Introduction to Machine Learning

A machine learning model, like a piece of clay, can be molded into many different forms and serve many different purposes. A more technical definition would be that a machine learning model is a block of code or framework that can be modified to solve different but related problems based on the data provided.

Important

A model is an extremely generic program (or block of code), made specific by the data used to train it. It is used to solve different problems.

How are model training algorithms used to train a model?

In the preceding section, we talked about two key pieces of information: a model and data. In this section, we show you how those two pieces of information are used to create a trained model. This process is called *model training*.

Model training algorithms work through an interactive process

Let's revisit our clay teapot analogy. We've gotten our piece of clay, and now we want to make a teapot. Let's look at the algorithm for molding clay and how it resembles a machine learning algorithm:

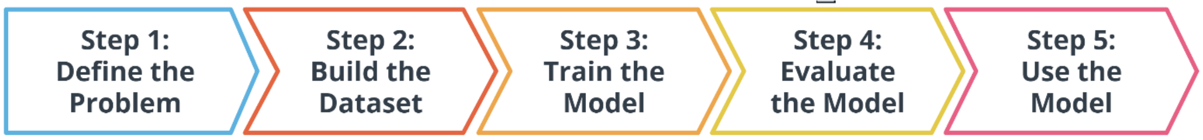
- **Think about the changes that need to be made.** The first thing you would do is inspect the raw clay and think about what changes can be made to make it look more like a teapot. Similarly, a model training algorithm uses the model to process data and then compares the results against some end goal, such as our clay teapot.
- **Make those changes.** Now, you mold the clay to make it look more like a teapot. Similarly, a model training algorithm gently nudges specific parts of the model in a direction that brings the model closer to achieving the goal.

- **Repeat.** By iterating these steps over and over, you get closer and closer to what you want, until you determine that you're close enough and then you can stop.

After finishing this lesson, you will have been introduced to the key terms and the most common techniques used to solve problems in machine learning.

- A **model** is an extremely generic program, made specific by the data used to train it.
- **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.
- **Model inference** is when the trained model is used to generate predictions.

5 Steps in Machine Learning Workflow


Step 1: Define the Problem
Step 2: Build the Dataset
Step 3: Train the Model
Step 4: Evaluate the Model
Step 5: Use the Model

Define the Problem

How do you start a machine learning task?

Step 1: Define a very specific task

Think back to the snow cone sales example. Now imagine that you own a frozen treats store and you sell snow cones along with many other products. You wonder, "How do I increase sales?" It's a valid question, but it's the opposite of a very specific task. The following examples demonstrate how a machine learning practitioner might attempt to answer that question.

Question: Does adding a $1.00 charge for sprinkles on a hot fudge sundae increase the sales of hot fudge sundaes?

Question: Does adding a $0.50 charge for organic flavors in your snow cone increase the sales of snow cones?

Step 2: Identify the machine learning task we might use to solve this problem

This helps you better understand the data you need for a project.

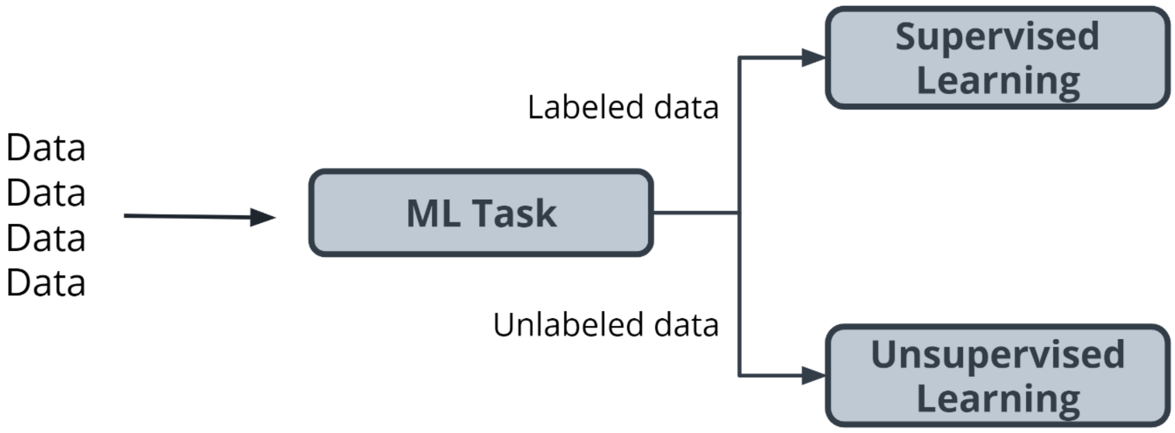
What exactly is a machine learning task?

All model training algorithms, and the models themselves, take data as their input. Their outputs can be very different and are classified into a few different groups, based on the task they are designed to solve.
Often, we use the kind of data required to train a model as part of defining a machine learning task.

In this lesson, we will focus on two common machine learning tasks:

- Supervised learning
- Unsupervised learning

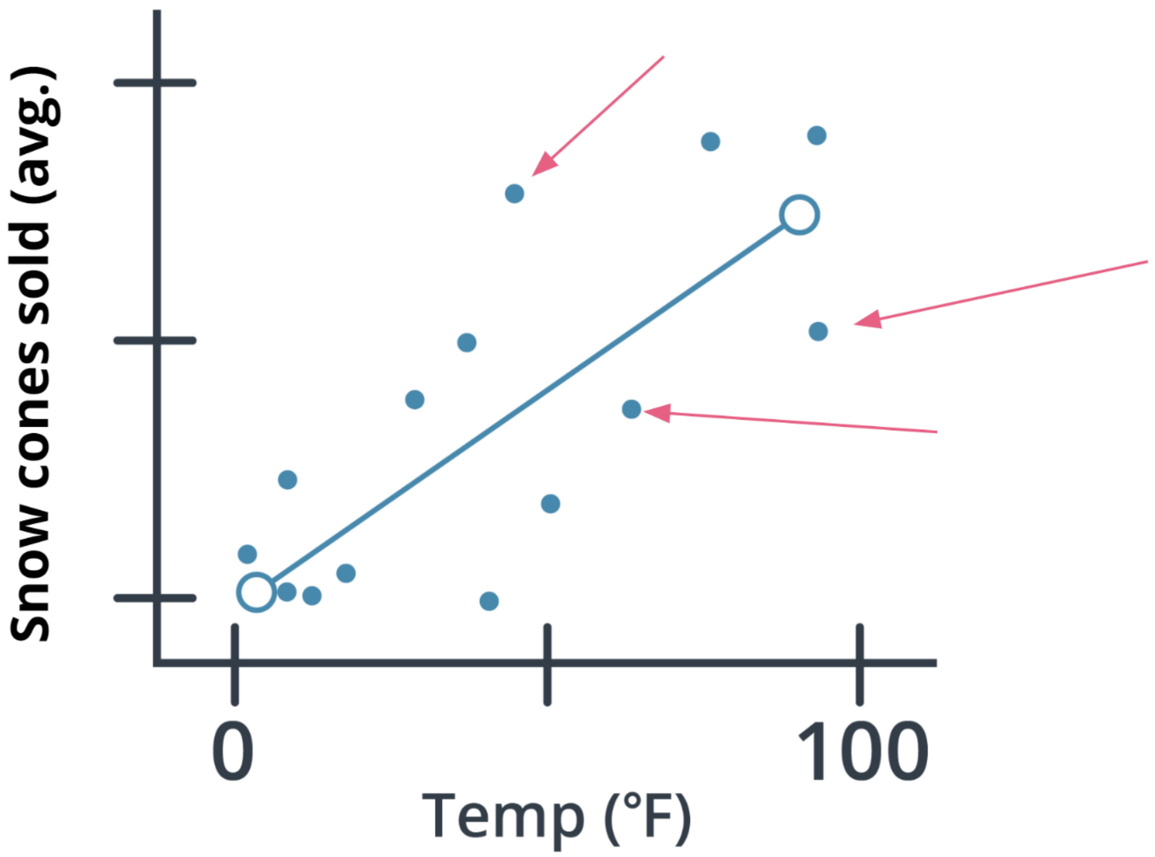
Supervised and Unsupervised learning



The presence or absence of labeling in your data is often used to identify a machine learning task.

