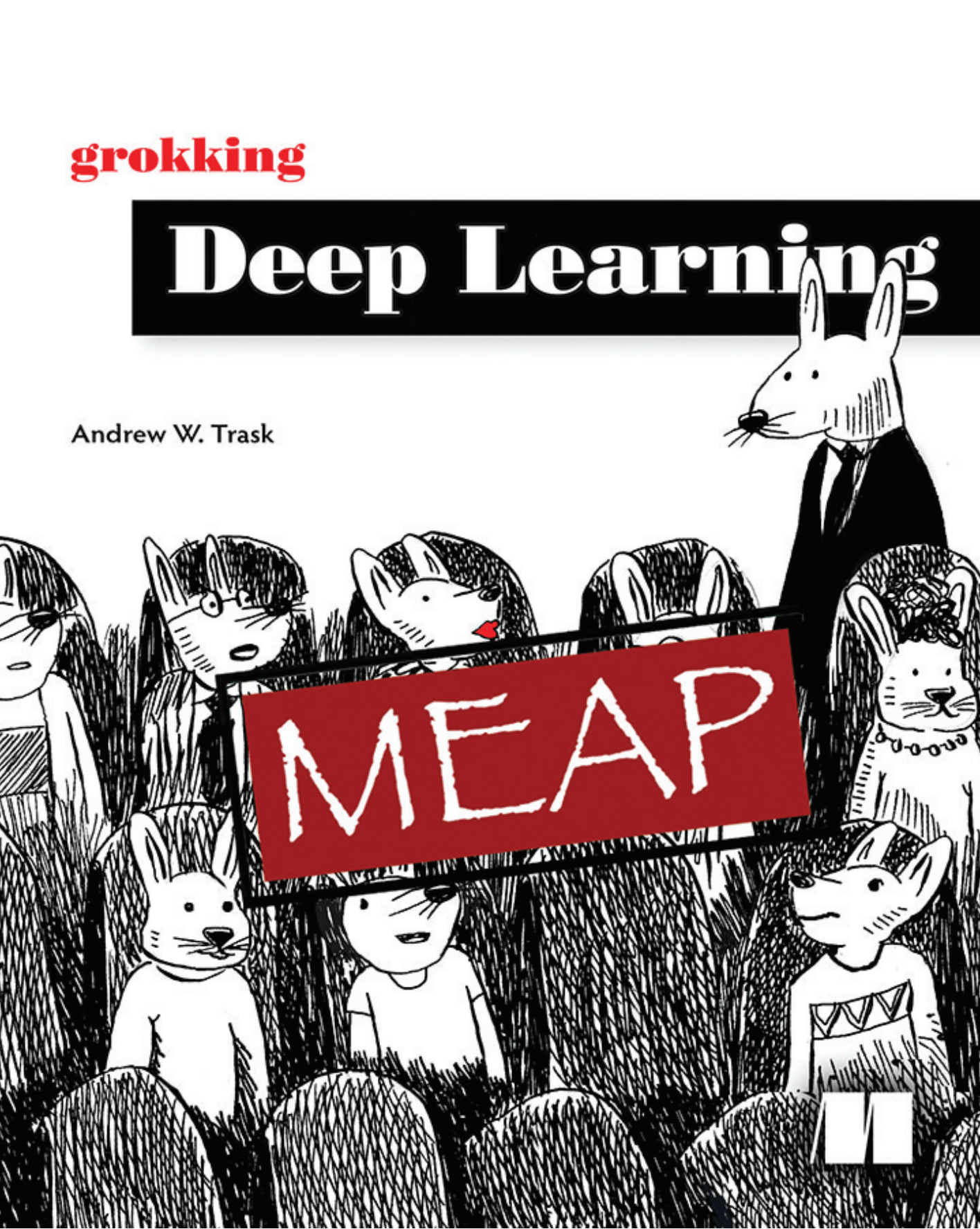
grokking

Deep Learning

Andrew W. Trask



MEAP Edition
Manning Early Access Program
Grokking Deep Learning
Version 10

For more information on this and other Manning titles go to
www.manning.com

welcome

Thank you so much for purchasing *Grokking Deep Learning*. This book will teach you the fundamentals of Deep Learning from an intuitive perspective, so that you can understand *how machines learn using Deep Learning*. This book is not focused on learning a framework such as Torch, TensorFlow, or Keras. Instead, it is focused on teaching you the Deep Learning *methods* behind well known frameworks. Everything will be built from scratch using only Python and numpy (a matrix library). In this way, you will understand every detail that goes into training a neural network, not just how to use a code library. You should consider this book a pre-requisite to mastering one of the major frameworks.

