

What is machine learning?

He defined machine learning as the field of study that gives computers the ability to learn without being explicitly programmed. Samuel's claim to fame was that back in the 1950s

### **Some history of machine learning**

he wrote a checkers playing program. The amazing thing about this program was that Arthur Samuel himself wasn't a very good checkers player. What he did was he had programmed the computer to play maybe tens of thousands of games against itself. By watching what social support positions tend to lead to wins and what positions tend to lead to losses the checkers plane program learned over time what are good or bad suport positions by trying to get a good and avoid bad positions, this program learned to get better and better at playing checkers because the computer had the patience to play tens of thousands of games against itself. It was able to get so much checkers playing experience that eventually it became a better checkers player than also, Samuel himself!!!

## **Question**

---

If Arthur Samuel's checkers-playing program had been allowed to play only 10 games against itself, how would this have affected its performance compared to when it was allowed to play over 10,000 games?

Answer :

Would have made it worse

we'll dive deeper together into what are the major types of machine learning algorithms?

The two main types of machine learning are supervised learning and unsupervised learning.

(we will define it next)

supervised learning is the type of machine learning that is used most in many real-world applications and has seen the most rapid advancements and innovation.

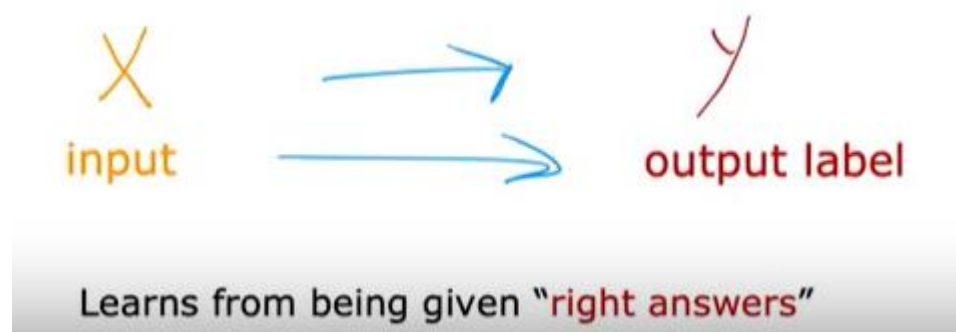
-----

## **Supervised learning:**

Machine learning is creating tremendous economic value today. I think 99 percent of the economic value created by machine learning today is through one type of machine learning, which is called supervised learning.

Supervised machine learning or more commonly, supervised learning, refers to algorithms that learn  $x$  to  $y$  or input to output mappings. The key characteristic of supervised learning is that you give your learning algorithm examples to learn from. That includes the right answers, whereby right answer, I mean, the correct label  $y$  for a given input  $x$ , and is by seeing correct pairs of input

x and desired output label y that the learning algorithm eventually learns to take just the input alone without the output label and gives a reasonably accurate prediction or guess of the output.



Let's look at some examples. If the input x is an email and the output y is this email, spam or not spam, this gives you your spam filter. Or if the input is an audio clip and the algorithm's job is output the text transcript, then this is speech recognition. Or if you want to input English and have it output to corresponding Spanish, Arabic, Hindi, Chinese, Japanese, or something else translation, then that's machine translation. Or the most lucrative form of supervised learning today is probably used in online advertising. Nearly all the large online ad platforms have a learning algorithm that inputs some information about an ad and some information about you and then tries to figure out if you will click on that ad or not. Because by showing you ads they're just slightly more likely to click on, for these large online ad platforms, every click is revenue, this actually drives a lot of revenue for these companies.

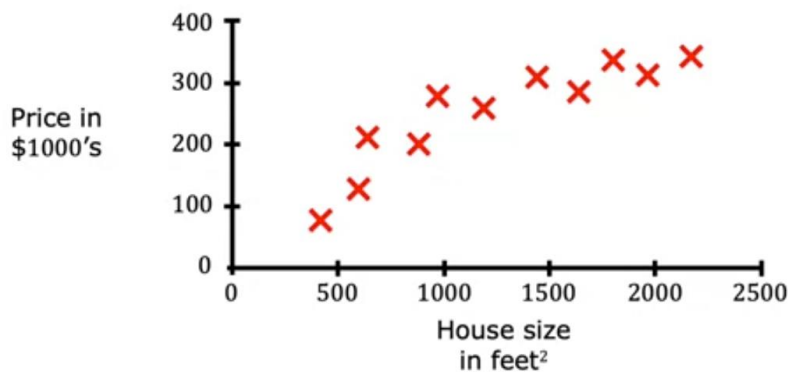
Input (x)	Output (y)	Application
email	spam? (0/1)	spam filtering
audio	text transcripts	speech recognition
English	Spanish	machine translation
ad, user info	click? (0/1)	online advertising

In all of these applications, you will first train your model with examples of inputs x and the right answers, that is the labels y. After the model has learned from these input, output, or x and y pairs, **they can then take a brand new input x**, something it has never seen before, and try to produce the appropriate corresponding output y

**Let's dive more deeply into one specific example.**

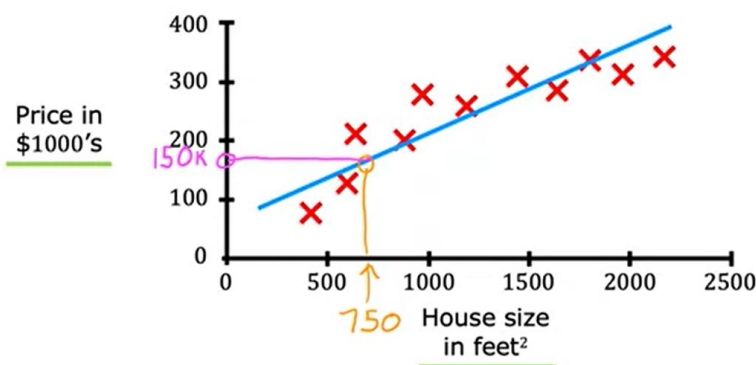
Say you want to predict housing prices based on the size of the house. You've collected some data and say you plot the data and it looks like this.

