Multi-Class Learnabitlity

Daniel Khashabi

Summer 2016
Last Update: September 28, 2016

1 Introduction

Let $\mathcal{X}$ denote the input space, and $\mathcal{Y}$ the output space. Let $D_{\mathcal{X}}$ denote the unknown distribution over the input space. The multi-class problem is usually modelled with two tastes:

- Mono-labels: or $\mathcal{Y} = \{1, 2, 3, \dots, k\}$.
- Binary-labels, or $\mathcal{Y} = \{\pm 1\}^k$

The difference between the two tastes is mostly mathematical convenience, and sometiems issues related to representability (see Section??). Suppose samples are i.i.d. samples from the unknown underlying distributiom:

$$S = \{(x_i, y_i)\}_{i=1}^m \in (\mathcal{X} \times \mathcal{Y})^m$$

The goal is to learn the hypothesis function $h \in \mathcal{H}$ whcih minimizes the empirical risk:

$$R(h) = \frac{1}{m} \sum_{1}^{m} \ell \left(h(x_i), y_i \right)$$

Here are the main challenges/concerns when dealing with such problems:

- Output representation: what is the effect of output representation/encoding in generalization/learnability
- The connection between output classes, or how the outputs are related/correaleted [*]
- Size of the output space: how to efficiently learn when the size of the output space k grows fast
- Imbalance datasets: how to handle the cases when certain classes have much more learning instances that others

First we introduce a couple of important multi-class families. Thereafter we provide some theoretical gurantees for them.

[*]This really can be the subset of the previous item under "representation", but I thought it is so broad that it can have its own item.

2 Mutli-class algorithms

We briefly go over importatn multi-class paradigms.

2.1 Multi-class SVM

A simple extension of the ideas for the standard (soft) SVM [†] is as follows: suppose we have weight vector per dimention $\mathbf{w}_l$, and the hypothesis space is defind as [‡]

$$\mathcal{H} = \left\{ (\mathbf{x}, y) \to \mathbf{w}_y.\Phi(\mathbf{x}) : \mathbf{W} = (\mathbf{w}_1, \ldots, \mathbf{w}_k)^\top, \sum_{l=1}^{k} \|\mathbf{w}_l\|^2 \leq \Lambda^2 \right\}$$

and given a $h \in \mathcal{H}$, the decision function is of the form $\mathbf{x} \to \arg\max_{l \in \mathcal{Y}} \mathbf{w}_l.\Phi(\mathbf{x})$. Here is the optimization problems:

$$\begin{aligned} &\min_{\mathbf{W},\xi} \left\{ \frac{1}{2} \sum_{l=1}^{k} \|\mathbf{w}_l\|^2 + C \sum_{i=1}^{n} \xi_i \right\} \\ &\quad \mathbf{w}_{y_i}.\Phi(\mathbf{x}_i) \geq \mathbf{w}_l.\Phi(\mathbf{x}_i) + 1 - \xi_i, \quad \forall i = 1, .., n, \forall l \in \mathcal{Y} \setminus \{y_i\} \\ &\quad \xi_i \geq 0, \quad \forall i = 1, .., n \end{aligned}$$

Just like the standard binary soft-SVM, this is a (quadaric) convex function. Applying Lagrangian to find the dual program gives us the following oprimization problem:

$$\begin{cases} \max_{\boldsymbol{\alpha} \in \mathbb{R}^{m \times k}} \sum_i \boldsymbol{\alpha}_i.\mathbf{e}_i - \frac{1}{2} \sum_{i,j} \boldsymbol{\alpha}_i \boldsymbol{\alpha}_i K(\mathbf{x}_i, \mathbf{x}_j) \\ \text{subject to:} \\ \qquad \boldsymbol{\alpha}_i.\mathbf{1} = 0, \forall i \in [n] \\ \qquad \alpha_{ij} \leq 0, \forall i \in [n], \forall j \neq y_i \\ \qquad 0 \leq \alpha_{iy_i} \leq C, \forall i \in [n] \end{cases} \tag{1}$$

with $\mathbf{e}_i$ being a one hot vector, with the l-th element set to one.

