PRINCETON UNIVERSITY, CENTER FOR INFORMATION TECHNOLOGY POLICY

Introduction to Machine Learning for the Humanities

Amy A. Winecoff *

January 3, 2023

0.1 Unnecessary Preamble

I've been doing data science work in some way, shape, or form since around 2016. But I didn't come into data science via the typical routes of physics, math, or computer science. I came to the field by way of social science. I did not–and for the most part still do not–understand a lot of the intricacies of the math underlying machine learning models. However, I have used machine learning to good effect both in industry and in academic research projects. Understanding something about the mathematical principles that make up the guts of machine learning models can be incredibly useful for figuring out what models to use, how to improve models, and how to appropriately evaluate them. But, being a researcher who *uses machine learning* as one tool in their methodological arsenal is not the same thing as being a research *who develops machine learning algorithms*. These two types of researchers have different goals; therefore, what they need to learn to perform their research is different as well.

Often, practitioners of machine learning can get off the ground with machine learning without spending years learning calculus, linear algebra, object oriented programming, and the like. One of the barriers to doing so is that a lot of introductory tutorials assume that you are already pretty comfortable with these subjects. As a result, such tutorials are difficult to understand for folks who do not have math and computer science backgrounds.

My goal in this tutorial is to touch on some of the basic concepts employed in machine learning then provide more detail by introducing a "minimum viable example." In this example, I'll walk you through how to create a very basic machine learning model step by step. My hope is that this introduction will help you feel more comfortable breaking down larger concepts in machine learning as you move forward in your journey.

*I am sure there are some nerds out there who will find errors with my math or my notation or something. If so, shoot me an email at aw0934 [at] princeton [dot] edu.

1 Why Bother?

You may well be wondering why you should bother learning anything about machine learning in the first place. This is a reasonable thing to wonder about and something I suspect many of us predisposed to use machine learning ought to do more often. But, there are many applications for which machine learning can be a helpful tool. Imagine that we were interested in tracing the intellectual history and mutation of a word or concept using handwritten documents. If we stuck to reviewing documents manually, we would only be able to work with a small proportion of the available data. This would likely make it quite difficult to understand how usage of a word or concept changed over time, especially if it was not particularly prevalent. On the other hand, we could use machine learning methods to (relatively) automatically convert images of handwritten documents into plain text files. We could then use these plain text files to easily search for the words of interest. We might even be able to use additional machine learning techniques to identify differently spelled words that are nevertheless still likely to map to similar underlying concepts (e.g., "doughnut" vs. "donut"). This would significantly expand the scope of what we would be able to do with our research effort.

We can also use machine learning algorithms to identify statistical regularities in datasets that may be of interest to us, even if the meaning of those patterns is not immediately obvious. For example, a machine learning algorithm could plausibly reveal similarities in some subset of artists' paintings, even if these artists are not often grouped together into the same artistic movement. The discovery of such patterns could be generative to a visual art history researcher who might not have otherwise considered these underlying similarities. Thus, insights can potentially stem from machine learning models even if the model is not intended to complete a well-defined task.

2 One Example to Rule Them All

Before dive into machine learning concepts, we need to get acquainted with a hypothetical domain of application. The examples I am going to employ in this tutorial will all be related to tasting spirits, something I love doing with my partner Joe and our friend Scott (See Figure 2.1). Wherever possible, I will introduce ideas using this example to make them more concrete.

Figure 2.1: Scott and Joe Drink All the Spirits

3 Characterizing Spirits

One of the most fun things to do when you're tasting spirits with your pals is to pretentiously discuss their characteristics. You might, for example, pontificate on the color of the spirits–whether they be very light or very dark–or on facets of their flavor, like smokiness. If you are a varsity level dork, you might go so far as to plot your assessments as in Figure 3.1 or even make a table out of them as in Table 3.1.

