NOTES 1

Bag of words: A technique used to extract features from the text. It counts how many times a word appears in a document (corpus), and then transforms that information into a dataset.

A **categorical** label has a discrete set of possible values, such as "is a cat" and "is not a cat."

Clustering. Unsupervised learning task that helps to determine if there are any naturally occurring groupings in the data.

CNN: Convolutional Neural Networks (CNN) represent nested filters over grid-organized data. They are by far the most commonly used type of model when processing images.

A **continuous (regression)** label does not have a discrete set of possible values, which means possibly an unlimited number of possibilities.

Data vectorization: A process that converts non-numeric data into a numerical format so that it can be used by a machine learning model.

Discrete: A term taken from statistics referring to an outcome taking on only a finite number of values (such as days of the week).

FFNN: The most straightforward way of structuring a neural network, the Feed Forward Neural Network (FFNN) structures neurons in a series of layers, with each neuron in a layer containing weights to all neurons in the previous layer.

Hyperparameters are settings on the model which are not changed during training but can affect how quickly or how reliably the model trains, such as the number of clusters the model should identify.

Log loss is used to calculate how uncertain your model is about the predictions it is generating.

Hyperplane: A mathematical term for a surface that contains more than two planes.

Impute is a common term referring to different statistical tools which can be used to calculate missing values from your dataset.

label refers to data that already contains the solution.

loss function is used to codify the model's distance from this goal

Machine learning, or ML, is a modern software development technique that enables computers to solve problems by using examples of real-world data.

Model accuracy is the fraction of predictions a model gets right. Discrete: A term taken from statistics referring to an outcome taking on only a finite number of values (such as days of the week). Continuous: Floating-point values with an infinite range of possible values. The opposite of categorical or discrete values, which take on a limited number of possible values.

Model inference is when the trained model is used to generate predictions.

model is an extremely generic program, made specific by the data used to train it.

Model parameters are settings or configurations the training algorithm can update to change how the model behaves.

Model training algorithms work through an interactive process where the current model iteration is analyzed to determine what changes can be made to get closer to the goal. Those changes are made and the iteration continues until the model is evaluated to meet the goals.

Neural networks: a collection of very simple models connected together. These simple models are called **neurons**. The connections between these models are trainable model parameters called **weights**.

Outliers are data points that are significantly different from others in the same sample.

Plane: A mathematical term for a flat surface (like a piece of paper) on which two points can be joined by a straight line.

Regression: A common task in supervised machine learning.

In **reinforcement learning**, the algorithm figures out which actions to take in a situation to maximize a reward (in the form of a number) on the way to reaching a specific goal.

RNN/LSTM: Recurrent Neural Networks (RNN) and the related Long Short-Term Memory (LSTM) model types are structured to effectively represent for loops in traditional computing, collecting state while iterating over some object. They can be used for processing sequences of data.

Silhouette coefficient: A score from -1 to 1 describing the clusters found during modeling. A score near zero indicates overlapping clusters, and scores less than zero indicate data points assigned to incorrect clusters. A

Stop words: A list of words removed by natural language processing tools when building your dataset. There is no single universal list of stop words used by all-natural language processing tools.

In **supervised learning,** every training sample from the dataset has a corresponding label or output value associated with it. As a result, the algorithm learns to predict labels or output values.

Test dataset: The data withheld from the model during training, which is used to test how well your model will generalize to new data.

Training dataset: The data on which the model will be trained. Most of your data will be here.

Transformer: A more modern replacement for RNN/LSTMs, the transformer architecture enables training over larger datasets involving sequences of data.

In **unlabeled data**, you don't need to provide the model with any kind of label or solution while the model is being trained.

In **unsupervised learning**, there are no labels for the training data. A machine learning algorithm tries to learn the underlying patterns or distributions that govern the data.

Data collection

Data collection can be as straightforward as running the appropriate SQL queries or as complicated as building custom web scraper applications to collect data for your project. You might even have to run a model over your data to generate needed labels. Here is the fundamental question:

> Does the data you've collected match the machine learning task and problem you have defined?

Data inspection

The quality of your data will ultimately be the largest factor that affects how well you can expect your model to perform. As you inspect your data, look for:

- Outliers
- Missing or incomplete values
- Data that needs to be transformed or preprocessed so it's in the correct format to be used by your model

Summary statistics

Models can assume how your data is structured.

Now that you have some data in hand it is a good best practice to check that your data is in line with the underlying assumptions of your chosen machine learning model.

With many statistical tools, you can calculate things like the mean, inner-quartile range (IQR), and standard deviation. These tools can give you insight into the *scope*, *scale*, and *shape* of the dataset.

Data visualization

You can use data visualization to see outliers and trends in your data and to help stakeholders understand your data.