Something to notice: the size of both primal/dual are $\Omega(kn)$, as expected. There exists variations of this model, which try to decrease this dependence to sublinear k (large number of classes).

2.2 Mutli-class boosting

Ideas from boosting [§] can be used for multi-class classification. Here we show only a few ways of doing this.

2.2.1 AdaBoost.MH algorithm

This is a simple extension of vanila AdaBoost. Each weak learner $g(x_i, l)$ gives a score per-instance per-labels, and the ultimate goal is to construct a strong learner $G_T(x, l) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t g_t(x, l)\right)$. Given $G_T(x, l)$ and input instance x, the output would the class with highest confidence score.

[†] http://web.engr.illinois.edu/~khashab2/learn/svm.pdf
[‡] We don't write any bias terms in the formulation; it is implciteIy subsumed in the feature vectors.
[§] http://web.engr.illinois.edu/~khashab2/learn/genlin.pdf

Following the ideas from boosting, the global objective function is written as an exponential loss function:

$$L(\boldsymbol{\alpha}) = \frac{1}{nk} \sum_{i=1}^{n} \sum_{l=1}^{k} e^{-y_i[l] \sum_{t=1}^{T} \alpha_t g_t(x_i, l)}$$

where $y[l] = 1$ iff the output label of instance i is l. Note that this is an upperbound on the empirical risk, since $I_{[u \le 0]} \le e^{-u}, \forall u \in \mathbb{R}$.

$$\hat{R}(h) = \frac{1}{nk} \sum_{i=1}^{n} \sum_{l=1}^{k} I_{\left[y_i[l] \sum_{t=1}^{T} \alpha_t g_t(x_i, l) \le 0\right]}$$

Algorithm 1: Adaboost.MH algorithm

```
Input: The set of weak learners, $g_t(x,l), \forall t \in [T], l \in [k]$
Output: The weights of the generalized learner $\{\alpha_t\}_{t=1}^{T}$
Initialize the weights, $w_{i,l}^{(1)} = \frac{1}{nk}$, $i \in [n], l \in [k]$
for $t = 1$ to $T$ do
    Fit the weak learner $g_t(x,l)$, with small error $\epsilon_t = \mathrm{P}_{(i,l)\sim D_{\mathcal{X}}}\left(g_t(x,l) \neq y_i[l]\right)$
    $\alpha_t = \frac{1}{2} \log \frac{1-\epsilon_t}{\epsilon_t}$.
    $Z_t = 2\sqrt{\epsilon_t(1-\epsilon_t)}$
    $w_{i,l}^{(t+1)} \leftarrow w_{i,l}^{(t)} \frac{\exp[-\alpha_t y_i g_t(x_i)]}{Z_t}$, $\forall i \in [n], l \in [k]$.
Return the output $G(x,l)$.
```

2.3 Decision-tree

2.4 Neural Networks

2.5 Structured Prediction

3 Margin-based Generalization Bound

Define the the hypothesis space $\mathcal{H}$ to be all h such that $\mathcal{X} \times \mathcal{Y} \to \mathbb{R}$. Given an input $x \in \mathcal{X}$ the output mapping is defined as

$$x \to \arg\max_{y \in \mathcal{Y}} h(x,y)$$

Define the margin to be

$$\rho_h(x,y) \triangleq h(x,y) - \max_{y' \neq y} h(x,y')$$

Define the margin loss $\ell_\rho(z)$ with fixed parameter ρ to be the following:

$$\ell_\rho(z) = \begin{cases} 0 & \text{if } \rho < z \\ 1 - x/\rho & \text{if } 0 \le z \le \rho \\ 1 & \text{if } z \le 0 \end{cases}$$

It should be evident, when margin is positive, the loss $\ell_\rho(\rho_h(x, y))$ is smaller (and sometimes it is zero). With loss defined, we can define the empirical risk:

$$\hat{R}(h) = \frac{1}{m} \sum_{i=1}^{m} \ell_\rho(\rho_h(x, y))$$

3.1 Generalization bound with Rademacher complexities

We start off by characterizing the rademacher comlpeity of the "max" function.

