Chapter 1

Introduction / Regression

Slides (pdf) Video (Panopto)

1.1 What is Machine Learning?

Definition 1.0.1: Machine Learning

Machine learning is the study of algorithms that can improve their performance at some task with experience.

From the Netflix series recommended to you, to Siri's assistance, to the GPS system on your phone, Machine learning is overwhelmingly prevalent in the majority of the services presented by the modern technological world. This is because with machine learning, we can build models that analyze and automate processes in the real world. The key idea behind a machine learning algorithm is to intake data from the real world and improve its model of the real world accordingly.

By modeling the real world as accurately as possible, machine learning aims to predict or categorize the unseen. It does so by exploiting relationships in data as we will explore later on in this chapter.

Example(s)

A common example of the use of machine learning is Netflix's series recommendations. If your Netflix account has been designated as a consistent watcher of a specific show, you will be recommended shows that Netflix thinks you will enjoy in the future. How does this happen?

Figure 1.0.1: With enough data, a machine learning algorithm can make decisions that seem intelligent.

Sometimes, such relationships in real world data are not easily conceivable to the human mind, which is what gives machine learning algorithms the illusion of being "intelligent", especially to those only familiar with AI in the context of science fiction. However, we invite you to entertain the thought that perhaps more exciting than a science-fiction cyborg is the idea that massive data computations are enough to present a sense of genuine intelligence like humans. Whether it is generating speech or classifying animals, such tasks humans for so long thought were only possible by their own brains became a matter of mathematical modeling.

Let us consider the Netflix example above. If a friend really enjoys "The Office", you might be inclined to recommend that they also watch "Parks and Rec" due to the similar style of comedy in the shows. This

is a simple prediction analysis you could have done in your own mind–no machine learning model required here. But what if you wanted to do the same for lots of people who have different show preferences than you? In our current digital era we now have massive amounts of data with thousands of variables–some of those variables interact with each other, and we don't exactly know which ones, or how, or why. This is where machine learning comes in. The applications of this technology have had profound impact on various arenas like medicine, language processing, robotics, and most importantly, predicting the winning contestant of The Bachelor!

1.2 Types of Machine Learning

There are different types of machine learning, defined by how the "learning" happens. The main three are:

Definition 1.0.2: Supervised Learning

Supervised Learning is the most basic type of machine learning. A model is presented with a dataset that has labeled input and output pairs.

This learning is "supervised" by humans with guidance on what the input is exactly and what the output is exactly. The model will improve its performance by training on these explicitly labeled pairs. A classic example is image classification, like sorting animals, where a bunch of labeled images are fed into a model so the model has lots of pre-labeled examples of "cats" and "dogs".

Definition 1.0.3: Unsupervised Learning

Unsupervised Learning analyzes an unlabeled dataset. There is no specified input or output, just a big mesh of data. Thus, it is up to the model to categorize and find patterns within the data.

Clustering things that are similar with each other is a popular technique in unsupervised machine learning. Image classification can be unsupervised too: The model is fed a bunch of pictures of cats and dogs but none of these images are labeled. Because all cats kind of look like each other (they are hence structurally similar in data), and all dogs kind of look like each other, but cats and dogs look different from each other relatively in the dataset, a model can assert that these are probably two distinct categories.

Definition 1.0.4: Reinforcement Learning

Reinforcement Learning is a type of machine learning that devises a method to maximize desired behavior in a model by using a reward system, and penalizes undesired behavior.

What makes reinforcement learning unique is that its reward system behaves as a separate agent so that short-term successes aren't stalled upon, and the model prioritizes long-term success. An example of reinforcement learning is in Pac-Man, where long-term planning coded into the algorithm can achieve superhuman performance. This type of learning is very common in gaming, though we will not expand upon it further in this text as its usage is limited.

For now, we will focus on Supervised learning, so don't worry if the other two don't make sense right now.

1.3 Main Applications of Machine Learning

The rest of this text will be organized in a way that follows the 5 main kinds of machine learning application:

1. Regression
2. Classification
3. Document Retrieval Clustering
4. Recommender Systems
5. Deep Learning

1.4 Case Study 1: Regression

1.4.1 What is Regression?

Definition 1.0.5: Regression

Regression is a supervised learning technique used to predict a dependent variable based on one or more independent variables by estimating a relationship between them.

