Applied Data Science

Module 7 - Introduction to Machine Learning

Presented By: Istvan Lengyel

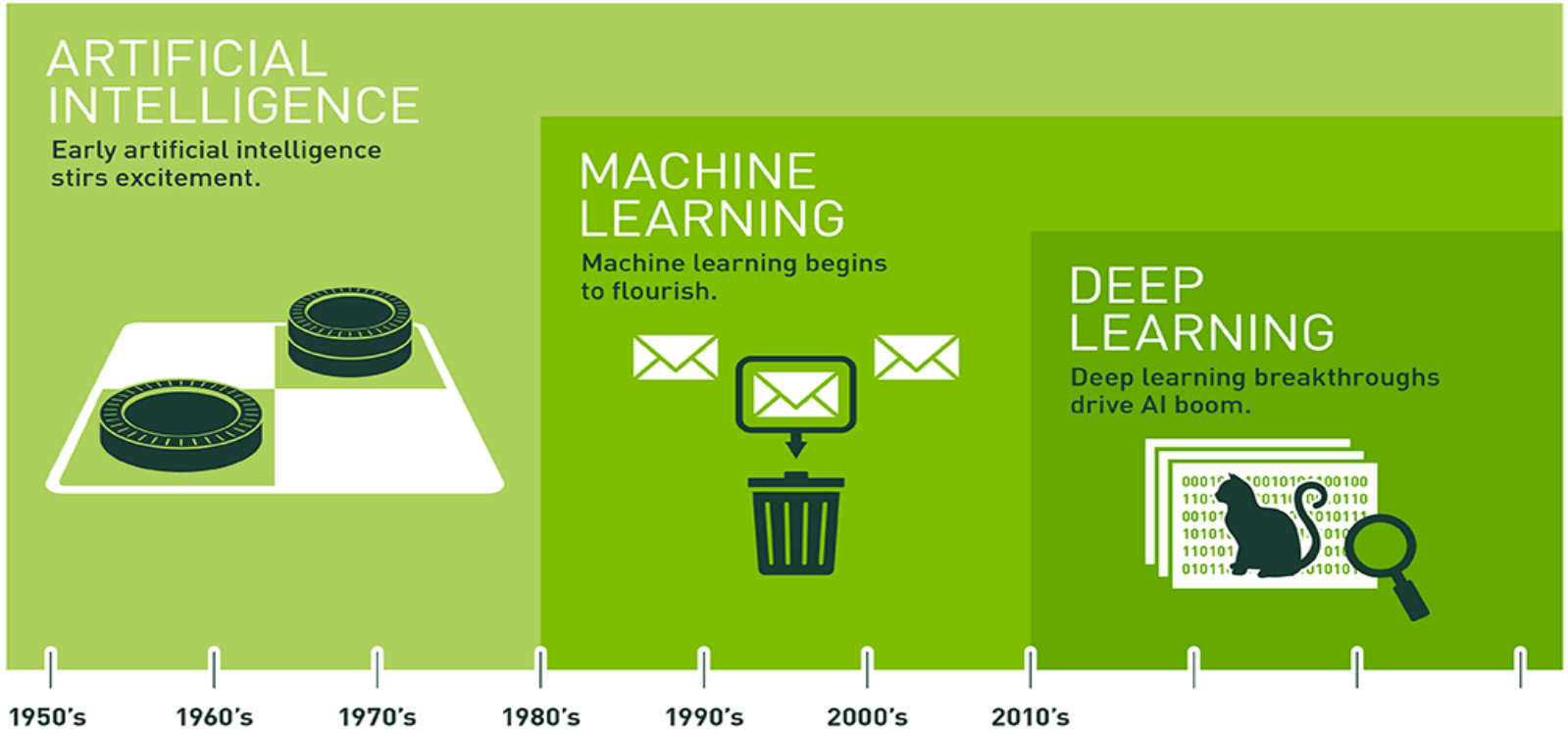
So what is Machine Learning

The term “Machine Learning” was coined by Arthur Samuel

- It is a field of study, where by computers were given the ability to learn without being explicitly programmed
- What we are looking at is a computer program that can use the image of cat
- And learns to identify that the image is truly a cat with almost 100% accuracy or in other words the least amount of error .002%

