Foundations of Machine Learning
Multi-Class Classification

Mehryar Mohri
Courant Institute and Google Research
mohri@cims.nyu.edu

Motivation

- Real-world problems often have multiple classes: text, speech, image, biological sequences.
- Algorithms studied so far: designed for binary classification problems.
- How do we design multi-class classification algorithms?
  - can the algorithms used for binary classification be generalized to multi-class classification?
  - can we reduce multi-class classification to binary classification?

Multi-Class Classification Problem

- Training data: sample drawn i.i.d. from set X according to some distribution D,

$$S = ((x_1, y_1), \ldots, (x_m, y_m)) \in X \times Y,$$

  - mono-label case: $\mathrm{Card}(Y) = k$.
  - multi-label case: $Y = \{-1, +1\}^k$.

- Problem: find classifier $h \colon X \to Y$ in H with small generalization error,
  - mono-label case: $R(h) = \mathrm{E}_{x \sim D}[1_{h(x) \neq f(x)}]$.
  - multi-label case: $R(h) = \mathrm{E}_{x \sim D}\left[\frac{1}{k} \sum_{l=1}^{k} 1_{[h(x)]_l \neq [f(x)]_l}\right]$.

Notes

- In most tasks considered, number of classes $k \leq 100$.
- For k large, problem often not treated as a multi-class classification problem (ranking or density estimation, e.g., automatic speech recognition).
- Computational efficiency issues arise for larger ks.
- In general, classes not balanced.

Multi-Class Classification - Margin

- Hypothesis set H:
  - functions $h\colon X \times Y \to \mathbb{R}$.
  - label returned: $x \mapsto \underset{y \in Y}{\operatorname{argmax}}\, h(x, y)$.
- Margin:
  - $\rho_h(x, y) = h(x, y) - \max_{y' \neq y} h(x, y')$.
  - error: $1_{\rho_h(x,y) \leq 0} \leq \Phi_\rho(\rho_h(x, y))$.
  - empirical margin loss:

$$\widehat{R}_\rho(h) = \frac{1}{m} \sum_{i=1}^{m} \Phi_\rho(\rho_h(x_i, y_i)).$$

Multi-Class Margin Bound

(MM et al. 2012; Kuznetsov, MM, and Syed, 2014)

- **Theorem**: let $H \subseteq \mathbb{R}^{X \times Y}$ with $Y = \{1, \ldots, k\}$. Fix $\rho > 0$. Then, for any $\delta > 0$, with probability at least $1 - \delta$, the following multi-class classification bound holds for all $h \in H$:

$$R(h) \leq \widehat{R}_\rho(h) + \frac{4k}{\rho} \mathfrak{R}_m(\Pi_1(H)) + \sqrt{\frac{\log \frac{1}{\delta}}{2m}},$$

with $\Pi_1(H) = \{x \mapsto h(x, y) \colon y \in Y, h \in H\}$.

Kernel-Based Hypotheses

- Hypothesis set $H_{K,p}$:
  - Φ feature mapping associated to PDS kernel K.
  - functions $(x, y) \mapsto \mathbf{w}_y \cdot \mathbf{\Phi}(x)$, $y \in \{1, \ldots, k\}$.
  - label returned: $x \mapsto \underset{y \in \{1,\ldots,k\}}{\operatorname{argmax}} \; \mathbf{w}_y \cdot \mathbf{\Phi}(x)$.
  - for any $p \geq 1$,

$$H_{K,p} = \{(x, y) \in X \times [1, k] \mapsto \mathbf{w}_y \cdot \mathbf{\Phi}(x) \colon \mathbf{W} = (\mathbf{w}_1, \ldots, \mathbf{w}_k)^\top, \|\mathbf{W}\|_{\mathbb{H},p} \leq \Lambda\}.$$

$$\mathfrak{R}_m(\Pi_1(\mathcal{H}_{K,p})) \leq \sqrt{\frac{r^2 \Lambda^2}{m}}.$$

Multi-Class Margin Bound - Kernels

(MM et al. 2012)

- **Theorem**: let $K\colon X \times X \to \mathbb{R}$ be a PDS kernel and let $\mathbf{\Phi}\colon X \to \mathbb{H}$ be a feature mapping associated to K. Fix $\rho > 0$. Then, for any $\delta > 0$, with probability at least $1-\delta$, the following multiclass bound holds for all $h \in H_{K,p}$:

$$R(h) \leq \widehat{R}_\rho(h) + 4k\sqrt{\frac{r^2\Lambda^2}{\rho^2 m}} + \sqrt{\frac{\log\frac{1}{\delta}}{2m}},$$

where $r^2 = \sup_{x \in X} K(x,x)$.