There are many other resources for learning Deep Learning. I'm glad that you came to this one, as I have intentionally written it with what I believe is the lowest barrier to entry possible. No knowledge of Linear Algebra, Calculus, Convex Optimization, or even Machine Learning is assumed. Everything from these subjects that is necessary to understand Deep Learning will be explained as we go. If you have passed high school mathematics and hacked around in Python, you're ready for this book, and when you complete this book, you will be ready to master a major deep learning framework like Torch, TensorFlow, or Keras.

Finally, as this is the MEAP, if there is any point in these first few chapters that something does not make sense, it is my hope that you would tweet your questions to me @iamtrask. I would be happy to help, and more importantly, I want to know if any section of the book is not fulfilling my personal commitment to the lowest barrier to entry possible so that I can adjust it for the final published work. Please, don't hesitate to reach out if you have questions.

These first few chapters will be walking you from a general introduction to Deep Learning all the way through to building your first working neural network. In these chapters, you will get a firm grasp on the philosophy behind how machines can learn the world you present to them. It's an exciting thing to see happen, and perhaps even more exciting, you will understand every nook and cranny of what makes this learning possible.

It is an honor to have your time and attention.

—Andrew Trask

brief contents

Fundamental Concepts
How do machines learn?

2

IN THIS CHAPTER

What are Deep Learning, Machine Learning, and Artificial Intelligence?

What is a Parametric Model?

What is a Non-Parametric Model?

What is Supervised Learning?

What is Unsupervised Learning?

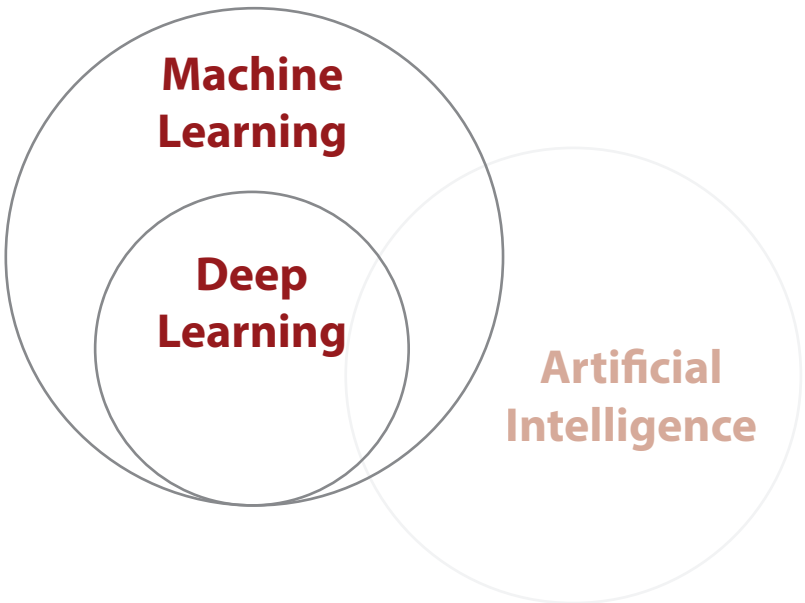
How can machines learn?

> "Machine Learning will cause every successful IPO win in 5 years"
>
> - Eric Schmidt (Google Chairman, 2016)

What is Deep Learning?

Deep Learning is a subfield of methods for Machine Learning

Deep Learning is a subset of Machine Learning, which is a field dedicated to the study and development of machines that can learn (sometimes with the goal of eventually attaining General Artificial Intelligence). In industry, Deep Learning is used to solve practical tasks in a variety of fields such as Computer Vision (Image), Natural Language Processing (Text), and Automatic Speech Recognition (Audio). In short, Deep Learning is a subset of *methods* in the Machine Learning toolbox, primarily leveraging **Artificial Neural Networks**, which are a class of algorithm loosely inspired by the human brain.



Notice in the figure above that not all of Deep Learning is focused around pursuing Generalized Artificial Intelligence (i.e. sentient machines like in the movies). In fact, many applications of this technology are applied to solve a wide variety of problems in industry. This book seeks to focus on learning the fundamentals of Deep Learning behind both cutting edge research and industry, helping to prepare you for either.