Image sourced: http://images4.fanpop.com/image/photos/16100000/Beautiful-Cat-cats-16123391-1280-800.jpg

Artificial Intelligence, Machine Learning and Deep Learning



Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

Image Sourced: https://blogs.nvidia.com/blog/2016/07/29/whats-difference-artificial-intelligence-machine-learning-deep-learning-ai/

Artificial Intelligence

- The Field of Artificial Intelligence was founded from a workshop held at Dartmouth College, New Hampshire, USA in 1956. The attendees of this workshop included:
  - Allen Newell (CMU – Carnegie Mellon University)
  - Herbert Simon (CMU – Carnegie Mellon University)
  - John McCarthy (MIT - Massachusetts institute of Technology)
  - Marvin Minsky (MIT - Massachusetts institute of Technology)
  - Arthur Samuel (IBM – International Business Machine)

- It was proposed that, **"Intelligence can in principle be so precisely described that a machine can be made to simulate it".**

The birth of the Perceptron

In 1958 Frank Rosenblatt invented the perceptron algorithm at the Cornell Aeronautical Laboratory which was funded by the US office of Naval research.

- This is the Mark I Perceptron machine
- The first implementation of the Perceptron algorithm.
- Connected to a camera with a 20 x20 Cadmium Sulfide Photcells to make a 400 pixel image
- On the right is an array of potentiometers that implemented the adaptive weights.

