COS 511: Theoretical Machine Learning

Lecturer: Rob Schapire
Scribe: Qinghua Liu

Lecture #20
April 17, 2019

1 Recap

In last lecture, we studied the density estimation problem. Specifically, we considered the problem of the following form:

- First, we are given:
  - A finite domain $\mathcal{X}$ with size N.
  - A set of features $f_1, f_2, \ldots, f_n$: $\mathcal{X} \to \mathbb{R}$.
- Second, we observe a set of samples $x_1, x_2, \ldots, x_m$ i.i.d. sampled from an unknown distribution D over $\mathcal{X}$.
- Our goal is to estimate the underlying distribution D from these samples.

We introduced two approaches to solve this problem.
The first one is motivated by the principle of maximum entropy.

$$q^* = \arg\max_{p \in \mathcal{P}} H(p), \text{ where } \mathcal{P} = \{p : \ \mathbb{E}_p[f_j] = \hat{\mathbb{E}}[f_j], \ j \in [n]\}, \tag{1}$$

where $\hat{\mathbb{E}}[f_j] = \frac{1}{m}\sum_{i=1}^m f_j(x_i)$ is the empirical average of feature f_j. Briefly speaking, we want to find a distribution q^* with maximum entropy such that the expectation of each feature over q^* is equal to its empirical average.

The second approach is to perform maximum likelihood estimation on the family of Gibbs distributions (or exponential-family distributions).

$$q^* = \arg\max_{q \in \bar{\mathcal{Q}}} \sum_{i=1}^m \ln q(x_i), \text{where } \mathcal{Q} = \left\{ q : \ q(x) = \exp\left(\sum_{j=1}^n \lambda_j f_j(x)\right) / Z_{\boldsymbol{\lambda}}, \ \boldsymbol{\lambda} \in \mathbb{R}^n \right\}, \tag{2}$$

where $Z_{\boldsymbol{\lambda}}$ is a normalization factor.

A key observation is that (1) and (2) are a pair of primal-dual problems. One can reformulate one of them into the other using the method of Lagrange multipliers. To go deeper, we have the following theorem.

Theorem 1. *The following are equivalent:*

- $q^* = \arg\max_{p \in \mathcal{P}} H(p)$.
- $q^* = \arg\max_{q \in \bar{\mathcal{Q}}} \sum_{i=1}^m \ln q(x_i)$.
- $q^* \in \mathcal{P} \cap \bar{\mathcal{Q}}$.

Furthermore, any one of these uniquely determines q^*.

Theorem 1 bridges two seemingly different formulations elegantly. However, it says nothing about how to solve them. The goal of this lecture is to introduce an algorithm based on coordinate descent to solve problem (2), and by the equivalence of the two problems the distribution we find will also be a solution of problem (1).

2 Algorithm

In this section, we will focus on the design of the algorithm.

2.1 A General Technique

First, note that maximizing log-likelihood is equivalent to minimizing the average log-loss, so we can rewrite problem (2) as

$$\min_{\boldsymbol{\lambda}\in\mathbb{R}^n} L(\boldsymbol{\lambda}) = -\frac{1}{m}\sum_{i=1}^{m} \ln q_{\boldsymbol{\lambda}}(x_i). \tag{3}$$

For simplicity of notation, we further define

$$q_{\boldsymbol{\lambda}}(x) = \frac{e^{g_{\boldsymbol{\lambda}}(x)}}{Z_{\boldsymbol{\lambda}}}, \text{ where } g_{\boldsymbol{\lambda}}(x) = \sum_{j=1}^{n} \lambda_j f_j(x).$$

Our goal is to find an iterative algorithm that can output a sequence $\{\boldsymbol{\lambda_t}\}_{t=1}^{\infty}$ satisfying that $L(\boldsymbol{\lambda_t})$ monotonically decreases with respect to t and converges to $\inf_{\boldsymbol{\lambda}} L(\boldsymbol{\lambda})$. Generally, the algorithm is of the following form.

- Choose $\boldsymbol{\lambda}_1$.
- For $t = 1, 2, 3, \ldots$
  - Compute $\boldsymbol{\lambda}_{t+1}$ from $\boldsymbol{\lambda}_t$.

