20, 40)$, and mixture proportions are set to $(\pi_1, \pi_2, \pi_3) = (0.4, 0.2, 0.4)$. We let the DDLK entropy regularization parameter $\lambda = 0.001$. Figure 3.1 visualizes two randomly selected dimensions of this data.

RESULTS. Figure 3.2 compares the average FDP and power (percentage of important features selected) of each knockoff generating method. The average FDP is an empirical estimate of the FDR. In the case of the `gaussian` dataset, all methods control FDR at or below the the nominal level, while achieving 100% power to select important features. The main difference between each method is in the calibration of null statistics. Recall that a knockoff filter assumes a null statistic to be positive or negative with equal probability, and features with negative statistics below a threshold are used to control the number of false discoveries when features with positive statistics above the same threshold are selected. DDLK produces the most well calibrated null statistics as evidenced by the closeness of its FDP curve to the dotted diagonal line.

Figure 3.2 also demonstrates the effectiveness of DDLK in modeling non-Gaussian covariates. In the case of the `mixture` dataset, DDLK achieves significantly higher power than the baseline methods, while controlling the FDR at nominal levels. To understand why this may be the case, we

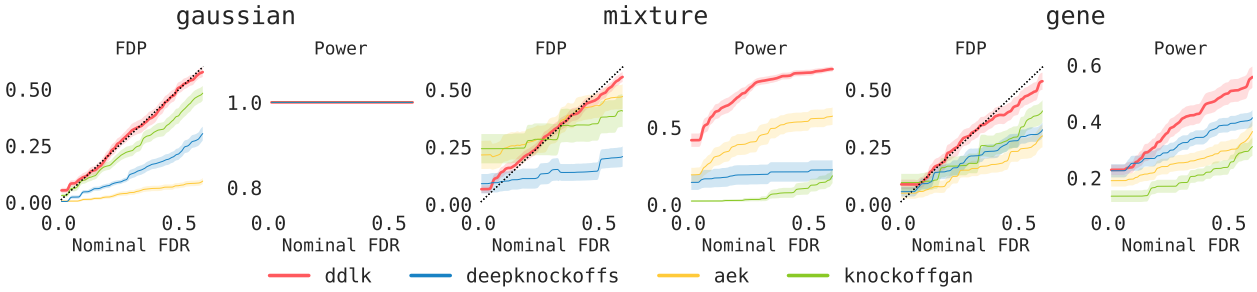


Figure 3.2: DDLK controls FDR at the nominal rate and achieves highest power on a variety of benchmarks. For each benchmark, we show the FDP and power of each knockoff method.

plot the joint distribution of two randomly selected features in fig. 3.1. DDLK and Auto-Encoding Knockoffs both seem to capture all three modes in the data. However, Auto-Encoding Knockoffs tend to produce knockoffs that are very similar to the original features, and yield lower power when selecting variables, shown in fig. 3.2. Deep Knockoffs manage to capture the first two moments of the data – likely due to an explicit second-order term in the objective function – but tend to over-smooth and fail to properly estimate the knockoff distribution. KnockoffGAN suffers from mode collapse, and fails to capture even the first two moments of the data. This yields knockoffs that not only have low power, but also fail to control FDR at nominal levels.

ROBUSTNESS OF DDLK TO ENTROPY REGULARIZATION. To provide guidance on how to set the entropy regularization parameter, we explore the effect of λ on both FDR control and power. Intuitively, lower values of λ will yield solutions of $\hat{q}_{\text{knockoff}}$ that may satisfy eq. (3.1) and control FDR well, but may also memorize the covariates and yield low power. Higher values of λ may help improve power, but at the cost of FDR control. In this experiment, we again use the `gaussian` dataset, but vary λ and the correlation parameter ρ. Figure 3.3 highlights the performance of DDLK over various settings of λ and ρ. We show a heatmap where each cell represents the RMSE between the nominal FDR and mean FDP curves over 30 simulations. In each of these settings DDLK achieves a power of 1, so we only visualize FDP. We observe that the FDP of DDLK is very close to its expected value for most settings where $\lambda \leq 0.1$. This is true over a wide range of ρ

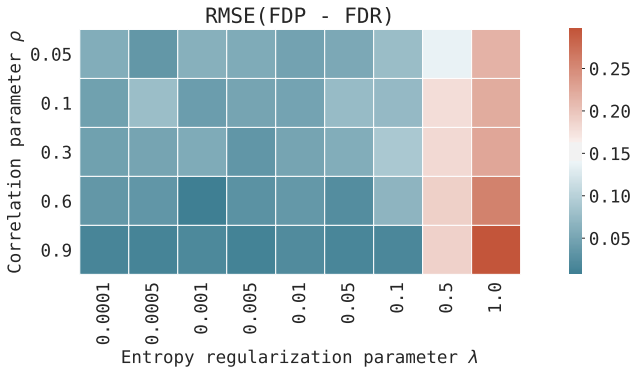


Figure 3.3: DDLK is robust to choices of entropy regularization parameter λ. For most choices of $\lambda \leq 0.1$, DDLK achieves FDP very close to that of the nominal FDR rate. This figure shows the RMSE between the expected and actual FDP curves.

explored, demonstrating that DDLK is not very sensitive to the choice of this hyperparameter. We also notice that data with weaker correlations see a smaller increase in FDP with larger values of λ. In general, checking the FDP on synthetic responses generated conditional on real covariates can aid in selecting λ.

3.6.4 Semi-synthetic benchmark

[gene]: In order to evaluate the FDR and power of each knockoff method using covariates found in a genomics context, we create a semi-synthetic dataset. We use RNA expression data of 963 cancer cell lines from the Genomics of Drug Sensitivity in Cancer study [Yang et al. 2012]. Each cell line has expression levels for 20K genes, of which we sample 100 such that every feature is highly correlated with at least one other feature. We create 30 independent replications of this experiment by repeating the following process. We first sample a gene $\mathbf{x}_1$ uniformly at random, adding it to the set $\mathcal{X}$. For $\mathbf{x}_j$, $j > 1$, we sample $\mathbf{x}_k$ uniformly at random from $\mathcal{X}$ and compute the set of 50 genes not in $\mathcal{X}$ with the highest correlation with $\mathbf{x}_k$. From this set of 50, we uniformly sample a gene $\mathbf{x}_j$ and add it to the feature set. We repeat this process for $j = 2, \ldots, 100$, yielding 100 genes in total.

We generate $\mathbf{y} \mid \mathbf{x}$ using a nonlinear response function adapted from a study on feature

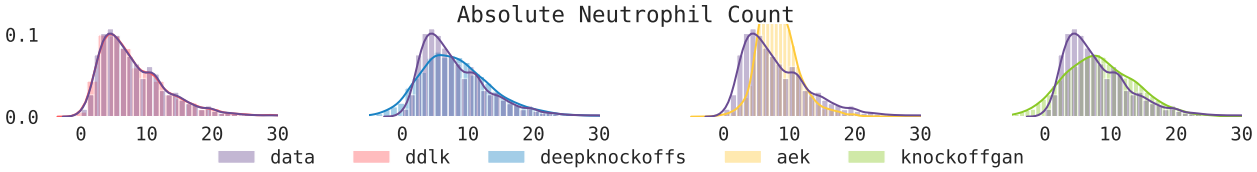


Figure 3.4: DDLK learns the marginals of COVID-19 data better than competing baselines. We plot the marginal distribution of a feature in a COVID-19 dataset, and the corresponding marginal of samples from each knockoff method.

selection in neural network models of gene-drug interactions [Liang et al. 2018]. The response consists of two first-order terms, a second-order term, and an additional nonlinearity in the form of a $\tanh$:

$$k \in [m/4]$$

$$\varphi_k^{(1)}, \varphi_k^{(2)} \sim \mathcal{N}(1, 1)$$

$$\varphi_k^{(3)}, \varphi_k^{(4)}, \varphi_k^{(5)}, \varphi_k^{(6)} \sim \mathcal{N}(2, 1)$$

$$\mathbf{y} \mid \mathbf{x} = \epsilon + \sum_{k=1}^{m/4} \varphi_k^{(1)} \mathbf{x}_{4k-3} + \varphi_k^{(3)} \mathbf{x}_{4k-2} + \varphi_k^{(4)} \mathbf{x}_{4k-3} \mathbf{x}_{4k-2} + \varphi_k^{(5)} \tanh(\varphi_k^{(2)} \mathbf{x}_{4k-1} + \varphi_k^{(6)} \mathbf{x}_{4k})$$

where m is the number of important features. In our experiments, we set $m = 20$. This means that the first 20 features are important, while the remaining 80 are unimportant. We let DDLK entropy regularization parameter $\lambda = 0.001$.

Results. Figure 3.2 (rightmost) highlights the empirical FDP and power of each knockoff generating method in the context of gene. All methods control the FDR below the nominal level, but the average FDP of DDLK at FDR thresholds below 0.3 is closer to its expected value. This range of thresholds is especially important as nominal levels of FDR below 0.3 are most used in practice. In this range, DDLK achieves power on par with Deep Knockoffs at levels below 0.1, and higher power everywhere else. Auto-Encoding Knockoffs and KnockoffGAN achieve noticeably lower power across all thresholds. Deep Knockoffs perform well here likely due to a lack of strong third

Feature	**DDLK**	Deep Knockoffs	AEK	KnockoffGAN	Validated
Eosinophils count	✓	✗	✗	✗	✓
Eosinophils percent	✓	✓	✗	✗	✗
Blood urea nitrogen	✓	✗	✗	✗	✓
Ferritin	✓	✗	✗	✗	✗
O2 Saturation	✓	✗	✗	✗	✓
Heart rate	✓	✗	✗	✗	✓
Respiratory rate	✓	✓	✗	✗	✓
O2 Rate	✓	✓	✓	✓	✓
On room air	✓	✓	✓	✓	✓
High O2 support	✓	✓	✓	✓	✓
Age	✗	✗	✓	✓	✗

Table 3.1: DDLK selects 10/37 features, 8 of which were found to be meaningful by doctors at a large metropolitan hospital. Here we show the union of `covid-19` features selected by each knockoff method at a nominal FDR of 0.2. Deep Knockoffs, Auto-Encoding Knockoffs, and KnockoffGAN exhibit lower power to select important features.

or higher moments of dependence between features. We attribute the success of DDLK and Deep Knockoffs to their ability to model highly correlated data.

3.6.5 COVID-19 ADVERSE EVENTS

[`covid-19`]: The widespread impact of COVID-19 has led to the deployment of machine learning models to guide triage. Data for COVID-19 is messy because of both the large volume of patients and the changing practice for patient care. Establishing trust in models for COVID-19 involves vetting the training data to ensure it does not contain artifacts that models can exploit. Conditional independence tests help achieve this goal in two ways: (a) they highlight which features are most important to the response, and (b) they prune the feature set for a deployed model, reducing the risk of overfitting to processes in a particular hospital. We apply each knockoff method to a large dataset from one of the epicenters of COVID-19 to understand the features most predictive of adverse events.

We use electronic health record data on COVID-positive patients from a large metropolitan health network. Our covariates include demographics, vitals, and lab test results from every com-

plete blood count (CBC) taken for each patient. The response $\mathbf{y} \mid \mathbf{x}$ is a binary label indicating whether or not a patient had an adverse event (intubation, mortality, ICU transfer, hospice discharge, emergency department representation, O2 support in excess of nasal cannula at 6 L/min) within 96 hours of their CBC. There are 17K samples of 37 covariates in the training dataset, 5K in a validation set, and 6K in a held-out test set. We let the DDLK entropy regularization parameter $\lambda = 0.1$. In this experiment, we use gradient boosted regression trees [Friedman 2002; Ke et al. 2017] as our $\hat{q}_{\text{response}}(\mathbf{y} \mid \mathbf{x}; \gamma)$ model, and expected log-likelihood as a knockoff statistic. We also standardize the data in the case of Deep Knockoffs since MMDs that use the radial basis function (RBF) kernel with a single bandwidth parameter work better when features are on the same scale.

Results. At the time of writing this thesis COVID-19 is a recently identified disease, and there is no ground truth set of important features for this dataset. We therefore use each knockoff method to help discover a set of features at a nominal FDR threshold of 0.2, and validate each feature by manual review with doctors at a large metropolitan hospital. Table 3.1 shows a list of features returned by each knockoff method, and indicates whether or not a team of doctors thought the feature should have clinical relevance.

We note that DDLK achieves highest power to select features, identifying 10 features, compared to 5 by DeepKnockoffs, and 4 each by Auto-Encoding Knockoffs and KnockoffGAN. To understand why, we visualize the marginal distributions of each covariate in, and the respective marginal distribution of samples from each knockoff method, in fig. 3.4.

We notice two main differences between DDLK and the baselines. First, DDLK is able to fit asymmetric distributions better than the baselines. Second, despite the fact that the implementation of DDLK using mixture density networks is misspecified for discrete variables, DDLK is able to model them better than existing baselines. This implementation uses continuous models for $\mathbf{x}$, but is still able approximate discrete distributions well. The components of each mixture appear centered around a discrete value, and have very low variance as shown in fig. 3.5. This yields a

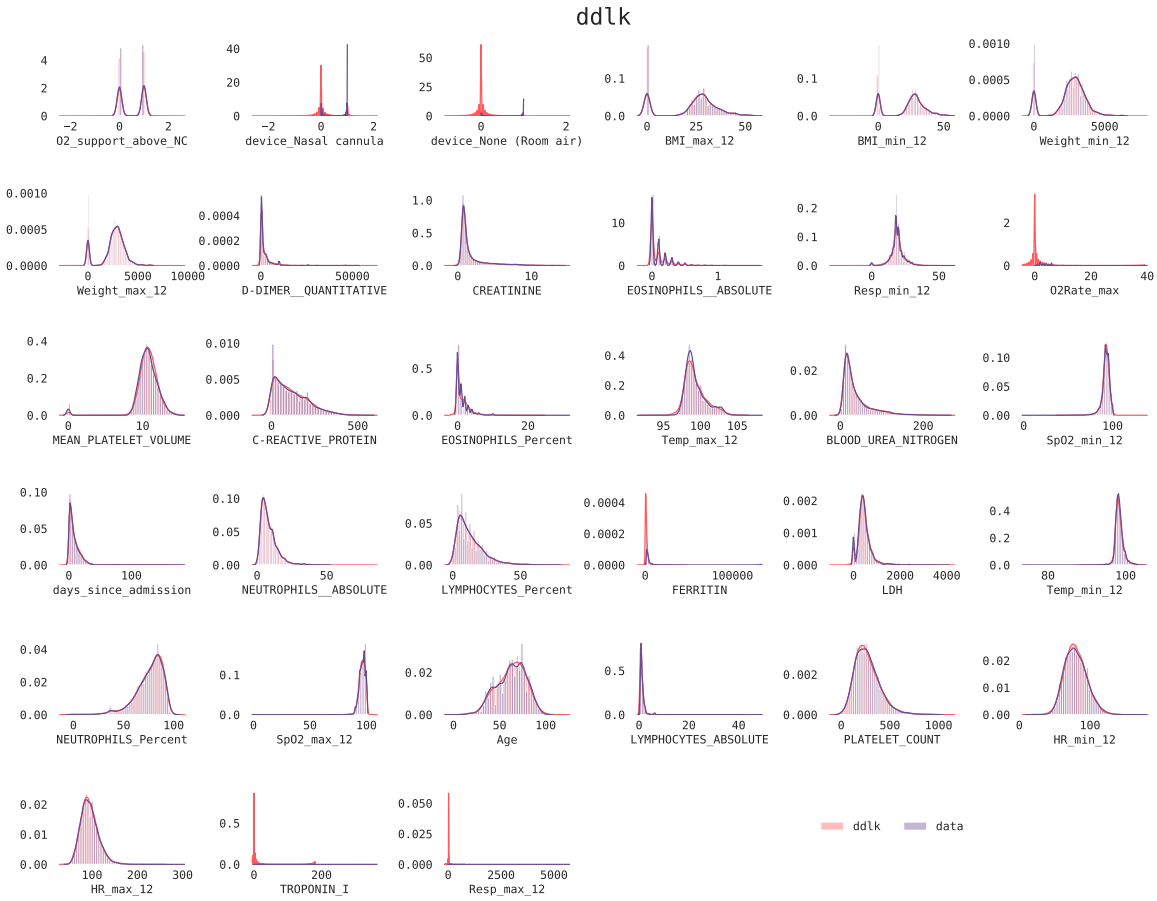


Figure 3.5: The marginal distributions of knockoff samples from DDLK look very similar to those from the data. Despite this implementation of DDLK using mixture density networks, the modes of each marginal line up with discrete values in the data.

close approximation to the true discrete marginal. We show the marginals of every feature for each knockoff method in figs. 3.5 to 3.8.

3.7 Discussion

DDLK is a generative model for sampling knockoffs that directly minimizes a KL divergence implied by the knockoff swap property. The optimization for DDLK involves first maximizing the explicit likelihood of the covariates, then minimizing the KL divergence between the joint distribution of covariates and knockoffs and any swap between them. To ensure DDLK satisfies

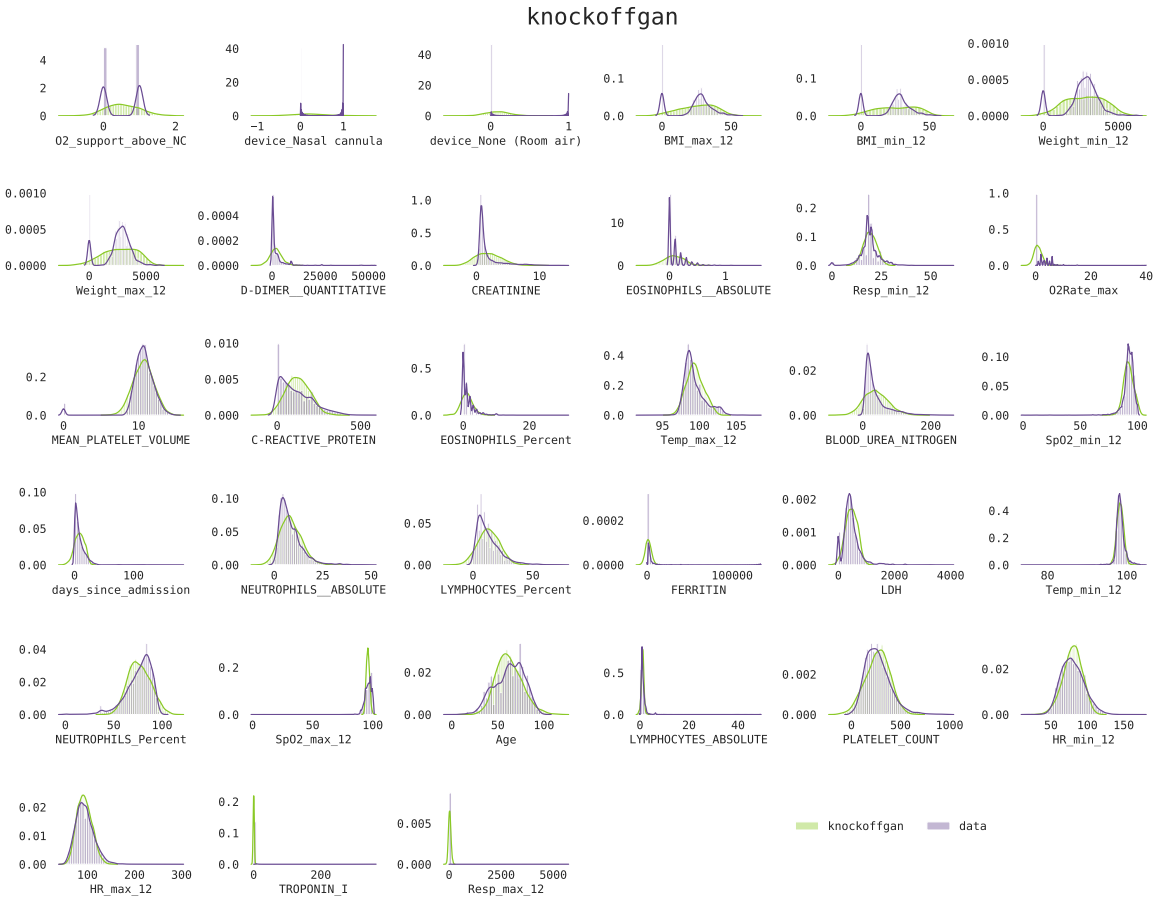


Figure 3.6: The marginals distributions of samples from KnockoffGAN match the data only when the feature $\mathbf{x}_j$ is univariate, and has roughly equal mass on either side of the mode.

the swap property under any swap indices, we use the Gumbel-Softmax trick to learn swaps that maximize the KL divergence. To generate knockoffs that satisfy the swap property while maintaining high power to select variables, DDLK includes a regularization term that encourages high conditional entropy of the knockoffs given the covariates. We find DDLK to outperform various baselines on several synthetic and real benchmarks including a task involving a large dataset from one of the epicenters of COVID-19.

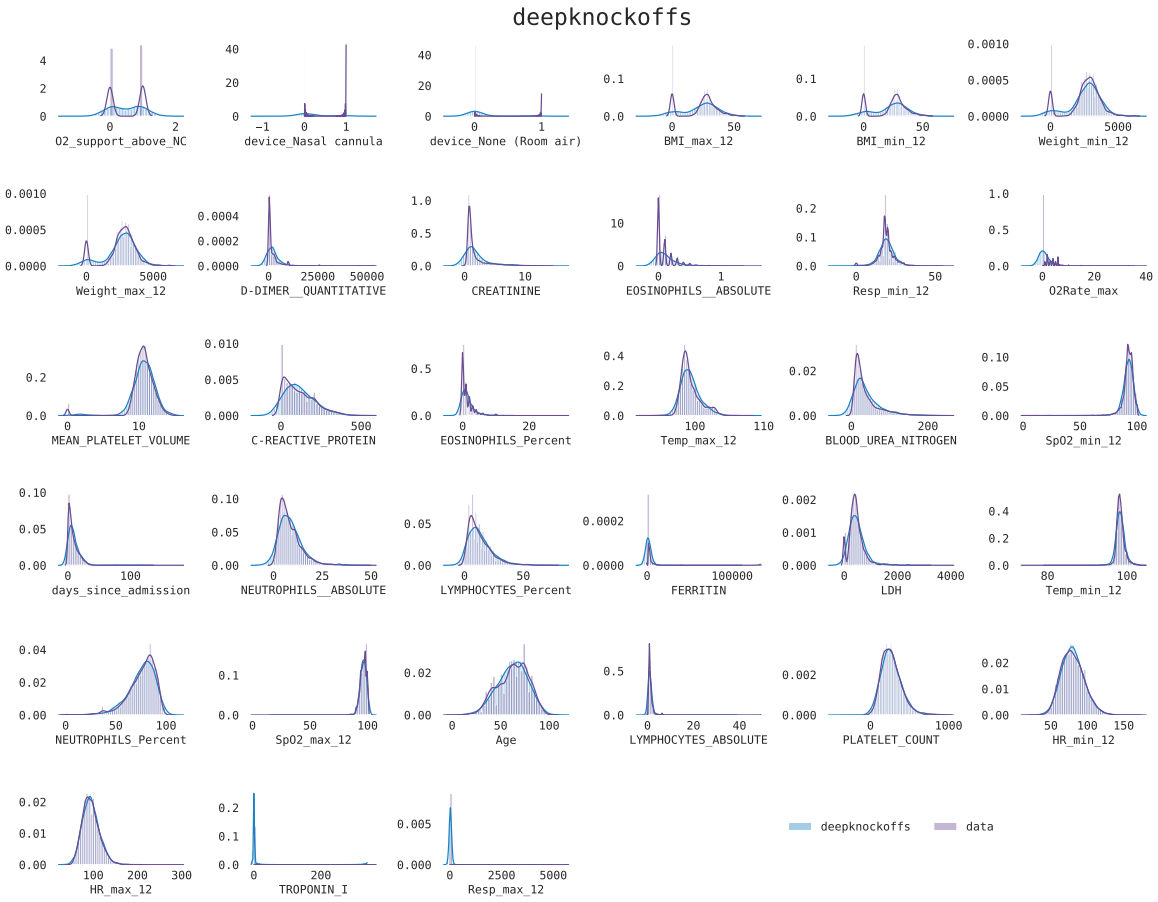


Figure 3.7: The marginals distributions of samples from Deep Knockoffs match the data only when the feature $\mathbf{x}_j$ is univariate and has fat tails.

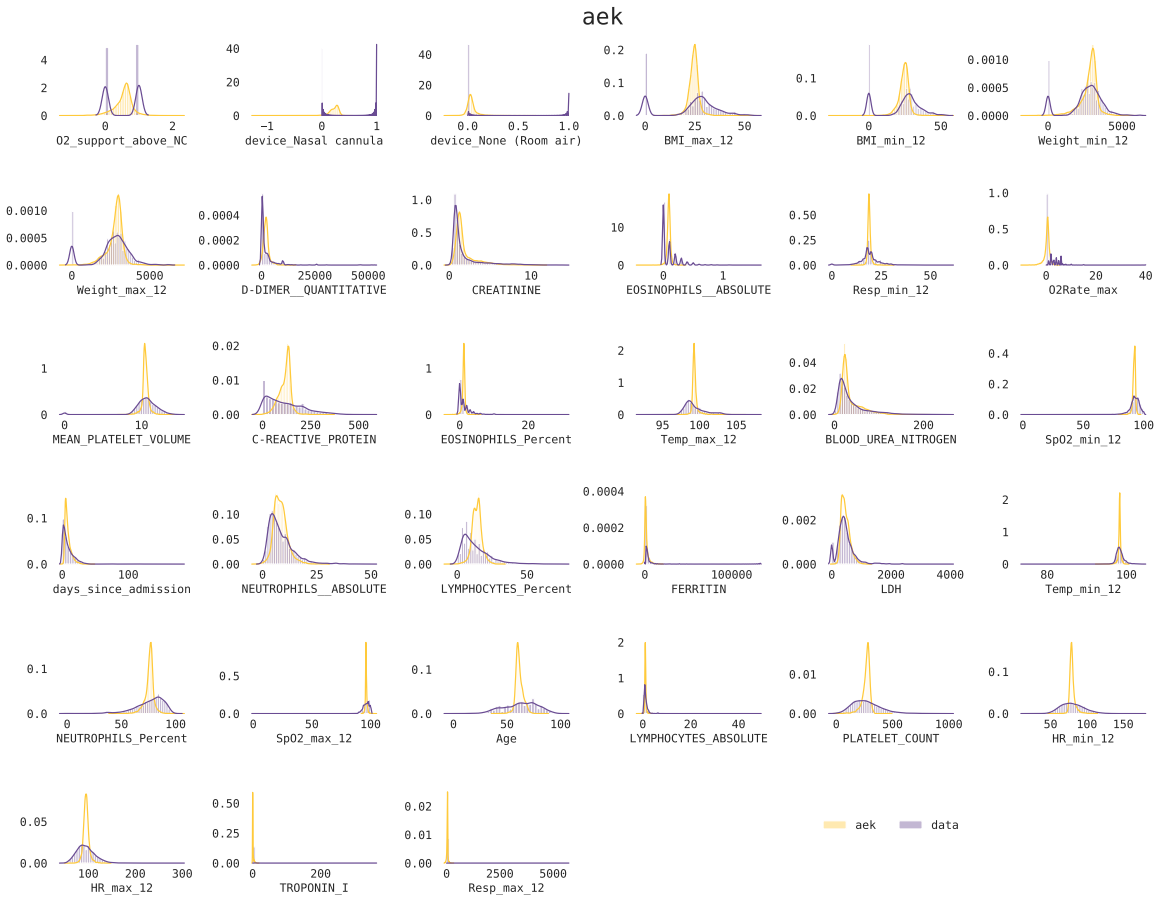


Figure 3.8: Auto-Encoding Knockoffs tend to learn underdispersed distributions for the covariates. Further, all of the marginal distributions learned are univariate and exhibit variance much smaller than that of the data.

4 | Contrarian randomization test (contra)

The holdout randomization test (HRT) discovers a set of covariates most predictive of a response. Given the distribution of covariates $\mathbf{x}$, HRTs can explicitly control the false discovery rate (FDR). However, if this distribution is unknown and must be estimated from data, HRTs can inflate the FDR. To alleviate the inflation of FDR, we propose the contrarian randomization test (CONTRA), which is designed explicitly for scenarios where the covariate distribution must be estimated from data and may even be misspecified. Our key insight is to use an equal mixture of two "contrarian" probabilistic models in determining the importance of a covariate. One model is fit with the real data, while the other is fit using the same data, but with the covariate being tested replaced with samples from an estimate of the covariate distribution. CONTRA is flexible enough to achieve a power of 1 asymptotically, can reduce the FDR compared to state-of-the-art CVS methods when the covariate distribution is misspecified, and is computationally efficient in high dimensions and large sample sizes. We further demonstrate the effectiveness of CONTRA on numerous synthetic benchmarks, and highlight its capabilities on a genetic dataset.

4.1 Motivation for CONTRA

Here we introduce review the working of an HRT and discuss some of its empirical issues.

Let $\mathbf{x} \in \mathbb{R}^d$ be a vector of covariates, $\mathbf{y} \in \mathbb{R}$ be a response, and $q(\mathbf{x}, \mathbf{y})$ be the generating distribution over $\mathbf{x}$ and $\mathbf{y}$. Let $(\mathbf{X}, \mathbf{Y}) := \{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^{n_{\text{train}}}$ be a training set of size n_{train}, and $(\mathbf{X}', \mathbf{Y}')$ be a test set of size n_{test}. Each sample $(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})$ in these datasets is drawn iid from $q(\mathbf{x}, \mathbf{y})$. Given samples of $(\mathbf{x}, \mathbf{y})$ data, the HRT tests the following hypothesis:

$$\mathcal{H}_0 : \mathbf{x}_j \perp \mathbf{y} \mid \mathbf{x}_{-j} \text{ vs } \mathcal{H}_1 : \mathbf{x}_j \not\perp \mathbf{y} \mid \mathbf{x}_{-j}. \tag{4.1}$$

It does so by first fitting a model $\hat{q}_{\text{model}}(\mathbf{y} \mid \mathbf{x})$ using $(\mathbf{X}, \mathbf{Y})$, then computing a statistic that uses $\hat{q}_{\text{model}}$'s empirical loss $\mathcal{L}$ on test set $(\mathbf{X}', \mathbf{Y}')$. Conditioned on the test set, the HRT samples M "null" datasets $\{\widetilde{\mathbf{X}}'^{(m)}\}_{m=1}^{M}$. Each dataset $\widetilde{\mathbf{X}}'^{(m)}$ consists of n_{test} samples where where the jth component of a sample $\widetilde{\mathbf{x}}^{(i)}$, $\widetilde{\mathbf{x}}_j^{(i)}$, is drawn from the conditional $q(\mathbf{x}_j \mid \mathbf{x}_{-j})$. A set of M statistics $\{\mathcal{L}(\mathbf{U}_j^{(m)}, \mathbf{Y}')\}_{m=1}^{M}$ is computed where $\mathbf{U}_j^{(m)}$ is a copy of $\mathbf{X}'$, but with the jth column swapped with that of $\widetilde{\mathbf{X}}'^{(m)}$. Finally, the importance of each $\mathbf{x}_j$ is assessed using the following p-value computation:

$$\frac{1}{M+1}\left(1 + \sum_{m=1}^{M} \mathbb{1}\left\{\mathcal{L}(\mathbf{X}', \mathbf{Y}') \geq \mathcal{L}(\mathbf{U}_j^{(m)}, \mathbf{Y}')\right\}\right). \tag{4.2}$$

Under the null hypothesis for $\mathbf{x}_j$, the swap property eq. (3.1) is satisfied by definition for the set $\{j\}$. As a result, the sequence:

$$\mathcal{T} := \{\mathcal{L}(\mathbf{U}_j^{(1)}, \mathbf{Y}'), \ldots, \mathcal{L}(\mathbf{U}_j^{(M)}, \mathbf{Y}'), \mathcal{L}(\mathbf{X}', \mathbf{Y}')\}$$

is exchangeable, so p-values computed using eq. (4.2) stochastically will dominate a Uniform(0,1) distribution [Tansey et al. 2018a]. Such p-values are sufficient to control the FDR at a nominal rate using standard multiple testing corrections like Benjamini and Yekutieli [2001] (these are summarized in appendix B of [Sudarshan et al. 2021]).

Under the alternate hypothesis, if $\mathcal{L}(\mathbf{X}', \mathbf{Y}')$ is typically smaller than $\mathcal{L}(\mathbf{U}_j^{(m)}, \mathbf{Y}')$, the HRT

will yield low p-values. The advantage of this property is that the power (the probability of selecting non-null covariates) depends entirely on how well $\hat{q}_{\text{model}}$ models $q(\mathbf{y} \mid \mathbf{x})$. Using a flexible model class can yield HRTs that have an asymptotic power of 1, meaning the important covariates are never missed.

EMPIRICAL ISSUES WITH HRTS. In practice, a few challenges exist with the HRT. First, the choice of performance metric can affect the power of the test to select important covariates. Second, the population distribution $q(\mathbf{x}_j \mid \mathbf{x}_{-j})$ is likely unknown and must be estimated from data. If null variables are sampled from estimated distributions, they may not satisfy the swap property (3.1) exactly. As a result, $\mathcal{L}(\mathbf{X}', \mathbf{Y}')$ may be consistently lower than $\mathcal{L}(\mathbf{U}_j^{(m)}, \mathbf{Y}')$, especially if $\hat{q}_{\text{model}}$ exhibits spurious dependence on a null covariate: a likely occurrence as shown by Efron [2012]. When $\hat{q}_{\text{model}}$ is evaluated on an out-of-distribution set $(\mathbf{U}_j^{(m)}, \mathbf{Y}')$, it will likely exhibit higher loss, regardless of whether or not the null hypothesis is true. In these situations, the HRT will artificially deflate the p-values computed using eq. (4.2). So, the HRT procedure will inflate FDR unless either $\hat{q}_{\text{model}}(\mathbf{y} \mid \mathbf{x}) = q(\mathbf{y} \mid \mathbf{x})$, or $\hat{q}_{\text{cc}}(\mathbf{x}_j \mid \mathbf{x}_{-j}) = q(\mathbf{x}_j \mid \mathbf{x}_{-j})$.

In attempt to circumvent this issue, Tansey et al. [2018a] introduce calibrated HRTs, which reweight terms in the p-value computation. These weights are learned by fitting B conditional distribution estimators $\{\hat{q}_{\text{cc}}^{(b)}(\mathbf{x}_j \mid \mathbf{x}_{-j})\}_{b=1}^{B}$, each using a different bootstrap of the data. Using these estimators, the authors weight the mth term in the p-value computation by

$$w^{(m)} = \begin{cases} \frac{\hat{q}_{\text{cc}}^{(l)}(\mathbf{x}_j|\mathbf{x}_{-j})}{\hat{q}_{\text{cc}}^{(1)}(\mathbf{x}_j|\mathbf{x}_{-j})} & \text{if } \mathcal{L}(\mathbf{x}, \mathbf{y}) < \mathcal{L}(\tilde{\mathbf{x}}_j^{(m)}, \mathbf{x}_{-j}, \mathbf{y}) \\ \frac{\hat{q}_{\text{cc}}^{(u)}(\mathbf{x}_j|\mathbf{x}_{-j})}{\hat{q}_{\text{cc}}^{(1)}(\mathbf{x}_j|\mathbf{x}_{-j})} & \text{otherwise} \end{cases}$$

where $\hat{q}_{\text{cc}}^{(l)}$ and $\hat{q}_{\text{cc}}^{(u)}$ are the lower and upper quantiles of the B estimators respectively. Intuitively, this reweighting of the p-value computation aims to make null p-values larger and the non-null p-values smaller. However, the effectiveness of this method is diminished as the sample size

increases. This is because the bootstrapped interval shrinks as sample size increases, and the lower and upper quantile estimators $\hat{q}_{\text{cc}}^{(l)}$ and $\hat{q}_{\text{cc}}^{(u)}$ will be closer to $\hat{q}_{\text{cc}}^{(1)}$, meaning $w^{(m)}$ is close to 1 and has no impact on eq. (4.2). So, if the $\hat{q}_{\text{cc}}$ models are misspecified, the ability of this technique to sufficiently calibrate p-values is reduced. Further, the number of estimators B is often large: Tansey et al. [2018b] set $B = 100$ in their experiments. This makes calibrated HRTs computationally expensive in high dimensions.

The issues discussed so far suggest a set of desiderata for any new CVS procedure. (1) It must be flexible enough to achieve a power of 1 asymptotically. (2) It must yield higher p-values than an HRT when the swap property in eq. (3.1) is violated. (3) It must be computationally efficient when performing CVS in high dimensions and large sample sizes.

4.2 Contrarian statistics

The primary goal of this section is to detail a procedure that is able to achieve a power of 1 asymptotically, while better controlling the FDR than HRTs when null variables must be estimated from data. We motivate our solution with the following intuition. The fundamental issue with HRTs is that null covariates drawn from estimated distributions can cause the loss $\mathcal{L}(\mathbf{X}', \mathbf{Y}')$ to be lower than $\mathcal{L}(\mathbf{U}_j^{(m)}, \mathbf{Y}')$ even if the jth covariate is not important to $\mathbf{y}$. This is because $\hat{q}_{\text{model}}$ performs worse on the dataset $(\mathbf{U}_j^{(m)}, \mathbf{Y}')$, which is not equal in distribution to $(\mathbf{X}', \mathbf{Y}')$. One solution to this problem is to bring the true and null losses closer together. By using a "contrarian" model $\hat{q}_{\text{mix}}$ – one that performs better than $\hat{q}_{\text{model}}$ on $(\mathbf{U}_j^{(m)}, \mathbf{Y}')$ but worse on $(\mathbf{X}', \mathbf{Y}')$ – p-values computed using eq. (4.2) can be made higher. Multiple testing correction procedures will then select fewer covariates, thus lowering the FDR.

In the next few sections, we will introduce CONTRA, a procedure to build such contrarian models, then discuss its useful theoretical and empirical properties.

4.2.1 Building a contrarian test

Let $(\mathbf{X}, \mathbf{Y})$ be a training set of size n_{train}, and $(\mathbf{X}', \mathbf{Y}')$ be a test set of size n_{test}. Each sample $(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})$ in these datasets is drawn iid from $q(\mathbf{x}, \mathbf{y})$. CONTRA first fits a probabilistic model $\hat{q}_{\text{model}}(\mathbf{y} \mid \mathbf{x})$ to $(\mathbf{X}, \mathbf{Y})$, and a set of conditional distribution estimators $\{\hat{q}_{\text{cc}}^{(j)}(\mathbf{x}_j \mid \mathbf{x}_{-j})\}_{j=1}^{d}$ using $\mathbf{X}$. Then, CONTRA generates $M + 1$ null datasets. One to train models: $\widetilde{\mathbf{X}}$, and M to compute p-values: $\{\widetilde{\mathbf{X}}'^{(m)}\}_{m=1}^{M}$. The jth coordinate of each element $\widetilde{\mathbf{x}}^{(i)}$ in $\widetilde{\mathbf{X}}$ is drawn from the *estimated* $\hat{q}_{\text{cc}}^{(j)}(\mathbf{x}_j \mid \mathbf{x}_{-j} = \mathbf{x}^{(i)})$ conditional on the ith training sample in $\mathbf{X}$. Each $\widetilde{\mathbf{X}}'^{(m)}$ is generated the same way, but conditioned on the test set $\mathbf{X}'$ instead.

The next step in CONTRA is to fit a set of d probabilistic models $\{\hat{q}_{\text{null}}^{(j)}(\mathbf{y} \mid \widetilde{\mathbf{x}}_j, \mathbf{x}_{-j})\}_{j=1}^{d}$. Each model $\hat{q}_{\text{null}}^{(j)}$ is fit using the data $(\mathbf{U}_j, \mathbf{Y})$, where $\mathbf{U}_j$ is identical to $\mathbf{X}$, but with the jth column of $\mathbf{X}$ replaced with the jth column of $\widetilde{\mathbf{X}}$. These models will serve as the basis for our contrarian models $\{\hat{q}_{\text{mix}}^{(j)}\}_{j=1}^{d}$, where

$$\hat{q}_{\text{mix}}^{(j)}(\mathbf{y} \mid \mathbf{x}) := \frac{1}{2}\left(\hat{q}_{\text{model}}(\mathbf{y} \mid \mathbf{x}) + \hat{q}_{\text{null}}^{(j)}(\mathbf{y} \mid \mathbf{x})\right).$$

Each $\hat{q}_{\text{mix}}^{(j)}$ is a mixture of the model fit to the true data $\hat{q}_{\text{model}}$, and the model fit to the null data $\hat{q}_{\text{null}}$ for the jth covariate.

To test the conditional independence of each covariate $\mathbf{x}_j$ with $\mathbf{y}$ conditioned on $\mathbf{x}_{-j}$, CONTRA first computes the following test statistic using the test set:

$$\ell^{(j)}(\mathbf{X}', \mathbf{Y}') = \sum_{i=1}^{n_{\text{test}}} -\log \hat{q}_{\text{mix}}^{(j)}(\mathbf{y} = \mathbf{y}^{(i)} \mid \mathbf{x} = \mathbf{x}^{(i)}).$$

Finally, a computation similar to eq. (4.2) is used to compute p-values for each covariate:

$$\frac{1}{M+1}\left(1 + \sum_{m=1}^{M} \mathbb{1}\left\{\ell^{(j)}(\mathbf{X}', \mathbf{Y}') \geq \ell^{(j)}(\mathbf{U}_j^{(m)}, \mathbf{Y}')\right\}\right) \tag{4.3}$$

where $\mathbf{U}_j^{(m)}$ is a copy of $\mathbf{X}'$, but with the jth column swapped with that of $\widetilde{\mathbf{X}}'^{(m)}$.

Intuitively, the use of $\hat{q}_{\text{mix}}^{(j)}$ over $\hat{q}_{\text{model}}$ will decrease the gap between $\ell^{(j)}(\mathbf{X}', \mathbf{Y}')$ and $\ell^{(j)}(\mathbf{U}_j^{(m)}, \mathbf{Y}')$. This is because the mixture of $\hat{q}_{\text{model}}$ and $\hat{q}_{\text{null}}^{(j)}$ will perform worse on the set $(\mathbf{X}', \mathbf{Y}')$, but better on $(\mathbf{U}_j^{(m)}, \mathbf{Y}')$. At first glance, this seems to mitigate the FDR control issue of HRTs but at the cost of power to select non-null covariates.

In the next few sections, we show that CONTRA retains the most important property of the HRT: finite sample FDR control when the null variables satisfy the swap property in eq. (3.1). *Despite using contrarian models*, CONTRA achieves power 1 asymptotically when the model distributions $\hat{q}_{\text{model}}$ and $\hat{q}_{\text{null}}^{(j)}$ converge in probability to $q(\mathbf{y} \mid \mathbf{x})$ and $q(\mathbf{y} \mid \mathbf{x}_{-j})$.

4.2.2 CONTRA CONTROLS FDR AND ACHIEVES POWER 1

To prove properties about CONTRA's FDR and power, we discuss the p-values produced by CONTRA.

FINITE SAMPLE FDR. Procedures that control the FDR require null p-values to exhibit stochastic dominance over a Uniform(0,1) random variable [Benjamini and Hochberg 1995; Benjamini and Yekutieli 2001].

Proposition 4.2.1. *Null p-values p_j produced by CONTRA on a test dataset $(\mathbf{X}', \mathbf{Y}')$ will stochastically dominate $\mathbf{u} \sim$ Uniform$(0, 1)$ for any covariate $\mathbf{x}_j$ that is independent of response $\mathbf{y}$ having observed the other covariates $\mathbf{x}_{-j}$.*

The proof of prop. 4.2.1 can be seen from the fact that CONTRA is a variant of the CRT. Before giving the full proof, we outline a sketch here. We note that for each null covariate $\mathbf{x}_j$, $(\mathbf{X}', \mathbf{Y}')$ is equal in distribution to any null dataset $(\mathbf{U}_j^{(m)}, \mathbf{Y}')$. As a result, $\ell^{(j)}(\mathbf{X}', \mathbf{Y}')$ is equal in distribution to $\ell^{(j)}(\mathbf{U}_j^{(m)}, \mathbf{Y}')$. Since the distribution functions of the test and null statistics are equal, they share the same cumulative distribution function (CDF). We can then show that the p-value computation in eq. (4.3) will yield a random variable whose CDF is always less than or equal to the CDF of a uniform random variable: the definition of stochastic dominance.

Proof. First, we will show that the datasets $\mathbf{X}'$ and $\mathbf{U}_j^{(m)}$ are equal in distributions. Under the null, $\mathbf{x}_j$ is independent of $\mathbf{y}$ given $\mathbf{x}_{-j}$. Using [Candes et al. 2018], this means the swap property mentioned earlier in eq. (3.1) is satisfied:

$$[\mathbf{x}_j, \mathbf{x}_{-j}] \stackrel{d}{=} [\widetilde{\mathbf{x}}_j, \mathbf{x}_{-j}].$$

Since the samples in $\mathbf{X}'$ and $\mathbf{U}_j^{(m)}$ are iid, the datasets $\mathbf{X}'$ and $\mathbf{U}_j^{(m)}$ will also be equal in distribution.

Recall that the components of $\hat{q}_{\text{mix}}^{(j)}$, $\hat{q}_{\text{model}}$ and $\hat{q}_{\text{null}}^{(j)}$ are fit using the training set, and are therefore independent of $(\mathbf{X}', \mathbf{Y}')$. Therefore, the statistics $\ell^{(j)}(\mathbf{X}', \mathbf{Y}')$ and $\ell^{(j)}(\mathbf{U}_j^{(m)}, \mathbf{Y}')$ are also equal in distribution.

Next, we define $F_{n_{\text{test}}}^{(j)}(t)$ to be the empirical CDF of $\ell^{(j)}(\mathbf{X}', \mathbf{Y}')$ where n_{test} is the number of samples in the test set. Note that the equality in distribution between $\mathbf{U}_j^{(m)}$ and $\mathbf{X}'$ implies that $\ell^{(j)}(\mathbf{U}_j^{(m)}, \mathbf{Y}')$ has the same CDF. We also define $F_{n_{\text{test}}}^{-1}(t) = \inf\{u \in \mathbb{R} : F_{n_{\text{test}}}^{(j)}(u) \geq t\}$: the generalized inverse CDF.

