Machine Learning Theory (CS 6783)

Lecture 4: Learning Frameworks, Examples

1 Setting up learning problems

1. **$\mathcal{X}$: instance space or input space**
   Examples:
   - Computer Vision: Raw $M \times N$ image vectorized $\mathcal{X} = [0, 255]^{M\times N}$, SIFT features (typically $\mathcal{X} \subseteq \mathbb{R}^d$)
   - Speech recognition: Mel Cepstral co-efficients $\mathcal{X} \subset \mathbb{R}^{12\times \text{length}}$
   - Natural Language Processing: Bag-of-words features ($\mathcal{X} \subset \mathbb{N}^{\text{document size}}$), n-grams
2. **$\mathcal{Y}$: Outcome space, label space**
   Examples: Binary classification $\mathcal{Y} = \{\pm 1\}$, multiclass classification $\mathcal{Y} = \{1, \ldots, K\}$, regression $\mathcal{Y} \subset \mathbb{R}$)
3. **$\ell : \mathcal{Y} \times \mathcal{Y} \mapsto \mathbb{R}$: loss function** (measures prediction error)
   Examples: Classification $\ell(y', y) = \mathbb{1}_{\{y' \neq y\}}$, Support vector machines $\ell(y', y) = \max\{0, 1 - y' \cdot y\}$, regression $\ell(y', y) = (y - y')^2$
4. **$\mathcal{F} \subset \mathcal{Y}^{\mathcal{X}}$: Model/ Hypothesis class** (set of functions from input space to outcome space)
   Examples:
   - Linear classifier: $\mathcal{F} = \{x \mapsto \text{sign}(f^\top x) : f \in \mathbb{R}^d\}$
   - Linear SVM: $\mathcal{F} = \{x \mapsto f^\top x : f \in \mathbb{R}^d, \|f\|_2 \leq R\}$
   - Neural Netoworks (deep learning): $\mathcal{F} = \{x \mapsto \sigma(W_{out}\sigma(W_K\sigma(\ldots\sigma(W_2(W_1\sigma(W_{in}x)))))\}$ where σ is some non-linear transformation (Eg. ReLU)

Learner observes sample: $S = (x_1, y_1), \ldots, (x_n, y_n)$

Learning Algorithm : (forecasting strategy, estimation procedure)

$$\hat{\mathbf{y}} : \mathcal{X} \times \bigcup_{t=1}^{\infty} (\mathcal{X} \times \mathcal{Y})^t \mapsto \mathcal{Y}$$

Given new input instance x the learning algorithm predicts $\hat{\mathbf{y}}(x, S)$. When context is clear (ie. sample S is understood) we will fudge notation and simply use notation $\hat{\mathbf{y}}(\cdot) = \hat{\mathbf{y}}(\cdot, S)$. $\hat{\mathbf{y}}$ is the predictor returned by the learning algorithm.

Example: linear SVM Learning algorithm solves the optimization problem:

$$\mathbf{w}_{\text{SVM}} = \underset{\mathbf{w}}{\text{argmin}} \sum_{t=1}^{n} \max\{0, 1 - y_t \mathbf{w}^\top x_t\} + \lambda \|\mathbf{w}\|$$

and the predictor is $\hat{\mathbf{y}}(x) = \hat{\mathbf{y}}(x, S) = \mathbf{w}_{\text{SVM}}^\top x$

1.1 PAC framework

$$\mathcal{Y} = \{\pm 1\}, \quad \ell(y', y) = \mathbb{1}_{\{y' \neq y\}}$$

Input instances generated as $x_1, \ldots, x_n \sim D_X$ where D_X is some unknown distribution over input space. The labels are generated as

$$y_t = f^*(x_t)$$

where target function $f^* \in \mathcal{F}$. Learning algorithm only gets sample S and does not know f^* or D_X.

Goal: Find $\hat{\mathbf{y}}$ that minimizes

$$\mathbb{P}_{x \sim D_X}\left(\hat{\mathbf{y}}(x) \neq f^*(x)\right)$$

1.2 Non-parametric Regression

$$\mathcal{Y} \subseteq \mathbb{R}, \quad \ell(y', y) = (y' - y)^2$$

Input instances generated as $x_1, \ldots, x_n \sim D_X$ where D_X is some unknown distribution over input space. The labels are generated as

$$y_t = f^*(x_t) + \varepsilon_t \qquad \text{where } \varepsilon_t \sim N(0, \sigma)$$

where target function $f^* \in \mathcal{F}$. Learning algorithm only gets sample S and does not know f^* or D_X.