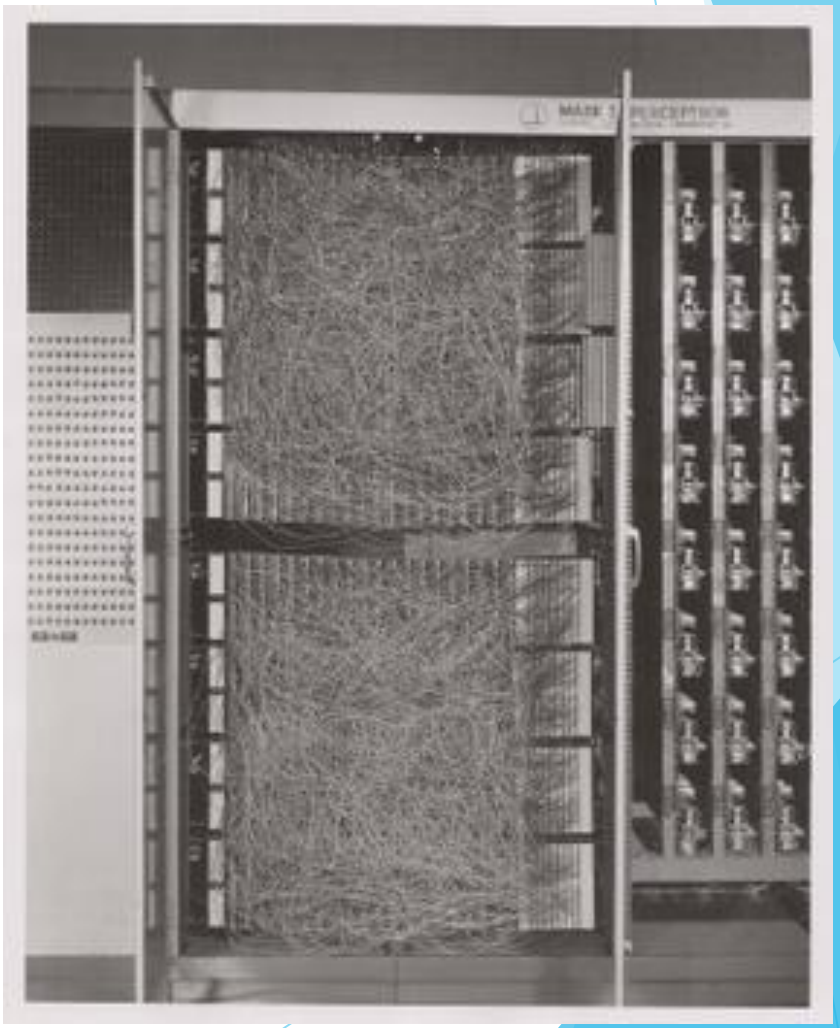


Image Sourced: https://en.wikipedia.org/wiki/Perceptron/

The First Demise of AI

The original perceptron however had a few limitations and issues.

- It was not good a recognising multiple classes
- In 1969 a book entitled Perceptron by Marvin Minsky and Seymour Papert showed that it was impossible for these classes of networks to learn an XOR function.
- The single layer Perceptron was only capable of learning linearly separable patterns

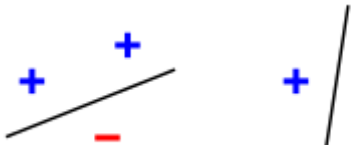


Image sourced: https://en.wikipedia.org/wiki/Linear_separability

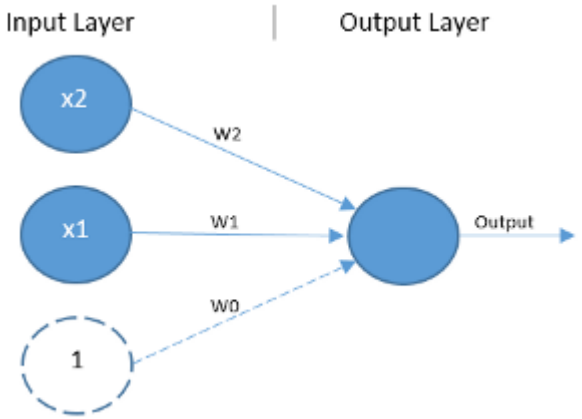


Image sourced: https://medium.com/@jayeshbahire/the-xor-problem-in-neural-networks-50006411840b

Machine Learning

In the 1980's research into neural networks started to make a come back, thanks to the rediscovery of **Backpropagation**

Additionally, instead of the **single layer Perceptron**, we now had a **multi-layer Perceptron** which was capable of the **XOR Function**

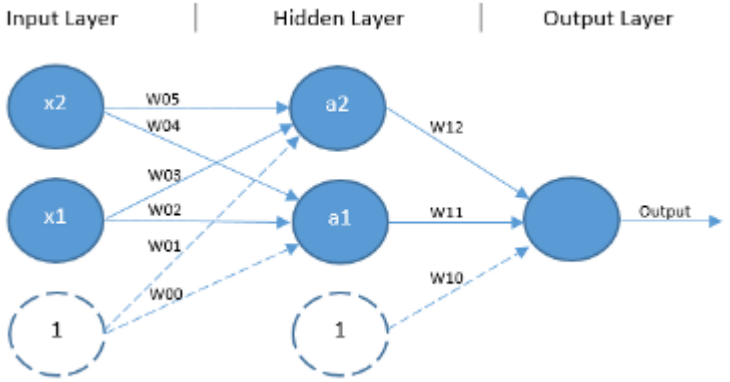


Image sourced: https://medium.com/@jayeshbahire/the-xor-problem-in-neural-networks-50006411840b

Example of Machine Learning using Naïve Bayes

A good example of machine learning during this time include the likes of spam filters using the Naïve Bayes algorithm