Using the empirical CDF, we can represent the null p-values in a more convenient form. In the limit of M, the number of null datasets sampled, the jth CONTRA p-value is simply:

$$p_j = \mathbb{P}\left\{\ell^{(j)}(\mathbf{X}', \mathbf{Y}') \geq \ell^{(j)}(\mathbf{U}_j^{(m)}, \mathbf{Y}')\right\} = F_{n_{\text{test}}}^{(j)}(\ell^{(j)}(\mathbf{U}_j^{(m)}, \mathbf{Y}')).$$

With this representation of the p-value, we see that

$$\begin{aligned}
\mathbb{P}\{p_j \leq \alpha\} &= \mathbb{P}\{F_{n_{\text{test}}}^{(j)}(\ell^{(j)}(\mathbf{U}_j^{(m)}, \mathbf{Y}')) \leq \alpha\} \\
&= \mathbb{P}\{\ell^{(j)}(\mathbf{U}_j^{(m)}, \mathbf{Y}') \leq F_{n_{\text{test}}}^{-1}(\alpha)\} \\
&= F_{n_{\text{test}}}^{(j)}(F_{n_{\text{test}}}^{-1}(\alpha)) = \alpha.
\end{aligned}$$

In the first line, we use the alternative representation of the null p-value. In the second line, we

apply the generalized inverse CDF to both sides. Finally, we use the definition of the CDF. Thus, the CDF of the null p-value is equal to that of a uniform random variable. In finite samples, adding 1 to the sum of M indicator functions, then dividing by $(M+1)$ ensures that p_j will stochastically dominate a Uniform(0,1) random variable since the minimum p-value is then $1/(M+1)$. □

ASYMPTOTIC POWER OF 1. The power of a CVS procedure is the probability that an important covariate $\mathbf{x}_j$ is selected. An important covariate $\mathbf{x}_j$ is selected only when its p-value is below a certain threshold. Intuitively then, the lower the p-value, the more likely $\mathbf{x}_j$ is to be selected.

Proposition 4.2.2. *If $\hat{q}_{model}$ and $\hat{q}^{(j)}_{null}$ converge in probability to distributions $q(\mathbf{y} \mid \mathbf{x})$ and $q(\mathbf{y} \mid \mathbf{x}_{-j})$ respectively, the CONTRA p-value for an important covariate $\mathbf{x}_j$ will converge in probability to 0 in the limit of the sample size, thus yielding a method with power 1.*

The proof sketch is as follows. We know that $\hat{q}^{(j)}_{\text{null}}$ converges in probability to $q(\mathbf{y} \mid \mathbf{x}_{-j})$ since $\widetilde{\mathbf{x}}_j$ is generated specifically to be independent of $\mathbf{y} \mid \mathbf{x}_{-j}$. We then analyze the difference of the two inner terms of the p-value computation in eq. (4.3) by showing that $\ell^{(j)}(\mathbf{X}', \mathbf{Y}') - \ell^{(j)}(\mathbf{U}^{(m)}_j, \mathbf{Y}') < 0$. We show that an upper bound for this difference is the sum of two negative KL terms, which will be strictly negative when the null hypothesis is not true.

Proof. Let $(\mathbf{X}, \mathbf{Y})$ be a training set consisting of n_{train} samples $(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})$. Let $\widetilde{\mathbf{X}}$ be a set of null data where the jth coordinate of its ith sample, $\widetilde{\mathbf{x}}^{(i)}_j$, is sampled from $\hat{q}^{(j)}_{\text{cc}}(\mathbf{x}_j \mid \mathbf{x}_{-j} = \mathbf{x}^{(i)}_{-j})$. Let $(\mathbf{U}_j, \mathbf{Y})$ be identical to $(\mathbf{X}, \mathbf{Y})$, but with the jth coordinate swapped out for the jth coordinate of $\widetilde{\mathbf{X}}$. Now let $\hat{q}_{\text{model}}(\mathbf{y} \mid \mathbf{x})$ be fit using $(\mathbf{X}, \mathbf{Y})$, and $\hat{q}^{(j)}_{\text{null}}$ be fit using $(\mathbf{U}_j, \mathbf{Y})$.

To formalize the assumptions in prop. 4.2.2, we use the average KL between each population distribution and its respective model distribution:

$$\lim_{n_{\text{train}} \to \infty} \mathbb{E}_{q(\mathbf{x})} \mathbb{E}_{q(\mathbf{y}|\mathbf{x})} \log \frac{q(\mathbf{y} \mid \mathbf{x})}{\hat{q}_{\text{model}}(\mathbf{y} \mid \mathbf{x})} = 0$$

$$\lim_{n_{\text{train}} \to \infty} \mathbb{E}_{q(\mathbf{x}_{-j})} \mathbb{E}_{\hat{q}_{\text{cc}}(\mathbf{x}_j|\mathbf{x}_{-j})} \mathbb{E}_{q(\mathbf{y}|\mathbf{x}_{-j})} \log \frac{q(\mathbf{y} \mid \mathbf{x}_{-j})}{\hat{q}^{(j)}_{\text{null}}(\mathbf{y} \mid \widetilde{\mathbf{x}}_j, \mathbf{x}_{-j})} = 0 \qquad \forall j \in \{1, \ldots, d\}.$$

Recall that $\hat{q}^{(j)}_{\text{null}}$ is fit using $(\mathbf{U}_j, \mathbf{Y})$, where the jth covariate is designed to have no dependence on $\mathbf{y}$. So $\hat{q}^{(j)}_{\text{null}}$, which has no dependence on $\mathbf{x}_j$, converges to $q(\mathbf{y} \mid \mathbf{x}_{-j})$ in KL.

Now, we use a result from Tsybakov [2008] that helps bound the squared Hellinger distance between two distributions using the KL between them:

$$\int \left(\sqrt{q(\mathbf{y} \mid \mathbf{x})} - \sqrt{\hat{q}_{\text{model}}(\mathbf{y} \mid \mathbf{x})}\right)^2 d\mathbf{y} \leq \text{KL}(q(\mathbf{y} \mid \mathbf{x}) \parallel \hat{q}_{\text{model}}(\mathbf{y} \mid \mathbf{x}))$$
$$\int \left(\sqrt{q(\mathbf{y} \mid \mathbf{x}_{-j})} - \sqrt{\hat{q}^{(j)}_{\text{null}}(\mathbf{y} \mid \widetilde{\mathbf{x}}_j, \mathbf{x}_{-j})}\right)^2 d\mathbf{y} \leq \text{KL}(q(\mathbf{y} \mid \mathbf{x}_{-j}) \parallel \hat{q}^{(j)}_{\text{null}}(\mathbf{y} \mid \widetilde{\mathbf{x}}_j, \mathbf{x}_{-j}))$$

This means that the log densities of the model and population distributions must be equal almost surely when $(\mathbf{x}, \mathbf{y}) \sim q(\mathbf{x})q(\mathbf{y} \mid \mathbf{x})$ and $\widetilde{\mathbf{x}}_j \sim q(\mathbf{x}_j \mid \mathbf{x}_{-j})$:

$$\lim_{n_{\text{train}} \to \infty} \log \frac{q(\mathbf{y} \mid \mathbf{x})}{\hat{q}_{\text{model}}(\mathbf{y} \mid \mathbf{x})} = 0$$
$$\lim_{n_{\text{train}} \to \infty} \log \frac{q(\mathbf{y} \mid \mathbf{x}_{-j})}{\hat{q}^{(j)}_{\text{null}}(\mathbf{y} \mid \widetilde{\mathbf{x}}_j, \mathbf{x}_{-j})} = 0.$$

Otherwise, the Hellinger distance will be positive, implying a positive KL between the model and population distributions.

Now, let $(\mathbf{X}', \mathbf{Y}')$ be a test set of covariates consisting of n_{test} samples. Let $\widetilde{\mathbf{X}}'^{(m)}$ be a null set sampled in the same way as $\widetilde{\mathbf{X}}$, but conditioned on samples from $\mathbf{X}'$, instead of $\mathbf{X}$. Let $\mathbf{U}^{(m)}_j$ be constructed the same way as $\mathbf{U}_j$, but using $\mathbf{X}'$ and $\widetilde{\mathbf{X}}'^{(m)}$ instead. In the limit of $n_{\text{train}}, n_{\text{test}}$, we want to show that

$$\ell^{(j)}(\mathbf{X}', \mathbf{Y}') < \ell^{(j)}(\mathbf{U}^{(m)}_j, \mathbf{Y}').$$

If this inequality is true as $(n_{\text{train}}, n_{\text{test}}) \to (\infty, \infty)$, then the indicator inside the p-value computation is always 0, leading to a p-value of 0.

Recall in the limit of n_{test}, $\ell^{(j)}(\mathbf{X}', \mathbf{Y}')$ is equal to $\mathbb{E}_{q(\mathbf{x},\mathbf{y})} \log \hat{q}_{\text{mix}}(\mathbf{y} \mid \mathbf{x})$. A similar limit exists

for $\ell^{(j)}(\mathbf{U}_j^{(m)}, \mathbf{Y}')$. Using this fact, we can expand $\ell^{(j)}(\mathbf{X}', \mathbf{Y}')$:

$$
\begin{aligned}
&\lim_{(n_{\text{train}}, n_{\text{test}}) \to (\infty, \infty)} \ell^{(j)}(\mathbf{X}', \mathbf{Y}') \\
&= -\mathbb{E}_{q(\mathbf{x}, \mathbf{y})} \log 0.5 \left(q(\mathbf{y} \mid \mathbf{x}) + q(\mathbf{y} \mid \mathbf{x}_{-j}) \right) \\
&= -\mathbb{E}_{q(\mathbf{y}, \mathbf{x}_{-j})} \mathbb{E}_{q(\mathbf{x}_j \mid \mathbf{y}, \mathbf{x}_{-j})} \log 0.5 (q(\mathbf{y} \mid \mathbf{x}) + q(\mathbf{y} \mid \mathbf{x}_{-j})) \\
&= -\mathbb{E}_{q(\mathbf{y}, \mathbf{x}_{-j})} \mathbb{E}_{q(\mathbf{x}_j \mid \mathbf{y}, \mathbf{x}_{-j})} \log 0.5 \left(\frac{q(\mathbf{y} \mid \mathbf{x}_{-j}) q(\mathbf{x}_j \mid \mathbf{x}_{-j}, \mathbf{y})}{q(\mathbf{x}_j \mid \mathbf{x}_{-j})} + q(\mathbf{y} \mid \mathbf{x}_{-j}) \right) \\
&= -\mathbb{E}_{q(\mathbf{y}, \mathbf{x}_{-j})} \mathbb{E}_{q(\mathbf{x}_j \mid \mathbf{y}, \mathbf{x}_{-j})} \log 0.5 q(\mathbf{y} \mid \mathbf{x}_{-j}) \left(\frac{q(\mathbf{x}_j \mid \mathbf{x}_{-j}, \mathbf{y})}{q(\mathbf{x}_j \mid \mathbf{x}_{-j})} + 1 \right) \\
&= -\mathbb{E}_{q(\mathbf{y}, \mathbf{x}_{-j})} \mathbb{E}_{q(\mathbf{x}_j \mid \mathbf{y}, \mathbf{x}_{-j})} \log q(\mathbf{y} \mid \mathbf{x}_{-j}) + \log 0.5 \left(\frac{q(\mathbf{x}_j \mid \mathbf{x}_{-j}, \mathbf{y})}{q(\mathbf{x}_j \mid \mathbf{x}_{-j})} + 1 \right) \\
&= -\mathbb{E}_{q(\mathbf{y}, \mathbf{x}_{-j})} \log q(\mathbf{y} \mid \mathbf{x}_{-j}) - \mathbb{E}_{q(\mathbf{y}, \mathbf{x}_{-j})} \mathbb{E}_{q(\mathbf{x}_j \mid \mathbf{y}, \mathbf{x}_{-j})} \log 0.5 \left(\frac{q(\mathbf{x}_j \mid \mathbf{x}_{-j}, \mathbf{y})}{q(\mathbf{x}_j \mid \mathbf{x}_{-j})} + 1 \right).
\end{aligned}
$$

We now use the fact that the geometric mean is less than the arithmetic mean to upper bound $\ell^{(j)}(\mathbf{X}', \mathbf{Y}')$:

$$
\begin{aligned}
\ell^{(j)}(\mathbf{X}', \mathbf{Y}') &\leq -\mathbb{E}_{q(\mathbf{y}, \mathbf{x}_{-j})} \log q(\mathbf{y} \mid \mathbf{x}_{-j}) - \mathbb{E}_{q(\mathbf{y}, \mathbf{x}_{-j})} \mathbb{E}_{q(\mathbf{x}_j \mid \mathbf{y}, \mathbf{x}_{-j})} \log \sqrt{\frac{q(\mathbf{x}_j \mid \mathbf{x}_{-j}, \mathbf{y})}{q(\mathbf{x}_j \mid \mathbf{x}_{-j})}} \\
&= -\mathbb{E}_{q(\mathbf{y}, \mathbf{x}_{-j})} \log q(\mathbf{y} \mid \mathbf{x}_{-j}) - 0.5 \cdot \mathbb{E}_{q(\mathbf{y}, \mathbf{x}_{-j})} \mathbb{E}_{q(\mathbf{x}_j \mid \mathbf{y}, \mathbf{x}_{-j})} \log \frac{q(\mathbf{x}_j \mid \mathbf{x}_{-j}, \mathbf{y})}{q(\mathbf{x}_j \mid \mathbf{x}_{-j})} \\
&= -\mathbb{E}_{q(\mathbf{y}, \mathbf{x}_{-j})} \log q(\mathbf{y} \mid \mathbf{x}_{-j}) - 0.5 \cdot \mathbb{E}_{q(\mathbf{y}, \mathbf{x}_{-j})} \mathrm{KL}(q(\mathbf{x}_j \mid \mathbf{x}_{-j}, \mathbf{y}) \parallel q(\mathbf{x}_j \mid \mathbf{x}_{-j})).
\end{aligned}
$$

Using similar arithmetic, we now expand $\ell^{(j)}(\mathbf{U}_j^{(m)}, \mathbf{Y}')$:

$$
\begin{aligned}
&\lim_{(n_{\text{train}}, n_{\text{test}}) \to (\infty, \infty)} \ell^{(j)}(\mathbf{U}_j^{(m)}, \mathbf{Y}') \\
&= -\mathbb{E}_{q(\mathbf{y}, \mathbf{x}_{-j})} \mathbb{E}_{q(\mathbf{x}_j \mid \mathbf{x}_{-j})} \log 0.5 (q(\mathbf{y} \mid \mathbf{x}) + q(\mathbf{y} \mid \mathbf{x}_{-j})) \\
&= -\mathbb{E}_{q(\mathbf{y}, \mathbf{x}_{-j})} \mathbb{E}_{q(\mathbf{x}_j \mid \mathbf{x}_{-j})} \log q(\mathbf{y} \mid \mathbf{x}_{-j}) + \log 0.5 \left(\frac{q(\mathbf{x}_j \mid \mathbf{x}_{-j}, \mathbf{y})}{q(\mathbf{x}_j \mid \mathbf{x}_{-j})} + 1 \right) \\
&= -\mathbb{E}_{q(\mathbf{y}, \mathbf{x}_{-j})} \mathbb{E}_{q(\mathbf{x}_j \mid \mathbf{x}_{-j})} \log q(\mathbf{y} \mid \mathbf{x}_{-j}) + \log \left(\frac{0.5 q(\mathbf{x}_j \mid \mathbf{x}_{-j}, \mathbf{y}) + 0.5 q(\mathbf{x}_j \mid \mathbf{x}_{-j})}{q(\mathbf{x}_j \mid \mathbf{x}_{-j})} \right)
\end{aligned}
$$

$$= -\mathbb{E}_{q(\mathbf{y},\mathbf{x}_{-j})} \log q(\mathbf{y} \mid \mathbf{x}_{-j}) - \mathbb{E}_{q(\mathbf{x}_j|\mathbf{x}_{-j})} \log \left(\frac{0.5q(\mathbf{x}_j \mid \mathbf{x}_{-j}, \mathbf{y}) + 0.5q(\mathbf{x}_j \mid \mathbf{x}_{-j})}{q(\mathbf{x}_j \mid \mathbf{x}_{-j})} \right)$$

$$= -\mathbb{E}_{q(\mathbf{y},\mathbf{x}_{-j})} \log q(\mathbf{y} \mid \mathbf{x}_{-j}) + \mathrm{KL}(q(\mathbf{x}_j \mid \mathbf{x}_{-j}) \parallel 0.5q(\mathbf{x}_j \mid \mathbf{x}_{-j}, \mathbf{y}) + 0.5q(\mathbf{x}_j \mid \mathbf{x}_{-j})).$$

Putting it all together:

$$\begin{aligned}
&\lim_{(n_{\text{train}}, n_{\text{test}}) \to (\infty, \infty)} \ell^{(j)}(\mathbf{U}_j^{(m)}, \mathbf{Y}') - \ell^{(j)}(\mathbf{X}', \mathbf{Y}') \\
\geq &\mathbb{E}_{q(\mathbf{y},\mathbf{x}_{-j})} \log q(\mathbf{y} \mid \mathbf{x}_{-j}) + 0.5 \cdot \mathbb{E}_{q(\mathbf{y},\mathbf{x}_{-j})} \mathrm{KL}(q(\mathbf{x}_j \mid \mathbf{x}_{-j}, \mathbf{y}) \parallel q(\mathbf{x}_j \mid \mathbf{x}_{-j})) \\
&- \left(\mathbb{E}_{q(\mathbf{y},\mathbf{x}_{-j})} \log q(\mathbf{y} \mid \mathbf{x}_{-j}) - \mathrm{KL}(q(\mathbf{x}_j \mid \mathbf{x}_{-j}) \parallel 0.5q(\mathbf{x}_j \mid \mathbf{x}_{-j}, \mathbf{y}) + 0.5q(\mathbf{x}_j \mid \mathbf{x}_{-j})) \right) \\
= &0.5 \cdot \mathbb{E}_{q(\mathbf{y},\mathbf{x}_{-j})} \mathrm{KL}(q(\mathbf{x}_j \mid \mathbf{x}_{-j}, \mathbf{y}) \parallel q(\mathbf{x}_j \mid \mathbf{x}_{-j})) \\
&+ \mathrm{KL}(q(\mathbf{x}_j \mid \mathbf{x}_{-j}) \parallel 0.5q(\mathbf{x}_j \mid \mathbf{x}_{-j}, \mathbf{y}) + 0.5q(\mathbf{x}_j \mid \mathbf{x}_{-j})) \\
> &0.
\end{aligned}$$

The key takeaway here is that the difference between $\ell^{(j)}(\mathbf{U}_j^{(m)}, \mathbf{Y}')$ and $\ell^{(j)}(\mathbf{X}', \mathbf{Y}')$ is lower bounded by the sum of two Kullback–Leibler divergence (KL) terms, both of which are strictly positive in the case of a non-null covariate. Thus, we have shown that in the limit of n_{test}, p-values produced by CONTRA will be 0 in distribution. This also implies the p-values converge to 0 in probability since 0 is a constant. □

Prop. 4.2.2 highlights a noteworthy property of CONTRA. Despite using $\hat{q}_{\text{mix}}^{(j)}$, which is designed *intentionally* to exhibit higher loss than $\hat{q}_{\text{model}}$ on the test set, the asymptotic guarantees of CONTRA are just as strong as those of any HRT. While it is theoretically possible for HRTs to enjoy higher power in small sample sizes, we will soon show empirically that this difference in power is negligible.

4.2.3 CONTRA PREVENTS FDR INFLATION

We have thus far seen that CONTRA preserves the useful attributes of HRTs: finite sample FDR control when $\hat{q}_{\text{cc}}^{(j)}(\mathbf{x}_j \mid \mathbf{x}_{-j}) = q(\mathbf{x}_j \mid \mathbf{x}_{-j})$ and an asymptotic power of 1. In this section, we will discuss the primary advantages of CONTRA over the HRT that make it a useful empirical procedure: (a) its null p-values are higher than those of the HRT when the swap property is violated, and (b) it is still computationally efficient with respect to HRTs.

HIGHER NULL p-VALUES. To highlight the main pitfall of HRTs in practice, consider the following scenario. Let $\mathbf{x}_k$ and $\mathbf{x}_j$ be two covariates that have high mutual information, but only $\mathbf{x}_k$ is in the Markov blanket of the response $\mathbf{y}$. In finite samples, $\hat{q}_{\text{model}}$ can exhibit spurious dependence on $\mathbf{x}_j$ [Efron 2012]. As a result, if the estimated $\hat{q}_{\text{cc}}^{(j)} \neq q(\mathbf{x}_j \mid \mathbf{x}_{-j})$, the loss of $\hat{q}_{\text{model}}$ on $(\mathbf{X}', \mathbf{Y}')$ will typically be less than its loss on $(\mathbf{U}_j^{(m)}, \mathbf{Y}')$, *even when* $\mathbf{x}_j$ *is not important to* $\mathbf{y}$. This is because the performance of $\hat{q}_{\text{model}}$ will suffer when it is evaluated on a distribution other than the one being trained, as studied in the domain adaption literature [Crammer et al. 2008; Daumé III 2009]. In these situations, the resulting p-values will be deflated, leading to a violation of FDR control.

Contrarian models prevent deflated p-values. To understand how CONTRA does so, consider the loss of $\hat{q}_{\text{mix}}^{(j)}$ on each of $(\mathbf{X}', \mathbf{Y}')$ and $(\mathbf{U}_j^{(m)}, \mathbf{Y}')$. The mixture $\hat{q}_{\text{mix}}^{(j)}$ contains $\hat{q}_{\text{null}}^{(j)}$, which is explicitly fit to data containing samples from $\hat{q}_{\text{cc}}^{(j)}$, and thus performs better than $\hat{q}_{\text{model}}$ on $(\mathbf{U}_j^{(m)}, \mathbf{Y}')$. Additionally, the inclusion of $\hat{q}_{\text{null}}^{(j)}$ in $\hat{q}_{\text{mix}}^{(j)}$ will also result in worse performance than $\hat{q}_{\text{model}}$ on $(\mathbf{X}', \mathbf{Y}')$. Consequently, the indicator function in the CONTRA p-value computation (4.3) will be 1 with greater probability than the inner term of the HRT p-value (4.2) across datasets $(\mathbf{X}', \mathbf{Y}', \mathbf{U}_j^{(m)})$.

A further advantage of using $\hat{q}_{\text{mix}}^{(j)}$ over $\hat{q}_{\text{model}}$ in practice is observed when the supports of $\hat{q}_{\text{cc}}^{(j)}$ and $q(\mathbf{x}_j \mid \mathbf{x}_{-j})$ do not match. In such cases, the log-likelihood of $\hat{q}_{\text{model}}$ is not well-defined, while the log-likelihood of $\hat{q}_{\text{mix}}^{(j)}$ is well-defined. This means CONTRA restricts the amount p-values can be deflated and leads to better FDR in practice, as we show in our experiments. This relates to the

theoretical analysis of Barber et al. [2020], who show that the empirical KL between $q(\mathbf{x}_j \mid \mathbf{x}_{-j})$ and $\hat{q}_{\text{cc}}^{(j)}$ bounds the FDR in the case of knockoffs. We discuss this analysis next in section 4.2.4.

4.2.4 Bounding FDR with KL

Requirements for FDR in the case of knockoffs. The knockoff filter requires that the null variables $\widetilde{\mathbf{X}}$ are exact. That is, the jth column of the ith sample $\widetilde{\mathbf{x}}_j^{(i)}$ must be drawn from the population distribution $q(\mathbf{x}_j \mid \mathbf{x}_{-j} = \mathbf{x}_{-j}^{(i)})$, where $\mathbf{x}_{-j}^{(i)}$ is from the ith sample of $\mathbf{X}$. This is required to ensure that under the null hypothesis, $Z_j(\mathbf{X}, \mathbf{Y})$ and $Z_j(\mathbf{U}_j, \mathbf{Y})$ will have the same distribution, meaning W_j is symmetric around 0. This property of W_j is termed the "flip-sign" property. Candes et al. [2018] show that when W_j satisfies the flip-sign property, the knockoff filter can control the FDR.

Issues with knockoffs in practice. Barber et al. [2020] acknowledge that practitioners seldom have access to the population distribution $q(\mathbf{x}_j \mid \mathbf{x}_{-j})$, and must resort to estimators $\hat{q}_{\text{cc}}^{(j)}(\mathbf{x}_j \mid \mathbf{x}_{-j})$. If the population and estimated distributions are not the same, the knockoff filter may inflate the FDR beyond nominal levels. The authors show that the level to which the knockoff filter inflates the FDR is bounded by the maximum empirical KL between each $\hat{q}_{\text{cc}}^{(j)}(\mathbf{x}_j \mid \mathbf{x}_{-j})$ and $q(\mathbf{x}_j \mid \mathbf{x}_{-j})$. They do so by relating the empirical KL to the flip-sign property. Let E_j be the empirical KL between $\hat{q}_{\text{cc}}^{(j)}(\mathbf{x}_j \mid \mathbf{x}_{-j})$ and $q(\mathbf{x}_j \mid \mathbf{x}_{-j})$. Then for any covariate $\mathbf{x}_j$ whose null hypothesis is true,

$$\frac{\mathbb{P}(W_j > 0, E_j \leq \epsilon \mid |W_j|, W_{-j})}{\mathbb{P}(W_j < 0 \mid |W_j|, W_{-j})} \leq \exp(\epsilon) \qquad \forall \epsilon \geq 0. \tag{4.4}$$

We refer the reader to Barber et al. [2020] for a detailed derivation and discussion of this equation. The main takeaway is that as E_j approaches 0, the probability that W_j is positive is equal to the probability that it is negative, thus satisfying the flip-sign property required to control the FDR.

Relation of flip-sign result to CRTs. We can show how the flip-sign result of Barber et al. [2020] relates to CRTs by using the fact that W_j being symmetric around 0 implies that $Z_j(\mathbf{X}, \mathbf{Y})$ and $Z_j(\mathbf{U}_j, \mathbf{Y})$ have the same distribution. A p-value using $Z_j(\mathbf{X}, \mathbf{Y})$ and $Z_j(\mathbf{U}_j, \mathbf{Y})$ by performing the following computation. Letting each $\mathbf{U}_j^{(m)}$ be drawn independently the same way as $\mathbf{U}_j$,

$$p_j = \frac{1}{M+1}\left(1 + \sum_{m=1}^{M} \mathbb{1}(Z_j(\mathbf{X}, \mathbf{Y}) \geq Z_j(\mathbf{U}_j^{(m)}, \mathbf{Y}))\right).$$

Note that for the choice of $Z_j = \mathcal{L}$, the empirical risk of a model, this p-value computation is exactly the p-value computation from eq. (4.2). Recall our proof of uniform null p-values for CONTRA from prop. 4.2.1. If $Z_j(\mathbf{X}, \mathbf{Y})$ and $Z_j(\mathbf{U}_j, \mathbf{Y})$ are equal in distribution, p_j will be uniform under the null. Thus, the closer W_j is to satisfying the flip-sign property, the closer null p-values are to being uniformly distributed.

If the empirical KL terms E_j are greater than 0, then the left hand side of eq. (4.4) will not be bounded, meaning W_j could have a positive or negative bias. As we discuss in the main text, when using the empirical risk of a model as Z_j, the bias of W_j tends to be negative, as the models exhibit higher losses on out-of-distribution data. A negative bias leads to deflated p-values, and ultimately inflated FDR.

Support mismatch between $q(\mathbf{x}_j \mid \mathbf{x}_{-j})$ and $\hat{q}_{\text{cc}}^{(j)}$. When $q(\mathbf{x}_j \mid \mathbf{x}_{-j})$ and $\hat{q}_{\text{cc}}^{(j)}$ have mismatched supports, E_j used in the Barber et al. [2020] proof above can be ∞ if $\hat{q}_{\text{cc}}^{(j)}$ puts no mass where $q(\mathbf{x}_j \mid \mathbf{x}_{-j})$ puts non-zero mass. This means that the bound in eq. (4.4) can be vacuous and lead to highly non-uniform p-values. We will illustrate an example where HRTs realize this bound and yield low p-values even in the case of null covariates.

Consider the case of HRTs that use the most powerful test statistic, as shown by Katsevich and Ramdas [2020]: the log-likelihood of $\hat{q}_{\text{model}}(\mathbf{y} \mid \mathbf{x})$. Let $\mathbf{x}_j$ be a null covariate. The model $\hat{q}_{\text{model}}$ is fit

to the data $(\mathbf{X}, \mathbf{Y})$ and its log-likelihood is well-defined; $\log \hat{q}_{\text{model}}(\mathbf{y} = \mathbf{y}^{(i)} \mid \mathbf{x} = \mathbf{x}^{(i)})$ will be in $(-\infty, 0]$ if $\mathbf{x}^{(i)} \sim q(\mathbf{x})$ and $\mathbf{y}^{(i)} \sim q(\mathbf{y} \mid \mathbf{x} = \mathbf{x}^{(i)})$. Assume that there is high dependence between $\mathbf{x}_j$ and some non-null $\mathbf{x}_k$, so $\hat{q}_{\text{model}}$ exhibits spurious dependence on $\mathbf{x}_j$. If $\hat{q}_{\text{model}}$ is evaluated on a sample $(\widetilde{\mathbf{x}}_j^{(i)}, \mathbf{x}_{-j}^{(i)}, \mathbf{y}^{(i)})$ where $\widetilde{\mathbf{x}}_j^{(i)} \sim \hat{q}_{\text{cc}}^{(j)}$ is not in the support of $q(\mathbf{x}_j \mid \mathbf{x}_{-j} = \mathbf{x}_{-j}^{(i)})$, then $\hat{q}_{\text{model}}(\mathbf{y} = \mathbf{y}^{(i)} \mid \mathbf{x}_j = \widetilde{\mathbf{x}}_j^{(i)}, \mathbf{x}_{-j} = \mathbf{x}_{-j}^{(i)})$ can be arbitrarily small. This means that $\mathcal{L}(\mathbf{U}_j^{(m)}, \mathbf{Y})$ can be arbitrarily large if even a single sample $(\widetilde{\mathbf{x}}_j^{(i)}, \mathbf{x}_{-j}^{(i)}, \mathbf{y}^{(i)}) \in \mathbf{U}_j^{(m)}$ contains coordinate $\widetilde{\mathbf{x}}_j^{(i)}$ that is out of support for $q(\mathbf{x}_j \mid \mathbf{x}_{-j} = \mathbf{x}_{-j}^{(i)})$. The more likely it is to draw a sample from $\hat{q}_{\text{cc}}^{(j)}$ that results in such an $\mathcal{L}(\mathbf{U}_j^{(m)}, \mathbf{Y})$, the more like the p-value will be 0. A realistic example of such cases is when $\hat{q}_{\text{cc}}^{(j)}$ captures only a single mode of $q(\mathbf{x}_j \mid \mathbf{x}_{-j})$. This leads to null features being selected, inflating the FDR.

Contrast this scenario with CONTRA instead of an HRT. Recall that the log-likelihood of $\hat{q}_{\text{mix}}^{(j)}$, which consists of an equal mixture of $\hat{q}_{\text{null}}^{(j)}$ and $\hat{q}_{\text{model}}$, is always well-defined. Then $\mathcal{L}(\mathbf{X}, \mathbf{Y})$ will be no greater than

$$\sum_{i=1}^{n_{\text{test}}} \log \frac{1}{2} \hat{q}_{\text{model}}(\mathbf{y} = \mathbf{y}^{(i)} \mid \mathbf{x} = \mathbf{x}^{(i)}),$$

and $\mathcal{L}(\mathbf{U}_j^{(m)}, \mathbf{Y})$ will be no less than

$$\sum_{i=1}^{n_{\text{test}}} \log \frac{1}{2} \hat{q}_{\text{null}}^{(j)}(\mathbf{y} = \mathbf{y}^{(i)} \mid \mathbf{x}_j = \widetilde{\mathbf{x}}_j^{(i)}, \mathbf{x}_{-j} = \mathbf{x}_{-j}^{(i)}).$$

This is because even if $\hat{q}_{\text{model}}(\mathbf{y} = \mathbf{y}^{(i)} \mid \mathbf{x}_j = \widetilde{\mathbf{x}}_j^{(i)}, \mathbf{x}_{-j} = \mathbf{x}_{-j}^{(i)}) = 0$, the term $\hat{q}_{\text{null}}^{(j)}(\mathbf{y} = \mathbf{y}^{(i)} \mid \mathbf{x}_j = \widetilde{\mathbf{x}}_j^{(i)}, \mathbf{x}_{-j} = \mathbf{x}_{-j}^{(i)})$ will not be 0 as $\hat{q}_{\text{null}}^{(j)}$ is fit to samples from $\hat{q}_{\text{cc}}^{(j)}$. This means that the log-probability of the mixture will be in $(-\infty, 0]$ and is therefore well-defined.

A consequence of the behavior of HRTs and CONTRA in these scenarios is that HRTs can arbitrarily deflate p-values. CONTRA alleviates this issue as it restricts the amount p-values can be deflated. As we show in our experiments, this leads to better FDR in practice.

4.2.5 Computational efficiency

CONTRA requires an estimator $\hat{q}_{\text{model}}(\mathbf{y} \mid \mathbf{x})$ for $q(\mathbf{y} \mid \mathbf{x})$ fit using training data $(\mathbf{X}, \mathbf{Y})$. For each covariate $\mathbf{x}_j$, it also requires a conditional model $\hat{q}_{\text{cc}}^{(j)}(\mathbf{x}_j \mid \mathbf{x}_{-j})$, and a single null model $\hat{q}_{\text{null}}^{(j)}(\mathbf{y} \mid \widetilde{\mathbf{x}}_j, \mathbf{x}_{-j})$ fit using $(\mathbf{U}_j, \mathbf{Y})$, where $\mathbf{U}_j$ is a copy of $\mathbf{X}$, but with the jth column replaced with samples from $\hat{q}_{\text{cc}}^{(j)}$. This means there are $2d + 1$ models fit in total.

To compute p-values using these models and a test set $(\mathbf{X}', \mathbf{Y}')$, M null datasets $\{\widetilde{\mathbf{X}}'^{(m)}\}_{m=1}^{M}$ must be sampled. For each covariate, $\hat{q}_{\text{mix}}$ must be evaluated on the test sets to compute loss $\ell^{(j)}$. This results in a total of $2d \cdot M$ model evaluations, as there are M null replications for each of the d covariates, and $\hat{q}_{\text{mix}}^{(j)}$ consists of both $\hat{q}_{\text{model}}$ and $\hat{q}_{\text{null}}^{(j)}$. It is worthy to note, in addition, that the computations required for the jth covariate are independent of those required for all other covariates, making CONTRA embarrassingly parallel.

In comparison to CONTRA, HRTs still need to fit $d + 1$ models ($\hat{q}_{\text{model}}$ and $\{\hat{q}_{\text{cc}}^{(j)}\}_{j=1}^{d}$), and also sample M null datasets. However, since the HRT loss only involves $\hat{q}_{\text{model}}$, a total of $d \cdot M$ model evaluations on the test sets are required.

Thus, CONTRA is able to lessen the FDR compared to HRTs when $\hat{q}_{\text{cc}}^{(j)} \neq q(\mathbf{x}_j \mid \mathbf{x}_{-j})$ at the cost of only a constant factor increase in the number of models fit and evaluated. This makes CONTRA a compelling method in practice.

4.3 Experiments

We analyze the performance of CONTRA on several synthetic and real datasets and compare it to several well-studied CVS baselines.

Baselines. We compare CONTRA to popular CRT-based CVS methods. Recall that the $\hat{q}_{\text{model}}$-based CRT statistic discussed by Liu and Janson [2020] requires $\mathcal{O}(M)$ models to be fit for *every* covariate [Tansey et al. 2018a]. This makes it highly impractical to use with model-based test statistics as

discussed in this chapter. As a result, we use CRTs with the computationally efficient marginal correlation statistic, which involves a p-value computation eq. (4.2) using

$$\mathcal{L}(\mathbf{X}', \mathbf{Y}') = \sum_{i=1}^{n_{\text{test}}} (\mathbf{x}_j^{(i)} - \bar{\mathbf{x}}_j)(\mathbf{y}^{(i)} - \bar{\mathbf{y}}),$$

where $\bar{\mathbf{x}}_j$ and $\bar{\mathbf{y}}$ are the sample averages of $\mathbf{x}_j$ and $\mathbf{y}$ respectively computed from the training set $(\mathbf{X}, \mathbf{Y})$. We term this the CORR-CRT. For HRTs, we use two different model-based statistics:

$$\mathcal{L}_1(\mathbf{X}', \mathbf{Y}') = \sum_{i=1}^{n_{\text{test}}} -\log \hat{q}_{\text{model}}(\mathbf{y} = \mathbf{y}^{(i)} \mid \mathbf{x} = \mathbf{x}^{(i)})$$

$$\mathcal{L}_2(\mathbf{X}', \mathbf{Y}') = \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \mathbb{1}\{\mathbf{y}^{(i)} \neq \hat{\mathbf{y}}^{(i)})\}$$

$$\hat{\mathbf{y}}^{(i)} \sim \hat{q}_{\text{model}}(\mathbf{y} \mid \mathbf{x} = \mathbf{x}^{(i)})$$

The statistic $\mathcal{L}_1$, termed the LL-HRT, is the negative log-likelihood of the test set using $\hat{q}_{\text{model}}$. The statistic $\mathcal{L}_2$, termed the 01-HRT, measures the misclassification rate of $\hat{q}_{\text{model}}$ on the test set when $\mathbf{y}$ is a discrete random variable. We exclude comparisons to calibrated HRTs, as they take many times as long to run. Fitting at least 100 $\hat{q}_{\text{cc}}$ models for every covariate, as suggested by code from Tansey et al. [2018b], proved to be significantly slower than other CVS methods for the synthetic experiments, and computationally infeasible for a high-dimensional genomics task.

4.3.1 Synthetic data experiments

Each experiment involving a synthetic dataset uses the following setup. First, we generate the training dataset $(\mathbf{X}, \mathbf{Y})$ of n_{train} samples and a held-out test set $(\mathbf{X}', \mathbf{Y}')$ of n_{test} samples from data distribution $q(\mathbf{x}, \mathbf{y})$. Each sample of covariates $\mathbf{x}^{(i)} \in \mathbb{R}^d$, and the responses $\mathbf{y}^{(i)} \in \{0, 1\}$.

Next, we create d conditional models: one $\hat{q}_{\text{cc}}^{(j)}(\mathbf{x}_j \mid \mathbf{x}_{-j})$ for each $j \in \{1, \ldots, d\}$. Since we need to be able to sample from each $\hat{q}_{\text{cc}}^{(j)}$, we implement neural histogram estimators [Miscouridou

et al. 2018], which are flexible approximations to conditional densities. Each $\hat{q}_{\text{cc}}^{(j)}$ is a two-layer fully connected networks with 32 units in each layer, and a softmax output with K classes. To fit $\hat{q}_{\text{cc}}^{(j)}$, we first bin the jth column of $\mathbf{X}$ by value into K bins, then fit the neural network to predict the bin of $\mathbf{x}_j^{(i)}$ given $\mathbf{x}_{-j}^{(i)}$. Each neural network is trained with the cross-entropy loss using SGD. In our experiments, we use $K = 20$. 18 of the bins in $\hat{q}_{\text{cc}}^{(j)}$ are uniformly spaced between the 5th and 95th quantiles of each $\mathbf{x}_j$. The remaining two bins represent any samples below the 5th quantile, or above the 95th quantile. To generate samples from $\hat{q}_{\text{cc}}^{(j)}$, we use the median value of training samples in the bin that corresponds to the network's prediction given $\mathbf{x}_{-j}^{(i)}$. These models are used to generate $M + 1$ null datasets $\widetilde{\mathbf{X}}$ and $\{\widetilde{\mathbf{X}}'^{(m)}\}_{m=1}^{M}$, where $\widetilde{\mathbf{X}}$ is generated conditional on $\mathbf{X}$, and each $\widetilde{\mathbf{X}}'^{(m)}$ is generated conditional on $\mathbf{X}'$. In each of our synthetic experiments, we set M to 100, unless otherwise specified.

For each of $\hat{q}_{\text{model}}$ and $\hat{q}_{\text{null}}$, we use random forests with 100 trees fit to the training set. In general, we suggest using the model, parametric or nonparametric, that performs best on a validation split of $(\mathbf{X}, \mathbf{Y})$ for high power.

Finally, we compute p-values for each of CONTRA, CORR-CRT, LL-HRT, and 01-HRT. A p-value threshold is obtained using the Benjamini and Hochberg [1995] procedure to select important covariates at a pre-specified FDR. We run each experiment on a 16-core CPU with 64GB of memory.

BENCHMARK DATASETS. Our tests on four different synthetic datasets highlight differences between each CVS approach. Datasets in this section consists of $N = 2000$ samples, and $d = 20$ covariates, unless otherwise specified. We use 70% of the data as a training set to fit each $\hat{q}_{\text{cc}}^{(j)}$, $\hat{q}_{\text{model}}$, and $\hat{q}_{\text{null}}^{(j)}$. We use the remaining 30% to compute p-values.

[`orng`, `orng-c`]: As a first example, we test the case where $\mathbf{y}$ is a nonlinear function of $\mathbf{x}$, we use the `orng` and `orng-c` datasets [Chen et al. 2018]. The data is generated in the following

manner:

$$\mathbf{x} \sim \mathcal{N}(0, \Sigma)$$
$$\mathbf{y} = \begin{cases} 1 & \text{if } \exp\left(\sum_{j=1}^{\ell} \mathbf{x}_j^2 - \ell\right) > 0.5 \\ 0 & \text{otherwise} \end{cases}$$

where Σ is the 20-dimensional identity in the case of `orng`. For `orng-c`, we set all off-diagonals to 0.2, and set diagonal values to 1. The variable ℓ controls the number of important covariates, which we set to 4 for both of these experiments.

[`xor`, `xor-c`]: The choice of test statistic can impact power when covariates on their own are not informative but together provide information. To explore this, we design the `xor` and `xor-c` datasets. For `xor` and `xor-c`, we first sample $\mathbf{x}$ in the same way as `orng` and `orng-c` respectively. An affine transformation is then applied to each sample, and $\mathbf{y}$ is generated in the following manner:

$$s_1, s_2 \sim 4 \cdot \text{Rademacher}(0.5)$$
$$(\mathbf{x}_1, \mathbf{x}_2) \leftarrow (\mathbf{x}_1 + s_1, \mathbf{x}_2 + s_2)$$
$$\mathbf{y} = \begin{cases} 0 & \text{if } s_1 s_2 < 0 \\ 1 & \text{if } s_1 s_2 > 0 \end{cases}.$$

Only the first two covariates $\mathbf{x}_1$ and $\mathbf{x}_2$ are in the Markov blanket of $\mathbf{y}$.

Selection results. For each synthetic benchmark and CVS method, we run 100 experiments as described earlier to obtain p-values for each covariate. In order to concisely summarize the performance of each CVS method, we compute the FCAUC [Yu 2012], which compute the area under a receiver operating characteristic (ROC) curve, but only up to a realistic nominal FDR.

Dataset	orng	orng-c	xor	xor-c
CONTRA	**0.95**	**1.00**	**0.97**	**0.95**
01-HRT	0.94	0.94	0.95	0.92
LL-HRT	**0.95**	0.95	0.95	0.93
CORR-CRT	0.22	0.35	0.45	0.38

Table 4.1: CONTRA achieves highest FCAUC ratios on synthetic data benchmarks. (Scores closer to 1 are better). While both CONTRA and the HRTs achieve similar power, the HRTs achieve worse FDP, yielding lower FCAUC ratios.

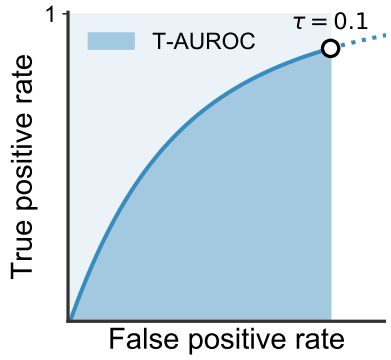


Figure 4.1: FCAUC ratio: ratio of dark blue area to all blue areas.

For example, practitioners are unlikely to be interested in controlling FDR at rates greater than 50%. To compute an FCAUC score, we first measure the true positive rate (TPR) (also known as power) and false positive rate (FPR) at every p-value threshold to compute a ROC curve. We then identify a nominal p-value threshold τ that corresponds to an FDR of 10% using the Benjamini and Hochberg [1995] procedure. Using the ROC curve, we compute two quantities: (A) the area under this curve from 0 to $\text{FPR}(\tau)$ (the FCAUC), and (B) the area of the rectangle defined by $(0, 0)$ and $(\text{FPR}(\tau), 1)$, where $\text{FPR}(\tau)$ is the FPR corresponding to threshold τ (see fig. 4.1 for illustration). The score we assign to each CVS method is the ratio of (A) to (B): the FCAUC ratio. Intuitively, the closer this score is to 1, the higher the performance of a CVS method. Table 4.1 shows the average of this score for every CVS method and dataset across each of the 100 runs. Standard errors are omitted from table 4.1 as they are each fewer than four decimal places.

CONTRA achieves a higher FCAUC ratio than competing baselines. At a nominal FDR rate of

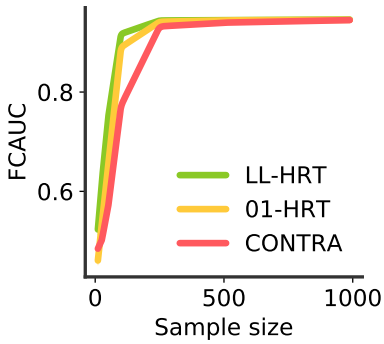


Figure 4.2: The loss in power due to the use of contarian models is reduced as sample size increases. Apart from very low sample sizes, CONTRA achieves power on par with both HRTs.

10%, the HRT methods tend to exhibit FDPs[1] of 15% or more, while CONTRA maintains the FDP at or below 10%. It is worth noting that this difference in FDP is the main driver of CONTRA's higher performance in table 4.1. Both the HRT methods achieve power equal to that of CONTRA.

We further observe that the CORR-CRT performs noticeably worse than the other methods. This is likely due to its inability to model interactions between covariates when computing the test statistic, resulting in low power.

Table 4.1 shows promising results, as it suggests that despite using contrarian model $\hat{q}^{(j)}_{\text{mix}}$, CONTRA suffers no loss in power compared to the baselines on these four benchmarks.

To understand the power lost due to contrarian models, we repeat the `orng` experiment at different sample sizes. We only compare CONTRA to each of the HRTs, as the power of CORR-CRT is much lower even at large sample sizes, as discussed in the experiments section. Figure 4.2 plots the average FCAUC for each method over all 100 replicates as a function of training sample size. The test set is equal to the training set in size. Error bars are omitted due to very small standard error.

We note there is a noticeable gap in FCAUC by CONTRA at low sample sizes (≤ 200), but minimal difference otherwise. The loss in power due to contrarian models is minimized as sample size increases. At just 500 samples, the power of CONTRA is almost equal to that of the HRTs.

[1] Another term for empirical FDR.

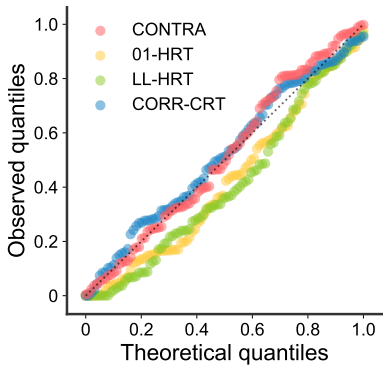


Figure 4.3: CONTRA exhibits null p-values that are well calibrated.

Having observed that the main difference between CONTRA and the baselines is primarily the control of FDR, we next explore questions that help further understand the useful FDR properties of CONTRA.