What is Machine Learning?

> "A field of study that gives computers the ability to learn without being explicitly programmed"
>
> - Arthur Samuel

Given that Deep Learning is a subset of Machine Learning, what is Machine Learning? Most generally, it is what its name implies. Machine Learning is a subfield of Computer Science wherein *machines learn* to perform tasks for which they were *not explicitly programmed*. In short, machines observe a pattern and attempt to imitate it in some way which can be either direct or indirect **imitation**.

machine learning ~= monkey see monkey do

I mention direct and indirect imitation as a parallel to the two main types of machine learning, **supervised** machine learning and **unsupervised** machine learning. Supervised machine learning is the direct imitation of a pattern between two datasets. It is always attempting to take an input dataset and transform it into an output dataset. This can be an incredibly powerful and useful capability. Consider the following examples: (**input** datasets in bold and *output* datasets in italic)

- Using the **pixels** of an image to detect the *presence or absence of a cat*
- Using the **movies you've liked** to predict *movies you may like*
- Using someone's **words** to predict whether they are *happy* or *sad.*
- Using weather sensor **data** to predict the *probability of rain.*
- Using car engine **sensors** to predict the optimal tuning *settings.*
- Using news **data** to predict tomorrow's stock *price*.
- Using an input **number** to predict a *number* double its size.
- Using a raw **audio file** to predict a *transcript* of the audio.

These are all supervised machine learning tasks. In all cases the machine learning algorithm is attempting to imitate the pattern between the two datasets in such a way that it can **use one dataset to predict the other**. For any example above, imagine if you had the power to predict the *output* dataset given only the **input** dataset. This would be profound.

Supervised Machine Learning

Supervised Learning transforms one dataset into another.

Supervised Learning is a method for transforming one dataset into another. For example, if we had a dataset of "Monday Stock Prices" which recorded the price of every stock on every Monday for the past 10 years, and a second dataset of "Tuesday Stock Prices" recorded over the same time period, a supervised learning algorithm might try to use one to predict the other.

If we successfully trained our supervised machine learning algorithm on 10 years of Mondays and Tuesdays, then we could predict the stock price of any Tuesday in the future given the stock price on the immediately preceeding Monday. I encourage you to stop and consider this for a moment.

Supervised Machine Learning is the bread and butter of applied Artificial Intelligence (i.e. "Narrow AI"). It is useful for taking *what we do know* as input and quickly transforming it into **what we want to know**. This allows supervised machine learning algorithms to extend human intelligence and capabilities in a seemingly endless number of ways.

The majority of work leveraging machine learning results in the training of a supervised classifier of some kind. Even unsupervised machine learning (which we will learn more about in a second) is typically done to aid in the development of an accurate supervised machine learning algorithm.

For the rest of this book, we will be creating algorithms that can take input data that is observable, recordable, and by extension **knowable** and transform it into valuable output data that requires logical analysis. This is the power of supervised machine learning.

Unsupervised Machine Learning

Unsupervised Learning groups your data.

Unsupervised learning shares a property in common with supervised learning. It transforms one dataset into another. However, the dataset that it transforms into is **not previously known or understood**. Unlike supervised learning, there is no "right answer" that we're trying to get the model to duplicate. We just tell an unsupervised algorithm to "find patterns in this data and tell me about them".

For example, *clustering a dataset into groups* is a type of unsupervised learning. "Clustering" transforms your sequence of *datapoints* into a sequence of *cluster labels*. If it learns 10 clusters, it's common for these labels to be the numbers 1-10. Each datapoint will get assigned to a number based on which cluster it is in. Thus, your dataset turns from a bunch of datapoints into a bunch of labels. Why are the labels numbers? The algorithm doesn't tell us what the clusters are. How could it know? It just says "Hey scientist!... I found some structure. It looks like there are some groups in your data. Here they are!".