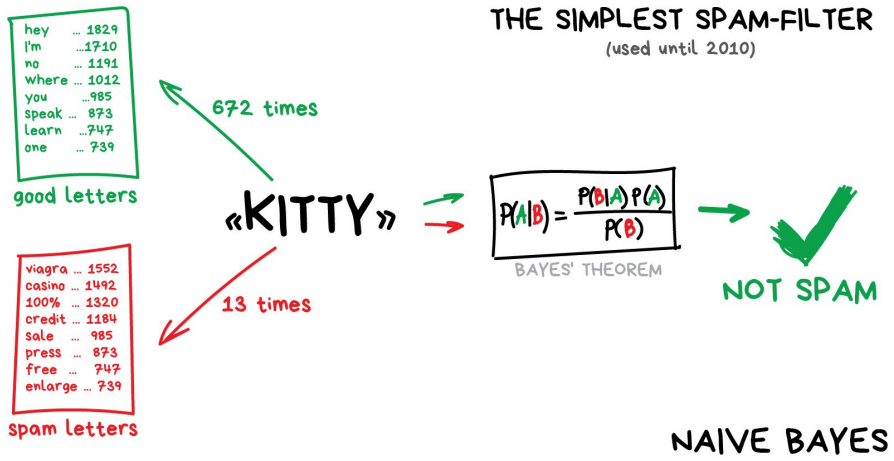


Image sourced: https://vas3k.com/blog/machine_learning/

Deep Learning

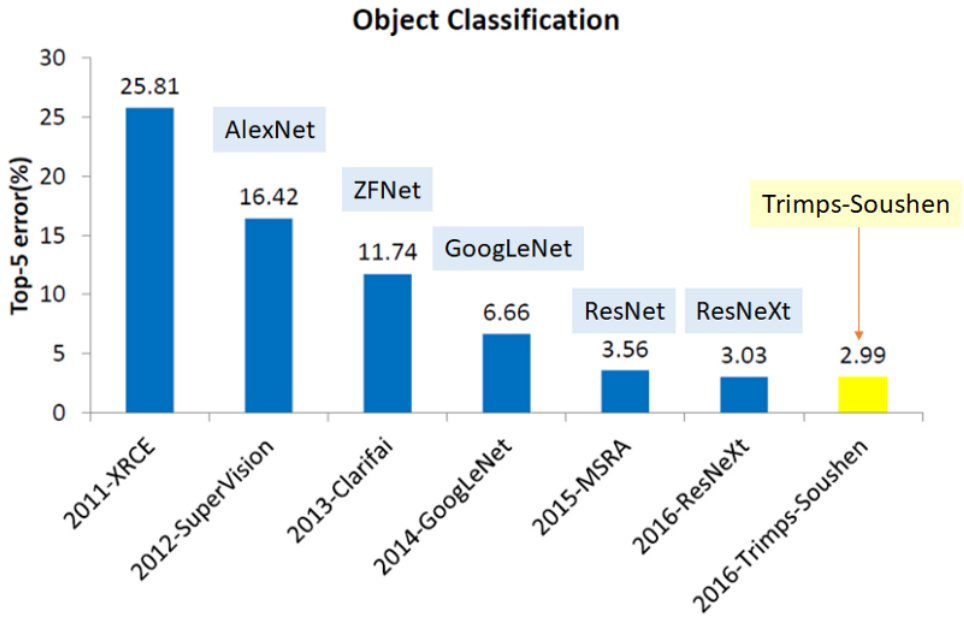
Deep Learning started to really make head ways in 2006, thanks to the Internet, Open Source and the GPU specifically NVIDIA GPUs.

- The Internet provided the Deep Learning community with a wealth of data often pre **labelled data**, which help with the **Training** Process
- The Deep Learning Community made their resources Open Source, allowing everyone to have access to these resources.
- GPU's were able to do Matrix multiplication fast. Much faster than CPUs.

