

Machine Learning – Learning, Reasoning, and Training

15-110 – Friday 04/03

Announcements

- **Check6-1** is due Monday
 - Don't forget to fill out the project survey!
Link: <https://forms.gle/haLsNXTQ6gPq6Nf7A>
 - Remember that Check6-1 also has a written component

Hiring 15-110 F26 TAs!

We need some new TAs to help teach the course in F26!

If you love teaching, understand the course material, and want to introduce computer science to other students, you should apply!

Link: <https://bit.ly/110-f26-ta>



Learning Goals

- Identify the three major categories of learning (**supervised**, **unsupervised**, and **reinforcement**) and the three major categories of reasoning (**classification**, **regression**, and **clustering**)
- Decide which combination of **learning** and **reasoning** categories are best used to solve a stated problem

Machine Learning and Models

Machine Learning Creates Models

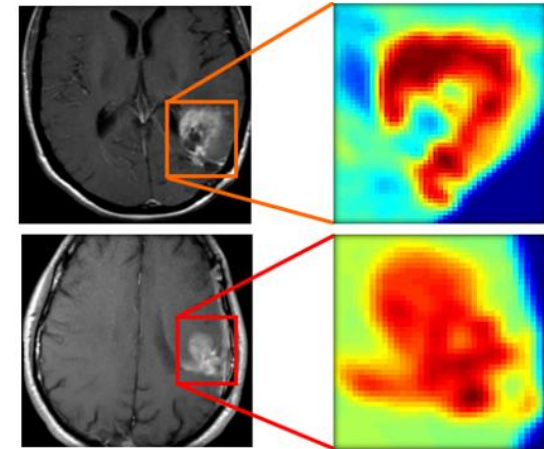
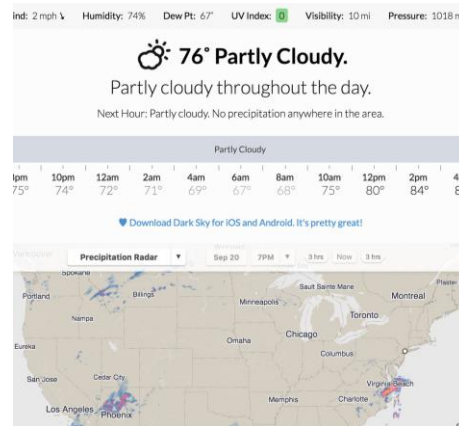
In data analysis and simulation, we designed algorithms and models to explore specific datasets.

Machine learning (ML) goes one step further. Instead of designing a model for a dataset, we apply a general machine learning algorithm to the data. This identifies patterns in the data and **generates the model automatically**. It's like outsourcing the work to the computer!

The programmer doesn't show the algorithm the pattern; they teach the algorithm how to find patterns generally. This means that the computer can generate many different models on its own, given the right data.

Machine Learning in Many Contexts

Machine Learning generates models that are used in thousands of contexts across the world, including speech recognition, weather prediction, and medical diagnosis.



We'll start by discussing the most common types of machine learning and how it works, then show examples of real-world machine learning used today.

Models

The **model** generated by a machine learning algorithm is used to perform reasoning tasks.

Once a model has been generated, we can apply it to new data to make predictions. A model trained on market behavior could make a prediction about which products produced by a business might be most popular in an upcoming quarter.

We can also analyze models to gain insights about the original dataset. For example, we might build a model to group open-ended feedback provided in a country-wide survey, then use those groups to identify common trends among answers.

Models are Based on Data

To train a model using machine learning, we must start with a large amount of **data**. The algorithm will try to find patterns and correlations in that data. A model is only as good as its dataset!

Student ID	Full Name	favourite.food	mealPlan	AGE
1	Sunil Huffmann	Strawberry yoghurt	Lunch only	4
2	Barclay Lynn	French fries	Lunch only	5
3	Jayendra Lyne	N/A	Breakfast and lunch	7
4	Leon Rossini	Anchovies	Lunch only	five
5	Chidiegwu Dunkel	Pizza	Breakfast and lunch	five
6	Güvenç Atilla	Ice cream	Lunch only	6

Each data entry must be broken down into a set of **features** so the machine learning algorithm can analyze features separately or in groups. If you have a table of data, the features would be the **columns**; if you have a set of text, the features would be **words** or **groups of words**.

