CSE 446: Machine Learning **Lecture 1**

Introduction

Instructor: Sham Kakade

1 What is ML?

Broadly, what is "learning"?

- Wikipedia: " Learning is the process of acquiring new or modifying existing knowledge, behaviors, skils, values, or preferences. Evidences that learning has occurred may be seen in changes in behavior from simle to complex."
- What is "machine learning"?
  Have a computer solve the learning process.
- Why is it 'technically' interesting?
  - statistically: why should the past say anything about the future?
  - computationally: in an automated manner, how do we go about finding a mapping from past data to a method which can (accurately) make future predictions?
- An AI centric viewpoint: ML is about getting computers to do the types of things people are good at, through "learning".
- How is it different from AI?
  When people say "AI" they almost always mean "ML."
- Why is it exciting?
  (note all hype) ML is starting to work:
  - No longer just an academic pursuit...
  - Almost "overnight" impacts to society:
    (threshold) improvements in performance translate into societal impact
  - AlphaGo, computer vision, search, Alexa/Siri, etc.

2 Logistics

Please read the course website for all policies.

- me: Sham Kakade (instructor)
- TAs: Kousuke Ariga, Benjamin Evans, Shobhit Hathi, Alina Liokumovich, Mathew (Xi) Liu, Thomas Merth, Patrick Spieker, Yuchong (Yvonna) Xiang.

- Course website: please read for all policies.
  `https://courses.cs.washington.edu/courses/cse446/18wi/`
- Contact: `cse446-staff@cs.washington.edu`
  Please only use this email for course related questions (unless privacy is needed).
- Discussion: please use canvas.
- Office hours: TBA.
- Textbooks:
  - "A Course in Machine Learning", Hal Daume.
  - "Machine Learning: A Probabilistic Perspective", Kevin Murphy.
  - See website for more supplementary info.
- Pre-reqs:
  - probability/statistics, multivariate calculus/linear algebra, programming abilities
  - HW0 will be a review of these topics.
  - If you are particularly weak in any of these topics, expect it to take maybe 10 additional hours per week per topic.

2.1 Grading

See the website for the official policy. This is meant to be a summary not the official policy:

- Homework (40%) + Midterm (20%) + Final exam (40%)
- Homework:
  - 5 in total.
  - both theory (pencil and paper) and applied (data/programming).
  - Late policy: 33% off for (up to) one day late; 66% off for (up to) two days late; ...
  - HW scores:

    $$\max\{0.1 * \text{HW0} + 0.225(\text{HW1} + \text{HW2} + \text{HW3} + \text{HW4}), 0 * \text{HW0} + 0.25(\text{HW1} + \text{HW2} + \text{HW3} + \text{HW4})\}$$

  - Assigned readings: They are helpful
- Caveat: Your grade may go up or down in extreme cases.
  (down) Failure to hand in all the HW, (up) very strong exam scores
- You MUST make the exam dates (unless you have an exception based on UW policies). Do not enroll in the course otherwise.

2.2 To-Do List

- Certify/submit you read the website.
- HW0 posted today

3 ML paradigms

Unsupervised learning:

- We wish to discover some properties of an underlying (and unknown) distribution.
- X is a datapoint (also referred to as an example, an instance, an input, a data point. X has some distribution $\Pr(X)$.
- Dataset: we get N datapoints $X_1 \dots X_N$ (sampled according to $\Pr(X)$.
- Examples: X_i's are images; we have a collection of images where each X_i's are sampled from some underlying distribution. X_i's are webpages or documents, and each example is sampled from some underlying distribution.
- possible goals: "cluster" the data points (and argue that the underlying distribution tends ot have groups); topic modeling; learn a method to predict the next word from the previous word.
- The objective is often not well defined; we usually have to decide what our objective is.

Supervised Learning:

- We have a set of N training examples, $\{(x_1, y_1), (x_2, y_2), \dots (x_n, y_n), \}$. Here, each x_i is referred to as the inputs, the feature vector, the covariates, or sometimes as the "x's"', and each y_i is referred to as the outputs, the labels, the targets, the dependent variable, or the "y's".
- Our goal is to learning a function f which accurately maps the x to y. A learning algorithm takes
- Here there is well defined objective/cost function; we want high accuracy in our predictions of the Y's.
- Examples: x's are images, and the y's is the name of the object in the image. $x's$ is a sentence in English and y is a sentence in French.

Reinforcement learning:

- This is near to the most general model. There is an "agent" acting in either a known or unknown world.
- Examples: robotic control (e.g. grasping an object), self driving cars, chess, alphaGo, etc.
- We can think of the process as follows: the agent is in some (often observed) "state"; the agent takes an action; the agent obtains some reward based on their current state and chosen action; the agent moves to the next state based on underlying model (e.g. the physics of the world or the rules of the game).
- The objective is (often) well defined: to maximize some long term notation of future reward. The difficulty is that our actions impact the future.

4 The Supervised Learning Setting

- Let $\mathcal{D}(x, y)$ define the (unknown) underlying probability of input/output pair (x, y), in "nature." (Almost always) we do not know this distribution.
- $\mathcal{X}$ is the *domain* of the x's. $\mathcal{Y}$ is the *domain* of the y's.

- The training data $D = \langle (x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N) \rangle$ are assumed to be *identical, independently, distributed* (i.i.d.) samples from $\mathcal{D}$.
- A *hypothesis* is our "predictor" which maps an x to a y, i.e. $\widehat{f} : \mathcal{X} \to \mathcal{Y}$.
- Let ℓ be a *loss function*; $\ell(y, \hat{y})$ is our loss by predicting $\hat{y}$ when y is the correct output.
- The *learning algorithm* maps the training set D to a some hypothesis $\widehat{f} : \mathcal{X} \to \mathcal{Y}$.
- Goal: find a *hypothesis* which as has "low" expected error, using the training set, where the expected error is:

$$\epsilon(f) = \sum_{(x,y)} \mathcal{D}(x, y) \cdot \ell(y, f(x)) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[\ell(y, f(x))]$$

  We care about our *expected error* (i.e. the expected loss, the true loss, ...) with regards to the underlying distribution $\mathcal{D}$.

Learning with the training set:

- The *training error* of f is the loss of f on the training set.

$$\widehat{\epsilon}(f) = \frac{1}{N} \sum_{n=1}^{N} \ell(y_n, f(x_n))$$

- Typically, we try to find an f that does well on $\widehat{\epsilon}(f)$
- Also: The **generalization error** is often referred to as the difference between the training error of $\hat{f}$ and the expected error of $\hat{f}$.

Example loss functions:

- $\ell(y, \hat{y})$ is our loss by outputting $\hat{y}$ when y is the correct output.
- Many loss functions:
  - For binary classification, where $y \in \{0, 1\}$:

$$\ell(y, \hat{y}) = \mathbf{1}\{y \neq \hat{y}\}$$

  - For multi-class classification, where y is one of k-outcomes:

$$\ell(y, \hat{y}) = \mathbf{1}\{y \neq \hat{y}\}$$

  - For regression, where $y \in \mathbb{R}$, we often use the square loss:

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