Deep Learning allowed us to create complex Neural Networks

Deep Learning today

- Deep learning has made some remarkable breakthroughs in just the last nine years. For example in 2012 a Convolutional Neural Network called AlexNet was used on the annual Imagnet Object Classification competition. AlexNet had succeeded in lowering the error rate down to 16.42 from the previous error rate of 25.81



Types of Machine Learning

Types of Machine Learning

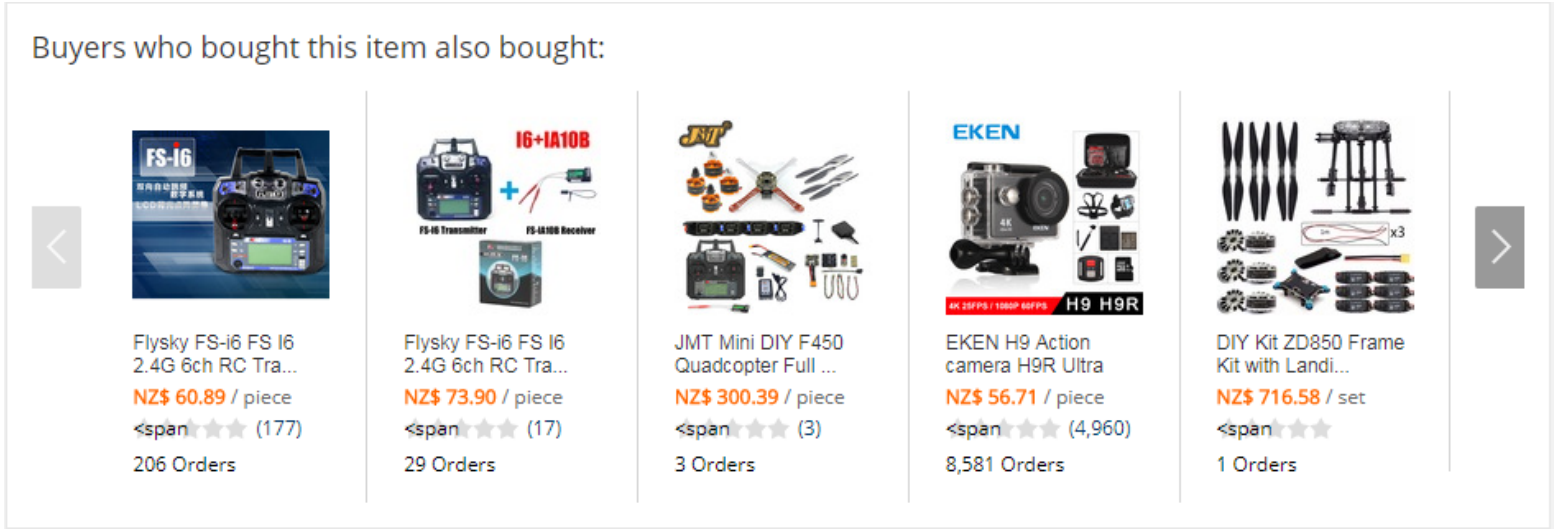
There are three types of machine learning:

- Supervised Learning
  - Labelled data provided (we will focus on this initially)
- Unsupervised Learning
  - No Labelled data provided
- Reinforcement Learning
  - Reward System

What is Supervised Learning

Supervised Learning is widely used in a majority of applications on the market today

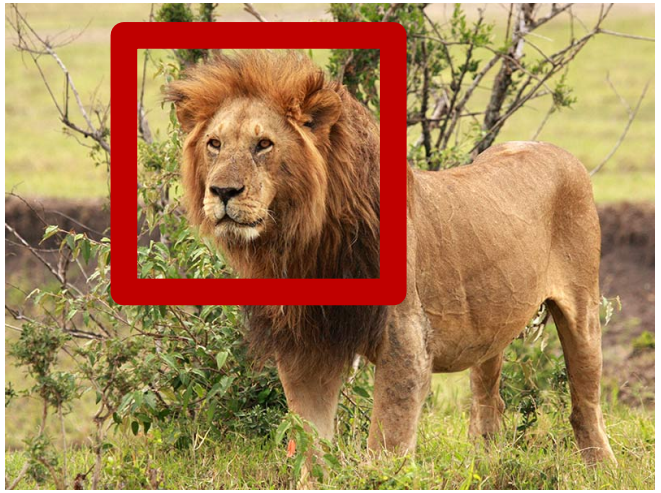
An example of this may be a web based application that recommends a product to you, based on your browsing and purchasing history



