Logistic Regression
Lecture 19

David Sontag
New York University

Slides adapted from Vibhav Gogate, Luke Zettlemoyer, Carlos Guestrin, and Dan Weld
Logistic Regression

- Learn $P(Y|X)$ directly!
  - Assume a particular functional form
  - Sigmoid applied to a linear function of the data:

\[
P(Y = 1|X) = \frac{1}{1 + \exp(w_0 + \sum_{i=1}^{n} w_i X_i)}
\]

\[
P(Y = 0|X) = \frac{\exp(w_0 + \sum_{i=1}^{n} w_i X_i)}{1 + \exp(w_0 + \sum_{i=1}^{n} w_i X_i)}
\]

Features can be discrete or continuous!

Logistic function (Sigmoid):
Logistic Function in n Dimensions

\[ P(Y = 1|X) = \frac{1}{1 + \exp(w_0 + \sum_{i=1}^{n} w_i X_i)} \]

Sigmoid applied to a linear function of the data:

Features can be discrete or continuous!
Logistic Regression: decision boundary

\[ P(Y = 1|X) = \frac{1}{1 + \exp(w_0 + \sum_{i=1}^{n} w_i X_i)} \]
\[ P(Y = 0|X) = \frac{\exp(w_0 + \sum_{i=1}^{n} w_i X_i)}{1 + \exp(w_0 + \sum_{i=1}^{n} w_i X_i)} \]

- **Prediction:** Output the \( Y \) with highest \( P(Y|X) \)
  - For binary \( Y \), output \( Y = 0 \) if
    \[ 1 < \frac{P(Y = 0|X)}{P(Y = 1|X)} \]
    \[ 1 < \exp(w_0 + \sum_{i=1}^{n} w_i X_i) \]
    \[ 0 < w_0 + \sum_{i=1}^{n} w_i X_i \]

**A Linear Classifier!**
Understanding Sigmoids

\[ g(w_0 + \sum_i w_i x_i) = \frac{1}{1 + e^{w_0 + \sum_i w_i x_i}} \]

\[ w_0 = -2, \; w_1 = -1 \]

\[ w_0 = 0, \; w_1 = -1 \]

\[ w_0 = 0, \; w_1 = -0.5 \]
Likelihood vs. Conditional Likelihood

Generative (Naïve Bayes) maximizes **Data likelihood**

\[
\ln P(D \mid w) = \sum_{j=1}^{N} \ln P(x^j, y^j \mid w)
\]

\[
= \sum_{j=1}^{N} \ln P(y^j \mid x^j, w) + \sum_{j=1}^{N} \ln P(x^j \mid w)
\]

**Discriminative (Logistic Regr.)** maximizes **Conditional Data Likelihood**

\[
\ln P(D_Y \mid D_X, w) = \sum_{j=1}^{N} \ln P(y^j \mid x^j, w)
\]

Discriminative models *can’t* compute \(P(x^j \mid w)\)!

Or, ... “They don’t *waste effort* learning \(P(X)\)”

Focus only on \(P(Y \mid X)\) - all that matters for classification.
Maximizing Conditional Log Likelihood

\[ l(w) \equiv \ln \prod_j P(y^j|x^j, w) \]

\[ = \sum_j y^j (w_0 + \sum_i^n w_i x_i^j) - \ln(1 + \exp(w_0 + \sum_i^n w_i x_i^j)) \]

**Bad news:** no closed-form solution to maximize \( l(w) \)

**Good news:** \( l(w) \) is concave function of \( w \)→

No local minima

Concave functions easy to optimize
Optimizing concave function – Gradient ascent

- Conditional likelihood for Logistic Regression is concave →

**Gradient:**

$$\nabla_w l(w) = \left[ \frac{\partial l(w)}{\partial w_0}, \ldots, \frac{\partial l(w)}{\partial w_n} \right]'$$

**Update rule:**

$$\Delta w = \eta \nabla_w l(w)$$

$$w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \frac{\partial l(w)}{\partial w_i}$$

- Gradient ascent is simplest of optimization approaches
Maximize Conditional Log Likelihood: Gradient ascent

\[
P(Y = 1|X, W) = \frac{\exp(w_0 + \sum_i w_i x_i^j)}{1 + \exp(w_0 + \sum_i w_i x_i^j)}
\]

\[
l(w) = \sum_j y^j (w_0 + \sum_i w_i x_i^j) - \ln(1 + \exp(w_0 + \sum_i w_i x_i^j))
\]

\[
\frac{\partial l(w)}{\partial w_i} = \sum_j \left[ \frac{\partial}{\partial w} y^j (w_0 + \sum_i w_i x_i^j) - \frac{\partial}{\partial w} \ln \left(1 + \exp(w_0 + \sum_i w_i x_i^j)\right) \right]
\]

\[
= \sum_j \left[ y^j x_i^j - \frac{x_i^j \exp(w_0 + \sum_i w_i x_i^j)}{1 + \exp(w_0 + \sum_i w_i x_i^j)} \right]
\]

\[
= \sum_j x_i^j \left[ y^j - \frac{\exp(w_0 + \sum_i w_i x_i^j)}{1 + \exp(w_0 + \sum_i w_i x_i^j)} \right]
\]

\[
\frac{\partial l(w)}{\partial w_i} = \sum_j x_i^j \left( y^j - P(Y^j = 1|x^j, w) \right)
\]
Gradient Ascent for LR

Gradient ascent algorithm: (learning rate $\eta > 0$)

do:

$$w_0^{(t+1)} \leftarrow w_0^{(t)} + \eta \sum_j [y^j - \hat{P}(Y^j = 1 \mid x^j, w)]$$