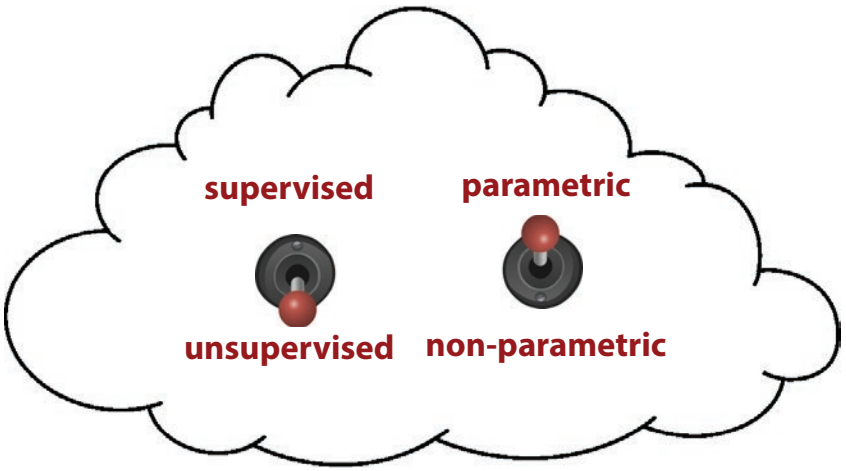
I have good news! This idea of clustering is something you can reliably hold onto in your mind as the definition of unsupervised learning. Even though there are many forms of unsupervised learning, *all forms of unsupervised learning can be viewed as a form of clustering*. We will discover more on this later in the book.

Check out the example above. Even though the algorithm didn't tell us what the clusters are named, can you figure out how it clustered the words? (1 == cute & 2 == delicious) Later, we will unpack how other forms of unsupervsied learning are also just a form of clustering and why these clusters are useful for supervised learning.

Parametric vs Non-Parametric Learning

Oversimplified: Trial and error learning versus counting and probability

The last two pages divided all of our machine learning algorithms into two groups, supervised and unsupervised. Now, we're going to discuss another way to divide the same machine learning algorithms into two groups, parametric and non-parametric. So, if we think about our little machine learning cloud, it has two settings:



As you can see, we really have four different types of algorithm to choose from. An algorithm is either unsupervised or supervised and it is either parametric or non-parametric. Whereas the previous section on supervision is really about the **type of pattern** being learned, parametricism is about the way the learning is **stored** and often by extension, the **method for learning**. First, let's look at the formal definition for parametricism vs non-parametricism. For the record, there is still some debate around the exact difference.

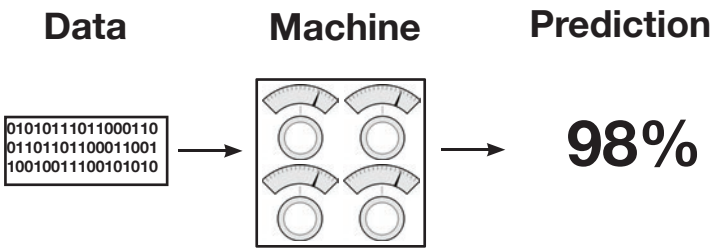
> A parametric model is characterized by having a *fixed* number of parameters whereas a non-parametric model's number of parameters is *infinite* (determined by data).

As an example, let's say the problem was to fit a square peg into the correct (square) hole. Some humans (such as babies) just jam it into all the holes until it fits somewhere (parametric). A teenager, however, might just count the number of sides (4) and then search for the hole with an equal number (non-parametric). Parametric models tend to use "trial and error", where non-parametric models tend to "count". Let's look closer.

Supervised Parametric Learning

Oversimplified: Trial and error learning using knobs

Supervised parametric learning machines are machines with a fixed number of knobs (that's the parametric part), wherein learning occurs by turning the knobs. **Input data** comes in, is processed based on the angle of the knobs, and is transformed into a *prediction*.



Learning is accomplished by turning the knobs to different angles. If we're trying to predict the probability that the Red Socks will win the World Series, then this model would first take data (such as sports stats like win/loss record or average number of toes) and make a prediction (such as 98% chance). Next, the model would observe whether or not the Red Socks actually won. After it knew whether they won, our learning algorithm would **update the knobs** to make a more accurate prediction the next time it sees the **same/similar input data.**

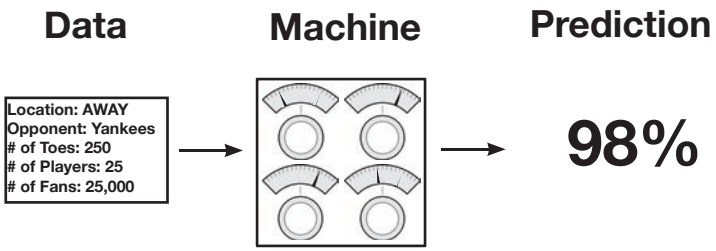
Perhaps it would "turn up" the "win/loss record" knob if the team's win/loss record was a good predictor. Inversely, it might turn down the "average number of toes" knob if that datapoint wasn't a good predictor. This is how parametric models learn!