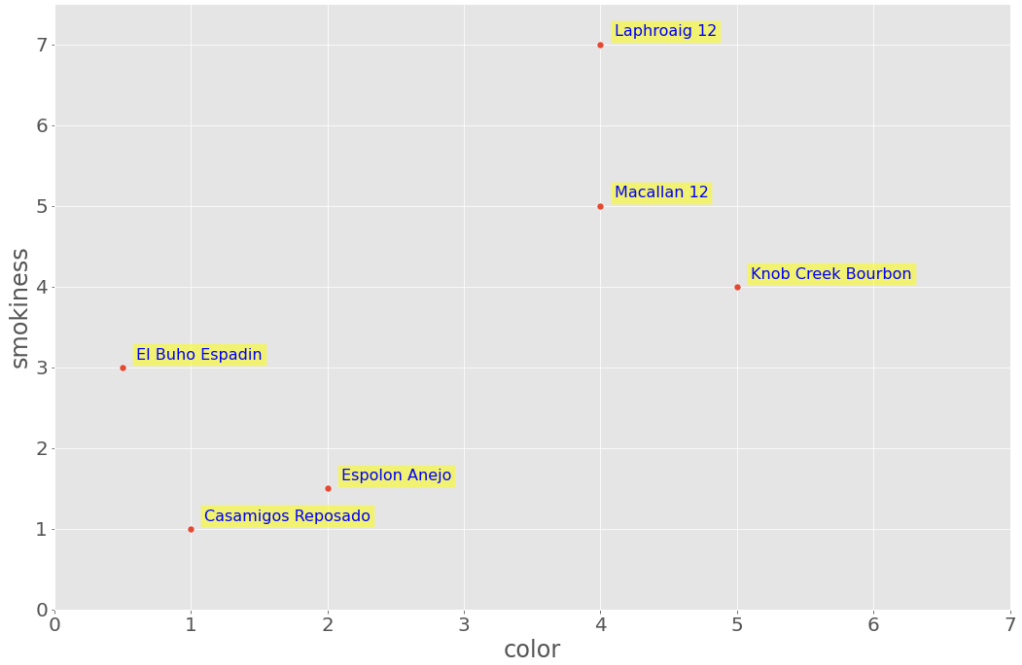

Figure 3.1: Smokiness & Color of Various Spirits

Name	Smokiness	Color
El Buho Espadin	3	0.5
Casamigos Reposado	1	1
Espolón Anejo	1.5	2
Laphroaig 12	7	4
Macallan 12	5	4
Knob Creek Bourbon	4	5

Table 3.1: Ratings of Smokiness & Color of Various Spirits

Each point on the plot and each row in the table represents an "example" or an instance in which we have recorded some information at the level of our unit of observation, which is a spirit the tasters have sampled. If you're anything like me, looking at the table doesn't tell you much, but looking at the plot does. We see two groups of spirits, one with relatively high smokiness and relatively dark color and one with relatively low smokiness and light color. This is no accident; the first group contains all whiskeys and the second contains all mezcals. [1] In other words, there is an association between the properties of the spirits and their types. In machine learning, these are usually referred to as "features" and "labels," respectively. We can expand our initial data table to contain a column for spirit type in Table 3.2

[1] Colloquially, we often refer to a smoky spirit made out of agave as mezcal, but technically, any spirit made out of agave is a mezcal. Tequila is a particular type of mezcal just as bourbon is a type of whiskey.

Name	Smokiness	Color	Type
El Buho Espadin	3	0.5	Mezcal
Casamigos Reposado	1	1	Mezcal
Espolón Anejo	1.5	2	Mezcal
Laphroaig 12	7	4	Whiskey
Macallan 12	5	4	Whiskey
Knob Creek Bourbon	4	5	Whiskey

Table 3.2: Spirit Features & Labels

4 The Machine Learning Process

Now that we have introduced some data, let's walk through some high-level concepts about the machine learning process. Machine learning usually follows a general process similar to that depicted in Figure 4.1. The first step in this process involves gathering data and transforming them into a format that makes sense for building a model. Then, in the second step the data are used to develop a machine learning model that (usually) creates some abstract representation of the relationships between the features and the labels. For example, we might want to mathematically encode the relationship between the characteristics of spirits and their types. Once we have a model, we can use it to make predictions in the third step. We can do this a few ways. The simplest way is to use all the data we have available to us to develop our model and evaluate how well our model performs on this same data. Usually this evaluation method is not a good idea because it will give us an unrealistic estimate of how good our model is for reasons we will briefly touch on later. Instead, typically we take all the data we have available to us and split it into two sets. We use the majority of the data to develop or "train" or model. We refer to this as the "training data" or "training set." Then we use the rest of the data, referred to as the "test set" to evaluate the model we have trained. The goal of this approach to evaluating models is to see how well our model does at making predictions on new data (i.e., how well it "generalizes" to other examples) that it has not "seen" before in the training process. In the fourth step, we compare the model's predictions on each example's label to its actual label. In other words, we perform an evaluation. There is an arrow at this step because usually the process of developing a machine learning model is iterative. You train a model, make some predictions, then evaluate how good those predictions are. If they leave something to be desired, you make some modifications to your model until you achieve a satisfactory performance.

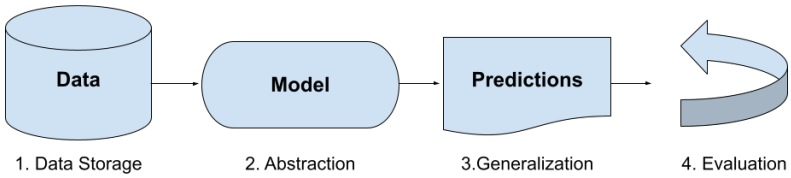

Figure 4.1: Machine Learning Process, Adapted from Lantz, 2015

Even though the details of machine learning algorithms often involve complex formulas, at a conceptual level many of these algorithms draw from ideas that are intuitive. Take as an example a type of machine learning model called a "decision tree." You likely have encountered something very like a decision tree in the form of a flowchart. In one of these charts, you are walked through a series of successive decisions that eventually lead you to an outcome or conclusion. Something similar happens in a machine learning decision tree. In these models, the algorithm creates decision points or "splits" based on the features that are most helpful in

determining the label. In an actual machine learning decision tree, the splits are determined based on a somewhat complex mathematical computation, but for our purposes, I just eyeballed it based on Figure 3.1. First, let's consider a decision tree that creates only one split as in Figure 4.2. Spirits that have smokiness ratings of less than 4 are predicted by the model to be mezcals, and spirits with smokiness ratings of equal to or greater than 4 are predicted by the model to be whiskeys.

