Mathematical Foundations of Machine Learning: Introduction

What are we "learning" in Machine Learning?

This is a hard question to which we can only give a somewhat fuzzy, qualitative answer. But at a high enough level of abstraction, I think that there are two answers[1]:

1. We are learning an *algorithm* that solves some kind of inference problem.
2. We are learning a *model* for the data set.

These answers are so abstract that they are probably completely unsatisfying. But let's (start to) clear things up by looking at some particular examples of "inference" and "modeling" problems.

[1] These two categories roughly correspond to what are called *supervised* and *unsupervised* learning in the literature.

Inference

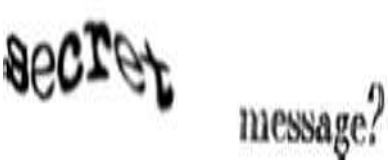
Loosely speaking, inference problems take in data, then output some kind of decision or estimate.

Examples:

- Does this image have a tree in it?



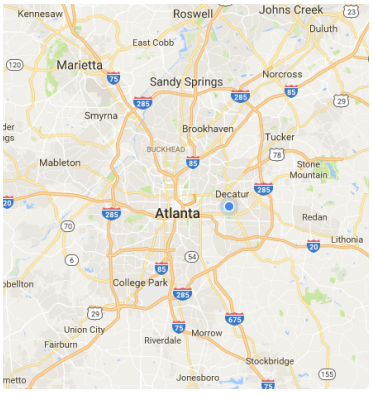

- What words are in this picture?



The three revolutions in parametric statistical inference are due to Laplace [148], Gauss and Laplace (1809–1811), and Fisher [67].

We use $p(\cdot)$ generically to denote a frequency function, continuous or discontinuous, and $p(x|\theta)$ to denote a statistical model defined on a given sample space and parameter space. Let $\underline{x} = (x_1, \ldots, x_n)$ denote a sample of n independent observations. From the model we can find the sampling distribution

- If I tell you the current temperature in Sandy Springs, Marietta, Mableton, College Park, and Decatur, can you tell me the temperature in downtown Atlanta?



- If I tell you a person's credit score, current income, and savings,

can you predict whether they will default within 7 years on a mortgage of $2000/month?

- If I give you a recording of somebody speaking, can you produce text of what they are saying?
- If I give you a video of a drone moving along with the signal coming out of the remote control used to fly it, can you discover the differential equations that govern its motion?

Each of these inference algorithms takes some piece of data, e.g. an image (a collection of pixels arranged on a grid, each with an associated number indicating the color) as an input. They can output either a hard decision ("this image does not have a tree in it" or "the drone is 15.2 meters away from the sensor") or a **probability distribution** over the range of possible outcomes ("there is a 44% chance this image contains a tree" or "the probability density function for the distance X of the drone to the sensor is $f_X(x) = e^{-2|x-15.2|}$").

Now, back to the question we started this discussion with:
What does a machine learning algorithm do?

Machine learning algorithms are **not** algorithms for performing inference. Rather, they are algorithms for *learning* an effective inference algorithm from examples. An inference algorithm takes a piece of data and outputs a decision (or a probability distribution over the decision space). A supervised machine learning algorithm takes many pieces of data, each *pre-labeled* with a decision, and outputs a **mapping** from data space to decision space[2].

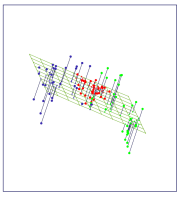
[2](Or from data space to the space of probability distributions over the range of possible decisions.)

Modeling

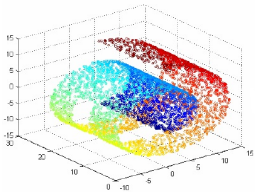
A second type of problem associated with the words "machine learning" might be roughly described as:
Given a data set, how can I succinctly describe it (in a quantitative, mathematical manner)?

Most models can be broken into two categories:

1. **Geometric models**. The general problem is that we have example data points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathbb{R}^D$ and we want to find some kind of geometric[3] structure that (approximately) describes them. Here are examples: given a set of vectors , what (low dimensional) subspace comes closest to containing them? What about a low-dimensional manifold?



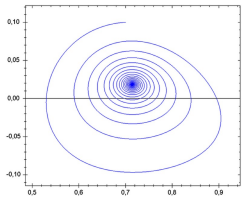
data points near a subspace



data points near a manifold

   Here is another: given samples of a trajectory $\boldsymbol{x}(t_1), \boldsymbol{x}(t_2), \ldots, \boldsymbol{x}(t_N)$, find $\boldsymbol{A}$ such that the solution to $\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t)$ comes as close as possible to matching these points. This also might be generalized to the more general problem of finding $F : \mathbb{R}^N \to \mathbb{R}^N$ of a certain form such that $\dot{\boldsymbol{x}}(t) = F(\boldsymbol{x}(t))$.

[3] All of the things discussed here could also be described as *algebraic* models; we could pose these as finding a system of equations that describes the data points. Geometry just makes for better visualization.



trajectory of a dynamical system

There are many, many variations on this theme. Of course, one of the keys to finding a good geometric model is to choose a good class of structures.

2. **Probabilistic models**. The basic task here is to find a *probability distribution* that describes the $\boldsymbol{x}_n$. The classical name for this problem is *density estimation* — given samples of a random variable, estimate its probability density function (pdf). This gets extremely tricky in high dimensions (large values of D) or when there are dependencies between the $\boldsymbol{x}_n$. Key to solving these problems is choosing the right way to describe your probability model.

In both cases above, having a concise model can go a tremendous way towards simplifying the data — each data point in $\mathbb{R}^D$ might be described in many fewer than D parameters. As a rule, if you have a simple, accurate model, this is tremendously helping in solving inference problems, as there are fewer parameters to consider and/or estimate.

The categories can also overlap with or complement each other. It is often the case that the same model can be interpreted as a geometric model or a probabilistic model.

This course

This course is not so much about actual machine learning algorithms. Rather, it will focus on the basic mathematical concepts on which these algorithms are built. In particular, to really understand anything about ML, you need to have a very good grasp of **linear algebra** and **probabilistic inference** (i.e. the mathematical theory of probability and how to use it).

All of you have probably (hopefully) had some exposure to these topics in the past. We will cover the key parts of these branches of applied mathematics in depth and in the context of machine learning. More specifically, the topics in this course can be broken down into four basic subject areas:

1. **Representations** for data and operators that map data to decisions/estimates. We will start with a thorough discussion of linear representations; these are important/useful by themselves, and also are used as building-blocks for nonlinear representations. Here is where we will need a lot of linear algebra and its extension (called *functional analysis* to infinite dimensions).
2. **Estimation**. What does it mean to estimate a parameter for/from a data set? We will try to put this question on as firm a mathematical footing as we can using the language of statistics.
3. **Modeling**. See above.
4. **Computing**. Finally, we will get a look at how computations for solving problems arising in ML are actually done. We will look at some basic algorithms from optimization, and some matrix factorization techniques from numerical linear algebra.

Although the material in this course is foundational, I expect that you will find it rigorous and challenging. We will not shy away at all from mathematical arguments in lecture or in the homework assignments.