HOW DOES THE CHOICE OF CVS METHOD AFFECT p-VALUE CALIBRATION? The effectiveness of CVS methods to control the FDR is greatly reduced when null p-values are not super-uniform Benjamini and Hochberg [1995]; Benjamini and Yekutieli [2001]. For FDR to be controlled effectively, null p-values must stochastically dominate a Uniform(0,1) random variable. In this experiment, we specifically look at how well p-values produced by each CVS method satisfy this requirement for FDR control. We again use `orng-c`, but with one modification: we increase the number of null covariates from 16 to 100. We then perform a Kolmogorov-Smirnov hypothesis test using the set of null p-values from each CVS method. This quantifies how uniform the null p-values are.

Figure 4.3 shows a quantile-quantile plot of the null p-values of each method. The closer the points match the dotted black diagonal, the closer the null p-values are to Uniform(0, 1). We first notice that both CONTRA and CORR-CRT are well calibrated with Kolmogorov-Smirnov p-values of 0.183 and 0.526 respectively. However, LL-HRT and 01-HRT yield Kolmogorov-Smirnov p-values of 0.009 and 0.005 respectively. At a type-1 error threshold of 1%, both HRTs appear to yield significantly non-uniform p-values, suggesting that HRT procedures may not control the FDR well using standard multiple correction techniques. Upon closer inspection, we observe this

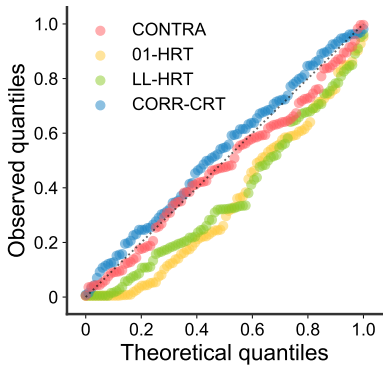


Figure 4.4: Despite null variable misspecification, CONTRA maintains FDR control.

issue as $\hat{q}_{\text{model}}$ tends to exhibit dependence on null covariates, and each $\hat{q}_{\text{cc}}^{(j)}$ is not exactly equal to the corresponding $q(\mathbf{x}_j \mid \mathbf{x}_{-j})$. As a result, HRT test statistics tend to overestimate the importance of the null covariates, and underestimate null p-values. This is seen in fig. 4.3, as the observed quantiles are below the theoretical quantiles, highlighting the deflationary behavior of the null p-values. Using contrarian models protects against this behavior, as does not using a model at all in the case of CORR-CRT.

WHAT IF $\hat{q}_{\text{cc}}$ IS MODELED INCORRECTLY? In this section, we investigate the effect of modeling the null variables incorrectly on null p-values. To generate covariates, we use a mixture of autoregressive Gaussians. This provides a more challenging benchmark as each covariate is multi-modal and highly correlated with several others, encouraging $\hat{q}_{\text{model}}$ to learn spurious dependencies.

We sample $\mathbf{x} \sim \sum_{k=1}^{K} \pi_k \mathcal{N}(\mu_k \cdot \mathbf{1}, \Sigma_k)$, where each Σ_k is a 104-dimensional covariance matrix whose (i, j)th entry is $\rho_k^{|i-j|}$, and $\mathbf{1}$ is a 104-dimensional 1's vector. We set $K = 3$, and $(\rho_1, \rho_2, \rho_3) = (0.6, 0.4, 0.2)$. Cluster centers are set to $(\mu_1, \mu_2, \mu_3) = (0, 5, 10)$, and mixture proportions are set to $(\pi_1, \pi_2, \pi_3) = (0.4, 0.2, 0.4)$. We model all $\hat{q}_{\text{cc}}^{(j)}$ jointly with a multivariate normal (MVN) distribution. For visualization, we show two adjacent dimensions of the data and the maximum likelihood estimation (MLE) solution for the MVN in fig. 4.5.

We sample $\mathbf{y}$ in the same way as `orng`, using only the first four covariates as non-null. For this experiment, we set the number of null resamples M to 200.

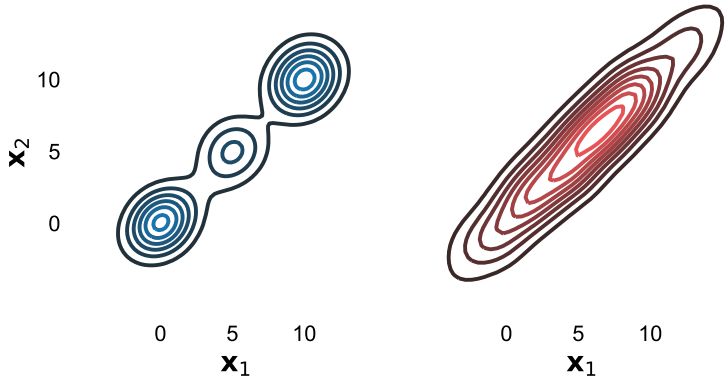


Figure 4.5: Data distribution: mixture of correlated Gaussians (left); Model distribution: MLE solution for multivariate Gaussian fit to data (right). Covariates $\mathbf{x}_1$ and $\mathbf{x}_2$ are visualized.

We run each CVS method on the data, and perform Kolmogorov-Smirnov hypothesis tests on the null p-values. We do not discuss the power of each method in this section, as all CVS methods other than CORR-CRT exhibit power 1 for any nominal FDR threshold above 5%. Figure 4.4 visualizes the null p-values for each CVS method. We observe that CONTRA and CORR-CRT both produce null p-values that appear uniform (Kolmogorov-Smirnov p-values of 0.684 and 0.399 respectively). The LL-HRT and 01-HRT produce p-values that appear to be stochastically dominated by a Uniform$(0, 1)$ random variable (Kolmogorov-Smirnov p-values of 1.059×10^{-5} and 9.342×10^{-6} respectively).

We further notice that in the range $[0, 0.15]$ on the x-axis, the HRT methods yield several p-values of $1/201$, the minimum possible given our setup. Upon closer investigation, we report the following observations that explain why this p-value deflation occurs. First, $\hat{q}_{\text{model}}$ is found to exhibit spurious dependence on null covariates that correlate highly with one of $\{\mathbf{x}_i\}_{i=1}^4$. Second, the mixture distribution has low support on covariates in the neighborhood around $(5, 5)$, while the $\hat{q}_{\text{cc}}^{(j)}$ models place considerable mass around this point. As a result, $\hat{q}_{\text{model}}$ is evaluated on data out of its support, and consistently exhibits higher losses on the null data $(\mathbf{U}_j^{(m)}, \mathbf{Y}')$ than on the test set $(\mathbf{X}', \mathbf{Y}')$, even when computing null p-values. Thus, the null p-values tend to be stochastically dominated by a Uniform$(0, 1)$ random variable and lead to the inflation of FDR.

	# Selected	Precision	Recall	Time (s)
CONTRA	12	66.67%	20%	9207
01-HRT	15	53.33%	20%	8871
LL-HRT	14	57.14%	20%	8912
CORR-CRT	118	5.08%	15%	1512

Table 4.2: CONTRA achieves power on par with state-of-the-art CVS methods while achieving higher precision. Here we compare CVS methods on their ability to identify biologically relevant SNPs for Celiac disease.

4.3.2 CELIAC DISEASE EXPERIMENT

Abnormalities in the genome of an individual have been found to associate with Celiac disease [Dubois et al. 2010]. To understand how well CVS methods are able to replicate the results of biological studies in a purely computational procedure, we study a large genetics dataset. We apply each CVS method to a large (cases = 3.7K, controls = 8.2K) Celiac disease dataset [Dubois et al. 2010].

In our dataset, the covariates $\mathbf{x}_j \in \{0, 1, 2\}$ represent SNPs, which measure the genetic variance for each individual with respect to a reference genome. The response $\mathbf{y}$ is a binary label indicating the presence of Celiac disease. We preprocess the data as suggested by Bush and Moore [2012]. First, the set of SNPs is preprocessed using linkage-disequilibrium pruning [Calus and Vandenplas 2018], a commonly used procedure in genomics to filter out redundant SNPs using pairwise correlation. The total number of SNPs after filtering is 1759. Then, genetic principal components are added as covariates[2] to $\hat{q}_{\text{model}}$ (and $\hat{q}_{\text{null}}$) to correct for population biases [Price et al. 2006]. To model $\hat{q}_{\text{cc}}$, we use the same approach as Candes et al. [2018], which uses $\hat{q}_{\text{cc}}^{(j)}$ models that condition only on a subset of SNPs in a neighborhood around $\mathbf{x}_j$, rather than all other SNPs. For exact implementation details, we refer the reader to section 7 of Candes et al. [2018]. Finally, we use L_1-penalized logistic regression for $\hat{q}_{\text{model}}$ and $\hat{q}_{\text{null}}$, and set the number of null replicates M to 500.

[2]These covariates are not tested or modeled using $\hat{q}_{\text{cc}}$.

SELECTION RESULTS. After running each CVS procedure on the data, we select important SNPs using a 5% FDR threshold. Using the list of SNPs returned by each method, we compare each one to the genetics literature. Specifically, we determine which SNPs have been shown to map to immunological pathways responsible for the development of Celiac disease Dubois et al. [2010]; Sollid [2002]; Adamovic et al. [2008]; Hunt et al. [2008]. If an identified SNP has been mentioned by one of these studies, we deem it important.

Table 4.2 shows that while CONTRA and the HRTs achieve the same recall, CONTRA achieves a higher precision (which is 1 - FDR). CORR-CRT fails to account for dependence between SNPs and tends to overestimate the variance of a single covariate, which leads to many false discoveries.

Finally, we time CONTRA and the HRTs and note that despite the high dimensionality of the problem and large M, CONTRA is only 5 minutes slower due to the fitting of $\hat{q}_{\text{null}}$ models (shown in table 4.2).

4.4 DISCUSSION

CVS procedures like the HRT are popular for their ability to control the FDR. However, they can deflate p-values when the covariate distribution is unknown, thus violating FDR control. CONTRA is designed specifically for situations where the covariate distribution must be estimated from data. CONTRA is able to control FDR in finite samples, and remarkably, achieves power 1 in the limit of data despite the use of contrarian models that yield more conservative p-values than HRTs. CONTRA exhibits state-of-the-art power on several synthetic and real benchmarks, while maintaining FDR at levels closer to the nominal rate than competing baselines.

5 | Decoupled independence tests

CRTs assess whether a variable $\mathbf{x}$ is predictive of another variable $\mathbf{y}$, having observed covariates $\mathbf{z}$. CRTs require fitting a large number of predictive models, which is often computationally intractable. Existing solutions to reduce the cost of CRTs typically split the dataset into a train and test portion, or rely on heuristics for interactions, both of which lead to a loss in power. We propose the DIET, an algorithm that avoids both of these issues by leveraging marginal independence statistics to test conditional independence relationships. DIET tests the marginal independence of two random variables: $F(\mathbf{x} \mid \mathbf{z})$ and $F(\mathbf{y} \mid \mathbf{z})$ where $F(\cdot \mid \mathbf{z})$ is a conditional CDF. These variables are termed "information residuals." We give sufficient conditions for DIET to achieve finite sample type-1 error control and power greater than the type-1 error rate. We then prove that when using the mutual information between the information residuals as a test statistic, DIET yields the most powerful conditionally valid test. Finally, we show DIET achieves higher power than other tractable CRTs on several synthetic and real benchmarks.

5.1 Motivation for DIET

A key question in many scientific disciplines is whether a variable $\mathbf{x}$ causes some outcome $\mathbf{y}$ [Lauritzen 1996; Pearl 2009]. In genetics for example, scientists test whether a particular gene causes cancer to design targeted therapies [Zhu et al. 2018]. When there are confounders $\mathbf{z}$ that may affect both $\mathbf{x}$ and $\mathbf{y}$, assessing the causal link between $\mathbf{x}$ and $\mathbf{y}$ corresponds to testing the

conditional independence (CI) between $\mathbf{x}$ and $\mathbf{y}$ given $\mathbf{z}$:

$$\text{null hypothesis } \mathcal{H}_0 : \mathbf{x} \perp\!\!\!\perp \mathbf{y} \mid \mathbf{z} \quad \text{vs} \quad \text{alternate hypothesis } \mathcal{H}_1 : \mathbf{x} \not\perp\!\!\!\perp \mathbf{y} \mid \mathbf{z}. \tag{5.1}$$

The advantage of using a hypothesis test for understanding such relationships is the ability to explicitly control the type-1 error rate: the probability of erroneously rejecting the null hypothesis where $\mathbf{x}$ is independent of $\mathbf{y}$ conditioned on $\mathbf{z}$. Consequently, constructing conditional independence hypothesis tests has become increasingly popular in the machine learning literature [Zhang et al. 2012; Doran et al. 2014; Sen et al. 2017; Runge 2018; Bellot and van der Schaar 2019].

Many existing tests however, have been shown to lose power when the dimensionality of $\mathbf{z}$ is high due to a reliance on kernels [Bellot and van der Schaar 2019] or fail to control the type-1 error rate when strong parametric assumptions about $p(\mathbf{y} \mid \mathbf{x}, \mathbf{z})$ are violated [Candes et al. 2018].

To test for conditional independence when $\mathbf{z}$ is high-dimensional and without making assumptions on the form of $p(\mathbf{y} \mid \mathbf{x}, \mathbf{z})$, Candes et al. [2018] proposed the conditional randomization test (CRT). The CRT calculates a p-value for eq. (5.1) by repeatedly comparing a scalar-valued test statistic $T(\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}})$ with draws from the null distribution $T(\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m)})$:

$$\frac{1}{M+1}\left(1 + \sum_{m=1}^{M} \mathbb{1}(T(\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}) \leq T(\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m)}))\right), \tag{5.2}$$

where $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$ is a set of N iid samples drawn from $p(\mathbf{x}, \mathbf{y}, \mathbf{z})$. Null samples $\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m)}$ are drawn from the distribution $p(\mathbf{z}, \mathbf{y})p(\mathbf{x} \mid \mathbf{z})$, where $\widetilde{\mathbf{x}} \sim p(\mathbf{x} \mid \mathbf{z})$ is by construction conditionally independent of $\mathbf{y}$ given $\mathbf{z}$. If the null hypothesis is true, then $T(\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}})$ will have the same distribution as each $T(\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m)})$.

In contrast with other conditional independence testing methods, the CRT assumes the ability to sample $p(\mathbf{x} \mid \mathbf{z})$ but makes no assumptions on the form of $p(\mathbf{y} \mid \mathbf{x}, \mathbf{z})$ or the test statistic T to control the type-1 error. This flexibility enables the use of powerful predictive models and

empirical risk test statistics [Tansey et al. 2018a; Liu and Janson 2020; Sudarshan et al. 2021] that lead to higher power and better type-1 error rates than classical methods.

However, CRTs are computationally expensive. For each null sample, the test statistic must be recomputed. When using predictive models in empirical risk test statistics, these models must correspondingly be refit for *every* null sample $\mathcal{D}_{\tilde{\mathbf{x}},\mathbf{y},\mathbf{z}}^{(m)}$. When the predictive models are computationally expensive to train, such as deep neural networks, the burden of running a CRT can become prohibitive.

5.2 Related work

Recent work in the Model-X space focuses on creating powerful but tractable CRT test statistics. Liu and Janson [2020] propose a pair of methods called distilled conditional randomization tests (DCRTs). The first method, the d_0-CRT constructs a CRT where the test statistic is the marginal dependence between $(\mathbf{y} - \mathbb{E}[\mathbf{y} \mid \mathbf{z}])$ and $(\mathbf{x} - \mathbb{E}[\mathbf{x} \mid \mathbf{z}])$. However, Liu and Janson [2020] demonstrate empirically that the d_0-CRT achieves low power when $\mathbf{y}$ is a function of some non-linear interaction between $\mathbf{x}$ and $\mathbf{z}$. To account for this issue, the authors also introduce the d_I-CRT. The d_I-CRT first uses a heuristic to select a small subset of $\mathbf{z}$ to explicitly construct a set of interaction terms with $\mathbf{x}$. It then fits a model $\hat{q}_{d_I}$ to estimate the conditional expectation of $\mathbf{y}$ given $(\mathbf{x} - \mathbb{E}[\mathbf{x} \mid \mathbf{z}]), \mathbb{E}[\mathbf{y} \mid \mathbf{z}]$, and each of the interaction terms. The d_I-CRT test statistic is some measure of feature importance of $\mathbf{x} - \mathbb{E}[\mathbf{x} \mid \mathbf{z}]$ in $\hat{q}_{d_I}$. If the heuristic pre-selection step fails to select the interactions that occur in the data, the d_I-CRT can fail to achieve power due to its reliance on conditional expectations.

The HRT [Tansey et al. 2018a] is another tractable yet flexible CRT. It splits samples of data into train and test sets, fits a predictive model on the train set, then uses this model to run a CRT only on the test set. While the HRT does not require heuristics for nonlinear interactions between $\mathbf{x}$ and $\mathbf{z}$, it often loses power compared to DCRTs in practice due to sample splitting between the

training and test set [Liu and Janson 2020].

5.3 Decoupled independence test (DIET)

Here we introduce a novel approach to distillation to create a tractable and powerful CRT. This section details the construction of the test statistic $T(\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}})$, which measures the marginal dependence between $F(\mathbf{x} \mid \mathbf{z})$ and $F(\mathbf{y} \mid \mathbf{z})$. It then details the computation of each null statistic $T(\mathcal{D}_{\tilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m)})$. Using the test and null statistics, DIET computes a p-value for testing $\mathbf{x} \perp\!\!\!\perp \mathbf{y} \mid \mathbf{z}$.

Fitting conditional CDF estimators. DIET tests the marginal independence between the conditional CDFs $F(\mathbf{x} \mid \mathbf{z})$ and $F(\mathbf{y} \mid \mathbf{z})$. As a first step, DIET estimates these conditional CDFs with two estimators: $\hat{Q}_{\text{CDF}}(\mathbf{x} \mid \mathbf{z}; \theta)$ and $\hat{Q}_{\text{CDF}}(\mathbf{y} \mid \mathbf{z}; \eta)$. Any conditional CDF estimation technique can be used. Flexible examples include kernel-based methods [Bhattacharya and Gangopadhyay 1990], nonparametric estimators, [Li and Racine 2008], and MDNs [Bishop 1994b]. DIET uses MDNs.

An MDN learns a neural network function $g : \mathbf{z} \mapsto \{\pi_\eta(\mathbf{z})[k], \mu_\eta(\mathbf{z})[k], \sigma_\eta(\mathbf{z})[k]\}_{k=1}^K$ to map values of $\mathbf{z}$ to the parameters of a gaussian mixture with K mixture components:

$$\hat{Q}_{\text{CDF}}(\mathbf{y} \mid \mathbf{z}; \eta) = \sum_{k=1}^{K} \pi_\eta(\mathbf{z})[k] \Phi\left(\frac{\mathbf{y} - \mu_\eta(\mathbf{z})[k]}{\sigma_\eta(\mathbf{z})[k]}\right).$$

The parameters η of $\hat{Q}_{\text{CDF}}(\mathbf{y} \mid \mathbf{z}; \eta)$ are learned via maximum likelihood estimation by optimizing over $(\mathbf{y}, \mathbf{z})$ pairs in dataset $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$:

$$\arg\max_\eta \frac{1}{N} \sum_{i=1}^{N} \log \hat{q}_{\text{PDF}}(\mathbf{y} = \mathbf{y}^{(i)} \mid \mathbf{z} = \mathbf{z}^{(i)}; \eta), \tag{5.3}$$

where $\hat{q}_{\text{PDF}}$ is the conditional density implied by $\hat{Q}_{\text{CDF}}$. MDNs are useful as both the conditional CDF and density can be computed easily.

Algorithm 5.1 The decoupled independence test (DIET)

Input: Labeled dataset $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$, marginal dependence statistic ρ

Output: p-value $\hat{\mathbf{p}}$

Generate null dataset $\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}$ by replacing each $\mathbf{x}$ in $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$ with a sample from $p(\mathbf{x} \mid \mathbf{z})$ Fit $\hat{Q}_{\text{CDF}}(\mathbf{x} \mid \mathbf{z}; \theta)$ and $\hat{Q}_{\text{CDF}}(\mathbf{y} \mid \mathbf{z}; \eta)$ using $(\mathbf{x}, \mathbf{z})$ pairs and $(\mathbf{y}, \mathbf{z})$ pairs from $\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}$ Generate null datasets $\{\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m)}\}_{m=1}^{M}$

Create information residual dataset $\mathcal{D}_{\hat{\boldsymbol{\epsilon}},\hat{\boldsymbol{\delta}}}$ by evaluating both $\hat{Q}_{\text{CDF}}$ models on $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$ **for** $m \in \{1, \ldots, M\}$ **do**

  Create null information residual dataset $\mathcal{D}^{(m)}_{\hat{\boldsymbol{\epsilon}},\hat{\boldsymbol{\delta}}}$ by evaluating both $\hat{Q}_{\text{CDF}}$ models on $\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m)}$

end

$\hat{\mathbf{p}} \leftarrow \frac{1}{M+1}\left(1 + \sum_{m=1}^{M} \mathbb{1}\left[\rho(\mathcal{D}_{\hat{\boldsymbol{\epsilon}},\hat{\boldsymbol{\delta}}}) \geq \rho(\mathcal{D}^{(m)}_{\hat{\boldsymbol{\epsilon}},\hat{\boldsymbol{\delta}}})\right]\right)$

A model for $F(\mathbf{x} \mid \mathbf{z})$, $\hat{Q}_{\text{CDF}}(\mathbf{x} \mid \mathbf{z}; \theta)$, is fit similarly but instead of using pairs of $(\mathbf{x}, \mathbf{z})$ from $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$, DIET uses only $\mathbf{z}$ from $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$ and draw samples of $\widetilde{\mathbf{x}} \sim p(\mathbf{x} \mid \mathbf{z})$ for each $\mathbf{z}$ data point. Note that the distribution of $(\widetilde{\mathbf{x}}, \mathbf{z})$ is equal to that of $(\mathbf{x}, \mathbf{z})$, so evaluating $\hat{Q}_{\text{CDF}}(\mathbf{x} \mid \mathbf{z}; \theta)$ on samples of $(\mathbf{x}, \mathbf{z})$ from $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$ will still be in-distribution.

COMPUTING THE TEST STATISTIC $T(\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}})$. The DIET test statistic measures the marginal dependence between two quantities $\hat{\boldsymbol{\epsilon}}$ and $\hat{\boldsymbol{\delta}}$ using a dataset of paired samples $\mathcal{D}_{\hat{\boldsymbol{\epsilon}},\hat{\boldsymbol{\delta}}}$. The variables $\hat{\boldsymbol{\epsilon}}$ and $\hat{\boldsymbol{\delta}}$, termed "information residuals" represent the residual information contained in $\mathbf{x} \mid \mathbf{z}$ and $\mathbf{y} \mid \mathbf{z}$. They are computed as follows. A sample of $\hat{\boldsymbol{\epsilon}}$ is generated by evaluating the conditional CDF $\hat{Q}_{\text{CDF}}(\mathbf{x} \mid \mathbf{z}; \theta)$ at a sample $(\mathbf{x}, \mathbf{z})$. Similarly, $\hat{\boldsymbol{\delta}} \leftarrow \hat{Q}_{\text{CDF}}(\mathbf{y} \mid \mathbf{z}; \eta)$. To generate the dataset $\mathcal{D}_{\hat{\boldsymbol{\epsilon}},\hat{\boldsymbol{\delta}}}$, a pair of $(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}})$ samples are computed for each $(\mathbf{x}, \mathbf{y}, \mathbf{z})$ sample in $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$ using the respective conditional CDFs.

Using the dataset of information residuals $\mathcal{D}_{\hat{\boldsymbol{\epsilon}},\hat{\boldsymbol{\delta}}}$, DIET measures the marginal dependence between $\hat{\boldsymbol{\epsilon}}$ and $\hat{\boldsymbol{\delta}}$ using the estimator of mutual information from Vinh et al. [2009]. In practice, any measure of dependence $\rho : (\mathbb{R} \times \mathbb{R})^N \to \mathbb{R}$ can be used.

COMPUTING NULL STATISTICS $T(\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m)})$. Computing each null statistic is very similar to computing the test statistic. First, a null dataset $\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m)}$ is sampled by copying $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$, then replacing

the $\mathbf{x}$ values with $\widetilde{\mathbf{x}} \sim p(\mathbf{x} \mid \mathbf{z})$. The same $\hat{Q}_{\text{CDF}}$ models are used to generate information residuals using the null data, after which their mutual information is estimated. This process is repeated M times to generate M null statistics.

Computing a p-value. Using the test statistic $T(\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}})$ and each null statistic $T(\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m)})$, DIET computes a p-value using eq. (5.2). The full algorithm is summarized in algorithm 5.1.

While the DIET algorithm is relatively straightforward, it is not obvious why DIET should control the type-1 error rate, or achieve power. In the next section, we explore the theoretical properties of DIET and provide an example where DIET achieves power where a baseline method provably cannot.

5.4 Theoretical analysis of DIET

Here we show that DIET achieves type-1 error control regardless of the data distribution. We then discuss when DIET can provably achieve power and characterize distributions where DIET is the most powerful test one can perform. The final part of this section provides a more general perspective on when distillation of a conditional randomization test into a marginal one is possible. We discuss how assumptions on the data generating process are always needed to guarantee power in a distillation procedure.

5.4.1 When can DIET control the type-1 error rate?

The type-1 error rate is the probability that the null hypothesis $\mathcal{H}_0$ is erroneously rejected: i.e. it is rejected when in reality $\mathbf{x} \perp\!\!\!\perp \mathbf{y} \mid \mathbf{z}$. To control this error rate at a user-specified level, the p-value under $\mathcal{H}_0$ must either be distributed uniformly over $[0, 1]$ or stochastically dominate[1] a Uniform$(0, 1)$ random variable (see appendix A.1 of Sudarshan et al. [2021] for a proof of this fact).

[1] A random variable $\mathbf{a}$ stochastically dominates a random variable $\mathbf{b}$ if the following partial ordering exists on the CDFs of $\mathbf{a}$ and $\mathbf{b}$: $\forall x : F_{\mathbf{a}}(x) \leq F_{\mathbf{b}}(x)$.

Prop. 5.4.1 shows that DIET p-values computed using algorithm 5.1 will stochastically dominate a Uniform$(0, 1)$ random variable.

Proposition 5.4.1. *Let $(\mathbf{x}, \mathbf{y}, \mathbf{z})$ be drawn from any distribution $p(\mathbf{x}, \mathbf{y}, \mathbf{z})$ and $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$ consist of N iid samples from this distribution. If $\mathbf{x} \perp\!\!\!\perp \mathbf{y} \mid \mathbf{z}$, then for any measure of marginal dependence $\rho : (\mathbb{R} \times \mathbb{R})^N \to \mathbb{R}$ the DIET p-value computed using algorithm 5.1 will stochastically dominate a Uniform$(0, 1)$ random variable.*

We detail the full proof next, but provide a sketch here. Under $\mathcal{H}_0$, the test statistic $T(\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}})$ is exchangeable with each of the null statistics $T(\mathcal{D}_{\tilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m)})$. As a result, the p-value $\hat{\mathbf{p}}$ computed using eq. (5.2) will be uniformly distributed over the set $\{\frac{1}{M+1}, \frac{2}{M+1}, \ldots, 1\}$. Such a p-value stochastically dominates a Uniform(0, 1) random variable. Prop. 5.4.1 ensures that if the practitioner rejects the null hypothesis when $\hat{\mathbf{p}} \leq \alpha$, the probability of an erroneous rejection is no greater than the significance level α.

Proof. Recall the DIET p-value introduced in algorithm 5.1:

$$\hat{\mathbf{p}} = \frac{1}{M+1}\left(1 + \sum_{m=1}^{M} \mathbb{1}\left[\rho(\mathcal{D}_{\hat{\boldsymbol{\epsilon}},\hat{\boldsymbol{\delta}}}) \geq \rho(\mathcal{D}_{\hat{\boldsymbol{\epsilon}},\hat{\boldsymbol{\delta}}}^{(m)})\right]\right).$$

We will prove that if a $\hat{q}$ estimator is trained on data $\mathcal{D}_{\tilde{\mathbf{x}},\mathbf{y},\mathbf{z}}$, the above p-value will be super-uniform. Using the technique from Candes et al. [2018], it suffices to show that the following sequence is exchangeable under the null, conditional on samples of $(\mathbf{z}, \mathbf{y})$:

$$\rho(\mathcal{D}_{\hat{\boldsymbol{\epsilon}},\hat{\boldsymbol{\delta}}}), \rho(\mathcal{D}_{\hat{\boldsymbol{\epsilon}},\hat{\boldsymbol{\delta}}}^{(1)}), \ldots, \rho(\mathcal{D}_{\hat{\boldsymbol{\epsilon}},\hat{\boldsymbol{\delta}}}^{(M)}).$$

Note that $\mathcal{D}_{\hat{\boldsymbol{\epsilon}},\hat{\boldsymbol{\delta}}}$, and $\{(\mathcal{D}_{\hat{\boldsymbol{\epsilon}},\hat{\boldsymbol{\delta}}}^{(m)})\}_{m=1}^{M}$ are datasets of information residuals. As such, the above sequence can be rewritten as:

$$\rho(\{\hat{\boldsymbol{\delta}}^{(i)}, \hat{\boldsymbol{\epsilon}}^{(i)}\}_{i=1}^{N}), \rho(\{\hat{\boldsymbol{\delta}}^{(i,1)}, \hat{\boldsymbol{\epsilon}}^{(i,1)}\}_{i=1}^{N}), \ldots, \rho(\{\hat{\boldsymbol{\delta}}^{(i,M)}, \hat{\boldsymbol{\epsilon}}^{(i,M)}\}_{i=1}^{N})$$

where $(\hat{\boldsymbol{\delta}}^{(i)}, \hat{\boldsymbol{\epsilon}}^{(i)})$ is the ith sample of $\mathcal{D}_{\hat{\boldsymbol{\epsilon}},\hat{\boldsymbol{\delta}}}$ and $(\hat{\boldsymbol{\delta}}^{(i,m)}, \hat{\boldsymbol{\epsilon}}^{(i,m)})$ is the ith sample of dataset $\mathcal{D}^{(m)}_{\hat{\boldsymbol{\epsilon}},\hat{\boldsymbol{\delta}}}$. As ρ is deterministic, it suffices to show that the following sequence is exchangeable conditional on $\{(\mathbf{y}^{(i)}, \mathbf{z}^{(i)})\}_{i=1}^N$:

$$\{\hat{\boldsymbol{\delta}}^{(i)}, \hat{\boldsymbol{\epsilon}}^{(i)}\}_{i=1}^N, \{\hat{\boldsymbol{\delta}}^{(i,1)}, \hat{\boldsymbol{\epsilon}}^{(i,1)}\}_{i=1}^N, \ldots, \{\hat{\boldsymbol{\delta}}^{(i,M)}, \hat{\boldsymbol{\epsilon}}^{(i,M)}\}_{i=1}^N$$

Note that $\hat{\boldsymbol{\delta}}^{(i)}, \hat{\boldsymbol{\epsilon}}^{(i)} \sim \hat{q}(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \mathbf{x}^{(i)}, \mathbf{y}^{(i)}, \mathbf{z}^{(i)})$. This means that the estimated information residuals can be written as $\hat{\boldsymbol{\delta}}^{(i)}, \hat{\boldsymbol{\epsilon}}^{(i)} = h(\boldsymbol{\alpha}^{(i)}, \mathbf{x}^{(i)}, \mathbf{y}^{(i)}, \mathbf{z}^{(i)})$, where h is a deterministic function (see appendix A of Trivedi and Zimmer [2007]). Rewriting the sampling process as a function of independent noise is similar in spirit to the reparameterization trick used in variational inference [Kingma et al. 2015].

In this alternative representation, $\boldsymbol{\alpha}^{(i)}$ is a sample of exogenous variable $\boldsymbol{\alpha}$ that represents the noise in $\hat{q}$. Without loss of generality, we can assume $\boldsymbol{\alpha}$ is independent of each subset of $\{\mathbf{x}, \mathbf{y}, \mathbf{z}\}$. Using the same notation, $\hat{\boldsymbol{\delta}}^{(i,m)}, \hat{\boldsymbol{\epsilon}}^{(i,m)} = h(\boldsymbol{\alpha}^{(i,m)}, \widetilde{\mathbf{x}}^{(i,m)}, \mathbf{y}^{(i)}, \mathbf{z}^{(i)})$, where $\widetilde{\mathbf{x}}^{(i,m)}$ is the ith sample of the mth null dataset $\widetilde{\mathbf{X}}^{(m)}$ and $\boldsymbol{\alpha}^{(i,m)}$ is another independent sample of $\boldsymbol{\alpha}$. This means the above sequence can be written as:

$$\begin{aligned}
&\{h(\boldsymbol{\alpha}^{(i)}, \mathbf{x}^{(i)}, \mathbf{y}^{(i)}, \mathbf{z}^{(i)})\}_{i=1}^N, \\
&\{h(\boldsymbol{\alpha}^{(i,1)}, \widetilde{\mathbf{x}}^{(i,1)}, \mathbf{y}^{(i)}, \mathbf{z}^{(i)})\}_{i=1}^N, \\
&\vdots \\
&\{h(\boldsymbol{\alpha}^{(i,M)}, \widetilde{\mathbf{x}}^{(i,M)}, \mathbf{y}^{(i)}, \mathbf{z}^{(i)})\}_{i=1}^N.
\end{aligned}$$

Since h is deterministic, exchangeability of the set of random variables above reduces to exchangeability of the following:

$$\{\{\boldsymbol{\alpha}^{(i)}, \mathbf{x}^{(i)}, \mathbf{y}^{(i)}, \mathbf{z}^{(i)}\}_{i=1}^N, \{\boldsymbol{\alpha}^{(i,1)}, \widetilde{\mathbf{x}}^{(i,1)}, \mathbf{y}^{(i)}, \mathbf{z}^{(i)}\}_{i=1}^N, \ldots \{\boldsymbol{\alpha}^{(i,M)}, \widetilde{\mathbf{x}}^{(i,M)}, \mathbf{y}^{(i)}, \mathbf{z}^{(i)}\}_{i=1}^N,$$

Under the null hypothesis $\mathcal{H}_0$, $p(\mathbf{x} \mid \mathbf{z}) = p(\mathbf{x} \mid \mathbf{z}, \mathbf{y})$. This means that the distribution of $\mathbf{x}^{(i)}$ is equal to the distribution of $\widetilde{\mathbf{x}}^{(i,m)}$ given $\{(\mathbf{y}^{(i)}, \mathbf{z}^{(i)})\}_{i=1}^N$ which together with the fact that $\{(\mathbf{y}^{(i)}, \mathbf{z}^{(i)})\}_{i=1}^N$ is constant across each element of the sequence means the joint distribution of $(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}, \mathbf{z}^{(i)})$ is the same. Using the fact that $\boldsymbol{\alpha}$ is independent of each subset of $\{\mathbf{x}, \mathbf{y}, \mathbf{z}\}$, the above sequence is exchangeable. □

5.4.2 When can DIET provably achieve power?

A CRT achieves power when the distribution of $T(\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}})$ is distinguishable from the distribution of each of the null statistics $T(\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m)})$. Here we provide assumptions on the data distribution that will ensure that DIET is able to distinguish between the distribution of the test statistic versus the null statistics.

Theorem 5.4.1. *Let $\hat{\boldsymbol{\epsilon}} = F(\mathbf{x} \mid \mathbf{z})$ and $\hat{\boldsymbol{\delta}} = F(\mathbf{y} \mid \mathbf{z})$ be random variables defined over $(\mathbf{x}, \mathbf{z})$ and $(\mathbf{y}, \mathbf{z})$ respectively. Let $F(\cdot \mid \mathbf{z})$ denote the conditional CDF. Assume F is invertible in the first argument and $(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}}) \perp\!\!\!\perp \mathbf{z}$. If there exists a marginal independence test $\psi : (\mathbb{R}\times\mathbb{R})^N \times [0,1] \to \{0,1\}$ that uses a measure of dependence ρ and achieves power greater than $\alpha \in [0,1]$, then DIET equipped with ρ and the conditional CDFs $F(\cdot \mid \mathbf{z})$ is a conditional independence test with power greater than α for data drawn from $p(\mathbf{x}, \mathbf{y}, \mathbf{z})$.*

The core assumption here is that $\hat{\boldsymbol{\epsilon}}$ and $\hat{\boldsymbol{\delta}}$ are jointly independent of the conditioning set of covariates $\mathbf{z}$. The conditional CDFs being invertible is a common assumption: e.g. when $\mathbf{x} \sim \mathcal{N}(\mathbf{z}_1, \sigma^2)$ or other continuous distributions.

We prove theorem 5.4.1 next: we show that given these conditions, DIET will provably be able to distinguish between the test and null statistics and achieve power to reject the null hypothesis. The proof establishes that when $\mathbf{x} \not\perp\!\!\!\perp \mathbf{y} \mid \mathbf{z}$, random variables $\hat{\boldsymbol{\epsilon}}$ and $\hat{\boldsymbol{\delta}}$ will be dependent. It also shows that under the null hypothesis $\mathcal{H}_0$, $\hat{\boldsymbol{\epsilon}} \perp\!\!\!\perp \hat{\boldsymbol{\delta}}$. Therefore, the test statistic, which measures the dependence of $\hat{\boldsymbol{\epsilon}}$ and $\hat{\boldsymbol{\delta}}$ will have a different distribution than the null statistics.

Proof. To test the conditional independence relationship $\mathbf{x} \perp\!\!\!\perp \mathbf{y} \mid \mathbf{z}$, DIET tests the marginal independence between $\hat{\boldsymbol{\epsilon}}$ and $\hat{\boldsymbol{\delta}}$. The aim of this proof is to show that $\hat{\boldsymbol{\epsilon}} \perp\!\!\!\perp \hat{\boldsymbol{\delta}}$ if and only if $\mathbf{x} \perp\!\!\!\perp \mathbf{y} \mid \mathbf{z}$. If this reduction holds, then under the alternate hypothesis $\mathcal{H}_1$ where $\mathbf{x} \not\perp\!\!\!\perp \mathbf{y} \mid \mathbf{z}$, the distribution of the test statistic $T(\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}})$ will be different from the distribution of each of the null statistics $T(\mathcal{D}_{\tilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m)})$. Then, given any marginal independence test that achieves power $> \alpha$ with statistic ρ, DIET with the same statistic is a *conditional* independence test with power $> \alpha$.

The proof is structured in the following manner. First, we will show that using the null data $\mathcal{D}_{\tilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m)}$, the sampled values of $\hat{\boldsymbol{\epsilon}}$ and $\hat{\boldsymbol{\delta}}$ will be independent. Then, we will show that using the true data $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$, the sampled values of $\hat{\boldsymbol{\epsilon}}$ and $\hat{\boldsymbol{\delta}}$ will be dependent. Finally, we discuss how the existence of a marginal independence test with power $> \alpha$ implies that DIET will also achieve power $> \alpha$ using data $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$.

PREREQUISITES. We first outline some properties will be used in both the null statistics and the test statistic section.

$$
\begin{aligned}
p(\epsilon, \mathbf{z}) &= \int p(\epsilon, \delta, \mathbf{z}) d\delta && \text{by marginalization} \\
&= \int p(\epsilon, \delta) p(\mathbf{z}) d\delta && \text{by data distribution} \\
&= p(\epsilon) p(\mathbf{z})
\end{aligned}
$$

This means that $\epsilon \perp\!\!\!\perp \mathbf{z}$. Using the same logic: $\delta \perp\!\!\!\perp \mathbf{z}$. Since theorem 5.4.1 considers DIET equipped with the true conditional CDFS $F(\cdot \mid \mathbf{z})$, $\hat{\boldsymbol{\epsilon}} = \epsilon$ and $\hat{\boldsymbol{\delta}} = \delta$. This means that the following must hold:

$$\hat{\boldsymbol{\epsilon}} \perp\!\!\!\perp \mathbf{z} \tag{5.4}$$

$$\hat{\boldsymbol{\delta}} \perp\!\!\!\perp \mathbf{z}. \tag{5.5}$$

NULL STATISTICS $T(\mathcal{D}_{\tilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m)})$. Recall that in each of the null datasets, the following factorization of the data distribution $p(\mathbf{x}, \mathbf{y}, \mathbf{z})$ holds by construction:

$$p(\mathbf{x}, \mathbf{y}, \mathbf{z}) = p(\mathbf{x} \mid \mathbf{z})p(\mathbf{y} \mid \mathbf{z})p(\mathbf{z}). \tag{5.6}$$

We can use this property to make the following sequence of deductions. Letting $p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}}, \mathbf{z})$ be the distribution implied by $(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}}, \mathbf{z})$,

$$\begin{aligned}
p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}}, \mathbf{z}) &= \int p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \mathbf{x}, \mathbf{y}, \mathbf{z})p(\mathbf{x}, \mathbf{y}, \mathbf{z})d\mathbf{x}d\mathbf{y} \\
&= \int p(\hat{\boldsymbol{\epsilon}} \mid \mathbf{x}, \mathbf{z})p(\hat{\boldsymbol{\delta}} \mid \mathbf{y}, \mathbf{z})p(\mathbf{x}, \mathbf{y}, \mathbf{z})d\mathbf{x}d\mathbf{y} && \hat{\boldsymbol{\epsilon}} \text{ and } \hat{\boldsymbol{\delta}} \text{ are each} \\
& && \text{functions of } \mathbf{z} \text{ and either } \mathbf{x} \text{ or } \mathbf{y} \\
&= \int p(\hat{\boldsymbol{\epsilon}} \mid \mathbf{x}, \mathbf{z})p(\hat{\boldsymbol{\delta}} \mid \mathbf{y}, \mathbf{z})p(\mathbf{x} \mid \mathbf{z})p(\mathbf{y} \mid \mathbf{z})p(\mathbf{z})d\mathbf{x}d\mathbf{y} && \text{by eq. (5.6)} \\
&= \int p(\hat{\boldsymbol{\epsilon}}, \mathbf{x} \mid \mathbf{z})p(\hat{\boldsymbol{\delta}}, \mathbf{y} \mid \mathbf{z})p(\mathbf{z})d\mathbf{x}d\mathbf{y} \\
&= p(\hat{\boldsymbol{\epsilon}} \mid \mathbf{z})p(\hat{\boldsymbol{\delta}} \mid \mathbf{z})p(\mathbf{z}) \\
p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \mathbf{z}) &= p(\hat{\boldsymbol{\epsilon}} \mid \mathbf{z})p(\hat{\boldsymbol{\delta}} \mid \mathbf{z}).
\end{aligned}$$

The distribution of $(\mathbf{y}, \mathbf{z})$ under the null is the same as distribution of $(\mathbf{y}, \mathbf{z})$ in the data. Then since $\hat{\boldsymbol{\delta}} \perp\!\!\!\perp \mathbf{z}$ (eq. 5.5) holds in the data distribution, the independence of $\hat{\boldsymbol{\delta}}$ and $\mathbf{z}$ also holds under the null distribution eq. (5.6):

$$\hat{\boldsymbol{\delta}} \perp\!\!\!\perp \mathbf{z} \qquad \text{where } (\mathbf{x}, \mathbf{y}, \mathbf{z}) \sim p(\mathbf{x} \mid \mathbf{z})p(\mathbf{y} \mid \mathbf{z})p(\mathbf{z}); \quad \hat{\boldsymbol{\delta}} = F(\mathbf{y} \mid \mathbf{z}).$$

Using the same logic, eq. (5.4) implies

$$\hat{\boldsymbol{\epsilon}} \perp\!\!\!\perp \mathbf{z} \qquad \text{where } (\mathbf{x}, \mathbf{y}, \mathbf{z}) \sim p(\mathbf{x} \mid \mathbf{z})p(\mathbf{y} \mid \mathbf{z})p(\mathbf{z}); \quad \hat{\boldsymbol{\epsilon}} = F(\mathbf{x} \mid \mathbf{z}).$$

Using the above facts,

$$\begin{aligned} p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}}) &= \int p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \mathbf{z}) p(\mathbf{z}) d\mathbf{z} && \text{by marginalization} \\ &= \int p(\hat{\boldsymbol{\epsilon}} \mid \mathbf{z}) p(\hat{\boldsymbol{\delta}} \mid \mathbf{z}) p(\mathbf{z}) d\mathbf{z} \\ &= \int p(\hat{\boldsymbol{\epsilon}}) p(\hat{\boldsymbol{\delta}}) p(\mathbf{z}) d\mathbf{z} \\ &= p(\hat{\boldsymbol{\epsilon}}) p(\hat{\boldsymbol{\delta}}). \end{aligned}$$

Therefore, when using a null dataset $\mathcal{D}_{\tilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m)}$, $\hat{\boldsymbol{\epsilon}} \perp\!\!\!\perp \hat{\boldsymbol{\delta}}$.

Test statistic $T(\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}})$. Under $\mathcal{H}_1$, $\mathbf{x} \not\perp\!\!\!\perp \mathbf{y} \mid \mathbf{z}$. In such cases, the sampled values of $\hat{\boldsymbol{\epsilon}}$ and $\hat{\boldsymbol{\delta}}$ using $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$ must be dependent. Specifically, the following sequence of implications must hold:

$$\mathbf{x} \not\perp\!\!\!\perp \mathbf{y} \mid \mathbf{z} \Rightarrow \hat{\boldsymbol{\epsilon}} \not\perp\!\!\!\perp \hat{\boldsymbol{\delta}} \mid \mathbf{z} \Rightarrow \hat{\boldsymbol{\epsilon}} \not\perp\!\!\!\perp \hat{\boldsymbol{\delta}}.$$

The first implication follows because both $F(\mathbf{x} \mid \mathbf{z})$ and $F(\mathbf{y} \mid \mathbf{z})$ are invertible for any fixed value of $\mathbf{z}$. Next we prove the second implication. This is equivalent to:

$$\hat{\boldsymbol{\delta}} \perp\!\!\!\perp \hat{\boldsymbol{\epsilon}} \Rightarrow \hat{\boldsymbol{\delta}} \perp\!\!\!\perp \hat{\boldsymbol{\epsilon}} \mid \mathbf{z}.$$

We know that $p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \mathbf{z}) = p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}})$, since $\hat{\boldsymbol{\epsilon}} = \epsilon$ and $\hat{\boldsymbol{\delta}} = \delta$ and $(\epsilon, \delta) \perp\!\!\!\perp \mathbf{z}$. It follows that $\hat{\boldsymbol{\delta}} \perp\!\!\!\perp \hat{\boldsymbol{\epsilon}} \Rightarrow \hat{\boldsymbol{\delta}} \perp\!\!\!\perp \hat{\boldsymbol{\epsilon}} \mid \mathbf{z}$:

$$\begin{aligned} p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \mathbf{z}) &= p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}}) \\ &= p(\hat{\boldsymbol{\epsilon}}) p(\hat{\boldsymbol{\delta}}) && \text{since } \hat{\boldsymbol{\delta}} \perp\!\!\!\perp \hat{\boldsymbol{\epsilon}} \\ &= p(\hat{\boldsymbol{\epsilon}} \mid \mathbf{z}) p(\hat{\boldsymbol{\delta}} \mid \mathbf{z}) && \text{by eqs. (5.4) and (5.5)} \end{aligned}$$

We have thus far established that under $\mathcal{H}_1$, $\hat{\boldsymbol{\delta}} \not\perp\!\!\!\perp \hat{\boldsymbol{\epsilon}}$, but under $\mathcal{H}_0$, $\hat{\boldsymbol{\delta}} \perp\!\!\!\perp \hat{\boldsymbol{\epsilon}}$. Now, consider $\psi(\mathcal{D}_{\hat{\boldsymbol{\epsilon}},\hat{\boldsymbol{\delta}}}, \alpha) : (\mathbb{R} \times \mathbb{R})^N \times [0,1] \to \{0,1\}$, a marginal independence test that uses statistic $\rho : (\mathbb{R} \times \mathbb{R})^N \to \mathbb{R}$ and has power greater than level α. This means that there exists a rejection region $R_\alpha = \{\mathcal{D} \in (\mathbb{R} \times \mathbb{R})^N : \psi(\mathcal{D}, \alpha) = 1\}$ where $\mathbb{P}_{\mathcal{H}_1}(R_\alpha) \geq \mathbb{P}_{\mathcal{H}_0}(R_\alpha)$. In other words, for a sample size of N and statistic ρ there is sufficient evidence to reject the null hypothesis.

