Foundations of Machine Learning Multi-Class Classification

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

Notes

- In most tasks considered, number of classes $k \leq 100$.
- For k large, problem often not treated as a multiclass 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 $h: X \times Y \to \mathbb{R}$.
- 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: $1_{\rho_h(x,y) \le 0} \le \Phi_{\rho}(\rho_h(x,y))$.
- empirical margin loss:

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

Multi-Class Margin Bound

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

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

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

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 $(x, y) \mapsto \mathbf{w}_y \cdot \mathbf{\Phi}(x), y \in \{1, \dots, k\}.$
- label returned: $x \mapsto \underset{y \in \{1,...,k\}}{\operatorname{argmax}} \mathbf{w}_y \cdot \mathbf{\Phi}(x)$.
- for any $p \ge 1$,

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

Multi-Class Margin Bound - Kernels

(MM et al. 2012)

Theorem: let $K: X \times X \to \mathbb{R}$ be a PDS kernel and let $\Phi: X \to \mathbb{H}$ 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 $h \in H_{K,p}$:

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

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:

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

subject to: $\mathbf{w}_{y_{i}} \cdot \mathbf{x}_{i} + \delta_{y_{i},l} \ge \mathbf{w}_{l} \cdot \mathbf{x}_{i} + 1 - \xi_{i}$
 $(i,l) \in [1,m] \times Y.$

Decision function:

$$h: x \mapsto \operatorname*{argmax}_{l \in Y} (\mathbf{w}_l \cdot \mathbf{x}).$$

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):

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

- 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: α_i *i*th row of matrix $\alpha \in \mathbb{R}^{m \times k}$.

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

subject to: $\forall i \in [1, m], (0 \le \alpha_{iy_{i}} \le C) \land (\forall j \ne y_{i}, \alpha_{ij} \le 0) \land (\boldsymbol{\alpha}_{i} \cdot \mathbf{1} = 0).$

Decision function:

$$h(x) = \underset{l=1}{\operatorname{argmax}} \left(\sum_{i=1}^{m} \alpha_{il} (\mathbf{x}_i \cdot \mathbf{x}) \right).$$

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: $(x_i, y_i) \rightarrow ((x_i, 1), y_i[1]), \dots, ((x_i, k), y_i[k]), i \in [1, m].$
- apply AdaBoost to the resulting problem.
- choice of α_t .
- Computational cost: mk distribution updates at each round.

AdaBoost.MH

 $H \subseteq (\{-1, +1\}^k)^{(X \times Y)}.$ ADABOOST.MH $(S = ((x_1, y_1), \dots, (x_m, y_m)))$ for $i \leftarrow 1$ to m do 1 2for $l \leftarrow 1$ to k do 3 $D_1(i,l) \leftarrow \frac{1}{mk}$ for $t \leftarrow 1$ to T do 4 $h_t \leftarrow \text{base classifier in } H \text{ with small error } \epsilon_t = \Pr_{D_t}[h_t(x_i, l) \neq y_i[l]]$ 5 $\alpha_t \leftarrow \text{choose} \quad \triangleright \text{ to minimize } Z_t$ 6 7 $Z_t \leftarrow \sum_{i,l} D_t(i,l) \exp(-\alpha_t y_i[l] h_t(x_i,l))$ 8 for $i \leftarrow 1$ to m do 9 for $l \leftarrow 1$ to k do $D_{t+1}(i,l) \leftarrow \frac{D_t(i,l)\exp(-\alpha_t y_i[l]h_t(x_i,l))}{Z_t}$ 10 11 $f_T \leftarrow \sum_{t=1}^T \alpha_t h_t$ return $h_T = \operatorname{sgn}(f_T)$ 12

Bound on Empirical Error

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

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

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*⊆([−1,1]^k)^{X×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

- $X \in \{$ blue, white, 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:

• $||X - a_0|| \le a$: partitioning with pieces of spheres.

Hypotheses

In each region R_t ,

 classification: majority vote - ties broken arbitrarily,

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

• 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:

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

Training

Problem: general problem of determining partition with minimum empirical error is NP-hard.

Heuristics: greedy algorithm.

• for all $j \in [1, N]$, $\theta \in \mathbb{R}$, $R^+(j, \theta) = \{x_i \in R : x_i[j] \ge \theta, i \in [1, m]\}$ $R^-(j, \theta) = \{x_i \in R : x_i[j] < \theta, i \in [1, m]\}.$

DECISION-TREES($S = ((x_1, y_1), \ldots, (x_m, y_m)))$

- 1 $P \leftarrow \{S\} \triangleright \text{initial partition}$
- 2 for each region $R \in P$ such that Pred(R) do
- 3 $(j,\theta) \leftarrow \operatorname{argmin}_{(j,\theta)} \operatorname{error}(R^{-}(j,\theta)) + \operatorname{error}(R^{+}(j,\theta))$ 4 $P \leftarrow P - R \cup \{R^{-}(j,\theta), R^{+}(j,\theta)\}$
- 5 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, $\operatorname{Pred}(R)$: $|R| > |n_0|$.
 - prune tree based on: $F(T) = \widehat{Loss}(T) + \alpha |T|$, $\alpha \ge 0$ 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 $h_l = \operatorname{sgn}(f_l)$.
- combine binary classifiers via voting mechanism, typically majority vote: $h: x \mapsto \underset{l \in Y}{\operatorname{argmax}} f_l(x).$
- 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 $h_{ll'}: X \rightarrow \{0, 1\}$.
- combine binary classifiers via majority vote: $h(x) = \underset{l' \in Y}{\operatorname{argmax}} |\{l : h_{ll'}(x) = 1\}|.$

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	$O(kB_{ ext{train}}(m))$ $O(km^{lpha})$	$O(kB_{\text{test}})$
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 $f_j: X \rightarrow \{0, 1\}$ for each column. Example x in class l labeled with M_{lj} .
- classifier output: $(\mathbf{f}(x) = (f_1(x), \dots, f_F(x))),$ $h: x \mapsto \operatorname*{argmin}_{l \in Y} d_{\operatorname{Hamming}} (\mathbf{M}_l, \mathbf{f}(x)).$

Illustration

8 classes, code-length: 6.





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 $\lfloor \frac{d-1}{2} \rfloor$ 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.
 - Margin loss: Note: Margin loss: Note: Margin loss: Note: $h(x) = \operatorname*{argmin}_{l \in \{1,...,k\}} \sum_{j=1} L(\mathbf{M}_{lj}f_j(x)).$

Applications

- One-vs-all approach is the most widely used.
- 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 kernelbased 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.