Look at the following two graphs. In the first graph, some data seems to have clustered into different groups. In the second graph, some data

points might be outliers.

Terminology

- *Impute* is a common term referring to different statistical tools which can be used to calculate missing values from your dataset.
- *Outliers* are data points that are significantly different from others in the same sample.

Additional reading

- In machine learning, you use several statistical-based tools to better understand your data. The `sklearn` library has many examples and tutorials, such as this example demonstrating outlier detection on a real dataset.

Extended Learning

This information hasn't been covered in the above video but is provided for the advanced reader.

Linear models

One of the most common models covered in introductory coursework, linear models simply describe the relationship between a set of input numbers and a set of output numbers through a linear function (think of $y = mx + b$ or a line on a x vs y chart).

Classification tasks often use a strongly related logistic model, which adds an additional transformation mapping the output of the linear function to the range [0, 1], interpreted as “probability of being in the target class.” Linear models are fast to train and give you a great baseline against which to compare more complex models. A lot of media

buzz is given to more complex models, but for most new problems, consider starting with a simple model.

Tree-based models

Tree-based models are probably the second most common model type covered in introductory coursework. They learn to categorize or regress by building an extremely large structure of nested *if/else blocks*, splitting the world into different regions at each if/else block. Training determines exactly where these splits happen and what value is assigned at each leaf region.

For example, if you're trying to determine if a light sensor is in sunlight or shadow, you might train tree of depth 1 with the final learned configuration being something like *if (sensor_value > 0.698), then return 1; else return 0*;.

The tree-based model XGBoost is commonly used as an off-the-shelf implementation for this kind of model and includes enhancements beyond what is discussed here. Try tree-based models to quickly get a baseline before moving on to more complex models.

Deep learning models

Extremely popular and powerful, deep learning is a modern approach based around a conceptual model of how the human brain functions. The model (also called a *neural network*) is composed of collections of *neurons* (very simple computational units) connected together by *weights* (mathematical representations of how much information to allow to flow from one neuron to the next). The process of training involves finding values for each weight.

Various neural network structures have been determined for modeling different kinds of problems or processing different kinds of data.

A short (but not complete!) list of noteworthy examples includes:

- **FFNN**: The most straightforward way of structuring a neural network, the Feed Forward Neural Network (FFNN) structures neurons in a series of layers, with each neuron in a layer containing weights to all neurons in the previous layer.

- **CNN**: Convolutional Neural Networks (CNN) represent nested filters over grid-organized data. They are by far the most commonly used type of model when processing images.
- **RNN/LSTM**: Recurrent Neural Networks (RNN) and the related Long Short-Term Memory (LSTM) model types are structured to effectively represent *for loops* in traditional computing, collecting state while iterating over some object. They can be used for processing sequences of data.
- **Transformer**: A more modern replacement for RNN/LSTMs, the transformer architecture enables training over larger datasets involving sequences of data.

Machine Learning Using Python Libraries

- For more classical models (linear, tree-based) as well as a set of common ML-related tools, take a look at `scikit-learn`. The web documentation for this library is also organized for those getting familiar with space and can be a great place to get familiar with some extremely useful tools and techniques.
- For deep learning, `mxnet`, `tensorflow`, and `pytorch` are the three most common libraries. For the purposes of the majority of machine learning needs, each of these is feature-paired and equivalent.

Additional reading

- The Wikipedia entry on the bias-variance trade-off can help you understand more about this common machine learning concept.
- In this AWS Machine Learning blog post, you can see how to train a machine-learning algorithm to predict the impact of weather on air quality using Amazon SageMaker.

Using Log Loss

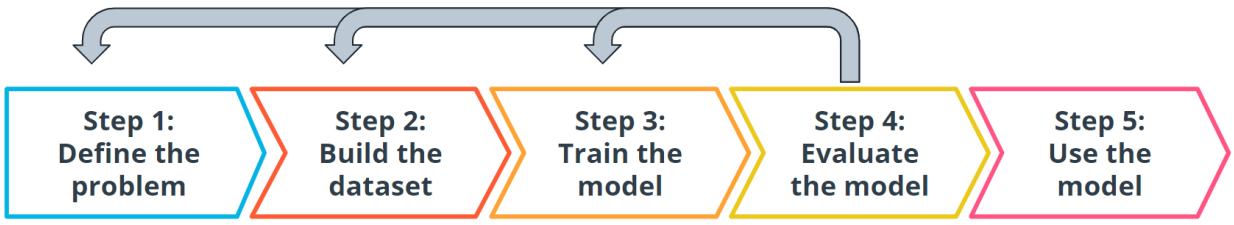
*Log los*s seeks to calculate how *uncertain* your model is about the predictions it is generating. In this context, uncertainty refers to how likely a model thinks the predictions being generated are to be correct.

