Mathematical Foundations of Machine Learning(CS 4783/5783)

Lecture 11: Online Linear and Convex Optimization

1 Online Linear Optimization

Online linear optimization is a special case of online learning for which the loss function is linear. The learner picks $\mathbf{f}$ from a convex (bounded) set $\mathcal{F}$ of vectors in $\mathbb{R}^d$ and the adversary plays d-dimensional vectors ∇_t also in some bounded set. The loss is simply $\nabla_t^\top \mathbf{f}_t$. The online learning protocol is as follows:

For $t = 1$ to n

1. Learner picks $\mathbf{f}_t \in \mathcal{F}$
2. Adversary picks $\nabla_t \in \mathbb{R}^d$
3. Learner observes vector ∇_t and suffers linear loss $\mathbf{f}_t^\top \nabla_t$

End For

The goal as usual is to minimize regret with respect to best linear predictor in hindsight:

$$\text{Reg}_n = \frac{1}{n}\sum_{t=1}^{n} \nabla_t^\top \mathbf{f}_t - \inf_{\mathbf{f}\in\mathcal{F}} \frac{1}{n}\sum_{t=1}^{n} \nabla_t^\top \mathbf{f}$$

2 Online Gradient Descent

The main algorithm we will consider for this problem is the so called online gradient descent algorithm. For this example we assume $\mathcal{F} = \{\mathbf{f} : \|\mathbf{f}\|_2 \leq R\}$ and assume that every ∇_t is such that $\|\nabla_t\|_2 \leq B$.

Algorithm :

$$f_{t+1} = \Pi_{\mathcal{F}}\left(f_t - \eta \nabla_t\right)$$

where $\Pi_{\mathcal{F}}$ is the Euclidean projection on to set $\mathcal{F}$ and $\eta > 0$ is referred to as step-size.

$$\Pi_F(f) = \begin{cases} f & \text{if } \|\mathbf{f}\|_2 \leq R \\ R\frac{\mathbf{f}}{\|\mathbf{f}\|_2} & \text{otherwise} \end{cases}$$

Claim 1. *If we use the online gradient descent algorithm with $\eta = \frac{R}{B\sqrt{n}}$ and $\mathbf{f}_1 = \mathbf{0}$, then*

$$\frac{1}{n}\sum_{t=1}^{n} \nabla_t^\top \mathbf{f}_t - \inf_{\mathbf{f}\in\mathcal{F}} \frac{1}{n}\sum_{t=1}^{n} \nabla_t^\top \mathbf{f} \leq \frac{RB}{\sqrt{n}}$$

Proof. Fix any $\mathbf{f}^* \in \mathcal{F}$. Note that,

$$\|\mathbf{f}_{t+1} - \mathbf{f}^*\|_2^2 = \|\Pi_{\mathcal{F}}(\mathbf{f}_t - \eta\nabla_t) - \mathbf{f}^*\|_2^2 \leq \|\mathbf{f}_t - \eta\nabla_t - \mathbf{f}^*\|_2^2 = \|\mathbf{f}_t - \mathbf{f}^*\|_2^2 + \eta^2\|\nabla_t\|_2^2 - 2\eta\nabla_t^\top(\mathbf{f}_t - \mathbf{f}^*)$$

Thus we can conclude that

$$\nabla_t^\top(\mathbf{f}_t - \mathbf{f}^*) \leq \frac{1}{2\eta}\left(\|\mathbf{f}_t - \mathbf{f}^*\|_2^2 - \|\mathbf{f}_{t+1} - \mathbf{f}^*\|_2^2\right) + \frac{\eta}{2}\|\nabla_t\|_2^2$$