Supervised tasks

A task is supervised if you are using labeled data. We use the term labeled to refer to data that already contains the solutions, called labels.
For example, predicting the number of snow cones sold based on the average temperature outside is an example of supervised learning.



In the preceding graph, the data contains both a temperature and the number of snow cones sold. Both components are used to generate the linear regression shown on the graph. Our goal was to predict the number of snow cones sold, and we feed that value into the model. We are providing the model with labeled data and therefore, we are performing a supervised machine learning task

Unsupervised tasks

- Take a look at the preceding picture. Did you notice the tree in the picture? What you just did, when you noticed the object in the picture and identified it as a tree, is called labeling the picture. Unlike you, a computer just sees that image as a matrix of pixels of varying intensity.
- Since this image does not have the labeling in its original data, it is considered unlabeled.

How do we further classify tasks when we don't have a label?

Unsupervised learning involves using data that doesn't have a label. One common task is called clustering. Clustering helps to determine if there are any naturally occurring groupings in the data.

Let's look at an example of how clustering works in unlabeled data.

Example: Identifying book micro-genres with unsupervised learning

Imagine that you work for a company that recommends books to readers.

The assumption is that you are fairly confident that micro-genres exist, and that there is one called Teen Vampire Romance. However, you don't know which micro-genres exist specifically, so you can't use supervised learning techniques.

This is where the unsupervised learning clustering technique might be able to detect some groupings in the data. The words and phrases used in a book's description might provide some guidance on its micro-genre.

Classifying based on Label Type

Initially, we divided tasks based on the presence or absence of labeled data while training our model. Often, tasks are further defined by the type of label that is present.

In **supervised learning,** there are two main identifiers that you will see in machine learning:

- A **categorical label** has a discrete set of possible values. In a machine learning problem in which you want to identify the type of flower based on a picture, you would train your model using images that have been labeled with the categories of the flower that you want to identify. Furthermore, when you work with categorical labels, you often carry out classification tasks, which are part of the supervised learning family.
- A **continuous (regression)** label does not have a discrete set of possible values, which often means you are working with numerical data. In the snow cone sales example, we are trying to predict the number of snow cones sold. Here, our label is a number that could, in theory, be any value.

In **unsupervised learning,** clustering is just one example. There are many other options, such as deep learning.

Wrap Up

In this module, you learned about labeled data versus unlabeled data, and how that plays a role in defining the machine learning task you are going to use. Furthermore, you started to explore which machine learning tasks are used to solve certain kinds of machine learning problems.
Here's a quick recap of the terms introduced in this lesson:

- **Clustering** is an unsupervised learning task that helps to determine if there are any naturally occurring groupings in the data.
- A **categorical label** has a discrete set of possible values, such as "is a cat" and "is not a cat."
- A **continuous (regression) label** does not have a discrete set of possible values, which means there are potentially an unlimited number of possibilities.
- **Discrete** is a term taken from statistics referring to an outcome that takes only a finite number of values (such as days of the week).
- A **label** refers to data that already contains the solution.
- Using **unlabeled data** means you don't need to provide the model with any kind of label or solution while the model is being trained.

Build the Dataset

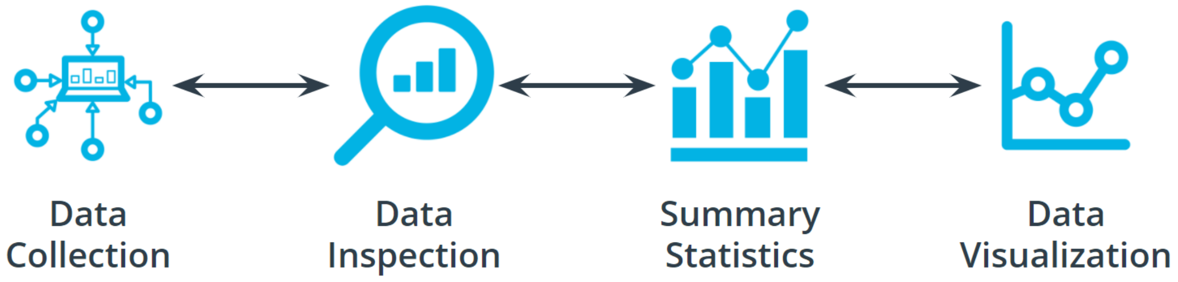
Building a dataset

The most important step of the machine learning process

Working with data is perhaps the most overlooked—yet most important—step of the machine learning process. In 2017, an O'Reilly study showed that machine learning practitioners spend 80% of their time working with their data.

The Four Aspects of Working with Data

You can take an entire class just on working with, understanding, and processing data for machine learning applications. Good, high-quality data is essential for any kind of machine learning project. Let's explore some of the common aspects of working with 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 make assumptions about how your data is structured.

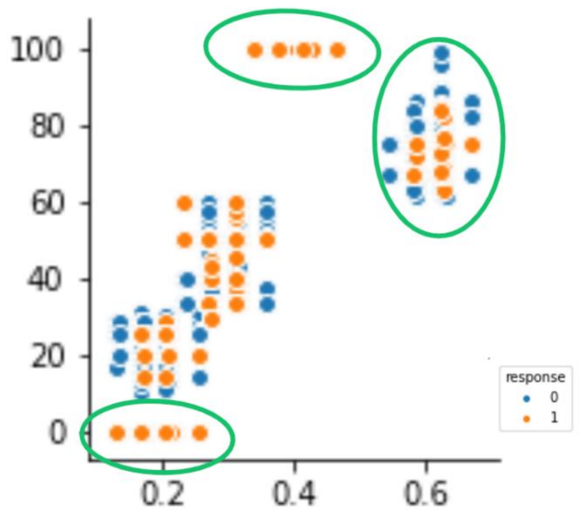
Now that you have some data in hand, it is a good best practice to check that your data is in line with the underlying assumptions of the machine learning model that you chose.

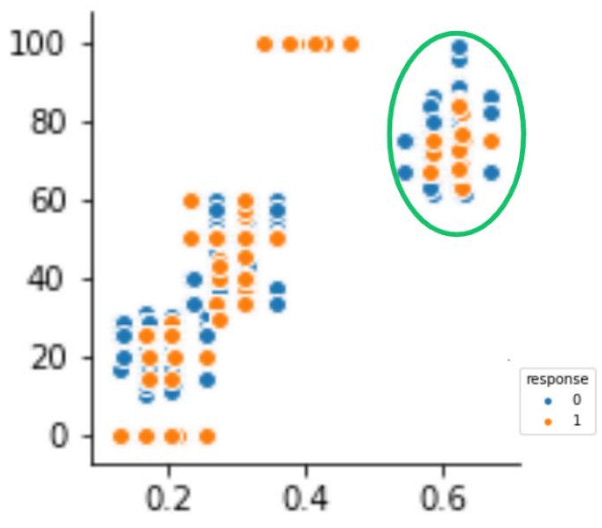
Using statistical tools, you can calculate things like the mean, inner-quartile range (IQR), and standard deviation. These tools can give you insights into the scope, scale, and shape of a dataset.

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 graph immediately preceding it, some data points might be outliers.





Wrap Up

1. You learned that having good data is key to being able to successfully answer the problem you have defined in your machine learning problem.