Regression is the first application of machine learning that we will discuss here. Regression is applicable when we want to know the answer to one thing, but we have clues coming from multiple sources. An example of this is house prices. What can you use to determine the price of a house? Its square footage? The neighborhood it's in? What about the view? What about when it was built? There are many independent variables here, and they may all be important for predicting our dependent variable, the price.

1.4.2 How Does Regression Work?

Let's first go over the general premise of how regression works. Consider our dataset of x and y pairs. The x's are training examples, or the input. If we are going off the house prices example, you can think of each x as a list of all the information about one specific house that helps determine its price, like the square footage and the year it was built. The y's are the values of our output. In this case each y is a price for one specific house. We will say that we have n number of x's and y's like so:

$$(x, y)^{(1)}, (x, y)^{(2)}, ..., (x, y)^{(n)} \tag{1.0.1}$$

So, $x^{(1)}$ is a list of facts about house 1, and $y^{(1)}$ is its price. If we let i be equal to some index number, then what we mean for any house i is that it has information $x^{(i)}$ and price $y^{(i)}$. Given all our x and y pairs, we want to build a predictor function that learns how to map $x^{(i)}$ to $y^{(i)}$. We would ideally like to be able to predict a house price (output) from a house's information (input).

It is important to note that a single x is a list. It includes various pieces of information about a house like its square footage and its year built. We will call these pieces of information "features".

Definition 1.0.6: Feature

A **feature** in machine learning is a single variable that is part of the input. Our input "house information", has features "square footage" and "year built".

Notation-wise, we will let capital X denote a single feature and d be the number of features we have in a single input $x^{(i)}$:

$$x^{(i)} = X_1^{(i)}, X_2^{(i)}, ..., X_d^{(i)} \tag{1.0.2}$$

These features are important to label distinctly because some features in an input may be more important for predicting the output than others.

For simplicity's sake, we will consider an input of one feature, square footage. So, let $x^{(i)}$ be a list of size $d = 1$ containing only square footage as input. Observe the pairings below:

$$\begin{aligned}
(x_1, y_1) &= (\,2318\, sq.ft.\,,\, \$\,315k) \\
(x_2, y_2) &= (\,1985\, sq.ft.\,,\, \$\,295k) \\
(x_3, y_3) &= (\,2861\, sq.ft.\,,\, \$\,370k) \\
&\vdots \qquad\qquad \vdots \\
(x_n, y_n) &= (\,2055\, sq.ft.\,,\, \$\,320k)
\end{aligned}$$

Figure 1.0.2: There are n number of x and y pairs, with x being the square footage and y being the price of each house.

To use regression, we have to assume that there exists a relationship between x and y. This means that we have to assume that somehow, for some reason, the square footage has an effect on the house price. Thus, we are making the assumption that:

$$y \approx f(x) \tag{1.0.3}$$

where $y \in \mathbb{R}$ and $x \in \mathbb{R}^d$. Remember that $\mathbb{R}$ is the set of all real numbers. So all this notation means is that y is a real number, and x is d number of real numbers (since our input can be made up of multiple features). Since we assumed that y has a relationship with x, we have asserted that some function applied to x should give us y. We call this the true function.

Definition 1.0.7: True Function

The **true function** in regression is the function we assume exists in the real world that maps out the input to the output. It is denoted as f.

Finally, remember that in this equation, x will always be an independent variable, or an input, and y will always be a dependent variable, the output, or the final thing we are trying to predict.

1.4.3 Defining the Model

After assuming that true function f exists, we now try to define a model that can be as close as possible to f since f is unknown. The model that we define ourselves to match close enough with f is denoted as $\hat{f}$.