Summing we get,

$$\begin{aligned}\sum_{t=1}^n(\nabla_t^\top\mathbf{f}_t - \nabla_t^\top\mathbf{f}^*) &\leq \frac{1}{2\eta}\sum_{t=1}^n\left(\|\mathbf{f}_t - \mathbf{f}^*\|_2^2 - \|\mathbf{f}_{t+1} - \mathbf{f}^*\|_2^2\right) + \frac{\eta}{2}\sum_{t=1}^n\|\nabla_t\|_2^2 \\ &= \frac{1}{2\eta}\left(\|\mathbf{f}_1 - \mathbf{f}^*\|_2^2 - \|\mathbf{f}_{n+1} - \mathbf{f}^*\|_2^2\right) + \frac{\eta}{2}nB^2 \\ &\leq \frac{1}{2\eta}R^2 + \frac{\eta}{2}nB^2\end{aligned}$$

Using the η from the claim and dividing throughout by n gives the result. □

What is the lower bound for this problem? In fact it is not hard to see that the lower bound for this problem is also $\frac{RB}{\sqrt{n}}$ at least when dimensionality is huge. To see this assume the adversary simply plays on each round vector orthogonal to current $\mathbf{f}_t$ and also orthogonal to previous $\nabla_1, \ldots, \nabla_{t-1}$.

3 Online Convex Optimization

While we have a nice result for linear losses, linear losses are not what we typically encounter in ML. While arbitrary non-convex losses are the most general setting one could hope to cover, non-convexity cones at the cost of being both too complex to learn sometimes and also being computationally intensive. A common approach in ML modeling is to model the loss as a convex loss. Next we consider the case of online convex optimization where the online learning problem is such that the loss is convex in $\mathbf{f} \in \mathcal{F}$. The goal as usual is to minimize regret with respect to best predictor in hindsight:

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

We assume $\mathcal{F}$ is a convex bounded set and $\ell(\mathbf{f}; (x, y))$ is convex in $\mathbf{f}$ for every x, y.

Though we are concerned with general convex losses, it suffices (in many cases with no additional cost) to only consider online linear optimization where the loss is linear rather than general convex. The reason for this is the following. First, given any $(x_1, y_1), \ldots, (x_n, y_n)$ let $\mathbf{f}^* = \operatorname{argmin}_{\mathbf{f}\in\mathcal{F}} \sum_{t=1}^n \ell(\mathbf{f}; (x_t, y_t))$. Now note that by convexity,

$$\begin{aligned}\sum_{t=1}^n \ell(\mathbf{f}_t; (x_t, y_t)) - \sum_{t=1}^n \ell(\mathbf{f}^*; (x_t, y_t)) &\leq \sum_{t=1}^n \langle\nabla\ell(\mathbf{f}_t; (x_t, y_t)), \mathbf{f}_t - \mathbf{f}^*\rangle \\ &\leq \sum_{t=1}^n \langle\nabla\ell(\mathbf{f}_t; (x_t, y_t)), \mathbf{f}_t\rangle - \inf_{\mathbf{f}\in\mathcal{F}}\sum_{t=1}^n \langle\nabla\ell(\mathbf{f}_t; (x_t, y_t)), \mathbf{f}\rangle\end{aligned}$$

Now since in the online learning protocol, learner picks $\mathbf{f}_t \in \mathcal{F}$ and then adversary picks (x_t, y_t), we can simply think of adversary as directly picking any ∇_t directly and this only increases the bound. Thus,

$$\frac{1}{n}\sum_{t=1}^{n} \ell(\mathbf{f}_t; (x_t, y_t)) - \inf_{\mathbf{f}\in\mathcal{F}} \frac{1}{n}\sum_{t=1}^{n} \ell(\mathbf{f}; (x_t, y_t)) \leq \frac{1}{n}\sum_{t=1}^{n} \langle \nabla_t, \mathbf{f}_t \rangle - \inf_{\mathbf{f}\in\mathcal{F}} \frac{1}{n}\sum_{t=1}^{n} \langle \nabla_t, \mathbf{f} \rangle$$

What the above means is that if we have an algorithm for online linear optimization, we can use it as an algorithm for online convex optimization assuming the instance received on round t is the gradients of the convex function at the point $\mathbf{f}_t$.

In view of the above, it is clear why the term "gradient" in online gradient descent makes sense. If we us the above reduction, for convex losses, we would use the gradient of the loss as ∇_t in the online gradient descent update. Each update we take one step in the direction of the gradient of the current loss.