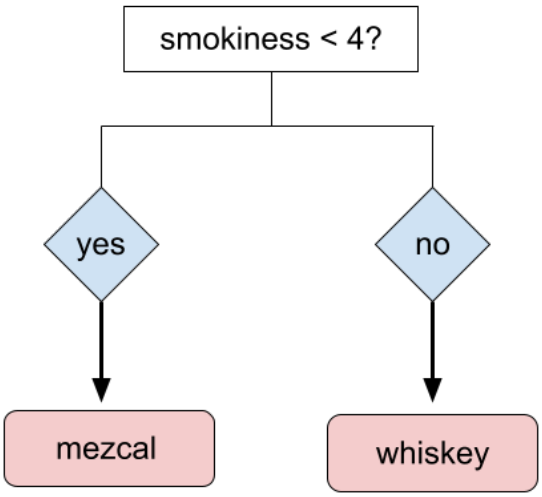

Figure 4.2: Decision Tree with One Split

We can then use this model to make a prediction about every spirit in our dataset as in Table 4.1. All of the correct predictions are in blue and the incorrect predictions are in red. This model would do reasonably well at predicting the types of spirits in our data. It would correctly classify every example except for Knob Creek (since its smokiness is 4, not less than 4). We calculate this model's "accuracy" by dividing the number of correct predictions the model made by the total number of predictions. Here, our model's accuracy is 5/6 or as a percentage 83.3%.

Name	**Smokiness**	**Type**	**Predicted Type**
El Buho Espadin	3	Mezcal	Mezcal
Casamigos Reposado	1	Mezcal	Mezcal
Espolón Anejo	1.5	Mezcal	Mezcal
Laphroaig 12	7	Whiskey	Whiskey
Macallan 12	5	Whiskey	Whiskey
Knob Creek Bourbon	4	Whiskey	Mezcal

Table 4.1: Spirit Predictions from Decision Tree with One Split

We could add another split to our model to improve its performance further as illustrated in Figure 4.3. In this model, there is a second split that provides additional granularity to how to classify the type of spirit that has smokiness of < 4. In these cases, if the color is also light (color equal to or less than 4), we classify that as a mezcal, but if it is dark (color greater than 4), we classify it as a whiskey. As before, we can use this updated model to make predictions on all the examples in our dataset as in Table 4.2.

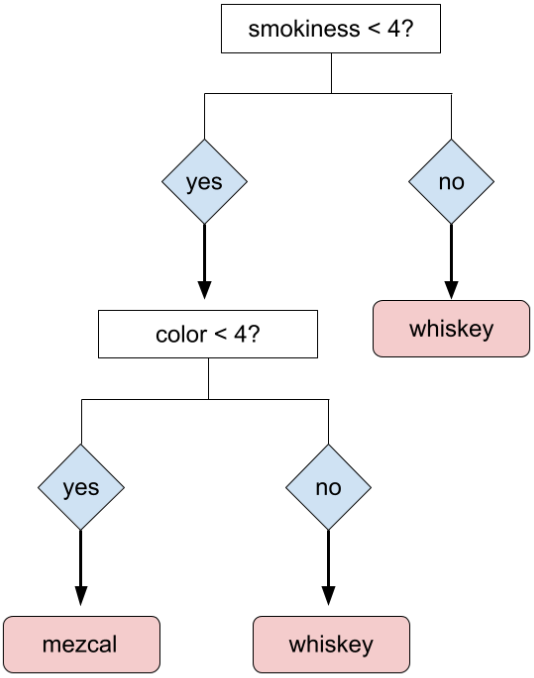

Figure 4.3: Decision Tree with Two Splits

Name	**Smokiness**	**Color**	**Type**	**Predicted Type**
El Buho Espadin	3	0.5	Mezcal	Mezcal
Casamigos Reposado	1	1	Mezcal	Mezcal
Espolón Anejo	1.5	2	Mezcal	Mezcal
Laphroaig 12	7	4	Whiskey	Whiskey
Macallan 12	5	4	Whiskey	Whiskey
Knob Creek Bourbon	4	5	Whiskey	Whiskey

Table 4.2: Spirit Predictions from Decision Tree with Two Splits

This updated model now correctly predicts the class of Knob Creek as well as every other example in our data. It has 100% accuracy. But how well might our model do on other spirits? Consider the case of mezcals that are very smokey (i.e., smokiness greater than 4). Our model would incorrectly classify these as whiskeys. Because we didn't have any very smokey mezcals in the dataset I used to determine the splits (via the eyeball method) in our data, our decision tree would not do a good job of correctly predicting their type. In other words, our model has done a excellent job of learning how spirits' features are associated with their labels in the specific dataset we used to develop the decision tree model, but it would not generalize well to datasets whose examples have slightly different associations. This is a phenomenon referred to as "overfitting" in machine learning and is a consistent bugaboo in the model development process. A detailed explanation of how to avoid this problem is outside of the scope of our introduction; however, a first step is to develop models so that they perform as well as possible on a dataset that was not used during the model development process (i.e., on an independent test dataset).

