**CSC 696H: Topics in Bandits and Reinforcement Learning Theory** **Fall 2024**

# Lecture 3: Online Learning

*Lecturer: Chicheng Zhang* *Scribe: Baoying Feng*

## 1 Introduction

Online Learning is a form of machine learning where data becomes available in a sequential order, and learning updates are made incrementally as each new data point is received. It contrasts with traditional batch learning, where the model is trained on the entire dataset at once.

## 2 Online Learning Protocol

In online learning, the goal is to make predictions sequentially while receiving data points one at a time. The learner needs to make a prediction before knowing the correct label, and the learner's performance is evaluated based on how close these predictions are to the true labels.

- **Instance space:** $\mathcal{X}$, the set of possible inputs.
- **Label space:** $\mathcal{Y}$, the set of possible labels.
- **Prediction space:** $\hat{\mathcal{Y}}$, where $\hat{y} \in \mathcal{Y}$.
- **Loss function:** $\ell(\hat{y}, y)$ is a measure of how far the predicted value $\hat{y}$ is from the true label $y$. The loss function provides a way to quantify the prediction error.

$$\ell(\hat{y}, y) = |\hat{y} - y| \quad \text{(absolute loss)}$$
$$\ell(\hat{y}, y) = (\hat{y} - y)^2 \quad \text{(squared loss)}$$
$$\ell_{log}(p, y) = log(\frac{1}{p(y)}) \quad \text{(log loss)}$$

  The loss function where $y$ is the discrete label space, and $\hat{y} = \Delta(y) = \{P : \text{p is a probability distribution over } y\}$ (conditional density estimation)

- **Predictor class:** In online learning, the learner chooses predictors from a predefined class of functions that map inputs to outputs.

  Formally we denote it by $\mathcal{F} \subseteq \mathcal{X} \to \hat{\mathcal{Y}}$, which is the set of functions or models the learner can choose from. These are functions that predict the output (label) based on the input.

At each time step $t$, the learner performs the following steps:

1. **Choose a predictor:** The learner selects a function $f_t \in \mathcal{F}$, which maps input $x_t$ to a prediction $\hat{y}_t$. This choice is typically made based on historical data $\{(x_s, y_s)\}_{s=1}^{t-1}$ observed in previous rounds.
2. **Observe new data:** The learner receives a new data point $(x_t, y_t)$, where $x_t$ is the input and $y_t$ is the corresponding label.

The objective of the learner is to choose predictors such that the total prediction error is minimized over time. The challenge lies in the fact that the learner only has access to past data and must make predictions without seeing future data.

## 2.1 Performance Measure: Regret

In online learning, the performance of the learner is typically measured using the concept of *regret*. Regret quantifies how well the learner performs compared to the best predictor in hindsight (i.e., after all the data has been seen).

- **Regret:** The regret after $T$ rounds is defined as:

$$\text{Regret}(\mathcal{F}, T) = \sum_{t=1}^{T} \ell(f_t(x_t), y_t) - \min_{f^* \in \mathcal{F}} \sum_{t=1}^{T} \ell(f^*(x_t), y_t)$$

  The first term is the cumulative loss incurred by the learner, and the second term is the cumulative loss of the best predictor $f^*$ in the class $\mathcal{F}$.

- If the regret grows sublinearly, i.e., $\frac{\text{Regret}(\mathcal{F},T)}{T} \to 0$ as $T \to \infty$, the learner is considered to be performing well, because this means that over time the learner's predictions are as good as the best predictor in hindsight.

## 2.2 Examples of Loss Functions

Several loss functions are commonly used to quantify prediction error:

1. **Absolute Loss:**

$$\ell(\hat{y}, y) = |\hat{y} - y|$$

   This loss measures the absolute difference between the predicted value $\hat{y}$ and the true label $y$. It is useful when the magnitude of the error is important.

2. **Square Loss:**

$$\ell(\hat{y}, y) = (\hat{y} - y)^2$$

   The squared loss penalizes larger errors more than smaller ones. This is a popular choice in regression problems where we care about minimizing the squared difference between predictions and actual values.

3. **Log Loss**

$$\ell_{\log}(p, y) = \log\left(\frac{1}{p(y)}\right) = -\log(p(y))$$

   This loss measures the error between the predicted probability $p(y)$ assigned to the true label $y$ and the actual outcome. The log loss penalizes lower confidence in the correct label and rewards high confidence. It is widely used in classification tasks where the model outputs probabilities, and minimizing this loss encourages the model to assign higher probabilities to the true label.
   If the model predicts the true class with high probability, the log loss will be small. However, if the predicted probability for the true class is small, the log loss will increase.