For example, let's say you're trying to predict how likely a customer is to buy either a jacket or t-shirt.

Log loss could be used to understand your model's uncertainty about a given prediction. In a single instance, your model could predict with 5% certainty that a customer is going to buy a t-shirt. In another instance, your model could predict with 80% certainty that a customer is going to buy a t-shirt. Log loss enables you to measure how strongly the model believes that its prediction is accurate.

In both cases, the model predicts that a customer will buy a t-shirt, but the model's certainty about that prediction can change.

Remember: This Process is Iterative



Iterative steps of machine learning

Every
step we have gone through is highly iterative and can be changed or re-scoped during the course of a project. At each step, you might find that you need to go back and reevaluate some assumptions you had in previous steps. Don't worry! This ambiguity is normal.

Terminology

Log loss seeks to calculate how *uncertain* your model is about the predictions it is generating.

Model Accuracy is the fraction of predictions a model gets right.

Additional reading

The tools used for model evaluation are often tailored to a specific use case, so it's difficult to generalize rules for choosing them. The following articles provide use cases and examples of specific metrics in use.

1. This healthcare-based example, which automates the prediction of spinal pathology conditions,
   demonstrates how important it is to avoid false positive and false negative predictions using the tree-based `xgboost` model.
2. The popular open-source library `sklearn` provides information about common metrics and how to use them.
3. This entry from the AWS Machine Learning blog demonstrates the importance of choosing the correct model evaluation
   metrics for making accurate energy consumption estimates using Amazon Forecast.

Terminology

- **Continuous**: Floating-point values with an infinite
  range of possible values. The opposite of categorical or discrete

values, which take on a limited number of possible values.

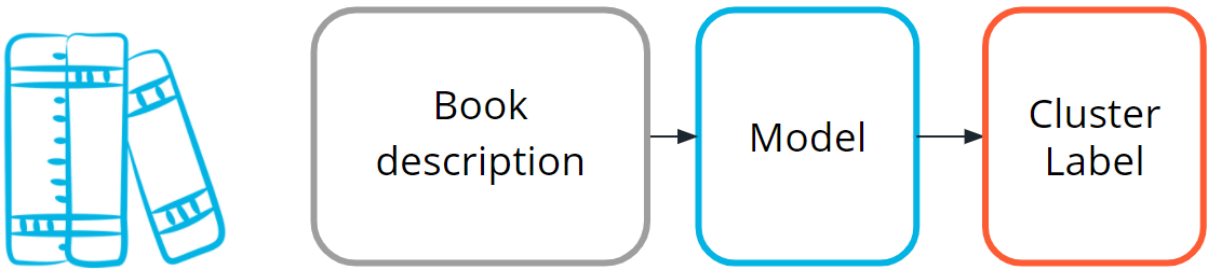
- **Hyperplane**: A mathematical term for a surface that contains more than two planes.
- **Plane**: A mathematical term for a flat surface (like a piece of paper) on which two points can be joined by a straight line.
- **Regression**: A common task in supervised machine learning.

Additional reading

The Machine Learning Mastery
blog is a fantastic resource for learning more about machine learning.
The following example blog posts dive deeper into training
regression-based machine learning models.

- How to Develop Ridge Regression Models in Python offers another approach to solving the problem in the example from this lesson.
- Regression is a popular machine learning task, and you can use several different model evaluation metrics with it.