## Regression: Housing price prediction



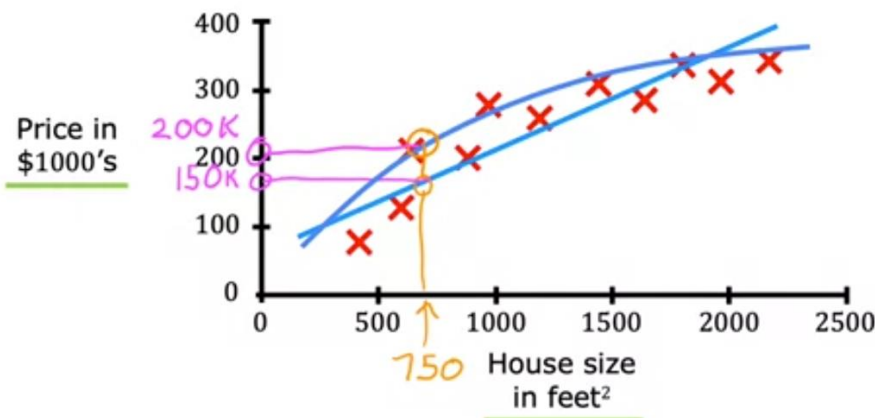
Here on the horizontal axis is the size of the house in square feet. Here on the vertical axis is the price of the house in, say, thousands of dollars. With this data, let's say a friend wants to know what's the price for their 750 square foot house. How can the learning algorithm help you? One thing a learning algorithm might be able to do is say, for the straight line to the data and reading off the straight line, it looks like your friend's house could be sold for maybe about, I don't know, \$150,000

## Regression: Housing price prediction



There are others that could work better for this application. For example, instead of fitting a straight line, you might decide that it's better to fit a curve, a function that's slightly more complicated or more complex than a straight line. If you do that and make a prediction here, then it looks like, well, your friend's house could be sold for closer to \$200,000.

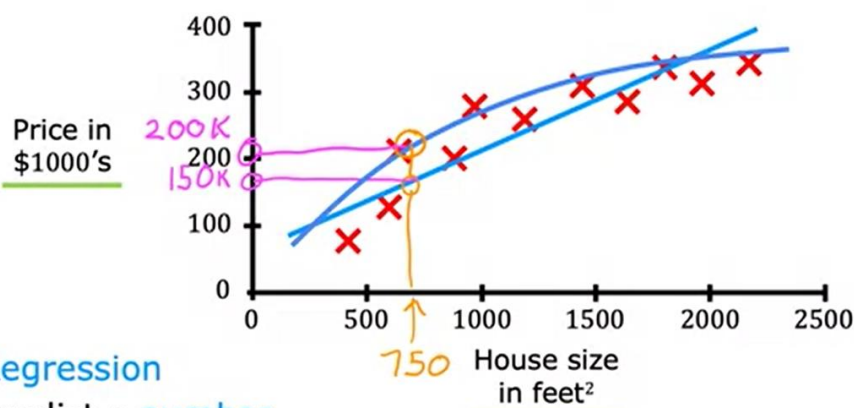
## Regression: Housing price prediction



Now, it doesn't seem appropriate to pick the one that gives your friend the best price, but one thing you see is how to get an algorithm to systematically choose the most appropriate line or curve or other thing to fit to this data. What you've seen in this slide is an example of supervised learning. Because we gave the algorithm a dataset in which the so-called right answer, that is the label or the correct price  $y$  is given for every house on the plot. The task of the learning algorithm is to produce more of these right answers, specifically predicting what is the likely price for other houses like your friend's house. That's why this is supervised learning. To define a little bit more terminology, this housing price prediction is the particular type of supervised learning called **regression**.

By regression, I mean we're trying to predict a number from infinitely many possible numbers such as the house prices in our example, which could be 150,000 or 70,000 or 183,000 or any other number in between. That's supervised learning, learning input, output, or  $x$  to  $y$  mappings. You saw in this video an example of regression where the task is to predict number.

## Regression: Housing price prediction



Regression

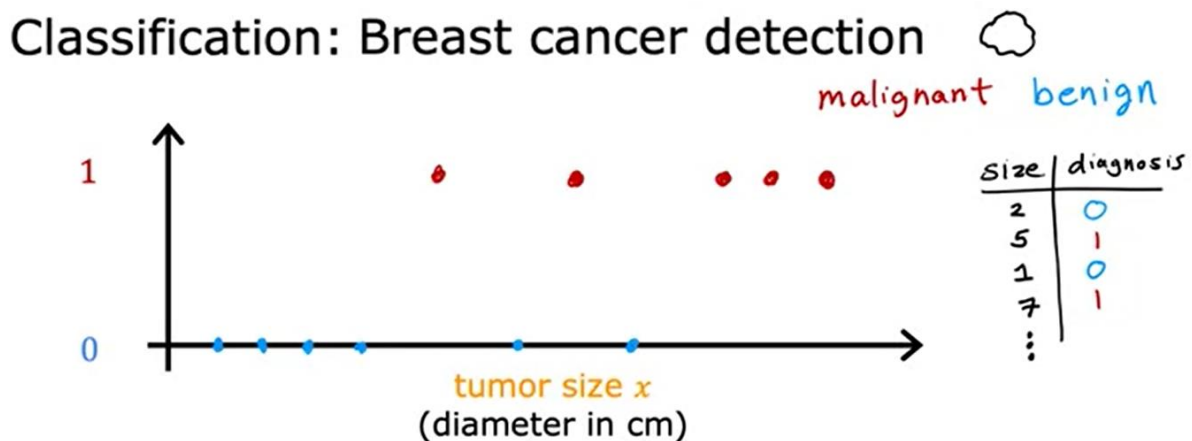
Predict a **number**

**infinitely** many possible outputs

But there's also a second major type of supervised learning problem called **classification**.

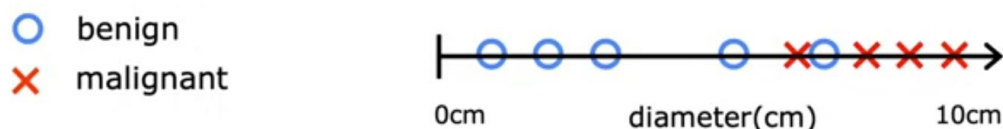
Let's take a look at what this means. Take breast cancer detection as an example of a classification problem. Say you're building a machine learning system so that doctors can have a diagnostic tool to detect breast cancer. This is important because early detection could potentially save a patient's life. Using a patient's medical records your machine learning system tries to figure out if a tumor that is a lump is malignant meaning cancerous or dangerous. Or if that tumor, that lump is benign, meaning that it's just a lump that isn't cancerous and isn't that dangerous.