You can create and add new features the same way you would add new columns in data analysis.

Shark1

Anastasia Shark
Mr. Meluso
Computer Multimedia
13 November 2009

About Me

As you may know I am a freshman at Passaic Valley. Coming into this school was really exciting! The first time I came here to Passaic Valley HS I was scared that I was going to get lost when I was going to my classes. It took me a couple days to figure out where everything is. I still don't know where everything is in this school but I am working on it. This school is so much different from my old middle school. There are not only a few different things but many different things about this school. One of the things is that it's so much bigger in PV. My last school was very small and I knew where every class was. This school is about three times bigger though. Another good thing about this school is that the food is so much better! At my other school the food looked really gross and disgusting. Passaic Valley HS is a much better school and I am happy to be in here.

This year for spring break I am going away to Mexico! I can't wait, my parents went to Mexico last year and had such a great time so I'm really excited go. The cool thing is that it's not only my family going my cousins and aunts and uncles are coming too! There's going to be so much people but I'll enjoy it. I love going away for spring break 2 years ago I went to the Bahamas with my mom, dad and my friend and her family. We had a great time. The hotel we stayed at was amazing there was a huge water park in it and many pools. I love going away on vacation because it's so warm

Learning Categories

Many Ways to Learn

There are several ways that machine learning algorithms learn patterns from data. Which approach you use depends on what type of problem you're trying to solve.

The three main approaches are **supervised learning**, **unsupervised learning**, and **reinforcement learning**.

Supervised Learning



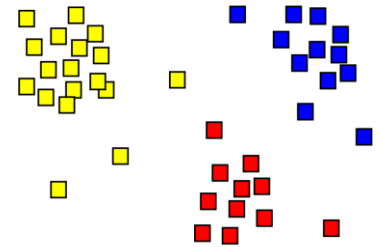
Supervised learning algorithms train models to predict outputs based on inputs. A data entry in the dataset is the input, and a **label** or **score** is the output.

Example: to train a model to predict a person's favorite ice cream based on other desert preferences, each data entry (set of desert preferences) must be accompanied by a label (that person's favorite ice cream).

Once the model is fully trained, it can be used to make predictions about unlabeled data.

This is like when you learn by completing a homework assignment. You're trying to predict a correct answer (label) which is known by the teacher. This means the teacher (ML algorithm) can check your work. Later, you'll apply your knowledge to new problems where the answer is unknown.

Unsupervised Learning



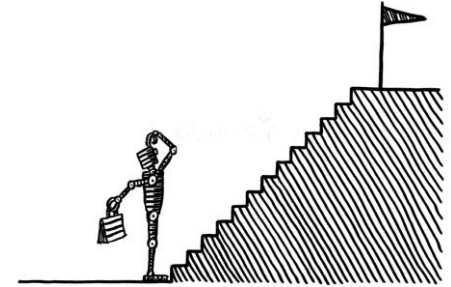
Unsupervised learning algorithms train models that infer the natural structure of a dataset. Which data entries are most similar to each other?

Example: identify common categories of favorite ice cream flavors by grouping together people who have similar desert preferences.

Unsupervised learning is often used when the data is **not** labeled – no true answer is known. That means that a human being will need to look over the model's results to see if they make sense or seem random.

This is like how researchers come up with theories to explain how the world works. Theories are based on data and may make sense, but might be changed or disproven later on.

Reinforcement Learning



Reinforcement learning algorithms train models that help an **artificial intelligence agent** (AI agent) approach an overall goal. Usually this involves taking a sequence of actions to bring the agent closer to the endpoint and giving feedback on those actions.

Example: train a robot to make ice cream by constantly changing the temperature and speed of an ice cream machine and measuring the consistency of the product until it's just right.

This is like playing a guessing game where the person with the secret word tells you if your guesses are hot or cold. You adjust your guesses and questions based on the person's feedback until you've gotten to the correct word.