One attractive way to compute $\boldsymbol{\lambda}_{t+1}$ is to find a $\boldsymbol{\lambda}$ that minimizes $L(\boldsymbol{\lambda}) - L(\boldsymbol{\lambda}_t)$ or at least makes this difference nonpositive. However, for many optimization problems, like this one, it's usually very difficult to optimize this quantity directly. To bypass this obstacle, we will introduce a very general technique in the following sections. The key point of it is to find an approximation (or upper bound) of $L(\boldsymbol{\lambda}) - L(\boldsymbol{\lambda}_t)$, which is usually much easier to optimize than the original one, and try to minimize this approximation exactly.

2.2 Approximating the Difference of Loss

Before we dive into mathematical details, let's make the following assumption without loss of generality:

$$f_j(x) \in [0, 1] \ \forall\ x \in \mathcal{X} \text{ and } j = 1, 2, \ldots, n. \tag{4}$$

To justify that this assumption can be made without loss of generality, it suffices to show we can rescale each feature f_j or add a constant to it arbitrarily without changing the optimization domain $\mathcal{Q}$. Recall that

$$\mathcal{Q} = \{q :\ q(x) = \exp(g_\lambda(x))/Z_\lambda, \text{ where } g_\lambda \in \text{span}\{f_1, f_2, \ldots, f_n\}\}, \tag{5}$$

where $\text{span}\{f_1, f_2, \ldots, f_n\}$ is the linear functional space spanned by the f_j's. Since rescaling the f_j's with nonzero constants will not change the linear space spanned by them, we justify the rescaling operation. Since any constant added to g_λ will be offset by the normalization factor Z_λ, we justify the shifting operation.

Now we are ready to derive an upper bound for $L(\boldsymbol{\lambda}_{t+1}) - L(\boldsymbol{\lambda}_t)$. In the following part, we will fix t and consider how to compute $\boldsymbol{\lambda}_{t+1}$ from $\boldsymbol{\lambda}_t$. To simplify notation, we let $\boldsymbol{\lambda} = \boldsymbol{\lambda_t}$ and $\boldsymbol{\lambda}' = \boldsymbol{\lambda}_{t+1}$. Then the update on the t'th iteration is as follows:

- On the t'th iteration
  - Choose an index j from $\{1, 2, \ldots, n\}$, and compute an update quantity α.
  - Update λ_j (the j'th coordinate of $\boldsymbol{\lambda_t}$) by adding α and keep the other coordinates unchanged, i.e.,

$$\lambda'_k = \begin{cases} \lambda_k + \alpha & \text{if } k = j, \\ \lambda_k & \text{else.} \end{cases} \tag{6}$$

A quick remark is that the algorithm is a special case of coordinate descent. Note that we haven't figured out how to choose j and α, so for now we just leave them as unspecified variables. Besides, since we only add α to the j'th coordinate of $\boldsymbol{\lambda}$ to get $\boldsymbol{\lambda}'$, we have $g_{\boldsymbol{\lambda}'}(x) = g_{\boldsymbol{\lambda}}(x) + \alpha f_j(x)$, which can be verified by using the definition of g.

Now, let's look at how the loss changes after the t'th iteration if we adapt the updating rule above.

$$\begin{aligned}
\Delta L &= L\left(\boldsymbol{\lambda}'\right) - L(\boldsymbol{\lambda}) \\
&= -\frac{1}{m}\sum_{i=1}^{m} \ln q_{\boldsymbol{\lambda}'}(x_i) + \frac{1}{m}\sum_{i=1}^{m} \ln q_{\boldsymbol{\lambda}}(x_i) && \text{expanding } L \\
&= -\frac{1}{m}\sum_{i=1}^{m} \ln\left(\frac{e^{g_{\boldsymbol{\lambda}'}(x_i)}}{Z_{\boldsymbol{\lambda}'}}\right) + \frac{1}{m}\sum_{i=1}^{m} \ln\left(\frac{e^{g_{\boldsymbol{\lambda}}(x_i)}}{Z_{\boldsymbol{\lambda}}}\right) && \text{by the definition of } q_{\boldsymbol{\lambda}} \text{ and } q_{\boldsymbol{\lambda}'} \\
&= \frac{1}{m}\sum_{i=1}^{m} \left(g_{\boldsymbol{\lambda}}(x_i) - g_{\boldsymbol{\lambda}'}(x_i)\right) + \ln\frac{Z_{\boldsymbol{\lambda}'}}{Z_{\boldsymbol{\lambda}}} \\
&= -\alpha\hat{\mathbb{E}}[f_j] + \ln\frac{Z_{\boldsymbol{\lambda}'}}{Z_{\boldsymbol{\lambda}}}. && \text{by the definition of } g \text{ and the updating rule (6)}
\end{aligned}$$