So maybe your dataset has tumors of various sizes. And these tumors are labeled as either benign, which I will designate in this example with a 0 or malignant, which will designate in this example with a 1. You can then plot your data on a graph like this where the horizontal axis represents the size of the tumor and the vertical axis takes on only two values 0 or 1 depending on whether the tumor is benign, 0 or malignant 1. One reason that this is different from regression is that we're trying to predict only a small number of possible outputs or categories. In this case two possible outputs 0 or 1, benign or malignant. This is different from regression which tries to predict any number, all of the infinitely many number of possible numbers. And so the fact that there are only two possible outputs is what makes this classification. Because there are only two possible outputs or two possible categories in this example, you can also plot this data set on a line like this.



Because there are only two possible outputs or two possible categories in this example, you can also plot this data set on a line like this. Right now, I'm going to use two different symbols to denote the category using a circle and O to denote the benign examples and a cross to denote the malignant examples. And if new patients walks in for a diagnosis and they have a lump that is this size, then the question is, will your system classify this tumor as benign or malignant?

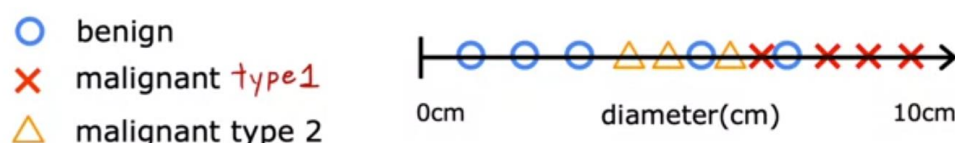
### Classification: Breast cancer detection



It turns out that in classification problems you can also have more than two possible output categories. Maybe you're learning algorithm can output multiple types of cancer diagnosis if it turns out to be malignant. So let's call two different types of cancer type 1 and type 2. In this case the average would have three possible output categories it could predict. And by the way in

classification, the terms output classes and output categories are often used interchangeably. So what I say class or category when referring to the output, it means the same thing.

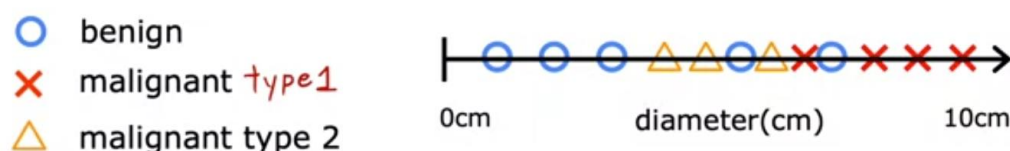
## Classification: Breast cancer detection



class category  
 Classification  
 predict categories

So to summarize classification algorithms predict categories. Categories don't have to be numbers. It could be non numeric for example, it can predict whether a picture is that of a cat or a dog. And it can predict if a tumor is benign or malignant. Categories can also be numbers like 0, 1 or 0, 1, 2. But what makes classification different from regression when you're interpreting the numbers is that classification predicts a small finite limited set of possible output categories such as 0, 1 and 2 but not all possible numbers in between like 0.5 or 1.7.

## Classification: Breast cancer detection



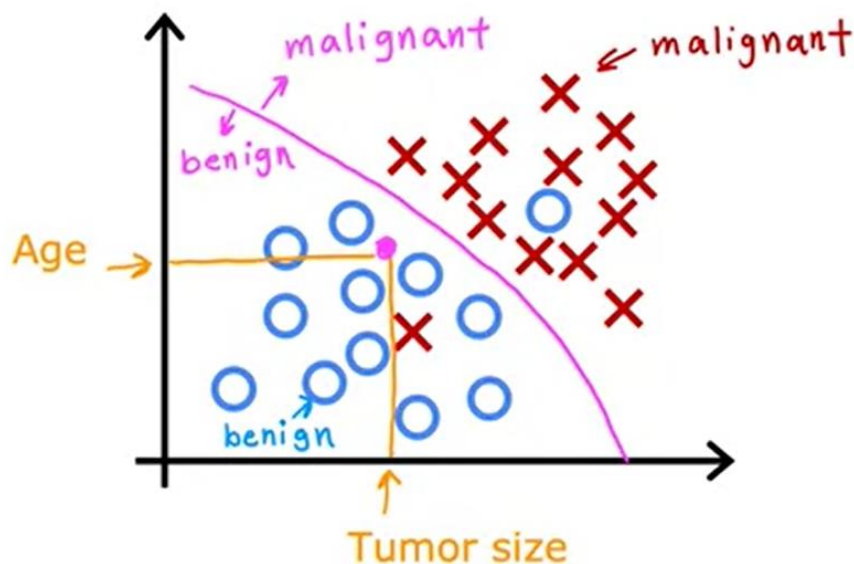
class category  
 Classification  
 predict categories    cat dog    benign malignant    0, 1, 2  
 small number of possible outputs

In the example of supervised learning that we've been looking at, we had only one input value the size of the tumor. But you can also use more than one input value to predict an output. Here's an example, instead of just knowing the tumor size, say you also have each patient's age in years. Your new data set now has two inputs, age and tumor size. What in this new dataset we're going to use circles to show patients whose tumors are benign and crosses to show the patients with a tumor that was malignant. So when a new patient comes in, the doctor can measure the patient's tumor size and also record the patient's age. And so given this, how can we predict if this patient's tumor is benign or malignant? Well, given the day said like this, what the learning algorithm might do is find some boundary that separates out the malignant tumors from the benign ones. So the learning algorithm has to decide how to fit a boundary line through



this data. The boundary line found by the learning algorithm would help the doctor with the diagnosis. In this case the tumor is more likely to be benign

## Two or more inputs



In other machine learning problems often many more input values are required.