Approaches

- Single classifier:
  - Multi-class SVMs.
  - AdaBoost.MH.
  - Conditional Maxent.
  - Decision trees.
- Combination of binary classifiers:
  - One-vs-all.
  - One-vs-one.
  - Error-correcting codes.

Multi-Class SVMs

(Weston and Watkins, 1999; Crammer and Singer, 2001)

- Optimization problem:

$$\min_{\mathbf{w}, \boldsymbol{\xi}} \frac{1}{2} \sum_{l=1}^{k} \|\mathbf{w}_l\|^2 + C \sum_{i=1}^{m} \xi_i$$

$$\text{subject to: } \mathbf{w}_{y_i} \cdot \mathbf{x}_i + \delta_{y_i, l} \geq \mathbf{w}_l \cdot \mathbf{x}_i + 1 - \xi_i$$

$$\xi_i \geq 0, (i, l) \in [1, m] \times Y.$$

- Decision function:

$$h \colon x \mapsto \underset{l \in Y}{\operatorname{argmax}} \, (\mathbf{w}_l \cdot \mathbf{x}).$$

Notes

- Directly based on generalization bounds.
- Comparison with (Weston and Watkins, 1999): single slack variable per point, maximum of slack variables (penalty for worst class):

$$\sum_{l=1}^{k} \xi_{il} \rightarrow \max_{l=1}^{k} \xi_{il}.$$

- PDS kernel instead of inner product
- Optimization: complex constraints, mk-size problem.
  - specific solution based on decomposition into m disjoint sets of constraints (Crammer and Singer, 2001).

Dual Formulation

- Optimization problem: $\boldsymbol{\alpha}_i$ ith row of matrix $\boldsymbol{\alpha} \in \mathbb{R}^{m \times k}$.

$$\max_{\boldsymbol{\alpha}=[\alpha_{ij}]} \sum_{i=1}^{m} \boldsymbol{\alpha}_i \cdot \mathbf{e}_{y_i} - \frac{1}{2} \sum_{i=1}^{m} (\boldsymbol{\alpha}_i \cdot \boldsymbol{\alpha}_j)(\mathbf{x}_i \cdot \mathbf{x}_j)$$

$$\text{subject to: } \forall i \in [1, m], (0 \leq \alpha_{iy_i} \leq C) \wedge (\forall j \neq y_i, \alpha_{ij} \leq 0) \wedge (\boldsymbol{\alpha}_i \cdot \mathbf{1} = 0).$$

- Decision function:

$$h(x) = \operatorname*{argmax}_{l \in [1,k]} \Big(\sum_{i=1}^{m} \alpha_{il} (\mathbf{x}_i \cdot \mathbf{x}) \Big).$$

AdaBoost

(Schapire and Singer, 2000)

- Training data (multi-label case):

$$(x_1, y_1), \ldots, (x_m, y_m) \in X \times \{-1, 1\}^k.$$

- Reduction to binary classification:
  - each example leads to k binary examples:

$$(x_i, y_i) \to ((x_i, 1), y_i[1]), \ldots, ((x_i, k), y_i[k]), i \in [1, m].$$

  - apply AdaBoost to the resulting problem.
  - choice of α_t.

- Computational cost: mk distribution updates at each round.

AdaBoost.MH

$H \subseteq (\{-1,+1\}^k)^{(X \times Y)}$.