2. To build a good dataset, there are four key aspects to be considered when working with your data. First, you need to collect the data. Second, you should inspect your data to check for outliers, missing or incomplete values, and to see if any kind of data reformatting is required. Third, you should use summary statistics to understand the scope, scale, and shape of the dataset. Finally, you should use data visualizations to check for outliers, and to see trends in your data.

Key terms from this lesson:

- Impute is a common term referring to different statistical tools that can be used to calculate missing values from your dataset.
- Outliers are data points that are significantly different from other date 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 that demonstrates outlier detection on a real dataset

Train the Model

Model Training

Model training is a process whereby the model's parameters are iteratively updated to minimize some loss function that has been previously defined

Splitting your Dataset

The first step in model training is to randomly split the dataset.

This allows you to keep some data hidden during training, so that the data can be used to evaluate your model before you put it into production. Specifically, you do this to test against the bias-variance trade-off. If you're interested in learning more, see the optional **Extended learning** section.

Splitting your dataset gives you two sets of data:

- Training dataset: The data on which the model will be trained. Most of your data will be here. Many developers estimate about 80%.
- Test dataset: The data withheld from the model during training, which is used to test how well your model will generalize to new data.

Putting it all Together and Key Modelling Terms

Model training terms

The model training algorithm iteratively updates a model's parameters to minimize some loss function.

Let's define those two terms:

- **Model parameters:** Model parameters are settings or configurations that the training algorithm can update to change how the model behaves. Depending on the context, you'll also hear other specific terms used to describe model parameters such as weights and biases. Weights, which are values that change as the model learns, are more specific to neural networks.

- **Loss function:** A loss function is used to codify the model's distance from a goal. For example, if you were trying to predict the number of snow cone sales based on the day's weather, you would care about making predictions that are as accurate as possible. So you might define a loss function to be "the average distance between your model's predicted number of snow cone sales and the correct number." You can see in the snow cone example; this is the difference between the two purple dots.

Putting it all together

The end-to-end training process is:

- Feed the training data into the model.
- Compute the loss function on the results.
- Update the model parameters in a direction that reduces loss.

You continue to cycle through these steps until you reach a predefined stop condition. This might be based on training time, the number of training cycles, or an even more intelligent or application-aware mechanism.

Advice from Experts

Remember the following advice when training your model.

1. Practitioners often use machine learning frameworks that already have working implementations of models and model training algorithms. You could implement these from scratch, but you probably won't need to do so unless you're developing new models or algorithms.
2. Practitioners use a process called model selection to determine which model or models to use. The list of established models is constantly growing, and even seasoned machine learning practitioners try many different types of models while solving a problem with machine learning.
3. *Hyperparameters* are settings on the model that 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.
4. Be prepared to iterate.

Pragmatic problem solving with machine learning is rarely an exact science, and you might have assumptions about your data or problem that turn out to be false. Don't get discouraged. Instead, foster a habit of trying new things, measuring success, and comparing results across iterations.

Wrap Up

1. The model training algorithm iteratively updates a model's parameters to minimize some loss function.
2. During model training, the training data is fed into the model, and then the loss function is computed based on the results. The model parameters are then updated in a direction that reduces loss. You will continue to cycle through these steps until your reach a predefined stop condition.

Key terms from this lesson:

- **Hyperparameters** are settings on the model that 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.
- A **loss function** is used to codify the model's distance from this goal.
- **Training dataset:** The data on which the model will be trained. Most of your data will be here.
- **Test dataset:** The data withheld from the model during training, which is used to test how well your model will generalize to new data.
- **Model parameters** are settings or configurations the training algorithm can update to change how the model behaves.

Extended Learning (Optional)

This information wasn't covered in the video from the previous section, but it 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 that is 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 thst is allowed 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.

Evaluate the Model

Evaluating a trained model

After you have collected your data and trained a model, you can start to evaluate how well your model is performing. The metrics used for evaluation are likely to be very specific to the problem you defined. As you grow in your understanding of machine learning, you will be able to explore a wide variety of metrics that can enable you to evaluate effectively.

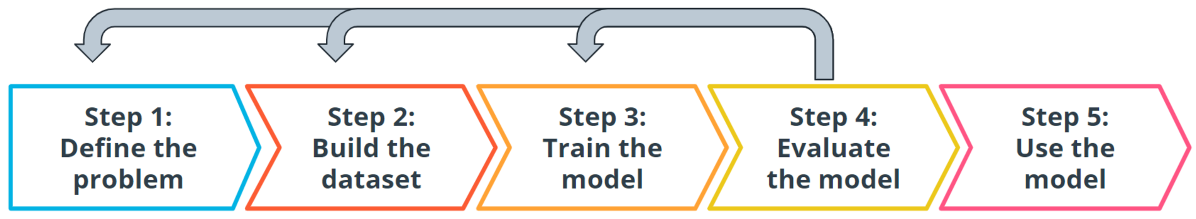
Model Accuracy

Model accuracy is a fairly common evaluation metric. *Accuracy* is the fraction of predictions a model gets right.

Here's a Example:

Imagine that you build a model to identify a flower as one of two common species based on measurable details like petal length. You want to know how often your model predicts the correct species. This would require you to look at your model's accuracy.

An Iterative Process


Step 1: Define the problem
Step 2: Build the dataset
Step 3: Train the model
Step 4: Evaluate the model
Step 5: Use the model

Every step we have gone through is highly iterative and can be changed or rescoped 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.

Wrap up

Good job completing this lesson. In this lesson, you were introduced to how models are evaluated. You also learned that the tools used to evaluate a model are often very specific to the problem defined in step one.

Extended Learning (Optional)

Using a Log Loss

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.

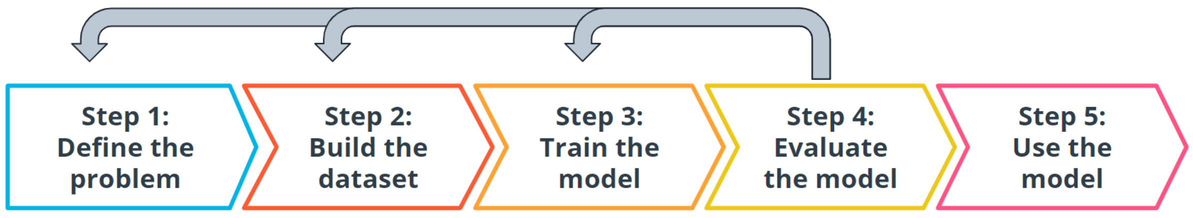
In both cases, the model predicts that a customer will buy a t-shirt, but the model's certainty about that prediction can change.

Use the Model

Model Inference

Congratulations! You're ready to deploy your model. Once you have trained your model, have evaluated its effectiveness, and are satisfied with the results, you're ready to generate predictions on real-world problems using unseen data in the field. In machine learning, this process is often called inference.

Machine Learning is Iterative


Step 1: Define the problem
Step 2: Build the dataset
Step 3: Train the model
Step 4: Evaluate the model
Step 5: Use the model

Even after you deploy your model, you're always monitoring to make sure your model is producing the kinds of results that you expect. There may be times where you reinvestigate the data, modify some of the parameters in your model training algorithm, or even change the model type used for training.

Wrap Up

1. Solving problems using machine learning is an evolving and iterative process.
2. To solve problem successfully in Machine Learning finding High quality data is essential.
3. To evaluate models, you often use statistical metrics. The metrics you choose are tailored to a specific use case.

Examples of Machine Learning

Case Studies that use Machine Learning

Through the remainder of the lesson, we will walk through three different case studies. In each example, you will see how machine learning can be used to solve real-world problems.

Supervised learning

- Using machine learning to predict housing prices in a neighborhood, based on lot size and the number of bedrooms.

Unsupervised learning

- Using machine learning to isolate micro-genres of books by analyzing the wording on the back cover description.