We'll talk more about how AI works later in the unit.

Other Types of Learning

Semi-supervised learning: mixed dataset where some entries are labeled but most aren't.

Active learning: the algorithm asks an expert to label specific entries to help it learn.

Both are used when a model needs to be trained on a lot of unlabeled data, like with Large Language Models (LLMs).

Demo: Supervised Learning

We can build an image-recognition model using supervised learning by labeling a bunch of pictures as belonging to specific classes.

Try it out here: <https://teachablemachine.withgoogle.com/train/image>

Reasoning Categories

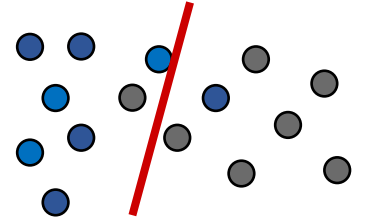
Many Ways to Reason

Once you've identified what type of learning you need to do, there are still lots of options for what type of algorithm you can use. One way to narrow the options down further is to ask what type of **reasoning** the algorithm will need to do.

For supervised learning, common reasoning approaches include **classification** and **regression** algorithms.

For unsupervised learning, common reasoning approaches include **clustering**.

Classification Algorithms

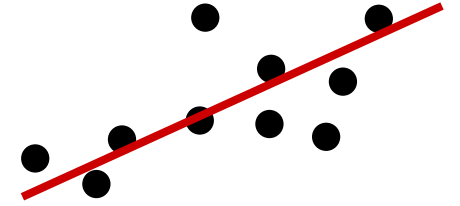


A **classification** algorithm takes labeled data of any type and produces a **model** (called a classifier) which determines which **category** a given data entry most likely belongs to. In other words, it produces a **categorical** (discrete) result.

For example, a classifier might take in data about an email (the sender, title, message content) and classify it as SPAM or NOT SPAM

Classification is commonly used in image recognition. Examples of classification algorithms include Naive Bayes, Decision Trees, and Support Vector Machines (SVMs).

Regression Algorithms

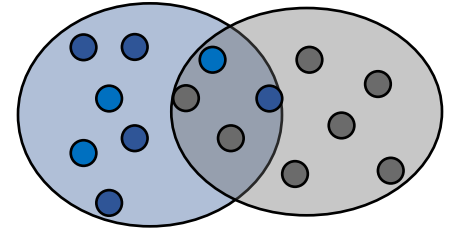


A **regression** algorithm takes numerically-labeled data and produces a function that maps data entries to **numerical** (continuous) results. Instead of predicting a specific category, it gives each data entry a **score**.

For example, a regression algorithm might take information about a house (GPS location, previous sell date, previous sell price, average price of nearby houses) and use it to predict its market value (a price).

Regression algorithms are used to predict things like stock market prices and weather temperatures. A linear regression is one kind of regression algorithm.

