Machine Learning **Fall 2023**

Lecture 10: Online Learning and Unsupervised Learning

Lecturer: Liwei Wang

Scribe: Haoran Liu, Hongjie Li, Ang Li, Jiaxuan Xie, Zhongwang Fu, Yuan Cao, Yunze Chen, Qiaoshu Li, Zifeng Wang, Jingxuan Wang, Yi Zhang

Disclaimer: *These notes have not been subjected to the usual scrutiny reserved for formal publications. They may be distributed outside this class only with the permission of the Instructor.*

10.1 Online Learning

The setting of online learning with expert advice can be described as follows:

1. There are T timesteps $t = 1, 2, \cdots, T$, each timestep t has a ground truth label $y_t \in \{0, 1\}$
2. There are N experts, at each timestep t, each expert i makes a prediction $\widetilde{y}_{t,i} \in \{0, 1\}$
3. The online learner needs to make a prediction $\widetilde{y}_t \in \{0, 1\}$ based on the experts' predictions up until now and the previous ground truth labels

The performance can be evaluated with ”regret”, which is the performance of the online learner minus the performance of the best expert.

10.1.1 Weighted Majority Vote

Algorithm 1 Weighted Majority Vote

```
Define online learner's prediction at timestep $t$ $\widetilde{y}_t$, ground truth label at timestep $t$ $y_t \in \{0,1\}$, the number of experts $N$
Input expert $i$'s advice at timestep $t$ $\widetilde{y}_{t,i}$, parameter $\beta \in (0,1)$
Initialize $w_{1,i} = 1$
for $t = 1, 2, ..., T$ do
    Predict $\widetilde{y}_t = I[\Sigma_i w_{t,i} \widetilde{y}_{t,i} \geq \frac{1}{2} \Sigma_i w_{t,i}]$
    if $\widetilde{y}_t = y_t$ then
        $w_{t+1,i} = w_{t,i}$
    else
        $w_{t+1,i} = \begin{cases} \beta w_{t,i}, & \widetilde{y}_{t,i} \neq y_t \\ w_{t,i}, & \widetilde{y}_{t,i} = y_t \end{cases}$
    end if
end for
```

Let $L_T = \Sigma_t I[\widetilde{y}_t \neq y_t], m_{T,i} = \Sigma_t I[\widetilde{y}_{t,i} \neq y_t], m_T^* = \min_i m_{T,i}$, then we have $L_T \leq \frac{\log \frac{1}{\beta}}{\log \frac{2}{1+\beta}} m_T^* + \frac{\log N}{\log \frac{2}{1+\beta}}$.

Proof:

Let $W_t = \Sigma_i w_{t,i}$, then for every t satisfying $\hat{y}_t \neq y_t$, at least half of the weights needs to be multiplied by β, so $W_{t+1} \leq \frac{1+\beta}{2} W_t$.

So we have $W_{T+1} \leq W_1 (\frac{1+\beta}{2})^{L_T} = N(\frac{1+\beta}{2})^{L_T}$.

At the same time, the best expert makes m_T^* mistakes, so its weight decreases at most m_T^* times, so after T timesteps the weight must be larger or equal to $\beta^{m_T^*}$, and we have $W_{T+1} \geq \beta^{m_T^*}$.

With $N(\frac{1+\beta}{2})^{L_T} \geq \beta^{m_T^*}$, we can get $L_T \leq \frac{\log \frac{1}{\beta}}{\log \frac{2}{1+\beta}} m_T^* + \frac{\log N}{\log \frac{2}{1+\beta}}$.

■

10.1.2 Randomized Weighted Majority Vote

Algorithm 2 Randomized Weighted Majority Vote

```
Define online learner's prediction at timestep $t$ $\widetilde{y}_t$, ground truth label at timestep $t$ $y_t \in \{0,1\}$, the number of experts $N$
Input expert $i$'s advice at timestep $t$ $\widetilde{y}_{t,i}$, parameter $\beta \in (0,1)$
Initialize $w_{1,i} = 1$
for $t = 1, 2, ..., T$ do
    Predict $\widetilde{y}_t \sim \text{Bernoulli}(\frac{\Sigma_i w_{t,i} \widetilde{y}_{t,i}}{\Sigma_i w_{t,i}})$
    $w_{t+1,i} = \begin{cases} \beta w_{t,i}, & \widetilde{y}_{t,i} \neq y_t \\ w_{t,i}, & \widetilde{y}_{t,i} = y_t \end{cases}$
end for
```

The only differences between this algorithm and the deterministic algorithm are that the online learner's prediction is stochastic and that the weights of experts who make wrong predictions are always decreased no matter whether the online learner makes a correct prediction.