   **Data Compression Example:** In the context of online learning, consider the scenario where the labels $y_t$ represent tokens that are part of a data compression task. The learner uses a coding strategy that assigns each token to a code word. The length of the code word is $log(\frac{1}{p(y)})$, where $p$ is the probability distribution over the tokens.
   This relates to loss minimization in that shorter code words are assigned to more frequent tokens, optimizing the overall efficiency of the coding strategy.

# 3 Connection to Supervised Learning

## 3.1 Key Concepts

- **Online Learning vs. Supervised Learning**: In online learning, the boundary between training and test data is blurred. The learner updates its model after every data point, whereas in supervised learning, the model is trained on a fixed training set and tested afterward.
- **No Assumption on Data Generation Process in Online Learning**: Online learning doesn't require the assumption that data points are independent and identically distributed (IID), a common assumption in supervised learning.

## 3.2 Online to Batch Conversion

There is a connection between online learning algorithms (which aim to minimize regret) and supervised learning algorithms (which aim to generalize well). Specifically, an online algorithm can be converted into a batch algorithm with low generalization error.

Here is the problem setup:

- **Online Learning Algorithm $\mathcal{O}$**: An online learning algorithm with low regret – suppose $\mathcal{O}$, on any sequence of examples, have a regret bound of $\text{Regret}_{\mathcal{O}}(\mathcal{F}, T)$.
- **Supervised Learning Algorithm $\mathcal{S}$**: The converted algorithm can be used on a dataset $\{(x_t, y_t)\}_{t=1}^{T} \sim$ iid from $D$ to output a predictor $f$ with low expected loss $L_D(f)$.

## 3.3 Online to Batch Conversion Algorithm

**Algorithm (Online to Batch Conversion)**: Input: Dataset $\{(x_t, y_t)\}_{t=1}^{T}$, Online learning algorithm $\mathcal{O}$

For $t = 1, 2, \ldots, T$:

1. Algorithm $\mathcal{O}$ outputs a predictor $\hat{f}_t$.
2. Algorithm $\mathcal{O}$ receives $(x_t, y_t)$ and update its state

Output $\overline{f}(x) = \frac{1}{T}\sum_{t=1}^{T} \hat{f}_t(x)$ Alternatively: $\overline{f} \sim \text{unif}(\{\hat{f}_i ... \hat{f}_T\})$.

## 3.4 Theorem and Assumptions

**Theorem**: Assume that for all $t$, the loss function $\ell(\hat{y}, y)$ is bounded within $[0, B]$, and that the loss is convex in $f\hat{y}$ (this is true for several loss functions like absolute and square loss). Then, with high probability (specifically, at least $1 - \delta$): algorithm $\mathcal{S}$ obtained by online to batch conversion on $\mathcal{O}$, outputs $\bar{f}$ such that:

$$L_D(\hat{f}) \overset{(a)}{\leq} \frac{1}{T}\sum_{t=1}^{T} L_D(\hat{f}_t) \overset{(b)}{\leq} L_D(f^*) + 2B\sqrt{\frac{2ln(\frac{4}{\delta})}{T}} + \frac{\text{Regret}_{\mathcal{O}}(\mathcal{F}, T)}{T}$$

Where $f^* = \arg\min_{f \in \mathcal{F}} L_D(f)$.

## 3.5 Proof of Eq. $(b)$

$\mathcal{O}$ guarantees that:

$$\frac{1}{T}\sum_{t=1}^{T} \ell(\hat{f}_t(x_t), y_t) \leq \frac{1}{T}\sum_{t=1}^{T} \ell(f^*(x_t), y_t) + \frac{\text{Regret}_{\mathcal{O}}(\mathcal{F}, T)}{T} \tag{1}$$

The proof of this theorem follows these key steps:

1. Find events $F_1, F_2$:
   $F_1 : \left|\frac{1}{T}\sum_{t=1}^{T} \ell(f^*(x_t), y_t) - L_D(f^*)\right| \leq B\sqrt{\frac{2\ln\frac{4}{\delta}}{T}}$
   $F_2 : \left|\frac{1}{T}\sum_{t=1}^{T} \ell(f_t(x_t), y_t) - \frac{1}{T}\sum_{t=1}^{T} L_D(\hat{f}_t)\right| \leq \varepsilon_T$; where $\varepsilon_T = B\sqrt{\frac{2\ln\frac{4}{\delta}}{T}}$
   show that $P(F_1), P(F_2) \geq 1 - \frac{\delta}{2}$
2. Argue that when $F = F_1 \bigcap F_2$ happens, Equation $(b)$ holds.
3. Argue that $P(F) \geq 1 - \delta$

Let's now carry out our plan.