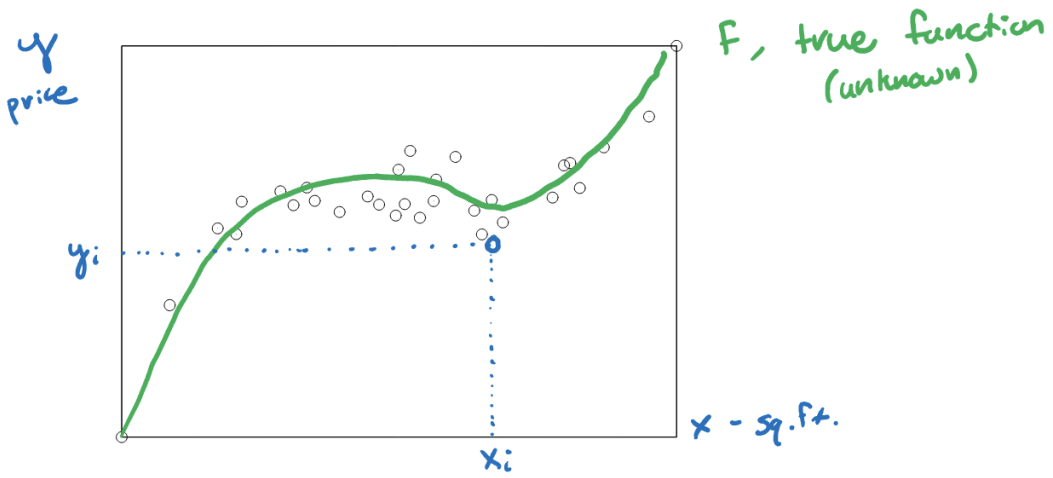

Figure 1.0.3: The dataset is plotted on a graph where the x-axis is square footages and the y-axis is house prices.

Consider the plot in Figure 1.1.3.

Suppose the true function f is the polynomial function plotted above, because for whatever reason, house prices correlate with square footage in that exact pattern in the real world. We can eye this plot and intuitively draw a curve ourselves that could summarize the data pretty well. However, if our input was more than one dimension, this could be very difficult to do and a regression algorithm can determine a better fitting function in any case. A regression algorithm will thus conduct a series of computations to determine which $\hat{f}$ best fits the data. We will discuss how regression algorithms determine what is "best" in the next section.

1.5 Linear Regression

Though the learning process for all types of regression is more or less the same, we will start with linear regression for simplicity's sake. If we are doing linear regression, we assert that the relationship between x and y is linear:

$$y^i \approx f(x^i) = wx^i + b \tag{1.0.4}$$

Hence our $\hat{f}$ will also be linear:

$$\hat{y^i} = f(\hat{x^i}) = \hat{w}x^i + \hat{b} \tag{1.0.5}$$

Note here the variables that have hats above them. Only x^i doesn't have a hat above it because it is the true input. But the other components of our model $\hat{w}$ and $\hat{b}$ are mere estimates of the true w and b, and the final output $\hat{y^i}$ is a prediction, not the given y-value from our dataset.

Now, our goal is to find the most accurate values for $\hat{w}$ and $\hat{b}$.

The big picture outline for the regression algorithm is as follows:

1. Start with a random line to fit the data. This means try out a random value for $\hat{w}$ and $\hat{b}$.

2. Calculate the error from this line to know how bad it is.
3. Try another random line.
4. Calculate the error again. If this line is worse than the one we just tried, try to go back closer to your older values for $\hat{w}$ and $\hat{b}$. If it's better, just keep going in this direction!
5. Repeat until your error can't get any smaller.