Deep neural network

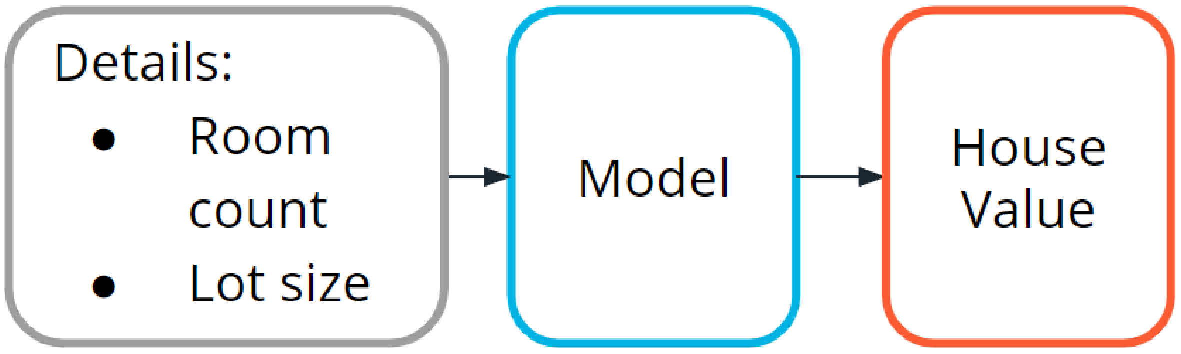
- You will see how it can be used to analyse raw images from lab video footage from security cameras, trying to detect chemical spills.

Example 1: Predicting Home Prices (Supervised Machine Learning)

House price prediction is one of the most common examples used to introduce machine learning.

Traditionally, real estate appraisers use many quantifiable details about a home (such as number of rooms, lot size, and year of construction) to help them estimate the value of a house.

You detect this relationship and believe that you could use machine learning to predict home prices.

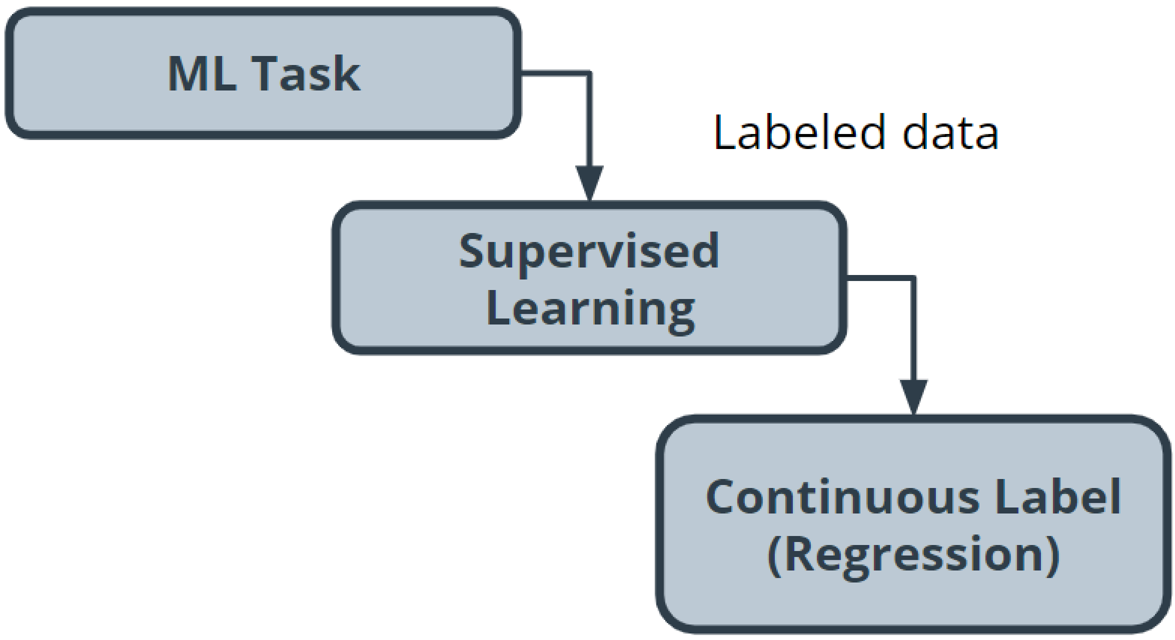


In the next sections, you will go through the 5 major steps in machine learning in the context of this example.

Step 1: Define the Problem

Can we estimate the price of a house based on lot size or the number of bedrooms?

You access the sale prices for recently sold homes or have them appraised. Since you have this data, this is a supervised learning task. You want to predict a continuous numeric value, so this task is also a regression task.



Step 2: Build the Dataset

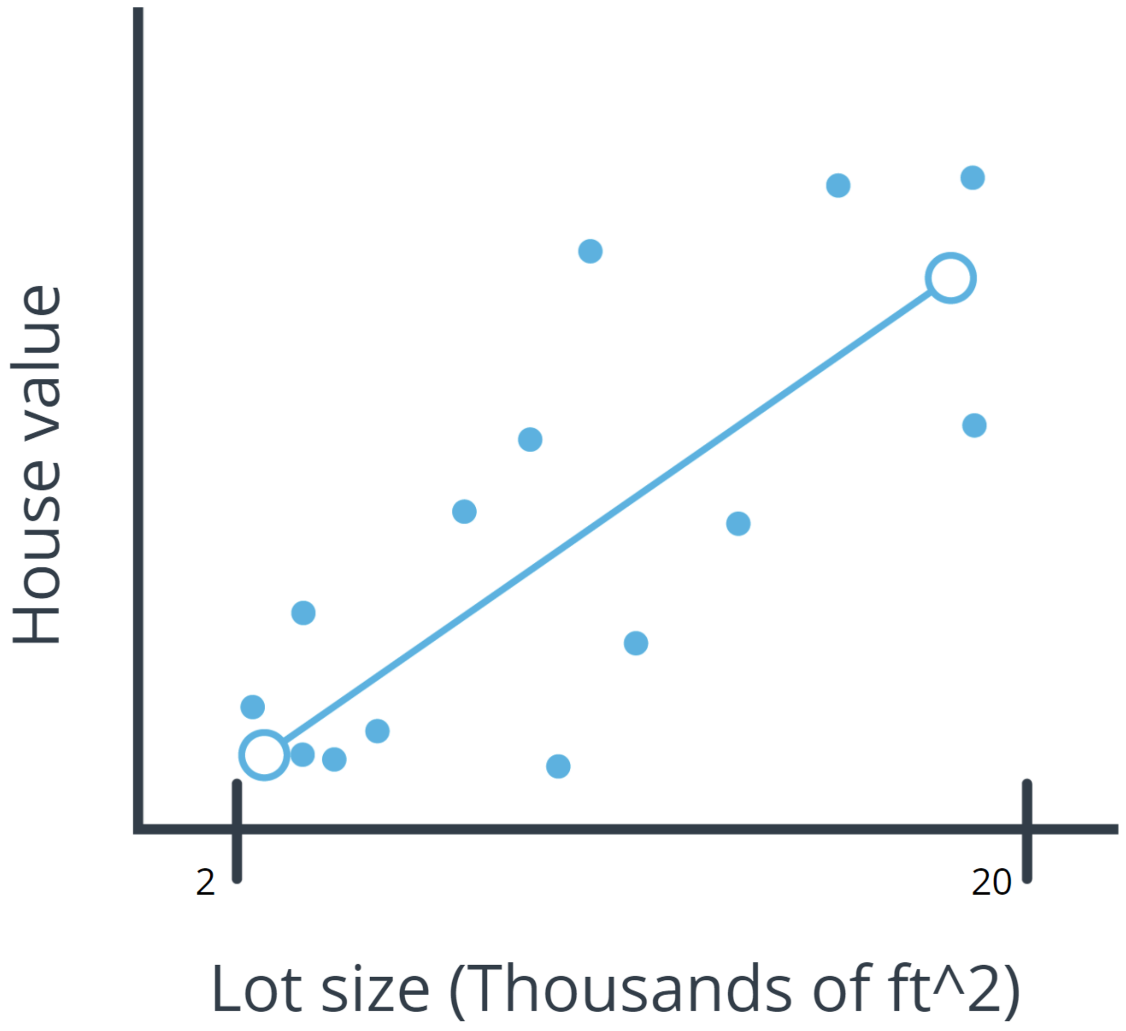
For this project, you need data about home prices, so you do the following tasks:

- Data collection: You collect numerous examples of homes sold in your neighborhood within the past year, and pay a real estate appraiser to appraise the homes whose selling price is not known.
- Data exploration: You confirm that all of your data is numerical because most machine learning models operate on sequences of numbers. If there is textual data, you need to transform it into numbers. You'll see this in the next example.
- Data cleaning: Look for things such as missing information or outliers, such as the 10-room mansion. You can use several techniques to handle outliers, but you can also just remove them from your dataset.