Then, DIET equipped with ρ, $F(\mathbf{x}, \mid \mathbf{z})$, and $F(\mathbf{y}, \mid \mathbf{z})$ is a conditional independence test $\zeta(\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}, \alpha) : (\mathbb{R} \times \mathbb{R} \times \mathbb{R}^{d_\mathbf{z}})^N \times [0,1] \to \{0,1\}$ with rejection region $S_\alpha = \{\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}} \in (\mathbb{R} \times \mathbb{R} \times \mathbb{R}^p)^N : \zeta(\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}, \alpha) = 1\}$ such that $\mathbb{P}_{\mathcal{H}_1}(S_\alpha) \geq \mathbb{P}_{\mathcal{H}_0}(S_\alpha)$. This follows directly from the previous fact because DIET uses the marginal dependence $\hat{\boldsymbol{\epsilon}}$ and $\hat{\boldsymbol{\delta}}$ to test the conditional independence between $\mathbf{x}$ and $\mathbf{y}$ given $\mathbf{z}$.

Thus, if there is a marginal test that achieves power greater than α, then DIET under the conditions of theorem 5.4.1 will also achieve power greater than α.

□

5.4.3 When is DIET the most powerful conditionally valid CRT?

Here we show that under the same conditions as theorem 5.4.1, DIET equipped with a measure of mutual information ρ is the *most powerful* conditionally valid CRT [Katsevich and Ramdas 2020].

The set of valid CRTs $\mathcal{C}_\alpha$ includes any CRT where the type-1 error is less than α using a dataset $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$. Given samples of $(\mathbf{y}, \mathbf{z})$, the set of conditionally valid CRTs at level α is a subset of $\mathcal{C}_\alpha$ where the samples of $(\mathbf{y}, \mathbf{z})$ in $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$ are fixed. A conditionally valid CRT is also a marginally valid CRT. The following proposition states that given access to the conditional CDFs $F(\mathbf{x} \mid \mathbf{z})$ and $F(\mathbf{y} \mid \mathbf{z})$, DIET is the most powerful conditionally valid CRT. Thus, the power of DIET is tied directly to the quality of the estimation of these conditional CDFs.

Proposition 5.4.2. *Let $\hat{\boldsymbol{\epsilon}} = F(\mathbf{x} \mid \mathbf{z})$ and $\hat{\boldsymbol{\delta}} = F(\mathbf{y} \mid \mathbf{z})$. For data generating processes where both $F(\cdot \mid \mathbf{z})$ functions are invertible in the first argument and $(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}}) \perp\!\!\!\perp \mathbf{z}$, DIET with the following*

mutual information-based marginal dependence measure ρ is the most powerful conditionally valid test:

$$\rho(\mathcal{D}_{\hat{\boldsymbol{\epsilon}},\hat{\boldsymbol{\delta}}}) = \frac{1}{N}\sum_{i=1}^{N} \log \frac{p(\hat{\boldsymbol{\delta}}^{(i)}, \hat{\boldsymbol{\epsilon}}^{(i)})}{p(\hat{\boldsymbol{\delta}}^{(i)})p(\hat{\boldsymbol{\epsilon}}^{(i)})}.$$

We prove prop. 5.4.2 next by showing that the likelihood ratio in prop. 5.4.2 is equivalent to the likelihood ratio of $p(\mathbf{y} \mid \mathbf{x}, \mathbf{z})$ and $p(\mathbf{y} \mid \mathbf{z})$: the most powerful conditionally valid CRT test statistic.

Proof. Using the conditional CDFs $F(\mathbf{x} \mid \mathbf{z})$ and $F(\mathbf{y} \mid \mathbf{z})$, define the following terms for convenience:

$$\begin{aligned}
\bar{f}_{\mathbf{z}}(\mathbf{x}) &:= F(\mathbf{x} \mid \mathbf{z}) \\
\bar{g}_{\mathbf{z}}(\mathbf{y}) &:= F(\mathbf{y} \mid \mathbf{z}) \\
J &= \begin{bmatrix} \frac{d}{d\mathbf{x}}\bar{f}_{\mathbf{z}}(\mathbf{x}) & \frac{d}{d\mathbf{y}}\bar{f}_{\mathbf{z}}(\mathbf{x}) \\ \frac{d}{d\mathbf{x}}\bar{g}_{\mathbf{z}}(\mathbf{y}) & \frac{d}{d\mathbf{y}}\bar{g}_{\mathbf{z}}(\mathbf{y}) \end{bmatrix} \\
&= \begin{bmatrix} \bar{f}'_{\mathbf{z}}(\mathbf{x}) & 0 \\ 0 & \bar{g}'_{\mathbf{z}}(\mathbf{y}) \end{bmatrix}.
\end{aligned}$$

The off-diagonals of J are 0 because $\bar{f}_{\mathbf{z}}(\mathbf{x})$ is not a function of $\mathbf{y}$ and $\bar{g}_{\mathbf{z}}(\mathbf{y})$ is not a function of $\mathbf{x}$. Then using change of variables, we can write:

$$\begin{aligned}
p(\mathbf{x}, \mathbf{y} \mid \mathbf{z}) &= p(\hat{\boldsymbol{\epsilon}} = \bar{f}_{\mathbf{z}}(\mathbf{x}), \hat{\boldsymbol{\delta}} = \bar{g}_{\mathbf{z}}(\mathbf{y}) \mid \mathbf{z}) \cdot |\det(J)| \\
&= p(\hat{\boldsymbol{\epsilon}} = \bar{f}_{\mathbf{z}}(\mathbf{x}), \hat{\boldsymbol{\delta}} = \bar{g}_{\mathbf{z}}(\mathbf{y}) \mid \mathbf{z}) \cdot |\bar{f}'_{\mathbf{z}}(\mathbf{x}) \cdot \bar{g}'_{\mathbf{z}}(\mathbf{y})| \\
&= p(\hat{\boldsymbol{\epsilon}} = \bar{f}_{\mathbf{z}}(\mathbf{x}), \hat{\boldsymbol{\delta}} = \bar{g}_{\mathbf{z}}(\mathbf{y}) \mid \mathbf{z}) \cdot \bar{f}'_{\mathbf{z}}(\mathbf{x}) \cdot \bar{g}'_{\mathbf{z}}(\mathbf{y}) \\
&= p(\hat{\boldsymbol{\epsilon}} = \bar{f}_{\mathbf{z}}(\mathbf{x}), \hat{\boldsymbol{\delta}} = \bar{g}_{\mathbf{z}}(\mathbf{y})) \cdot \bar{f}'_{\mathbf{z}}(\mathbf{x}) \cdot \bar{g}'_{\mathbf{z}}(\mathbf{y}).
\end{aligned}$$

The second last step follows because $\bar{f}_{\mathbf{z}}$ and $\bar{g}_{\mathbf{z}}$ are monotonically non-decreasing, meaning their derivatives with respect to $\mathbf{x}$ or $\mathbf{y}$ for a fixed $\mathbf{z}$ are non-negative. The absolute value of the product of two non-negative quantities is just the product of the two quantities. The last step uses the assumption that $(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}}) \perp\!\!\!\perp \mathbf{z}$. Using similar reasoning,

$$p(\mathbf{x} \mid \mathbf{z}) = p(\hat{\boldsymbol{\epsilon}} = \bar{f}_{\mathbf{z}}(\mathbf{x}) \mid \mathbf{z}) \cdot \bar{f}'_{\mathbf{z}}(\mathbf{x}) = p(\hat{\boldsymbol{\epsilon}} = \bar{f}_{\mathbf{z}}(\mathbf{x})) \cdot \bar{f}'_{\mathbf{z}}(\mathbf{x})$$
$$p(\mathbf{y} \mid \mathbf{z}) = p(\hat{\boldsymbol{\delta}} = \bar{g}_{\mathbf{z}}(\mathbf{y}) \mid \mathbf{z}) \cdot \bar{g}'_{\mathbf{z}}(\mathbf{y}) = p(\hat{\boldsymbol{\delta}} = \bar{g}_{\mathbf{z}}(\mathbf{y})) \cdot \bar{g}'_{\mathbf{z}}(\mathbf{y}).$$

Using the above change of variable results, we can manipulate the likelihood ratio statistic that Katsevich and Ramdas [2020] prove is the conditionally most powerful against point alternatives.

$$\begin{aligned}
\frac{1}{N}\sum_{i=1}^{N} \log \frac{p(\mathbf{y}^{(i)} \mid \mathbf{x}^{(i)}, \mathbf{z}^{(i)})}{p(\mathbf{y}^{(i)} \mid \mathbf{z}^{(i)})} &= \frac{1}{N}\sum_{i=1}^{N} \log \frac{p(\mathbf{x}^{(i)}, \mathbf{y}^{(i)} \mid \mathbf{z}^{(i)})}{p(\mathbf{x}^{(i)} \mid \mathbf{z}^{(i)}) p(\mathbf{y}^{(i)} \mid \mathbf{z}^{(i)})} \\
&= \frac{1}{N}\sum_{i=1}^{N} \log \frac{p(\hat{\boldsymbol{\epsilon}} = \bar{f}_{\mathbf{z}^{(i)}}(\mathbf{x}^{(i)}), \hat{\boldsymbol{\delta}} = \bar{g}_{\mathbf{z}^{(i)}}(\mathbf{y}^{(i)})) \cdot \bar{f}'_{\mathbf{z}^{(i)}}(\mathbf{x}^{(i)}) \cdot \bar{g}'_{\mathbf{z}^{(i)}}(\mathbf{y}^{(i)})}{p(\hat{\boldsymbol{\epsilon}} = \bar{f}_{\mathbf{z}}(\mathbf{x})) \cdot \bar{f}'_{\mathbf{z}}(\mathbf{x}) \cdot p(\hat{\boldsymbol{\delta}} = \bar{g}_{\mathbf{z}}(\mathbf{y})) \cdot \bar{g}'_{\mathbf{z}}(\mathbf{y})} \\
&= \frac{1}{N}\sum_{i=1}^{N} \log \frac{p(\hat{\boldsymbol{\epsilon}} = \bar{f}_{\mathbf{z}^{(i)}}(\mathbf{x}^{(i)}), \hat{\boldsymbol{\delta}} = \bar{g}_{\mathbf{z}^{(i)}}(\mathbf{y}^{(i)}))}{p(\hat{\boldsymbol{\epsilon}} = \bar{f}_{\mathbf{z}}(\mathbf{x})) \cdot p(\hat{\boldsymbol{\delta}} = \bar{g}_{\mathbf{z}}(\mathbf{y}))}.
\end{aligned}$$

Note that this final term on is exactly the mutual-information based marginal dependence measure in the statement of prop. 5.4.2. Therefore, the optimal DIET solution is the most powerful conditionally valid test against point alternatives.

□

5.4.4 Multiple testing and variable selection

A common application of CRTs is controlled variable selection [Candes et al. 2018]. Let $\mathbf{x} = \{\mathbf{x}_1, \ldots, \mathbf{x}_d\}$ be a set of covariates, and $\mathbf{y}$ be a response. Controlled variable selection methods identify a subset of important covariates by testing the conditional independence of each covariate $\mathbf{x}_j$ and $\mathbf{y}$ given all other covariates $\mathbf{x}_{-j}$. If the hypothesis test for $\mathbf{x}_j$ results in a rejection, that

variable is "selected." The goal of controlled variable selection is to select as many variables as possible, while controlling for the FDR: an analog for type-1 error in multiple testing.

We apply the following procedure to use DIET for CVS. To test $\mathbf{x}_j \perp\!\!\!\perp \mathbf{y} \mid \mathbf{x}_{-j}$ for each $\mathbf{x}_j$, we run algorithm 5.1 where $\mathbf{z} \leftarrow \mathbf{x}_{-j}$, $\mathbf{y} \leftarrow \mathbf{y}$, and $\mathbf{x} \leftarrow \mathbf{x}_j$. The resulting set of p-values is used with standard FDR-controlling procedures [Benjamini and Hochberg 1995; Benjamini and Yekutieli 2001] to select important covariates.

5.4.5 Can we further generalize the assumptions made by DIET?

Here we explore the limits of distillation: is it possible to generalize the set of distributions for which power is achievable beyond DIET? We first outline a general way to distill a conditional independence test into a marginal one. We then discuss how the existence of a distillation procedure does not necessarily imply a distilled CRT that achieves power.

A general distillation procedure. Let $L^2_{\mathbf{x},\mathbf{z}}$ denote the space of real-valued functions f of $(\mathbf{x}, \mathbf{z})$, where $\mathbb{E}[f(\mathbf{x}, \mathbf{z})^2] < \infty$. Let $L^2_{\mathbf{y},\mathbf{z}}$ be defined analogously. Then Daudin [1980] shows that, for all functions $f \in L^2_{\mathbf{x},\mathbf{z}}$ and $g \in L^2_{\mathbf{y},\mathbf{z}}$ such that $\mathbb{E}[f(\mathbf{x}, \mathbf{z}) \mid \mathbf{z}] = 0$ and $\mathbb{E}[g(\mathbf{y}, \mathbf{z}) \mid \mathbf{z}] = 0$,

$$\mathbf{x} \perp\!\!\!\perp \mathbf{y} \mid \mathbf{z} \iff \mathbb{E}[f(\mathbf{x}, \mathbf{z})g(\mathbf{y}, \mathbf{z})] = 0.$$

This means that if $\mathbf{y}$ is conditionally dependent on $\mathbf{x}$ given $\mathbf{z}$, then there must exist functions f and g such that their correlation is non-zero. Rather than testing the marginal independence of conditional CDFs $F(\mathbf{x} \mid \mathbf{z})$ and $F(\mathbf{y} \mid \mathbf{z})$, one could instead test the marginal independence of $f(\mathbf{x}, \mathbf{z})$ and $g(\mathbf{y}, \mathbf{z})$ instead.

If f and g are known beforehand, testing this marginal independence will yield a conditional independence test with power. However, in reality f and g must be learned using data. Further, as discussed in section 5.4.1, FDR control in computationally tractable CRTs that avoid sample split-

ting requires f and g be inferred without using triples of $(\mathbf{x}, \mathbf{y}, \mathbf{z})$. Only the marginal distributions of $(\mathbf{x}, \mathbf{z})$ and $(\mathbf{y}, \mathbf{z})$ can be used.

This presents a problem because no distilled CRT that learns f and g from these two marginals without prior knowledge can discriminate between the following two data generating processes:

$$\mathbf{y} = \mathbf{x} + \mathbf{z} \mod 1 \qquad \mathbf{x}, \mathbf{z} \sim \text{Uniform}(0, 1) \qquad \text{(conditional dependence)}$$

$$\mathbf{y}, \mathbf{x}, \mathbf{z} \sim \text{Uniform}(0, 1), \qquad \text{(conditional independence)}$$

where $\mathbf{a} + \mathbf{b} \mod 1$ is defined as $\mathbf{a} + \mathbf{b}$ if $\mathbf{a} + \mathbf{b} < 1$ and $\mathbf{a} + \mathbf{b} - 1$ if $\mathbf{a} + \mathbf{b} \geq 1$. Note that the marginals of $(\mathbf{x}, \mathbf{z})$ and $(\mathbf{y}, \mathbf{z})$ are the same across both processes. Therefore, without assumptions on the data generating process, it is impossible to guarantee that a distilled CRT will learn f and g that can differentiate between these two data generating processes.

In the next section we provide sufficient conditions to achieve power with a distillation-based CRT. Theorem 5.4.2 provides assumptions on the data generating process and outlines a learning procedure for the functions f and g such that power is provably achieved. In the context of theorem 5.4.2 we also discuss assumptions the d_0-CRT must make to achieve power.

5.4.6 Sufficient conditions for power with a general distillation-based CRT

In this section we consider data generating processes of the following form:

$$\mathbf{z} \sim p(\mathbf{z}) \qquad (\epsilon, \delta) \sim p(\epsilon, \delta) \qquad \mathbf{x} = f(\epsilon, \mathbf{z}) \qquad \mathbf{y} = g(\delta, \mathbf{z}).$$

One way to interpret distillation procedures like DIET or the d_0-CRT is as follows. First estimate ϵ and δ from samples of $(\mathbf{x}, \mathbf{y}, \mathbf{z})$, then test the marginal independence of these estimates: $\hat{\boldsymbol{\epsilon}} \perp\!\!\!\perp \hat{\boldsymbol{\delta}}$. Since ϵ and δ are unobserved, samples in $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$ map to a distribution $p(\delta, \epsilon \mid \mathbf{x}, \mathbf{y}, \mathbf{z})$ over the

possible values of (ϵ, δ). The distribution $p(\epsilon, \delta \mid \mathbf{x}, \mathbf{y}, \mathbf{z})$ is also unknown; it must be estimated using an estimator $\hat{q}(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \mathbf{x}, \mathbf{y}, \mathbf{z})$.

However, not all $\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \sim \hat{q}(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \mathbf{x}, \mathbf{y}, \mathbf{z})$ will yield power to reject the null hypothesis $\mathcal{H}_0 : \mathbf{x} \perp\!\!\!\perp \mathbf{y} \mid \mathbf{z}$. In some cases $\hat{\boldsymbol{\epsilon}} \perp\!\!\!\perp \hat{\boldsymbol{\delta}}$ but $\mathbf{x} \not\perp\!\!\!\perp \mathbf{y} \mid \mathbf{z}$. As an example reconsider the data generating process in the previous section where $\mathbf{x} = \hat{\boldsymbol{\epsilon}}$ and $\mathbf{y} = \hat{\boldsymbol{\delta}}$, let $\hat{\boldsymbol{\epsilon}} \sim \text{Uniform}(0, 1)$, $\hat{\boldsymbol{\delta}} \sim \text{Uniform}(0, 1)$, and

$$\mathbf{z} = \begin{cases} \hat{\boldsymbol{\epsilon}} + \hat{\boldsymbol{\delta}} & \text{if } \hat{\boldsymbol{\epsilon}} + \hat{\boldsymbol{\delta}} \leq 1 \\ \hat{\boldsymbol{\epsilon}} + \hat{\boldsymbol{\delta}} - 1 & \text{otherwise} \end{cases}.$$

In this example, $\hat{\boldsymbol{\epsilon}}$ and $\hat{\boldsymbol{\delta}}$ are independent of each other, but $\mathbf{x}$ and $\mathbf{y}$ are clearly dependent given $\mathbf{z}$. The following theorem, theorem 5.4.2, gives sufficient conditions on $\hat{q}(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \mathbf{x}, \mathbf{y}, \mathbf{z})$ to ensure that $\hat{\boldsymbol{\epsilon}} \perp\!\!\!\perp \hat{\boldsymbol{\delta}}$ if and only if $\mathbf{x} \perp\!\!\!\perp \mathbf{y} \mid \mathbf{z}$. We later show that the only way to satisfy the conditions in theorem 5.4.2 are through assumptions on the data generating process.

Theorem 5.4.2. *Let* $(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}}, \mathbf{x}, \mathbf{y}, \mathbf{z}) \sim \hat{q}(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \mathbf{x}, \mathbf{y}, \mathbf{z}) p(\mathbf{x}, \mathbf{y}, \mathbf{z})$. *Further, let:*

$$\hat{q}(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \mathbf{x}, \mathbf{y}, \mathbf{z}) = p(\hat{\boldsymbol{\epsilon}} \mid \mathbf{x}, \mathbf{z}) p(\hat{\boldsymbol{\delta}} \mid \mathbf{y}, \mathbf{z}), \qquad \text{(factorization)}$$

$$\exists \tilde{f}, \tilde{g} \quad \textit{s.t.} \quad \mathbf{x} \overset{a.s.}{=} \tilde{f}(\hat{\boldsymbol{\epsilon}}, \mathbf{z}), \quad \textit{and} \quad \mathbf{y} \overset{a.s.}{=} \tilde{g}(\hat{\boldsymbol{\delta}}, \mathbf{z}), \qquad \text{(reconstruction)}$$

$$(\delta, \hat{\boldsymbol{\delta}}) \perp\!\!\!\perp \mathbf{z} \qquad (\epsilon, \hat{\boldsymbol{\epsilon}}) \perp\!\!\!\perp \mathbf{z}. \qquad \text{(joint independence)}$$

Let $\psi(\mathcal{D}_{\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}}}, \alpha) : (\mathbb{R} \times \mathbb{R})^N \times [0, 1] \to \{0, 1\}$ *be a marginal independence test that uses statistic* $\rho : (\mathbb{R} \times \mathbb{R})^N \to \mathbb{R}$ *and has power greater than* α. *Let* $\mathcal{D}_{\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}}}$ *be a dataset of* N *samples of* $(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}})$ *generated using* $\hat{q}(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \mathbf{x}, \mathbf{y}, \mathbf{z})$ *and* $\mathcal{D}_{\mathbf{x}, \mathbf{y}, \mathbf{z}}$. *Then,* ψ *using* $\mathcal{D}_{\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}}}$ *and* ρ *is also a conditional test of independence for* $\mathbf{x} \perp\!\!\!\perp \mathbf{y} \mid \mathbf{z}$ *with power greater than* α.

Proof. The core of this proof is to show that if factorization, reconstruction, and joint indepen-

dence are satisfied, then

$$\mathbf{x} \perp\!\!\!\perp \mathbf{y} \mid \mathbf{z} \Leftrightarrow \hat{\boldsymbol{\epsilon}} \perp\!\!\!\perp \hat{\boldsymbol{\delta}}.$$

If this reduction is possible, then under $\mathcal{H}_1$, $\hat{\boldsymbol{\epsilon}} \not\perp\!\!\!\perp \hat{\boldsymbol{\delta}}$, but under $\mathcal{H}_0$, $\hat{\boldsymbol{\epsilon}} \perp\!\!\!\perp \hat{\boldsymbol{\delta}}$. This implies that the distribution of the marginal dependence test statistic $\rho(\mathcal{D}_{\hat{\boldsymbol{\epsilon}},\hat{\boldsymbol{\delta}}})$ is different from that of each null statistic $\rho(\mathcal{D}^{(m)}_{\hat{\boldsymbol{\epsilon}},\hat{\boldsymbol{\delta}}})$. Thus, the p-value computed by ψ will be close to 0:

$$\hat{\mathbf{p}} = \frac{1}{M+1}\left(1 + \sum_{m=1}^{M} \mathbb{1}(\rho(\mathcal{D}_{\hat{\boldsymbol{\epsilon}},\hat{\boldsymbol{\delta}}}) \leq \rho(\mathcal{D}^{(m)}_{\hat{\boldsymbol{\epsilon}},\hat{\boldsymbol{\delta}}}))\right).$$

Let $p(\delta, \epsilon, \hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}}, \mathbf{z})$ be a distribution over variables $\delta, \epsilon, \hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}}, \mathbf{z}$. The variables $\hat{\boldsymbol{\delta}}$ and $\hat{\boldsymbol{\epsilon}}$ are samples from $\hat{q}(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \mathbf{x}, \mathbf{y}, \mathbf{z})$. For simplicity, we show the proof of theorem 5.4.2 when all random variables are continuous, but the same reasoning holds for discrete random variables.

Null statistics. For null statistics $\rho(\mathcal{D}^{(m)}_{\hat{\boldsymbol{\epsilon}},\hat{\boldsymbol{\delta}}})$ computed using null data $\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m)}$, $\hat{\boldsymbol{\delta}}, \hat{\boldsymbol{\epsilon}} \sim \hat{q}(\hat{\boldsymbol{\delta}}, \hat{\boldsymbol{\epsilon}} \mid \widetilde{\mathbf{x}}, \mathbf{y}, \mathbf{z})$ must be independent. In the null data, $\widetilde{\mathbf{x}} \perp\!\!\!\perp \mathbf{y} \mid \mathbf{z}$ by construction, so the following must hold:

$$\widetilde{\mathbf{x}} \perp\!\!\!\perp \mathbf{y} \mid \mathbf{z} \Rightarrow \hat{\boldsymbol{\epsilon}} \perp\!\!\!\perp \hat{\boldsymbol{\delta}}. \tag{5.7}$$

We show this fact by manipulating the distribution $\hat{q}(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \widetilde{\mathbf{x}}, \mathbf{y}, \mathbf{z})p(\widetilde{\mathbf{x}}, \mathbf{y}, \mathbf{z})$. In this proof, we write $\hat{q}(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \widetilde{\mathbf{x}}, \mathbf{y}, \mathbf{z})$ as $p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \widetilde{\mathbf{x}}, \mathbf{y}, \mathbf{z})$ to simplify the notation:

$$\begin{aligned}
p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}}, \mathbf{z}) &= \int p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \widetilde{\mathbf{x}}, \mathbf{y}, \mathbf{z}) p(\widetilde{\mathbf{x}}, \mathbf{y}, \mathbf{z}) d\widetilde{\mathbf{x}} d\mathbf{y} \\
&= \int p(\hat{\boldsymbol{\epsilon}} \mid \widetilde{\mathbf{x}}, \mathbf{z}) p(\hat{\boldsymbol{\delta}} \mid \mathbf{y}, \mathbf{z}) p(\widetilde{\mathbf{x}}, \mathbf{y}, \mathbf{z}) d\widetilde{\mathbf{x}} d\mathbf{y} && \text{By factorization} \\
&= \int p(\hat{\boldsymbol{\epsilon}} \mid \mathbf{x}, \mathbf{z}) p(\hat{\boldsymbol{\delta}} \mid \mathbf{y}, \mathbf{z}) p(\widetilde{\mathbf{x}} \mid \mathbf{z}) p(\mathbf{y} \mid \mathbf{z}) p(\mathbf{z}) d\widetilde{\mathbf{x}} d\mathbf{y} && \text{In } \mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}\ \widetilde{\mathbf{x}} \perp\!\!\!\perp \mathbf{y} \mid \mathbf{z} \\
&= \int p(\hat{\boldsymbol{\epsilon}}, \widetilde{\mathbf{x}} \mid \mathbf{z}) p(\hat{\boldsymbol{\delta}}, \mathbf{y} \mid \mathbf{z}) p(\mathbf{z}) d\widetilde{\mathbf{x}} d\mathbf{y} \\
&= p(\hat{\boldsymbol{\epsilon}} \mid \mathbf{z}) p(\hat{\boldsymbol{\delta}} \mid \mathbf{z}) p(\mathbf{z})
\end{aligned}$$

Consequently, $p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \mathbf{z}) = p(\hat{\boldsymbol{\delta}} \mid \mathbf{z})p(\hat{\boldsymbol{\delta}} \mid \mathbf{z}) \Leftrightarrow \hat{\boldsymbol{\epsilon}} \perp\!\!\!\perp \hat{\boldsymbol{\delta}} \mid \mathbf{z}$. Here, if $\hat{\boldsymbol{\delta}} \perp\!\!\!\perp \mathbf{z}$ and $\hat{\boldsymbol{\epsilon}} \perp\!\!\!\perp \mathbf{z}$, then

$$\hat{\boldsymbol{\epsilon}} \perp\!\!\!\perp \hat{\boldsymbol{\delta}} \mid \mathbf{z} \Rightarrow \hat{\boldsymbol{\epsilon}} \perp\!\!\!\perp (\hat{\boldsymbol{\delta}}, \mathbf{z}) \Rightarrow \hat{\boldsymbol{\epsilon}} \perp\!\!\!\perp \hat{\boldsymbol{\delta}}$$

Thus, if $\widetilde{\mathbf{x}} \perp\!\!\!\perp \mathbf{y} \mid \mathbf{z}$, as is the case in the computation of each of the null statistics $\rho(\mathcal{D}^{(m)}_{\hat{\boldsymbol{\epsilon}},\hat{\boldsymbol{\delta}}})$, then for $\hat{\boldsymbol{\delta}}, \hat{\boldsymbol{\epsilon}} \sim \hat{q}(\hat{\boldsymbol{\delta}}, \hat{\boldsymbol{\epsilon}} \mid \widetilde{\mathbf{x}}, \mathbf{y}, \mathbf{z})$, $\hat{\boldsymbol{\delta}} \perp\!\!\!\perp \hat{\boldsymbol{\epsilon}}$.

Test statistic under $\mathcal{H}_1$. For the test statistic $\rho(\mathcal{D}_{\hat{\boldsymbol{\epsilon}},\hat{\boldsymbol{\delta}}})$ computed using the data $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$, $\hat{\boldsymbol{\delta}}$ and $\hat{\boldsymbol{\epsilon}}$ must be dependent. Under $\mathcal{H}_1$, $\mathbf{x} \not\perp\!\!\!\perp \mathbf{y} \mid \mathbf{z}$, so the following sequence of implications must hold:

$$\mathbf{x} \not\perp\!\!\!\perp \mathbf{y} \mid \mathbf{z} \Rightarrow \tilde{f}(\hat{\boldsymbol{\epsilon}}, \mathbf{z}) \not\perp\!\!\!\perp \tilde{g}(\hat{\boldsymbol{\delta}}, \mathbf{z}) \mid \mathbf{z} \Rightarrow \hat{\boldsymbol{\epsilon}} \not\perp\!\!\!\perp \hat{\boldsymbol{\delta}} \mid \mathbf{z} \Rightarrow \hat{\boldsymbol{\epsilon}} \not\perp\!\!\!\perp \hat{\boldsymbol{\delta}}. \tag{5.8}$$

The first implication follows directly from reconstruction, the second holds because $\hat{\boldsymbol{\epsilon}}$ and $\hat{\boldsymbol{\delta}}$ are the only sources of variance when $\mathbf{z}$ is fixed. This last implication is equivalent to the following statement, which we will subsequently prove:

$$\hat{\boldsymbol{\delta}} \perp\!\!\!\perp \hat{\boldsymbol{\epsilon}} \Rightarrow \hat{\boldsymbol{\delta}} \perp\!\!\!\perp \hat{\boldsymbol{\epsilon}} \mid \mathbf{z}.$$

First, note the following properties. Using joint independence, we show that the distribution $p(\hat{\boldsymbol{\delta}}, \mathbf{z})$ factorizes, implying that $\hat{\boldsymbol{\delta}}$ and $\mathbf{z}$ are marginally independent:

$$p(\hat{\boldsymbol{\delta}}, \mathbf{z}) = \int p(\delta, \hat{\boldsymbol{\delta}}, \mathbf{z}) d\delta = \int p(\delta, \hat{\boldsymbol{\delta}}) p(\mathbf{z}) d\delta = p(\hat{\boldsymbol{\delta}}) p(\mathbf{z}), \tag{5.9}$$

$$p(\hat{\boldsymbol{\epsilon}}, \mathbf{z}) = \int p(\epsilon, \hat{\boldsymbol{\epsilon}}, \mathbf{z}) d\delta = \int p(\epsilon, \hat{\boldsymbol{\epsilon}}) p(\mathbf{z}) d\epsilon = p(\hat{\boldsymbol{\epsilon}}) p(\mathbf{z}). \tag{5.10}$$

Further, joint independence implies the following:

$$\frac{p(\hat{\boldsymbol{\delta}}, \delta, \mathbf{z})}{p(\delta)} = \frac{p(\hat{\boldsymbol{\delta}}, \delta) p(\mathbf{z})}{p(\delta)} = p(\hat{\boldsymbol{\delta}} \mid \delta) p(\mathbf{z}) = p(\hat{\boldsymbol{\delta}} \mid \delta) p(\mathbf{z} \mid \delta) \quad \text{Since } \mathbf{z} \perp\!\!\!\perp \delta \text{ by definition} \tag{5.11}$$

$$\frac{p(\hat{\boldsymbol{\epsilon}}, \epsilon, \mathbf{z})}{p(\epsilon)} = \frac{p(\hat{\boldsymbol{\epsilon}}, \epsilon)p(\mathbf{z})}{p(\epsilon)} = p(\hat{\boldsymbol{\epsilon}} \mid \epsilon)p(\mathbf{z}) = p(\hat{\boldsymbol{\epsilon}} \mid \epsilon)p(\mathbf{z} \mid \epsilon) \qquad \text{Since } \mathbf{z} \perp\!\!\!\perp \epsilon \text{ by definition} \tag{5.12}$$

Next, note that factorization implies:

$$\begin{aligned}
p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \mathbf{z}, \epsilon, \delta) &= p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \mathbf{x}, \mathbf{y}, \mathbf{z}, \epsilon, \delta) && \mathbf{x} \text{ and } \mathbf{y} \text{ are fully determined by } (\mathbf{z}, \delta, \epsilon) \\
&= p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \mathbf{x}, \mathbf{y}, \mathbf{z}) && (\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}}) \text{ are functions of only } (\mathbf{x}, \mathbf{y}, \mathbf{z}) \text{ and exogenous noise} \\
&= p(\hat{\boldsymbol{\epsilon}} \mid \mathbf{x}, \mathbf{z})p(\hat{\boldsymbol{\delta}} \mid \mathbf{y}, \mathbf{z}) && \text{factorization assumption} \\
&= p(\hat{\boldsymbol{\epsilon}} \mid \mathbf{x}, \mathbf{z}, \epsilon)p(\hat{\boldsymbol{\delta}} \mid \mathbf{y}, \mathbf{z}, \delta) && \hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \text{ are functions of } (\mathbf{x}, \mathbf{z}) \\
& && \text{and } (\mathbf{y}, \mathbf{z}) \text{ respectively and exogenous noise} \\
&= p(\hat{\boldsymbol{\epsilon}} \mid \epsilon, \mathbf{z})p(\hat{\boldsymbol{\delta}} \mid \delta, \mathbf{z}) && \mathbf{x} = f(\epsilon, \mathbf{z}), \mathbf{y} = g(\delta, \mathbf{z}).
\end{aligned} \tag{5.13}$$

Using the above facts, we then show that $p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \mathbf{z}) = p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}})$:

$$\begin{aligned}
p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \mathbf{z}) &= \int p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}}, \epsilon, \delta \mid \mathbf{z}) d\epsilon d\delta && \text{By marginalization} \\
&= \int p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \mathbf{z}, \epsilon, \delta)p(\epsilon, \delta \mid \mathbf{z}) d\epsilon d\delta \\
&= \int p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \mathbf{z}, \epsilon, \delta)p(\epsilon, \delta) d\epsilon d\delta && \text{By definition of the data generating process} \\
&= \int p(\hat{\boldsymbol{\epsilon}} \mid \mathbf{z}, \epsilon)p(\hat{\boldsymbol{\delta}} \mid \mathbf{z}, \delta)p(\epsilon, \delta) d\epsilon d\delta && \text{By eq. (5.13)} \\
&= \int p(\hat{\boldsymbol{\epsilon}} \mid \epsilon)p(\hat{\boldsymbol{\delta}} \mid \delta)p(\epsilon, \delta) d\epsilon d\delta && \text{By eqs. (5.11) and (5.12)}
\end{aligned}$$

$$\begin{aligned}
p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}}) &= \int p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \mathbf{z}, \epsilon, \delta)p(\mathbf{z}, \epsilon, \delta) d\mathbf{z} d\epsilon d\delta && \text{By marginalization} \\
&= \int p(\hat{\boldsymbol{\epsilon}} \mid \mathbf{z}, \epsilon)p(\hat{\boldsymbol{\delta}} \mid \mathbf{z}, \delta)p(\mathbf{z}, \epsilon, \delta) d\mathbf{z} d\epsilon d\delta && \text{By eq. (5.13)} \\
&= \int p(\hat{\boldsymbol{\epsilon}} \mid \epsilon)p(\hat{\boldsymbol{\delta}} \mid \delta)p(\mathbf{z}, \epsilon, \delta) d\mathbf{z} d\epsilon d\delta && \text{By eqs. (5.11) and (5.12)} \\
&= \int p(\hat{\boldsymbol{\epsilon}} \mid \epsilon)p(\hat{\boldsymbol{\delta}} \mid \delta)p(\epsilon, \delta \mid \mathbf{z})p(\mathbf{z}) d\mathbf{z} d\epsilon d\delta
\end{aligned}$$

$$
\begin{aligned}
&= \int p(\hat{\boldsymbol{\epsilon}} \mid \epsilon) p(\hat{\boldsymbol{\delta}} \mid \delta) p(\epsilon, \delta) p(\mathbf{z}) d\mathbf{z} d\epsilon d\delta && \text{By definition of the data generating process} \\
&= \int p(\hat{\boldsymbol{\epsilon}} \mid \epsilon) p(\hat{\boldsymbol{\delta}} \mid \delta) p(\epsilon, \delta) \left(\int p(\mathbf{z}) d\mathbf{z} \right) d\epsilon d\delta \\
&= \int p(\hat{\boldsymbol{\epsilon}} \mid \epsilon) p(\hat{\boldsymbol{\delta}} \mid \delta) p(\epsilon, \delta) d\epsilon d\delta
\end{aligned}
$$

Using all of the above facts, it follows that $\hat{\boldsymbol{\delta}} \perp\!\!\!\perp \hat{\boldsymbol{\epsilon}} \Rightarrow \hat{\boldsymbol{\delta}} \perp\!\!\!\perp \hat{\boldsymbol{\epsilon}} \mid \mathbf{z}$:

$$
\begin{aligned}
p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \mathbf{z}) &= p(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}}) \\
&= p(\hat{\boldsymbol{\epsilon}}) p(\hat{\boldsymbol{\delta}}) && \text{Since } \hat{\boldsymbol{\delta}} \perp\!\!\!\perp \hat{\boldsymbol{\epsilon}} \\
&= p(\hat{\boldsymbol{\epsilon}} \mid \mathbf{z}) p(\hat{\boldsymbol{\delta}} \mid \mathbf{z}) && \text{By eqs. (5.9) and (5.10),}
\end{aligned}
$$

thus satisfying the sequence of implications in eq. (5.8).

We have thus far established that under $\mathcal{H}_1$, $\hat{\boldsymbol{\delta}} \not\perp\!\!\!\perp \hat{\boldsymbol{\epsilon}}$, but under $\mathcal{H}_0$, $\hat{\boldsymbol{\delta}} \perp\!\!\!\perp \hat{\boldsymbol{\epsilon}}$. Therefore, given a marginal independence test $\psi(\mathcal{D}_{\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}}}, \alpha) : (\mathbb{R} \times \mathbb{R})^N \times [0, 1] \to \{0, 1\}$ that is known to achieve power greater than level α, using ψ with a dataset of samples from $\hat{q}(\hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{\delta}} \mid \mathbf{x}, \mathbf{y}, \mathbf{z})$ will result in a conditional test with power greater than α.

□

5.5 Experiments

We analyze the performance of DIET on several synthetic and real datasets and compare it to well-studied methods designed to make CRTs tractable. First we detail the setup of both DIET and each baseline CRT, then the setup of each experiment. Finally we explore each result in detail.

5.5.1 CRT setup

Here we detail the setup of DIET and several baseline CRTs: the d_0-CRT, d_I-CRT, and the HRT.

DIET SETUP. Each of $\hat{Q}_{\text{CDF}}(\mathbf{y} \mid \mathbf{z}; \theta)$ and $\hat{Q}_{\text{CDF}}(\mathbf{x} \mid \mathbf{z}; \eta)$ is modeled using MDNs. We give details about modelling $\hat{Q}_{\text{CDF}}(\mathbf{y} \mid \mathbf{z}; \theta)$, but the model for $\hat{Q}_{\text{CDF}}(\mathbf{x} \mid \mathbf{z}; \eta)$ is identical. The network consists of six consecutive fully-connected layers each followed by batch normalization and ReLU activation. For each input $\mathbf{z}^{(i)}$, the neural network outputs mixture parameters π_θ, mean parameters μ_θ, and variance parameters σ_θ, each consisting of K dimensions. Then, the log-likelihood of $\mathbf{y}^{(i)} \mid \mathbf{z}^{(i)}$ is computed as:

$$\log \sum_{k=1}^{K} \pi_\theta^{(k)} \mathcal{N}(\mathbf{y}^{(i)}; \mu_\theta^{(k)}, \sigma_\theta^{(k)}).$$

The training objective involves maximizing average of this quantity over all samples $(\mathbf{z}^{(i)}, \mathbf{y}^{(i)})$ in the dataset with respect to the parameters $\theta := \{\pi_\theta, \mu_\theta, \sigma_\theta\}$. This is shown in eq. (5.3). Letting Φ be the CDF of a standard normal random variable, the empirical CDF implied by parameters θ evaluated at a point $(\mathbf{z}^{(i)}, \mathbf{y}^{(i)})$ is:

$$\hat{Q}_{\text{CDF}}(\mathbf{y} = \mathbf{y}^{(i)} \mid \mathbf{z} = \mathbf{z}^{(i)}; \theta) = \sum_{k=1}^{K} \pi_\theta^{(k)} \Phi\left(\frac{\mathbf{y}^{(i)} - \mu_\theta^{(k)}}{\sigma_\theta^{(k)}}\right).$$

We employ the Adam [Kingma and Ba 2014] optimizer with an initial learning rate of 1×10^{-3}. In our experiments, we fix $K = 10$. Our choice of marginal dependence statistic ρ discretizes $\hat{\boldsymbol{\epsilon}}$ and $\hat{\boldsymbol{\delta}}$, then applies the adjusted mutual information estimator from Vinh et al. [2009].

d_0-CRT SETUP. Here we review the full p-value computation for d_0-CRTs. We implement the Lasso-based models prescribed by Liu and Janson [2020]. This involves first fitting two regressions with ℓ_1 regularization:

$$\arg\min_\theta \sum_{i=1}^{N} (\mathbf{y}^{(i)} - \mathbf{z}^{(i)}\theta)^2 + \lambda_\theta ||\theta||_1, \quad \arg\min_\eta \sum_{i=1}^{N} (\widetilde{\mathbf{x}}^{(i)} - \mathbf{z}^{(i)}\eta)^2 + \lambda_\eta ||\eta||_1.$$

The regularization coefficients λ_θ and λ_η are found using 5-fold cross-validation. The test statistics $T(\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}})$ and $T(\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m)})$ are computed as follows:

$$T(\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}) = \left(\frac{\sum_{i=1}^{N} (\mathbf{y}^{(i)} - \mathbf{z}^{(i)}\theta)(\mathbf{x}^{(i)} - \mathbf{z}^{(i)}\eta)}{\sum_{i=1}^{N} (\mathbf{x}^{(i)} - \mathbf{z}^{(i)}\eta)^2} \right)^2$$
$$T(\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m)}) = \left(\frac{\sum_{i=1}^{N} (\mathbf{y}^{(i)} - \mathbf{z}^{(i)}\theta)(\widetilde{\mathbf{x}}^{(i,m)} - \mathbf{z}^{(i)}\eta)}{\sum_{i=1}^{N} (\widetilde{\mathbf{x}}^{(i,m)} - \mathbf{z}^{(i)}\eta)^2} \right)^2,$$

where $\widetilde{\mathbf{x}}^{(i,m)}$ is the ith sample of $\widetilde{\mathbf{x}}$ in $\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}$. Finally, the p-value for the d_0-CRT is computed as:

$$\frac{1}{M+1} \left(1 + \sum_{m=1}^{M} \mathbb{1}(T(\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}) \leq T(\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m)})) \right).$$

d_I-CRT SETUP. Here we review the full p-value computation for d_I-CRTs. We implement the method used in Liu and Janson [2020]. First, the following regressions are fit:

$$\arg\min_{\theta} \sum_{i=1}^{N} (\mathbf{y}^{(i)} - \mathbf{z}^{(i)}\theta)^2 + \lambda_\theta ||\theta||_1, \quad \arg\min_{\eta} \sum_{i=1}^{N} (\widetilde{\mathbf{x}}^{(i)} - \mathbf{z}^{(i)}\eta)^2 + \lambda_\eta ||\eta||_1.$$

The regularization coefficients λ_θ and λ_η are found using 5-fold cross-validation.

The test statistic $T(\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}})$ is computed in the following manner. First, the "top k" dimensions in $\mathbf{z}$ are selected using a Lasso heuristic. Let the set of the top k dimensions be called S_k. The dimensions of $\mathbf{z}$ in S_k are those with the highest corresponding $|\theta_j|$, where θ_j is the jth coordinate of θ. The d_I-CRT then fits a model from $(\mathbf{x} - d_{\mathbf{x}}, d_{\mathbf{y}}, \mathbf{z}_{\text{top}(k)})$ to $\mathbf{y}$. To explicitly involve first-order interactions, the d_I-CRT we implement includes interaction terms between $(\mathbf{x} - d_{\mathbf{x}})$ and each $\mathbf{z}_j \in \mathbf{z}_{\text{top}(k)}$. Using these interaction terms, the following regression is fit:

$$\arg\min_{\beta, \{\beta_j\}_{j \in S_k}} \sum_{i=1}^{N} \left((\mathbf{y}^{(i)} - \mathbf{z}^{(i)}\theta) - \beta(\mathbf{x}^{(i)} - \mathbf{z}^{(i)}\eta) - \sum_{j \in S_k} \beta_j \mathbf{z}_j^{(i)} (\mathbf{x}^{(i)} - \mathbf{z}^{(i)}\eta) \right)^2.$$

Finally, $T(\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}) := \beta^2 + \frac{1}{k} \sum_{j \in S_k} \beta_j^2$. This second regression is fit during each evaluation of the

test statistic on dataset $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$.

The test statistic $T(\mathcal{D}_{\tilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m)})$ is computed identically, but with samples from $\mathcal{D}_{\tilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m)}$ instead. The p-value is computed in the same way as the d_0-CRT. Since the Lasso heuristic requires a choice of hyperparameter k, we use $k = 2 \log d_{\mathbf{z}}$, where $d_{\mathbf{z}}$ is the number of coordinates in $\mathbf{z}$, as recommended by Liu and Janson [2020].

HRT SETUP. Finally we review the full p-value computation for the HRTs used in our experiments. We use the cross-validated HRT from Tansey et al. [2018a], who show it achieves higher power than the standard HRT. First, the dataset $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$ is split in half into a train and test set: $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}{}^{(\text{train})}$ and $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}{}^{(\text{test})}$. The null datasets $\{\mathcal{D}_{\tilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m,\text{train})}\}_{m=1}^{M}$ are correspondingly split into sets $\{\mathcal{D}_{\tilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m,\text{train})}\}_{m=1}^{M}$ and $\{\mathcal{D}_{\tilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m,\text{test})}\}_{m=1}^{M}$. Then, the model $\hat{q}_{\text{model}}(\mathbf{y} \mid \mathbf{x}, \mathbf{z})$, a neural network in this case, is fit using $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}{}^{(\text{train})}$. We use the same training setup as with the MDNs in DIET. P-values are then computed using only the test sets.