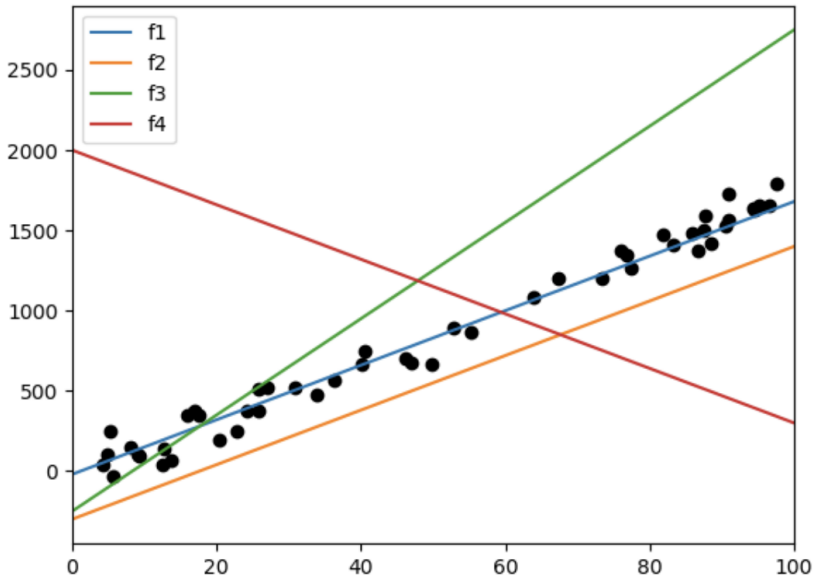

Figure 1.0.4: Regression starts with a random line and then tries to gradually make it better and better based on error calculation.

There are many ways to calculate error but the most popular metric and the one we will use for regression is Mean-Squared Error (MSE).

Definition 1.0.8: Mean-Squared Error

Mean-Squared Error (MSE) is a type of error calculation. To calculate MSE you sum up all the differences between your true y values from each x and your predicted $\hat{y}$ values from each x and square them. Then, you divide this number by your dataset size n:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y^i - \hat{y^i})^2 \tag{1.0.6}$$

A smaller MSE indicates a smaller loss which indicates a better model $\hat{f}$. So, our goal is to *minimize* MSE. Since in the MSE equation the changing values are w and b, we can define MSE as a function with two parameters. Mathematically, we solve for:

$$\hat{w}, \hat{b} = argmin MSE(w, b) \tag{1.0.7}$$

We minimize MSE to get the best $\hat{f}$ through a process called Gradient Descent.

Definition 1.0.9: Gradient Descent

Gradient Descent is an optimization algorithm aimed at finding a local minimum in an error graph. It uses calculus to utilize the gradient (slope) of a given error.

Gradient Descent uses the slope of an error graph to determine which direction signals a higher error versus a lower error. You can think of it like rolling a ball down a hill.

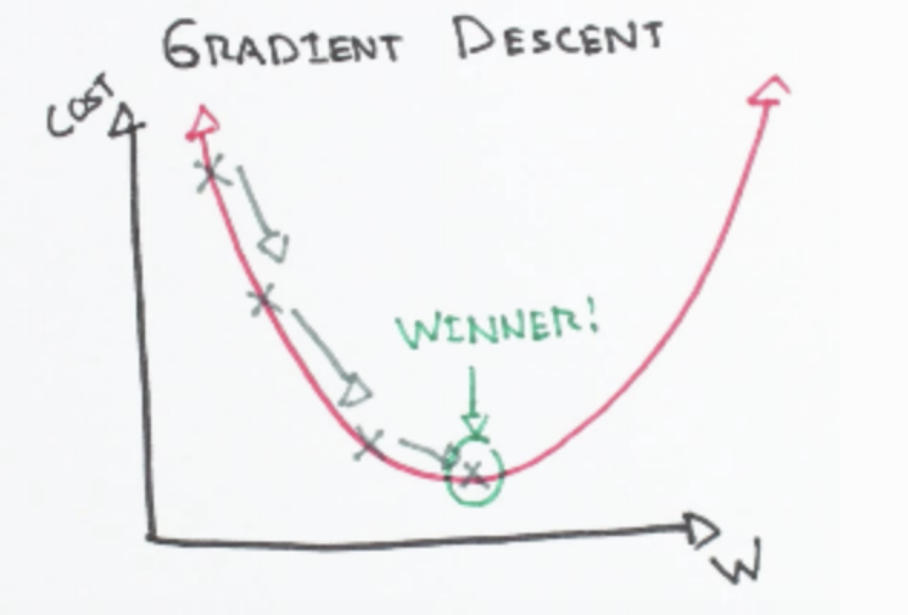

Figure 1.0.5: You can think of Gradient Descent as rolling a ball down a hill.