LECTURE 1/23

CS 181

Contents

1 What is this course about?

1. AI is very powerful but also dangerous.
2. Goal of this course is to learn how to develop ML models with rigor, professionalism, and ethics.

1.1 Class vs. Real Life

Heart of the class: given task description, data, model class, and loss objective → create a trained model.

1.1.1 The Bad News

But in real life, we are not given the model class or loss objective. Rather, we need to determine which model class or loss function to use.

1.1.2 The Good News

The main ideas and insights learned from this class can be generalized to real world applications.

To bridge the gap between textbook ML and real ML, we need to manipulate multiple noisy data sources and understand when and why models fail.

2 Data-Centric ML

This is a taste of research happening in ML right now! The work presented here has been conducted by Professor Melis (partly done while at Microsoft Research) **(author?)** 1.

Three Pillars of Data-Centric ML

1. Data Learning - rather than keeping the data fixed as we modify the model, we modify the data to fit a model
2. Model Interpretability - learn about the inner workings of a model and see if something in the model is behaving not how we expect it to
3. ML with Structured Data - think about generalizing ML to operate on mathematical structures, ex. graphs, biomedical applications with molecules

2.1 Data Learning

The status-quo for ML is training a model on one dataset and then using the model in a different domain, which is known as transfer learning. But, nowadays, we are using larger and larger models so adaptation of a model is very expensive. Thus, new research proposes moving away from model-centric way of doing ML. With data-centric learning, we optimize the data, and the model can be fixed or optimized. The main mathematical tool used here

is called optimal transport, which we will discuss later in lecture.

In the previous model-centric method, we end up with multiple copies of a model, which requires a lot of memory. By transforming the data to fit the model, not only do we optimize on memory but we also can impose additional constraints like ensuring privacy, size, etc.

Key takeaway: Rather than modifying the model to fit the data, the dataset itself is the optimization variable.

2.1.1 Optimal Transport

Optimal transport is a sub-field at the intersection of optimization, statistics and geometry. The goal is to transform datasets using gradient flows. Optimal Transport Dataset Distance (OTDD) measures the distance between datasets using optimal transport, considering the size, privacy concerns, and invariances. The process requires solving optimization problems over an infinite-dimensional, non-Euclidean probability space. Each point in the space is an entire dataset; as we optimize, the entire dataset is changing.

For example, we can modify an existing dataset to look like another second dataset while adding the constraint that the modified dataset is linearly separable.

Real-world application: detecting cancer based on histopathology pictures. We can leverage a model trained on a much larger dataset. Without modifying the model, we can use it on our target dataset.

Key takeaway: Models can be repurposed without modifying them.

3 How do we get there?

Now that we have looked at a taste of current ML research, how do we build up foundational knowledge to get to this point? Throughout the course, you will need two key ingredients to gain a rigorous and thorough understanding of machine learning principles: **zen and hard work**.

Zen is a depth of understanding that goes beyond the equations. The goal is to learn a foundation that can be applied to any ML situation. Obtaining zen requires spending time to grapple with the conceptual meaning behind course content. More than ever, we have models that are being tossed out into the world. After OpenAI released their model, people are now racing to release their own but there is a lack of evaluation of these models. In the class, we will learn how to evaluate models.

Throughout this course, we hope you strike a balance between strength training (like rigorous mathematics) and mental, conceptual understanding.