$\textsc{AdaBoost.MH}(S = ((x_1, y_1), \ldots, (x_m, y_m)))$

for $i \leftarrow 1$ **to** m **do**
  for $l \leftarrow 1$ **to** k **do**
    $D_1(i, l) \leftarrow \frac{1}{mk}$
for $t \leftarrow 1$ **to** T **do**
  $h_t \leftarrow$ base classifier in H with small error $\epsilon_t = \Pr_{D_t}[h_t(x_i, l) \neq y_i[l]]$
  $\alpha_t \leftarrow$ choose $\quad \triangleright$ to minimize Z_t
  $Z_t \leftarrow \sum_{i,l} D_t(i, l) \exp(-\alpha_t y_i[l] h_t(x_i, l))$
  for $i \leftarrow 1$ **to** m **do**
    for $l \leftarrow 1$ **to** k **do**
      $D_{t+1}(i, l) \leftarrow \frac{D_t(i,l) \exp(-\alpha_t y_i[l] h_t(x_i,l))}{Z_t}$
$f_T \leftarrow \sum_{t=1}^{T} \alpha_t h_t$
return $h_T = \operatorname{sgn}(f_T)$

Bound on Empirical Error

- **Theorem**: The empirical error of the classifier output by AdaBoost.MH verifies:

$$\widehat{R}(h) \leq \prod_{t=1}^{T} Z_t.$$

- **Proof**: similar to the proof for AdaBoost.
- **Choice of α_t**:
  - for $H \subseteq (\{-1, +1\}^k)^{X \times Y}$, as for AdaBoost, $\alpha_t = \frac{1}{2} \log \frac{1-\epsilon_t}{\epsilon_t}$.
  - for $H \subseteq ([-1, 1]^k)^{X \times Y}$, same choice: minimize upper bound.
  - other cases: numerical/approximation method.