Step One: Define the Problem



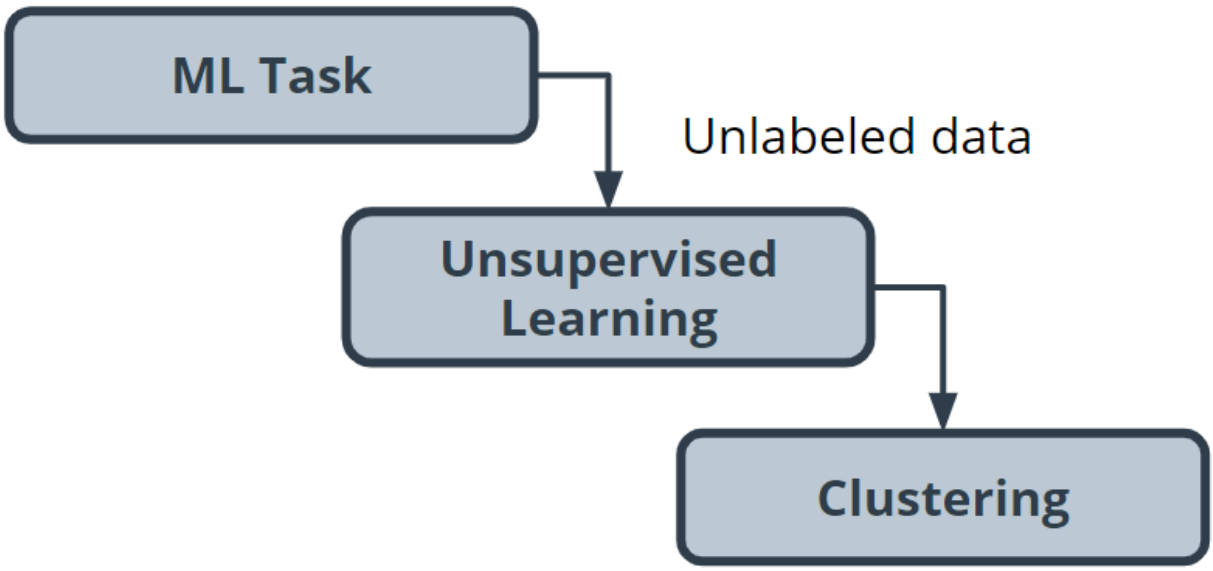
Model used to predict micro-genres

> Find clusters of similar books based on the presence of common words in the book descriptions.

You do editorial work for a book recommendation company, and you want to write an article on the largest book trends of the year. You believe that a trend called "micro-genres" exists, and you have confidence that you can use the book description text to identify these micro-genres.

By using an unsupervised machine learning technique called *clustering*, you can test your hypothesis that the book description text can be used to identify these "hidden" micro-genres.

Earlier in this lesson, you were introduced to the idea of unsupervised learning. This machine learning task is especially useful when your data is not labeled.



Unsupervised learning using clustering

Step Two: Build your Dataset

To test the hypothesis, you gather book description text for 800 romance books published in the current year.

Data exploration, cleaning and preprocessing

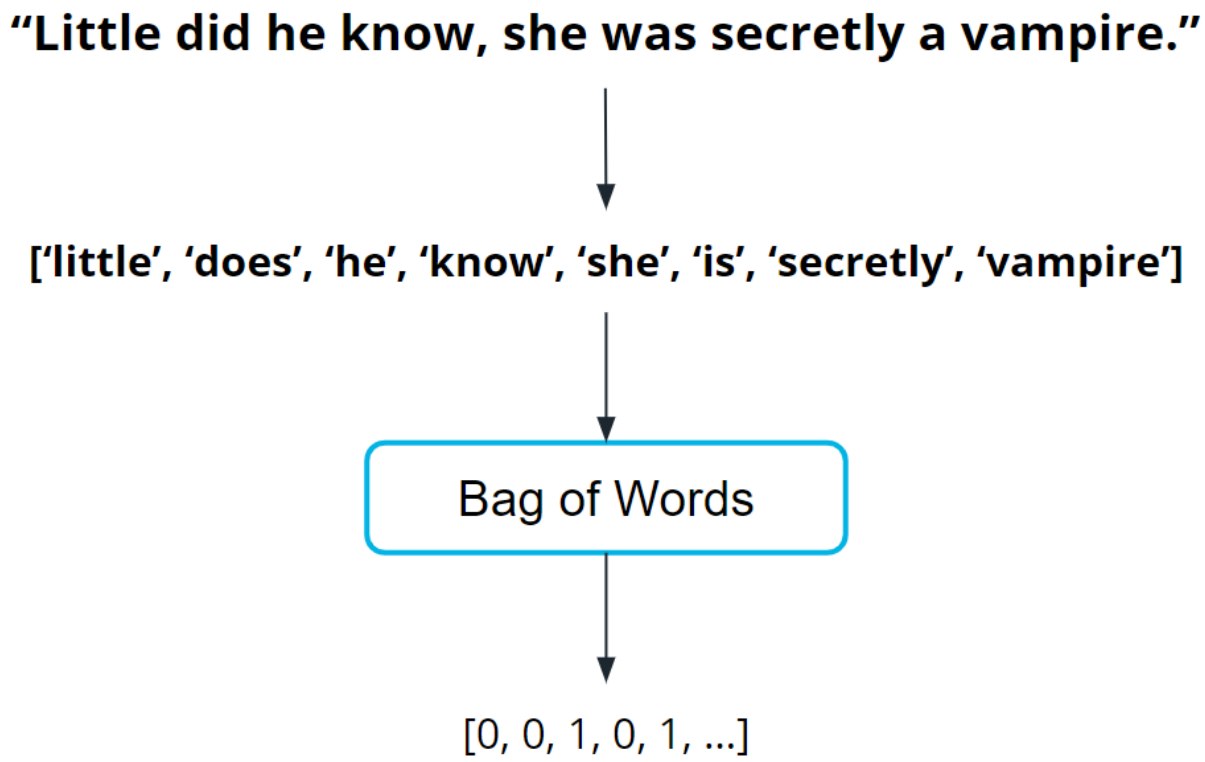
For this project, you believe capitalization and verb tense will not matter, and therefore you remove capitals and convert all verbs to the same tense using a Python library built for processing human language.