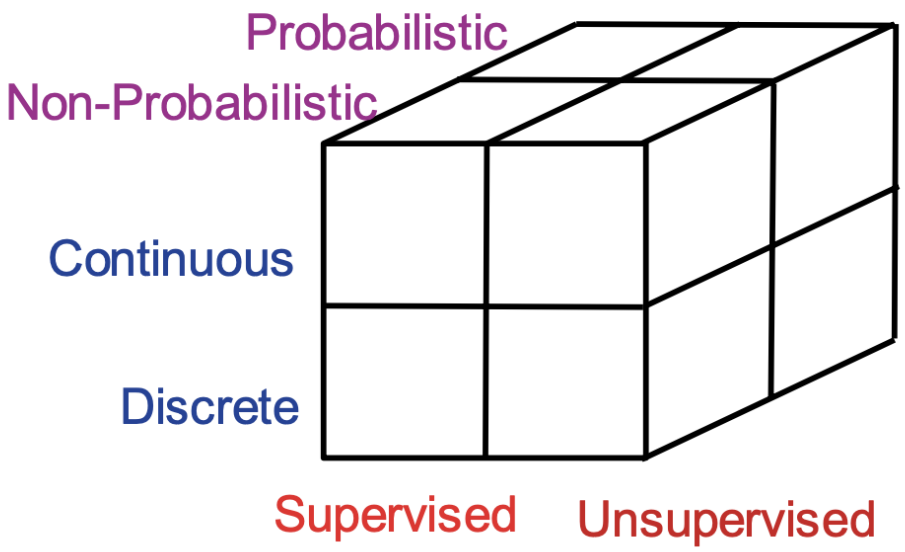
3.1 High-Level Setup

The first level of responsibility is professional ethics. Imagine building a plane. If you don't put the parts of the plane on correctly, then the plane will not fly. Similarly, if you do not build all the parts of your model correctly, then the model will not work as intended. The parts of a ML model include the data and the metric being used.

The second level of responsibility is societal ethics. Where does the data come from? Where do the tasks come from? Who designed the task and does it actually solve the problem? For example, Amazon used an AI hiring tool trained on biased data that does not hire women. Even if the first level of professional ethics was satisfied, the biased source of data led to a biased model. Another example is an ML model for ecology and birding. The model found that birds tend to birds tend to hang out new highways and streets. But the reality is that those were the locations where people tended to take photos of birds that were in the dataset. In Professor Finale's own research, she thinks about how doctors interact with machine recommendations - so ML is not just about the recommendations themselves but how people interact with them and thinking about the larger picture.

3.2 Structure of the Course

We will learn about machine learning techniques using the cube. Our cube has three axes. The first axis is the output domain. The domain of our output will be either: **discrete or continuous**. The second axis of the cube is the statistical nature of the machine learning technique. This will fall into either: **probabilistic or non-probabilistic**. Lastly, the third axis of the cube describes the type of training we will use. The type of training is: **supervised or unsupervised**.



Ex. "swipe typing" on phone:

- discrete (because the output is a word)

- probablistic
- supervised (because there is a correct answer)

Ex. Google news:

- discrete
- probablistic
- unsupervised (because the output is a summary)

3.3 Lecture Structure

We will always begin with a story/real-life example to remain grounded in real-world applications. In future lectures, there will be a Google Doc for quiet questions. Active participation and hand-raising is highly participated in lectures, but if you would rather write your question down, you can add it to the doc and a member of the course staff will read your question out loud. Lastly, classes end with concept checks.

Collaboration Policy: work together! But turn in write-ups in your own words and cite if you use AI tools.

3.4 Nonparametric Regression

3.4.1 K-Nearest Neighbors

1. First, find the k nearest inputs to x^*: $\{x_1, \ldots x_k\}$.
2. Then, predict $\hat{y^*} = \frac{1}{k} \sum_{k=1}^{K} y_k$.

The notation $\hat{y^*}$ is for representing prediction of y^*. Let's examine an example.
Say we are given the following datapoints and we want to predict the y^* for a specified x^*.

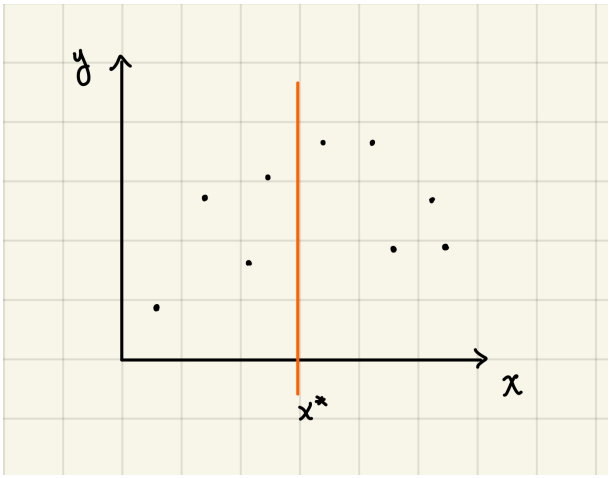


Figure 1: k-nearest neighbors

If we set $k = 2$, then the two nearest neighbors would be the points circled in green in Fig. 2. We find the average of the respective y's to produce our $\hat{y^*}$.