For the second term $\ln\frac{Z_{\boldsymbol{\lambda}'}}{Z_{\boldsymbol{\lambda}}}$, we have

$$\begin{aligned}
\frac{Z_{\boldsymbol{\lambda}'}}{Z_{\boldsymbol{\lambda}}} &= \sum_{x\in\mathcal{X}} \frac{\exp\left(g_{\boldsymbol{\lambda}}(x) + \alpha f_j(x)\right)}{Z_{\boldsymbol{\lambda}}} && \text{expanding } Z_{\boldsymbol{\lambda}'} \text{ and using the updating rule (6)} \\
&= \sum_{x\in\mathcal{X}} \frac{\exp\left(g_{\boldsymbol{\lambda}}(x)\right)}{Z_{\boldsymbol{\lambda}}} \exp\left(\alpha f_j(x)\right) \\
&= \sum_{x\in\mathcal{X}} q_{\boldsymbol{\lambda}}(x) \exp\left(\alpha f_j(x)\right) && \text{by the definition of } q_{\boldsymbol{\lambda}} \\
&= \mathbb{E}_{q_{\boldsymbol{\lambda}}}[\exp\left(\alpha f_j(x)\right)] \\
&\leq \mathbb{E}_{q_{\boldsymbol{\lambda}}}[1 + (e^{\alpha} - 1) f_j(x)] \\
&= 1 + (e^{\alpha} - 1)\mathbb{E}_{q_{\boldsymbol{\lambda}}}[f_j(x)],
\end{aligned}$$

where the inequality follows from the fact that for any $z \in [0, 1]$, we have $\exp(\alpha z) \leq 1 + (\exp(\alpha) - 1)z$.

Combining the two upper bounds, we get

$$\Delta L \leq -\alpha\hat{\mathbb{E}}[f_j] + \ln\left(1 + (e^{\alpha} - 1)\mathbb{E}_{q_{\boldsymbol{\lambda}}}[f_j(x)]\right). \tag{7}$$

Note that the right-hand side of (7) is a convex differentiable function of α, and its derivative with respect to α is

$$-\hat{\mathbb{E}}[f_j] + \frac{\mathbb{E}_{q_{\boldsymbol{\lambda}}}[f_j(x)]e^{\alpha}}{1 + (e^{\alpha} - 1)\,\mathbb{E}_{q_{\boldsymbol{\lambda}}}[f_j(x)]}. \tag{8}$$

By setting (8) equal to zero, we get

$$\alpha = \ln\left(\frac{\hat{\mathbb{E}}[f_j]}{1-\hat{\mathbb{E}}[f_j]} \cdot \frac{1-\mathbb{E}_{q_{\boldsymbol{\lambda}}}[f_j]}{\mathbb{E}_{q_{\boldsymbol{\lambda}}}[f_j]}\right). \tag{9}$$

For this minimizing choice of α, the right-hand side of (7) is equal to

$$-\mathrm{RE}(\hat{\mathbb{E}}[f_j]||\mathbb{E}_{q_{\boldsymbol{\lambda}}}[f_j]). \tag{10}$$

2.3 Formal Definition of the Algorithm

Now, we are ready to give the formal definition of our algorithm for solving problem (2).