Lemma 3.1. *If $Hy_1, ..., \mathcal{H}_k$ are k hypothesis spaces, and $\mathcal{G} = \{\max\{h_1, ..., h_k\} : h_i \in \mathcal{H}_i\}$ be the combined hypothesis. For any sample S of size m, the empirical Rademacher complexity of $\mathcal{G}$* [¶] *can be written as follows:*

$$\hat{\mathfrak{R}}_S(\mathcal{G}) \leq \sum_i \hat{\mathfrak{R}}_S(\mathcal{H}_i)$$

Proof. TODO ■

We first present the main thereom which contains the margin-based generalization bound.

Theorem 3.2. *For any $\rho > 0$ (note strictly bigger than zero), and any $\delta > 0$, with probably at least $1 - \delta$, the following generalization bound holds for any $h \in \mathcal{H}$:*

$$R(h) \leq \hat{R}(h) + \frac{2k^2}{\rho} 2\mathfrak{R}_m(\mathcal{H}) + \sqrt{\frac{\log \frac{1}{\delta}}{2m}}$$

where $\mathcal{H}$ is hypothesis used per class and prediction is made using max rule, i.e. $y = \max_i h_i(x), h_i \in \mathcal{H}$.

Before we jump into the proof, note that this bound holds for a fixed margin parameter ρ. One can generalize this so that it holds for any choice of ρ, at the cost of an extra term $\sqrt{\frac{\log log 2/\rho}{n}}$. This term gets smaller as the desired margin gets bigger, while the empirical risk goes higher.

Proof. We use the general thereom whcih gives generalization bounds using any definition of the Rademacher average (See Theorem 3.2, [¶]).

The space of functional fed to the Rademacher function is $\ell_\rho \circ \mathcal{K}$, where ℓ_ρ is a $1/\rho$-Lipchitz loss, $\mathcal{K} = \{z = (x, y) \to \rho_h(x, y) : h\}$. Using the contraction lemma we have,

$$R(h) \leq \hat{R}(h) + \frac{2k^2}{\rho} \mathfrak{R}_m(\mathcal{K}) + \sqrt{\frac{\log 1/\delta}{2n}}, \text{ with probability at least } 1 - \delta$$

[¶] The definitions can be found here: http://web.engr.illinois.edu/~khashab2/learn/rademacher.pdf

Next we simplify $\mathfrak{R}_m(\mathcal{K})$:

$$\begin{aligned}
\mathfrak{R}_m(\mathcal{K}) &= \frac{1}{n}\mathbb{E}_{S,\sigma}\left[\sup_{h\in\mathcal{H}}\sum_{i=1}^{n}\sigma_i\rho_h(x_iy_i)\right] \\
&= \frac{1}{n}\mathbb{E}_{S,\sigma}\left[\sup_{h\in\mathcal{H}}\sum_{i=1}^{n}\sum_{y\in\mathcal{Y}}\sigma_i\rho_h(x_iy)I_{[y=y_i]}\right] \\
&\leq \frac{1}{n}\sum_{y\in\mathcal{Y}}\mathbb{E}_{S,\sigma}\left[\sup_{h\in\mathcal{H}}\sum_{i=1}^{n}\sigma_i\rho_h(x_iy)I_{[y=y_i]}\right] \\
&\leq \frac{1}{n}\sum_{y\in\mathcal{Y}}\mathbb{E}_{S,\sigma}\left[\sup_{h\in\mathcal{H}}\sum_{i=1}^{n}\sigma_i\rho_h(x_iy)\left(\frac{2I_{[y=y_i]}-1}{2}+\frac{1}{2}\right)\right]
\end{aligned}$$