To compute $T(\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}{}^{(\text{test})})$, we let:

$$T(\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}{}^{(\text{test})}) = \frac{1}{N/2} \sum_{i=1}^{N} \mathcal{L}(\hat{q}_{\text{model}}, \mathbf{y}_{\text{test}}^{(i)}, \mathbf{x}_{\text{test}}^{(i)}, \mathbf{z}_{\text{test}}^{(i)}),$$

where $\mathcal{L}$ is a loss function evaluated using $\hat{q}_{\text{model}}$ and a sample from $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}{}^{(\text{test})}$. When response $\mathbf{y}$ is a continuous random variable:

$$\mathcal{L}(\hat{q}_{\text{model}}, \mathbf{y}_{\text{test}}^{(i)}, \mathbf{x}_{\text{test}}^{(i)}, \mathbf{z}_{\text{test}}^{(i)}) = (\mathbf{y}^{(i)} - \hat{\mathbf{y}}^{(i)})^2,$$

where $\hat{\mathbf{y}}^{(i)}$ is the predicted value of $\hat{q}_{\text{model}}(\mathbf{y} \mid \mathbf{x} = \mathbf{x}_{\text{test}}^{(i)}, \mathbf{z} = \mathbf{z}_{\text{test}}^{(i)})$. If $\mathbf{y}$ is discrete, the loss function is the log-probability of observing $\mathbf{y}$ given $\mathbf{x}$ and $\mathbf{z}$:

$$\mathcal{L}(\hat{q}_{\text{model}}, \mathbf{y}_{\text{test}}^{(i)}, \mathbf{x}_{\text{test}}^{(i)}, \mathbf{z}_{\text{test}}^{(i)}) = \log \hat{q}_{\text{model}}(\mathbf{y} = \mathbf{y}_{\text{test}}^{(i)} \mid \mathbf{x} = \mathbf{x}_{\text{test}}^{(i)}, \mathbf{z} = \mathbf{z}_{\text{test}}^{(i)}).$$

The null statistic $T(\mathcal{D}_{\tilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(\text{test})})$ is computed in a similar way with the same $\hat{q}_{\text{model}}$.

Next, a p-value, $\hat{\mathbf{p}}_1$, of the HRT is computed by

$$\frac{1}{M+1}\left(1+\sum_{m=1}^{M}\mathbb{1}(T(\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}{}^{(\text{test})})\geq T(\mathcal{D}_{\tilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m,\text{test})}))\right).$$

Finally, to compute a cross-validated p-value using the HRT, we repeat all the steps above to obtain another p-value $\hat{\mathbf{p}}_2$, but exchanging the roles of the train and test sets. These two p-values $\hat{\mathbf{p}}_1$ and $\hat{\mathbf{p}}_2$ are combined by taking $\min(1, 2\cdot\min(\hat{\mathbf{p}}_1, \hat{\mathbf{p}}_2))$.

5.5.2 Experimental setup.

Single hypotheses. Each synthetic experiment follows the same basic structure for a single run, unless specified otherwise. First, a dataset $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$ is sampled. Then, each method is used to test the hypothesis $\mathbf{x} \perp\!\!\!\perp \mathbf{y} \mid \mathbf{z}$ and a p-value is computed using $M = 100$ null datasets. We perform 100 runs of each synthetic experiment and report aggregate results.

The power of each method at a specific rejection threshold α is estimated by computing the percentage of times a hypothesis is rejected, over the 100 runs. A hypothesis is rejected if the p-value $\hat{\mathbf{p}} \leq \alpha$.

Multiple hypotheses. For controlled variable selection experiments, we test the hypothesis $\mathbf{x}_j \perp\!\!\!\perp \mathbf{y} \mid \mathbf{x}_{-j}$ for each dimension j of the covariate vector $\mathbf{x}$. We then apply the Benjamini-Hochberg procedure [Benjamini and Hochberg 1995] to account for multiple testing while controlling the FDR.

Given a set of d covariates $\mathbf{x} = \{\mathbf{x}_1, \ldots, \mathbf{x}_d\}$ and a response $\mathbf{y}$, we test the conditional independence of each coordinate $\mathbf{x}_j$ with $\mathbf{y}$ having observed all other coordinates of $\mathbf{x}_{-j}$. For simplicity, we focus on the CI test for only a single coordinate $\mathbf{x}_j$ in this section. The procedure for the other coordinates is identical. For consistency with previous sections we refer to $\mathbf{x}_j$ as $\mathbf{x}$ and

$\mathbf{x}_{-j}$ as $\mathbf{z}$.

Every CRT method assumes the ability to sample from $p(\mathbf{x} \mid \mathbf{z})$ but in some of our experiments we do not allow access to this distribution. DIET with MDNs can directly model $p(\mathbf{x} \mid \mathbf{z})$, so its approximation can be used to sample null datasets $\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}$. However, neither the DCRTs nor the HRT have this facility. For these models, we use deep generative models to sample from $p(\mathbf{x} \mid \mathbf{z})$ [Romano et al. 2020; Sudarshan et al. 2020; Jordon et al. 2018].

Romano et al. [2020] train a generative model $\hat{q}_{\text{knockoff}}(\widetilde{\mathbf{x}}, \widetilde{\mathbf{z}} \mid \mathbf{x}, \mathbf{z})$ from samples of $(\mathbf{x}, \mathbf{z})$, which models $(\widetilde{\mathbf{x}}, \widetilde{\mathbf{z}}) \mid (\mathbf{x}, \mathbf{z})$, where $\widetilde{\mathbf{x}}$ and $\widetilde{\mathbf{z}}$ are random variables that satisfy the following property:

$$[\widetilde{\mathbf{x}}, \widetilde{\mathbf{z}}, \mathbf{x}, \mathbf{z}] \overset{d}{=} [\mathbf{x}, \widetilde{\mathbf{z}}, \widetilde{\mathbf{x}}, \mathbf{z}] \overset{d}{=} [\widetilde{\mathbf{x}}, \mathbf{z}, \mathbf{x}, \widetilde{\mathbf{z}}] \overset{d}{=} [\mathbf{x}, \mathbf{z}, \widetilde{\mathbf{x}}, \widetilde{\mathbf{z}}]. \qquad \text{(swap property)}$$

The model $\hat{q}_{\text{knockoff}}$ can then be used to generate a null dataset $\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}$. The ith sample of $\widetilde{\mathbf{x}}$ in $\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}$ is sampled by drawing $\widetilde{\mathbf{x}}^{(i)}, \widetilde{\mathbf{z}}^{(i)}$ from $\hat{q}_{\text{knockoff}}(\widetilde{\mathbf{x}}, \widetilde{\mathbf{z}} \mid \mathbf{x} = \mathbf{x}^{(i)}, \mathbf{z} = \mathbf{z}^{(i)})$, then discarding $\widetilde{\mathbf{z}}^{(i)}$. Due to the swap property, the sample $\widetilde{\mathbf{x}}^{(i)} \mid \mathbf{z}^{(i)} \overset{d}{=} \mathbf{x}^{(i)} \mid \mathbf{z}^{(i)}$, but is conditionally independent of $\mathbf{y}^{(i)} \mid \mathbf{z}^{(i)}$. This makes $\widetilde{\mathbf{x}}^{(i)}$ drawn from $\hat{q}_{\text{knockoff}}$ a valid null sample when used in each Model-X method's p-value computation. The null datasets $\{\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(m)}\}_{m=1}^{M}$ can be drawn the same way.

It is critical to note that if type-1 error is to be controlled using the conditions laid out by prop. 5.4.1, sample splitting is required. Since the proof of prop. 5.4.1 requires that the same function W be applied to the sequence

$$W(\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}), W(\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(1)}), \ldots, W(\mathcal{D}_{\widetilde{\mathbf{x}},\mathbf{y},\mathbf{z}}{}^{(M)}),$$

any estimator for $p(\mathbf{x} \mid \mathbf{z})$ must be fit using a separate dataset. As such, we split the dataset $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}$ into a train set $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}{}^{(\text{train})}$ and a test set $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}}{}^{(\text{test})}$. We fit models for $p(\mathbf{x} \mid \mathbf{z})$ and the HRT model $\hat{q}_{\text{model}}$ using the training set, then compute p-values using the test set. Each synthetic variable selection experiment is run 100 times. We set $M = 2000$.

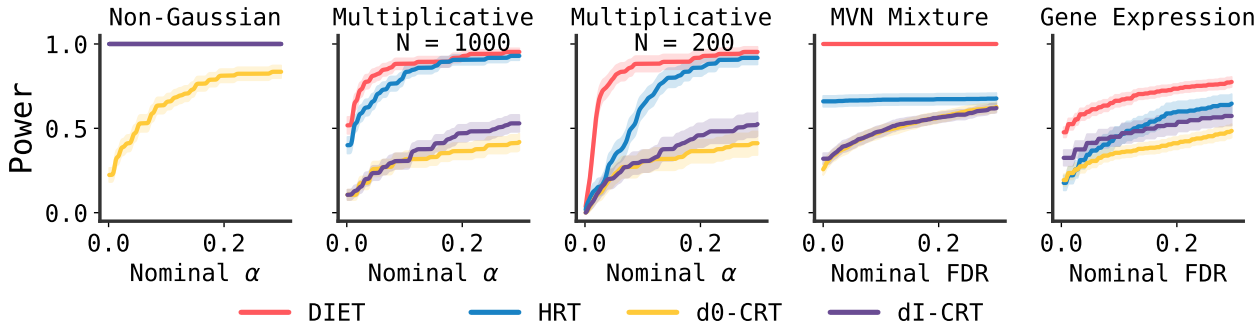


Figure 5.1: DIET achieves high power across numerous synthetic benchmarks. In this figure, we show the power of each method as a function of nominal type-1 error rate α or FDR in the case of variable selection.

5.5.3 Synthetic experiments

Here we describe the benchmarks used to compare DIET to each of the baselines.

5.5.3.1 Univariate Gaussian data

This experiment is designed mainly to confirm that each method performs as intended. The data is drawn as follows: $\mathbf{z} \sim \mathcal{N}(0, 0.1)$, $\mathbf{x} \mid \mathbf{z} \sim \mathcal{N}(\mathbf{z}, 0.1)$, and $\mathbf{y} \mid \mathbf{x}, \mathbf{z} \sim \mathcal{N}(\mathbf{x}+\mathbf{z}, 0.1)$. The training dataset consists of 500 samples.

Results: As expected, the estimated power of each method is 1 for $\alpha \in (0, 0.3]$. We do not explore larger α, as a practitioner would realistically set their nominal error rate within this range. As a graph is unnecessary to visualize this result, we omit it.

Non-Gaussian and multiplicative data. These experiments are designed primarily to understand the effect of violating an additivity assumption in the data generating process. Using noise $\varepsilon \sim \mathcal{N}(0, 0.025)$ and coefficients $\beta \in \mathbb{R}^{100}$ where $\beta_j \sim \mathcal{N}(0, 0.025)$,

$$\begin{aligned} \mathbf{z} &\sim \mathcal{N}(0, 0.025 \cdot I_{100}), \mathbf{x} \mid \mathbf{z} \sim \mathcal{N}\left(\textstyle\sum_{j=1}^{10} \mathbf{z}_j, 1\right) \\ \mathbf{y} \mid \mathbf{x}, \mathbf{z}, \varepsilon &= (\mathbf{x} + \varepsilon + \textstyle\sum_{j=5}^{100} \mathbf{z}_j \beta_j)^3 \end{aligned} \qquad \text{(Non-Gaussian)}$$

$$\mathbf{z} \sim \mathcal{N}(0, 0.1 \cdot I_{100}), \mathbf{x} \sim \mathcal{N}(0, 0.1)$$
$$\mathbf{y} \mid \mathbf{z}, \mathbf{x} \sim \mathcal{N}(\mathbf{z}_1 \mathbf{x}, 0.025) \qquad \text{(Multiplicative)}$$

Both datasets consist of 1000 samples.

Results: We observe that each CRT manages to control the type-1 error rate at or below nominal levels. In terms of power, most methods perform well on the non-Gaussian dataset, as shown in the first column of fig. 5.1. All but the d_0-CRT are able to achieve full power for almost every $\alpha \in (0, 0.3]$.

In the case of multiplicative data, there is a clear deterioration in the performance of the d_I-CRT, as shown in the second column of fig. 5.1. The d_I-CRT achieves marginally higher power, but is still quite far from DIET or HRT. Upon investigation, we observed that the heuristic used to choose dimensions in $\mathbf{z}$ in d_I-CRT only selects $\mathbf{z}_1$ at random. Since DCRTs forbid using samples of the triple $(\mathbf{x}, \mathbf{y}, \mathbf{z})$ during training, it is difficult to choose a robust heuristic. Next we explore why DIET achieves higher power using a toy example.

SHORTCOMINGS OF THE d_I-CRT: A WORKED EXAMPLE. Consider the following example:

$$\mathbf{x} \sim \mathcal{N}(\mathbf{x}; 0, \sigma_{\mathbf{x}}^2)$$
$$\mathbf{z}_j \sim \mathcal{N}(\mathbf{z}_j; 0, 1) \, \forall j \in \{1, \ldots, d\}$$
$$\mathbf{y} \mid \mathbf{x}, \mathbf{z} \sim \mathcal{N}(\mathbf{y}; \beta_1 \mathbf{x} \mathbf{z}_1 + \sum_{j=2}^{d} \beta_j \mathbf{z}_j, 1)$$
$$\beta_1, \ldots, \beta_d \in \mathbb{R}, \; |\beta_1| < \cdots < |\beta_d|.$$

Since this example extends the motivating example for d_I-CRTs from Liu and Janson [2020], we focus only on the behavior of the d_I-CRT here. Recall the d_I-CRT test statistic computation:

1. The d_I-CRT first identifies a subset of k variables in $\mathbf{z}$ with which to explicitly compute

interaction terms. This is done by fitting a regression from $\mathbf{z}$ to $\mathbf{y}$, then using some measure of feature importance to select the top k most important features, $\mathbf{z}_{\text{top}(k)}$

2. The distillation function $d_{\mathbf{y}} = \mathbb{E}[\mathbf{y} \mid \mathbf{z}]$ is computed
3. Then, the distillation function $d_{\mathbf{x}} = \mathbb{E}[\mathbf{x} \mid \mathbf{z}]$ is computed
4. Next, a model from $(\mathbf{x} - d_{\mathbf{x}}, d_{\mathbf{y}}, \mathbf{z}_{\text{top}(k)})$ to $\mathbf{y}$ is fit
5. Finally, a measure of feature importance for $\mathbf{x} - d_{\mathbf{x}}$ in this model is used to compute the test statistic T

To compute each null statistic, steps 3-5 are repeated using the null datasets. Given the set of M null statistics and the test statistic T, a p-value is computed as shown in the CRT p-value computation eq. (5.2). Now, observe the behavior of the d_I-CRT in this example.

First, a model is fit from $\mathbf{z}$ to $\mathbf{y}$. This is equivalent to estimating the function $\mathbb{E}[\mathbf{y} \mid \mathbf{z}]$. To see the functional form of this quantity let's first evaluate the density $f(\mathbf{y} \mid \mathbf{z})$:

$$
\begin{aligned}
f(\mathbf{y} \mid \mathbf{z}) &= \int_{-\infty}^{\infty} f(\mathbf{y} \mid \mathbf{x}, \mathbf{z}) f(\mathbf{x} \mid \mathbf{z}) d\mathbf{x} \\
&= \int_{-\infty}^{\infty} f(\mathbf{y} \mid \mathbf{x}, \mathbf{z}) f(\mathbf{x}) d\mathbf{x} \\
&= \int_{-\infty}^{\infty} \frac{\mathrm{e}^{-\frac{\left(-\beta_1 \mathbf{z}_1 \mathbf{x} - \sum_{j=2}^{d} \beta_j \mathbf{z}_j + \mathbf{y}\right)^2}{2} - \frac{\mathbf{x}^2}{2{\sigma_{\mathbf{x}}}^2}}}{\sqrt{4\pi^2 {\sigma_{\mathbf{x}}}^2}} \, \mathrm{d}\mathbf{x} \\
&= \int_{-\infty}^{\infty} \frac{\mathrm{e}^{-\frac{\left(-M_1 \mathbf{x} + M_y\right)^2}{2} - \frac{\mathbf{x}^2}{2{\sigma_{\mathbf{x}}}^2}}}{\sqrt{4\pi^2 {\sigma_{\mathbf{x}}}^2}} \, \mathrm{d}\mathbf{x} \qquad \{\text{letting } M_1 = \beta_1 \mathbf{z}_1, M_y = \mathbf{y} - \sum_{j=2}^{d} \beta_j \mathbf{z}_j\} \\
&= \int_{-\infty}^{\infty} \frac{\mathrm{e}^{-\frac{M_1^2 \mathbf{x}^2 - 2 M_y M_1 \mathbf{x} + M_y^2}{2} - \frac{\mathbf{x}^2}{2{\sigma_{\mathbf{x}}}^2}}}{\sqrt{4\pi^2 {\sigma_{\mathbf{x}}}^2}} \, \mathrm{d}\mathbf{x} \\
&= \frac{\mathrm{e}^{-M_y^2/2}}{\sqrt{4\pi^2 {\sigma_{\mathbf{x}}}^2}} \int_{-\infty}^{\infty} \mathrm{e}^{-\frac{M_1^2 \mathbf{x}^2 - 2 M_y M_1 \mathbf{x}}{2} - \frac{\mathbf{x}^2}{2{\sigma_{\mathbf{x}}}^2}} \, \mathrm{d}\mathbf{x} \\
&= \frac{\mathrm{e}^{-M_y^2/2}}{\sqrt{4\pi^2 {\sigma_{\mathbf{x}}}^2}} \int_{-\infty}^{\infty} \mathrm{e}^{-\frac{1}{2}\left(\mathbf{x}^2\left(M_1^2 + \frac{1}{{\sigma_{\mathbf{x}}}^2}\right) - 2 M_y M_1 \mathbf{x} + \frac{M_y^2 M_1^2}{\left(M_1^2 + \frac{1}{{\sigma_{\mathbf{x}}}^2}\right)} - \frac{M_y^2 M_1^2}{\left(M_1^2 + \frac{1}{{\sigma_{\mathbf{x}}}^2}\right)}\right)} \, \mathrm{d}\mathbf{x}
\end{aligned}
$$

$$
\begin{aligned}
&= \frac{\mathrm{e}^{-M_y^2/2}}{\sqrt{4\pi^2{\sigma_\mathbf{x}}^2}} \int_{-\infty}^{\infty} \mathrm{e}^{-\frac{1}{2}\left(\mathbf{x}\left(M_1^2+\frac{1}{{\sigma_\mathbf{x}}^2}\right)^{1/2}-\frac{M_y M_1}{\left(M_1^2+\frac{1}{{\sigma_\mathbf{x}}^2}\right)^{1/2}}\right)^2+\frac{M_y^2 M_1^2}{\left(M_1^2+\frac{1}{{\sigma_\mathbf{x}}^2}\right)}} \mathrm{d}\mathbf{x} \\
&= \frac{\mathrm{e}^{-M_y^2/2+\frac{M_y^2 M_1^2}{2\left(M_1^2+\frac{1}{{\sigma_\mathbf{x}}^2}\right)}}}{\sqrt{4\pi^2{\sigma_\mathbf{x}}^2}} \int_{-\infty}^{\infty} \mathrm{e}^{-\frac{\left(M_1^2+\frac{1}{{\sigma_\mathbf{x}}^2}\right)}{2}\left(\mathbf{x}-\frac{M_y M_1}{\left(M_1^2+\frac{1}{{\sigma_\mathbf{x}}^2}\right)}\right)^2} \mathrm{d}\mathbf{x} \\
&= \frac{\mathrm{e}^{-M_y^2\left(\frac{1}{2}-\frac{1}{2\left(1+\frac{1}{M_1^2{\sigma_\mathbf{x}}^2}\right)}\right)}}{\sqrt{4\pi^2{\sigma_\mathbf{x}}^2}} \int_{-\infty}^{\infty} \mathrm{e}^{-\frac{\left(M_1^2+\frac{1}{{\sigma_\mathbf{x}}^2}\right)}{2}\left(\mathbf{x}-\frac{M_y M_1}{\left(M_1^2+\frac{1}{{\sigma_\mathbf{x}}^2}\right)}\right)^2} \mathrm{d}\mathbf{x} \\
&= \frac{\mathrm{e}^{-M_y^2\left(\frac{1}{2}-\frac{1}{2\left(1+\frac{1}{M_1^2{\sigma_\mathbf{x}}^2}\right)}\right)}}{\sqrt{4\pi^2{\sigma_\mathbf{x}}^2}} \sqrt{\frac{2\pi}{\left(M_1^2+\frac{1}{{\sigma_\mathbf{x}}^2}\right)}} \\
&= \frac{\mathrm{e}^{-M_y^2\left(\frac{\left(1+\frac{1}{M_1^2{\sigma_\mathbf{x}}^2}\right)-1}{2\left(1+\frac{1}{M_1^2{\sigma_\mathbf{x}}^2}\right)}\right)}}{\sqrt{2\pi\left({\sigma_\mathbf{x}}^2 M_1^2+1\right)}} \\
&= \frac{\mathrm{e}^{-\frac{M_y^2}{2\left({\sigma_\mathbf{x}}^2 M_1^2+1\right)}}}{\sqrt{2\pi\left({\sigma_\mathbf{x}}^2 M_1^2+1\right)}} \\
&= \frac{\mathrm{e}^{-\frac{\left(\mathbf{y}-\sum_{j=2}^d \beta_j \mathbf{z}_j\right)^2}{2\left({\sigma_\mathbf{x}}^2 M_1^2+1\right)}}}{\sqrt{2\pi\left({\sigma_\mathbf{x}}^2 M_1^2+1\right)}} \\
&= \mathcal{N}(\mathbf{y}; \sum_{j=2}^{d} \beta_j \mathbf{z}_j, 1+\beta_1^2 \sigma_\mathbf{x}^2 \mathbf{z}_1^2).
\end{aligned}
$$

This is a Gaussian distribution with mean $\mathbb{E}[\mathbf{y} \mid \mathbf{z}] = \sum_{j=2}^{d} \beta_j \mathbf{z}_j$, which is not a function of $\mathbf{z}_1$. Therefore, $\mathbf{z}_{\text{top}(k)}$ will not include $\mathbf{z}_1$ for any $k < d$. To compute $d_\mathbf{x} = \mathbb{E}[\mathbf{x} \mid \mathbf{z}]$, note that $\mathbf{x}$ and $\mathbf{z}$ are independent, and $\mathbb{E}[\mathbf{x}] = 0$.

Next, let's consider a model from $(\mathbf{x} - d_\mathbf{x}, d_\mathbf{y}, \mathbf{z}_{\text{top}(k)})$ to $\mathbf{y}$. Again, this is equivalent to estimating $\mathbb{E}[\mathbf{y} \mid \mathbf{x} - d_\mathbf{x}, d_\mathbf{y}, \mathbf{z}_{\text{top}(k)}] = \mathbb{E}[\mathbf{y} \mid \mathbf{x}, \sum_{j=2}^{d} \beta_j \mathbf{z}_j, \mathbf{z}_{\text{top}(k)}]$. Since $\mathbf{z}_1$ is not in the conditioning set of this expectation, it reduces to $\mathbb{E}[\mathbf{y} \mid \sum_{j=2}^{d} \beta_j \mathbf{z}_j, \mathbf{z}_{\text{top}(k)}]$; this follows from expanding the

conditional expectation and noting $\mathbb{E}[\mathbf{z}_1] = 0$. Thus any model from $(\mathbf{x} - d_{\mathbf{x}}, d_{\mathbf{y}}, \mathbf{z}_{\text{top}(k)})$ to $\mathbf{y}$ will assign no feature importance to $\mathbf{x} - d_{\mathbf{x}}$. Assuming that a feature importance score of 0 indicates an unimportant feature, the score assigned to $\mathbf{x} - d_{\mathbf{x}}$ will be 0.

The same holds true when repeating the d_I-CRT steps 3-5 with the null datasets. Regardless of what values of $\mathbf{x}$ are used in the model that estimates $\mathbb{E}[\mathbf{y} \mid \sum_{j=2}^{d} \beta_j \mathbf{z}_j, \mathbf{z}_{\text{top}(k)}]$, the importance score of $\mathbf{x} - d_{\mathbf{x}}$ will always be zero. Since the distribution of the test statistic is indistinguishable from the distribution of the null statistics, the d_I-CRT will achieve power no greater than the size of the test.

Next, consider the case of DIET. Recall that its test statistic uses the dataset

$$D_{\mathbf{x},\mathbf{y},\mathbf{z}} = \{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}, \mathbf{z}^{(i)})\}_{i=1}^{n}$$

to compute samples of $\hat{\boldsymbol{\delta}} = F_{\mathbf{y}|\mathbf{z}}(\mathbf{y}, \mathbf{z})$ and $\hat{\boldsymbol{\epsilon}} = F_{\mathbf{x}|\mathbf{z}}(\mathbf{x}, \mathbf{z}) = F_{\mathbf{x}}(\mathbf{x})$, then uses these samples to estimate the marginal dependence between $\hat{\boldsymbol{\delta}}$ and $\hat{\boldsymbol{\epsilon}}$. We will now show that in the example above, $\hat{\boldsymbol{\delta}}$ and $\hat{\boldsymbol{\epsilon}}$ will be dependent using the true data $D_{\mathbf{x},\mathbf{y},\mathbf{z}}$, but will be independent when using the null data $D_{\tilde{\mathbf{x}},\mathbf{y},\mathbf{z}}$, yielding power > 0.

First note the following equivalences:

$$F_{\mathbf{y}|\mathbf{z}}(\mathbf{y}, \mathbf{z}) = \Phi\left(\frac{\mathbf{y} - \sum_{j=2}^{d} \mathbf{z}_j}{\sqrt{1 + \beta_1^2 \sigma_{\mathbf{x}}^2 \mathbf{z}_1^2}}\right)$$

$$F_{\mathbf{x}}(\mathbf{x}) = \Phi\left(\frac{\mathbf{x}}{\sigma_{\mathbf{x}}}\right)$$

$$\mathbf{y} = \beta_1 \mathbf{x}\mathbf{z}_1 + \sum_{j=2}^{d} \beta_j \mathbf{z}_j + \eta_{\mathbf{y}}$$

where Φ is the CDF of a standard gaussian and $\eta_{\mathbf{y}} \sim \mathcal{N}(0, 1)$. To show that $\hat{\boldsymbol{\delta}}$ and $\hat{\boldsymbol{\epsilon}}$ are dependent, we must show that

$$\mathbb{P}(\hat{\boldsymbol{\delta}} \leq a \mid \hat{\boldsymbol{\epsilon}} = b) = \mathbb{P}(\hat{\boldsymbol{\delta}} \leq a).$$

When using the true data $D_{\mathbf{x},\mathbf{y},\mathbf{z}}$, the following must hold:

$$\begin{aligned}
\mathbb{P}(\hat{\boldsymbol{\delta}} \leq a \mid \hat{\boldsymbol{\epsilon}} = b) &= \mathbb{P}\left(\Phi\left(\frac{\mathbf{y} - \sum_{j=2}^{d} \beta_j \mathbf{z}_j}{\sqrt{1+\beta_1^2\sigma_{\mathbf{x}}^2\mathbf{z}_1^2}}\right) \leq a \mid \Phi\left(\frac{\mathbf{x}}{\sigma_{\mathbf{x}}}\right) = b\right) \\
&= \mathbb{P}\left(\frac{\mathbf{y} - \sum_{j=2}^{d} \beta_j \mathbf{z}_j}{\sqrt{1+\beta_1^2\sigma_{\mathbf{x}}^2\mathbf{z}_1^2}} \leq \Phi^{-1}(a) \mid \mathbf{x} = \sigma_{\mathbf{x}}\Phi^{-1}(b)\right) \\
&= \mathbb{P}\left(\frac{\beta_1\mathbf{x}\mathbf{z}_1 + \eta_{\mathbf{y}}}{\sqrt{1+\beta_1^2\sigma_{\mathbf{x}}^2\mathbf{z}_1^2}} \leq \Phi^{-1}(a) \mid \mathbf{x} = \sigma_{\mathbf{x}}\Phi^{-1}(b)\right) \\
&= \mathbb{P}\left(\frac{\beta_1\mathbf{z}_1\sigma_{\mathbf{x}}\Phi^{-1}(b) + \eta_{\mathbf{y}}}{\sqrt{1+\beta_1^2\sigma_{\mathbf{x}}^2\mathbf{z}_1^2}} \leq \Phi^{-1}(a)\right).
\end{aligned}$$

The first equation uses the definitions of $\hat{\boldsymbol{\delta}}$ and $\hat{\boldsymbol{\epsilon}}$. The second equation uses the invertibility of the Gaussian CDF. The third equation holds because $\mathbf{y}$ can be rewritten as a function of $\mathbf{x}$, $\mathbf{z}$, and noise $\eta_{\mathbf{y}}$. Finally, the last equation uses the value of $\mathbf{x}$ as a function of b and that rvx is jointly independent of $\mathbf{z}_1$ and $\eta_{\mathbf{y}}$. Clearly, the conditional probability $\mathbb{P}(\hat{\boldsymbol{\delta}} \leq a \mid \hat{\boldsymbol{\epsilon}} = b)$ cannot be written as $\mathbb{P}(\hat{\boldsymbol{\delta}} \leq a)$ using the true data $D_{\mathbf{x},\mathbf{y},\mathbf{z}}$. This means that $\hat{\boldsymbol{\delta}}$ and $\hat{\boldsymbol{\epsilon}}$ will be dependent.

When computing the dependence of $\hat{\boldsymbol{\delta}}$ and $\hat{\boldsymbol{\epsilon}}$ using null datasets:

$$\begin{aligned}
\mathbb{P}(\hat{\boldsymbol{\delta}} \leq a \mid \hat{\boldsymbol{\epsilon}} = b) &= \mathbb{P}\left(\Phi\left(\frac{\mathbf{y} - \sum_{j=2}^{d} \beta_j \mathbf{z}_j}{\sqrt{1+\beta_1^2\sigma_{\mathbf{x}}^2\mathbf{z}_1^2}}\right) \leq a \mid \Phi\left(\frac{\tilde{\mathbf{x}}}{\sigma_{\mathbf{x}}}\right) = b\right) \\
&= \mathbb{P}\left(\frac{\mathbf{y} - \sum_{j=2}^{d} \beta_j \mathbf{z}_j}{\sqrt{1+\beta_1^2\sigma_{\mathbf{x}}^2\mathbf{z}_1^2}} \leq \Phi^{-1}(a) \mid \tilde{\mathbf{x}} = \sigma_{\mathbf{x}}\Phi^{-1}(b)\right) \\
&= \mathbb{P}\left(\frac{\beta_1\mathbf{x}\mathbf{z}_1 + \eta_{\mathbf{y}}}{\sqrt{1+\beta_1^2\sigma_{\mathbf{x}}^2\mathbf{z}_1^2}} \leq \Phi^{-1}(a) \mid \tilde{\mathbf{x}} = \sigma_{\mathbf{x}}\Phi^{-1}(b)\right) \\
&= \mathbb{P}\left(\frac{\beta_1\mathbf{x}\mathbf{z}_1 + \eta_{\mathbf{y}}}{\sqrt{1+\beta_1^2\sigma_{\mathbf{x}}^2\mathbf{z}_1^2}} \leq \Phi^{-1}(a)\right) \\
&= \mathbb{P}\left(\hat{\boldsymbol{\delta}} \leq a\right).
\end{aligned}$$

The first 3 equations follow from earlier. The 4th and 5th steps hold because $\mathbf{y}$ is not a function of $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{x}}$ is jointly independent of all other random variables. Therefore, when computing each

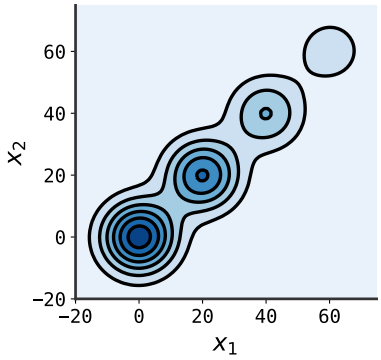


Figure 5.2: Synthetic cvs dataset

null statistic using null data $D_{\tilde{\mathbf{x}},\mathbf{y},\mathbf{z}}$, $\hat{\boldsymbol{\delta}}$ and $\hat{\boldsymbol{\epsilon}}$ will be independent.

Since DIET will identify dependence between $\hat{\boldsymbol{\delta}}$ and $\hat{\boldsymbol{\epsilon}}$ when using the true data, and no dependence when using the null data, the distribution of the test statistic will not be equal to that of each null statistic. Thus, it follows that DIET can achieve power > 0.

Then, to understand the cost of sample splitting, we reduced the sample size of the multiplicative data to 200 and re-ran our experiments. The third column of fig. 5.1 shows that the HRT suffers the greatest loss in power. This is likely due to the HRT splitting the sample and using only 100 samples during training.

5.5.3.2 Controlled variable selection

This experiment evaluates each CRT on its ability to perform controlled variable selection while using an estimated $p(\mathbf{x} \mid \mathbf{z})$ distribution. The $\mathbf{x}$ data is a 100-dimensional mixture of autoregressive Gaussians and is sampled as follows: $\mathbf{x} \sim \sum_{k=1}^{4} \pi_k \mathcal{N}(\mu_k \cdot \mathbf{1}, \Sigma_k)$. Each Σ_k is a 100-dimensional covariance matrix whose (i,j)th entry is $\rho_k^{|i-j|}$. We set

$$(\rho_1, \rho_2, \rho_3, \rho_4) = (0.7, 0.6, 0.5, 0.4)$$

$$(\pi_1, \pi_2, \pi_3, \pi_4) = (0.4, 0.3, 0.2, 0.1)$$

$$(\mu_1, \mu_2, \mu_3, \mu_4) = (0, 20, 40, 60).$$

Figure 5.2 visualizes the first two dimensions of this data. The response $\mathbf{y} \mid \mathbf{x}$ is drawn from $\mathcal{N}(\langle \mathbf{x}, \beta \rangle, 1)$, where β is a coefficient vector. Each non-zero element of β is drawn from $3 \cdot \text{Rademacher}(0.5)$; there are 20 non-zero elements chosen randomly in each run. These non-zero elements represent the important variables each method aims to recover. The dataset consists of 1000 samples.

Results: We evaluate the average power and the FDP across runs for each method in the fourth column of fig. 5.1 and fig. 5.3 respectively. The average FDP is an empirical estimate of the FDR. We notice that most methods are able to keep the average FDP below the nominal FDR rate α for $\alpha > 0.2$. However, when $\alpha \leq 0.1$, the d_I-CRT and the HRT inflate the FDP, suggesting they are sensitive to poor estimations of the $p(\mathbf{x}_j \mid \mathbf{x}_{-j})$ distributions, as shown by Sudarshan et al. [2021]. We also observe that loss of power in the HRT is mainly due to sample splitting. Using 3000 samples instead helped increase the power of the HRT closer to that of DIET.

5.5.4 Semi-synthetic genetics experiment

A common application area of Model-X methods is biology [Candes et al. 2018; Bates et al. 2020; Sudarshan et al. 2020; Sesia et al. 2019]. We evaluate each CRT using a setup similar to that of Sudarshan et al. [2020], which uses RNA expression data of 963 cancer cell lines and 20K genes per cell line from Yang et al. [2012]. The datasets $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}} \in \mathbb{R}^{963 \times 100}$ are generated as follows.

100 genes are sampled sequentially from the set of 20K such that the resulting set contains genes with strong pairwise correlations. We use a synthetic $\mathbf{y} \mid \mathbf{x}$ response function from Tansey et al. [2018a]. Specifically, to generate each dataset $\mathcal{D}_{\mathbf{x},\mathbf{y},\mathbf{z}} \in \mathbb{R}^{963 \times 100}$, we first sample a set of 100 genes $\{\mathbf{x}_j\}_{j=1}^{100}$ from a set of 20K. Let O be the running set of genes, and S be the full set of 20K genes. The first gene $\mathbf{x}_1$ is sampled uniformly from S and added to O, and removed from S. For each $j > 1$, we apply the following procedure. A gene $\mathbf{x}_k$ is drawn uniformly from O. The correlation between $\mathbf{x}_j$ and each gene in S is computed and the top 50 strongest correlated genes F are selected. The gene $\mathbf{x}_j \sim \text{Uniform}(F)$, and is added to O and removed from S. This process

is repeated until S contains 100 genes.

To sample $\mathbf{y} \mid \mathbf{x}$, we apply the following procedure defined by Liang et al. [2018]. The response has four main parts: two first order terms, a second order term, and a final nonlinearity term.

$$
\begin{aligned}
k &\in [m/4] \\
\varphi_k^{(1)}, \varphi_k^{(2)} &\sim \mathcal{N}(1, 1) \\
\varphi_k^{(3)}, \varphi_k^{(4)}, \varphi_k^{(5)}, \varphi_k^{(6)} &\sim \mathcal{N}(2, 1)
\end{aligned}
$$

$$
\mathbf{y} \mid \mathbf{x} = \epsilon + \sum_{k=1}^{m/4} \varphi_k^{(1)} \mathbf{x}_{4k-3} + \varphi_k^{(3)} \mathbf{x}_{4k-2} + \varphi_k^{(4)} \mathbf{x}_{4k-3} \mathbf{x}_{4k-2} + \varphi_k^{(5)} \tanh(\varphi_k^{(2)} \mathbf{x}_{4k-1} + \varphi_k^{(6)} \mathbf{x}_{4k}).
$$

The variable m determines the number of important features. We set m to 20 in our experiments.

We perform 30 replicates of this experiment; $\mathbf{x}_{1:20}$ are the important features in each one.

Results: We show the average power for each CRT in the last column of fig. 5.1. All methods are able to control the average FDP below the nominal level. DIET consistently achieves power higher than the baselines. We also observe that the HRT achieves higher power than the d_I-CRT at nominal FDR above 0.1. At lower nominal FDR, the HRT does not select many features as its non-null p-values are generally higher than those of the d_I-CRT.

5.5.5 Electronic health records

CRTs have found use in clinical model deployment pipelines as methods to prune a set of input features [Razavian et al. 2020]. This pruning reduces the amount of auditing and engineering needed for model deployment. We perform controlled variable selection using an electronic health record (EHR) dataset from a large metropolitan hospital to understand which variables are most predictive of an adverse event within 96 hours for patients that tested positive for COVID-

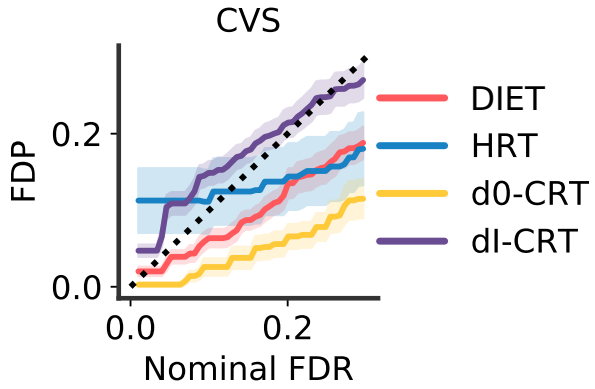


Figure 5.3: FDP of each method on synthetic cvs data.

	DIET	HRT	d_0-CRT	d_I-CRT
Selected	**60%**	40%	25%	55%

Table 5.1: DIET selects a larger portion of covariates previously identified by highly-cited medical papers. See table 5.2 for a list of selections.

19.

The data contains 28K samples with 29 features on the results of a blood test, basic vital signs, and demographics. A full list of variables is provided in table 5.2.

We run each CRT method on the EHR dataset and apply the Benjamini and Hochberg [1995] procedure, selecting covariates at a nominal FDR of 10%.

Results: To evaluate the effectiveness of the selections made by each CRT, we compare selected covariates to those reported by several papers related to adverse events in COVID-19 patients from well-known medical journals [Petrilli et al. 2020; Sattar et al. 2020; Mei et al. 2020; Castro et al. 2020; Zhang et al. 2020; Zhong and Peng 2021; Ruan et al. 2020; Zhou et al. 2020a].

To score each CRT, we consider covariates found to be important by at least one of the above papers. We compute the fraction of these covariates selected by each CRT and report them in table 5.1. We show the full list of selections in table 5.2.

DIET selects a larger percent of the important covariates, which indicates higher power. While the d_I-CRT selects almost as many, upon closer inspection, it also selects redundant features. For

example, the d_I-CRT selects both count and percentage of Eosinophils, and both High O2 support and O2 device while DIET only selects one of each.

5.6 Discussion

Existing methods to speed up model-based CRTs either make restrictive assumptions about the data generating process, use heuristics to model interactions between $\mathbf{x}$ and $\mathbf{y}$, or lose power due to sample splitting. DIET provides a flexible way to avoid each of these issues and is applicable to a wide range of data generating distributions. It uses conditional CDF estimators to reduce high-dimensional model-based CRTs to tests of marginal independence.

We show theoretically that DIET will achieve type-1 error control regardless of data distribution $p(\mathbf{x}, \mathbf{y}, \mathbf{z})$, then we characterize a class of data distributions for which DIET can provably achieve power. Future work in this area can study weaker assumptions on the data generating process to provably achieve power in a distillation-based CRT. This can lead to further insight into when a conditional independence test can be reduced to a marginal one without sacrificing power.

Feature	DIET	HRT	d_0-CRT	d_I-CRT	Reference(s)
Age	•	•	•	•	(a, b, c, d, e, f, g, h)
Sex	•	•	•	•	
BMI	•	•			(a, b)
Race					
Weight		•		•	
Temperature					
Heart rate	•				(a)
Smoker					
Lymphocytes count					(g, h)
Lymphocytes percent					
Days since admission	•	•		•	(g)
Respiratory rate				•	(h)
Neutrophils count					(a)
Neutrophils percent					
Eosinophils count		•	•	•	(d, g, h)
Eosinophils percent	•			•	(d)
Blood urea nitrogen	•	•	•	•	(c, d, g)
Troponin					(a, c, d, g, h)
Ferritin	•			•	(b, d, g, h)
Platelet volume	•				(b, f, h)
Platelet count					(g, h)
Creatinine					(c)
Lactate dehydrogenase					(a, g, h)
D-dimer	•	•		•	(a, c, d, e, h)
C-reactive protein	•			•	(a, b, d, g)
O2 Saturation	•	•	•	•	(a, b)
O2 device			•	•	
High O2 support	•	•	•	•	(a, g)
On room air			•	•	

Table 5.2: DIET with MDNs selects many medically relevant variables in the health records task, while omitting variables that provide similar but redundant information. This table shows which variables each method selects. We evaluate each CRT by comparing to variables found in well-cited medical articles: (a) [Petrilli et al. 2020], (b) [Sattar et al. 2020], (c) [Mei et al. 2020], (d) [Castro et al. 2020], (e) [Zhang et al. 2020], (f) [Zhong and Peng 2021], (g) [Ruan et al. 2020], (h) [Zhou et al. 2020a].

6 A deployed model for predicting adverse events in the ICU

Thus far we have seen three contributions to the controlled variable selection (cvs) literature. Here we detail the application of cvs to a problem faced by doctors at NYU Langone during the height of the Covid-19 pandemic in March 2020. Doctors needed to know which patients they could safely discharge from the ICU to make room for patients in need of urgent care. After much discussion with doctors about what tool might be most helpful, we decided to build a machine learning model to estimate a patient's probability of experiencing an adverse event. To foster trust in the model's predictions, we decided that the model should (1) be interpretable, and (2) use as few features as possible so that its predictions could be easily explained. For the latter consideration, we decided to employ cvs to identify a small set of highly predictive features for the model. In this section, we provide a detailed motivation behind the problem, then discuss the model that was built and deployed for doctors to view in real time.

6.1 Motivation

The COVID-19 pandemic has challenged front-line clinical decision-making, leading to numerous published prognostic tools. However, as of May 2020, few models had been prospectively validated and none reported implementation in practice. Here, we use 3,345 retrospective and

474 prospective hospitalizations to develop and validate an interpretable model to identify patients with favorable outcomes within 96 hours of a prediction, based on real-time lab values, vital signs, and oxygen support variables. In retrospective and prospective validation, the model achieves high average precision (88.6% 95% CI: [88.4–88.7] and 90.8% [90.8–90.8]) and discrimination (95.1% [95.1–95.2] and 86.8% [86.8–86.9]) respectively. We implemented and integrated the model into Epic Systems EHR software, achieving a positive predictive value of 93.3% with 41% sensitivity. Our results show that clinicians adopted these scores into their clinical workflows.