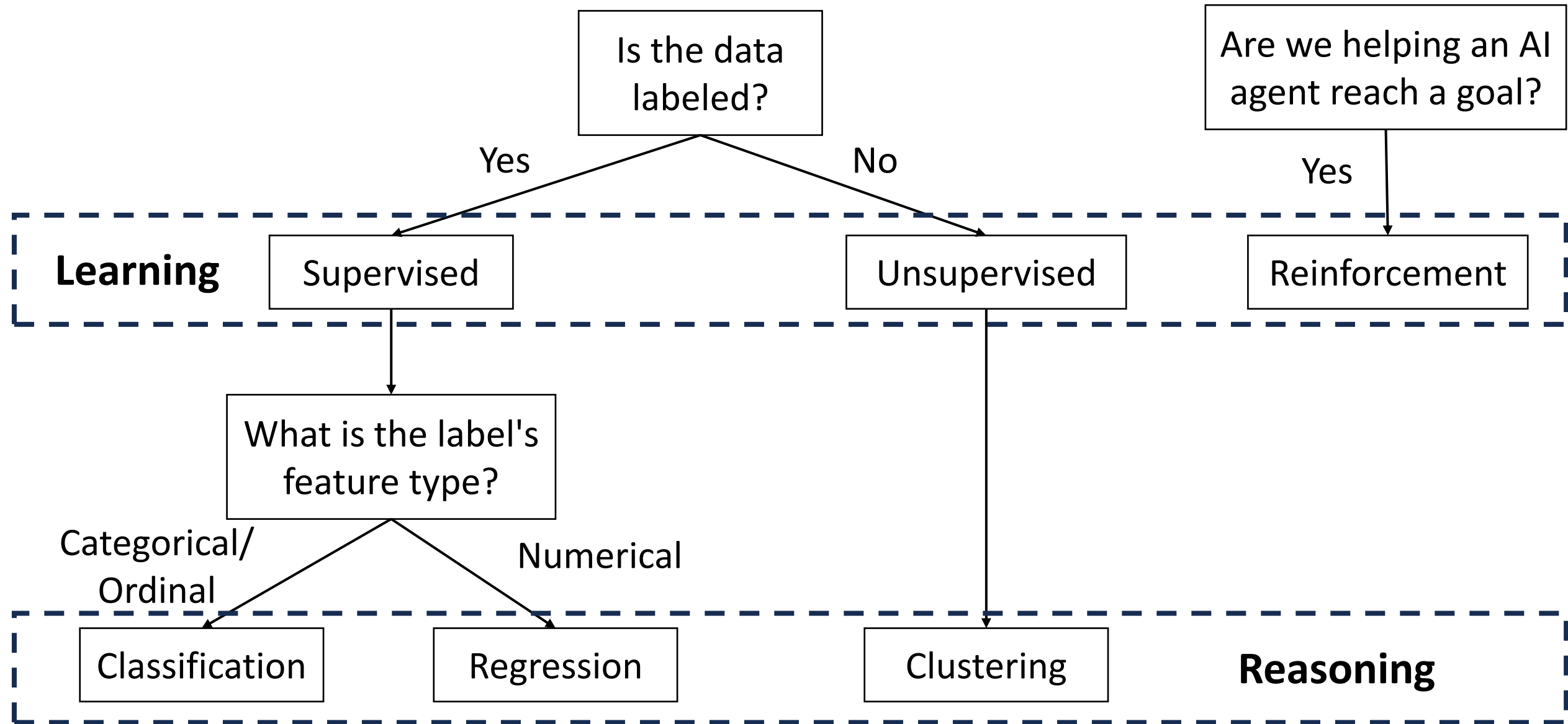
Clustering Algorithms



A **clustering** algorithm takes a set of data and produces a model that breaks the data into some number of **clusters** by identifying data entries that are similar to each other. We don't actually know the true categories or scores of the provided data, so it's hard to 'test' the resulting model, as we aren't sure what the actual result should look like.

For example, a clustering algorithm could recommend a new set of plant species by finding similarities between plant features in a given region.

Clustering is used in consumer analysis, data compression, and bioinformatics. K-means clustering and affinity propagation are examples of clustering algorithms.



Activity: Choose How to Reason

You do: you want to use machine learning to solve a problem. What combination of learning category + reasoning category would you use?

- You have a dataset of the restaurants that people have visited in the past three months. How would you create new customer-profile groupings of people based on this data?
- You have a dataset of survey results, including participant's favorite TV shows, sports, and hobbies. How would you predict someone's favorite sport based on the rest of the survey data?
- You have a dataset of exercise logs (time spent in a variety of exercise activities). How would you estimate the amount of time a person spends running on a treadmill based on other activities?

Many Other Algorithms

Classification, regression, and clustering are just the most common types of reasoning algorithms. There are plenty of others that we don't have time to discuss in this class.

To make a model, though, you just need one algorithm: the best fit for the problem you're trying to solve.

Let's dive a bit deeper into how we can apply machine learning with supervised algorithms specifically.

Training a Model

ML Process: Decide, Train, Validate, then Test

To apply machine learning to a supervised task, follow a simple process

First: **decide** which learning algorithm you'll use, what it should produce and which features you'll train on.