You also remove punctuation and words you don't think have useful meaning, like '*a*' and '*the*'. The machine learning community refers to these words as *stop words*.

Before you can train the model, you need to do some data preprocessing, called *data vectorization*, to convert text into numbers.

You transform this book description text into what is called a `bag of words` representation shown in the following image so that it is understandable by machine learning models.

How the bag of words representation works is beyond the scope of this course. If you are interested in learning more, see the **Additional Reading** section at the bottom of the page.



Step Three: Train the Model

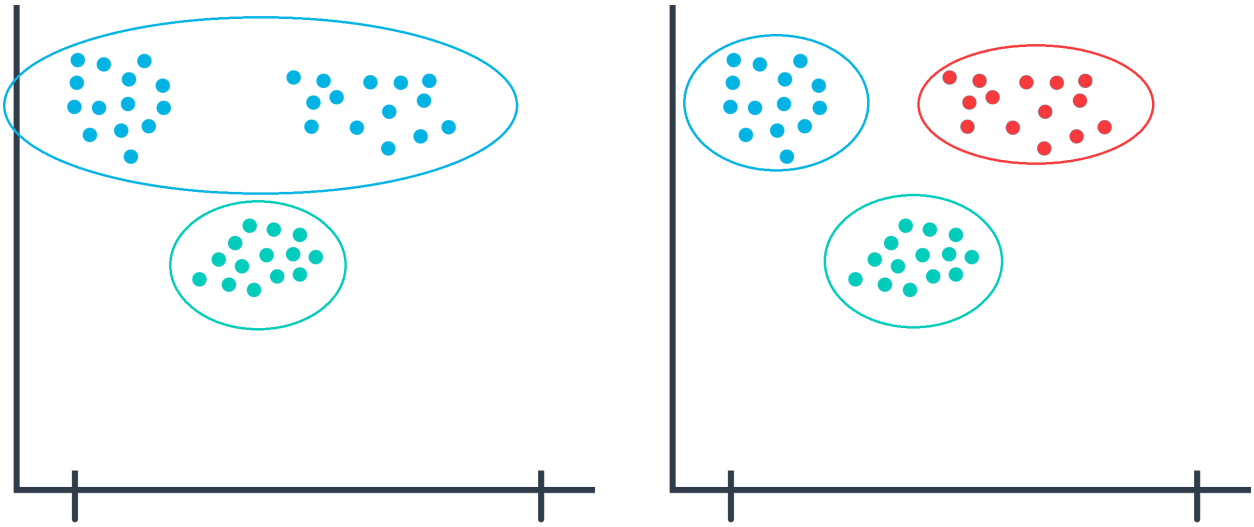
Now you are ready to train your model.

You pick a common cluster-finding model called `k-means`. In this model, you can change a model parameter, `k`, to be equal to how many clusters the model will try to find in

your dataset.

Your data is unlabeled: you don't how many microgenres might exist.
So you train your model multiple times using different values for `k` each time.

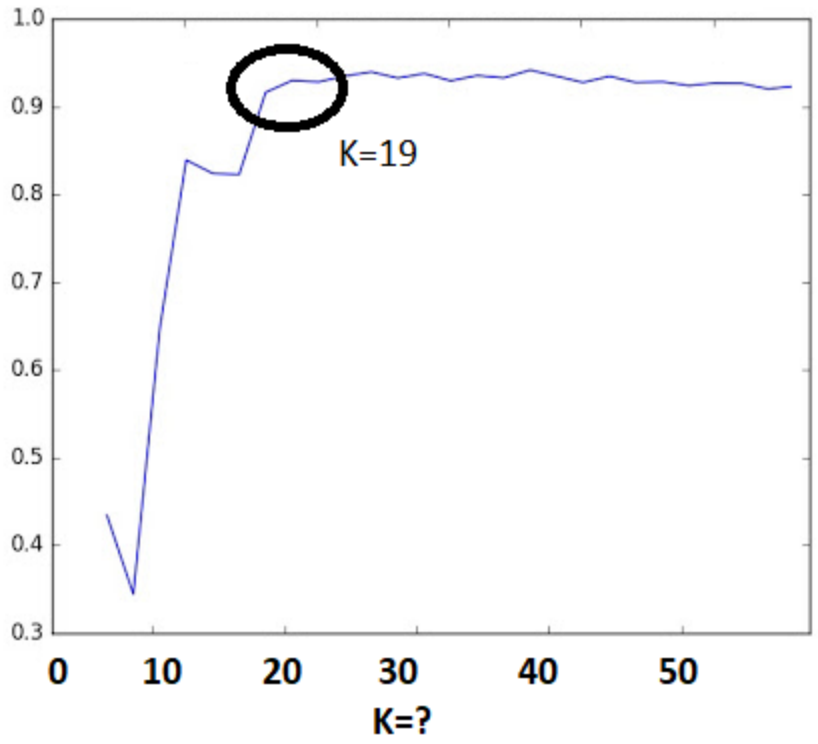
What does this even mean? In the following graphs, you can see examples of when `k=2` and when `k=3`.