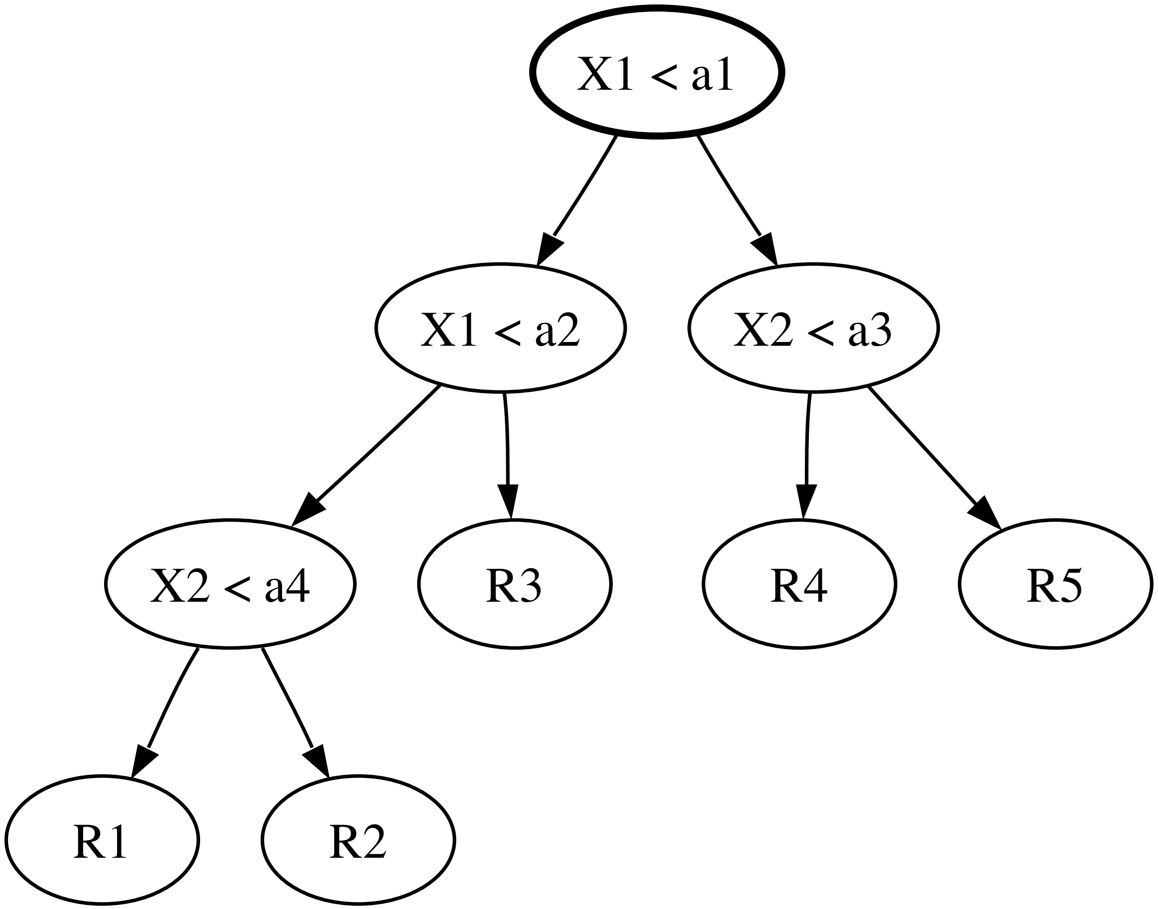
Notes

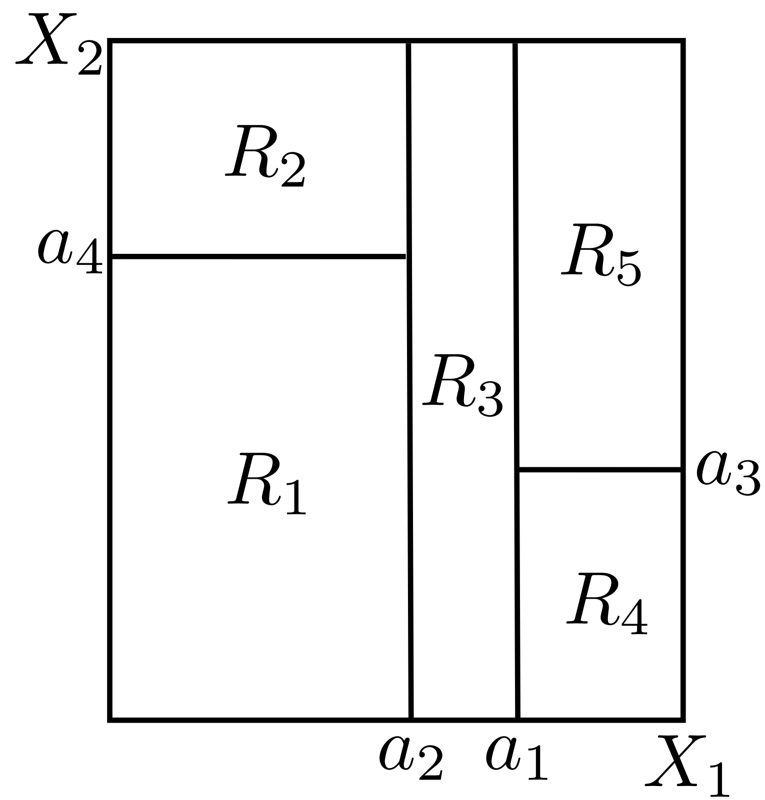
- Objective function:

$$F(\boldsymbol{\alpha}) = \sum_{i=1}^{m} \sum_{l=1}^{k} e^{-y_i[l] f_n(x_i, l)} = \sum_{i=1}^{m} \sum_{l=1}^{k} e^{-y_i[l] \sum_{t=1}^{n} \alpha_t h_t(x_i, l)}.$$

- All comments and analysis given for AdaBoost apply here.
- Alternative: Adaboost.MR, which coincides with a special case of RankBoost (ranking lecture).

Decision Trees





Different Types of Questions

- Decision trees
  - $X \in \{\text{blue}, \text{white}, \text{red}\}$: categorical questions.
  - $X \leq a$: continuous variables.
- Binary space partition (BSP) trees:
  - $\sum_{i=1}^{n} \alpha_i X_i \leq a$: partitioning with convex polyhedral regions.
- Sphere trees:
  - $||X - a_0|| \leq a$: partitioning with pieces of spheres.