To go back to the machine learning process diagram in Figure 4.1, we took the following steps. First, we gathered our data of spirits' features and labels. Second, we developed a decision tree

model using these data. Third, we made predictions using the decision tree model on each of the examples in our data. And then fourth, we evaluated the accuracy of the model at predicting the spirit type [2]. Real-world machine learning processes usually follow a series of steps that are a bit more complex than this, but they still resemble the general structure.

Now that we have walked through the machine learning process at a high level, we'll now move on to actually developing a machine learning model. The details of how to create a machine learning decision tree are a bit complex, so we will use a different algorithm that is simpler to understand at the mathematical level. The algorithm we will be developing takes advantage of the fact that data that have similar features are likely to have similar labels. We'll dive into that in the next section.

5 Similarity Between Things

One idea that shows up again and again in machine learning algorithms is "similarity"–the extent to which data points resemble each other. To explain this idea, I'm going to go off on a little bit of a high school math tangent (har, har!). Remember how we used to calculate the length of a hypotenuse of a triangle? Yep, me neither. Here is an equation to jog all of our memories:

$$a^2 + b^2 = c^2 \tag{5.1}$$

This is the Pythagorean theorem. In addition to indicating the length of the hypotenuse of a triangle, it also provides a mathematical measure of the similarity between two points (AKA, examples).

Here's how. Imagine we have two points in a space with two axes, x and y (See Figure 5.1). We can represent the distance between these two points by drawing a line between them. If we want to calculate the distance, we can make a right triangle out of the line running between the two points (See Figure 5.2). Then we can use the Pythagorean theorem to calculate the hypotenuse, which will give us the distance. In Figure 5.2, the line indicated by a is how far apart the two points are on the x-axis. The line indicated by b is how far apart the two points are on the y-axis. We use the values of a and b to calculate c, which equals roughly 18.03 (See Eq. 5.2).

$$\begin{aligned} a^2 + b^2 &= c^2 \\ (20-10)^2 + (20-5)^2 &= c^2 \\ 10^2 + 15^2 &= c^2 \\ 100 + 225 &= c^2 \\ \sqrt{325} &= c \\ 18.03 &= c \end{aligned} \tag{5.2}$$

We can use the distance between these points as a measure of their similarity. Points that are really close together in space can be considered very similar whereas points that are really far apart in space can considered very dissimilar. Note that when we're talking about distance,

[2] For simplicity, I've constrained the example explanation to training and evaluating on the same dataset, but as I have already mentioned, best practices requires that we train and evaluate on separate data