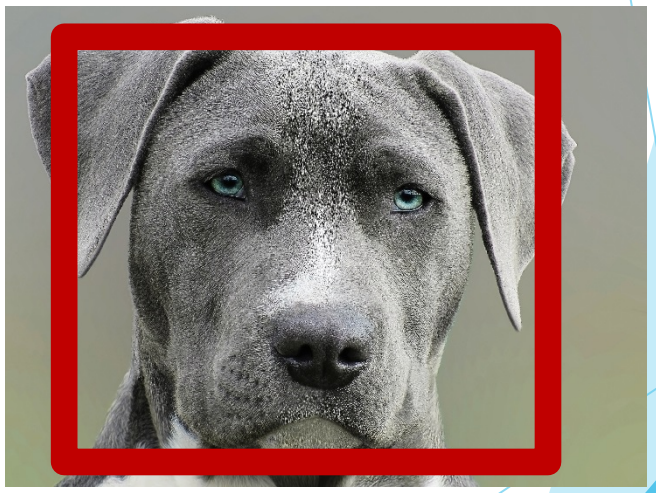
What is Supervised Learning Expanded

Another example is an image detection system that can Identify animals for a new wildlife application

Giraffe



Lion



Dog

How does Supervised Learning work

So how does supervised learning work?

To start off with we need to know what our **input (x)** is, as this will be used to predict our **output (y)**.

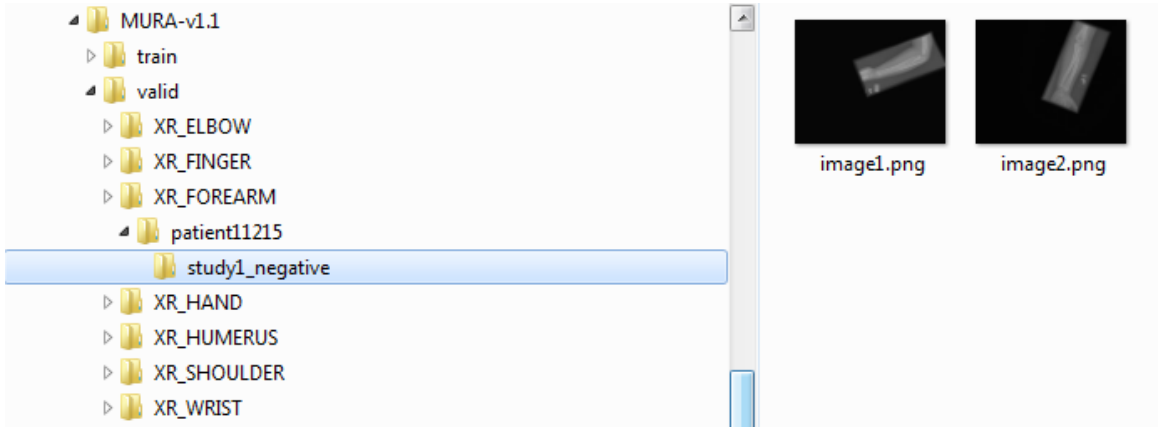
Input(x)	Output(y)	Applied	Type of NN
House Features	Price	Housing Market	NN
User info	Click on ad	Online retailers	NN
Images	Label object	Image Identification	CNN
Audio	Transcription	Speech Recognition	RNN
Language	French	Translator	RNN
Realtime Images, LIDAR, GPS	Position tracking	Autonomous Navigation	Hybrid