Second: **train** a model created by the algorithm by providing it with the data.

Third: **validate** the model against a subset of data to make changes that can improve its performance.

Third: **test** the model on a different set of data. This helps determine how accurate the model actually is.

Training Identifies Key Features

During the training process the algorithm identifies which features contribute the most to the underlying pattern. It does this by picking features that reduce the **error** that results from running the model on known data.

It's rare for a machine learning algorithm to identify a single feature that can definitively be used to answer a question. Usually, the algorithm uses a **combination of several features** which are weighted based on how well they correlate with the correct answer.

The algorithm needs to learn from a **lot** of examples to get a good sense of what the real pattern is.

Example: Is It a Dog?

Let's use a very simplified example to demonstrate how a machine learning algorithm might learn which features best match a label.

We want to train a model to recognize pictures of dogs. What kinds of features might it identify?

(The features we describe here are far too high-level, but the general idea is about right).

Example: Is It a Dog?

Guess: ?

Answer: 

Prior knowledge:

???

Updated knowledge:

- Has Pointy Ears
- Has Fur
- Screen in background



Example: Is It a Dog?

Prior knowledge:

- Has Pointy Ears
- Has Fur
- Screen in background

Updated knowledge:

- Has Pointy Ears
- Has Fur
- Grey/Black Coloring

Guess:



Answer:



Example: Is It a Dog?

Guess:



Answer:



Prior knowledge:

- Has Pointy Ears
- Has Fur
- Grey/Black Coloring

Updated knowledge:

???



Real-World Example: ChatGPT

Let's consider the popular LLM [ChatGPT](#). How was this system trained?

ChatGPT was trained to generate sequences of text using a HUGE dataset based on the internet and a large corpus of books. This taught the model which words most often go together and allowed it to **generate new text**.

ChatGPT was trained in how to communicate using an advanced algorithm that combined **supervised learning** and **reinforcement learning**.

- For supervised learning, the programmers created example conversations with human and chatbot participants to serve as training data
- For reinforcement learning, ChatGPT generated several possible responses and humans ranked them

Next Time: Validating and Testing the Model

Once we've built a model, how can we optimize it to help it perform at a high level?

We can use **validation** to find the best model possible and **testing** to see how well that model actually works. We'll talk about how to do that in the next machine learning lecture.

Learning Goals

- Identify the three major categories of learning (**supervised**, **unsupervised**, and **reinforcement**) and the three major categories of reasoning (**classification**, **regression**, and **clustering**)
- Decide which combination of **learning** and **reasoning** categories are best used to solve a stated problem

Machine Learning with scikit-learn

Bonus material

Real Example: Favorite Ice Cream Model

Now let's try building a real model with machine learning!

We'll use our ice cream data from the Data Analysis lecture. Our goal is to predict a person's 3rd favorite ice cream based on their 1st and 2nd preferences.

There are a ton of possible flavors, so we'll **hand code** the data to group similar flavors together and reduce the number of options to predict. This results in 6 possible categories: chocolate, coffee/tea, cookie, fruit, vanilla, and other.

Our approach is **supervised** (we already know what people's favorites are) and predicts a **categorical** result (an ice cream flavor). We need to use a classification algorithm.

Machine Learning Libraries

You can implement a machine learning algorithm yourself, but it isn't usually necessary. There are lots of great **external libraries** that have already implemented machine learning algorithms for you. All you have to do is plug in data!

We'll use the library **scikit-learn** here. You can install it in Thonny by going to Manage Packages and searching 'scikit-learn', or by running **pip** with:

```
pip install scikit-learn
```

We won't expect you to write code with **scikit-learn** this semester; these slides are just here in case you're curious about what coding with machine learning truly looks like.

Naive Bayes Classifier

Naive Bayes is a classification algorithm. The classifier it creates uses **conditional probabilities** to predict categories.

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}$$

When Naive Bayes is given a new data entry X, it uses the data it has already collected to determine the **probability** that X belongs to each possible category.