Goal: Find $\hat{\mathbf{y}}$ that minimizes

$$\mathbb{E}_{x \sim D_X}\left[(\hat{\mathbf{y}}(x) - f^*(x))^2\right] =: \|\hat{\mathbf{y}} - f^*\|_{L_2(D_X)}$$

1.3 Statistical Learning (Agnostic PAC)

Generic $\mathcal{X}$, $\mathcal{Y}$, ℓ and $\mathcal{F}$

Samples generated as $(x_1, y_1), \ldots, (x_n, y_n) \sim D$ where D is some unknown distribution over $\mathcal{X} \times \mathcal{Y}$.

Goal: Find $\hat{\mathbf{y}}$ that minimizes

$$\mathbb{E}_{(x,y) \sim D}\left[\ell(\hat{\mathbf{y}}(x), y)\right] - \inf_{f \in \mathcal{F}} \mathbb{E}_{(x,y) \sim D}\left[\ell(f(x), y)\right]$$

For any mapping $g : \mathcal{X} \mapsto \mathcal{Y}$ we shall use the notation $L_D(g) = \mathbb{E}_{(x,y) \sim D}\left[\ell(g(x), y)\right]$ and so our goal can be re-written as:

$$L_D(\hat{\mathbf{y}}) - \inf_{f \in \mathcal{F}} L_D(f)$$

Remarks:

1. $\hat{\mathbf{y}}$ is a random quantity as it depends on the sample
2. Hence formal statements we make will be in high probability over the sample or in expectation over draw of samples

1.4 Online Learning

For $t = 1$ to n

(a) Input instance $x_t \in \mathcal{X}$ is produced

(b) Learning algorithm outputs prediction $\hat{y}_t$

(c) True outcome y_t is revealed to learner

End For

One can think of $\hat{y}_t = \hat{\mathbf{y}}_t(x_t, ((x_1, y_1), \ldots, (x_{t-1}, y_{t-1})))$.

Goal: Find learning algorithm $\hat{\mathbf{y}}$ that minimizes regret w.r.t. hypothesis class $\mathcal{F} \subset \mathcal{Y}^{\mathcal{X}}$ given by,

$$\mathrm{Reg}_n = \frac{1}{n}\sum_{t=1}^{n} \ell(\hat{y}_t, y_t) - \inf_{f\in\mathcal{F}} \frac{1}{n}\sum_{t=1}^{n} \ell(f(x_t), y_t)$$

2 Example 1: Classification using Finite Class, Realizable Setting

In this section we consider the classification setting where $\mathcal{Y} = \{\pm 1\}$ and $\ell(y', y) = \mathbf{1}\{y' \neq y\}$. We further make the realizability assumption meaning $y_t = f^*(x_t)$ where f^* is obviously not known to the learner.

2.1 Online Framework

The online framework is just as described earlier with the realizability assumption added in. That is, at every round the true label y_t revealed to us is set as $y_t = f^*(x_t)$ for some fixed f^* not known to the learning algorithm. However x_t's can be presented to us arbitrarily. First note that under the realizability assumption, we have that

$$\min_{f\in\mathcal{F}} \frac{1}{n}\sum_{t=1}^{n} \ell(f(x_t), y_t) = \frac{1}{n}\sum_{t=1}^{n} \mathbf{1}\{f^*(x_t) \neq y_t\} = 0$$

Hence the aim in such a framework is to simply minimize number of mistakes $\sum_{t=1}^{n} \ell(\hat{y}_t, y_t)$ and prove mistake bounds.

Now say $\mathcal{F} = \{f_1, \ldots, f_N\}$, a finite set of hypothesis. What strategy can we provide for this problem? How well does it work?

If we simply pick some hypothesis that has not made a mistake so far, such an algorithm can make a large number of mistakes (Eg. as many as N). A simple strategy that works in this scenario is the following. At any point t, we have observed $x_1, \ldots, x_{t-1}$ and labels $y_1, \ldots, y_{t-1}$. Now say

$$\mathcal{F}_t = \{f \in \mathcal{F} : \forall i \in [t-1],\ f(x_i) = y_i\}\ .$$

Now given x_t, we pick $\hat{y}_t = \mathrm{sign}(\sum_{f\in\mathcal{F}_t} f(x_t))$. That is we go with the majority of predictions by hypothesis in $\mathcal{F}_t$. How well does this algorithm work?

Claim 1. *For any sequence $x_1, \ldots, x_n$, the above algorithm makes at most $\lceil \log_2 N \rceil$ number of mistakes.*

Proof. Notice that each time we make a mistake, ie. $\mathrm{sign}(\sum_{f \in \mathcal{F}_t} f(x_t)) \neq y_t$, then we know that at least half the number of functions in $\mathcal{F}_t$ are wrong and so each time we make a mistake, $|\mathcal{F}_{t+1}| \leq |\mathcal{F}_t|/2$ and hence, we can make at most $\log_2 N$ number of mistakes. $\square$

That is the average error is $\frac{\log_2 N}{n}$.