The doctors in breast cancer detection use many additional inputs, like the thickness of the tumor clump, uniformity of the cell size, uniformity of the cell shape and so on.

So to recap supervised learning maps input  $x$  to output  $y$ , where the learning algorithm learns from the quote right answers. The two major types of supervised learning are regression and classification. In a regression application like predicting prices of houses, the learning algorithm has to predict numbers from infinitely many possible output numbers. Whereas in classification the learning algorithm has to make a prediction of a category, all of a small set of possible outputs.

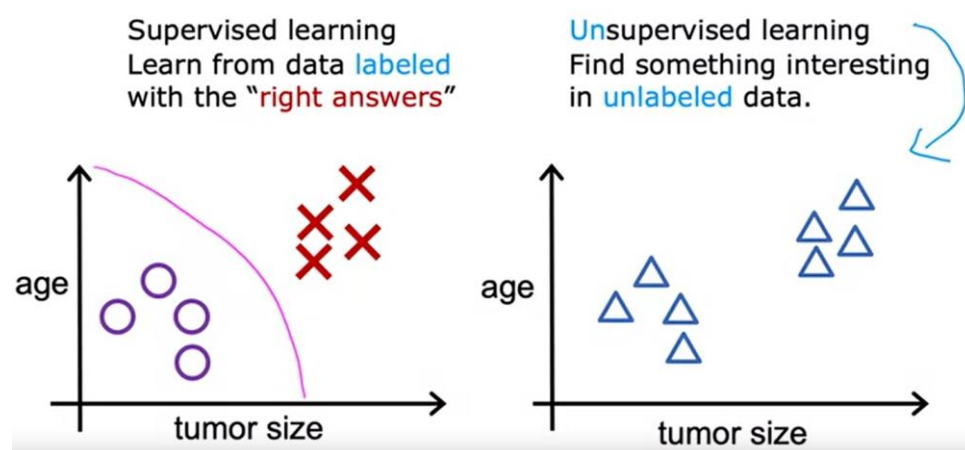
---

## unsupervised learning:

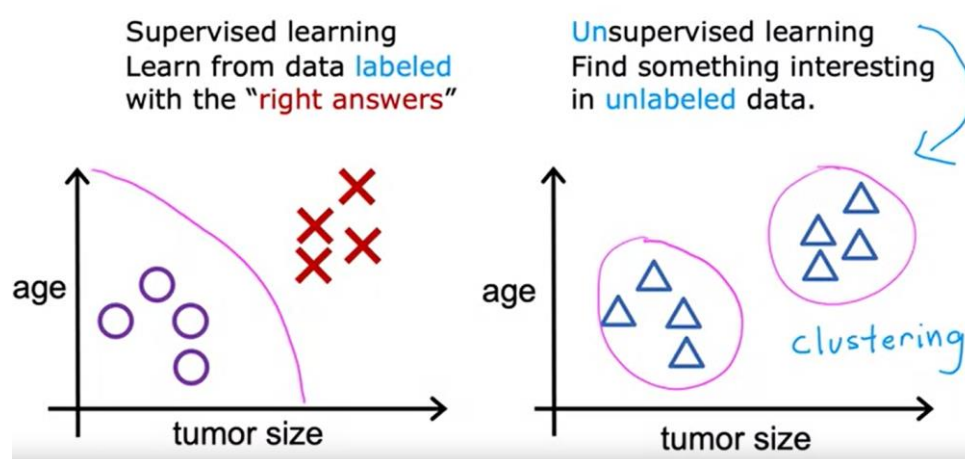
After supervised learning, the most widely used form of machine learning is unsupervised learning. Let's take a look at what that means.

When we're looking at supervised learning in the last video recalled, it looks something like this in the case of a classification problem. Each example, was associated with an output label  $y$  such as benign or malignant, designated by the poles and crosses in unsupervised learning. Were given data that isn't associated with any output labels  $y$ , say you're given data on patients and their tumor size and the patient's age. But not whether the tumor was benign or malignant, so the dataset looks like this on the right. We're not asked to diagnose whether the tumor is benign or malignant, because we're not given any labels. Why in the dataset, instead, our job is to find

some structure or some pattern or just find something interesting in the data. This is unsupervised learning, we call it unsupervised because we're not trying to supervise the algorithm. To give some quote right answer for every input, instead, we asked the our room to figure out all by yourself what's interesting.



An unsupervised learning algorithm, might decide that the data can be assigned to two different groups or two different clusters. And so it might decide, that there's one cluster what group , and there's another cluster or group (in the image). This is a particular type of unsupervised learning, called a **clustering algorithm**. Because it places the unlabeled data, into different clusters and this turns out to be used in many applications

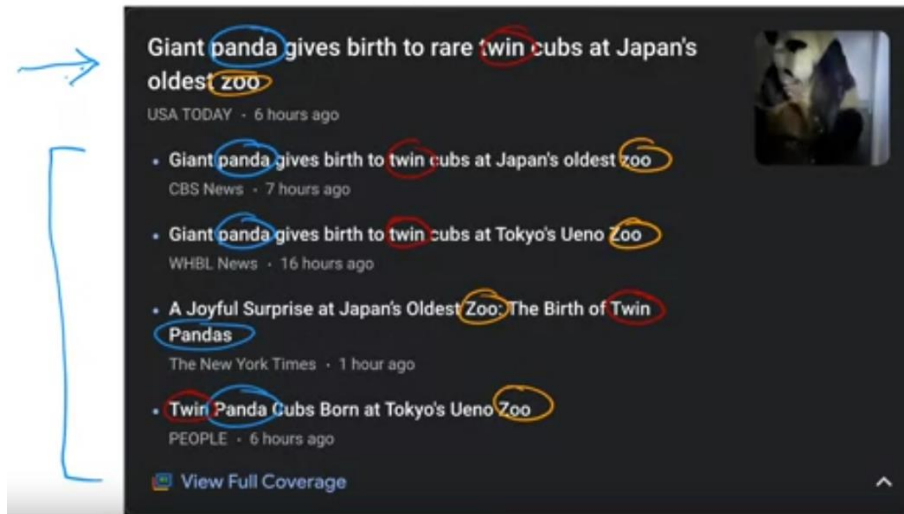


For example, here is a sample from Google News, where the headline of the top article, is giant panda gives birth to rear twin cubs at Japan's oldest zoo.

you might notice that below this are other related articles. Maybe from the headlines alone, you can start to guess what clustering might be doing. Notice that the word panda appears in a lot of spaces and in the image notice that the word twin also appears in all five articles. And the word Zoo also appears in all of these articles, so the clustering algorithm is finding articles. All of all the hundreds of thousands of news articles on the internet that day, finding the articles that mention similar words and grouping them into clusters.