6.2 Introduction

COVID-19 has created a public health crisis unseen in a century. As of June 12, 2020, worldwide cases exceeded 7 million and deaths have surpassed 410,000, with over 110,000 deaths occurring in the United States alone [COVID 19]. New York emerged as an early epicenter, and the increase in case burden strained the healthcare system. Although New York's daily case count peaked in late March 2020, the number of infections continued to increase worldwide for months [Ghebreyesus 2020]. The significant impact of COVID-19 is likely to persist until herd immunity is achieved, effective therapies are developed, or a vaccine is broadly implemented. Faced with a novel disease with complex multi-organ manifestations and an uncertain disease progression course, frontline clinicians responded by sharing anecdotal management practices among peers. However, collective expert opinion is suboptimal and susceptible to selection and cognitive biases. Epidemiologic studies partially address these challenges [Petrilli et al. 2020], but they do not provide targeted information for individual patients at the point of care. Machine learning methods are uniquely positioned to rapidly aggregate the collective experiences of thousands of patients to generate tailored predictions for each patient. As a consequence, these methods have great potential to augment COVID-19 care. To be effective, solutions involving machine learning must 1) address a clearly defined use case that clinical leaders will champion and 2) motivate

changes in clinical management based on model predictions [Drysdale et al. 2019; Kelly et al. 2019]. During the COVID-19 pandemic, the operational needs of frontline clinicians have rapidly shifted. Early in the pandemic for example – with testing in short supply – predicting which patients likely had COVID-19 before a test result had great importance to triage and cohorting. As the availability and speed of testing progressed, this use case became obsolete. Similarly, while predicting deterioration is clinically important, the NYU health system had already implemented a general clinical deterioration predictive model and did not have an immediate use case for a COVID-19-specific deterioration model [Epic 2020]. Further, since Intensive Care Unit (ICU) beds were already limited to patients in immediate need of requiring higher levels of care, predicting future needs would not dramatically change clinical management. After collaboration with clinical leaders, we selected identification of patients at the lowest risk of adverse events – i.e. those predicted to have favorable outcomes – as a primary focus. This prediction task fulfills each of the requirements listed above, as handling the surge of COVID-19 patients with a limited bed capacity was a critical challenge faced by many hospitals. Discharging patients safely to free up beds for incoming patients is ideal as it does not require expanding human (e.g. nursing/physician) or structural (beds/medical equipment) resources. Given clinical uncertainty about patient trajectories in this novel disease, accurate predictions could help augment clinical decision making at the time the prediction is made. Finally, clinical leaders overseeing inpatient units committed to support the adoption of the prediction model. As of the time of writing, at least 30 peer-reviewed papers describing prognostic COVID-19 models have been published [Bi et al. 2020; Chen et al. 2020; Dong et al. 2020; Gong et al. 2020; Hong et al. 2020; Huang et al. 2020; Ji et al. 2020a,b; Jiang et al. 2020; Li et al. 2020a,c; Liang et al. 2020; Liu et al. 2020a,b,b; McRae et al. 2020; Shang et al. 2020; Wang et al. 2020b; Xia et al. 2020; Yan et al. 2020; Yu et al. 2020; Zhang et al. 2020; Zhou et al. 2020b; Cheng et al. 2020; Toussie et al. 2020; Al-Najjar and Al-Rousan 2020; Borghesi et al. 2020; Burian et al. 2020; Cecconi et al. 2020; Galloway et al. 2020]. These models use variables including patient demographics, clinical values, and radiographic images to predict

adverse events, including severe pneumonia, intubation, transfer to ICU, and death. Most models use more than one variable and most models predict composite outcomes. Of the 30 models, 23 were trained on patients in China [Bi et al. 2020; Chen et al. 2020; Dong et al. 2020; Gong et al. 2020; Hong et al. 2020; Huang et al. 2020; Ji et al. 2020a,b; Jiang et al. 2020; Li et al. 2020a,c; Liang et al. 2020; Liu et al. 2020a,b; McRae et al. 2020; Shang et al. 2020; Wang et al. 2020b; Xia et al. 2020; Yan et al. 2020; Yu et al. 2020; Zhang et al. 2020; Zhou et al. 2020b], 2 were trained on patients in the United States [Cheng et al. 2020; Toussie et al. 2020], and 5 were trained on patients in South Korea [Al-Najjar and Al-Rousan 2020] or Europe [Borghesi et al. 2020; Burian et al. 2020; Cecconi et al. 2020; Galloway et al. 2020]. Only 8 of the models underwent validation on either held-out or external datasets [Bi et al. 2020; Gong et al. 2020; Ji et al. 2020b; Liang et al. 2020; Liu et al. 2020b; Wang et al. 2020b; Yan et al. 2020; Galloway et al. 2020], and 1 underwent prospective validation [Li et al. 2020c]. No model predicted favorable outcomes and no studies reported clinical implementation. In this section, we describe how a collaboration among data scientists, electronic health record (EHR) programmers (vendor- and health system-based), clinical informaticians, frontline physicians and clinical leadership led to the development, prospective validation, and implementation of a machine learning model for real-time prediction of favorable outcomes within a 96 hour window among hospitalized COVID-19 patients. Our approach differs from prior work in that we: 1) predict favorable outcomes (as opposed to adverse outcomes), 2) use a large COVID-19 patient cohort admitted across our hospitals, 3) design a model that can easily be extended to other institutions, 4) prospectively validate performance, and 5) integrate our model in the EHR to provide a real-time clinical decision support tool. We followed the NYU Langone School of Medicine IRB protocol, and completed the checklist for IRB requirements for activities designated for quality improvement [Richardson et al. 2020]. This work met the NYU Langone School of Medicine IRB criteria for quality improvement work and did not require IRB review.

6.3 Results

A retrospective cohort for model creation and validation included all COVID-19 positive adults hospitalized at any of the four hospitals of our institution from March 3, 2020 through April 26, 2020. This cohort included a total of 3,317 unique patients and 3,345 admissions. These patients were largely White (44.6%) with an average age of 63.5 years: table 6.1. More men (61.6%) than women were included, consistent with other studies Grasselli et al. [2020]; Li et al. [2020b]; Wang et al. [2020a]. We defined a favorable outcome as absence of adverse events: significant oxygen support (including nasal cannula at flow rates >6 L/min, face mask or high-flow device, or ventilator), admission to ICU, death (or discharge to hospice), or return to the hospital after discharge within 96 hours of prediction. Patients could experience multiple adverse events during the course of their admission, e.g. requiring significant oxygen support before admission to the ICU and death. Almost half (45.6%) of patients required significant oxygen supporting devices (beyond nasal cannula) at some point during their stay and one fifth (20.3%) spent time in an ICU. The all time in-hospital mortality rate was 21.2% with another 3.1% of patients being discharged to hospice. Consistent with published literature [Henry et al. 2020; Du et al. 2020; Zeng et al. 2020; Zheng et al. 2020; Yun et al. 2020; Lindsley et al. 2020], we find that patients' admission laboratory values differ between those who do and do not go on to experience an adverse event during their hospitalization: lower lymphocyte percentage (12.6% among patients with adverse event vs. 19.7% among patients without adverse event; table 6.1) and eosinophil percentage (0.28% vs. 0.70%), with higher neutrophil percentage (79.3% vs. 70.0%), blood urea nitrogen (27.5 vs. 21.7 mg/dL), D-dimer (1573.6 vs. 987.0 ng/mL), C-reactive protein (149.0 vs. 97.0 mg/L), creatinine (1.6 vs. 1.4 mg/dL), ferritin (1609.4 vs. 1009.2 ng/mL), and troponin I (0.41 vs. 0.13 ng/mL). Similarly, patients who had adverse events had higher maximum heart rate (96.5 vs. 90.8), respiratory rate (25.8 vs. 21.6), and temperature (99.9 vs. 99.6 Fahrenheit), with lower minimum SpO2 rates (91.0% vs 93.9%) in the first 12 hours after admission prior to their first complete blood count (CBC) test

result.

6.4 Model development

Stage 1: Black-box model. Four models (a Logistic Regression, a Random Forest, LightGBM [Ke et al. 2017], and an ensemble of these three models) were trained with all 65 variables (demographics, vital signs, laboratory results, O2 utilization variables, and length-of-stay) from all prediction instances (each time a CBC result becomes available) on a training set (60% of retrospective cohort, 1,990 unique patients, contributing 17,614 prediction instances). After tuning the hyperparameters for each model via grid search and comparing each model, the best performance on the validation set (20% of retrospective cohort, 663 unique patients, contributing 4,903 prediction instances) was achieved by a LightGBM model with the following hyperparameters: 500 decision trees, learning rate of 0.02, max of 5 leaves in one tree, 0.5 sampling rate for features, max depth of 4 per tree, 1.0 L1 regularization and 2.0 L2 regularization, the minimal gain to perform split set to 0.05, and minimal sum of Hessian in one leaf set to 5.0.

Stage 2: Parsimonious Model. We set up a CRT [Candes et al. 2018] using the black-box model from Stage 1. For a review of CRTs, see section 2.1.3. Here is the procedure we used to compute the test statistic. We fit the black-box model to predict adverse events given all 65 input variables. We then evaluated the model's performance on the training dataset. The null statistics were computed similarly, but null data was sampled from a model for $p(\mathbf{x} \mid \mathbf{z})$ where $\mathbf{x}$ is the variable being tested, and $\mathbf{z}$ is the set of all other input variables. We employed the conditional histogram estimator from Miscouridou et al. [2018] to model $p(\mathbf{x} \mid \mathbf{z})$.

Using the CRT, we obtained p-values for each variable, shown in table 6.2. Using a p-value significance threshold of 0.2, 16 features were selected. These features were combined into a final "parsimonious" model as a logistic regression after quantile normalization of each variable. The

magnitude of each final model coefficient is proportional to its contribution to the final score. Positive coefficients were associated with a favorable outcome, while negative coefficients were associated with a decreased likelihood of a favorable outcome. We then performed an ablation analysis to remove features in the linear model that did not improve its performance. This analysis led to the removal of age, BMI, and maximum oxygen saturation (in the last 12 hours). Of the 13 features included in the linear model, the maximum value of nasal cannula oxygen flow rate (in the last 12 hours) feature had a non-linear, U-shaped individual conditional expectation plot with a maximum at a value of 3 L/min, and was therefore split into three binary indicators with cutoffs at 0 and 3.

6.5 Model Retrospective Validation

Model performance was measured by discrimination (area under the receiver operating characteristic curve; AUROC) and average precision (area under the precision-recall curve; AUPRC), assessed on a held-out set (independent from training or validation sets) including 20% of the retrospective cohort: 664 unique patients, contributing 5,914 prediction instances overall. The black box and parsimonious models achieved AUPRC of 90.3% (95% bootstrapped confidence interval [CI]: 90.2–90.5) and 88.6% (95% CI: 88.4–88.7) respectively, while maintaining an AUROC of 95–96% (fig. 6.1a,b). Both black box and parsimonious models maintained high AUPRC (90.8%, 95% CI: [90.7–91.0] and 89.5%, [89.3–89.6], respectively) for prediction times when patients were not receiving significant oxygen support (any device beyond nasal cannula with 6 L/min) but AUROC decreased for this subgroup (80.0% [79.9–80.2] and 78.1% [77.9–78.3]; fig. 6.1c,d). Similarly, both models maintained high performance when applied to a subset of predictions made after the patient was transferred out of the ICU (AUPRC [95% CI] of 90.7% [90.3–91.2] and 85.7% [85.1–86.3] for black box and parsimonious, respectively; AUROC [95% CI] of 95.4% [95.2–95.6] and 94.2% [93.9–94.4], respectively; fig. 6.1e,f).

6.6 Model Deployment

The final parsimonious model was implemented into NYU's EHR to make predictions every 30 minutes for each eligible patient.

6.6.1 Communicating Risk with Color

Predictions were split into three color-coded groups. The lowest risk, green-colored group were those with a score above a threshold selected at 90% positive predictive value (PPV), 53% sensitivity within the held-out set (threshold = 0.817). The moderate risk, orange-colored group were those patients with a score lower than green but above a second threshold corresponding to 80% PPV, 85% sensitivity (threshold = 0.583). The highest risk, red-colored group were all remaining predictions. In the held-out set, these two thresholds separated all predictions into three groups where favorable outcomes within 96 hours are observed in 90.0% of green, 67.3% of orange and 7.9% of red patients.

6.6.2 Assessing Face Validity with Chart Review

Prior to displaying the model predictions to clinicians, a team of medical students and practicing physicians assessed the face validity, timing, and clinical utility of predictions. A variety of patient types were reviewed including 30 patients who had a green score, 8 of whom had left the ICU and 22 who had not. Overall, 76.7% (23 of 30) of the green predictions were labeled clinically valid where the primary clinical team acknowledged the patient as low-risk or were beginning to consider discharge. Timing of those green predictions either aligned with actions by the primary clinical team or preceded those actions by one or two days (a total of 34 days earlier, an average of 1.13 days). Invalid green predictions typically had other active conditions unrelated to their COVID-19 disease (e.g. untreated dental abscess), while those patients discharged as orange or red

typically had pre-hospitalization oxygen requirements (e.g. BIPAP for obstructive sleep apnea).

6.6.3 Timing of Green Predictions

For all patients in the held-out set discharged alive, 77.8% of patients (361 of 464) had at least one green score, and their first green score occurred a median 3.2 (interquartile range: [1.4–5.4]) days before discharge. The vast majority of green patients who were discharged alive never received care in an ICU (91.4%; 330 of 361). Those that did receive ICU care had much longer length of stay before their first green score (fig. 6.2a) but once green, they had similar remaining length of stay before discharge (fig. 6.2b).

6.6.4 Electronic Health Record Integration and Visualization

The resulting scores, colors, and contributions populated both a patient list column viewable by clinicians and a patient-specific COVID-19 summary report, which aggregates data important for care including specific vitals, biomarkers, medications. The core component of the visualization was a colored oval containing that patient's risk score (fig. 6.3). The column hover-bubble and report section displayed a visualization containing the colored score, a trendline of recent scores, and variables with their values and contributions (fig. 6.3).

6.6.5 Prospective Validation

The model was integrated into the EHR and its real-time predictions were displayed to clinicians starting May 15, 2020. Prospective performance was assessed using data collected from May 15 to May 28, 2020 (predictions until May 24 with 96 hour follow-up). In those ten days, 109,913 predictions were generated for 445 patients and 474 admissions. Among these prospectively scored patients, 35.1% (156) required significant oxygen support, 5.4% (24) required more than 6 L/min of oxygen while on nasal cannula, 7.2% (32) died, 2.2% (10) were discharged to hospice care,

19.8% (88) were transferred to the ICU, and 1.8% (8) were discharged and readmitted within 96 hours. Overall, 44.0% (196 patients) experienced an adverse event within 96 hours of a prediction instance, which is lower than the rate observed in our retrospective cohort (51.6%, 1712 of 3317, p = 0.003 by two-tailed Fisher's exact test), consistent with prior reports from our institution showing a temporal improvement in outcomes3. Prospective evaluation of the model achieved an AUPRC of 90.8% (95% CI: 90.8–90.8; fig. 6.4a) and AUROC of 86.8% (95% CI: 86.8–86.9; fig. 6.4b), similar to retrospective performance (AUPRC: 88.6%, and AUROC: 95.1%). Using the predefined green threshold, the real-time model identified 41.0% of predictions as green with 93.3% PPV and 67.8% sensitivity (compared to 90% PPV and 53% sensitivity in the retrospective held-out set), and favorable outcomes are observed in 93.3%, 72.4%, and 23.5% of green, orange, and red predictions, respectively, consistently higher than the retrospective held-out set (90.0%, 67.3%, and 7.9%).

6.6.6 Adoption into Clinical Practice

Since integration into the EHR, we monitored two high-level metrics to assess score adoption into clinical practice. The model predictions are visible in two places: multiple patients shown in a patient list column (fig. 6.3) and a single patient shown in a COVID-19 Summary report. A patient list column metric counts the number of times the model scores are shown in patient lists (not counting each patient displayed). A summary report metric counts the number of times a provider navigated to the COVID-19 Summary report to review data on a single patient. More specifically, during the three weeks May 16 to June 5, 2020 (omitting the partial day of May 15), scores are shown in a total of 1,122 patient lists and 3,374 COVID-19 reports. Temporal trends in these metrics suggest an increasing trend in the rate of patient lists per day but a decreasing trend in COVID-19 reports (fig. 6.5). Together, these metrics describe an adoption of users adding the patient list column, a result of outreach and communication to users, and a decline in the number of COVID-19 reports accessed, which may be explained by a decline in the number of hospitalized COVID-19 patients. Future work will assess the impact of these scores on physician

perspectives and decision-making.

6.7 Discussion

The COVID-19 pandemic energized an existing inter-disciplinary collaboration at our institution to successfully develop a predictive model that was accurate and relevant for clinical care, could be rapidly deployed within our EHR, and could be readily disseminated to other institutions. The final parsimonious model exhibited strong model performance for the clinical task (fig. 6.1) and could be maintained with only 14 of the original 65 variables combined in a logistic regression that is transparently explainable (fig. 6.3). Yet model accuracy is not sufficient to ensure measurable success; the prediction must be clinically applicable at the time of prediction. We determined that our model predicts patients at high probability of favorable outcomes a median of 3.2 days before discharge (fig. 6.2b), providing sufficient lead time to commence and prepare for earlier and safer discharges. Our chart review results suggest the green transition occurs, in many cases, before any discharge planning is documented. By identifying patients at low risk of an adverse event with high precision, this system could support clinicians in prioritizing patients who could safely transition to lower levels of care or be discharged. By contrast, using published models that predict occurrence of adverse events to guide discharge decisions may not be as effective. The distinction between identification of patients at low-risk of experiencing an adverse event rather than those at high-risk is key. Although the binary outcome of an adverse event or none is reciprocal, the methodology of tuning model hyperparameters to identify the best model and then selecting a threshold based on PPV is not. If the target outcome is reversed, we would expect our methodology to discover a different parsimonious model. The key strengths of our approach are twofold. First, a reduced variable set helps prevent overfitting by making it less likely that a machine learning model will learn site-specific details48. Second, our approach is easily integrated into third-party EHR systems. Collaborating with our clinical decision support (CDS) experts, we

incorporated our intervention directly into standard clinical workflows (fig. 6.5): 1) the patient lists clinicians used when reviewing and prioritizing their patients, and 2) the standard report clinicians rely on to summarize COVID-19 aspects of care. By incorporating the prediction at the appropriate time and place in the EHR for the users responsible for discharge decisions, we expect to maximize the impact of this intervention in the care of COVID-19 patients [Qin et al. 2020]. Although integration into an EHR system maximizes its impact and simplifies dissemination to other institutions, it also adds several significant constraints institutions must consider. Potentially useful data available on retrospective data queries may not be reliably accessible in real-time to make a prediction. For example, codified comorbidities and prior medications may be incomplete at the time of prediction, particularly for new patients who have never received care within the health system. Therefore, only data collected during admission are suitable for generalizable modeling. Extraction of complex features such as means are infeasible within the current EHR's cognitive computing platform. These data access challenges inside the EHR are part of the rationale behind our two-step model development that produces a parsimonious model reliant on a small number of inputs. Despite the above constraints, the two-step methodology applied to construct the parsimonious model did reveal previously described [Petrilli et al. 2020] prognostic indicators of adverse events in COVID-19 patients including vital signs such as hypoxia, C-reactive protein and lactate dehydrogenase (table 6.2). Yet many features commonly associated with worsening prognosis, such as age, gender, lymphocyte count, and D-dimer ultimately did not contribute to the final model. There are a variety of potential explanations for this apparent discrepancy. Differences between patients with and without adverse events were observed for both neutrophil percent and lymphocyte percent (and their absolute counts; Table 1) but the parsimonious model used only eosinophil percent, as the alternatives were not found to provide further information over eosinophils (table 6.2), reflecting probable redundancy between white blood cell biomarkers. Both eosinophils percent and platelet count have positive coefficients (table 6.2) suggesting a positive association between immune characteristics [Gao et al. 2016] and

thrombocytosis with fewer adverse outcomes. Similar redundancy might also explain why lactate dehydrogenase and C-reactive protein contributed to the ultimate model while D-dimer and troponin did not. While age and sex are marginally associated with adverse outcomes, neither contibute to the final model, suggesting other variables account for variance in these demographics such that they no longer aid prediction. The reasoning for why these variables do not directly contribute is unclear. Epidemiologic studies have been critical in helping clinicians understand this evolving disease entity and expedite predictive model development. Yet the volume of clinical features associated with adverse events precludes easy assimilation by clinicians at the point of care. At our institution, a COVID-19 specific summary report for each patient trends over 17 variables. The ability of machine learning to synthesize and weigh multiple data inputs facilitates more accurate application of the data to directly impact care. Another advantage of our approach is that model explanations were made available to the clinicians along with real-time predictions. Our parsimonious model, being linear, enabled a seamless computation of contributing factors. Providing insight into contributing factors helps improve trust in the model and we believe will improve its incorporation into clinician decision making. These explanations also helped mitigate some inherent limitations of real-time models. For example, clinicians could discount the model's predictions if they found that some of the inputs, like respiratory rate, were documented inaccurately. Similarly, the model could not discriminate between patients receiving BIPAP for chronic obstructive sleep apnea versus for acute respiratory failure. A clinician would have this background and could consider the model's score in that context. Front-line clinicians continued to evolve their care for patients with COVID-19 in response to research findings. Particularly during the retrospective study period, March and April 2020, there were rapid changes in testing and treatment practices. The data collected about a COVID-19 patient in March is likely very different from a similar patient seen in the prospective cohort in late May 2020. For example, the volume of D-dimer values for patients increased dramatically from early March to April as clinicians incorporated D-dimer screening into their care plans. These expected differences in model

variables and outcomes challenge the generalizability of any predictive model, which emphasized the importance of prospective validation. Using oxygen therapy as both an input variable and an outcome measure led the model to learn that patients on O2 devices are likely going to continue to remain on O2 devices in the near future. Consequently, the model coefficient for significant oxygen support overshadowed other variables and patients on significant O2 devices uniformly had very low favorable outcome scores. In consultation with our clinical leads, this model behavior was acceptable given that these patients on significant oxygen devices were clinically unlikely to be safe for discharge. Furthermore, when excluding significant O2 support as an input variable or omitting periods of significant O2 support, the model performed worse overall and among patients not using O2 devices. Thus, we retained this variable and analyzed the subset of patients without O2 devices separately, which demonstrated excellent performance (Figs. 6.1c,d). Construction of the parsimonious model as a linear model also impacted how each variable's contribution was explained to the clinician. This constraint resulted in some explanations that were clinically concerning, like hypothermic temperatures displaying as a mildly protective feature (Table 1). This phenomenon occurs because a linear model fits a linear slope to each variable and misses U-shaped risk curves. In summary, our model's predictions were accurate, clinically relevant, and presented in real time within the clinician's workflow. These features all enhance the likelihood that the model will be clinically successful. To assess our model's impact on clinically important outcomes, a randomized controlled trial is underway examining knowledge of favorable outcome prediction on patient length of stay. With clinical value confirmed, future work in this area should can involve further collaboration with the vendor community to rapidly disseminate models such as ours to customers.

Patient Characteristics	All Cohort (% of n=3,317)	With Adverse (% of n=1,712)	Without Adverse (% of n=1,605)	P-value*
Demographics				
Age, mean (sd)	63.5 (16.5)	65.3 (15.8)	61.5 (17.0)	<0.0001
Sex, n (%)				<0.0001
Female	1275 (38.4%)	571 (33.4%)	704 (43.9%)	
Male	2042 (61.6%)	1141 (66.6%)	901 (56.1%)	
Race, n (%)				<0.0001
White	1481 (44.6%)	794 (46.4%)	687 (42.8%)	
Black	508 (15.3%)	204 (11.9%)	304 (18.9%)	
Asian	246 (7.4%)	141 (8.2%)	105 (6.5%)	
Other Race	916 (27.6%)	478 (27.9%)	438 (27.3%)	
Unknown	164 (4.9%)	84 (4.9%)	80 (5.0%)	
Adverse Event Outcomes, n (%)				
Mortality (For all time)	702 (21.2%)			
Hospice Discharge	102 (3.1%)			
ICU Admission	673 (20.3%)			
O2 Support Devices Beyond Nasal Cannula	1513 (45.6%)			
O2 Flow Rate >6 L/min on Nasal Cannula	365 (11.0%)			
Readmission within 96 hours of discharge	20 (0.60%)			
Biomarkers, first value measured, mean (sd)				
Neutrophils Count (103/uL)	6.2 (5.4)	7.3 (6.7)	4.9 (3.0)	<0.0001
Neutrophils Percent	74.8 (12.8)	79.3 (11.2)	70.0 (12.8)	<0.0001
Lymphocytes Count (103/uL)	1.1 (1.7)	1.1 (2.3)	1.2 (0.74)	0.014
Lymphocytes Percent	16.0 (10.4)	12.6 (8.7)	19.7 (10.8)	<0.0001
Eosinophils Count (103/uL)	0.03 (0.12)	0.02 (0.11)	0.05 (0.12)	<0.0001
Eosinophils Percent	0.49 (1.2)	0.28 (1.0)	0.70 (1.4)	<0.0001
Platelet Count (103/uL)	225.9 (98.45)	222.0 (95.6)	230.1 (101.2)	0.017
Blood Urea Nitrogen (mg/dL)	24.7 (22.8)	27.5 (23.8)	21.7 (21.2)	<0.0001
Creatinine (mg/dL)	1.5 (1.8)	1.6 (1.7)	1.4 (1.9)	0.027
C-Reactive Protein (mg/L)	124.4 (86.3)	149.0 (87.8)	97.0 (75.8)	<0.0001
D-Dimer (ng/mL DDU)	1295.7 (3582.4)	1573.6 (4101.2)	987.0 (2869.9)	<0.0001
Ferritin (ng/mL)	1324.0 (2315.4)	1609.4 (2767.8)	1009.2 (1624.3)	<0.0001
Lactate Dehydrogenase (U/L)	399.3 (243.9)	457.0 (279.5)	337.0 (178.8)	<0.0001
Troponin I (ng/mL)	0.28 (2.7)	0.41 (3.5)	0.13 (1.3)	0.0032
Vital signs, first 12 hour, mean (sd)				
HR max	93.7 (17.9)	96.5 (19.4)	90.8 (15.7)	<0.0001
Resp max	23.8 (7.1)	25.8 (8.3)	21.6 (4.7)	<0.0001
SpO2 max (%)	96.3 (2.4)	96.0 (2.6)	96.6 (2.1)	<0.0001
Temp max (F)	99.8 (1.5)	99.9 (1.6)	99.6 (1.4)	<0.0001
HR min	80.2 (14.2)	80.9 (14.8)	79.5 (13.5)	0.0045
Resp min	18.9 (3.6)	19.3 (4.2)	18.5 (2.6)	<0.0001
SpO2 min (%)	92.4 (4.9)	91.0 (5.8)	93.9 (2.9)	<0.0001
Temp min (F)	98.4 (0.97)	98.5 (1.0)	98.4 (0.88)	0.12

Table 6.1: Demographics, outcomes, biomarkers, and vital signs of retrospective cohort.

	Variable Explanation	Conditional Independence p-value	Used in Final Model	Final Model Coefficient (+ toward a favorable outcome)
Model Intercept	1.43			
1	Age	0.016	✗	
2	Oxygen support device greater than nasal cannula	0.016	✓	-7.31
3	Respiratory rate, maximum in last 12 hours	0.016	✓	-1.23
4	Oxygen saturation, maximum in last 12 hours	0.016	✗	0
5	Oxygen support device of nasal cannula	0.016	✓	-0.816
6	Nasal cannula oxygen flow rate, maximum value in last 12 hours	0.016	✓	0 if flow >3L/min
7	Oxygen saturation, minimum value in last 12 hours	0.016	✓	+1.12 if 0 <flow <= 3L/min +0.424 if flow = 0
8	Temperature, maximum value in last 12 hours	0.016	✓	1.52 -0.439
9	Lactate dehydrogenase, most recent value	0.016	✓	-0.168
10	Platelet count, most recent value	0.016	✓	0.755
11	Blood urea nitrogen, most recent value	0.016	✓	-1.3
12	C-reactive protein, most recent value	0.016	✓	-0.558
13	Heart rate, minimum value in last 12 hours	0.033	✓	-0.437
14	Respiratory rate, minimum value in last 12 hours	0.033	✓	-0.407
15	Eosinophils percent, most recent value	0.148	✓	0.916
16	Body mass index, maximum value in last 12 hours	0.148	✗	
17	No oxygen support device (i.e. room air)	0.803	✗	
18	Heart rate, maximum value in last 12 hours	0.967	✗	
19	Neutrophil count, most recent value	0.967	✗	
20	Temperature, minimum value in last 12 hours	0.984	✗	
21	Eosinophil count, most recent value	0.984	✗	
22	Weight, maximum value in last 12 hours	0.984	✗	
23	Mean platelet volume, most recent value	0.984	✗	
24	Categorical variable of historical smoking behavior: e.g. non-smoker or smoker	1	✗	
25	Lymphocyte count, most recent value	1	✗	
26	Female sex	1	✗	
27	Number of days since admission	1	✗	
28	Lymphocytes percent, most recent value	1	✗	
29	Categorical variable of current smoking behavior: e.g. never, former, current smoker	1	✗	
30	Troponin I, most recent value	1	✗	
31	Neutrophils percent, most recent value	1	✗	
32	Body mass index, minimum value in last 12 hours	1	✗	
33	Creatinine, most recent value	1	✗	
34	D-dimer, most recent value	1	✗	
35	Ferritin, most recent value	1	✗	
36	Weight, minimum value in last 12 hours	1	✗	
37	Categorical variable of patient race and ethnicity	1	✗	

Table 6.2: Distillation of a parsimonious model as a combination of conditionally independent variables

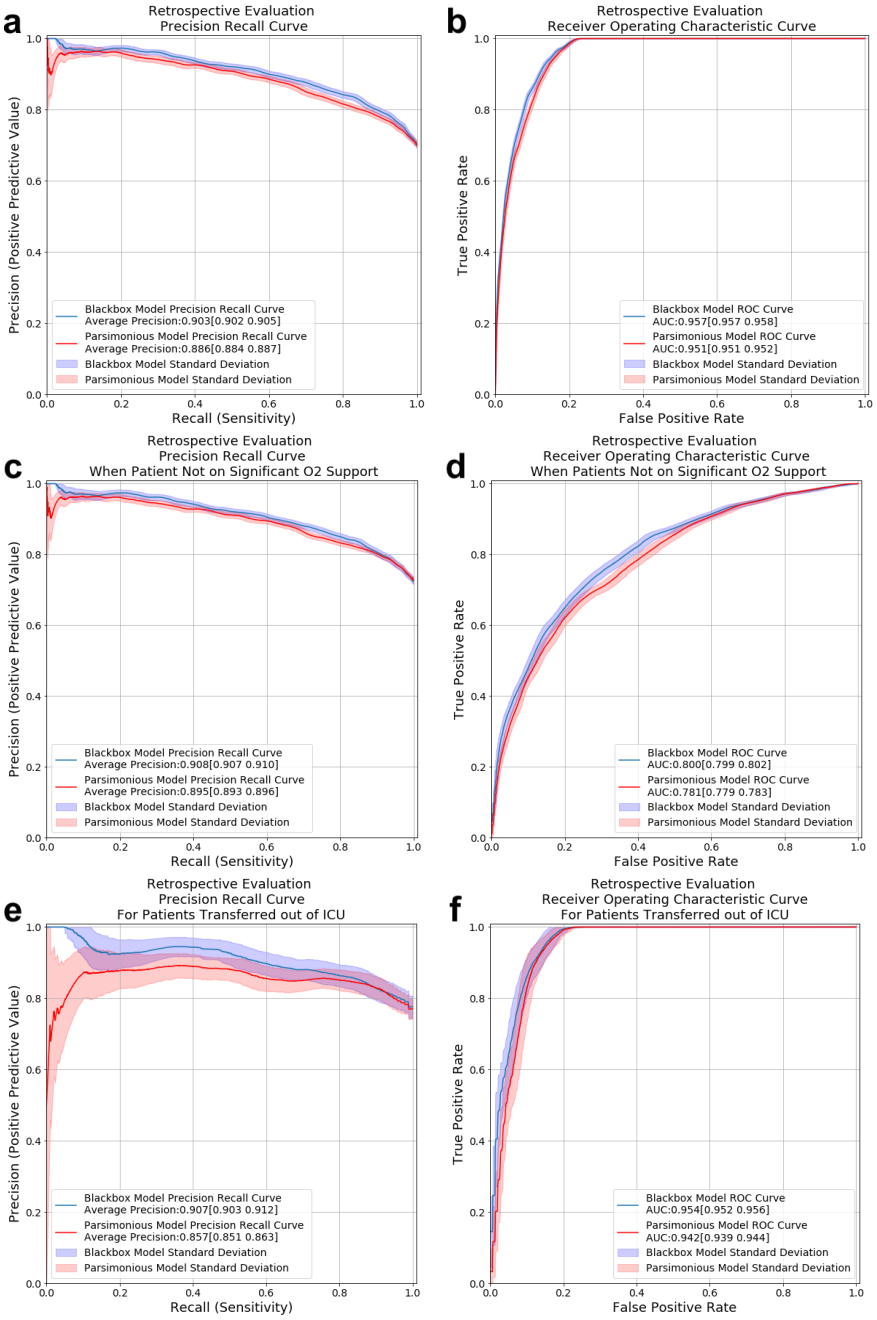


Figure 6.1: Predictive performance of the black box and parsimonious models on retrospective held-out set. Model performance in an unseen 20% sample of data including 664 unique patients and a total of 5,914 prediction instances. Panel a shows precision recall curve (PRC) for all patients. Panel b shows the receiver operating characteristic (ROC) curve for all patients. Panel c shows PRC for patients at times when patient does not need O2 support beyond nasal cannula at 6 L/min. Panel d shows the ROC curve for patients at times when patient does not need O2 support beyond nasal cannula of 6 L/min. Panel e shows PRC for patients transferred out of ICU, and panel f shows the ROC curve for patients transferred out of ICU. The shaded areas around each curve depict the empirical bounds of one standard deviation computed with a bootstrap procedure with 100 iterations where, in each iteration, 50% of the held-out set is sampled with replacement.

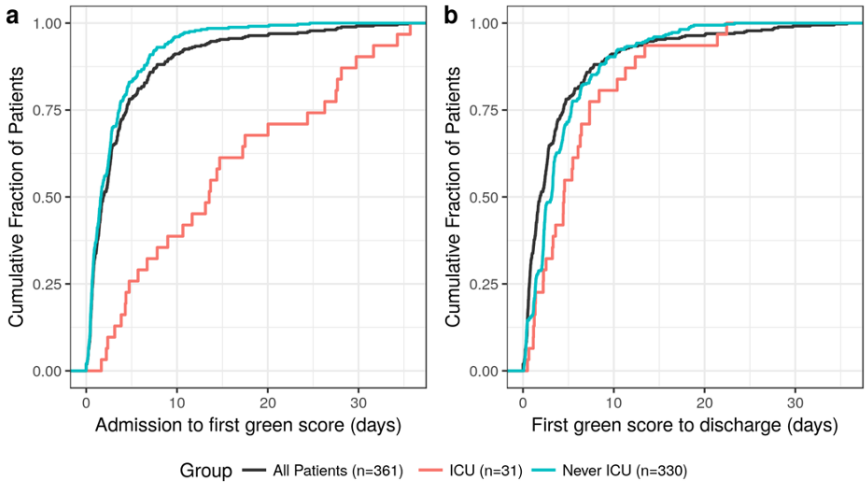


Figure 6.2: Timing of the first "green" prediction for patients discharged alive from the retrospective held-out set. Panel a shows the the time from admission to the first green score, while panel b shows the time from the first green score to discharge. This analysis includes all held-out set patients with at least one green score who were discharged alive (n=361) and stratifies that group into patients that received some of their care in an ICU (n=31) and those who received no ICU care (n=330).

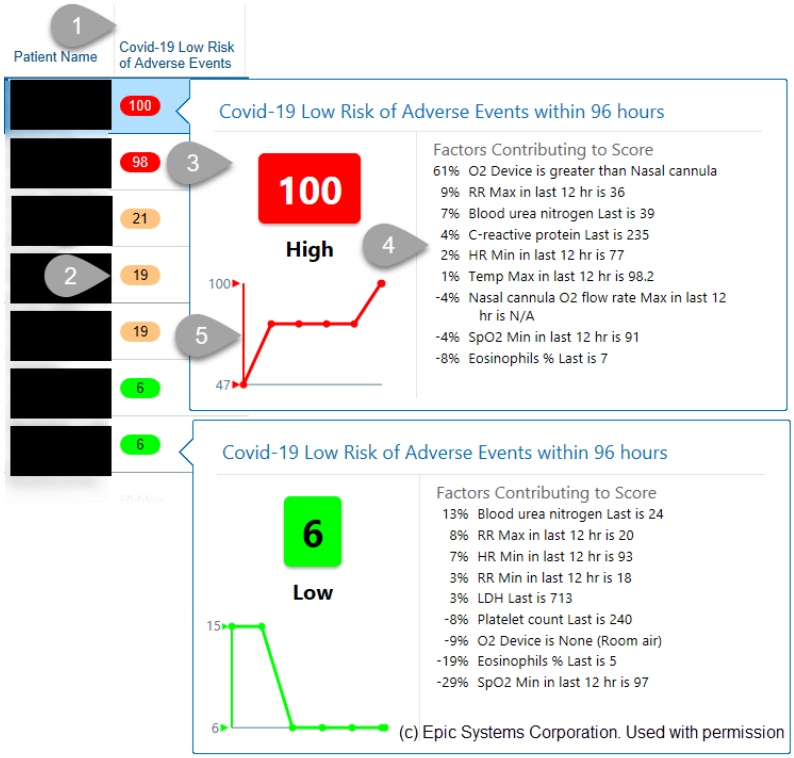


Figure 6.3: Electronic Health Record integration and visualization of predictions. Provider-facing view showing: (1) a patient list column, (2) displaying model scores for a clinician's list of patients. Hovering over the score triggers a dialog box (3) displaying model scores along with (4) an explanation of contributing factors and (5) a trend line of recent scores. To reduce potential for confusion by clinicians, we display the inverse of the model prediction raw score (i.e 1 - score) and scale the score from 0–100. Consequently, lower scores represent patients at lower risk of adverse outcomes. Negative feature contributions are protective. Note, in the first prediction, the variable "Nasal cannula O2 flow rate Max in last 12 hrs" has a value of "N/A" because their O2 device is greater than Nasal cannula.

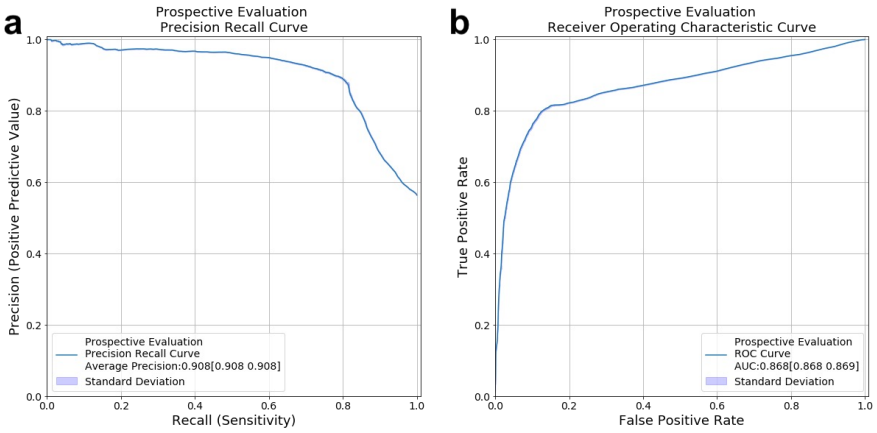


Figure 6.4: Prospective deployment and evaluation on real-time predictions. A total of 109,913 predictions were generated on 30-minute intervals for 445 patients and 474 admissions. Panel a shows the precision recall curve. Panel b shows the receiver operating characteristic curve. The shaded areas around each curve depict the empirical bounds of one standard deviation computed with a bootstrap procedure with 100 iterations, where in each iteration, 50% of the held-out set is sampled with replacement. Note: the shaded standard deviation of fig. 6.4 are present but very small as the many predictions made at a 30-minute frequency decreases variance.

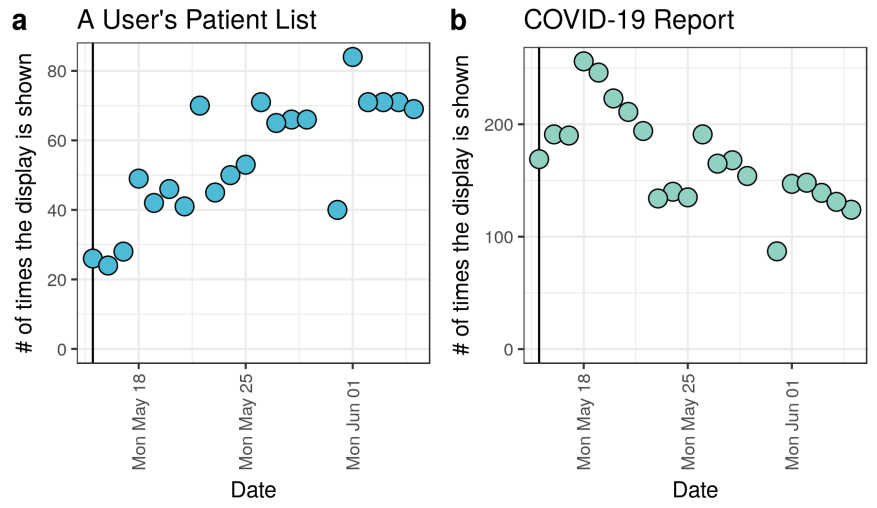


Figure 6.5: Display of model scores to users within the EHR. Model scores can be shown to users in two different displays that correspond to alternative clinical workflows. Panel a shows a patient list (fig. 6.3) display report, which indicates the number of times users navigated to a patient list that includes our model scores. Panel b shows the COVID-19 report, which describes the number of times a user navigated to a summary report that contained various COVID-19 specific components including our model scores.

7 | Building CRT tools for cancer genomics

In this chapter we detail another application of CRTs to a cancer genomics problem. We first motivate the problem, then describe the specifics of a CRT designed specifically for this task. Finally, we explore results on semi-synthetic simulations.

7.1 Motivation

Memorial Sloan Kettering (MSK) recently introduced a test called the integrated mutation profiling of actionable cancer targets (IMPACT). The goal of the test is to identify whether a patient's tumor has mutations that make their cancer vulnerable to particular drugs [Cheng et al. 2015]. MSK then matches patients with such mutations to available therapies or to ongoing clinical trials that will benefit them the most.

The MSK-IMPACT test involves collecting two DNA samples from each individual: one from tumor tissue and another from normal tissue. The normal tissue is often a sample of the individual's blood. IMPACT then compares the genome of the tumor to that of the normal tissue to identify mutations that are specific to the tumor and not generally present in the patient. The set of genes tracked by IMPACT is standardized across studies. For example, different scientists studying different phenotypes may collect the mutation profiles of the same set of genes. This

data is illustrated in fig. 7.1. While the phenotypes collected may differ across studies, the set of tumor mutations is the same.

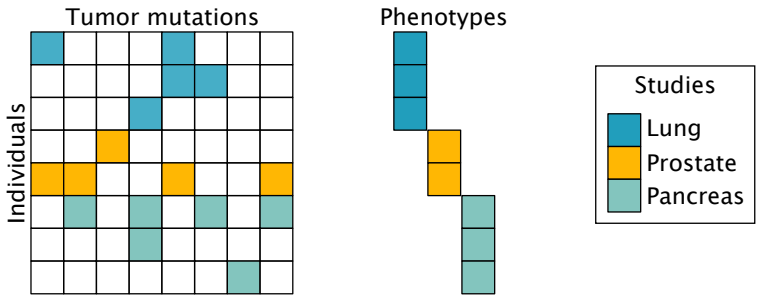


Figure 7.1: MSK-IMPACT dataset. The $\mathbf{x}$ data is shown as a matrix where each row is an individual and each column is a particular tumor mutation. If the mutation is present, the cell is filled in. Phenotypes are collected only for a particular study. For example, scientists may collect metastasis information at three different sites: Lung, Prostate, and Pancreas. The values of the Lung phenotypes for patients in the Prostate and Pancreas study are not collected.

To identify patients that will benefit from existing drugs, MSK requires a tool to first identify causal mutations. The set of causal mutations depends on the phenotype a scientist wishes to study. Not all mutations cause metastasis for example.

Error control is another objective of scientists at MSK. The false discovery of a non-causal cancer mutation can result in patients being flagged for treatment when their tumor mutations don't have any effect on the phenotype. In the best case, this is a waste of money and time, and in the worst case the patient could receive unnecessary medication. Scientists must carefully balance the objective of error control with the objective to identifying as many patients as possible.

The objective of error control and the structure of the data collected suggest that CRTs can be useful. As mentioned in earlier chapters, CRTs can enable precise control of false discovery rates when the covariate distribution is modeled well. This may be possible due to the way scientists collect data for studies at MSK. While each study may collect phenotype information about a small group of individuals, the covariate data – the mutation profiles – contains the same genes across studies. Since the same tumor mutations are collected across different studies, a good model for the covariate distribution can be constructed.

In the next few sections of this chapter, we describe a challenging semi-synthetic experiment that compares CRTs to conventional methods employed by scientists to identify causal tumor mutations.

7.2 Experiments

We first describe the covariate data $\mathbf{x}$, then outline a synthetic phenotype generating process $\mathbf{y} \mid \mathbf{x}$. We then detail the CRT we employ to select causal mutations, followed by two commonly used baseline methods.

7.2.1 Covariate data

The covariate data $\mathbf{x}$ consists of over 22K samples of 458 gene mutations. The jth gene mutation $\mathbf{x}_j$ is 1 if a mutation is present, and 0 otherwise. We perform several preprocessing steps to clean the data.

Some studies do not sequence the full set of MSK-IMPACT genes, which results in missingness in the dataset. First, we drop genes with missingness greater than 30%. This reduces the number of genes by roughly 17%, yielding 385 genes with low missingness. We further reduce the number of genes by dropping those with standard deviation below 0.025. The resulting set of genes has 285 elements. Finally, we drop samples with any missing values in the remaining 285 genes. This reduces the total sample size by 2K. The final $\mathbf{x}$ dataset contains 285 genes with roughly 20K samples.

To generate the $\mathbf{y} \mid \mathbf{x}$ data, we design a process that makes it difficult to correctly identify important genes. We first compute a gene correlation matrix $C \in [-1, 1]^{285 \times 285}$. We then identify 60 highly correlated genes using the following greedy algorithm. Let set $S = \emptyset$. We identify the gene $\mathbf{x}_j$ that has the highest average absolute correlation with all other genes and add it to the set S. We then add to S the gene that has the highest average absolute correlation with the

genes in S (that is not already in S). This process is repeated until $|S| = 60$. From this set of correlated genes S, we pick 30 uniformly at random and deem them the "causal" genes. Using only the causal genes, we generate a random 2 layer neural network with ReLU activations after each layer. This $\mathbf{y} \mid \mathbf{x}$ generation process is repeated 10 times by reselecting the set of causal genes each time. Each of these 10 phenotypes is termed a "replicate" of this experiment.

7.2.2 CRTs

We run separate CRTs for each study. The data contains 20 different studies, each containing samples from different cancer sites. These cancer sites include: Lung, Pancreas, Breast, Prostate, Colon, Ovary, Uterus, Rectum, Bladder, Skin, Sigmoid Colon, Liver, Stomach, Thyroid, Kidney, Esophagus, Unknown, Ascending Colon, Testis, and Cecum. While the CRT test statistic is computed using data specific to a study, the covariate distribution model is fit using data across studies.

MODELING THE COVARIATE DISTRIBUTION. We use a logistic factor model to model the joint distribution of the mutation data. Under this model, the distribution of the covariates given some latent factor $\mathbf{z}$ can be written as:

$$p(\mathbf{x}_1, \ldots, \mathbf{x}_{285} \mid \mathbf{z}) = \prod_{j=1}^{285} p(\mathbf{x}_j \mid \mathbf{z}).$$

We take advantage of the following property to run CRTs:

$$\mathbf{y} \perp\!\!\!\perp \mathbf{x}_j \mid \mathbf{x}_{-j} \Leftrightarrow \mathbf{y} \perp\!\!\!\perp \mathbf{x}_j \mid \mathbf{z}.$$

Testing the conditional independence of a covariate $\mathbf{x}_j$ with $\mathbf{y}$ given $\mathbf{x}_{-j}$ is equivalent to testing the conditional independence of $\mathbf{x}_j$ with $\mathbf{y}$ given the latent factor $\mathbf{z}$ instead. This means that null data for the CRT test statistic can be generated by sampling from $p(\mathbf{x}_j \mid \mathbf{z})$ instead of from

$p(\mathbf{x}_j \mid \mathbf{x}_{-j})$. This is convenient because the factor model learns all the $p(\mathbf{x}_j \mid \mathbf{z})$ distributions jointly. In our experiments, we use the factor model from Ranganath and Perotte [2018] trained on the entire dataset of 20K samples. This model has the form of an autoencoder, where the latent factor $\mathbf{z}$ is a function of the covariates $\mathbf{x}$.