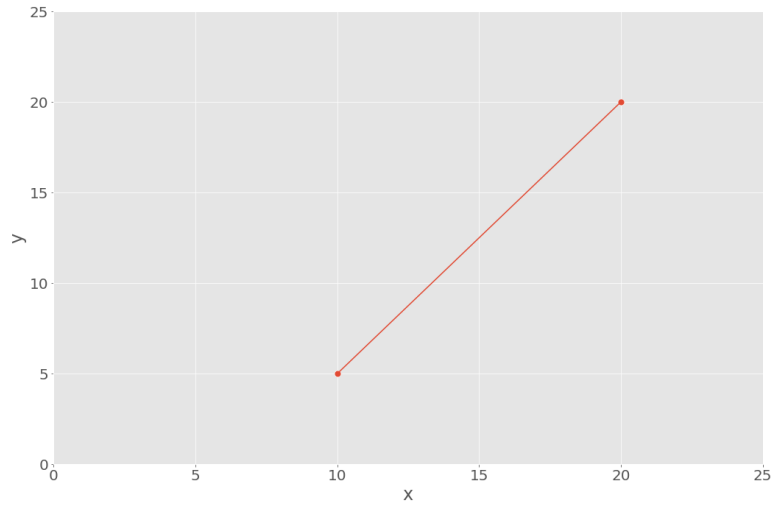

Figure 5.1: Distance between Two Points

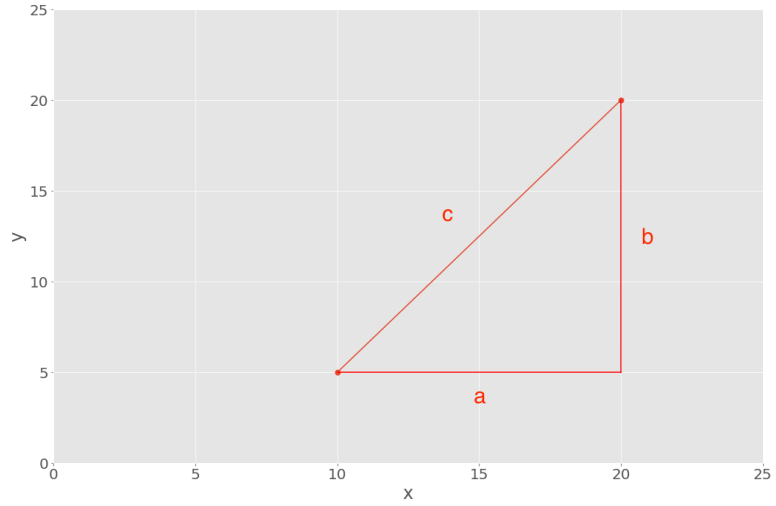

Figure 5.2: Distance Using a Right Triangle

similarity is higher when the distance is shorter (i.e., smaller).

When we calculate distance this way, we often call it the Euclidean distance. You may encounter a formula for the Euclidean distance, D, written something like this:

$$D(point1, point2) = \sqrt{\sum_{i=1}^{N} (point1_i - point2_i)^2} \tag{5.3}$$

This formula is kind of a doozy, so I'll break it down. On the left side of the equation, we have $D(point1, point2)$. You have probably surmised that D stands for distance. In machine learning, D is a type of function–a mathematical object that takes some number of inputs and renders them into outputs. Our distance function D accepts two inputs, $point1$ and $point2$ corresponding to two different points in space and will produce their Euclidean distance as an output. So far, we have talked about points that make up a triangle. In this case, each point is

represented by its x and y coordinates. We could also think of these two axes as representing features of the two points. Although for the triangle example, we only have two features (i.e., axes), in theory, we could have an infinite number of features.

The $\sum$ symbol stands for a summation. When you use the summation symbol, you put the starting point of stuff you want to sum at the bottom and the stopping point of stuff you want to sum at the top. For example, if I wanted to sum the number 2 raised to the 1st, 2nd, 3rd, and 4th power, I could write:

$$S = \sum_{i=1}^{4} 2^i \tag{5.4}$$

The letter i is the index, which keeps track of the element we are currently summing. So, that means that we could expand the equation above (Eq. 5.4) to the following:

$$S = 2^1 + 2^2 + 2^3 + 2^4 \tag{5.5}$$

In the distance function D, we are performing the summation for all of the features of $point1$ and $point2$, starting from the first feature and running through the last feature, N. Again, for triangles, we only have two "features." The first feature is the value of the x-axis for each point, and the second feature is the value of the y-axis. So, to calculate the distance, we take the x value for $point1$, subtract the x value for $point2$, then square that difference. Then we do the same thing for the y value. After we have summed the squared difference for each of the features of $point1$ and $point2$, then we take the square root. Expanding out the distance equation Eq. 5.3 for this example, we then have:

$$D(point1, point2) = \sqrt{(point1_x - point2_x)^2 + (point1_y - point2_y)^2} \tag{5.6}$$

We go through all of this squaring and unsquaring because we don't really care about whether the difference is positive or negative, we just care about how large the difference is. Think about this, if you walked some number of feet from your house, that distance should be positive, regardless of whether you walked east, west, south, or north. Squaring the differences solves this problem since all the squared distances will be positive. We then take the square root of the sum to get it back in the original units. You might think, "Well, why not just take the absolute value then?" Sick dude, you just reinvented what is known as the Manhattan distance [3].

Those ideas in the abstract will make more sense when we apply them to the a concrete scenario, so let's get back to our spirits example. We can use the Euclidean distance to capture the distance between any two spirits. Let's take El Buho and Laphroaig, displayed for your convenience in Figure 5.3.

For these two spirits, the values we would be working with would be the two spirits' color and