1. We will tackle item 3 first. We begin by applying a basic result from probability theory regarding the union of events.

   **Lemma 1.** *if $E_1$ and $E_2$ are events such that $P(E_1) \geq 1 - \delta_1$ and $P(E_2) \geq 1 - \delta_2$, then:*

   $$P(E_1 \cap E_2) \geq 1 - (\delta_1 + \delta_2)$$

   This means that the probability of both events occurring is at least $1 - (\delta_1 + \delta_2)$.

   *Proof.* Our goal is to show that:
   $$P(\mathcal{F}_1 \cup \mathcal{F}_2^c) \leq \delta_1 + \delta_2$$
   Since:
   $$P(\mathcal{F}_1^c) \leq \delta_1 \quad \text{and} \quad P(\mathcal{F}_2^c) \leq \delta_2$$
   We can conclude that:
   $$P(\mathcal{F}_1^c \cup \mathcal{F}_2^c) \leq \delta_1 + \delta_2$$

   □

2. For item 2, it simply follows by Eq. (1) and the definition of event $F$. (Note that when $F$ happens, the terms on the left and right of Eq. (1) can be replaced by $\frac{1}{T}\sum_{t=1}^{T} L_D(\hat{f}_t)$ and $L_D(f^*)$ respectively, with additional slack $2\epsilon_T$ added.)

3. For the rest of the proof we focus on verifying item 1.

   Next, we use Hoeffding's inequality to bound the probability of deviations from the expected loss for $F_1$.

   From our previous lecture, we know that

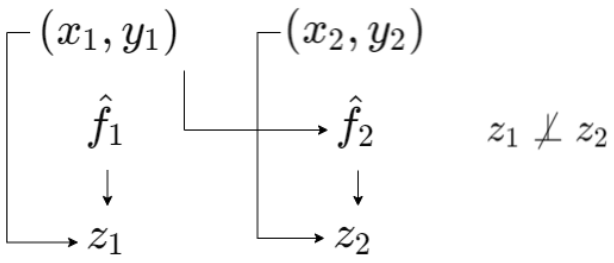
   $$P\left(\left|\frac{1}{T}\sum_{t=1}^{T} \ell(f^*(x_t), y_t) - L_D(f^*)\right| \geq \epsilon_T\right) \leq 2\exp\left(\frac{-2T\epsilon_T^2}{B^2}\right) \leq \delta/2,$$

   so $P(F_1) \geq 1 - \delta/2$.

   For $F_2$, ideally, we would like to use Hoeffding's inequality to get the same inequality:

   $$P\left(\left|\frac{1}{T}\sum_{t=1}^{T} \ell(f_t(x_t), y_t) - L_D(f_t)\right| \geq \epsilon_T\right) \leq 2\exp\left(\frac{-2T\epsilon_T^2}{B^2}\right)$$

   However, Hoeffding's inequality is no longer applicable, since $\frac{1}{T}\sum_{t=1}^{T}\left(\ell(f_t(x_t), y_t) - L_D(f_t)\right)$ is not an average of iid random variables.




4. To handle the deviation of noniid random variables, we utilize Azuma's inequality, which applies to martingales. Consider a sequence of random variables $Y_1, Y_2, \ldots, Y_T$ such that $Y_t \in [-B, B]$ and the conditional expectation $\mathbb{E}[Y_t|Y_1, \ldots, Y_{t-1}] = 0$. Then:

$$P\left(\left|\sum_{t=1}^{T} Y_t\right| \geq \epsilon\right) \leq 2\exp\left(\frac{-\epsilon^2}{2TB^2}\right)$$

Azuma's inequality generalizes Hoeffding's inequality for sequences that evolve in a dependent manner.

5. **Equivalently**: For any $\eta > 0$, we have:

$$P\left(\left|\sum_{t=1}^{T} Y_t\right| \geq \sqrt{2TB^2 \ln \frac{2}{\eta}}\right) \leq \eta$$

6. We now apply Azuma's inequality to our sequence of losses. Let $Z_t = \ell(f_t(x_t), y_t) - L_D(f_t)$. We need to check two conditions:

$$Z_t = \frac{1}{T}\sum_{t=1}^{T} (\ell(f_t(x_t), y_t) - L_D(f_t))$$

- $Z_t \in [-B, B]$: This is true, as the loss function is bounded.
- $\mathbb{E}[Z_t|Z_1, \ldots, Z_{t-1}] = 0$:
  This holds because:

  $\mathbb{E}[\ell(f_t(x_t), y_t) - L_D(\hat{f}_t)[\text{history up to } t-1, \hat{f}_t] = 0 \Rightarrow \mathbb{E}[\ell(\hat{f}_t(x_t), y_t) - L_D(\hat{f}_t) \mid Z_t, Z_{t-1}, \ldots] = 0$