# of Rooms	Lot Size (ft^2)	House Value ($)
4	10,454	339,900
3	9,147	239,000
3	10,890	250,000
~~10~~	~~25,877~~	~~877,000~~

You also want to look for trends in your data, so you use **data visualization** to help you find them.

You can plot home values against each of your input variables to look for trends in your data. In the following chart, you see that when lot size increases, house value increases.



Step 3 : Model Training

Prior to actually training your model, you need to split your data. The standard practice is to put 80% of your dataset into a training dataset and 20% into a test dataset.

Linear model selection

As you see in the preceding chart, when lot size increases, home values increase too. This relationship is simple enough that a linear model can be used to represent this relationship.
A linear model across a single input variable can be represented as a line. It becomes a plane for two variables, and then a hyperplane for more than two variables. The intuition, as a line with a constant slope, doesn't change.

Using a Python library

The Python *scikit-learn* library has tools that can handle the implementation of the model training algorithm for you.

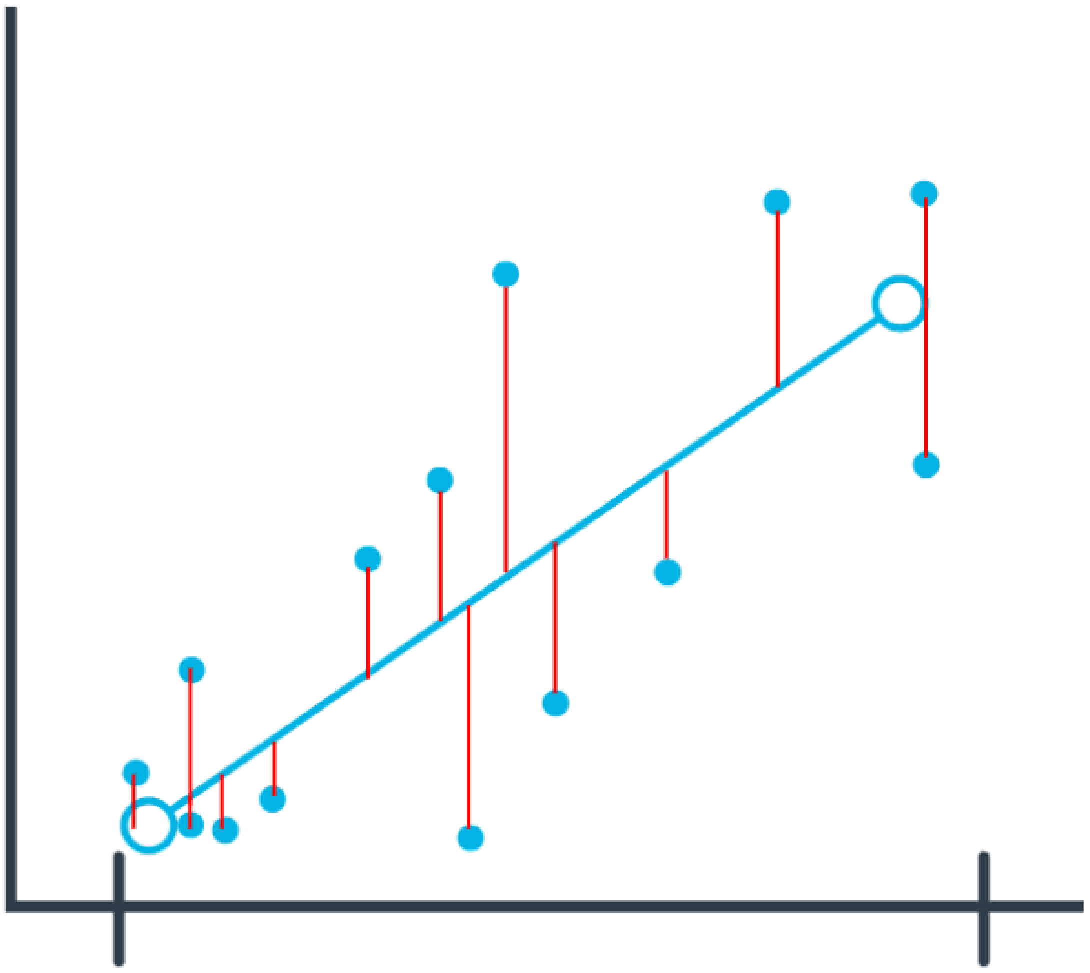
Step 4 : Model Evaluation

One of the most common evaluation metrics in a regression scenario is called root mean square or RMS. The math is beyond the scope of this lesson, but RMS can be thought of roughly as the "average error" across your test dataset, so you want this value to be low.

$$RMS = \sqrt{\frac{1}{n}\sum_i x_i^2}$$

In the following chart, you can see where the data points are in relation to the blue line. You want the data points to be as close to the "average" line as possible, which would mean less net error.

You compute the root mean square between your model's prediction for a data point in your test dataset and the true value from your data. This actual calculation is beyond the scope of this lesson, but it's good to understand the process at a high level.



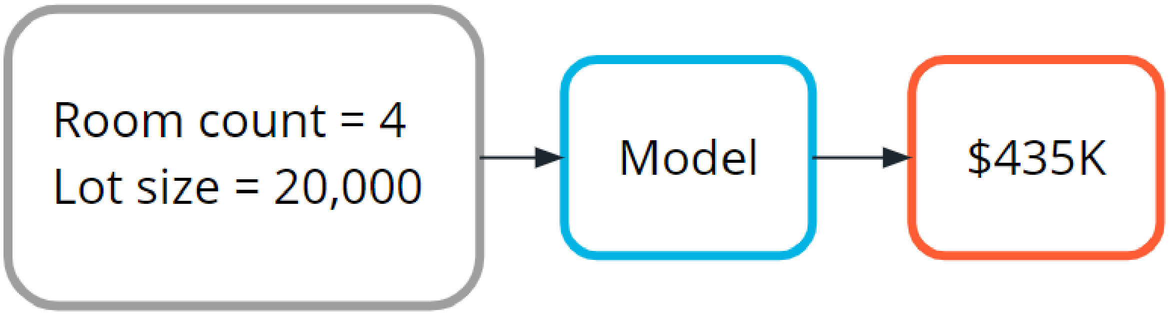
Interpreting Results

In general, as your model improves, you see a better RMS result. You may still not be confident about whether the specific value you've computed is good or bad.

Many machine learning engineers manually count how many predictions were off by a threshold (for example, $50,000 in this house pricing problem) to help determine and verify the model's accuracy.

Step 5 : Model Inference

Now you are ready to put your model into action. As you can see in the following image, this means seeing how well it predicts with new data not seen during model training.



Wrap Up

1. Solving problems using machine learning is an evolving and iterative process.
2. To Solve a problem successfully in machine learning, finding high quality data is essential.
3. To evaluate models, you often use statistical metrics. The metrics you choose are tailored to a specific use case.