During the model evaluation phase, you plan on using a metric to find which value for **k** is most appropriate.

Step Four: Model Evaluation

In machine learning, numerous statistical metrics or methods are available to evaluate a model. In this use case, the *silhouette coefficient* is a good choice. This metric describes how well your data was clustered by the model. To find the optimal number of clusters, you plot the silhouette coefficient as shown in the following image below. You find the optimal value is when `k=19`.



Optimum number (k=19) of clusters

Often, machine learning practitioners do a manual evaluation of the model's findings.

You find one cluster that contains a large collection of books you can categorize as “paranormal teen romance.” This trend is known in your industry, and therefore you feel somewhat confident in your machine learning approach. You don’t know if every cluster is going to be as cohesive as this, but you decide to use this model to see if you can find anything interesting about which to write an article.

Step Five: Inference (Use the Model)

As you inspect the different clusters found when `k=19`, you find a surprisingly large cluster of books. Here's an example from fictionalized cluster #7.

Cluster Label	Book Description
7	“Susan’s crush just moved away..”
7	“Can Alice and Bob keep their relationship together three hundred miles apart?”
7	“When Hank’s fiance George got offered a new job in New York...”

Clustered data

As you inspect the preceding table, you can see that most of these text snippets are indicating that the characters are in some kind of long-distance relationship. You see a few other self-consistent clusters and feel you now have enough useful data to begin writing an article on unexpected modern romance microgenres.

Terminology

- **Bag of words**: A technique used to extract features from the text. It counts how many times a word appears in a document (corpus), and then transforms that information into a dataset.
- **Data vectorization**: A process that converts non-numeric data into a numerical format so that it can be used by a machine learning model.
- **Silhouette coefficient**: A score from -1 to 1 describing the clusters found during modeling. A score near zero indicates overlapping clusters, and scores less than zero indicate data points assigned to incorrect clusters. A score approaching 1 indicates successful identification of discrete non-overlapping clusters.
- **Stop words**: A list of words removed by natural language processing tools when building your dataset. There is no single universal

list of stop words used by all-natural language processing tools.

Additional reading

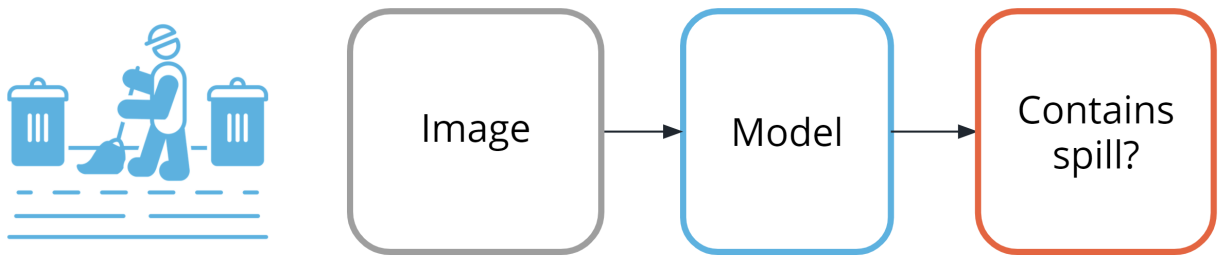
Machine Learning Mastery is a great resource for finding examples of machine learning projects.

- The How to Develop a Deep Learning Bag-of-Words Model for Sentiment Analysis (Text Classification) blog post provides an example using a bag of words–based approach pair with a deep learning model.

Step One: Defining the Problem

Imagine you run a company that offers specialized on-site janitorial services. A client, an industrial chemical plant, requires a fast response for spills and other health hazards. You realize if you could *automatically* detect spills using the plant's surveillance system, you could mobilize your janitorial team faster.

Machine learning could be a valuable tool to solve this problem.