  Here, the "$\Rightarrow$" step uses the Law of the Iterative Expectation:

$$\mathbb{E}[\mathbb{E}[A \mid C]] = \mathbb{E}[\mathbb{E}[A \mid B, C] \mid C]$$

Therefore, Azuma's inequality can be applied to sequence $\{Y_t\}_{t=1}^{T} = \{Z_t\}_{t=1}^{T}$. We have the bound from Azuma's inequality:

$$\eta = \frac{\delta}{2} \Rightarrow \left|\sum_{t=1}^{T} Z_t\right| \leq \sqrt{2TB^2 \ln \frac{4}{\delta}} \quad \text{happens with probability } 1 - \frac{\delta}{2}$$

Thus, we conclude that:

$$P(F_2) \geq 1 - \frac{\delta}{2}$$

By combining the bounds from Hoeffding's and Azuma's inequalities, we conclude that with probability at least $1 - \delta$, the expected loss of the online learner is close to the expected loss of the best predictor, up to the regret term and a small deviation $\epsilon_T$.

### 3.6 Proof of Eq. $(a)$

Eq. $(a)$ is a consequence of the convexity of the loss function $\ell(\hat{y}, y)$ with respct to prediction $\hat{y}$. We leave this to be an exercise. We recall the definition of convex functions below:

A function $f : \mathcal{X} \to \mathbb{R}$ is said to be convex if, for all $x_1, x_2 \in \mathcal{X}$ and $\alpha \in [0, 1]$:

$$f(\alpha x_1 + (1 - \alpha)x_2) \leq \alpha f(x_1) + (1 - \alpha)f(x_2)$$

Convexity is crucial for proving regret bounds and ensuring that the algorithm converges to the optimal solution. In many cases, loss functions used in online learning, such as the squared loss, are convex, which simplifies the analysis.

## 4 Basic Online Learning Algorithm & Analysis

The goal is to Design an **Online Learning Algorithm** with **sublinear regret**.

### 4.1 Iterative Empirical Risk Minimization (ERM)

Also known as **Follow the Leader (FTL)**.

At each time step $t$, the learner chooses:

$$\hat{f}_t = \arg\min_{f \in \mathcal{F}} \sum_{s=1}^{t-1} \ell(f(x_s), y_s)$$

where $\hat{f}_t$ is the predictor chosen based on past observations up to time $t - 1$.

#### 4.1.1 Example

- Hypothesis class: $\mathcal{F} = \{f_+, f_-\}$
- Input space: $\mathcal{X} = \{z\}$
- Label space: $\mathcal{Y} = \{-1, +1\}$
- Loss function: $\ell(\hat{y}, y) = |\hat{y} - y|$ (absolute loss)

Here is what happens if we run FTL on a sequence of examples $(z, -1), (z, +1), (z, -1), (z, +1), \ldots$:

| $t$ | $(z, y_t)$ | $\hat{f}_t$ | $\ell(f_t(x_t), y_t)$ | $\ell(f_t(x_t), y_t)$ | $\ell(f_t(x_t), y_t)$ |
|---|---|---|---|---|---|
| 1 | $(z, -1)$ | $f_+$ | 2 | 2 | 0 |
| 2 | $(z, +1)$ | $f_-$ | 2 | 0 | 2 |
| 3 | $(z, -1)$ | $f_+$ | 2 | 2 | 0 |

... and so on for subsequent time steps.

#### 4.1.2 Cumulative Loss, Benchmark, and Regret Analysis

The cumulative loss of the algorithm is calculated as the total loss incurred by the chosen predictors over all time steps. For the FTL algorithm, this can result in high cumulative loss in certain cases. In this example, the cumulative loss of the algorithm is:

$$\sum_{t=1}^{T} \ell(\hat{f}_t(x_t), y_t) = 2T$$

Next, we compare this with the performance of the best predictor in hindsight. The benchmark loss for the best fixed predictor, which minimizes the total loss over all time steps, is:

$$\min_{f \in \mathcal{F}} \sum_{t=1}^{T} \ell(f(x_t), y_t) \leq \frac{T}{2} \times 2 = T$$

Finally, the regret of the algorithm, which is the difference between the cumulative loss of the algorithm and the benchmark, can be computed as follows:

$$\text{Regret}(\mathcal{F}, T) \geq 2T - T = T$$

This analysis shows that the Follow the Leader (FTL) algorithm can suffer from linear regret, meaning that the regret grows proportionally with $T$, making it unsuitable for achieving sublinear regret in this scenario.