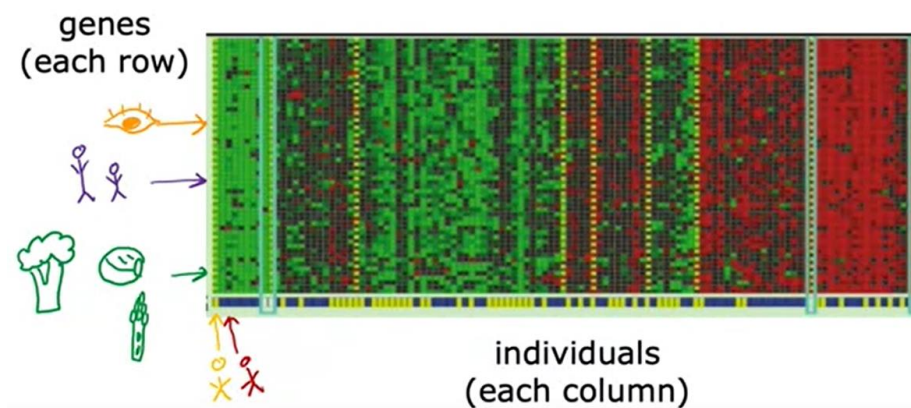
## Clustering: Google news



All of all the hundreds of thousands of news articles on the internet that day, finding the articles that mention similar words and grouping them into clusters. Now, what's cool is that this clustering algorithm figures out on his own which words suggest, that certain articles are in the same group. What I mean is there isn't an employee at google news who's telling the algorithm to find articles that the word panda. And twins and zoo to put them into the same cluster, the news topics change every day.

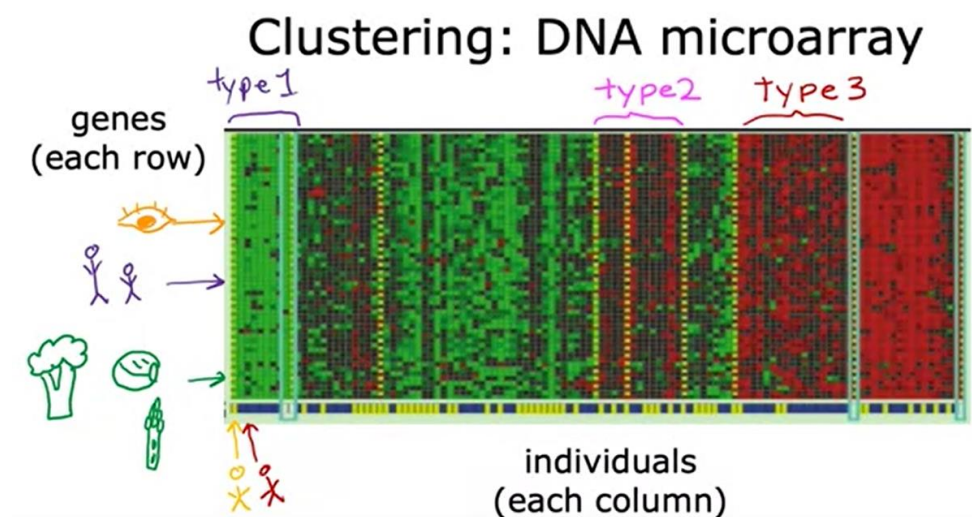
Let's look at the second example of unsupervised learning applied to clustering genetic or DNA data. This image shows a picture of DNA micro array data, these look like tiny grids of a spreadsheet. And each tiny column represents the genetic or DNA activity of one person, So for example, this entire Column here is from one person's DNA. And this other column is of another person, each row represents a particular gene. So just as an example, perhaps this role here might represent a gene that affects eye color, or this role here is a gene that affects how tall someone is. Researchers have even found a genetic link to whether someone dislikes certain vegetables, such as broccoli, or brussels sprouts, or asparagus.

## Clustering: DNA microarray



So next time someone asks you why didn't you finish your salad, you can tell them, maybe it's genetic for DNA micro race. The idea is to measure how much certain genes, are expressed for each individual person.

The idea is to measure how much certain genes, are expressed for each individual person. So these colors red, green, gray, and so on, show the degree to which different individuals do, or do not have a specific gene active. And what you can do is then run a clustering algorithm to group individuals into different categories. Or different types of people like maybe these individuals that group together, and let's just call this type one. And these people are grouped into type two, and these people are groups as type three. This is unsupervised learning, because we're not telling the algorithm in advance, that there is a type one person with certain characteristics. Or a type two person with certain characteristics, instead what we're saying is here's a bunch of data. I don't know what the different types of people are but can you automatically find structure into data. And automatically figure out whether the major types of individuals, since we're not giving the algorithm the right answer for the examples in advance.



Let's give a slightly more **formal definition** of unsupervised learning and take a quick look at some other types of unsupervised learning other than clustering. Whereas in supervised learning, the data comes with both inputs  $x$  and input labels  $y$ , in unsupervised learning, the data comes only with inputs  $x$  but not output labels  $y$ , and the algorithm has to find some structure or some pattern or something interesting in the data. We're seeing just one example of unsupervised learning called a clustering algorithm, which groups similar data points together. In this specialization, you'll learn about clustering as well as two other types of unsupervised learning. One is called anomaly detection, which is used to detect unusual events. This turns out to be really important for fraud detection in the financial system, where unusual events, unusual transactions could be signs of fraud and for many other applications. You also learn about dimensionality reduction. This lets you take a big data-set and almost magically compress it to a much smaller data-set while losing as little information as possible.

# Unsupervised learning

Data only comes with inputs  $x$ , but not output labels  $y$ .  
Algorithm has to find **structure** in the data.

## Clustering

Group similar data points together.