Hypotheses

- In each region R_t,
  - classification: majority vote - ties broken arbitrarily,

$$\widehat{y}_t = \operatorname*{argmax}_{y \in Y} \left|\{x_i \in R_t \colon i \in [1, m], y_i = y\}\right|.$$

  - regression: average value,

$$\widehat{y}_t = \frac{1}{|S \cap R_t|} \sum_{\substack{x_i \in R_t \\ i \in [1, m]}} y_i.$$

- Form of hypotheses:

$$h \colon x \mapsto \sum_t \widehat{y}_t 1_{x \in R_t}.$$

Training

- Problem: general problem of determining partition with minimum empirical error is NP-hard.
- Heuristics: greedy algorithm.
  - for all $j \in [1, N]$, $\theta \in \mathbb{R}$, $R^+(j,\theta) = \{x_i \in R \colon x_i[j] \geq \theta, i \in [1,m]\}$
    $R^-(j,\theta) = \{x_i \in R \colon x_i[j] < \theta, i \in [1,m]\}$.

$\textsc{Decision-Trees}(S = ((x_1, y_1), \ldots, (x_m, y_m)))$

$P \leftarrow \{S\}$ $\triangleright$ initial partition
for each region $R \in P$ such that $\text{Pred}(R)$ **do**
    $(j, \theta) \leftarrow \text{argmin}_{(j,\theta)}\, \text{error}(R^-(j,\theta)) + \text{error}(R^+(j,\theta))$
    $P \leftarrow P - R \cup \{R^-(j,\theta), R^+(j,\theta)\}$
return P

Splitting/Stopping Criteria

- Problem: larger trees overfit training sample.
- Conservative splitting:
  - split node only if loss reduced by some fixed value $\eta > 0$.
  - issue: seemingly bad split dominating useful splits.
- Grow-then-prune technique (CART):
  - grow very large tree, $\text{Pred}(R)$: $|R| > |n_0|$.
  - prune tree based on: $F(T) = \widehat{\text{Loss}}(T) + \alpha |T|$, $\alpha \geq 0$ parameter determined by cross-validation.

Decision Tree Tools

- Most commonly used tools for learning decision trees:
  - CART (classification and regression tree) (Breiman et al., 1984).
  - C4.5 (Quinlan, 1986, 1993) and C5.0 (RuleQuest Research) a commercial system.
- Differences: minor between latest versions.

Approaches

- Single classifier:
  - SVM-type algorithm.
  - AdaBoost-type algorithm.
  - Conditional Maxent.
  - Decision trees.
- Combination of binary classifiers:
  - One-vs-all.
  - One-vs-one.
  - Error-correcting codes.

One-vs-All

- Technique:
  - for each class $l \in Y$ learn binary classifier $h_l = \text{sgn}(f_l)$.
  - combine binary classifiers via voting mechanism, typically majority vote: $h \colon x \mapsto \operatorname*{argmax}_{l \in Y} f_l(x)$.
- Problem: poor justification (in general).
  - calibration: classifier scores not comparable.
  - nevertheless: simple and frequently used in practice, computational advantages in some cases.

One-vs-One

- **Technique**:
  - for each pair $(l, l') \in Y, l \neq l'$ learn binary classifier $h_{ll'} : X \to \{0, 1\}$.
  - combine binary classifiers via majority vote:

$$h(x) = \operatorname*{argmax}_{l' \in Y} \Big| \{l : h_{ll'}(x) = 1\} \Big|.$$

- **Problem**:
  - computational: train $k(k-1)/2$ binary classifiers.
  - overfitting: size of training sample could become small for a given pair.

Computational Comparison

	Training	Testing
One-vs-all	$O(kB_{\text{train}}(m))$ $O(km^{\alpha})$	$O(kB_{\text{test}})$
One-vs-one	$O(k^2 B_{\text{train}}(m/k))$ (on average) $O(k^{2-\alpha} m^{\alpha})$	$O(k^2 B_{\text{test}})$ *smaller* N_{SV} *per* B

Time complexity for SVMs, α less than 3.