- Initialize $\boldsymbol{\lambda}_1 = \mathbf{0}$.
- For $t = 1, 2, 3, \ldots$
  - Select $j_t = \arg\max_j \mathrm{RE}(\hat{\mathbb{E}}[f_j]||\mathbb{E}_{q_{\boldsymbol{\lambda}_t}}[f_j])$.
  - Compute

$$\alpha_t = \ln\left(\frac{\hat{\mathbb{E}}[f_{j_t}]}{1-\hat{\mathbb{E}}[f_{j_t}]} \cdot \frac{1-\mathbb{E}_{q_{\boldsymbol{\lambda}_t}}[f_{j_t}]}{\mathbb{E}_{q_{\boldsymbol{\lambda}_t}}[f_{j_t}]}\right). \tag{11}$$

  - For $k = 1, 2, \ldots, n$, update

$$\lambda_{t+1,k} = \begin{cases} \lambda_{t,k} + \alpha_t & \text{if } k = j_t, \\ \lambda_{t,k} & \text{else.} \end{cases} \tag{12}$$

The analysis in the last subsection immediately gives us
Claim 1. For any $t \geq 0$,

$$L\left(\boldsymbol{\lambda}_{t+1}\right) - L(\boldsymbol{\lambda_t}) \leq -\max_j \mathrm{RE}(\hat{\mathbb{E}}[f_j]||\mathbb{E}_{q_{\boldsymbol{\lambda}_t}}[f_j]). \tag{13}$$

3 Convergence Analysis

The goal of this section is to show $\{q_{\boldsymbol{\lambda}_t}\}_{t=1}^{\infty}$ converges to q^*, i.e., the unique global optimum of problem (2) .

Theorem 2. *Let $p_t = q_{\boldsymbol{\lambda}_t}$. Then $p_t \to q^*$.*

The key point of our proof is to construct and utilize an auxiliary function satisfying the following definition.

Definition 1. *A function $A : \{$probability distributions on $\mathcal{X}\} \to \mathbb{R}$ is an auxiliary function if it has the following three properties*

1. *A is continuous.*
2. *$L(\boldsymbol{\lambda}_{t+1}) - L(\boldsymbol{\lambda_t}) \leq A(p_t) \leq 0$.*
3. *For any distribution p on $\mathcal{X}$, if $A(p) = 0$, then $\mathbb{E}[f_j] = \mathbb{E}_p[f_j]$, $j = 1, 2, \ldots, n$.*

Proof.
Step 1.
Let's suppose such A exists and the distribution sequence $\{p_t\}_{t=1}^{\infty}$ converges to some distribution p. Since $p_t \in \mathcal{Q}$, we have $p \in \bar{\mathcal{Q}}$. Note that $\{L(\boldsymbol{\lambda_t})\}_{t=1}^{\infty}$ is a monotonically decreasing sequence lower bounded by 0, so $L(\boldsymbol{\lambda}_{t+1}) - L(\boldsymbol{\lambda_t})$ converges to zero. Using the second property of the auxiliary function that $A(p_t)$ is sandwiched by $L(\boldsymbol{\lambda}_{t+1}) - L(\boldsymbol{\lambda_t})$ and 0, we have $A(p_t)$ converges to zero as well.

Furthermore, by the continuity of A, we have

$$A(p) = A(\lim_{t\to\infty} p_t) = \lim_{t\to\infty} A(p_t) = 0.$$

The third property of the auxiliary function immediately gives us

$$\mathbb{E}[f_j] = \mathbb{E}_p[f_j],\ j = 1, 2, \ldots, n,$$

which is equivalent to saying that $p \in \mathcal{P}$. Therefore, $p \in \mathcal{P} \cap \bar{\mathcal{Q}}$. Recall that Theorem 1 tells us $\mathcal{P} \cap \bar{\mathcal{Q}}$ only contains one element q^*, which implies $p = q^*$.
Step 2.
Let's show

$$A(p) = -\max_j \mathrm{RE}(\hat{\mathbb{E}}[f_j] || \mathbb{E}_p[f_j])$$

is such an auxiliary function. First, it can be verified that $A(p)$ is continuous and non-positive. Besides, we have

$$A(p) = 0 \ \Rightarrow\ \forall j:\ \mathrm{RE}(\hat{\mathbb{E}}[f_j] || \mathbb{E}_p[f_j]) = 0.$$