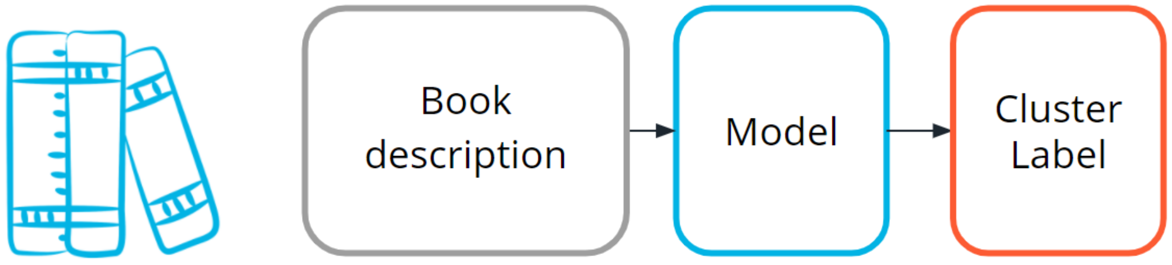
Terminology

- **Regression:** A common task in supervised machine learning used to understand the relationship between multiple variables from a dataset.
- **Continuous:** Floating-point values with an infinite range of possible values. This is 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 drawing a straight line.

Example 2: Using ML to predict Book genre (Unsupervised Learning)

Step 1: Define the problem

Is it possible to find clusters of similar books based on the presence of common words in the book descriptions?



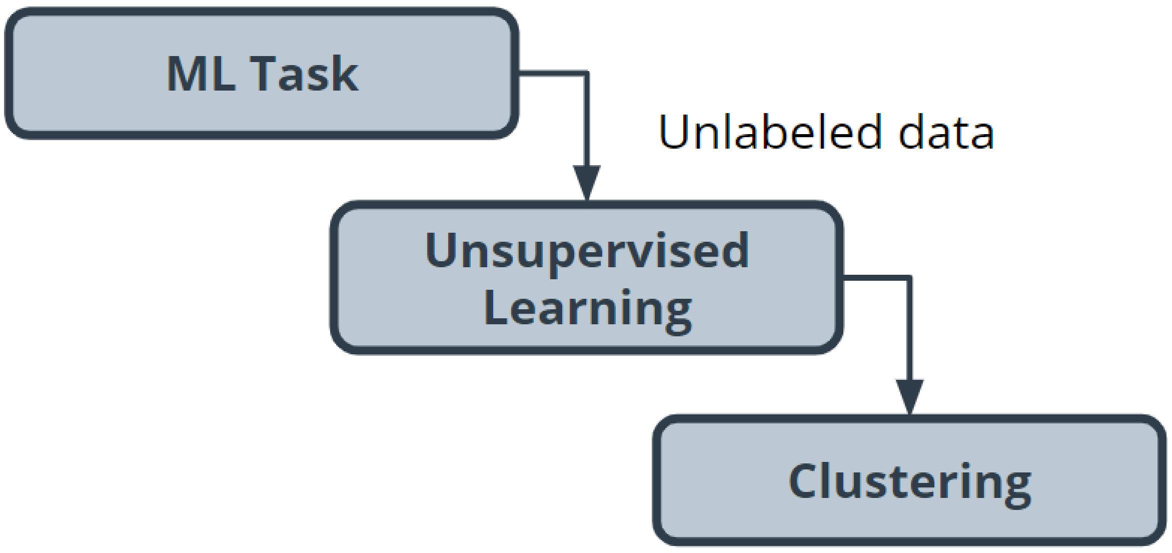
You do editorial work for a book recommendation company, and you want to write an article on the largest book trends of the year. You believe that a trend called "micro-genres" exists, and you have confidence that you can use the book description text to identify these micro-genres.

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.

Identify the machine learning task you could use

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.



Step 2: Build your dataset

To test the hypothesis, you gather book description text for 800 romance books published in the current year. You plan to use this text as your dataset.

Data exploration, cleaning, and preprocessing

In the lesson about building your dataset, you learned about how sometimes it is necessary to change the format of the data that you want to use. In this case study, we need use a process called *vectorization*. Vectorization is a process whereby words are converted into numbers.

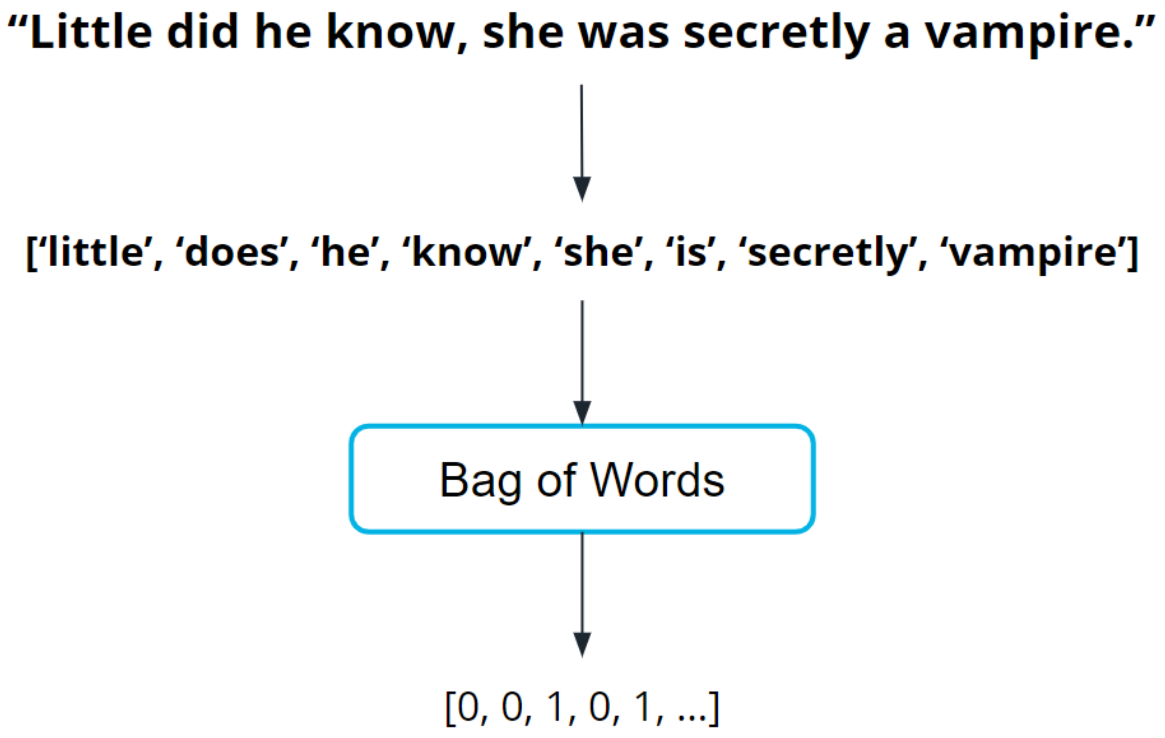
Data cleaning and exploration

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*.

Data preprocessing

Before you can train the model, you need to do a type of data preprocessing called *data vectorization*, which is used to convert text into numbers.
As shown in the following image, you transform this book description text into what is called a *bag of words representation*, so that it is understandable by machine learning models.

How the *bag of words representation* works is beyond the scope of this lesson. If you are interested in learning more, see the **What's next** section at the end of this chapter.


“Little did he know, she was secretly a vampire.”
['little', 'does', 'he', 'know', 'she', 'is', 'secretly', 'vampire']
Bag of Words
[0, 0, 1, 0, 1, ...]