[3] Not to be confused with the Manhattan cocktail

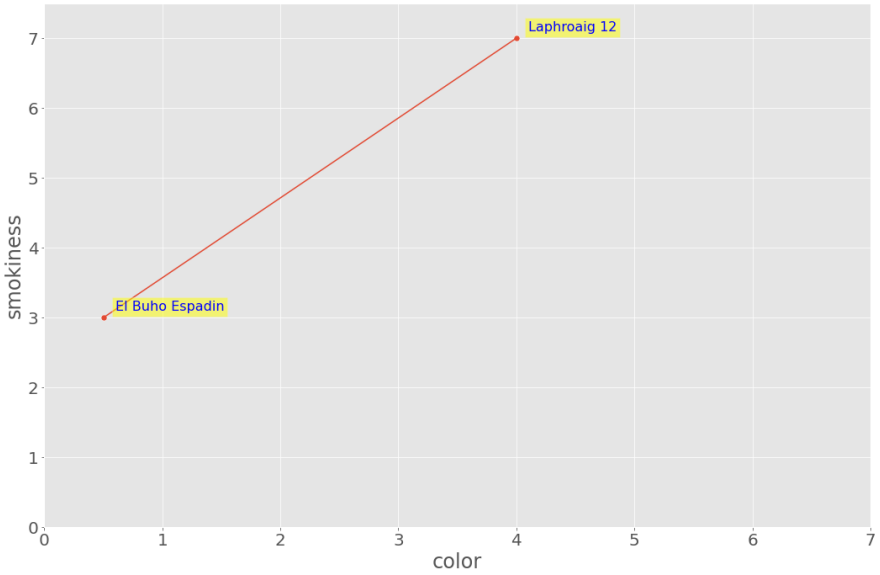

Figure 5.3: El Buho and Laphroaig

smokiness. In the formula below, El Buho is "EB" and Laphroaig is "L."

$$\begin{aligned} D(L,EB) &= \sqrt{\sum_{i=1}^{N}(L_i - EB_i)^2} \\ D(L,EB) &= \sqrt{(4-0.5)^2 + (7-3)^2} \\ D(L,EB) &= \sqrt{28.25} \\ D(L,EB) &= 5.32 \end{aligned} \tag{5.7}$$

Again, in non-mathematical language, we first subtract the color of El Buho (0.5) from the color of Laphroaig (4). We then square this distance. We then do the same for smokiness, subtracting the smokiness of El Buho (3) from the smokiness of Laphroaig (7), then squaring the difference. We add these two values together, then take the square root to get the distance, which equals 5.32.

6 Making Predictions Using the Similarity Between Things

I mentioned before that we would come back to the association between examples' features and labels. Just to remind you all of what I'm talking about here, the labels are the type of spirit, as shown in Table 6.1

Imagine that there are some bottles that Scott has sneakily removed the labels from. Joe decides to taste one of these spirits, which we'll refer to as the "unknown spirit." He rates its smokiness as 2 an its color as 3. He wonders, "Is this spirit a mezcal or a whiskey?" Fortunately, we can use our handy-dandy distances to make a prediction. To do this, he first calculates the Euclidean distance between the unknown spirit and all the other spirits he has tried. It would probably take up a whole page to calculate each of these, so I'll just show the calculation for the unknown spirit (US) and Espolón Anejo (EA).

Name	Smokiness	Color	Type
El Buho Espadin	3	0.5	Mezcal
Casamigos Reposado	1	1	Mezcal
Espolón Anejo	1.5	2	Mezcal
Laphroaig 12	7	4	Whiskey
Macallan 12	5	4	Whiskey
Knob Creek Bourbon	4	5	Whiskey
Unknown Spirit	2	3	???

Table 6.1: Spirit Features & Labels

Spirit	Distance to Unknown Spirit	Type
Unknown Spirit	0	?
Espolon Anejo	1.12	Mezcal
Casamigos Reposado	2.24	Mezcal
El Buho Espadin	2.69	Mezcal
Knob Creek Bourbon	2.83	Whiskey
Macallan 12	3.16	Whiskey
Laphroaig 12	5.1	Whiskey

Table 6.2: Distance to the Unknown Spirit

$$\begin{aligned} D(US,EA) &= \sqrt{\sum_{i=1}^{N}(US_i - EA_i)^2} \\ D(US,EA) &= \sqrt{(3-2)^2 + (2-1.5)^2} \\ D(US,EA) &= \sqrt{1.25} \\ D(US,EA) &= 1.12 \end{aligned} \tag{6.1}$$

Table 6.2 contains all of the distances between each spirit and the unknown spirit. By definition, the distance between a point and itself is always 0. The rest of the spirits are listed from the shortest distance to the longest from the unknown spirit. From this table, we can see that all of the mezcals are closer–that is, more similar–to the unknown spirit than all of the whiskeys. Put another way, the unknown spirit's three "nearest neighbors" are mezcals (whereas all of its furthest neighbors are whiskeys). Based on this, it's reasonable for Joe to predict that the unknown spirit is probably a mezcal [4].

But what if Joe had instead rated the smokiness of the unknown spirit as 2.5 instead of 2? That changes the the math a bit. I've added what the distances would look like with that change in Table 6.3, again ordered from shortest to longest distance from the unknown spirit. You can see that now two out of the three of the unknown spirit's neighbors are mezcals and one is a whiskey. Mescal is still the most prevalent of the three nearest neighbors, so Joe can predict that the spirit is a mezcal albeit with less confidence than if all three of the nearest neighbors were mezcals.