Since the relative entropy is zero if and only if its arguments are equal, this completes the proof. □

Remark 1. *At the beginning of the proof, we assumed the sequence $\{p_t\}$ converges. This assumption can be justified in the following way: first, note that all the p_t's are in the N-dimensional simplex which is a compact set, so every subsequence of $\{p_t\}$ has a converging subsubsequence; second, the proof of Theorem 2 tells us all the limit points of $\{p_t\}$ belong to $\mathcal{P} \cap \bar{\mathcal{Q}}$; finally, since $\mathcal{P} \cap \bar{\mathcal{Q}}$ contains only one element q^*, we conclude that every subsequence of $\{p_t\}$ has a converging subsubsequence whose limit point is q^*, which can be shown to imply that $\{p_t\} \to q^*$.*

4 Next: Online Log-loss

After working on a batch-version log-loss minimization problem, let's turn to the online setting. There are two different motivations for using log-loss in the online setting.

4.1 Log-loss in Online Learning

The first motivation is online learning. For instance, suppose we are betting on horse racing and there are some experts providing predictions for each round. Particularly, the predictions are given in the form of winning distributions over the horses. At the beginning of each round, we can observe the predictions of all the experts and then combine them to form our own prediction which is also a winning distribution q over the horses. Then, we

will observe the winner x of this round, which will incur a loss equal to $-\ln q(x)$, where $q(x)$ is the probability that we assigned to the winner in this round. More generally, we have

- For $t = 1 \dots T$
  - Each expert i predicts a distribution $p_{t,i}$ over $\mathcal{X}$.
  - Learner chooses distribution q_t over $\mathcal{X}$.
  - Observe $x_t \in \mathcal{X}$.
  - Incur loss $-\ln q_t(x_t)$.

Similar to the online learning problems we discussed before, our goal is to come up with an algorithm whose performance is nearly as good as the best expert even when the data is given in an adversarial way. So we seek a bound of the following form:

$$-\sum_{t=1}^{T} \ln q_t(x_t) \leq \min_i \left(-\sum_{t=1}^{T} \ln p_{t,i}(x_t) \right) + \text{small regret}. \tag{14}$$

4.2 Log-loss in Coding Theory

Another motivation for using log-loss in the online setting arises from coding theory. Not surprisingly, this is about the story saying that Alice wants to send Bob a message.

First, let's start with the simplest setting. Suppose Alice wants to send Bob a single English letter chosen from $\mathcal{X} = \{a, b, \dots\}$, and they both believe the probability of sending x is $p(x)$. Then they will use $-\log_2 p(x)$ bits to encode x, if they want to achieve minimum average coding length $H(x)$.

Now suppose Alice wants to send Bob a sequence of letters or a whole sentence. In this case, our prediction for the next letter will heavily depend on the context, i.e., the letters already sent. For example, if Bob has received "I am goin__", then he should predict the next letter to be 'g' with high probability. Therefore, in this setting, the main problem is how to assign probability to each letter based on the letters already sent so that we can use less bits to encode the same amount of information. Mathematically, it's equivalent to estimating the distribution of x_t given $x_1^{t-1} = \langle x_1, \dots, x_{t-1} \rangle$.

The story of Alice and Bob gives us another perspective to understand log-loss for the online setting. Suppose we have N coding methods, and let $p_{t,i}(x_t)$ be the estimated conditional probability of x_t conditioned on x_1^{t-1} provided by the i'th coding method. Also, similar to the problems related with experts before, we do not know which coding method is the best ahead of time. We want to find an algorithm to combine all these coding methods such that its coding length is nearly as short as the best coding method. Then, we can reinterpret (14) in the following way

- $\sum_{t=1}^{T} (-\log q_t(x_t))$ is the number of bits our algorithm uses to transmit the entire message $x_1, \dots, x_T$.
- $\sum_{t=1}^{T} (-\log p_{t,i}(x_t))$ is the number of bits the i'th coding method uses.

Thus, a bound like (14) tells us that for any message the algorithm will use almost as few bits as the best of the given coding methods, since its coding length is bounded by the coding length of the best coding method plus a small term.