## Dimensionality reduction

Compress data using fewer numbers.

## Anomaly detection

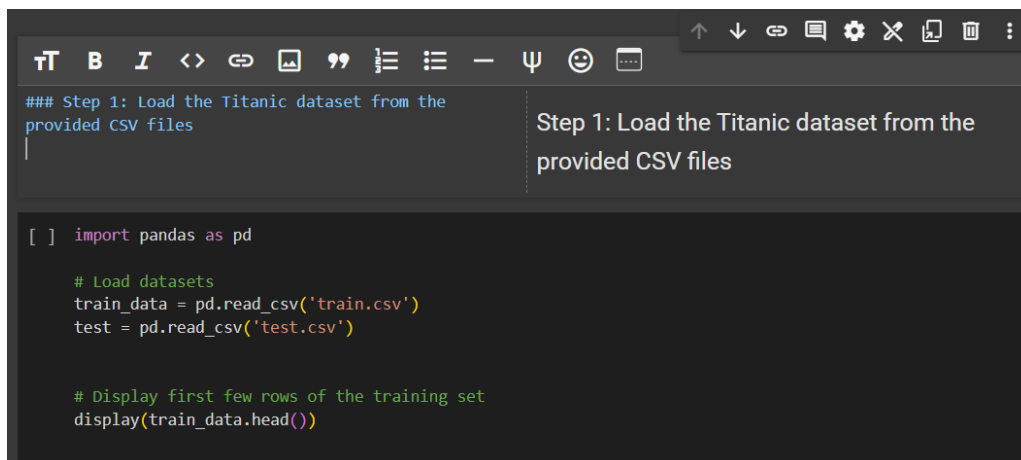
Find unusual data points.

---

## Jupyter Notebook

The most widely used tool by machine learning and data science practitioners today is the Jupyter Notebook.

This is the default environments that a lot of us use to code up and experiment and try things out.



```
### Step 1: Load the Titanic dataset from the
provided CSV files

[ ] import pandas as pd

# Load datasets
train_data = pd.read_csv('train.csv')
test = pd.read_csv('test.csv')

# Display first few rows of the training set
display(train_data.head())
```

You might notice that there are two types of these blocks, also called cells in the notebook and there are two types of cells. One is what's called a Markdown cell, which means a bunch of text. Here you can actually edit the text if you don't like the text that we wrote, but this is text that describes the code. Then there's a second type of block or cell which looks like this, which has a code cell.

If you want more informations how Jupyter Notebook work check

[https://github.com/Hisoka742/spbu\\_python\\_course](https://github.com/Hisoka742/spbu_python_course)

---

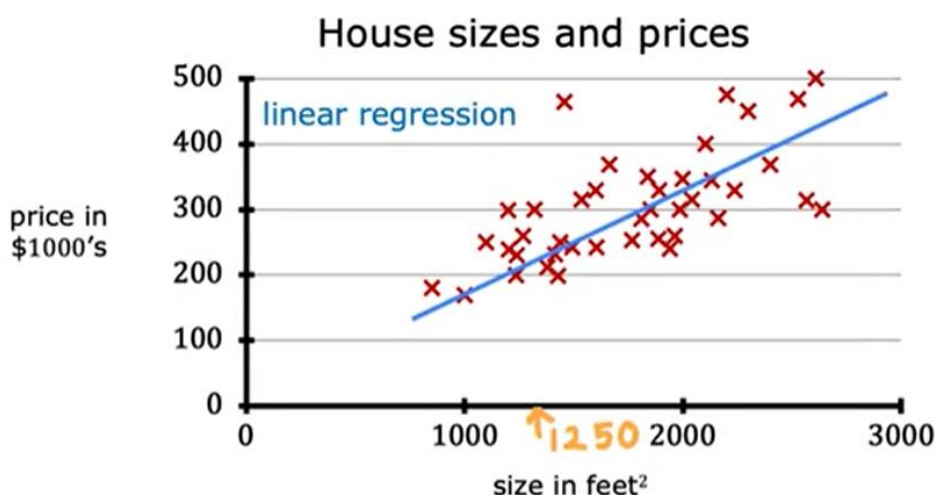
## Linear Regression

That just means fitting a straight line to your data. It's probably the most widely used learning algorithm in the world today. As you get familiar with linear regression, many of the concepts you see here will also apply to other machine learning models that you'll see later in this specialization.

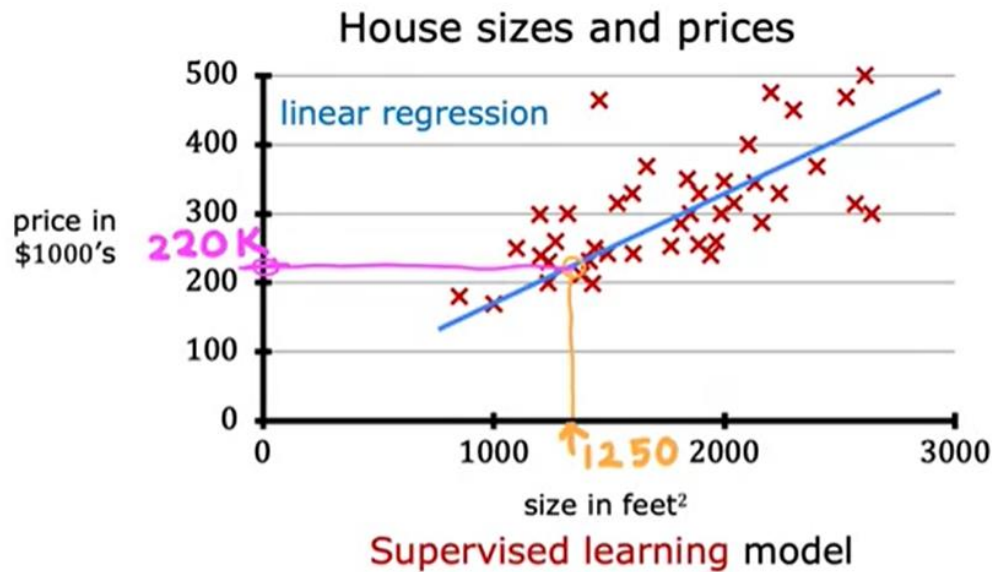
Let's start with a problem that you can address using linear regression.