Detecting spills with machine learning

Step Two: Model Training (and selection)

This task is a supervised classification task, as shown in the following image. As shown in the image above, your goal will be to predict if each image belongs to one of the following classes:

- **Contains spill**

- **Does not contain spill**

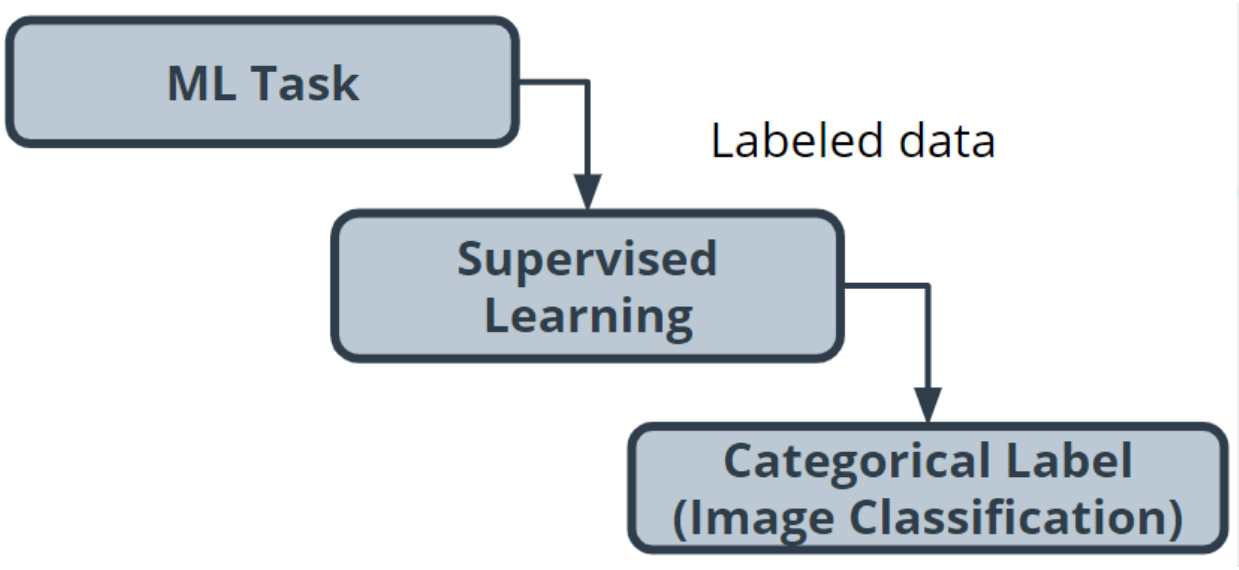


Image classification

Step Two: Building a Dataset

- **Collecting**
  - Using historical data, as well as safely staged spills, you quickly build a collection of images that contain both spills and non-spills in multiple lighting conditions and environments.
- **Exploring and cleaning**
  - You go through all the photos to ensure the spill is clearly in the shot. There are Python tools and other techniques available to improve image quality, which you can use later if you determine a need to iterate.
- **Data vectorization** (converting to numbers)
  - Many models require numerical data, so all your image data needs to be transformed into a numerical format. Python tools can help you do this automatically.
  - In the following image, you can see how each pixel in the image on the left can be represented in the image on the right by a number