CRT TEST STATISTIC. The CRT test statistic we explore here is CONTRA, as detailed in chapter 4. To recap, the test statistic T is computed as follows. Given $(\mathbf{x}, \mathbf{y})$ samples, we split the data in half. A "true model" is fit to $\mathbf{y} \mid \mathbf{x}$ using one half of the data. Using that same half of data, a null dataset is generated by replacing values of $\mathbf{x}_j$ with samples from $p(\mathbf{x}_j \mid \mathbf{z})$. A "null model" is fit to predict $\mathbf{y}$ using the null dataset. The CONTRA test statistic here is the average of the losses achieved by the true and null models on the held-out half of the data. Similarly, the CONTRA null statistic is the same function applied to a null dataset. We explore two choices of regression for the true and null models: random forest regressors, and a cross-validated lasso model.

7.2.3 BASELINES

We compare CRTS to two popular feature selection baselines: random forest feature importance scores, and lasso model coefficients.

RANDOM FORESTS. The score for each feature $\mathbf{x}_j$ is computed as follows. A score is assigned to each feature $\mathbf{x}_j$ in each tree of the random forest, then these scores are averaged across trees.

The importance score within each tree relies on a quantity called the "weighted decrease in mean squared error (MSE)." Here's how it is computed. For illustration consider fig. 7.2. At some branching point P, the random forest chooses the feature TP53 to split the data on. All the samples with a TP53 mutation go into the left node L, and the rest go into the right bin R. The MSE achieved by the tree at point P is MSE_P. The same notation follows for L and R. The

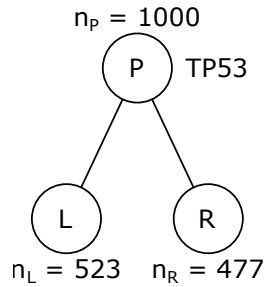


Figure 7.2: Decision tree branching point.

weighted decrease in MSE at a point P is then written as:

$$w_P = n_P \times \text{MSE}_P - n_L \times \text{MSE}_L - n_R \times \text{MSE}_R.$$

The importance score for a particular feature like the gene TP53 is sum of all w_P where the branching point P uses TP53 divided by the number of samples at the root node of the tree. If a feature is never used to split the data, it receives a score of 0.

Finally, to compute feature importance scores across the random forest, the mean score for each feature across trees is computed. These scores provide a way to rank the inputs to the $\mathbf{y} \mid \mathbf{x}$ model by importance.

Lasso model. Extracting feature importance scores from the Lasso model is fairly straightforward. First a linear regression with ℓ_1 regularization is fit. The strength of the regularization is determined via cross-validation on a held-out portion of the training data. The importance scores assigned to each feature are the absolute values of the corresponding regression coefficient.

7.3 Results

The goal of the following experiments is to rank covariates in such a way that the important covariates rank higher than unimportant ones. Recall that in our semi-synthetic simulations, the important covariates are those that are used to generate $\mathbf{y}$ from the $\mathbf{x}$ data. To compare covariate

rankings between methods, we plot precision-recall curves and measure the average precision of each ranking. Error bars are shown by averaging precision-recall curves over replicates.

To compute a ranking of covariates with baseline methods, we used the corresponding importance score. We use the inverse of the p-value to rank covariates using each version of CONTRA.

In the following plots, we compare using the feature importance scores of a baseline directly to using the same $\mathbf{y} \mid \mathbf{x}$ model inside a CRT test statistic. In fig. 7.3, we compare the Lasso coefficients to CONTRA with a Lasso model. We plot precision-recall curves for three different studies: Lung, Pancreas, and Prostate. There is a clear difference in both the average precision of each method and in the precision recall curves. Upon switching to a more powerful $\mathbf{y} \mid \mathbf{x}$ model, the random forest, this performance gap decreases, but the CRTs are generally better. We see this result in fig. 7.4.

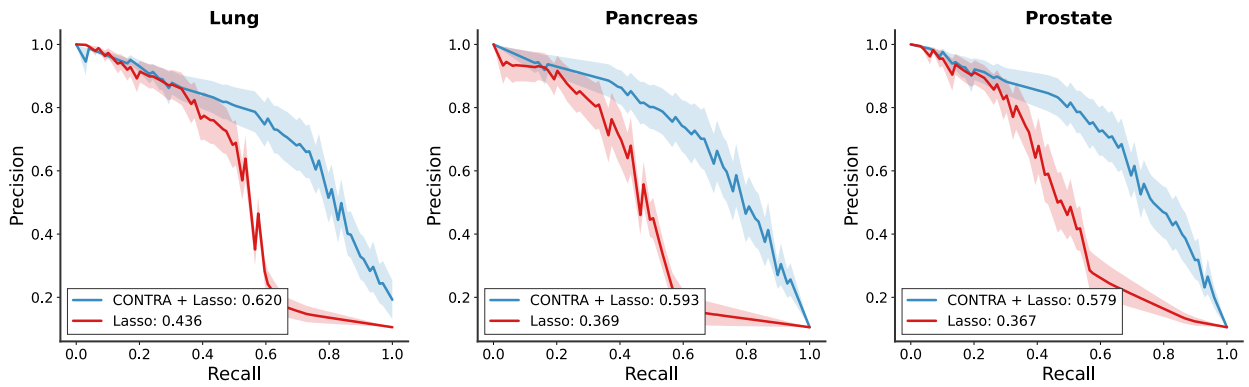


Figure 7.3: CONTRA + Lasso achieves noticeably higher average precision than the Lasso feature importance scores.

We also show 17 further results on other studies done at MSK in figs. 7.5 and 7.6. The studies are ordered in order of decreasing sample size. We make the following observations. When $\mathbf{y} \mid \mathbf{x}$ is not modeled well, the CRTs do a much better job at ensuring that unimportant covariates do not achieve a higher rank than important ones. When $\mathbf{y} \mid \mathbf{x}$ is modeled well, the difference between a non-CRT method and a CRT method is smaller. It is important to note that the CRTs are never significantly worse than the baselines in any of our results.

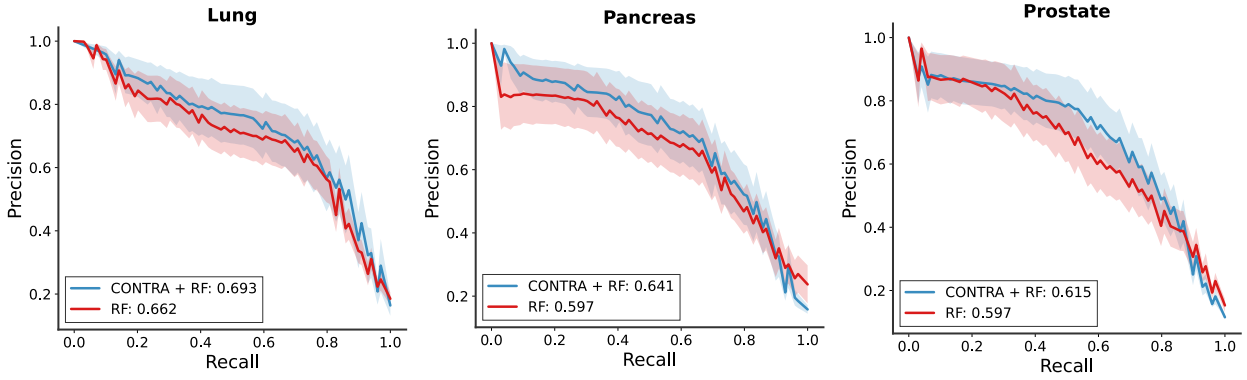


Figure 7.4: CONTRA + Random forest achieves performance on par with random forest feature importance scores.

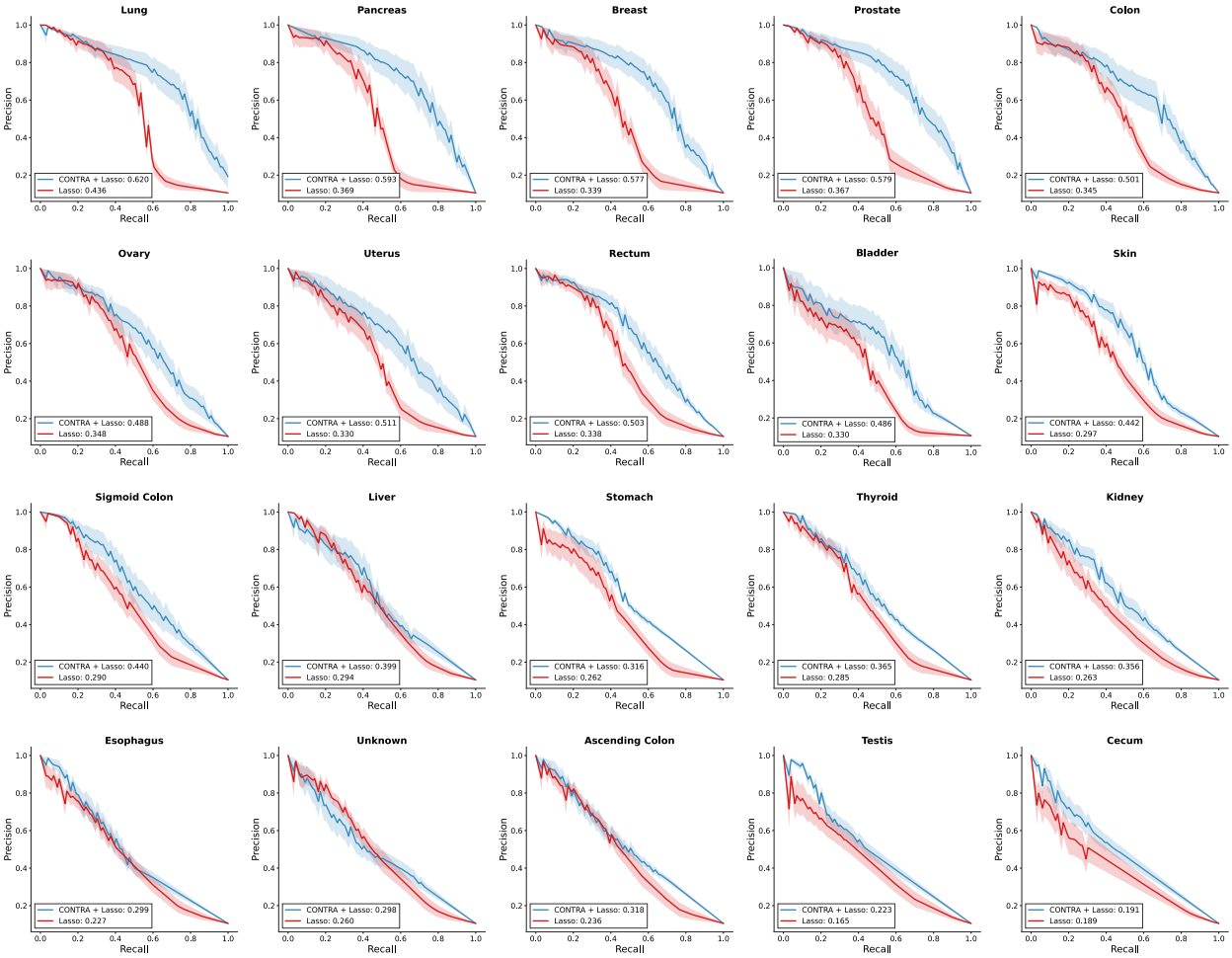


Figure 7.5

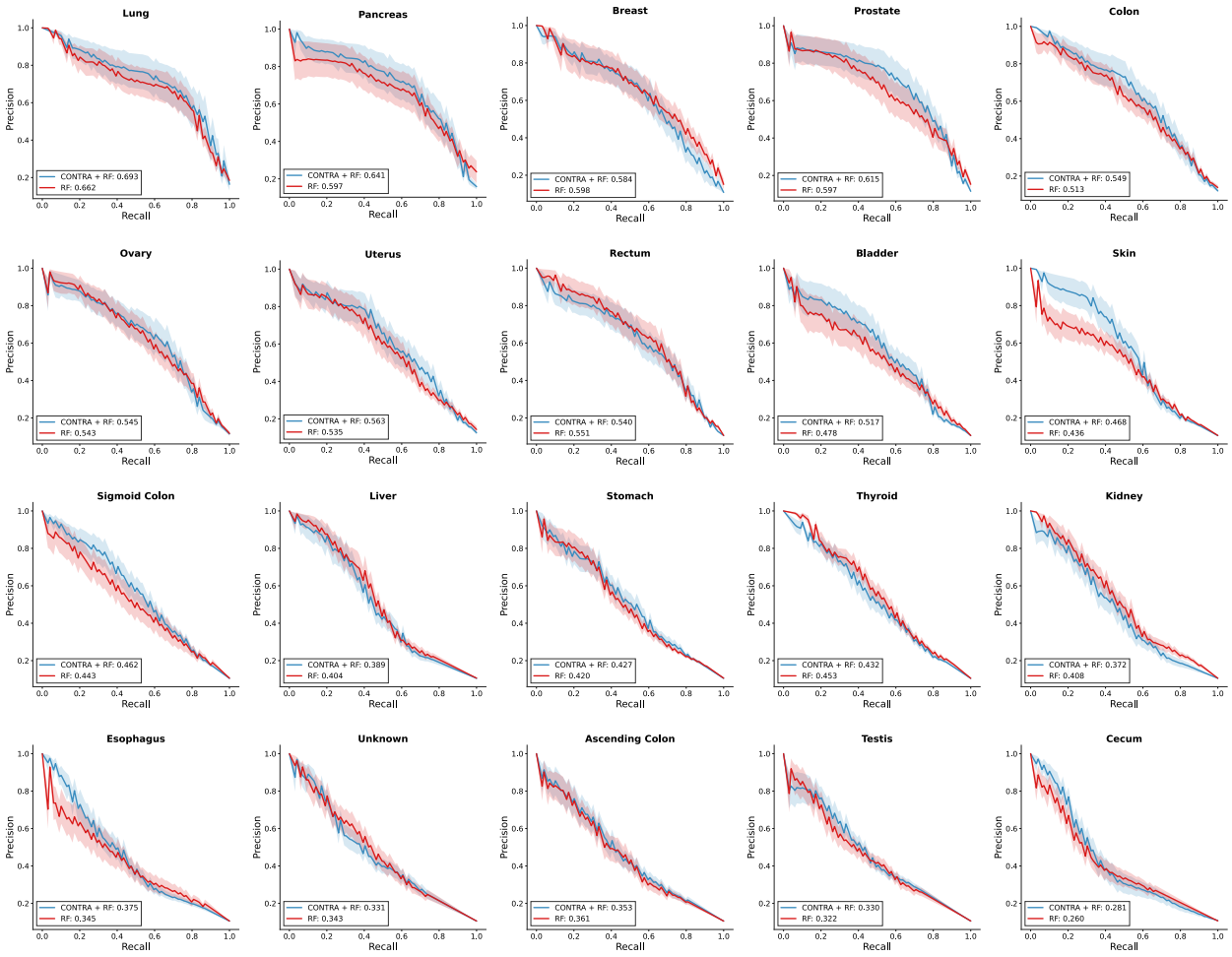

Lung
CONTRA + RF: 0.693
RF: 0.662
Pancreas
CONTRA + RF: 0.641
RF: 0.597
Breast
CONTRA + RF: 0.584
RF: 0.598
Prostate
CONTRA + RF: 0.615
RF: 0.597
Colon
CONTRA + RF: 0.549
RF: 0.513
Ovary
CONTRA + RF: 0.545
RF: 0.543
Uterus
CONTRA + RF: 0.563
RF: 0.535
Rectum
CONTRA + RF: 0.540
RF: 0.551
Bladder
CONTRA + RF: 0.517
RF: 0.478
Skin
CONTRA + RF: 0.468
RF: 0.436
Sigmoid Colon
CONTRA + RF: 0.462
RF: 0.443
Liver
CONTRA + RF: 0.389
RF: 0.404
Stomach
CONTRA + RF: 0.427
RF: 0.420
Thyroid
CONTRA + RF: 0.432
RF: 0.453
Kidney
CONTRA + RF: 0.372
RF: 0.408
Esophagus
CONTRA + RF: 0.375
RF: 0.345
Unknown
CONTRA + RF: 0.331
RF: 0.343
Ascending Colon
CONTRA + RF: 0.353
RF: 0.361
Testis
CONTRA + RF: 0.330
RF: 0.322
Cecum
CONTRA + RF: 0.281
RF: 0.260
Precision
Recall

Figure 7.6

8 | Interpretable models for RNA splicing

Molecular biology is an area where machine learning can lead to scientific discovery. Genomes encode information about life through multiple complex and partly overlapping codes. A classical example is the genetic code, describing how proteins are encoded in DNA, through the use of three DNA bases to encode each amino acid. Being sufficiently simple, the key ingredients of the genetic code were discovered in the 1960s without the use of machine learning models. However, most other biological codes appear to be more complex, and have so far been recalcitrant to attempts to deciphering them. Recent work has demonstrated accurate predictions of biological codes [Jaganathan et al. 2019], but often uses black box models which are unable to *explain* the underlying codes. Prediction alone is insufficient to yield generalizable discoveries that apply to other biological contexts. This limits the application of such recent work to the identification of molecular mechanisms, or the development of therapeutic interventions.

Here we present an explainable neural network model that provides novel insights into biological codes. In addition to achieving predictive performance on par with the state of the art, the model's decision making process is highly interpretable, leading to several biologically-verifiable hypotheses. The neural network is trained using data from a high-throughput targeted experiment.

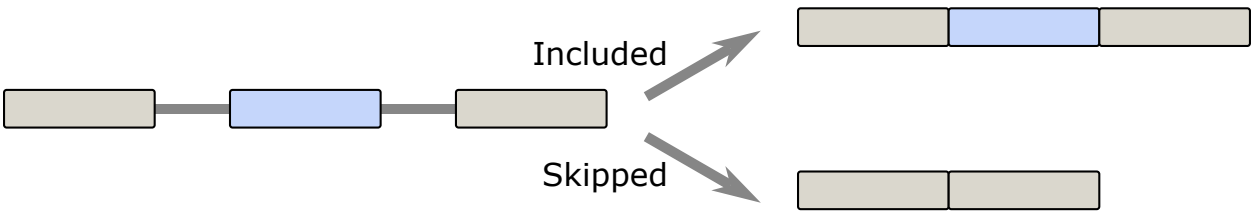
The structure of this work is as follows. First we provide an overview of the splicing code.

We then outline a dataset that can help a machine learning model learn the determinants of the splicing code. Next we present the main findings about the splicing code that were generated by our model. Finally, we describe in detail the construction of the interpretable machine learning model.

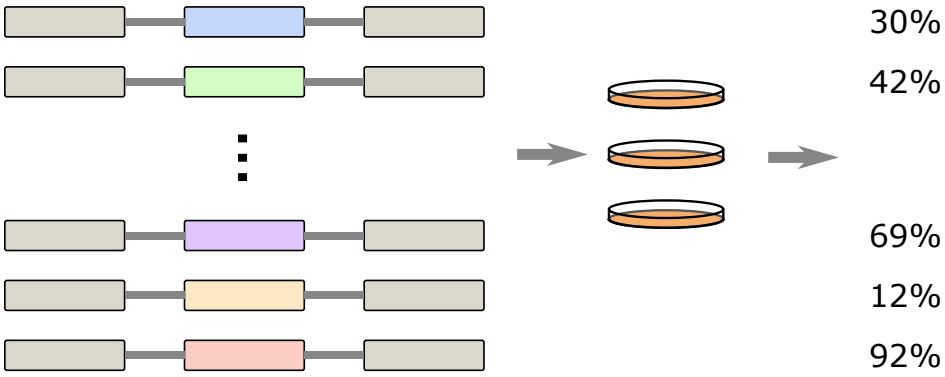
8.1 The splicing code

RNA processing plays critical roles in the fundamental transfer of information from DNA to functional RNA and protein products. One key RNA processing step is splicing. During splicing, parts of an RNA transcript known as introns are removed, and the remaining parts, known as exons, are jointed together to form the mature RNA transcript. The determination of which parts of the transcript are exons depends on a complex code known as the **splicing code**. While some canonical sequence features are necessary for defining exons (splice sites delimiting the exons and branch points), the sequence of the exon itself is known to play a critical role in determining exon definition [Kashima and Manley 2003; Cheung et al. 2019]. It is still not fully understood how the sequence of an exon determines whether it would be included or skipped.

Many short sequence features have been reported to contribute to exon definition. These sequences are known to be identified by RNA binding proteins involved in splicing. However, their quantitative contribution and how they combine to form exon definition decisions has remained unclear. It is also unclear whether there are additional, yet-unidentified, sequence features that strongly contribute to exon definition. Finally, the effect of RNA folding (secondary structure) on splicing decision is ambiguous. In summary, while some components of the splicing code were identified, understanding the splicing code as a whole remains a significant challenge. In particular, it is difficult to take a random exon sequence and understand the logic leading to its splicing decision.



(a) Alternative splicing. The mature RNA transcript from a given sequence is not deterministic: there are multiple possible events. The rectangles represent exons, the lines represent introns. The middle exon, shown in blue is either included or skipped from the mature RNA transcript. The probability that a given RNA transcript results in inclusion of the middle exon is termed "PSI."



(b) An assay is generated by uniformly sampling the middle exon. These sequences are transfected into cells, then their PSI is observed.

Figure 8.1: Reporter assay: generating training/validation data for our machine learning model.

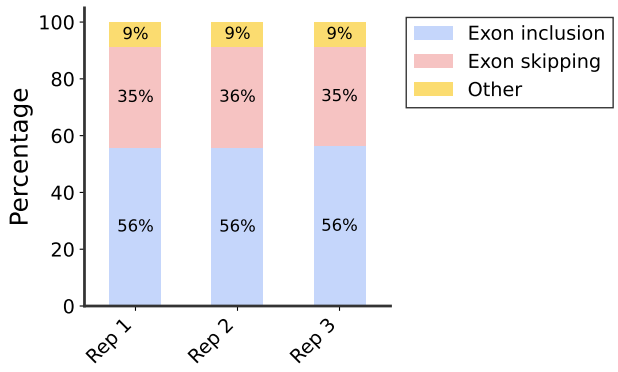


Figure 8.2: The majority of splicing products correspond to exon inclusion or exon skipping.

8.2 Dataset for machine learning

To address the challenge of understanding RNA splicing, we create a massive dataset containing over 200K exon sequences, each being a random 70 nucleotide sequence. This dataset contains observations of the following process: an RNA transcript consists of three exons separated by introns. The mature RNA transcript sometimes contains the middle exon, and sometimes does not. This is a specific instance of a phenomenon called alternative splicing, illustrated in fig. 8.1(a). Each RNA transcript is labeled by the measured "percent spliced in" (PSI) value, which represents the percentage of events where the middle exon is included in the mature RNA transcript.

To generate this dataset, we use a synthetic reporter assay, shown in fig. 8.1(b). The assay allows for massively parallel, high-throughput, quantitative PSI measurements across all reporters in a single experiment. All reporters in the assay share the same three-exon design, with the same minimum elements required for expression and splicing (promoter, splice sites, branchpoints, and introns). The reporters vary in their middle exon, which contains a different random 70 nucleotide-long sequence. Each random exon sequence is paired with a unique barcode at the end of the third exon so that exon identity can be inferred in exon skipping products.

PSI for each reporter are measured after transfection into human HeLa cells. The vast majority of splicing products correspond to exon inclusion or exon skipping products as shown in fig. 8.2.

Method	KL	RMSE
GRU	0.084	0.165
Transformer	0.102	0.183
Ours	0.100	0.180

Table 8.1: Interpretable machine learning model achieves performance on par with state-of-the-art.

We filtered our data to exclude spurious splicing products, generating paired input-output data where each middle exon is associated with a measured PSI value. Also shown in fig. 8.2, three biological replicates of the assay showed excellent agreement, and their results were merged for all downstream analysis: the total number of inclusion events and skipping events were summed across replicates for each exon.

8.3 Neural network explanations of splicing logic

State-of-the-art deep learning models trained on our dataset achieve excellent prediction accuracy on a held-out test set as shown in table 8.1. However, it is difficult to extract biological insight from these models because they are not explainable. We therefore propose a novel neural network with an easily interpretable structure. The predictive accuracy of our network is comparable to that of the state-of-the-art black box models. This suggests that explainability need not come at the expense of accuracy.

To achieve explainability, the network was designed in a modular fashion as shown in fig. 8.3. Each module can be explained and visualized, and the various modules are combined using a simple mathematical rule. Specifically, one-dimensional convolutional filters are applied to the input RNA sequence to identify short sequence features of up to six nucleotides. Half are designated as contributing to exon inclusion, and the remaining are for exon skipping. An additional bank of longer filters is applied to secondary structure from a physics-based program that estimates the minimum free energy structure [Lorenz et al. 2011].

Next, to account for possible differences in the contribution of sequence filters along the exon, the output of each of these filters is adjusted linearly based on position along the exon. Finally, the position-adjusted output is passed through a Softplus non-linearity, which serves as a threshold-like function with values below the a threshold zeroed out. We refer to the resulting values as the "forces" contributed by each sequence feature. Since the combination of 1D convolution, followed by a position-specific bias, followed by a Softplus activation is used in several locations, we refer to the sequential application of these three functions as a single module termed "force computation unit" (FCU).

To reach the final prediction, we compute the total inclusion force minus the total skipping force. This difference in total force is then used to compute the predicted PSI through a learned link function. To aid in explainability and eliminate training artifacts, we employ regularization on the model-predicted forces: we encourage the network to minimize the predicted forces as much as possible while still maintaining high predictive accuracy. We also employ a custom training schedule to ensure that the patterns learned by the short convolutional filters do not overlap with those of the longer filters.

Figure 8.4 shows how our model works. In step 1 of fig. 8.4(a), the network computes the force of various sequence and structure features along the length of an exon. These sequence features are shown in fig. 8.4(b), along with their position-specific force modifier. Remarkably, many of these sequence elements agree with well-characterized RNA binding proteins involved in splicing, and known to affect exon inclusion in the same way. Structural elements resulting from RNA folding that contribute to exon skipping are also identified. In addition to previously characterized sequence elements, the network also identified novel sequence elements not previously reported in the literature, notably a G-poor element that strongly contributed to exon skipping, which we discuss in a later section.

In step 2, the model adds the total force for inclusion and the total force for skipping. Next, the model computes the difference between the total inclusion force and skipping force. Finally, it

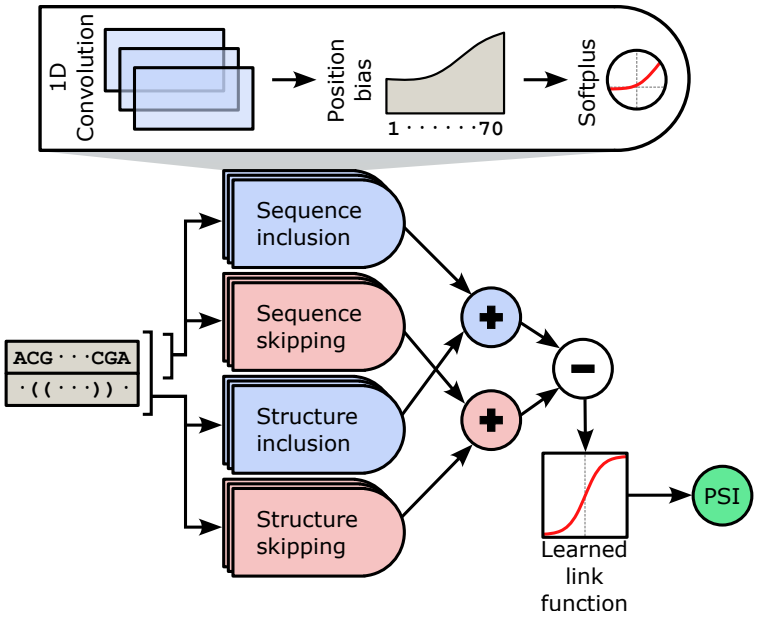


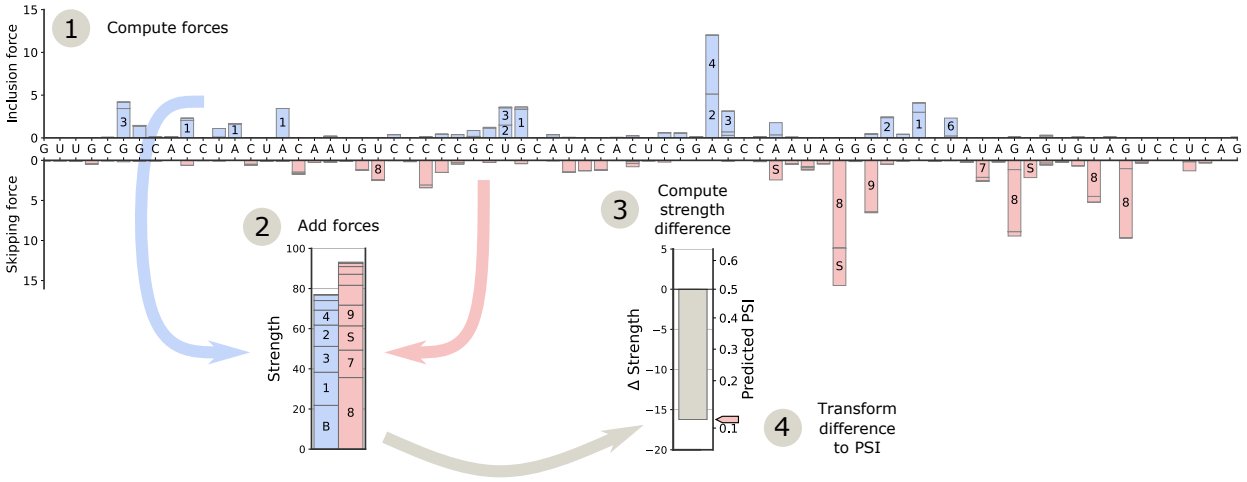
Figure 8.3: Interpretable machine learning model to predict exon inclusion. There are two force computation units (FCUs) for inclusion and two for skipping: one for sequence features and the other for secondary structure features.

uses a learned function – called the Tuner in fig. 8.3 – to map the difference in force to a predicted PSI.

8.3.1 Using interpretable machine learning for scientific discoveries

Our model yields several insights into its predictions. Here we provide specific discoveries that the model enabled. In the subsequent section, we provide details about how we validated these model-driven discoveries.

Only a few sequence features are required to determine splicing outcomes. We find that our model achieves predictive performance on par with state-of-the-art black box models despite having a relatively small number of convolutional filters. As shown in fig. 8.4(b), there are only a handful of distinct sequence motifs that determine splicing. Many of the convolutional filters learn similar sequence motifs, so we cluster similar ones together. We detail this process in section 8.5.4. Motifs contribute exclusively towards one of inclusion or skipping of the middle



(a) From sequence to PSI: under the hood of our interpretable model of splicing.

Inclusion				Skipping			
#	Motif	Strength	RBP	#	Motif	Strength	RBP
1			SRSF3/7	7			HNRNPDL
2			SRSF1/2	8			HNRNPA1
3			RBM4	9			HNRNPF/H
4			SRSF1/2	10			HNRNPK
5			RBM22	P	G-poor		
6				S	Stem loop		

(b) Sequence features learned by interpretable model.

Figure 8.4: Interpretable splicing model: discovered sequence features and model prediction logic.

exon in the reporter. We observe that increasing the number of learnable filters had no impact on model performance. Further, note that none of the sequence motifs even need 6 nucleotides: they are primarily four nucleotides long or shorter.

Splicing logic is additive. The model's FCUs compute the total force towards inclusion and skipping. The predicted PSI is a monotonic function of the difference between the total inclusion and total skipping force. Thus, given any sequence, the model can pinpoint the driving factors behind the predicted PSI.

Secondary structure contributes strongly to splicing. The model identifies secondary structures that contribute the overall splicing decision. These structures primarily consist of stem-loops of various lengths. Some examples are shown in fig. 8.5. The stems of these hairpin loop structures are strong drivers of forces for exon skipping, as evidenced by the coloring. The deepest red nucleotides in the stem contribute between 4 and 8 units of force to the splicing decision, which is a significant amount given the scale of the forces. See fig. 8.4(a) for perspective.

Secondary structure contributes only to exon skipping. We notice that the model does not learn any secondary structure features in its inclusion FCUs. This is a consistent trend across various bootstraps and random seeds.

Long G-poor regions sequences contribute to exon skipping. Using its longer convolutional filters, our model also identifies long G-poor regions in many exons without any particular secondary structure, visualized in fig. 8.6 These regions contribute to forces for exon skipping. This filter is less specific than others, and is very sensitive to the mutation of a G to a C. Even a single mutation can significantly affect the skipping force generated by the convolutional filter corresponding to the poor-G region.

As a result of this interpretable structure, once trained, the network's weights explicitly iden-

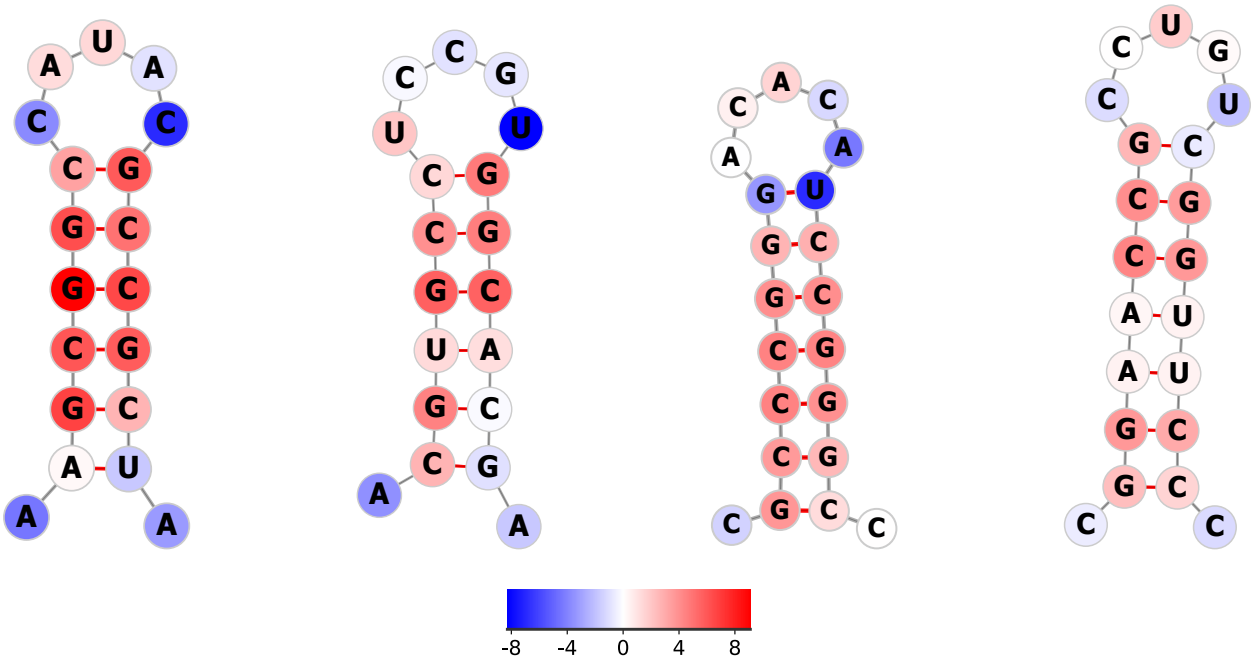


Figure 8.5: Secondary structures identified by splicing prediction model. Colorbar indicates the force contribution of each nucleotide. The redder colors contribute more to exon skipping, the blue colors contribute more to exon inclusion.

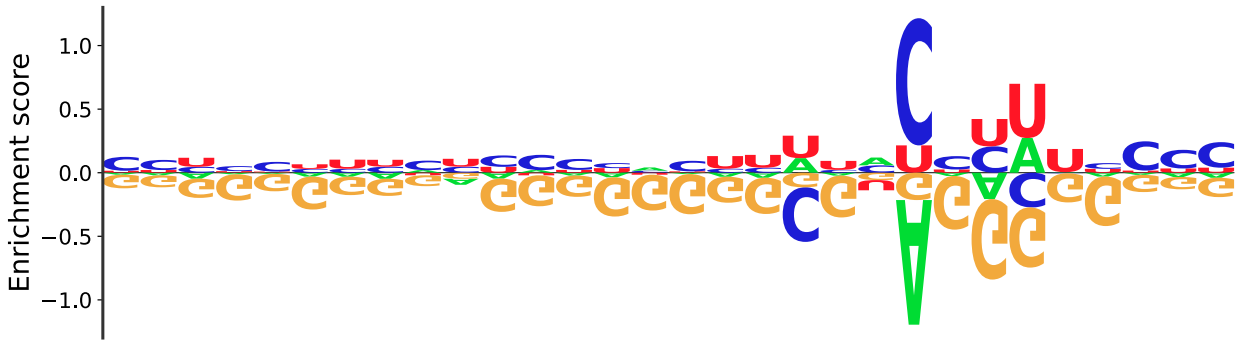


Figure 8.6: Porg

tify RNA features contributing to exon inclusion or skipping, their quantitative contribution, and how these contributions combine to determine whether an exon is included or skipped, and predicts PSI scores. The network determines whether a given exon is included or skipped by comparing the sum total of exon inclusion forces and exon skipping forces. The greater sum of forces determines PSI directionality: if the sum of exon inclusion forces is greater then the sum of exon skipping forces, then the exon is predicted to be included Importantly, the logic behind the network's PSI prediction can be examined locally for any given exon.

8.4 Experimental validation of novel sequence features

To rule out training artifacts as the source of model-driven discoveries, we conduct experiments to validate the contribution of features learned by our model.

We start by exploring the contributions of structure elements to splicing decisions. We randomly pick exons (base exons) from our library that demonstrate high stem loop force. In all these selected base exons, the hairpin structure does not involve either the 3' or 5' splice site flanking the exon, ruling out the possibility that the specific secondary structure contributes to exon skipping by occluding spliceosome assembly on the splice site. For each base exon, we generate three additional exons. The first has a single nucleotide mutation in the upstream arm of the stem that abolishes the secondary structure force. The second exon similarly has a single nucleotide mutation in the downstream arm. Importantly, the mutations are chosen so that when both are present, as in the third exon we generated, the secondary structure forms again. Notably, all mutations were chosen so that changes in forces of other motifs are minimal, ensuring that changes in prediction are truly due to changes in secondary structure, and not due to the introduction or disruption of other sequence motifs.

We tested three such exon quadruples. All three exhibited the same clear pattern, matching our predictions fig. 8.7. In both single nucleotide mutated exons (upstream and downstream), the

measured PSI increased dramatically, in agreement with the predicted lack of stem loop force. However, when both compensatory mutations are present simultaneously, the measured PSI returns to low levels, similar to that of the base exon. Together, these experiments show that RNA structure, rather than sequence, strongly contributes to individual exon skipping decisions.

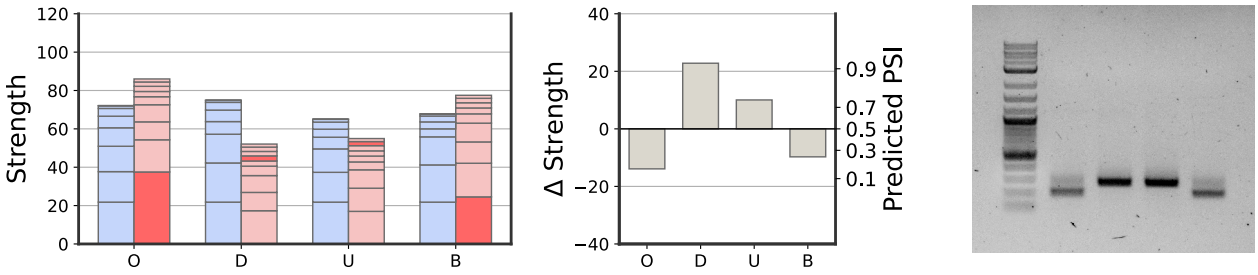
The network also identifies a G-poor element as a major contributor to exon skipping. To validate this novel element, we randomly picked four exons from our library that are predicted to contain that element. In each such exon, we introduced a single nucleotide mutation that is predicted to reduce the skipping force contributed by that element. As before, we ensured that predicted forces of other motifs are only minimally affected. Remarkably, all four exon pairs exhibited the expected pattern of increase PSI upon disruption of the G-poor element: fig. 8.8. Collectively, these validation experiments demonstrate that the structure and sequence elements identified by the network are part of the cellular splicing logic.

8.5 Methods

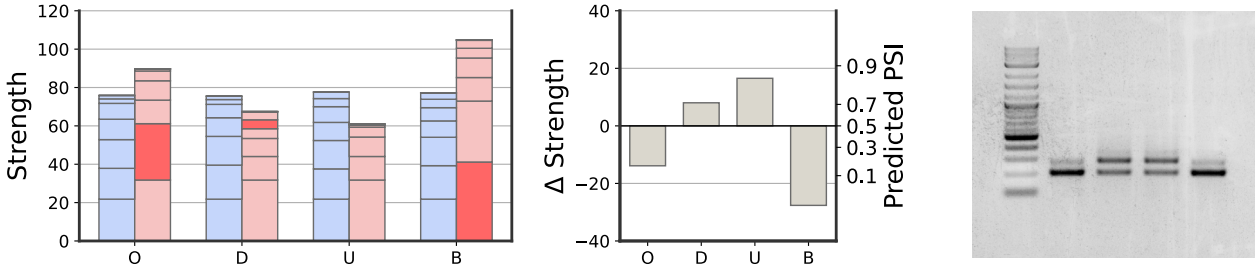
Here we describe in detail the construction of our interpretable splicing model. First we outline the data preprocessing steps we took to train the model. Then we discuss the model architecture and training procedure. We then provide exact details on how to visualize sequence and structure motifs. Finally, we give a description of how we created biological validation experiments to verify hypotheses generated by our model.

8.5.1 Data preprocessing

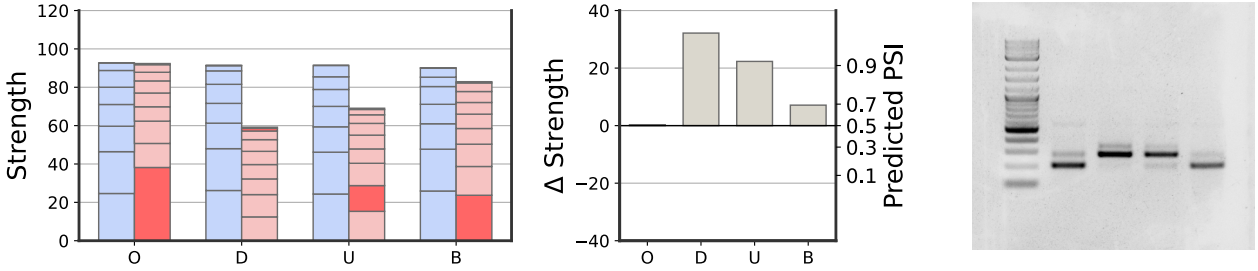
In the data preprocessing stage, we coupled barcodes to exons. We iterated over reads from the DNA sequencing of our assay to identify all exons that were coupled to a particular barcode. We filtered out barcodes associated with more than a single exon, but allowed for some ambiguous reads: the barcode could be kept if its second most coupled exon occurred no more than once.



(a) Secondary structure validation sequence 1.



(b) Secondary structure validation sequence 2.



(c) Secondary structure validation sequence 3.

Figure 8.7: Model predictions for secondary structure are validated biologically. The left column shows a breakdown of our model's forces. Red forces are towards exon skipping, blue forces are towards exon inclusion. For simplicity we omit labels for non-structure related forces. Structure is highlighted in deep red. Each force plot in the first column consists of four exons: the original (O), one with a downstream mutation (D), one with an upstream mutation (U), and one with both mutations (B). The middle column shows the difference in force between inclusion and skipping, and the model's predicted PSI. The third column shows biological validation. The higher the bars, the higher the PSI. Note the high level of agreement with model predictions.

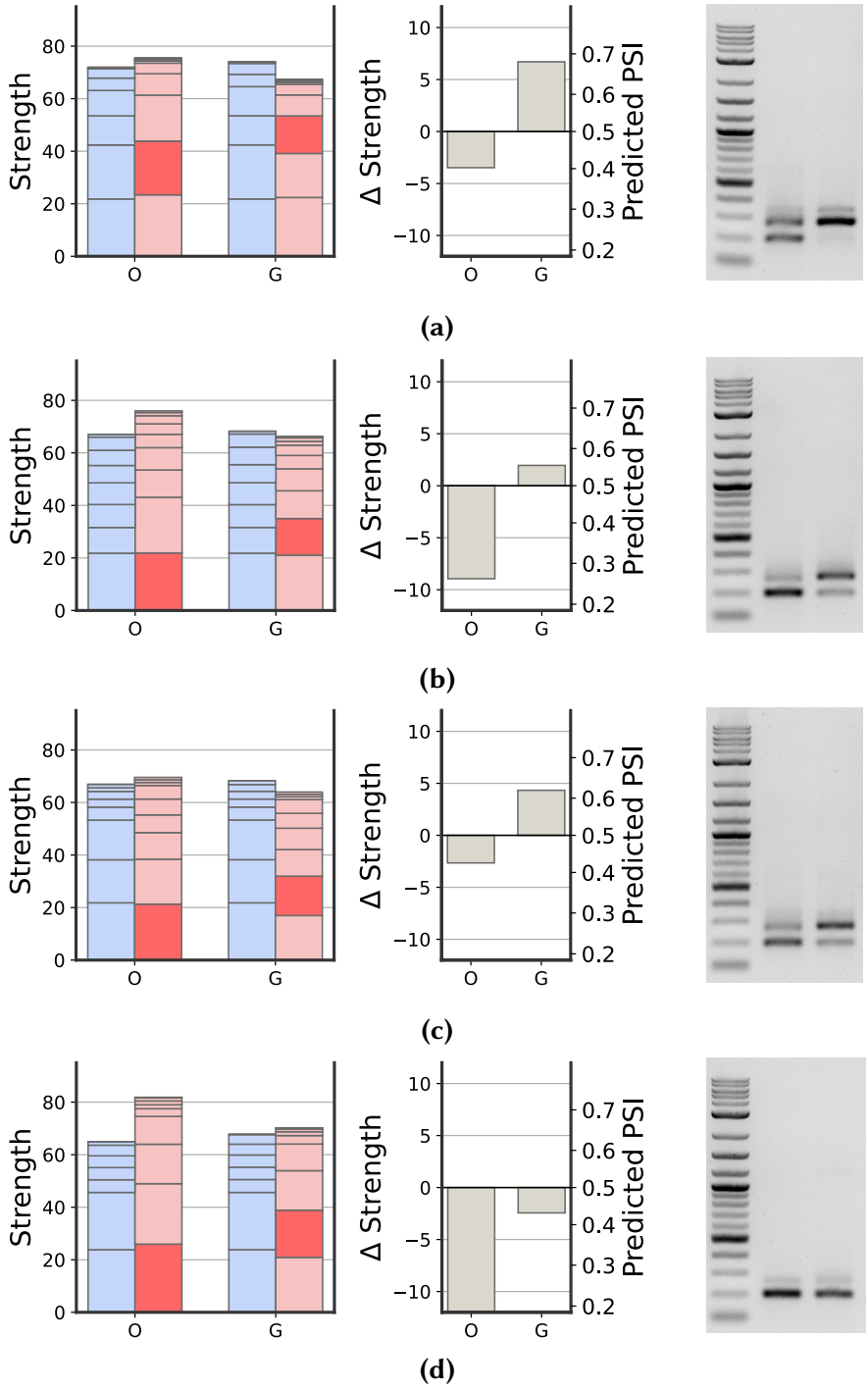


Figure 8.8: Model-driven discovery of the poor-G is validated biologically. The left column shows a breakdown of our model's forces. Red forces are towards exon skipping, blue forces are towards exon inclusion. For simplicity we omit labels for non-structure related forces. The force due to poor-G is highlighted in deep red. Each force plot in the first column consists of two exons: the original (O), and one where a C is replaced with a (G). The middle column shows the difference in force between inclusion and skipping, and the model's predicted PSI. The third column shows biological validation. The higher the bars, the higher the PSI. Note the high level of agreement with model predictions.