Say you want to predict the price of a house based on the size of the house. We're going to use a dataset on house sizes and prices from Portland, a city in the United States. Here we have a graph where the horizontal axis is the size of the house in square feet, and the vertical axis is the price of a house in thousands of dollars. Let's go ahead and plot the data points for various houses in the dataset. Here each data point, each of these little crosses is a house with the size and the price that it most recently was sold for. Now, let's say you're a real estate agent in Portland and you're helping a client to sell her house. She is asking you, how much do you think I can get for this house?

This dataset might help you estimate the price she could get for it. You start by measuring the size of the house, and it turns out that the house is 1250 square feet. How much do you think this house could sell for? One thing you could do this, you can build a linear regression model from this dataset. Your model will fit a straight line to the data, which might look like this.

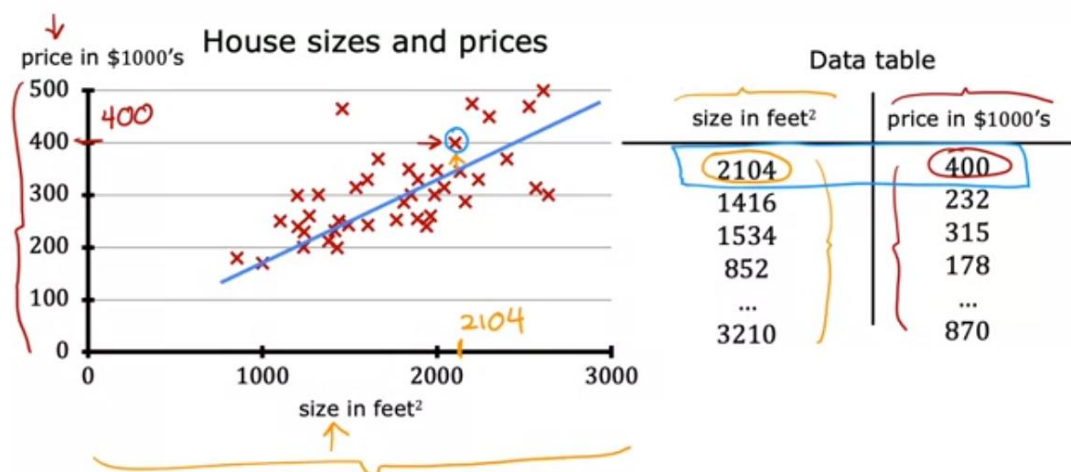


Based on this straight line fit to the data, you can see that the house is 1250 square feet, it will intersect the best fit line, and if you trace that to the vertical axis on the left, you can see the price is maybe around here, say about \$220,000.



In addition to visualizing this data as a plot here on the left (in the next image), there's one other way of looking at the data that would be useful, and that's a data table here on the right. The data comprises a set of inputs. This would be the size of the house, which is this column here. It also has outputs. You're trying to predict the price, which is this column here.

Notice that the horizontal and vertical axes correspond to these two columns, the size and the price. If you have, say, 47 rows in this data table, then there are 47 of these little crosses on the plot of the left, each cross corresponding to one row of the table. For example, the first row of the table is a house with size, 2,104 square feet, so that's around here, and this house is sold for \$400,000 which is around here. This first row of the table is plotted as this data point (in the image ).



Now, let's look at some notation for describing the data.

This is notation that you find useful throughout your journey in machine learning. As you increasingly get familiar with machine learning terminology, this would be terminology they can use to talk about machine learning concepts with others as well since a lot of this is quite standard across AI, you'll be seeing this notation multiple times in this specialization, so it's okay if you don't remember everything for assign through, it will naturally become more familiar overtime.

The dataset that you just saw and that is used to train the model is called a training set. Note that your client's house is not in this dataset because it's not yet sold, so no one knows what the price is.

To predict the price of your client's house, you first train your model to learn from the training set and that model can then predict your client's houses price. In Machine Learning, the standard notation to denote the input here is lowercase  $x$ , and we call this the input variable, is also called a feature or an input feature.

## Terminology

Training  
set:

Data used to train the model

size in feet <sup>2</sup>	price in \$1000's
2104	400
1416	232
1534	315
852	178
...	...
3210	870

For example, for the first house in your training set,  $x$  is the size of the house, so  $x$  equals 2,104. The standard notation to denote the output variable which you're trying to predict, which is also sometimes called the target variable, is lowercase  $y$ . Here,  $y$  is the price of the house, and for the first training example, this is equal to 400, so  $y$  equals 400. The dataset has one row for each house and in this training set, there are 47 rows with each row representing a different training example. We're going to use lowercase  $m$  to refer it to the total number of training examples, and so here  $m$  is equal to 47.



## Terminology

Training set:  $x$  Data used to train the model  $y$

	$x$ size in feet <sup>2</sup>	$y$ price in \$1000's
(1)	2104	400
(2)	1416	232
(3)	1534	315
(4)	852	178
...	...	...
(47)	3210	870

$m = 47$

$x = 2104$   $y = 400$

Notation:

$x$  = "input" variable  
feature

$y$  = "output" variable  
"target" variable

$m$  = number of training examples

To indicate the single training example, we're going to use the notation parentheses  $x, y$ . For the first training example,  $(x, y)$ , this pair of numbers is (2104, 400). Now we have a lot of different training examples. We have 47 of them in fact. To refer to a specific training example, this will correspond to a specific row in this table on the left, I'm going to use the notation  $x$  superscript in parenthesis,  $i$ ,  $y$  superscript in parentheses  $i$ . The superscript tells us that this is the  $i$ th training example, such as the first, second, or third up to the 47th training example.

## Terminology

Training set:  $x$  Data used to train the model  $y$

	$x$ size in feet <sup>2</sup>	$y$ price in \$1000's
(1)	2104	400
(2)	1416	232
(3)	1534	315
(4)	852	178
...	...	...
(47)	3210	870

$m = 47$

$x = 2104$   $y = 400$

$(x, y) = (2104, 400)$

Notation:

$x$  = "input" variable  
feature

$y$  = "output" variable  
"target" variable

$m$  = number of training examples

$(x, y)$  = single training example

$(x^{(i)}, y^{(i)})$   
 $(x^{(i)}, y^{(i)})$  =  $i$ th training example  
 (1st, 2nd, 3rd ...)