Note that the entirety of what the model has learned can be captured in the positions of the knobs at any given time. One can also think of this type of learning model as a search algorithm. We are "searching" for the appropriate knob configuration by trying configurations, adjusting them, and retrying.

Note further that the notion of trial and error isn't the formal definition, but it is a very common (with exceptions) property to parametric models. When there is an arbitrary (but fixed) number of nobs to turn, then it requires some level of searching to find the optimal configuration. This is in contrast to non-parametric learning, which is often "count" based and (more or less) "adds new knobs" when it finds something new to count. Let's break down supervised parametric learning into its three steps.

Step 1: Predict

To illustrate supervised parametric learning, let's continue with our sports analogy where we're trying to predict whether or not the Red Socks will win the World Series. The first step, as mentioned, is to gather sports statistics, send them through our machine, and make a prediction on the probability that the Red Socks will win.



Step 2: Compare to Truth Pattern

The second step is to compare the prediction (98%) with the pattern that we care about (whether the Red Socks won). Sadly, they lost, so our comparison is:

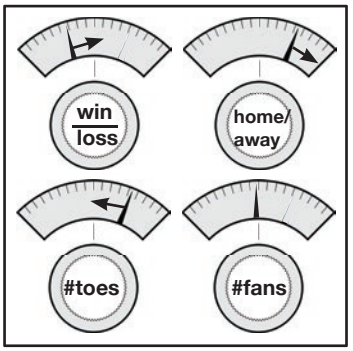
Pred: **98%** > Truth: **0%**

This step simply recognizes that if our model had predicted 0%, it would have perfectly predicted the upcoming loss of the team. We want our machine to be accurate, which takes us to Step 3.

Step 3: Learn the Pattern

This step adjusts the knobs by studying both how **much** the model missed (98%) and what the input data **was** (sports stats) at the time of prediction. It then turns the knobs to make a more accurate prediction given the input data. In theory, the next time it saw the same sports stats, the prediction would be lower than 98%. Note that each knob represents the *prediction's sensitivity to different types of input data*. That's what we're changing when we "learn".

Adjusting Sensitivity
By Turning Knobs



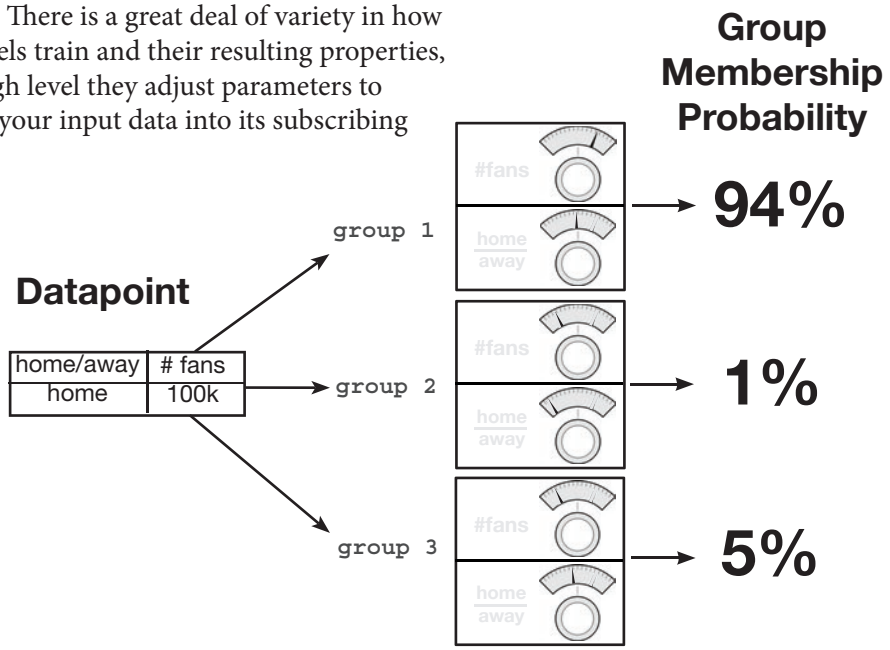
Unsupervised Parametric Learning

Unsupervised parametric learning leverages a very similar approach. Let's walk through the steps at a high level. Remember that unsupervised learning is all about grouping your data. Unsupervised *parametric* learning uses knobs to group your data. However, in this case, it usually has several knobs for each group, each that maps your input data's affinity to that particular group (with exception and nuance, this is a high level description). Let's look at an example where we assume we want to divide our data into three groups.

