Class Notes Summary on Sep 16

by Andres Rodriguez

1. REVIEW FROM PREVIOUS LECTURE

Note on Bayes Nets:

$$P(A,B) = P(A)P(B|A) = P(B)P(B|A) = \phi(A,B)/z$$

This means that $A \to B$ is equivalent to $B \to A$ and $A - -B$.

However, Bayes nets were originally made to encode causality. It is much easier for a doctor to tell the symptoms given a disease, than to tell the disease given the symptoms (since many diseases shared the same symptoms).

2. ON-LINE LEARNING

This area is concerned with making decisions from limited information. We begin by studying the problem of "predicting from expert advice." (Keep in mind that the word "expert" may be deceiving since many of these "experts" may in reality have no clue.)

Let's say each day we have the same n experts that predict yes (1) or no (0) on whether is going to rain. After they make their prediction, the algorithm makes its own prediction, and then finds out if it actually it rains. Our goal is do perform nearly as well as the best expert so far (being competitive with respect to the best single expert).

2.1 Weighted Majority Algorithm

This algorithm maintains a list of weights $w_1, ..., w_n$ (one for each expert $x_1, ..., x_n$), and predicts based on a weighted majority vote, penalizing mistakes by multiplying their weight by half.

Algorithm

1. Set all the weights to 1.
2. Predict 1 (rain) if $\sum_{x_i=1} w_i \geq \sum_{x_i=0} w_i$, and 0 otherwise.
3. Penalize experts that are wrong: for all i s.t. x_i made a mistake, $w_i^{t+1} \leftarrow \frac{1}{2} w_i^t$.
4. Goto 2

Analysis of Algorithm The sum of the weights is $w \leq \left(\frac{3}{4}\right)^m n = \left(\frac{4}{3}\right)^{-m} n$, where m is the number of mistakes. The weight of the best expert $w_i^* = \left(\frac{1}{2}\right)^{m^*} = 2^{-m^*} \leq w$. Therefore,

$$2^{-m^*} \leq w \leq \left(\frac{4}{3}\right)^{-m} n.$$

Taking the $\log_2$ and solving for m gives,

$$m \leq \frac{m^* + \log_2 n}{\log \frac{4}{3}} = 2.41(m^* + \log_2 n)$$

2.2 Randomized Weighted Majority Algorithm

In this algorithm, we view the weights as probabilities, and predict each outcome with probability to its weight. We also penalized each mistake by β instead of just half.

Algorithm

1. Set all the weights to 1.
2. Choose expert i in proportion to w_i.
3. Penalize experts that are wrong: for all i s.t. x_i made a mistake, $w_i^{t+1} \leftarrow \beta w_i^t$.
4. Goto 2.

Analysis The bound is

$$E[m] \leq \frac{m * \ln(1/\beta) + \ln(n)}{1 - \beta}.$$

We want β to be small if we only have a few time steps available and vice-verse.

2.3 General Weighted Majority Algorithm

Algorithm

1. Set all the weights to 1.
2. Predict expert i in proportion to w_i.
3. Receive the correct value y_t.
4. Adjust weights s.t. $w_i^{t+1} \rightarrow w_i^t \exp^{-\epsilon l(i,y_t)} \forall i$, where l is the lost function, and ϵ is the penalizer.
5. Goto 2.

Analysis The bound is

$$E[R] \leq \epsilon \sum l(i^*) + \frac{1}{\epsilon} \ln(n),$$

where R is the regret. If ϵ is large, we learn fast. If ϵ is small, we learn slow but we'll have little regret.

3. INTRO TO NEXT LECTURE

In a convex set, the line between any two points in the set is also in the set. A convex function can be defined s.t.

$$f(ax_1 + bx_2) \leq af(x_1) + bf(x_2),$$

where $a + b = 1$, and $a, b \geq 0$.