This is not  $x$  to the power 2. It just refers to the second training example. This  $i$ , is just an index into the training set and refers to row  $i$  in the table.

## linear regression with one variable

Recall that a **training set** in supervised learning includes both the input features, such as the size of the house and also the output targets, such as the price of the house.

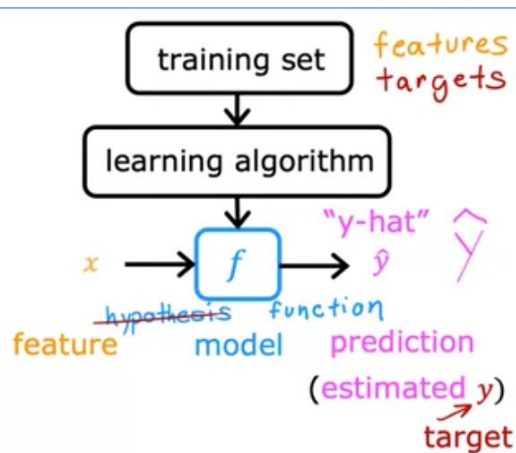
The output targets are the right answers to the model we'll learn from.

To train the model, you feed the training set, both the input features and the output targets to your learning algorithm. Then your supervised learning algorithm will produce some function.

We'll write this function as lowercase  $f$ , where  $f$  stands for function. Historically, this function used to be called a hypothesis, but I'm just going to call it a function  $f$  in this class.

The job with  $f$  is to take a new input  $x$  and output an estimate or a prediction, which I'm going to call **y-hat**, and it's written like the variable  $y$  with this little hat symbol on top.

In machine learning, the convention is that  $\hat{y}$  is the estimate or **the prediction** for  $y$ . The function  $f$  is called the model.  $x$  is called the input or the input feature, and the output of the model is the prediction,  $\hat{y}$ . The model's prediction is the estimated value of  $y$ . When the symbol is just the letter  $y$ , then that refers to the target, which is the actual true value in the training set.



In contrast,  $\hat{y}$  is an estimate. It may or may not be the actual true value. Well, if you're helping your client to sell the house, well, the true price of the house is unknown until they sell it. Your model  $f$ , given the size, outputs the price which is the estimator, that is the prediction of what the true price will be.

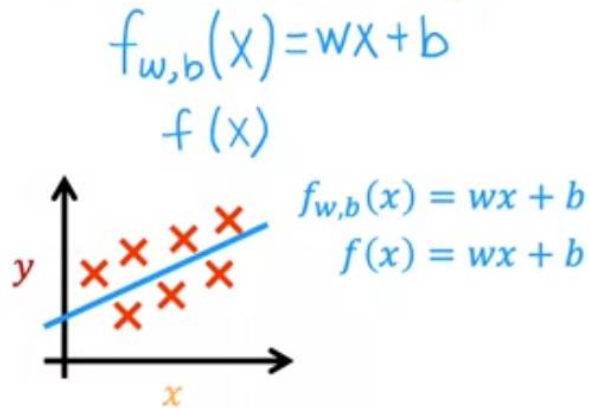
Now, when we design a learning algorithm, a key question is, **how are we going to represent the function  $f$ ? Or in other words, what is the math formula we're going to use to compute  $f$ ?**

For now, let's stick with  $f$  being a straight line. Your function can be written as  $f_w, b$  of  $x$  equals, I'm going to use  $w$  times  $x$  plus  $b$ . I'll define  $w$  and  $b$  soon. But for now, just know that  $w$  and  $b$  are numbers, and the values chosen for  $w$  and  $b$  will determine the prediction  $\hat{y}$  based on the input feature  $x$ . This  $f_w, b$  of  $x$  means  $f$  is a function that takes  $x$  as input, and depending on the values of  $w$  and  $b$ ,  $f$  will output some value of a prediction  $\hat{y}$ . As an alternative to writing this,  $f_w, b$  of  $x$ , I'll sometimes just write  $f$  of  $x$  without explicitly including  $w$  and  $b$  into subscript. Is just a simpler notation that means exactly the same thing as  $f_w, b$  of  $x$ .

Let's plot the training set on the graph where the input feature  $x$  is on the horizontal axis and the output target  $y$  is on the vertical axis. Remember, the algorithm learns from this data and generates the best-fit line like maybe this one here. This straight line is the linear function  $f_w, b$  of  $x$  equals  $w$  times  $x$  plus  $b$ . Or more simply, we can drop  $w$  and  $b$  and just write  $f$  of  $x$  equals  $w$  times  $x$  plus  $b$ .

plus  $b$ . Here's what this function is doing, it's making predictions for the value of  $y$  using a streamline function of  $x$ .

How to represent  $f$ ?



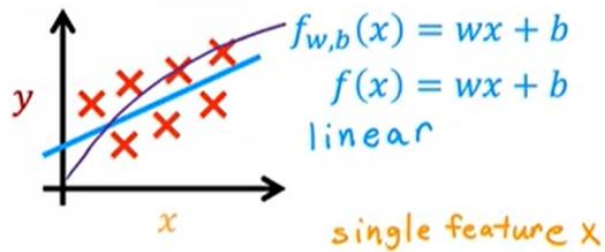
You may ask, why are we choosing a linear function, where linear function is just a fancy term for a straight line instead of some non-linear function like a curve or a parabola? Well, sometimes you want to fit more complex non-linear functions as well, like a curve like this. But since this linear function is relatively simple and easy to work with, let's use a line as a foundation that will eventually help you to get to more complex models that are non-linear. This particular model has a name, it's called linear regression. More specifically, this is **linear regression with one variable**, where the phrase one variable means that there's a single input variable or feature  $x$ , namely the size of the house.

Another name for a linear model with one input variable is **univariate linear regression**, where uni means one in Latin, and where variate means variable. Univariate is just a fancy way of saying one variable.

How to represent  $f$ ?

$$f_{w,b}(x) = wx + b$$

$$f(x)$$



Linear regression with **one** variable.

**size**

Univariate linear regression.