2.2 PAC Framework

In the PAC framework, $x_1, \ldots, x_n$ are drawn iid from some fixed distribution $D_{\mathcal{X}}$ and our goal is to minimize $P_{x \sim D_x}(\hat{\mathbf{y}}(x) \neq f^*(x))$ either in expectation or high probability over sample $\{x_1, \ldots, x_n\}$. Unlike the online setting, in the PAC setting one can simply pick any hypothesis that has not made any mistakes on training sample. That is,

$$\hat{\mathbf{y}}(\cdot, S) = \mathrm{argmin}_{f \in \mathcal{F}} \sum_{(x_t, y_t) \in S} \mathbf{1}\{f(x_t) \neq y_t\} \ .$$

How well does this algorithm work? How should we analyze this?

Let us show a bound of error with high probability over samples. To this end we will use the so called Bernstein concentration bound.

Fact: Consider binary r.v. $Z_1, \ldots, Z_n$ drawn iid. Let $\mu = \mathbb{E}[Z]$ be their expectation. We have the following bound on the average of these random variables. (notice that since Z's are binary their variance if given by $\mu - \mu^2$)

$$P\left(\mu - \frac{1}{n}\sum_{t=1}^{n} Z_t > \theta\right) \leq \exp\left(-\frac{n\theta^2}{2\mu + \frac{\theta}{3}}\right)$$

Now for any $f \in \mathcal{F}$, let $Z_t^f = \mathbf{1}\{f(x_t) \neq f^*(x_t)$ where x_t are drawn from $D_{\mathcal{X}}$. Note that $\mathbf{E}[Z^f] = P_{x \sim D_{\mathcal{X}}}(f(x) \neq f^*(x))$. Hence note that for any single $f \in \mathcal{F}$,

$$P_S\left(P_{x \sim D_{\mathcal{X}}}(f(x) \neq f^*(x)) - \frac{1}{n}\sum_{t=1}^{n} \mathbf{1}\{f(x_t) \neq f^*(x_t)\} > \theta\right) \leq \exp\left(-\frac{n\theta^2}{2\mu + \frac{\theta}{3}}\right)$$

Let use write the R.H.S. above as δ, and hence, rewriting, we have that with probability at least $1 - \delta$ over sample,

$$P_{x \sim D_{\mathcal{X}}}(f(x) \neq f^*(x)) - \frac{1}{n}\sum_{t=1}^{n} \mathbf{1}\{f(x_t) \neq f^*(x_t)\} \leq \frac{\log(1/\delta)}{3n} + \sqrt{\frac{P_{x \sim D_{\mathcal{X}}}(f(x) \neq f^*(x)) \log(1/\delta)}{n}}$$

This upon further massaging (use inequality $\sqrt{ab} \leq a/2 + b/2$) leads to the bound

$$P_{x \sim D_{\mathcal{X}}}(f(x) \neq f^*(x)) - \frac{2}{n}\sum_{t=1}^{n} \mathbf{1}\{f(x_t) \neq f^*(x_t)\} \leq \frac{2\log(1/\delta)}{n}$$

Using union bound, we have that for any $\delta > 0$, with probability at least $1 - \delta$ over sample, simultaneously,

$$\forall f \in \mathcal{F} \quad P_{x \sim D_{\mathcal{X}}}(f(x) \neq f^*(x)) - \frac{2}{n} \sum_{t=1}^{n} \mathbf{1}\{f(x_t) \neq f^*(x_t)\} \leq \frac{2 \log(|\mathcal{F}|/\delta)}{n}$$

Since $\hat{\mathbf{y}} \in \mathcal{F}$, from the above we conclude that, for any $\delta > 0$, with probability at least $1 - \delta$ over sample,

$$P_{x \sim D_{\mathcal{X}}}(\hat{\mathbf{y}}(x) \neq f^*(x)) - \frac{2}{n} \sum_{t=1}^{n} \mathbf{1}\{\hat{\mathbf{y}}(x_t) \neq f^*(x_t)\} \leq \frac{2 \log(|\mathcal{F}|/\delta)}{n}$$

But note that by realizability assumption and the definition of $\hat{\mathbf{y}}$, we have that

$$\sum_{t=1}^{n} \mathbf{1}\{\hat{\mathbf{y}} \neq f^*(x_t)\} = \sum_{t=1}^{n} \mathbf{1}\{\hat{\mathbf{y}} \neq y_t\} = 0$$

and so, with probability at least $1 - \delta$ over sample,

$$P_{x \sim D_{\mathcal{X}}}(\hat{\mathbf{y}}(x) \neq f^*(x)) \leq \frac{2 \log(|\mathcal{F}|/\delta)}{n}$$