Multiplicative Weights and Boosting

Saagar Sanghavi, Jin Seok Won, Kevin Zhu

Ensemble Methods

In the context of machine learning, there are many different models you can apply in a given situation.

In many applications, we can get the best results through using many different models at the same time, or using same model with different data set or features.

Experts Problem

In the Experts problem setup, you have experts that you can choose to trust every day. Everyday, following things happen:

You choose an expert to trust and they make a claim

At the end of the day, depending on what happens on that day (maybe it can be a stock market trade) each expert accumulates a loss

Your Regret is the difference in loss between your chosen expert and the daily best expert (one with least loss)

Multiplicative Weights

At first glance, it might be hard to believe that there is an optimal way to choose an expert. However, with clever use of probability and update function, multiplicative weights method proved to be the optimal solution to the problem.

Multiplicative Weights are optimal in the sense that an adversary who knows how you will bet can incur the least amount of regret at the end of the day.

Weight updates

$$w_i^{(0)} = 1$$

$$w_i^{(t+1)} = w_i^{(t)} \cdot (1-\epsilon)^{\ell_i^{(t)}}$$

You bet on expert i with probability w^^i

Bagging

Bias variance trade off shows that when applying a model to a data set, getting good bias (low bias) often leads to high variance, and vice versa. There are also techniques that perform very inconsistently depending on the data set or the features, like decision trees.

Bagging addresses this problem by applying the model to different sets of data set or features and then combining them by simply taking the average. This results in lower variance if different iterations are relatively uncorrelated.

Boosting

Boosting combines models (especially weak learners) together for better performance

Focus of the boosting:

- Find learners that can correctly classify data point that current model is wrong about
- Weighing models by current performance each time a model is added
- “Harder” points classified with more models

Connection to OMP

In both cases:

- Overall predictor is an additive combination of pieces selected through greedy process
- Keeps track of residual which is used to perform some form of line search

Adaboost

Short for adaptive boosting

Algorithm:

1. Initialize the weights $w_i = \frac{1}{n}$ for all $i = 1, \ldots, n$ training points.
2. Repeat for $m = 1, \ldots, M$:
   (a) Build a classifier $G_m : \mathbb{R}^d \to \{-1, 1\}$, where in the training process the data are weighted according to w_i.
   (b) Compute the weighted error $e_m = \frac{\sum_{i \text{ misclassified}} w_i}{\sum_i w_i}$.
   (c) Re-weight the training points as

$$w_i \leftarrow w_i \cdot \begin{cases} \sqrt{\frac{1-e_m}{e_m}} & \text{if misclassified by } G_m \\ \sqrt{\frac{e_m}{1-e_m}} & \text{otherwise} \end{cases}$$

   (d) Optional: normalize the weights w_i to sum to 1.