We can extend this process to any number of the bottles that Scott removed the labels from, calculating their distances to all the examples where Joe does know their type and predicting

[4] We're making the assumption here that the unknown spirit is *either* a mezcal *or* a whiskey

Spirit	Distance to Unknown Spirit	Type
Unknown Spirit	0	?
Espolon Anejo	1.41	Mezcal
Casamigos Reposado	2.5	Mezcal
Knob Creek Bourbon	2.5	Whiskey
El Buho Espadin	2.55	Mezcal
Macallan 12	2.69	Whiskey
Laphroaig 12	4.61	Whiskey

Table 6.3: Modified Distance to the Unknown Spirit

whatever label is most prevalent amongst their nearest neighbors. If Scott were feeling magnanimous, perhaps he could let Joe know how many of the predictions he made on the unlabeled bottles were actually correct, allowing him to evaluate the accuracy of his predictions. This set up is akin to a scenario where we develop a model on the training set and evaluate its performance on the test set.

7 K-Nearest Neighbors and Beyond

The model we have developed here is called "K Nearest Neighbors (KNN)." In this type of model, we first calculate the distance between an example we'd like to make a prediction about and all of the other examples in our dataset. We then assign this example the label that that is the most prevalent amongst its k nearest neighbors. When the label we're trying to predict is a category or "class," we call our model a "classifier." Both KNN as well as lots of other types of machine learning models can be used for classification (we have also discussed another, the decision tree). Other times, we are trying to predict a numeric outcome. So, for example, we might want to use the features of a home to predict what its eventual sales price would be. When the outcome we are trying to predict is numeric, we call our model a "regression" model. KNN can also be adapted to solve regression problems. If our labels were not the type of spirit but the price of the spirits, we could simply take the mean price of the unknown spirit's k nearest neighbors, and use that as the prediction for the price of the unknown spirit. There are fancier things we can do to make even better predictions, such as by weighting more heavily the examples that are more similar to the example we are trying to make a prediction for, but you get the idea.

What we do not do with KNN models is create an abstract representation of our data in the form of a model that can be used to make predictions. In KNN, for each prediction we make, we essentially use the data itself to make predictions. This is a bit different from the decision tree example, where the model learns a representation in the form of the feature splits. Because models like KNN do not learn an abstract model representation, they are sometimes referred to as "lazy learners." But don't let the pejorative fool you; lazy learners are often quite powerful. In fact, many of the product recommendations you encounter on websites (e.g., "because you liked this, you may like that") likely use some form of a lazy learner under the hood.

8 Model Supervision

In both the decision tree example and the KNN example, we were aiming to create models that can use the features of the data to predict the label. These types of models are referred to as

"supervised learning." Google's machine learning glossary provides a great definition:

> **Supervised learning:** *Training a model from features and their corresponding labels. Supervised machine learning is analogous to learning a subject by studying a set of questions and their corresponding answers. After mastering the mapping between questions and answers, a student can then provide answers to new (never-before-seen) questions on the same topic.*

This is exactly what we did in the case of the unknown spirit: We used the labels of the spirits most similar to the unknown spirit to make a prediction about the unknown spirit's type. As you might have intuited, there is also a class of models referred to as "unsupervised learning." Again, we'll rely on Google's glossary to provide the definition:

> **Unsupervised learning:** *Training a model to find patterns in a dataset, typically an unlabeled dataset. The most common use of unsupervised machine learning is to cluster data into groups of similar examples.*

For better or worse, many real-world datasets do not have labels. Yet unsupervised learning models can still be used to understand meaningful patterns in data. Conceptually applying the idea of unsupervised clustering to our spirits example, imagine that we had no labels for any of our data. We could still determine which groups of spirits were most similar to each other using unsupervised learning methods that again rely on distance. In essence, these methods would allow us to draw circles around the spirits that are most similar to each other (See Figure 8.1). Even though we wouldn't know their underlying type, we would know the spirits in the top right corner are more similar to each other than they are to the spirits in the bottom left corner.

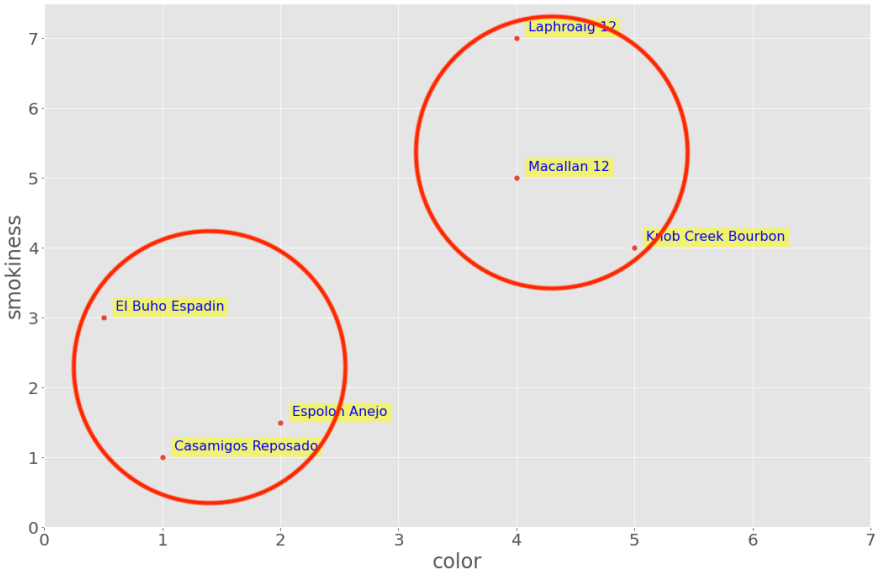

Figure 8.1: Clusters of Spirits

9 Beyond Basic Evaluations

I would be remiss if I said nothing else about model evaluation. The choices you make about how to evaluate your model can dramatically influence your sense of its performance. When