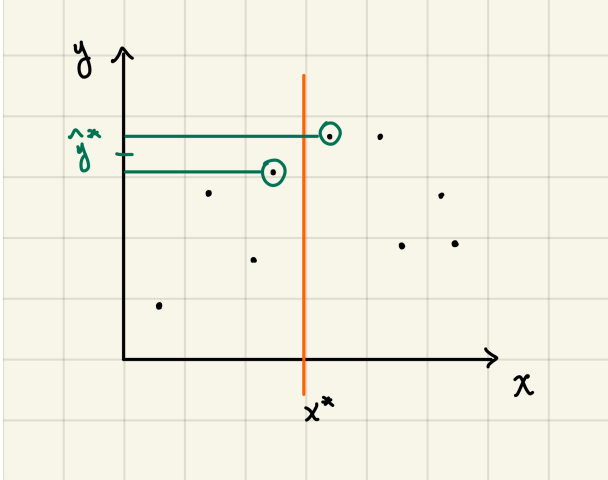


Figure 2: k = 2

3.4.2 Kernelized Regression

1. First, specify a kernel function.
2. Then, predict $\hat{y^*} = \frac{\sum_n^N K(x^*,x_n)y_n}{\sum_n^N K(x^*,x_n)}$.

The kernel function $K(x_1, x_2)$ describes the similarity between x_1 and x_2. Our prediction forumla can be rewritten as

$$\hat{y^*} = \sum_n^N \left[\frac{K(x^*, x_n)}{\sum_{n'}^N K(x^*, x_{n'})} \right] y_n.$$

Therefore, the prediction is a weighted average of all y_n's in the dataset. Thus, the predicted y value cannot exceed the biggest y_n and the predicted y value cannot be less than the smallest y_n.

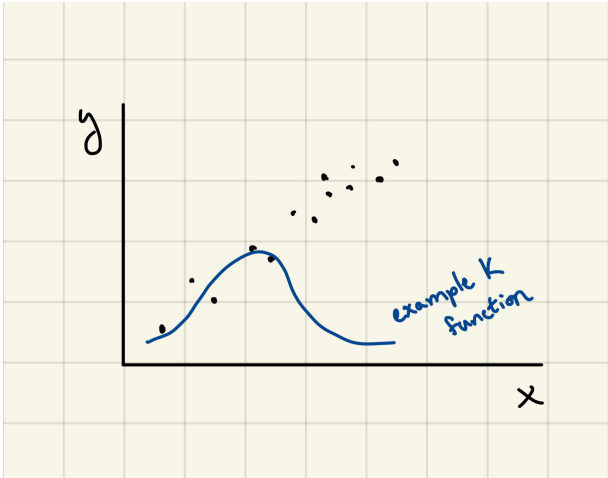


Figure 3: kernel regression

If the blue line is the kernel, then the points further from the line will not be weighted

very much. Kernel regression ensures that if a x_n is not really relevent, then the point is not weighted much. The points that are nearby get weighted more.

Q. Should we use KNN or kernelized regression?
With KNN, the predicted $\hat{y}$ comes from the nearest cluster. We can think about KNN has snapping us to the nearest cluster. With kernelization, we do not care about the far away points and these far away points play less of a role in the prediction.

Q. When is KNN better than kernelized regression?
It may seem that kernelized regression is just a better version of KNN. So, what upperhand does KNN have? With kernelized regression, for every prediction, we need to run through the entire datapoint which take a lot of time. Thus, one trade-off here is computational efficiency. KNN is more computationally efficient.

3.4.3 More ways to think about KNN vs Kernelized Regression

I want 10 friends.

vs.

I want friends within a 10 feet radius of me.

KNN is like specifying the number of friends you want, while kernelized regression is like specifying the constraint rather than the number of friends.

I value each of the 10 friends equally.

vs.

I value friends who are closer to me in distance.

With KNN, we find the average of the y_n's so every datapoint is valued equally. With kernelized regression, we have a weighted average. Thus, datapoints that are more similar to the target x^* based on the kernel function are valued more.

References

[1] J. H. L. M. N. T. Jiaojiao Fan, Nicolò Fusi.