Supervised Learning Dataset

- With Supervised learning we are always using a dataset of correct answers.
- This is what we call **labelled** data

train	4/05/2018 12:04 a....	File folder	
valid	4/05/2018 12:33 a....	File folder	
train_image_paths.csv	23/05/2018 3:27 a....	Microsoft Excel C...	2,367 KB
train_labeled_studies.csv	23/05/2018 3:27 a....	Microsoft Excel C...	761 KB
valid_image_paths.csv	23/05/2018 3:27 a....	Microsoft Excel C...	206 KB
valid_labeled_studies.csv	23/05/2018 3:27 a....	Microsoft Excel C...	68 KB

145	MURA-v1.1/valid/XR_WRIST/patient11227/study1_positive/image1.png
146	MURA-v1.1/valid/XR_WRIST/patient11227/study1_positive/image2.png
147	MURA-v1.1/valid/XR_WRIST/patient11227/study1_positive/image3.png
148	MURA-v1.1/valid/XR_WRIST/patient11227/study2_negative/image1.png
149	MURA-v1.1/valid/XR_WRIST/patient11227/study2_negative/image2.png
150	MURA-v1.1/valid/XR_WRIST/patient11227/study2_negative/image3.png
151	MURA-v1.1/valid/XR_WRIST/patient11227/study3_negative/image1.png
152	MURA-v1.1/valid/XR_WRIST/patient11227/study3_negative/image2.png



145	MURA-v1.	0
146	MURA-v1.	0
147	MURA-v1.	0
148	MURA-v1.	0
149	MURA-v1.	0
150	MURA-v1.	0
151	MURA-v1.	0
152	MURA-v1.	0
153	MURA-v1.	0
154	MURA-v1.	0

Supervised Learning – Categorising problems

With supervised learning we are typically solving problems, these problems can be categorised as either a:

- **Regression** Problem
  - In this scenario we are trying to predict results within a continuous output. This means we are trying to map input variables to some continuous function
    - Example – given an image we are trying to predict a persons age
    - Example – given data about the size of houses on the market try to predict their price
- **Classification** Problem
  - Here we are trying to predict results in a discrete output, more precisely we are mapping variables into discrete categories
    - Example – given a patient with a tumour we need to predict whether the tumour is malignant or benign
    - Example – given data about the size of houses on the market we want to know if our house will sell above or below the asking price

What is Unsupervised Learning

Unsupervised learning learns to find patterns in data, patterns that we as humans may not understand, this is one of the benefits of unsupervised learning.

Data is clustered into groups using a number of algorithms that include:**Hierarchical clustering**: In this technique the algorithm builds a multilevel hierarchy of clusters by creating a cluster tree

- **k-Means clustering**: Here data gets partitions into k distinct clusters based on distance to the centroid of a cluster. (https://en.wikipedia.org/wiki/K-means_clustering)
- **Gaussian mixture models**: Algorithm builds a model in which clusters are a mixture of multivariate normal density components. (https://en.wikipedia.org/wiki/Mixture_model#Gaussian_mixture_model)
- **Self-organizing maps (SOM)**: Uses neural networks that learns the topology and distribution of the data. SOM is also used for dimensionality reduction. (https://en.wikipedia.org/wiki/Self-organizing_map)
- **Hidden Markov models**: Simply uses observed data to recover the sequence of states. (https://en.wikipedia.org/wiki/Hidden_Markov_model)
- **Hierarchical clustering**: In this technique the algorithm builds a multilevel hierarchy of clusters by creating a cluster tree. (https://en.wikipedia.org/wiki/Hierarchical_clustering)

Unsupervised Learning - Example

Suppose you had a basket and it is filled with some different types of fruits, your task is to arrange them as groups. You don't know any thing about the fruits, and this is the first time you have seen them. So how would you arrange them?