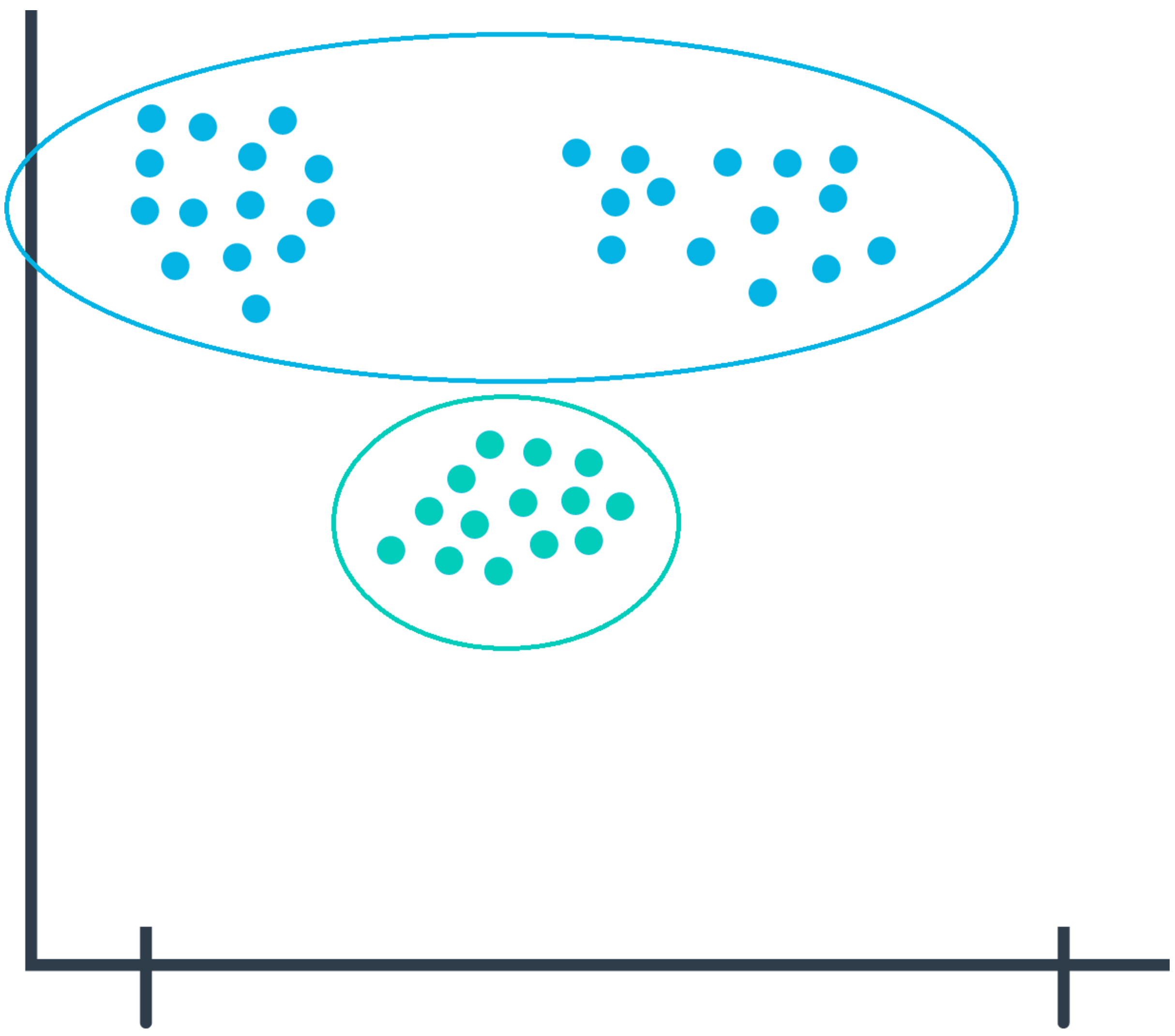
Step 3: Train the Model

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 and you don't how many micro-genres might exist. So, you train your model multiple times using different values for *k* each time.

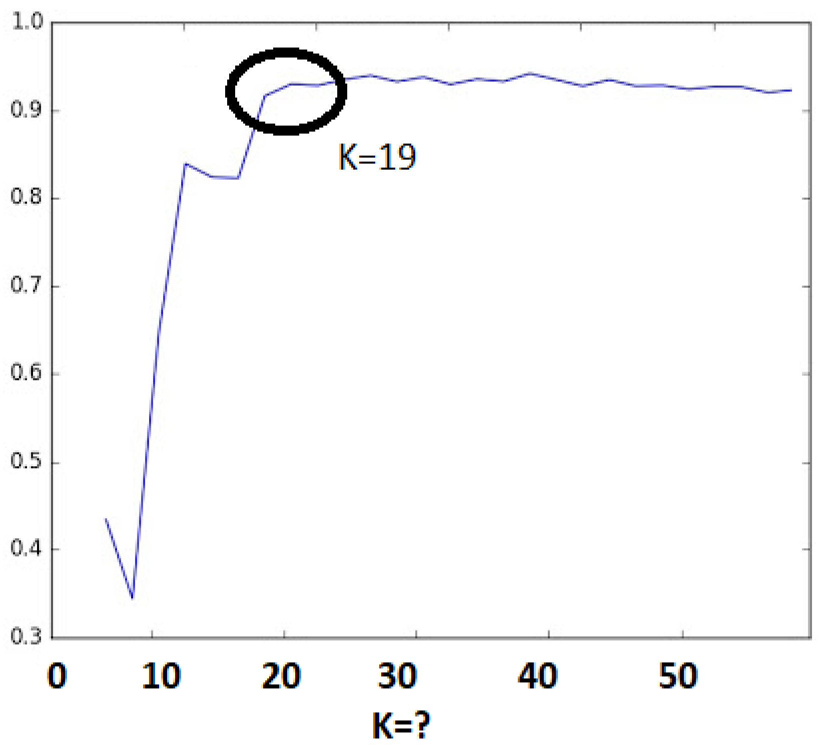
What does this even mean? In the following graphs, you can see examples of when *k=2* and when *k=3*.



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

Step 4: 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*.



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

You find one cluster that contains a large collection of books that 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 5: Model Inference

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...”

As you inspect the preceding table, you can see that most of these text snippets indicate 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 micro-genres.

Wrap Up

In this example, you saw how you can use machine learning to help find micro-genres in books by using the text found on the back of the book. Here is summary of key moments from the lesson you just finished.

1. For some applications of machine learning, you need to not only clean and preprocess the data but also convert the data into a format that is machine readable. In this example, the words were converted into numbers through a process called data vectorization.
2. Solving problems in machine learning requires iteration. In this example you saw how it was necessary to train the model multiples times for different values of k. After training your model over multiple iterations you saw how the silhouette coefficient could be use to determine the optimal value for k.
3. During model inference you continued to inspect the clusters for accuracy to ensure that your model was generative useful predictions.

Terminology

- **Bag of words**: A technique used to extract features from 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 coefficients**: 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.

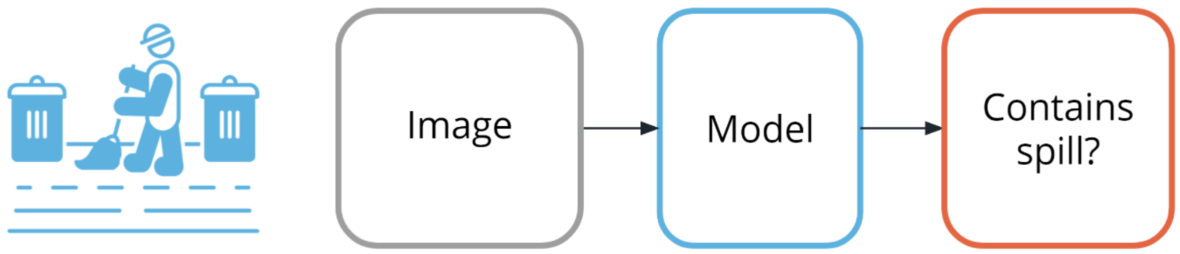
Example 3: Using ML to Detect Spills (Deep Neural Networks)

This example uses a neural network. The algorithm for how a neural network works is beyond the scope of this lesson. However, there is still value in seeing how machine learning applies, in this case.