Error-Correcting Code Approach

(Dietterich and Bakiri, 1995)

- Idea:
  - assign F-long binary code word to each class:

    $$\longrightarrow \mathbf{M} = [\mathbf{M}_{lj}] \in \{0,1\}^{[1,k]\times[1,F]}.$$

  - learn binary classifier $f_j \colon X \to \{0,1\}$ for each column. Example x in class l labeled with M_{lj}.
  - classifier output: $\Big(\mathbf{f}(x) = \big(f_1(x), \ldots, f_F(x)\big)\Big)$,

    $$h \colon x \mapsto \operatorname*{argmin}_{l \in Y} d_{\text{Hamming}}\Big(\mathbf{M}_l, \mathbf{f}(x)\Big).$$

Illustration

- 8 classes, code-length: 6.

codes

classes

	1	2	3	4	5	6
1	0	0	0	1	0	0
2	1	0	0	0	0	0
3	0	1	1	0	1	0
4	1	1	0	0	0	0
5	1	1	0	0	1	0
6	0	0	1	1	0	1
7	0	0	1	0	0	0
8	0	1	0	1	0	0

$f_1(x)$	$f_2(x)$	$f_3(x)$	$f_4(x)$	$f_5(x)$	$f_6(x)$
0	1	1	0	1	1

new example x

Error-Correcting Codes - Design

- Main ideas:
  - independent columns: otherwise no effective discrimination.
  - distance between rows: if the minimal Hamming distance between rows is d, then the multi-class can correct $\lfloor \frac{d-1}{2} \rfloor$ (classification) errors.
  - columns may correspond to features selected for the task.
  - one-vs-all and one-vs-one (with ternary codes) are special cases.

Extensions

(Allwein et al., 2000)

- Matrix entries in $\{-1, 0, +1\}$:
  - examples marked with 0 disregarded during training.
  - $\longrightarrow$ one-vs-one becomes also a special case.
- Margin loss L: function of $yf(x)$, e.g., hinge loss.
  - Hamming loss:

$$h(x) = \operatorname*{argmin}_{l \in \{1,\ldots,k\}} \sum_{j=1}^{F} \frac{1 - \operatorname{sgn}\big(\mathbf{M}_{lj} f_j(x)\big)}{2}.$$

  - Margin loss:

$$h(x) = \operatorname*{argmin}_{l \in \{1,\ldots,k\}} \sum_{j=1}^{F} L\big(\mathbf{M}_{lj} f_j(x)\big).$$

Applications

- One-vs-all approach is the most widely used combination method.
- No clear empirical evidence of the superiority of other approaches (Rifkin and Klautau, 2004).
  - except perhaps on small data sets with relatively large error rate.
- Large structured multi-class problems: often treated as ranking problems (see ranking lecture).

References

- Erin L. Allwein, Robert E. Schapire and Yoram Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of Machine Learning Research*, 1:113-141, 2000.

- K. Crammer and Y. Singer. Improved output coding for classification using continuous relaxation. In Proceedings of *NIPS*, 2000.

- Koby Crammer and Yoram Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2:265–292, 2001.

- Koby Crammer and Yoram Singer. On the Learnability and Design of Output Codes for Multiclass Problems. *Machine Learning 47*, 2002.

- Thomas G. Dietterich, Ghulum Bakiri: Solving Multiclass Learning Problems via Error-Correcting Output Codes. *Journal of Artificial Intelligence Research (JAIR) 2*: 263-286, 1995.

- Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. Foundations of Machine Learning, the MIT Press, 2012.

- John C. Platt, Nello Cristianini, and John Shawe-Taylor. Large Margin DAGS for Multiclass Classification. In *Advances in Neural Information Processing Systems 12 (NIPS 1999)*, pp. 547-553, 2000.

References

- Ryan Rifkin. "Everything Old Is New Again: A Fresh Look at Historical Approaches in Machine Learning." Ph.D. Thesis, MIT, 2002.
- Rifkin and Klautau. "In Defense of One-Vs-All Classification." *Journal of Machine Learning Research*, 5:101-141, 2004.
- Robert E. Schapire. The boosting approach to machine learning: An overview. In D. D. Denison, M. H. Hansen, C. Holmes, B. Mallick, B. Yu, editors, *Nonlinear Estimation and Classification*. Springer, 2003.
- Robert E. Schapire, Yoav Freund, Peter Bartlett and Wee Sun Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *The Annals of Statistics*, 26(5):1651-1686, 1998.
- Robert E. Schapire and Yoram Singer. BoosTexter: A boosting-based system for text categorization. *Machine Learning*, 39(2/3):135-168, 2000.
- Jason Weston and Chris Watkins. Support Vector Machines for Multi-Class Pattern Recognition. *Proceedings of the Seventh European Symposium On Artificial Neural Networks (ESANN '99)*, 1999.