your evaluation strategy is poor, you do not get a good idea of how well your model would do in situations you care about. For example, imagine that I were developing a model for predicting whether newborns have hemophilia. Hemophilia is very rare, with an estimated prevalence of less than 1%. Therefore, without using any machine learning at all, I can can correctly predict whether a newborn will be affected by hemophilia more 99% of the time by just always predicting that the newborns do not have hemophilia. But of course, this is not a good evaluation metric. A better choice, for example, would be to calculate how often the model correctly predicts a newborn's status when the newborn actually *does* have hemophilia. I have developed a module on machine learning model evaluation that you can access here [5]. This module is a bit more technical than the current document, but the concepts therein may still be helpful for a less technical reader.

10 Conclusion

In this tutorial, we have covered a lot of conceptual ground and a little bit of technical ground related to machine learning. We have covered the machine learning process at a high level, have discussed a few different modeling approaches, and have created a simple toy model. We have also broken down several mathematical formulas, which is a skill that is useful when trying to understand machine learning tutorials or research papers since lots of them use formulas with mathematical notation [6] If you continue forth in your machine learning journey, your next step will likely involve learning some additional math and coding. But hopefully this provided you with a friendly introduction that help you dive into those topics with a bit more confidence.

11 Glossary of Terms

Unless otherwise indicated, definitions are taken from Google's Machine Learning Glossary with minor modifications.

- **Accuracy:** The number of correct classification predictions divided by the total number of predictions. For example, a model that made 40 correct predictions and 10 incorrect predictions would have an accuracy of 80%.
- **Class:** A category that a label can belong to. For example, in a binary classification model that detects spam, the two classes might be spam and not spam. In a multi-class classification model that identifies dog breeds, the classes might be poodle, beagle, pug, and so on.
- **Classification model:** A model whose prediction is a class. For example, the following are all classification models: 1) A model that predicts an input sentence's language (French? Spanish? Italian?); 2) a model that predicts tree species (Maple? Oak? Baobab?); 3) a model that predicts the positive or negative class for a particular medical condition.
- **Decision tree:** A supervised learning model composed of a set of conditions and "leaves" (endpoints in the decision tree) organized hierarchically.
- **Distance:** A numerical measurement of how far apart objects or points are. Adapted from Wikipedia.
- **Example:** The values of one row of features and possibly a label.
- **Feature:** An input variable to a machine learning model.
- **Function:** A mathematical object that takes numbers as inputs and gives numbers as outputs. Adapted from Savov.

[5] If you access this link before January 19th, 2023, this repository will likely still be under construction!
[6] For a really great resource on notation, check out this work by Edward Scheinerman.

- ***K*-nearest neighbors classifier:** A classifier model in which the label of a new example is predicted based on the *k*-closest data points in the training set. Adapted from Kuhn & Johnson
- **Label:** In supervised machine learning, the "answer" or "result" portion of an example.
- **Lazy learning:** In machine learning, lazy learning is a learning method in which generalization of the training data is, in theory, delayed until a query is made to the system, as opposed to eager learning, where the system tries to generalize the training data before receiving queries. From Wikipedia.
- **Overfitting:** Creating a model that matches the training data so closely that the model fails to make correct predictions on new data.
- **Regression model:** Informally, a model that generates a numerical prediction. For example, the following are all regression models: 1) A model that predicts a certain house's value, such as 423,000 Euros; 2) a model that predicts a certain tree's life expectancy, such as 23.2 years; 3) a model that predicts the amount of rain that will fall in a certain city over the next six hours, such as 0.18 inches.
- **Similarity:** A metric used to determine how alike any two examples are.
- **Supervised learning:** Training a model from features and their corresponding labels. Supervised machine learning is analogous to learning a subject by studying a set of questions and their corresponding answers. After mastering the mapping between questions and answers, a student can then provide answers to new (never-before-seen) questions on the same topic.
- **Training data/training set:** The subset of the dataset used to train a model.
- **Test set**: A subset of the dataset reserved for testing a trained model. Each example in a dataset should belong to only one of the training or test sets. For instance, a single example should not belong to both the training set and the test set.
- **Unsupervised learning:** Training a model to find patterns in a dataset, typically an unlabeled dataset. The most common use of unsupervised machine learning is to cluster data into groups of similar examples. For example, an unsupervised machine learning algorithm can cluster songs based on various properties of the music.