home/away	# fans
home	**100k**
away	50k
home	**100k**
home	**99k**
away	50k
away	*10k*
away	*11k*

In the dataset on the right, I have identified three clusters in the data that we might want our parametric model to find. I identified them via formatting as **group 1**, group 2, and *group 3*. Let's propagate our first datapoint through a trained unsupervised model below. Notice that it maps most strongly to **group one**.

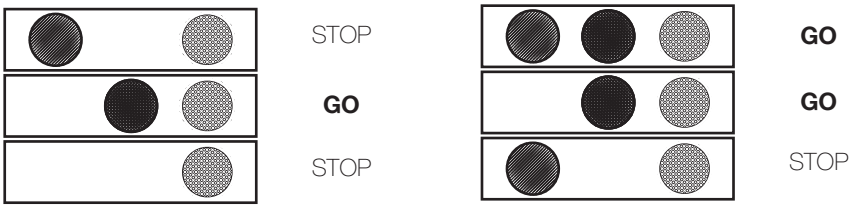
Each group's machine attempts to transform the input data to a number between 0 and 1, telling us the *probability that the input data is a member of that group*. There is a great deal of variety in how these models train and their resulting properties, but at a high level they adjust parameters to transform your input data into its subscribing group(s).



Non-Parametric Learning

Oversimplified: Counting based methods

Non-Parametric learning is a class of algorithm wherein the number of parameters is based on data (instead of pre-defined). This lends itself to methods that generally count in one way or another, thus increasing the number of parameters based on the number of items being counted within the data. In the supervised setting, for example, a non-parametric model might count the number of times a particular color of streetlight causes cars to "go". After counting only a few examples, this model would then be able to predict that *middle* lights always (100%) cause cars to "go" and *right* lights only sometimes (50%) cause cars to "go".



Notice that this model would have 3 parameters, 3 counts indicating the number of times each colored light turned on and cars "go" (perhaps divided by the number of total observations). If there had been 5 lights, there would have been 5 counts (5 parameters). What makes this simple model *non-parametric* is this trait wherein the number of parameters changes based on the data (in this case, the number of lights). This is in contrast to parametric models, which start with a set number of parameters and, more importantly, can have more or less parameters purely at the disgretion of the scientist training the model (irrespective of data).

A close eye might question this idea. Our parametric model from before seemed to have a knob for each input datapoint. In fact, most parametric models still have to have some sort of *input* that is based on the number of classes in the data. Thus you can see that there is a bit of *grayness* between parametric and non-parametric algorithms. Even parametric algorithms still are somewhat influenced by the number of classes in the data, even if they are not explicitly counting patterns.

Perhaps this also illumates that *parameters* is actually a very generic term, referring only to the set of numbers used to model a pattern (without any limitation on how those numbers are used). Counts are parameters. Weights are parameters. Normalized variants of counts or weights are parameters. Correlation coefficients can be parameters. It's simply referring to the set of numbers used to model a pattern. As it happens, Deep Learning is a class of parametric models. We won't be discussing non-parametric models further in this book, but they are a very interesting and powerful class of algorithm.

Conclusion

In this chapter, we have gone a level deeper into the various flavors of Machine Learning. We have learned that a Machine Learning algorithm is either Supervised or Unsupervised and either Parametric or Non-Parametric. Furthermore, we have explored exactly what makes these 4 different groups of algorithms distinct. We have learned that Supervised Machine Learning is a class of algorithm where we learn to predict one dataset given another and that Unsupervised Learning generally groups a single dataset into various kinds of clusters. We learned that Parametric algorithms have a fixed number of *parameters* and that Non-Parametric algorithms adjust the number of parameters they have based on the dataset.

Deep Learning leverages Neural Networks to perform both supervised and unsupervised prediction. Up until now, we have stayed at mostly a conceptual level, gaining our bearings on the field as a whole and our place in it. In the next chapter, we will be building our first neural network, and all subsequent chapters will be *project based*. So, pull out your Jupyter notebook and let's jump right in!