With the same notations in the above section, we have $\mathbb{E} L_T \leq (2-\beta) m_T^* + \frac{\log N}{1-\beta}$

Proof:

We have $\mathbb{E} L_T = \Sigma_t \Sigma_i \frac{w_{t,i}}{W_t} |\widetilde{y}_{t,i} - y_t|$.

If we define p_t as the possibility of the online learner makes a wrong prediction at timestep t, then let W_t be the potential function, then $W_{t+1} = W_t - \Sigma_i (1-\beta) w_{t,i} I[\widetilde{y}_{t,i} - y_t] = (1 - p_t(1-\beta)) W_t \leq \exp(-p_t(1-\beta)) W_t$

So $W_{T+1} \leq \exp(-(1-\beta)\Sigma_t p_t) W_1 = N \exp(-(1-\beta)\mathbb{E} L_T)$

As $W_{T+1} \geq \beta^{m_T^*}$, we have $N \exp(-(1-\beta)\mathbb{E} L_T) \geq \beta^{m_T^*}$, that is $\log N - (1-\beta)\mathbb{E} L_T \geq m_T^* \log \beta$, which means $\mathbb{E} L_T \leq -\frac{m_T^* \log \beta}{1-\beta} + \frac{\log N}{1-\beta}$

With Taylor expansion we have $\log(1 + (\beta - 1))) \geq (\beta-1) - (\beta-1)^2$ as $\beta - 1 \leq 0$, so $\mathbb{E} L_T \leq (2-\beta) m_T^* + \frac{\log N}{1-\beta}$

■

Besides, if we set β to be $1 - \sqrt{\frac{\log N}{T}}$, we have $\mathbb{E} \frac{L_T}{T} \leq (1 + \sqrt{\frac{\log N}{T}}) \frac{m_T^*}{T} + \sqrt{\frac{\log N}{T}} \leq \frac{m_T^*}{T} + 2\sqrt{\frac{\log N}{T}}$

If we don't know T in advance, we can use the doubling trick or other similar methods by adjusting β in the process. (This is part of the optional homework, so the answer won't be provided here.)

10.2 Unsupervised Learning

10.2.1 Clustering

Clustering is an unsupervised learning task, and is described as follows:

Given a set of data $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in \mathbb{R}^d$ and a non-zero integer $k \leq n$, clustering (k-means clustering) aims to divide these n data into k sets $S_1, S_2, \ldots, S_k$ so as to minimize the following loss function:

$$\phi = \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \mathbf{c}_{(i)}\|^2$$

where $\mathbf{c}_i$ is the cluster center of S_i.

The most common algorithm is $k-$means algorithm

Algorithm 3 k-means algorithm

```
Initialize: choose $k$ points randomly as the cluster centers $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_k$
repeat
    Assign each data to the cluster center with the nearest mean
    $S_i \leftarrow \{\mathbf{x}_j : \mathbf{x}_j$ is assigned to $\mathbf{c}_i\}$
    $\mathbf{c}_i \leftarrow$ the mean of points in $S_i$
until $k$ cluster centers not change
return $\{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_k\}$
```

However, this naive algorithm is only guaranteed to find a local optimum. $k-$means++ is an improvement over the original k-means algorithm. We can optimize the initialization step in $k-$means.

Algorithm 4 Improved initialization

```
Choose $\mathbf{c}_1$ from $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ uniformly.
for $i = 2, ..., k$ do
    Choose $\mathbf{c}_i$ from $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ with the distribution of $\frac{D(\mathbf{x}_i)}{\sum_j D(x_j)}$,
    where $D(x_i) = \min_{1 \leq s < i} \left\{\|\mathbf{x}_i - \mathbf{c}_s\|^2\right\}$
end for
```

Thm Let ϕ be the loss of $k-$means++, let ϕ_{OPT} be the optimal value of the objective function. Then

$$\mathbb{E}[\phi] \leq 8(\log k + 2) \cdot \phi_{OPT}$$

References

[1] https://www.cl.cam.ac.uk/teaching/2122/RandAlgthm/lec14_experts.pdf