Online Convex Programing

Lecturer: Drew Bagnell *Scribe:Minh Vo* [1]

1 Online gradient descent

1.1 Instantaneous regret

Let $l_i = (w_t^T f_i - y_i)^2$ our loss functions, where w is an expert and f is a feature. We want to minimize the total regret in retrospect with respect to the best expert w^*:

$$R(w) = \sum_{t=0}^{T} l_t(w_t) - l_t(w^*). \tag{1}$$

We call $l_t(w_t) - l_t(w^*)$ the instantaneous regret for some w_t at time t.

1.2 Global lower bound property of projected subgradient

As we all know, subgradient $\nabla f(x)$ at point x has the property that $f(y) \geq f(x) + \nabla f(x)^T (y - x)$ for any $x \neq y$.For our instantaneous regret, we have

$$\begin{aligned} l_t(w^*) &\geq l_t(w_t) + \nabla l_t(w_t)^T (w^* - w_t) \\ l_t(w_t) - l_t(w^*) &\leq \nabla l_t(w_t)^T (w_t - w^*). \end{aligned} \tag{2}$$

The left hand side is the instantaneous regret, and the right hand side is some linear function times $(w_t - w^*)$. Thus our total regret, $R(w) = \sum_{t=0}^{T} l_t(w_t) - l_t(w^*)$, will be bounded by $\sum_{t=0}^{T} \nabla l_t(w_t)^T (w_t - w^*)$.

[1] Some content adapted from previous scribes: Siyuan Feng and Ji Zhang.

1.3 Algorithm for projected online subgradient descent

This algorithm is a method to minimize the regret for a online convex optimization problem.
Line 5 projects $\hat{w}_{t+1}$ back into the convex set C, and α in line 4 is the learning rate. Smaller α

Algorithm 1 Projected Subgradient Descent():

1: choose w_0
2: **for** $t = 1...T$ **do**
3: Incur loss $l(w_t)$ and receive any $\nabla l_t(w_t)$
4: $\hat{w}_{t+1} \leftarrow w_t - \alpha \nabla l_t(w_t)$
5: $w_{t+1} \leftarrow Proj_c[\hat{w}_{t+1}]$
6: **end for**

pays a larger upfront cost but is more likely to converge and has a lower regret over time. α can also be dependent on t. Note that the projection will not cause the loss to grow, because it will bring $\hat{w}_{t+1}$ closer to any member of C, and thus closer to the optimal expert w^* too.

2 Regret bounds for projected subgradient descent

2.1 Distance between w_t and w^*

The distance between w_t and w^* at time t is defined as

$$D(w_t, w^*) = (w_t - w^*)^T (w_t - w^*) \tag{3}$$

Now we look at

$$\begin{aligned} & D(w_{t+1}, w^*) - D(w_t, w^*) \\ & = (w_t - \alpha \nabla l_t(w_t) - w^*)^2 - (w_t - w^*)^2 \\ & = (z_t - \alpha \nabla l_t(w_t))^2 - z_t^2 \\ & = \alpha^2 (\nabla l_t(w_t))^2 - 2\alpha \nabla l_t^T(w_t) z_t, \end{aligned} \tag{4}$$

where $z_t = w_t - w^*$. If we sum all the term over time, the intermediate terms will all cancel out and leave us just $D(w_T, w^*) - D(w_0, w^*)$.

$$\begin{aligned} & = \sum_t D(w_{t+1,w^*}) - D(w_t, w^*) \\ & = -2\alpha \sum_t (w_t - w^*) \nabla l_t + \alpha^2 \sum_t |\nabla l_t|^2 \\ & = D(w_T, w^*) - D(w_0, w^*) \\ & \leq -2\alpha \sum_t (w_t - w^*) \nabla l_t + \alpha^2 GT, \end{aligned} \tag{5}$$

where $|\nabla l_t|^2 \leq G$. Thus we have

$$2\alpha R_T \leq 2\alpha \sum_t (w_t - w^*) \nabla l_t \leq D(w_0, w^*) - D(w_T, w^*) + \alpha^2 GT \tag{6}$$

Since the distance between w_T and w^* is always non negative, we can throw away the $D(w_T, w^*)$ term and still keep the inequality valid.

$$R_T \leq \sum_t (w_t - w^*)\nabla l_t \leq \frac{D(w_0, w^*)}{2\alpha} + \frac{\alpha GT}{2} \leq \frac{\alpha GT}{2} + \frac{F}{2\alpha}, \tag{7}$$

where F is the largest distance between any two experts in the set.
Suppose we set $alpha = \sqrt{\frac{F}{GT}}$, then the upper bound for total regret is bounded by $\sqrt{GTF}$, growing sub linearly of T.

3 Portfolio Optimization - No Regret Portfolio

3.1 General algorithm description

We want to invest in n different stocks given a set of investment weights w_i s.t. $w_i \in R$, $w_i \geq 0$, and $\sum w_i = 1$. We also know the market returns ratios $r_i = \frac{value^i_{t+1}}{value^i_t}$. So the daily increase in wealth is $w_t^T r_t$, and the total wealth over time is $\prod_t w_t^T r_t$ We want to maximize $\log \Pi w_t^T r_t = \sum \log w_t^T r_t$. We can use use the following algorithm:

$$w_0 \leftarrow \frac{1}{n} \tag{8}$$

$$w_{t+1} \leftarrow Proj[w_t + \frac{\alpha r_t}{w_t^T r_t}] \tag{9}$$

We will optimize $\sum \log w_t^T r_t$ instead of the original cost $w_t^T r_t$. The procedure of this problem is described in Algorithm 1.

In the next lecture, we will look at the regret bound along with some problems of applying the naive version of this algorithm in the real world.