For each category C, it checks how **probable** it is that each feature in X occurs with C by checking how often it occurred in past data entries. By multiplying these probabilities together, it can form a general probability that the data entry belongs in that category.

$$P(C|X) = P(X_1|C) * P(X_2|C) * P(X_3|C) * P(C)$$

The category with the highest probability wins!

Naive Bayes Example

Say we've collected the following data:

- #1: Chocolate
#2: Cookie
#3: Chocolate
- #1: Chocolate
#2: Coffee/Tea
#3: Vanilla
- #1: Cookie
#2: Cookie
#3: Vanilla
- #1: Coffee/Tea
#2: Vanilla
#3: Chocolate
- #1: Cookie
#2: Vanilla
#3: Chocolate
- #1: Coffee/Tea
#2: Vanilla
#3: Vanilla

If we encounter a new data point where the #1 favorite is Chocolate and the #2 favorite is Vanilla, we'd calculate the odds of each class as follows:

- $P(\text{\#3 Chocolate} \mid \text{\#1 Chocolate, \#2 Vanilla}) = P(\text{\#1 Chocolate} \mid \text{\#3 Chocolate}) * P(\text{\#2 Vanilla} \mid \text{\#3 Chocolate}) * P(\text{\#3 Chocolate}) = 1/3 * 2/3 * 1/2 = 1/9$
- $P(\text{\#3 Vanilla} \mid \text{\#1 Chocolate, \#2 Vanilla}) = P(\text{\#1 Chocolate} \mid \text{\#3 Vanilla}) * P(\text{\#2 Vanilla} \mid \text{\#3 Vanilla}) * P(\text{\#3 Vanilla}) = 1/3 * 1/3 * 1/2 = 1/18$

Chocolate has a higher probability than Vanilla, so Naive Bayes would predict that classification for the entry.

Naive Bayes in scikit-learn

In `scikit-learn`, we can employ Naive Bayes with the function `sklearn.naive_bayes.CategoricalNB()`.

This will create an 'empty' model. We can then train that model on data using `model.fit(data, labels)`, and run it on new data using `model.predict(newData)`.

However, we'll need to rearrange our data to make it fit into the library's expectations! `CategoricalNB` wants integers, not strings. We can solve this by mapping each category to a number; in fact, this has been done already in the code posted to the course website.

0: chocolate	1: coffee/tea	2: cookie
3: fruit	4: vanilla	5: other

Setting Aside Test Data

Before training, we need to split our data into two groups- **training** data and **testing** data. That will make it possible for us to check our work later on.

We have information about when the data was collected, so let's just test on the most recent semester for now.

```
...
trainInput = []
trainLabels = []
testInput = []
testLabels = []
# allData = [ [ sem, #1, #2, #3 ] ]
for point in allData:
    semester = point[0]
    inputValues = point[1:3]
    outputValue = point[3]
    if semester == "S26":
        testInput.append(inputValues)
        testLabels.append(outputValue)
    else:
        trainInput.append(inputValues)
        trainLabels.append(outputValue)
```

Training the Model

Once the data is organized, training the model is easy. Just generate a new instance of the model using `CategoricalNB`, then call the `fit` method on that model using the data.

We can use the trained model to make predictions on new data points. For example, let's say we want to predict someone's 3rd favorite if their 1st favorite is chocolate and 2nd favorite is cookie. Create a list of the two category numbers (0 and 2), put that in another list, and call the `predict` method on the model.

The function returns 4, which corresponds to the vanilla category! A reasonable guess.

```
from sklearn.naive_bayes import CategoricalNB

model = CategoricalNB()
model.fit(trainInput, trainLabels)

print(model.predict([ [0, 2] ])) # [4]
```

Testing the Model

We'll talk about testing models next time. To test our model in the code example, we just need to call the `score` method on the model using the test data we set aside earlier.

The result will be an accuracy between [0, 1]. We get an accuracy of 26.9% - not great, but also not terrible considering the amount of variety in the possible answers.

Note that we had to set aside the test data first for this to work! If we trained on the test data, we'd get a faulty result.

```
accuracy = model.score(testInput, testLabels)
print(accuracy) # 0.2692307692307692
```