Define $\sigma_i' = 2I_{[y=y_i]} - 1$, and using the fact that suppremum of summation is smaller than summation of suppremums we have:

$$\mathfrak{R}_m(\mathcal{K}) \leq \frac{1}{2n}\sum_{y\in\mathcal{Y}}\mathbb{E}_{S,\sigma}\left[\sup_{h\in\mathcal{H}}\sum_{i=1}^{n}\sigma_i\sigma_i'\rho_h(x_iy)\right] + \frac{1}{2n}\sum_{y\in\mathcal{Y}}\mathbb{E}_{S,\sigma}\left[\sup_{h\in\mathcal{H}}\sum_{i=1}^{n}\sigma_i\rho_h(x_iy)\right]$$

Since the distribution of $\sigma_i\sigma_i'$ and σ_i are the same, we have:

$$\mathfrak{R}_m(\mathcal{K}) \leq \frac{1}{n}\sum_{y\in\mathcal{Y}}\mathbb{E}_{S,\sigma}\left[\sup_{h\in\mathcal{H}}\sum_{i=1}^{n}\sigma_i\rho_h(x_iy)\right]$$

Next we wil place in the defintion of $\rho_h(x_i, y)$:

$$\begin{aligned}
\mathfrak{R}_m(\mathcal{K}) &\leq \frac{1}{n}\sum_{y\in\mathcal{Y}}\mathbb{E}_{S,\sigma}\left[\sup_{h\in\mathcal{H}}\sum_{i=1}^{n}\sigma_i\left(h(x_i,y) - \max_{y'\neq y}h(x_i,y')\right)\right] \\
&\leq \frac{1}{n}\sum_{y\in\mathcal{Y}}\mathbb{E}_{S,\sigma}\left[\sup_{h\in\mathcal{H}}\sum_{i=1}^{n}\sigma_i h(x_i,y)\right] + \frac{1}{n}\sum_{y\in\mathcal{Y}}\mathbb{E}_{S,\sigma}\left[\sup_{h\in\mathcal{H}}\sum_{i=1}^{n}\sigma_i\max_{y'\neq y}h(x_i;y')\right]
\end{aligned}$$

We distributed the suppremum on the summation again; also we used the fact that $-\sigma_i$ has the same distribution as σ_i.

$$\begin{aligned}
\mathfrak{R}_m(\mathcal{K}) &\leq \frac{1}{n}\sum_{y\in\mathcal{Y}}\mathbb{E}_{S,\sigma}\left[\sup_{h\in\mathcal{H}}\sum_{i=1}^{n}\sigma_i h(x_i,y)\right] + \frac{1}{n}\sum_{y\in\mathcal{Y}}\mathbb{E}_{S,\sigma}\left[\sup_{g\in\mathcal{G}_{k-1}}\sum_{i=1}^{n}\sigma_i g(x_i)\right] \\
&\leq k\left(k\mathfrak{R}_m(\mathcal{H}) + (k-1)\mathfrak{R}_m(\mathcal{H})\right) = k^2\mathfrak{R}_m(\mathcal{H})
\end{aligned}$$

where $\mathcal{G}_k = \{\max\{h_1, ..., h_k\} : h_i \in \mathcal{H}_i\}$ ■

Remark 3.3. In SVMs, the empirical risk $R(\hat{h})$ is directly calculated within optimization problem. More specifically it can be found via $\frac{1}{n}\sum_{i=1}^{n}\xi_i$. For multi-class problem, (section 2.1) this equivalent to $\xi_i = \max\left(1 - \left\{\mathbf{w}_{y_i}.\Phi(\mathbf{x}_i) - \max_{y' \neq y_i} \mathbf{w}_{y'}.\Phi(\mathbf{x}_i)\right\}, 0\right)$.

4 Natarajan Dimension

We introduces an extesion of the VC-dimension for studying multiple output learning. Just like VC-dimension it is a measure of how complex is the hypothesis space (or how many parameters does it have).