For i=1 to n: (iterate over features)

$$w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \sum_j x_i^j [y^j - \hat{P}(Y^j = 1 \mid x^j, w)]$$

until “change” < $\varepsilon$

Loop over training examples!
Large parameters...

\[
\frac{1}{1 + e^{-ax}}
\]

- Maximum likelihood solution: prefers higher weights
  - higher likelihood of (properly classified) examples close to decision boundary
  - larger influence of corresponding features on decision
  - *can cause overfitting!!!*

- Regularization: penalize high weights
That’s all MLE. How about MAP?

\[ p(w \mid Y, X) \propto P(Y \mid X, w)p(w) \]

• One common approach is to define priors on \( w \)
  – Normal distribution, zero mean, identity covariance
  – “Pushes” parameters towards zero

\[
p(w) = \prod_{i} \frac{1}{\kappa \sqrt{2\pi}} \frac{-w_i^2}{e^{2\kappa^2}}
\]

• **Regularization**
  – Helps avoid very large weights and overfitting

• **MAP estimate:**

\[
w^* = \arg \max_w \ln \left[ p(w) \prod_{j=1}^{N} P(y_j \mid x_j, w) \right]
\]
MAP as Regularization

\[ \mathbf{w}^* = \arg \max_{\mathbf{w}} \ln \left[ p(\mathbf{w}) \prod_{j=1}^{N} P(y^j | x^j, \mathbf{w}) \right] \quad p(\mathbf{w}) = \prod_{i} \frac{1}{\kappa \sqrt{2\pi}} \frac{-w_i^2}{e^{2\kappa^2}} \]

- Add \( \log p(\mathbf{w}) \) to objective:
  
  \[ \ln p(\mathbf{w}) \propto -\frac{\lambda}{2} \sum_i w_i^2 \quad \frac{\partial \ln p(\mathbf{w})}{\partial w_i} = -\lambda w_i \]
  
  - Quadratic penalty: drives weights towards zero
  - Adds a negative linear term to the gradients

Penalizes high weights, just like we did with SVMs!
MLE vs. MAP

- Maximum conditional likelihood estimate

\[ w^* = \arg \max_w \ln \prod_{j=1}^{N} P(y^j | x^j, w) \]

\[ w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \sum_j x_i^j [y^j - \hat{P}(Y^j = 1 | x^j, w)] \]

- Maximum conditional a posteriori estimate

\[ w^* = \arg \max_w \ln \left[ p(w) \prod_{j=1}^{N} P(y^j | x^j, w) \right] \]

\[ w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \left\{ -\lambda w_i^{(t)} + \sum_j x_i^j [y^j - \hat{P}(Y^j = 1 | x^j, w)] \right\} \]
Naïve Bayes vs. Logistic Regression

**Learning:** $h: X \mapsto Y$
- $X$ – features
- $Y$ – target classes

**Generative**
- Assume functional form for
  - $P(X|Y)$ **assume cond indep**
  - $P(Y)$
  - Est params from train data
- Gaussian NB for cont features
- Bayes rule to calc. $P(Y|X= x)$
  - $P(Y|X) \propto P(X|Y) P(Y)$
- **Indirect** computation
  - Can also generate a sample of the data

**Discriminative**
- Assume functional form for
  - $P(Y|X)$ **no assumptions**
  - Est params from training data
- Handles discrete & cont features
- Directly calculate $P(Y|X=x)$
  - Can’t generate data sample
Naïve Bayes vs. Logistic Regression

[Ng & Jordan, 2002]

• Generative vs. Discriminative classifiers

• Asymptotic comparison

  (# training examples $\rightarrow$ infinity)

  – when model correct

    • NB and LDA (with class independent variances) and Logistic Regression produce identical classifiers

  – when model incorrect

    • LR is less biased – does not assume conditional independence

      – therefore LR expected to outperform NB
Naïve Bayes vs. Logistic Regression

[Ng & Jordan, 2002]

- Generative vs. Discriminative classifiers
- Non-asymptotic analysis
  - convergence rate of parameter estimates,
    \( n = \# \text{ of attributes in } X \)
  - Size of training data to get close to infinite data solution
  - Naïve Bayes needs \( O(\log n) \) samples
  - Logistic Regression needs \( O(n) \) samples

- Naïve Bayes converges more quickly to its \textit{(perhaps less helpful)} asymptotic estimates
Some experiments from UCI data sets

Figure 1: Results of 15 experiments on datasets from the UCI Machine Learning repository. Plots are of generalization error vs. $m$ (averaged over 1000 random train/test splits). Dashed line is logistic regression; solid line is naïve Bayes.
Logistic regression for discrete classification

Logistic regression in more general case, where set of possible $Y$ is $\{y_1, \ldots, y_R\}$

- Define a weight vector $w_i$ for each $y_i$, $i=1,\ldots,R-1$

$$P(Y = 1|X) \propto \exp(w_{10} + \sum_i w_{1i}X_i)$$

$$P(Y = 2|X) \propto \exp(w_{20} + \sum_i w_{2i}X_i)$$

$$\ldots$$

$$P(Y = r|X) = 1 - \sum_{j=1}^{r-1} P(Y = j|X)$$
Logistic regression for discrete classification

- Logistic regression in more general case, where $Y$ is in the set $\{y_1, \ldots, y_R\}$

for $k < R$

$$P(Y = y_k | X) = \frac{\exp(w_{k0} + \sum_{i=1}^{n} w_{ki}X_i)}{1 + \sum_{j=1}^{R-1} \exp(w_{j0} + \sum_{i=1}^{n} w_{ji}X_i)}$$

for $k = R$ (normalization, so no weights for this class)

$$P(Y = y_R | X) = \frac{1}{1 + \sum_{j=1}^{R-1} \exp(w_{j0} + \sum_{i=1}^{n} w_{ji}X_i)}$$

Features can be discrete or continuous!