E1 245: Online Prediction & Learning **Fall 2014**

Lecture 1 — August 5

Lecturer: Aditya Gopalan *Scribe: Sayak Ray Chowdhury*

1.1 Course organisation

Part 1: Learning without distributional assumptions.
–Prediction with expert advice
Part 2: Learning in stochastic models
–Bandit problems
Part 3: Choice of topics:
–Reinforcement learning
–Stochastic games

1.2 What is learning?

Learning is the ability to **improve performance by observing data**. Consider, an agent in an environment who has a task to do. The agent may be any forecaster or algorithm, the environment may be adversarial in nature and the task is learning the setup, if environment is unknown or optimize the performance, if it is known. Learning problem can also be thought of as a repeated game between the agent and the environment or adversary, which we will be describing in detail later.
Learning can be broadly described in two ways:

- **Batch learning:** Given a batch of data, want to extract rules.
- **Online learning:** Data comes in one at a time.

Here, we will focus on online learning.

1.2.1 Online learning

(i) Learn one step at a time.
(ii) Data is being revealed sequentially.

1.2.2 Motivation

(i) Data is really being revealed sequentially with time e.g. **stock market**.
(ii) Too expensive to store/crunch/process all past data.

1.2.3 Examples to motivate online learning

1. Weather prediction:
Here weather is the environment, prediction is the task and weather prediction office is the agent.
2. Recommender systems:
Recommender systems are a subclass of information filtering system that seek to predict the rating or preference that user would give to an item. Say, Amazon.com wants to recommend a useful product to a customer. Here agent is amazon, task is recommending the product and environment is customer. Other examples include customized web search etc.
3. Cognitive radio:
A cognitive radio is an intelligent radio that can be programmed and configured dynamically. Its transceiver is designed to use the best wireless channels in its vicinity. Such a radio automatically detects available channels in wireless spectrum, then accordingly changes its transmission or reception parameters to allow more concurrent wireless communications in a given spectrum band at one location. Here Agent is the radio, task is transmitting and environment is an available channel in the spectrum.
4. Clinical trials:
Investigating the effects of different experimental treatments while minimizing patient losses. This is one of the very first learning problems developed by Thompson in 1930s. The multi-armed bandit problem is motivated by this example.
5. Congestion control in networks:
Adaptive routing efforts for minimizing delays/controlling congestion in a network is a well known learning problem and also motivates the bandit problem.
6. Source coding in unknown source:
In information theory, data compression or source coding involves encoding information using fewer bits than the original representation. Here task is to encode and environment is the unknown source.
7. Robot trying to navigate in a room:
Here agent is robot, task is navigating and environment is room.
8. Finance:
Examples in finance include Online auction strategies, Universal portfolio allocation, Black-Scholes option pricing theory etc.

1.2.4 Historical background/evolution (Brief Survey)

1. Playing repeated games: Research in this area is motivated from the works of Blackwell, Hannan and Robbins, dating back to the 1950s.
2. Universal compression: Rissanen, Lempel, Cover (1970s).
3. Finance/gambling: Started with Kelly in 1950s, followed by Cover's work in universal portfolio allocation in 1990s.
4. In more recent times:
Weighted majority algorithm(Littlestone-Warmuth 1992), Online convex optimization(Zinkevich 2003)etc.

1.3 Notation

We will follow the notations below:
1. Capital letters to denote random variables.
2. Lower-case letters to denote non random variables.
3. $a := b^2$ means a is assigned/defined to be b^2.

1.4 Warm-up: 1-bit prediction with expert advice

Suppose you are trying to predict movement of a stock each day(can be extended to any time unit). This can be formalized as below:

Sequence of rounds: t=1, 2, 3,..., T
In each round: environment picks $y_t \in \{0,1\}$, for example stock price goes up($y_t = 1$)or goes down($y_t = 0$).
Pool of experts: E=$\{1,2,3,...,N\}$, for example financial analysts, news media, algorithms or schemes.
At each time t=1, 2, 3, ...
(i) Observe experts' recommendations $(f_{i,t})_{i=1}^{N}$; $(f_{i,t}) \in \{0,1\}$
(ii) Make a prediction $\widehat{y_t} \in \{0,1\}$, using all information upto round t-1
(iii) Then get to observe y_t, outcome picked by the environment

1.4.1 Goal

Your goal will be to minimize the number of mistakes/wrong predictions after T rounds, which can be written as:
$M_T(A) = \sum_{t=1}^{T} \mathbb{1}\{\widehat{y_t} \neq y_t\}$, where A is any rule/algorithm you choose to predict.

We will now describe a common heuristic/rule for prediction with expert advice.

1.4.2 Majority/Halving algorithm(MAJ)

We start with the assumption that there exists atleast an expert who is always guessing correctly(later this will be relaxed). The algorithm is as follows:
1. At the beginning each expert is in the set of ”trustworthy” experts denoted by S.
2. At each round t,
(a) Go with the majority vote of all the experts in S.
(b) After seeing y_t, throw out all the experts from S that made a wrong prediction in this round.
The performance of this algorithm(MAJ) is given by the following theorem:

Theorem 1.1. *If there exists a perfect expert, majority algorithm will make at most* $\log_2 N$*number of mistakes, i.e.*
$M_T(MAJ) \leqslant \log_2 N$

Note that, this bound is independent of total number of rounds T, it depends only on the number of experts. Also, one can see that this bound of $\log_2 N$ is especially tight for deterministic algorithms.

Proof: By our assumption that a perfect expert always exists, the set S can never be empty. Now, each time MAJ makes a mistake, we remove at least half the experts form S. So, after i mistakes of MAJ, number of remaining experts $\leqslant N/2^i$. Now, as S is never empty, we have
$N/2^i \geqslant 1 \Longrightarrow i \leqslant \log_2 N$
Hence, the bound follows.

□

Clearly, the assumption that there always exists a perfect expert is rather strong and unrealistic. In the real world, nobody is perfect. So, consider the case when even the best expert makes m mistakes. In that case, a straightforward modification of majority algorithm will give us the bound $O((m+1)\log_2 N)$