between 0 and 1, with 0 being completely black and 1 being completely white.

```
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1
1 1 1 1 1 1 0 0 1 0 0 0 0 0 0 0
1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 0 0 0 0 0 0 0 0 0 1 1 1
1 1 1 1 0 0 0 0 0 0 0 0 0 1 1 1
1 1 1 0 0 0 0 0 0 0 0 0 0 1 1 1
1 1 1 0 0 0 0 0 0 0 0 0 0 1 1 1
1 1 0 0 0 0 0 0 0 0 0 0 1 1 1 1
1 1 0 0 0 0 0 0 0 0 0 0 1 1 1 1
1 1 0 0 0 0 0 0 0 0 0 0 1 1 1 1
1 1 0 0 0 0 0 0 0 0 0 1 1 1 1 1
1 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1
0 0 0 0 0 0 0 0 1 0 0 0 0 1 1 1
0 0 0 0 0 0 0 0 1 1 0 0 0 1 1 1
0 0 0 0 0 0 0 1 1 1 1 0 0 1 1 1
0 0 0 0 0 0 1 1 1 1 1 0 1 1 1 1
0 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

Split the data

- You split your image data into a training dataset and a test dataset.

Step Three: Model Training

Traditionally, solving this problem would require hand-engineering features on top of the underlying pixels (for example, locations of prominent edges and corners in the image), and then training a model on these features.

Today, deep neural networks are the most common tool used for solving this kind of problem. Many deep neural network models are structured to learn the features on top of the underlying pixels so you don’t have to learn them. You’ll have a chance to take a deeper look at this in the next lesson, so we’ll keep things high-level for now.

CNN (convolutional neural network)

Neural networks are beyond the scope of this lesson, but you can think of them as a collection of very simple models connected together. These simple models are called *neurons*, and the connections between these models are trainable model parameters called *weights.*

Convolutional neural networks are a special type of neural network particularly good at processing images.

Step Four: Model Evaluation

As you saw in the last example, there are many different statistical metrics you can use to evaluate your model. As you gain more experience in machine learning, you will learn how to research which metrics can help you evaluate your model most effectively. Here's a list of common metrics:

Aa Accuracy	False positive rate	Precision
Confusion matrix	False negative rate	Recall
F1 Score	Log Loss	ROC curve
Untitled	Negative predictive value	Specificity

In cases such as this, accuracy might not be the best evaluation mechanism.

Why not? You realize the model will see the '**Does not contain spill'** class almost all the time, so any model that just predicts "**no spill**" most of the time will seem pretty accurate.

> What you really care about is an evaluation tool that rarely misses a real spill.

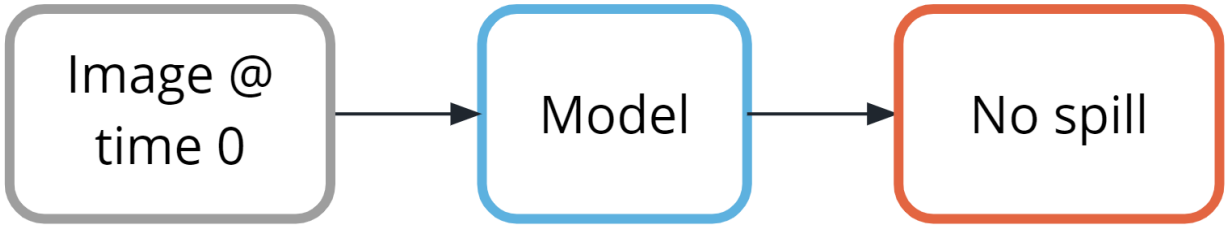
After doing some internet sleuthing, you realize this is a common problem and that `Precision` and `Recall` will be effective. You can think of *precision* as answering the question, "Of all predictions of a spill, how many were right?" and *recall* as answering the question, "Of all actual spills, how many did we detect?"

Manual evaluation plays an important role. You are unsure if your staged spills are sufficiently realistic compared to actual spills. To get a better sense how well your model performs with actual spills, you find additional examples from historical records. This allows you to confirm that your model is performing satisfactorily.

Step Five: Model Inference

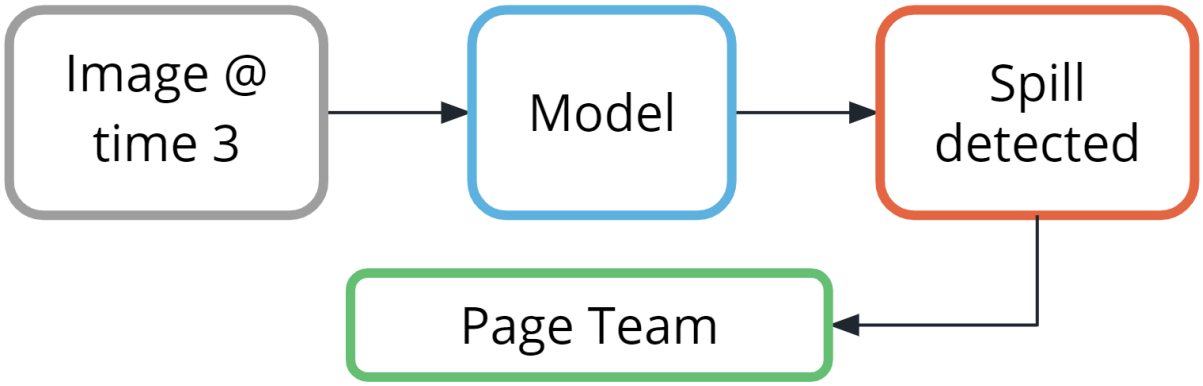
The model can be deployed on a system that enables you to run machine learning workloads such as AWS Panorama.

Thankfully, most of the time, the results will be from the class '**Does not contain spill.**'



No spill detected

But, when the class '**Contains spill**' is detected, a simple paging system could alert the team to respond.



Spill detected

Terminology

Convolutional neural networks(CNN) are a special type of neural network particularly good at processing images.

Neural networks: a collection of very simple models connected together.

- These simple models are called **neurons**
- the connections between these models are trainable model parameters called **weights.**

Additional reading

As you continue your machine learning journey, you will start to recognize problems that are excellent candidates for machine learning.

The AWS Machine Learning Blog is a great resource for finding more examples of machine learning projects.

- In the Protecting people from hazardous areas through virtual boundaries with Computer Vision blog post, you can see a more detailed example of the deep learning process described in this lesson.

Machine learning step
Action taken at this step

Step 1: Define the problem

Thinking of this problem as a classification task.

Step 2: Build the dataset

Flipping through photos to ensure the spill is clearly in shot.

Step 3: Train the model

Identifying a CNN as having a good chance of matching your data and task.

Step 4: Evaluate the model

Measuring model accuracy alone won't give you confidence that the trained model is performing as intended.

Step 5: Use the model

Deploying the model to a system capable of processing images in the surveillance system.