Definition 4.1 (Shattering for mulitclass classification). *We say a set $C \subset X$ is shattered by $\mathcal{H}$ if there exists two functions $f_0, f_1 \in \mathcal{H}$ such that*

- $\forall x \in C, f_0(x) \neq f_1(x)$
- *For any subset $B \subset C$ there exists a function $h \in \mathcal{H}$ such that*

$$\forall x \in B, h(x) = f_0(x) \text{ and } \forall x \in C \setminus B, h(x) = f_1(x)$$

Definition 4.2 (Natarajan dimension). *The Natarajan dimension of $\mathcal{H}$, Ndim($\mathcal{H}$) is the maximal size of $C \subset X$.*

Theorem 4.1 (Fundamental Theorem of Multi-class Classification). *For every hypothesis class $\mathcal{H}$ and functions from X to $[k]$ with Natarajan dimension of d, there exist constants $C_1, C_2 > 0$ such that the followings hold:*

- *$\mathcal{H}$ has uniform convergence with sample complexity obounded in:*

$$C_1 \frac{d + \log\frac{1}{\delta}}{\epsilon^2} \leq m_{\mathcal{H}}^{UC}(\epsilon, \delta) \leq C_2 \frac{d\log(k) + \log\frac{1}{\delta}}{\epsilon^2}$$

- *$\mathcal{H}$ is agnostic PAC learnable with sample complexity obounded in:*

$$C_1 \frac{d + \log\frac{1}{\delta}}{\epsilon^2} \leq m_{\mathcal{H}}(\epsilon, \delta) \leq C_2 \frac{d\log(k) + \log\frac{1}{\delta}}{\epsilon^2}$$

- *$\mathcal{H}$ is PAC learnable (assuming realizablity) with sample complexity:*

$$C_1 \frac{d + \log\frac{1}{\delta}}{\epsilon} \leq m_{\mathcal{H}}(\epsilon, \delta) \leq C_2 \frac{d\log\left(\frac{kd}{\epsilon}\right) + \log\frac{1}{\delta}}{\epsilon^2}$$

Lemma 4.2 (Natarajan).

$$|\mathcal{H}| \leq |\mathcal{X}|^{nDim(\mathcal{H})} . k^{2nDim(\mathcal{H})}$$

4.1 Calculating Natarajan dimension

One-vs-all.

Multiclass-to-binary.

Linear multi-class.

Open problems TODO

5 Structured Prediction

Denote the feature space corresponding to a pair $(x, y) \in \mathcal{X} \times \mathcal{Y}$, with $\Phi(x, y)$. A popular hypothesis for structured models is defined as a set functions $h : \mathcal{X} \to \mathcal{Y}$ such that

$$h(x) = \arg\max_{y \in \mathcal{Y}} \mathbf{w}.\Phi(x, y), \quad \forall x \in \mathcal{X}$$

The objective function for multi-class SVM model can be written as following:

$$\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \max_{y \neq y_i} \max\left(0, 1 - \mathbf{w}.\left[\Phi(x_i, y_i) - \Phi(x_i, y)\right]\right)$$

Suppose we are given a loss fnction $L(y, y')$, which need to be incorporated into the objective function somehow. This can be done in various ways. One way of applying this loss is to set it as margin threshold. Hence the bigher this prediction loss is, the bigger loss will be incured, upon violation. This would lead to *Max-Margin Markov Networks* (M^{3}N):

$$\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \max_{y \neq y_i} \max\left(0, L(y_i, y) - \mathbf{w}.\left[\Phi(x_i, y_i) - \Phi(x_i, y)\right]\right)$$

Another way is weighting the margin violation with this loss value. This would result in *SVM-struct* algorhtm:

$$\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \max_{y \neq y_i} L(y_i, y) \max\left(0, 1 - \mathbf{w}.\left[\Phi(x_i, y_i) - \Phi(x_i, y)\right]\right)$$

Instead of using hinge-loss, one can use exponential loss (which can be thought of a softmax). This would lead to objective function of *Conditional Random Fields (CRF)*:

$$\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \log \sum_{y \in \mathcal{Y}} \exp\left(L(y_i, y) - \mathbf{w}.\left[\Phi(x_i, y_i) - \Phi(x_i, y)\right]\right)$$