Step 1: Defining the Problem

Imagine you run a company that offers specialized on-site janitorial services. One 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.

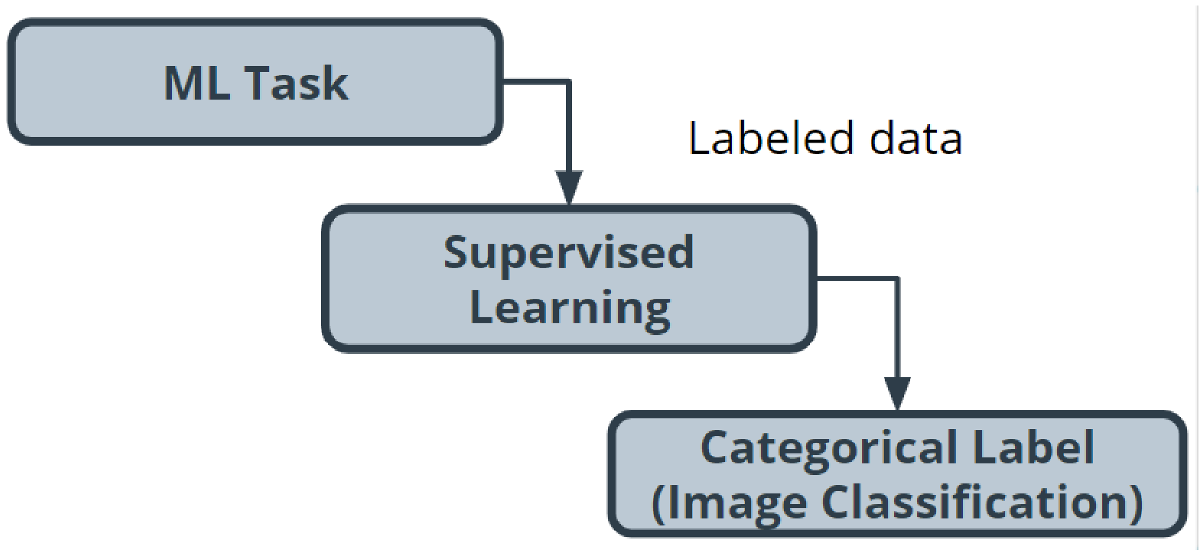
Machine learning could be a valuable tool to solve this problem.



Choosing a model

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



Step 2: Building a Dataset

Collecting

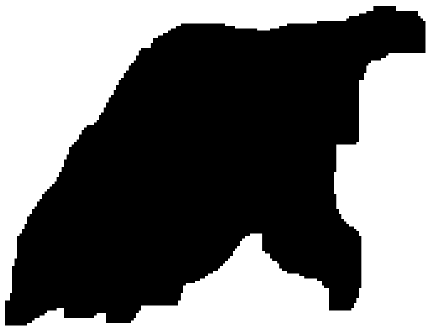
- Using historical data, as well as safely staged spills, quickly build a collection of images that contain both spills and non-spills in multiple lighting conditions and environments.

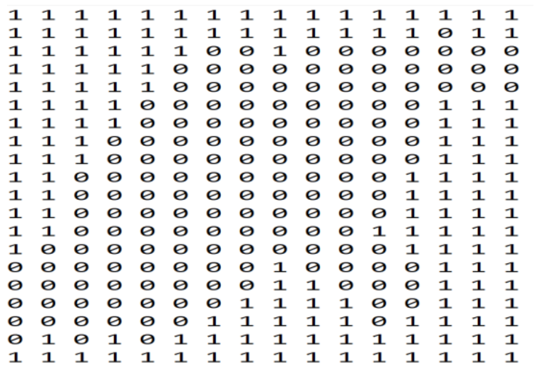
Exploring and cleaning

- Go through all of the photos to ensure that 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 that you need to iterate.

Data vectorization (converting to numbers)

- Many models require numerical data, so you must transform all of 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 immediately below can be represented in the image beneath it using 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

- Split your image data into a training dataset and a test dataset.

Step 3: 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 that is particularly good at processing images.

Step 4: Model Evaluation

As you saw in the last example, there are many different statistical metrics that 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:

- Accuracy
- Confusion matrix
- F1 score
- False positive rate
- False negative rate
- Log loss
- Negative predictive value
- Precession
- Recall
- ROC Curve
- Specificity

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

Why not? 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.

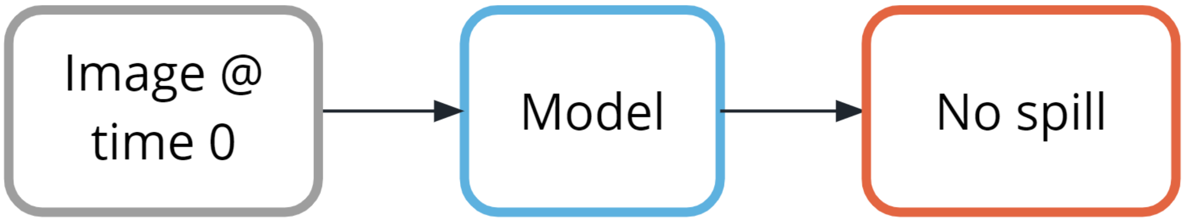
After doing some internet sleuthing, you realize this is a common problem and that *precision* and *recall* will be effective. Think of *precision* as answering the question, "Of all predictions of a spill, how many were right?" and *recall* as answering the question, "Of all actual spills, how many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

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

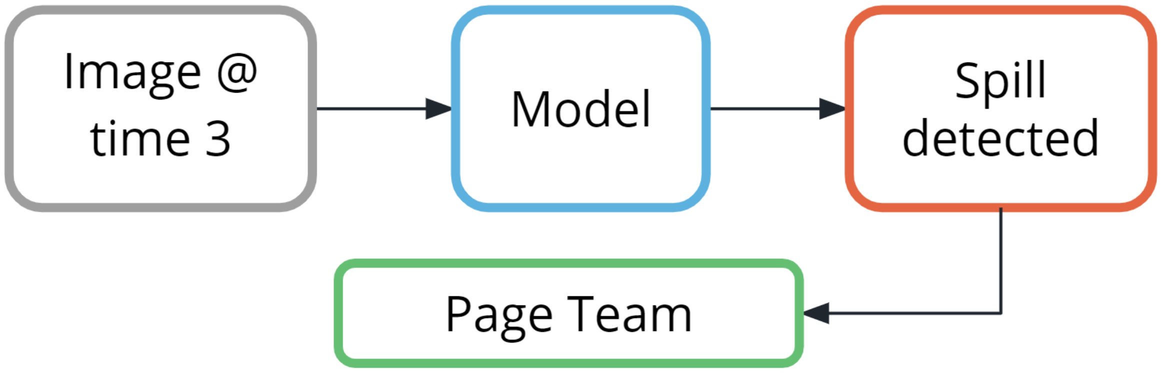
Step 5: Model Inference

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*.



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



Wrap Up

1. For some applications of machine learning, you need to use more complicated techniques to solve the problem. While modern neural networks are a powerful tool, don't forget their cost in terms of being easily explained.
2. High quality data once again was very important to the success of this application, to the point where even staging some fake data was required. Once again, the process of data vectorization was required so it was important to convert the images into numbers so that they could be used by the neural network.
3. During model inference you continued to inspect the predictions for accuracy. It is especially important in this case because you created some fake data to use when training your model.

Terminology

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**.

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