We also filtered out barcodes with fewer than two reads in total. The output of this stage was a list of (barcode, exon) tuples that consist of unique barcodes and their most commonly coupled exon.

We computed splicing outcome statistics for the three replicates of our assay using their RNA sequencing reads. For each replicate, each read was identified by barcode and was assigned a splicing outcome label. The potential labels included: exon skipping, exon inclusion, intron retention, splicing inside exon, or unknown splicing. The output of this stage was a set of three tables. Each table contained a barcode, its corresponding exon, and the number of times each splicing outcome label was measured for that exon.

Using unique molecular identifiers (UMIs) [König et al. 2010; Kivioja et al. 2012] we estimated the fraction of duplicate reads in each replicate to be below 23%. We therefore expect that duplicate reads would have a minimal impact on the downstream analysis.

We processed the splicing outcome tables from the previous stage into training data for our machine learning model. The first step was to sum the count statistics for each (barcode, exon) pair across replicates. For each (barcode, exon) pair, we computed the PSI:

$$\text{PSI} = \frac{n_{\text{inclusion}}}{n_{\text{skipping}} + n_{\text{inclusion}}},$$

where $n_{\text{inclusion}}$ is the number of exon inclusion reads. The quantity n_{skipping} is defined similarly. We filtered out exons with fewer than 60 total reads, exons that contained an Esp3I restriction site in either strand of the exon or its barcode, and exons where inclusion or skipping made up less than 80% of all reads.

We included fixed flanking sequences to the ends of each exon. These flanking sequences are the 10nt upstream and downstream of the exon and yielded a total input length of 90nt to the model. We also provided our model with the secondary structure of each flanked exon. This structure was predicted and encoded in dot-bracket notation using the RNAFold program in the

Vienna RNA 2.4.17 package [Lorenz et al. 2011]. We used the default parameters for RNAFold. We also provided the model with an indicator variable at each position along the flanked exon to indicate whether a particular nucleotide is part of a non-Watson Crick base pair (G-U). While this base-pairing information can be inferred from the other inputs to the model, providing it explicitly allowed the model to use this information directly.

The exon sequence and structure were encoded as one-hot vectors of length 90 (70 random nucleotides + 10nt flanking on either side). Finally, the training data was split randomly (with a fixed seed for reproducibility) into a training set and a test set in an 80/20 split.

8.5.2 Model design

Let sequence inputs $(x_{\text{seq}}, x_{\text{struct}}, x_{\text{wobble}})$ of length d be defined as

$$x_{\text{seq}} \in \{A, C, G, U\}^d \qquad \text{(sequence input)}$$

$$x_{\text{struct}} \in \{(, .,)\}^d \qquad \text{(structure input)}$$

$$x_{\text{wobble}} \in \{0, 1\}^d. \qquad \text{(wobble pair input)}$$

A Force-Computation Unit (FCU) is a neural net module $f_{\text{a}}^{\text{b}}(x; \alpha_{\text{a}}^{\text{b}}, \beta_{\text{a}}^{\text{b}})$ defined as follows:

$$f_{\text{a}}^{\text{b}} : x \mapsto \text{Sum}(\text{Softplus}(\text{Position-Bias}(\text{Convolution}(x; \alpha_{\text{a}}^{\text{b}}); \beta_{\text{a}}^{\text{b}}))) \qquad \text{(FCU)}$$

$$\alpha_{\text{a}}^{\text{b}} \in \mathbb{R}^{w_{\text{a}}^{\text{b}} \times c_{\text{a}}^{\text{b}} \times k_{\text{a}}^{\text{b}}}, \qquad \beta_{\text{a}}^{\text{b}} \in \mathbb{R}^{(d - w_{\text{a}}^{\text{b}} + 1) \times k_{\text{a}}^{\text{b}}}$$

where $x \in \{[x_{\text{seq}}], [x_{\text{seq}}, x_{\text{struct}}, x_{\text{wobble}}]\}$, a $\in \{\text{incl}, \text{skip}\}$, and b $\in \{\text{seq}, \text{struct}\}$. The 1D convolutional layer Convolution$(\cdot; \alpha_{\text{a}}^{\text{b}})$ consists of k_{a}^{b} filters each of width w_{a}^{b}. The number of input channels c_{a}^{b} is 4 if the input to the FCU is only $[x_{\text{seq}}]$, and 8 otherwise. The output of the Convolution layer is a $(d - w_{\text{a}}^{\text{b}} + 1) \times k_{\text{a}}^{\text{b}}$ matrix z of "raw" forces. The Position-Bias layer maps inputs z to $z + \beta_{\text{a}}^{\text{b}}$. This allows each raw force to be adjusted based on its position along the exon. Each

position-adjusted force passes through a Softplus activation. The output of the FCU f_{a}^{b} is the sum of all values in the output of the Softplus layer.

The splicing prediction model $m(x_{\text{seq}}, x_{\text{struct}}, x_{\text{wobble}}; \theta)$ can then be defined using FCUs as follows:

$$\begin{aligned} m(x_{\text{seq}}, x_{\text{struct}}, x_{\text{wobble}}; \theta) = &\text{Tuner}\big(f_{\text{incl}}^{\text{seq}}([x_{\text{seq}}]) + f_{\text{incl}}^{\text{struct}}([x_{\text{seq}}, x_{\text{struct}}, x_{\text{wobble}}]) \\ &- f_{\text{skip}}^{\text{seq}}([x_{\text{seq}}]) - f_{\text{skip}}^{\text{struct}}([x_{\text{seq}}, x_{\text{struct}}, x_{\text{wobble}}]); \gamma\big). \end{aligned}$$

This model computes the total force for inclusion and for skipping and uses their difference to predict splicing outcomes. The function $\text{Tuner}(\cdot; \gamma) : \mathbb{R} \to [0, 1]$ is a learned nonlinear activation function that maps this difference to a splicing probability. It consists of a 3-layer fully connected network with a residual connection from the input to the output layer, followed by a sigmoid activation. The parameter set θ contains each parameter of each FCU and the parameter γ.

Recall from earlier that we used $d = 90$ for the model's input length. Our model used 20 convolutional filters of width 6 for each sequence FCU, and 8 convolutional filters of width 30 for each structure FCU.

8.5.3 Model training

We implemented our model in Python 3.8 [Van Rossum and Drake 2009] using Tensorflow 2.6 [Abadi et al. 2015] and Numpy 1.20 [Harris et al. 2020]. We used batched gradient descent to optimize the model's parameters using the Adam optimizer. Hyperparameters such as regularization parameters were tuned with grid search. Training the model took 45 minutes on a single CPU core with 16GB of RAM.

We created a custom training schedule to maximize the interpretability of our model. To ensure that sequence motifs are not learned by structure FCUs, we trained the full model in steps, progressively adding learnable parameters in each step. The first model we trained has the

form:

$$\sigma(\nu(f_{\text{incl}}^{\text{seq}}([x_{\text{seq}}]) - f_{\text{skip}}^{\text{seq}}([x_{\text{seq}}])) + \eta)$$
$$\nu, \eta \in \mathbb{R}.$$

This model computes only sequence forces, applies a linear transformation via ν, η to the difference in sequence forces, then applies a sigmoid transformation σ. We then trained the following model, initializing the sequence FCU weights to those from the previous model:

$$f_{\text{incl}} = f_{\text{incl}}^{\text{seq}}([x_{\text{seq}}]) + f_{\text{incl}}^{\text{struct}}([x_{\text{seq}}, x_{\text{struct}}, x_{\text{wobble}}])$$
$$f_{\text{skip}} = f_{\text{skip}}^{\text{seq}}([x_{\text{seq}}]) + f_{\text{skip}}^{\text{struct}}([x_{\text{seq}}, x_{\text{struct}}, x_{\text{wobble}}])$$
$$\sigma(\nu(f_{\text{incl}} - f_{\text{skip}}) + \eta)$$

Finally, we replaced the transformation defined by ν, η and σ with the learned nonlinear activation function $\text{Tuner}(\cdot; \gamma)$:

$$\text{Tuner}(f_{\text{incl}} - f_{\text{skip}}; \gamma).$$

This final model's FCU weights were initialized to those of the previous model.

To remove unnecessary motifs and to prevent the same motifs from being learned in both inclusion and skipping FCUs, we employed activity regularization. This involved adding a term to the loss function of the splicing model. This term computes the ℓ_1 norm of the output of the SoftPlus layer of each FCU and scales it by a hyperparameter $\lambda_{\text{activity}}$.

To ensure that the FCUs' Position-Bias layers are interpretable, we applied smoothness regularization to its weights. We added the ℓ_2 norm between each pair of adjacent weights in the Position-Bias layer, and weight this sum by a hyperparameter $\lambda_{\text{smoothness}}$.

We optimized each hyperparameter on two criteria: held-out KL-divergence and sparsity of

activations. In this context, sparsity means the minimum number of activations needed per exon to achieve sufficiently low KL. Given a set of models that are sufficiently performant and sparse, we chose the one with the highest smoothness regularization.

8.5.4 Visualizing sequence and structure motifs

We visualize the sequence motifs learned by the model by applying the following procedure to the sequence FCUs. We identify 6-mers that highly activate each convolutional filter in the FCUs and use these 6-mers to compute a sequence logo for each filter. To avoid reporting redundant motifs, we apply a clustering procedure to group similar motifs.

For each pair of one-hot-encoded 6-mer and sequence convolutional filter we computed the dot-product between the 6-mer and the filter's weights. The resulting motif for the filter was generated by computing a sequence logo [Schneider and Stephens 1990] using all 6-mers whose dot-product is above zero.

To cluster redundant motifs, we computed the total activation of each convolutional filter along the length of each exon. We then applied hierarchical clustering using Scipy [Virtanen et al. 2020] to group similar convolutional filters.

We also visualized structure motifs learned by our model. Since the structure FCU's convolution filters are of length 30 and the set of 30-mers cannot be tractably enumerated, we chose to randomly sample 30-mers from contiguous segments of exons in our dataset. Using roughly 60K 30-mers, we computed a "sequence" logo for structure in a manner similar to the sequence motifs. We used an alphabet of dots and brackets instead of $\{A, C, G, U\}$. Stem-loop filters were identified by manual inspection of the structure logos.

8.5.5 Visualizing representative secondary structures

To understand which types of secondary structures yield strong activations within the model, we designed a neural network saliency approach that visualizes the output of the network when a single nucleotide is perturbed.

First, we defined a scalar-valued "response" function that computes the total structure activation across positions and filters in the structure filter group given a sequence. The response function was applied to the sequence we sought to visualize to generate a “baseline” activation. For each position in a sequence, we created three mutated sequences: these sequences contain one of the remaining three mutations at that position. For example, if a sequence is CG, then for position 1, we create the set {AG, UG, GG}. The response function was applied to each of the mutated sequences for that position to create three mutated activations. Finally, the mean absolute difference between each mutated activation and the baseline activation is defined to be the saliency for a particular position.

We compute the saliency for all positions across an exon. For ease of interpretations, we normalize the saliencies for a given sequence by dividing each position’s saliency by the maximum for the sequence. Intuitively, if a mutation at a position causes structure to break, the saliency of that position will be high.

Finally, we used this saliency computation to differentially color

8.5.6 Secondary structure validation

We selected candidate exons from our library that highly activate our model’s medium length stem-loop filter. The medium length filter activates most frequently among the model’s stem-loop filters. We did so by computing the total force attributed to this stem-loop filter for each exon, then selecting exons whose total stem-loop force was above 20, or roughly the 99th percentile. We counted the number of activations for each exon by computing a signal that measures a stem-

loop filter's activation at every location along the exon, then counting the number of peaks in this signal that exceed a threshold. This threshold was set via visual inspection of a random sampling of a few signals. The output of this stage is a list of 2384 exons.

We then counted the number of times each exon activates the model's three stem-loop filter. We filtered out exons where the total number of activations across the stem-loop filters is greater than 1. This is done so any structure within the exon may be broken easily with a single mutation. For each of the remaining 418 exons, we subsequently used the model to localize the stem and loop. Note that we did not use Vienna RNA and instead relied on the model's own structure FCUs. The output of this stage is a list of candidate exons that pass these filtering steps.

For each exon from the previous stage, we created a set of four constructs. The first construct is the original exon. The second construct breaks the structure present in the original exon. The upstream nucleotide of the base-pair in the middle of the stem formed by the original exon is mutated to break the base-pairing. The third construct is created similarly by mutating the downstream nucleotide in the base-pair instead. The final construct contains the mutations of both the second and third constructs. We ensured that secondary structure was reintroduced in this construct by choosing mutations that prevent base-pairing when only one mutation is applied, but enable base-pairing when both mutations are applied. These mutations act as compensatory mutations [Williamson et al. 1989] for the presence of secondary structure. The output of this stage is a list of 4-tuples; each 4-tuple contains the set of constructs for a candidate exon.

8.5.7 Porg motif validation

One convolutional filter in a structure FCU of our model learns a long G-poor sequence motifs called "Poor-in-Gs" (Porgs). This experiment asked whether the presence of Porgs in exons has any impact on the ψ, or if Porgs are simply an artifact of our model. For exons that activate our model's Porg filter, we introduced a single mutation that breaks the Porg. We describe each stage of this process next.

We used our model to select candidate exons that highly activate the Porg filter exactly once along the length of the exon and have a predicted PSI within $[0.1, 0.8]$. The output of this stage is a list of candidate exons.

Using the list of filtered candidate exons, we mutated the nucleotide in the center of the window where the Porg filter activates. We replaced the existing nucleotide with a G. To ensure that adding this G did not accidentally create or destroy other sequence or structure motifs, we filtered out exons where the non-Porg activations were different from those of their mutated counterpart. This allowed us to observe the effect of the Porg while keeping the other parts of the exon relatively unchanged.

9 | Conclusion

In this thesis, we presented several ways machine learning can be used to generate scientific discoveries in healthcare. We introduced several methodological contributions in the form of conditional independence tests via DDLK, DIET, and CONTRA. We applied such tests to two important problems in healthcare. We identified a small subset of highly predictive variables for adverse events in the ICU, and helped doctors quickly free up beds for patients in need at NYU Langone. We built methods to help scientists at Memorial Sloan Kettering Cancer Center identify causal tumor mutations. With this knowledge, doctors can identify patients who have tumor mutations that make their cancers more easily treatable.

Finally, we detailed an interpretable machine learning model designed to shed light on an important biological process, RNA splicing. Using this model, we generated several novel insights about this process and validated them in the wet lab.

We hope that the methods and results presented in this paper enrich the discovery toolkit of scientists and inspire future work on this critical area.

Bibliography

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mane, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viegas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Adamovic, S., Amundsen, S., Lie, B., Gudjonsdottir, A., Ascher, H., Ek, J., Van Heel, D., Nilsson, S., Sollid, L., and Naluai, Å. T. (2008). Association study of il2/il21 and fcgriia: significant association with the il2/il21 region in scandinavian coeliac disease families. *Genes and immunity*, 9(4):364.

Al-Najjar, H. and Al-Rousan, N. (2020). A classifier prediction model to predict the status of coronavirus covid-19 patients in south korea. *European Review for Medical and Pharmacological Sciences*.

Baehrens, D., Schroeter, T., Harmeling, S., Kawanabe, M., Hansen, K., and Müller, K.-R. (2010). How to explain individual classification decisions. *The Journal of Machine Learning Research*, 11:1803–1831.

Barber, R. F., Candes, E. J., Samworth, R. J., et al. (2020). Robust inference with knockoffs. *Annals of Statistics*, 48(3):1409–1431.

Bates, S., Candès, E., Janson, L., and Wang, W. (2021). Metropolized knockoff sampling. *Journal of the American Statistical Association*, 116(535):1413–1427.

Bates, S., Sesia, M., Sabatti, C., and Candès, E. (2020). Causal inference in genetic trio studies. *Proceedings of the National Academy of Sciences*, 117(39):24117–24126.

Bellot, A. and van der Schaar, M. (2019). Conditional independence testing using generative adversarial networks. In *Advances in Neural Information Processing Systems*, pages 2202–2211.

Benjamini, Y. and Hochberg, Y. (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal statistical society: series B (Methodological)*, 57(1):289–300.

Benjamini, Y. and Yekutieli, D. (2001). The control of the false discovery rate in multiple testing under dependency. *Annals of statistics*, pages 1165–1188.

Bhattacharya, P. K. and Gangopadhyay, A. K. (1990). Kernel and nearest-neighbor estimation of a conditional quantile. *The Annals of Statistics*, pages 1400–1415.

Bi, X., Su, Z., Yan, H., Du, J., Wang, J., Chen, L., Peng, M., Chen, S., Shen, B., and Li, J. (2020). Prediction of severe illness due to covid-19 based on an analysis of initial fibrinogen to albumin ratio and platelet count. *Platelets*, 31(5):674–679.

Bishop, C. M. (1994a). Mixture density networks. Technical report, Aston University, Birmingham, UK.

Bishop, C. M. (1994b). Mixture density networks.

Borghesi, A., Zigliani, A., Golemi, S., Carapella, N., Maculotti, P., Farina, D., and Maroldi, R. (2020). Chest x-ray severity index as a predictor of in-hospital mortality in coronavirus disease 2019: A study of 302 patients from italy. *International Journal of Infectious Diseases*, 96:291–293.

Burian, E., Jungmann, F., Kaissis, G. A., Lohöfer, F. K., Spinner, C. D., Lahmer, T., Treiber, M., Dommasch, M., Schneider, G., Geisler, F., et al. (2020). Intensive care risk estimation in covid-19 pneumonia based on clinical and imaging parameters: experiences from the munich cohort. *Journal of Clinical Medicine*, 9(5):1514.

Bush, W. S. and Moore, J. H. (2012). Genome-wide association studies. *PLoS computational biology*, 8(12):e1002822.

Calus, M. P. and Vandenplas, J. (2018). Snprune: an efficient algorithm to prune large snp array and sequence datasets based on high linkage disequilibrium. *Genetics Selection Evolution*, 50(1):34.

Candes, E., Fan, Y., Janson, L., and Lv, J. (2018). Panning for gold:'model-x'knockoffs for high dimensional controlled variable selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 80(3):551–577.

Castro, V. M., McCoy, T. H., and Perlis, R. H. (2020). Laboratory findings associated with severe illness and mortality among hospitalized individuals with coronavirus disease 2019 in eastern massachusetts. *JAMA network open*, 3(10):e2023934–e2023934.

Cecconi, M., Piovani, D., Brunetta, E., Aghemo, A., Greco, M., Ciccarelli, M., Angelini, C., Voza, A., Omodei, P., Vespa, E., et al. (2020). Early predictors of clinical deterioration in a cohort of 239 patients hospitalized for covid-19 infection in lombardy, italy. *Journal of clinical medicine*, 9(5):1548.

Chen, J., Song, L., Wainwright, M. J., and Jordan, M. I. (2018). Learning to explain: An information-theoretic perspective on model interpretation. *arXiv preprint arXiv:1802.07814*.

Chen, R., Liang, W., Jiang, M., Guan, W., Zhan, C., Wang, T., Tang, C., Sang, L., Liu, J., Ni, Z., et al. (2020). Risk factors of fatal outcome in hospitalized subjects with coronavirus disease 2019 from a nationwide analysis in china. *Chest*, 158(1):97–105.

Cheng, D. T., Mitchell, T. N., Zehir, A., Shah, R. H., Benayed, R., Syed, A., Chandramohan, R., Liu, Z. Y., Won, H. H., Scott, S. N., et al. (2015). Memorial sloan kettering-integrated mutation profiling of actionable cancer targets (msk-impact): a hybridization capture-based next-generation sequencing clinical assay for solid tumor molecular oncology. *The Journal of molecular diagnostics*, 17(3):251–264.

Cheng, F.-Y., Joshi, H., Tandon, P., Freeman, R., Reich, D. L., Mazumdar, M., Kohli-Seth, R., Levin, M. A., Timsina, P., and Kia, A. (2020). Using machine learning to predict icu transfer in hospitalized covid-19 patients. *Journal of clinical medicine*, 9(6):1668.

Cheng, J., Bell, D., and Liu, W. (1998). Learning bayesian networks from data: An efficient approach based on information theory. *On World Wide Web at http://www. cs. ualberta. ca/˜ jcheng/bnpc. htm.*

Cheung, R., Insigne, K. D., Yao, D., Burghard, C. P., Wang, J., Hsiao, Y.-H. E., Jones, E. M., Goodman, D. B., Xiao, X., and Kosuri, S. (2019). A multiplexed assay for exon recognition reveals that an unappreciated fraction of rare genetic variants cause large-effect splicing disruptions. *Molecular cell*, 73(1):183–194.

Covert, I. and Lee, S.-I. (2021). Improving kernelshap: Practical shapley value estimation using linear regression. In *International Conference on Artificial Intelligence and Statistics*, pages 3457–3465. PMLR.

COVID, C. (19). global cases by the center for systems science and engineering (csse) at johns hopkins university (jhu).

Crammer, K., Kearns, M., and Wortman, J. (2008). Learning from multiple sources. *Journal of Machine Learning Research*, 9(Aug):1757–1774.

Daudin, J. (1980). Partial association measures and an application to qualitative regression. *Biometrika*, 67(3):581–590.

Daumé III, H. (2009). Frustratingly easy domain adaptation. *arXiv preprint arXiv:0907.1815*.

De Campos, L. M. and Huete, J. F. (2000). A new approach for learning belief networks using independence criteria. *International Journal of Approximate Reasoning*, 24(1):11–37.

Dong, Y., Zhou, H., Li, M., Zhang, Z., Guo, W., Yu, T., Gui, Y., Wang, Q., Zhao, L., Luo, S., et al. (2020). A novel simple scoring model for predicting severity of patients with sars-cov-2 infection. *Transboundary and Emerging Diseases*, 67(6):2823–2829.

Doran, G., Muandet, K., Zhang, K., and Schölkopf, B. (2014). A permutation-based kernel conditional independence test. In *UAI*, pages 132–141. Citeseer.

Drysdale, E., Dolatabadi, E., Chivers, C., Liu, V., Saria, S., and Sendak, M. (2019). Implementing ai in healthcare.

Du, R.-H., Liang, L.-R., Yang, C.-Q., Wang, W., Cao, T.-Z., Li, M., Guo, G.-Y., Du, J., Zheng, C.-L., Zhu, Q., et al. (2020). Predictors of mortality for patients with covid-19 pneumonia caused by sars-cov-2: a prospective cohort study. *European Respiratory Journal*, 55(5).

Dubois, P. C., Trynka, G., Franke, L., Hunt, K. A., Romanos, J., Curtotti, A., Zhernakova, A., Heap, G. A., Ádány, R., Aromaa, A., et al. (2010). Multiple common variants for celiac disease influencing immune gene expression. *Nature genetics*, 42(4):295.

Efron, B. (2012). *Large-scale inference: empirical Bayes methods for estimation, testing, and prediction*, volume 1. Cambridge University Press.

Epic (2020). Epic ai helps clinicians predict when covid-19 patients might need intensive care.

Falcon, W. (2019). Pytorch lightning. *GitHub. Note: https://github. com/williamFalcon/pytorch-lightning Cited by*, 3.

Figurnov, M., Mohamed, S., and Mnih, A. (2018). Implicit reparameterization gradients. In *Advances in Neural Information Processing Systems*, pages 441–452.

Friedman, J. H. (2002). Stochastic gradient boosting. *Computational statistics & data analysis*, 38(4):367–378.

Fukumizu, K., Gretton, A., Sun, X., and Schölkopf, B. (2007). Kernel measures of conditional dependence. *Advances in neural information processing systems*, 20.

Galloway, J. B., Norton, S., Barker, R. D., Brookes, A., Carey, I., Clarke, B. D., Jina, R., Reid, C., Russell, M. D., Sneep, R., et al. (2020). A clinical risk score to identify patients with covid-19 at high risk of critical care admission or death: an observational cohort study. *Journal of Infection*, 81(2):282–288.

Gao, S., Ver Steeg, G., and Galstyan, A. (2016). Variational information maximization for feature selection. *Advances in neural information processing systems*, 29.

Geirhos, R., Jacobsen, J.-H., Michaelis, C., Zemel, R., Brendel, W., Bethge, M., and Wichmann, F. A. (2020). Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673.

Germain, M., Gregor, K., Murray, I., and Larochelle, H. (2015). MADE: masked autoencoder for distribution estimation. *CoRR*, abs/1502.03509.

Ghebreyesus, T. A. (2020). Who director-general's opening remarks at the media briefing on covid-19 - 20 may 2020.

Gong, J., Ou, J., Qiu, X., Jie, Y., Chen, Y., Yuan, L., Cao, J., Tan, M., Xu, W., Zheng, F., et al. (2020). A tool for early prediction of severe coronavirus disease 2019 (covid-19): a multicenter study using the risk nomogram in wuhan and guangdong, china. *Clinical infectious diseases*, 71(15):833–840.

Grasselli, G., Zangrillo, A., Zanella, A., Antonelli, M., Cabrini, L., Castelli, A., Cereda, D., Coluccello, A., Foti, G., Fumagalli, R., et al. (2020). Baseline characteristics and outcomes of 1591 patients infected with sars-cov-2 admitted to icus of the lombardy region, italy. *Jama*, 323(16):1574–1581.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. (2012). A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. (2017). Improved training of wasserstein gans. *CoRR*, abs/1704.00028.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Rio, J. F., Wiebe, M., Peterson, P., Gerard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., and Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, 585(7825):357–362.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). Random forests. In *The elements of statistical learning*, pages 587–604. Springer.

Heinze-Deml, C., Peters, J., and Meinshausen, N. (2018). Invariant causal prediction for nonlinear models. *Journal of Causal Inference*, 6(2).

Henry, B. M., De Oliveira, M. H. S., Benoit, S., Plebani, M., and Lippi, G. (2020). Hematologic, biochemical and immune biomarker abnormalities associated with severe illness and mortality

in coronavirus disease 2019 (covid-19): a meta-analysis. *Clinical Chemistry and Laboratory Medicine (CCLM)*, 58(7):1021–1028.

Hong, Y., Wu, X., Qu, J., Gao, Y., Chen, H., and Zhang, Z. (2020). Clinical characteristics of coronavirus disease 2019 and development of a prediction model for prolonged hospital length of stay. *Annals of translational medicine*, 8(7).

Huang, J., Cheng, A., Lin, S., Zhu, Y., and Chen, G. (2020). Individualized prediction nomograms for disease progression in mild covid-19. *Journal of medical virology*, 92(10):2074–2080.

Hunt, K. A., Zhernakova, A., Turner, G., Heap, G. A., Franke, L., Bruinenberg, M., Romanos, J., Dinesen, L. C., Ryan, A. W., Panesar, D., et al. (2008). Novel celiac disease genetic determinants related to the immune response. *Nature genetics*, 40(4):395.

Jaganathan, K., Panagiotopoulou, S. K., McRae, J. F., Darbandi, S. F., Knowles, D., Li, Y. I., Kosmicki, J. A., Arbelaez, J., Cui, W., Schwartz, G. B., et al. (2019). Predicting splicing from primary sequence with deep learning. *Cell*, 176(3):535–548.

Jang, E., Gu, S., and Poole, B. (2016). Categorical reparameterization with gumbel-softmax. *CoRR*, abs/1611.01144.

Jethani, N., Sudarshan, M., Covert, I. C., Lee, S.-I., and Ranganath, R. (2021). Fastshap: Real-time shapley value estimation. In *International Conference on Learning Representations*.

Ji, D., Zhang, D., Xu, J., Chen, Z., Yang, T., Zhao, P., Chen, G., Cheng, G., Wang, Y., Bi, J., et al. (2020a). Prediction for progression risk in patients with covid-19 pneumonia: the call score. *Clinical Infectious Diseases*, 71(6):1393–1399.

Ji, M., Yuan, L., Shen, W., Lv, J., Li, Y., Chen, J., Zhu, C., Liu, B., Liang, Z., Lin, Q., et al. (2020b). A predictive model for disease progression in non-severely ill patients with coronavirus disease 2019. *European Respiratory Journal*, 56(1).

Jiang, X., Coffee, M., Bari, A., Wang, J., Jiang, X., Huang, J., Shi, J., Dai, J., Cai, J., Zhang, T., et al. (2020). Towards an artificial intelligence framework for data-driven prediction of coronavirus clinical severity. *Computers, Materials & Continua*, 63(1):537–551.

Jordon, J., Yoon, J., and van der Schaar, M. (2018). Knockoffgan: Generating knockoffs for feature selection using generative adversarial networks. In *International Conference on Learning Representations*.

Jordon, J., Yoon, J., and van der Schaar, M. (2019). KnockoffGAN: Generating knockoffs for feature selection using generative adversarial networks. In *International Conference on Learning Representations*.

Kashima, T. and Manley, J. L. (2003). A negative element in smn2 exon 7 inhibits splicing in spinal muscular atrophy. *Nature genetics*, 34(4):460–463.

Katsevich, E. and Ramdas, A. (2020). A theoretical treatment of conditional independence testing under model-x. *arXiv preprint arXiv:2005.05506*.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in neural information processing systems*, pages 3146–3154.

Kelly, C. J., Karthikesalingam, A., Suleyman, M., Corrado, G., and King, D. (2019). Key challenges for delivering clinical impact with artificial intelligence. *BMC medicine*, 17(1):1–9.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kingma, D. P., Salimans, T., and Welling, M. (2015). Variational dropout and the local reparameterization trick. In *Advances in neural information processing systems*, pages 2575–2583.

Kivioja, T., Vähärautio, A., Karlsson, K., Bonke, M., Enge, M., Linnarsson, S., and Taipale, J. (2012). Counting absolute numbers of molecules using unique molecular identifiers. *Nature methods*, 9(1):72–74.

Koller, D. and Friedman, N. (2009). *Probabilistic graphical models: principles and techniques*. MIT press.

König, J., Zarnack, K., Rot, G., Curk, T., Kayikci, M., Zupan, B., Turner, D. J., Luscombe, N. M., and Ule, J. (2010). iclip reveals the function of hnrnp particles in splicing at individual nucleotide resolution. *Nature structural & molecular biology*, 17(7):909–915.

Lauritzen, S. L. (1996). *Graphical models*, volume 17. Clarendon Press.

Lee, S. and Honavar, V. (2017). A kernel conditional independence test for relational data. In *33rd Conference on Uncertainty in Artificial Intelligence, UAI 2017*.

Li, J., Li, M., Zheng, S., Li, M., Zhang, M., Sun, M., Li, X., Deng, A., Cai, Y., and Zhang, H. (2020a). Plasma albumin levels predict risk for nonsurvivors in critically ill patients with covid-19. *Biomarkers in Medicine*, 14(10):827–837.

Li, L.-q., Huang, T., Wang, Y.-q., Wang, Z.-p., Liang, Y., Huang, T.-b., Zhang, H.-y., Sun, W., and Wang, Y. (2020b). Covid-19 patients' clinical characteristics, discharge rate, and fatality rate of meta-analysis. *Journal of medical virology*, 92(6):577–583.

Li, Q. and Racine, J. S. (2008). Nonparametric estimation of conditional cdf and quantile functions with mixed categorical and continuous data. *Journal of Business & Economic Statistics*, 26(4):423–434.

Li, Q., Zhang, J., Ling, Y., Li, W., Zhang, X., Lu, H., and Chen, L. (2020c). A simple algorithm helps early identification of sars-cov-2 infection patients with severe progression tendency. *Infection*, 48(4):577–584.

Liang, F., Li, Q., and Zhou, L. (2018). Bayesian neural networks for selection of drug sensitive genes. *Journal of the American Statistical Association*, 113(523):955–972.

Liang, W., Liang, H., Ou, L., Chen, B., Chen, A., Li, C., Li, Y., Guan, W., Sang, L., Lu, J., et al. (2020). Development and validation of a clinical risk score to predict the occurrence of critical illness in hospitalized patients with covid-19. *JAMA internal medicine*, 180(8):1081–1089.

Lindsley, A. W., Schwartz, J. T., and Rothenberg, M. E. (2020). Eosinophil responses during covid-19 infections and coronavirus vaccination. *Journal of Allergy and Clinical Immunology*, 146(1):1–7.

Liu, J., Liu, Y., Xiang, P., Pu, L., Xiong, H., Li, C., Zhang, M., Tan, J., Xu, Y., Song, R., et al. (2020a). Neutrophil-to-lymphocyte ratio predicts critical illness patients with 2019 coronavirus disease in the early stage. *Journal of translational medicine*, 18(1):1–12.

Liu, M. and Janson, L. (2020). Fast and powerful conditional randomization testing via distillation. *arXiv preprint arXiv:2006.03980*.

Liu, X., Shi, S., Xiao, J., Wang, H., Chen, L., Li, J., and Han, K. (2020b). Prediction of the severity of corona virus disease 2019 and its adverse clinical outcomes. *Japanese journal of infectious diseases*, pages JJID–2020.

Liu, Y. and Zheng, C. (2018). Auto-encoding knockoff generator for fdr controlled variable selection. *arXiv preprint arXiv:1809.10765*.

Lorenz, R., Bernhart, S. H., Höner zu Siederdissen, C., Tafer, H., Flamm, C., Stadler, P. F., and Hofacker, I. L. (2011). Viennarna package 2.0. *Algorithms for molecular biology*, 6(1):1–14.

Lundberg, S. M. and Lee, S.-I. (2017). A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, pages 4765–4774.

Maddison, C. J., Mnih, A., and Teh, Y. W. (2016). The concrete distribution: A continuous relaxation of discrete random variables. *CoRR*, abs/1611.00712.

McRae, M. P., Simmons, G. W., Christodoulides, N. J., Lu, Z., Kang, S. K., Fenyo, D., Alcorn, T., Dapkins, I. P., Sharif, I., Vurmaz, D., et al. (2020). Clinical decision support tool and rapid point-of-care platform for determining disease severity in patients with covid-19. *Lab on a Chip*, 20(12):2075–2085.

Mei, Y., Weinberg, S. E., Zhao, L., Frink, A., Qi, C., Behdad, A., and Ji, P. (2020). Risk stratification of hospitalized covid-19 patients through comparative studies of laboratory results with influenza. *EClinicalMedicine*, 26:100475.

Mescheder, L., Geiger, A., and Nowozin, S. (2018). Which training methods for gans do actually converge? *arXiv preprint arXiv:1801.04406*.

Mescheder, L., Nowozin, S., and Geiger, A. (2017). The numerics of gans. In *Advances in Neural Information Processing Systems*, pages 1825–1835.

Miscouridou, X., Perotte, A., Elhadad, N., and Ranganath, R. (2018). Deep survival analysis: Nonparametrics and missingness. In *Machine Learning for Healthcare Conference*, pages 244–256.

Pearl, J. (2009). Causal inference in statistics: An overview. *Statistics surveys*, 3:96–146.

Peters, J., Bühlmann, P., and Meinshausen, N. (2016). Causal inference by using invariant prediction: identification and confidence intervals. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 78(5):947–1012.

Petrilli, C. M., Jones, S. A., Yang, J., Rajagopalan, H., O'Donnell, L., Chernyak, Y., Tobin, K. A., Cerfolio, R. J., Francois, F., and Horwitz, L. I. (2020). Factors associated with hospital admission and critical illness among 5279 people with coronavirus disease 2019 in new york city: prospective cohort study. *bmj*, 369.

Price, A. L., Patterson, N. J., Plenge, R. M., Weinblatt, M. E., Shadick, N. A., and Reich, D. (2006). Principal components analysis corrects for stratification in genome-wide association studies. *Nature genetics*, 38(8):904–909.

Qin, C., Ziwei, M. P. L. Z. M., Tao, S. Y. M. Y., Ke, P. C. X. M. P., and Shang, M. M. P. K. (2020). Dysregulation of immune response in patients with covid-19 in wuhan, china; clinical infectious diseases; oxford academic. *Clinical Infectious Diseases*.

Ramdas, A., Reddi, S. J., Póczos, B., Singh, A., and Wasserman, L. (2015). On the decreasing power of kernel and distance based nonparametric hypothesis tests in high dimensions. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*.

Ranganath, R. and Perotte, A. (2018). Multiple causal inference with latent confounding. *arXiv preprint arXiv:1805.08273*.

Razavian, N., Major, V. J., Sudarshan, M., Burk-Rafel, J., Stella, P., Randhawa, H., Bilaloglu, S., Chen, J., Nguy, V., Wang, W., et al. (2020). A validated, real-time prediction model for favorable outcomes in hospitalized covid-19 patients. *NPJ digital medicine*, 3(1):1–13.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144. ACM.

Richardson, S., Hirsch, J. S., Narasimhan, M., Crawford, J. M., McGinn, T., Davidson, K. W., Barnaby, D. P., Becker, L. B., Chelico, J. D., Cohen, S. L., et al. (2020). Presenting characteristics, comorbidities, and outcomes among 5700 patients hospitalized with covid-19 in the new york city area. *Jama*, 323(20):2052–2059.

Romano, Y., Sesia, M., and Candès, E. (2020). Deep knockoffs. *Journal of the American Statistical Association*, 115(532):1861–1872.

Romano, Y., Sesia, M., and Candès, E. J. (2018). Deep knockoffs.

Ruan, Q., Yang, K., Wang, W., Jiang, L., and Song, J. (2020). Clinical predictors of mortality due to covid-19 based on an analysis of data of 150 patients from wuhan, china. *Intensive care medicine*, 46(5):846–848.

Runge, J. (2018). Conditional independence testing based on a nearest-neighbor estimator of conditional mutual information. In *International Conference on Artificial Intelligence and Statistics*, pages 938–947. PMLR.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). Improved techniques for training gans. In *Advances in neural information processing systems*, pages 2234–2242.

Sattar, N., McInnes, I. B., and McMurray, J. J. (2020). Obesity is a risk factor for severe covid-19 infection: multiple potential mechanisms. *Circulation*, 142(1):4–6.

Schneider, T. D. and Stephens, R. M. (1990). Sequence logos: A new way to display consensus sequences. *Nucleic Acids Res.*, 18:6097–6100. http://dx.doi.org/10.1093/nar/18.20.6097, https://alum.mit.edu/www/toms/papers/logopaper/.

Sen, R., Suresh, A. T., Shanmugam, K., Dimakis, A. G., and Shakkottai, S. (2017). Model-powered conditional independence test. *Advances in neural information processing systems*, 30.

Sesia, M., Sabatti, C., and Candès, E. J. (2017). Gene hunting with knockoffs for hidden markov models. *arXiv preprint arXiv:1706.04677*.

Sesia, M., Sabatti, C., and Candès, E. J. (2019). Gene hunting with hidden Markov model knockoffs. *Biometrika*, 106(1):1–18.

Shah, R. D. and Peters, J. (2020). The hardness of conditional independence testing and the generalised covariance measure. *The Annals of Statistics*, 48(3):1514–1538.

Shang, W., Dong, J., Ren, Y., Tian, M., Li, W., Hu, J., and Li, Y. (2020). The value of clinical parameters in predicting the severity of covid-19. *Journal of medical virology*, 92(10):2188–2192.

Silverstein, C., Brin, S., Motwani, R., and Ullman, J. (2000). Scalable techniques for mining causal structures. *Data Mining and Knowledge Discovery*, 4(2):163–192.

Sollid, L. M. (2002). Coeliac disease: dissecting a complex inflammatory disorder. *Nature Reviews Immunology*, 2(9):647.

Spector, A. and Janson, L. (2022). Powerful knockoffs via minimizing reconstructability. *The Annals of Statistics*, 50(1):252–276.

Spirtes, P., Glymour, C. N., Scheines, R., and Heckerman, D. (2000). *Causation, prediction, and search*. MIT press.

Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. (2014). Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*.

Sudarshan, M., Puli, A., Subramanian, L., Sankararaman, S., and Ranganath, R. (2021). Contra: Contrarian statistics for controlled variable selection. In *International Conference on Artificial Intelligence and Statistics*, pages 1900–1908. PMLR.

Sudarshan, M., Tansey, W., and Ranganath, R. (2020). Deep direct likelihood knockoffs.

Sundararajan, M., Taly, A., and Yan, Q. (2017). Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR.

Tansey, W., Veitch, V., Zhang, H., Rabadan, R., and Blei, D. M. (2018a). The holdout randomization test: Principled and easy black box feature selection. *arXiv preprint arXiv:1811.00645*.

Tansey, W., Wang, Y., Blei, D. M., and Rabadan, R. (2018b). Black box fdr. *arXiv preprint arXiv:1806.03143*.

Toussie, D., Voutsinas, N., Finkelstein, M., Cedillo, M. A., Manna, S., Maron, S. Z., Jacobi, A., Chung, M., Bernheim, A., Eber, C., et al. (2020). Clinical and chest radiography features determine patient outcomes in young and middle-aged adults with covid-19. *Radiology*, 297(1):E197.

Trivedi, P. K. and Zimmer, D. M. (2007). *Copula modeling: an introduction for practitioners*. Now Publishers Inc.

Tsamardinos, I., Aliferis, C. F., Statnikov, A. R., and Statnikov, E. (2003). Algorithms for large scale markov blanket discovery. In *FLAIRS conference*, volume 2, pages 376–380. St. Augustine, FL.

Tsybakov, A. B. (2008). *Introduction to nonparametric estimation*. Springer Science & Business Media.

Tukey, J. W. (1953). The problem of multiple comparisons. *Multiple comparisons*.

Van Rossum, G. and Drake, F. L. (2009). *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA.

Vinh, N. X., Epps, J., and Bailey, J. (2009). Information theoretic measures for clusterings comparison: is a correction for chance necessary? In *Proceedings of the 26th annual international conference on machine learning*, pages 1073–1080.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, I., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0

Contributors (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.

Wang, D., Hu, B., Hu, C., Zhu, F., Liu, X., Zhang, J., Wang, B., Xiang, H., Cheng, Z., Xiong, Y., et al. (2020a). Clinical characteristics of 138 hospitalized patients with 2019 novel coronavirus–infected pneumonia in wuhan, china. *Jama*, 323(11):1061–1069.

Wang, K., Zuo, P., Liu, Y., Zhang, M., Zhao, X., Xie, S., Zhang, H., Chen, X., and Liu, C. (2020b). Clinical and laboratory predictors of in-hospital mortality in patients with coronavirus disease-2019: a cohort study in wuhan, china. *Clinical infectious diseases*, 71(16):2079–2088.

Williamson, C. L., Desai, N. M., and Burke, J. M. (1989). Compensatory mutations demonstrate that p8 and p6 are rna secondary structure elements important for processing of a group i intron. *Nucleic acids research*, 17(2):675–689.

Xia, X., Wen, M., Zhan, S., He, J., and Chen, W. (2020). An increased neutrophil/lymphocyte ratio is an early warning signal of severe covid-19. *Nan fang yi ke da xue xue bao= Journal of Southern Medical University*, 40(3):333–336.

Yan, L., Zhang, H.-T., Goncalves, J., Xiao, Y., Wang, M., Guo, Y., Sun, C., Tang, X., Jing, L., Zhang, M., et al. (2020). An interpretable mortality prediction model for covid-19 patients. *Nature machine intelligence*, 2(5):283–288.

Yang, W., Soares, J., Greninger, P., Edelman, E. J., Lightfoot, H., Forbes, S., Bindal, N., Beare, D., Smith, J. A., Thompson, I. R., et al. (2012). Genomics of drug sensitivity in cancer (gdsc): a resource for therapeutic biomarker discovery in cancer cells. *Nucleic acids research*, 41(D1):D955–D961.

Yoon, J., Jordon, J., and van der Schaar, M. (2019). INVASE: Instance-wise variable selection using neural networks. In *International Conference on Learning Representations*.

Yu, C., Lei, Q., Li, W., Wang, X., Liu, W., Fan, X., and Li, W. (2020). Clinical characteristics, associated factors, and predicting covid-19 mortality risk: a retrospective study in wuhan, china. *American journal of preventive medicine*, 59(2):168–175.

Yu, T. (2012). Rocs: receiver operating characteristic surface for class-skewed high-throughput data. *PloS one*, 7(7):e40598.

Yun, H., Sun, Z., Wu, J., Tang, A., Hu, M., and Xiang, Z. (2020). Laboratory data analysis of novel coronavirus (covid-19) screening in 2510 patients. *Clinica Chimica Acta*, 507:94–97.

Zeiler, M. D. and Fergus, R. (2014). Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer.

Zeng, F., Huang, Y., Guo, Y., Yin, M., Chen, X., Xiao, L., and Deng, G. (2020). Association of inflammatory markers with the severity of covid-19: a meta-analysis. *International Journal of Infectious Diseases*, 96:467–474.

Zhang, K., Peters, J., Janzing, D., and Schölkopf, B. (2012). Kernel-based conditional independence test and application in causal discovery. *arXiv preprint arXiv:1202.3775*.

Zhang, L., Yan, X., Fan, Q., Liu, H., Liu, X., Liu, Z., and Zhang, Z. (2020). D-dimer levels on admission to predict in-hospital mortality in patients with covid-19. *Journal of thrombosis and haemostasis*, 18(6):1324–1329.

Zheng, Y., Zhang, Y., Chi, H., Chen, S., Peng, M., Luo, L., Chen, L., Li, J., Shen, B., and Wang, D. (2020). The hemocyte counts as a potential biomarker for predicting disease progression in covid-19: a retrospective study. *Clinical Chemistry and Laboratory Medicine (CCLM)*, 58(7):1106–1115.

Zhong, Q. and Peng, J. (2021). Mean platelet volume/platelet count ratio predicts severe pneumonia of covid-19. *Journal of clinical laboratory analysis*, 35(1):e23607.

Zhou, F., Yu, T., Du, R., Fan, G., Liu, Y., Liu, Z., Xiang, J., Wang, Y., Song, B., Gu, X., et al. (2020a). Clinical course and risk factors for mortality of adult inpatients with covid-19 in wuhan, china: a retrospective cohort study. *The lancet*, 395(10229):1054–1062.

Zhou, Y., He, Y., Yang, H., Yu, H., Wang, T., Chen, Z., Yao, R., and Liang, Z. (2020b). Development and validation a nomogram for predicting the risk of severe covid-19: A multi-center study in sichuan, china. *PloS one*, 15(5):e0233328.

Zhu, Z., Zheng, Z., Zhang, F., Wu, Y., Trzaskowski, M., Maier, R., Robinson, M. R., McGrath, J. J., Visscher, P. M., Wray, N. R., et al. (2018). Causal associations between risk factors and common diseases inferred from gwas summary data. *Nature communications*, 9(1):1–12.