- You will take a piece of fruit and you will arrange them by considering the physical character of that particular fruit. suppose you have considered **colour.**
- Then you will arrange them on considering base condition as **colour.**
- Then the groups will be some thing like this.
- RED COLOUR GROUP: apples & cherry fruits.
- GREEN COLOUR GROUP: bananas & grapes.
- so now you will take another physical character such as **size** .
- RED COLOUR AND BIG SIZE: apple.
- RED COLOUR AND SMALL SIZE: cherry fruits.
- GREEN COLOUR AND BIG SIZE: bananas.
- GREEN COLOUR AND SMALL SIZE: grapes.

clustering comes under unsupervised learning.

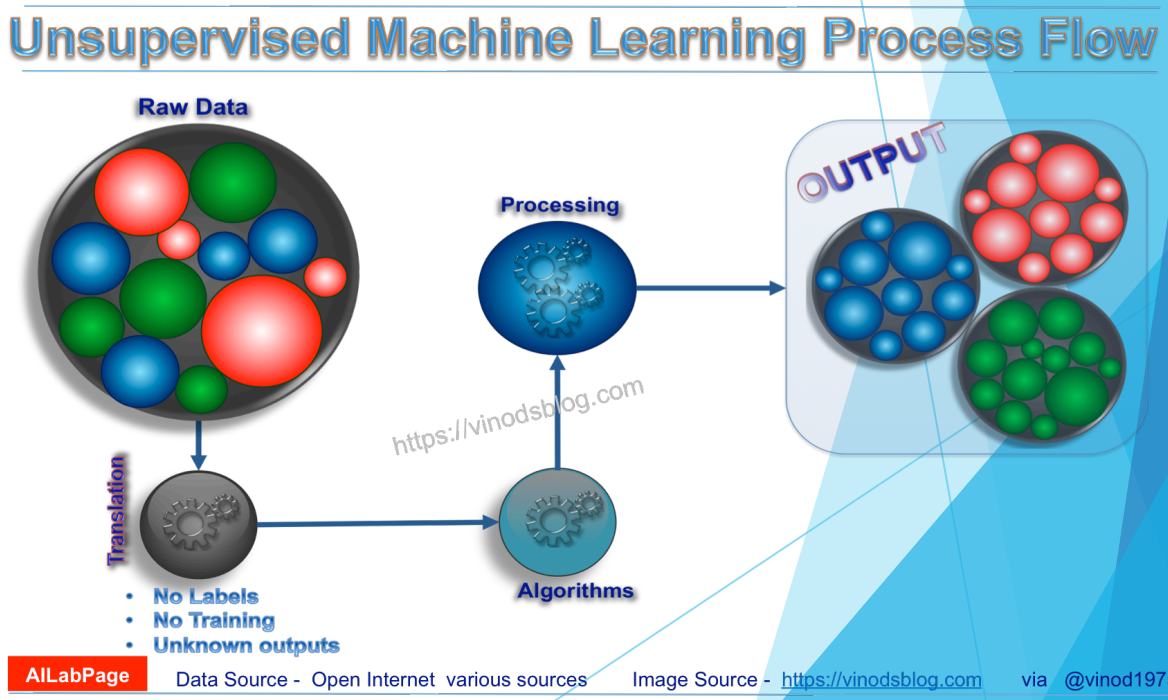


Image Source:
https://www.datasciencecentral.com/profiles/blogs/unsupervised-learning-an-angle-for-unlabelled-data-world

What is Reinforcement Learning

Reinforcement learning is a field of machine learning, in which an **agent** learns to perform tasks by **trial-and-error**, while receiving feedback in form of **reward** signals.

Solving such tasks involves dealing with high-dimensional state and action spaces, sparse reward signals, and uncertainties in the agent's observations.

In recent years, much of the successes of scaling up reinforcement learning to more complex tasks has come from leveraging the successes of deep neural networks, coining the term *deep* reinforcement learning.

Typically formulated using the **Markov Decision Process**.

Additional, material can be read https://en.wikipedia.org/wiki/Reinforcement_learning

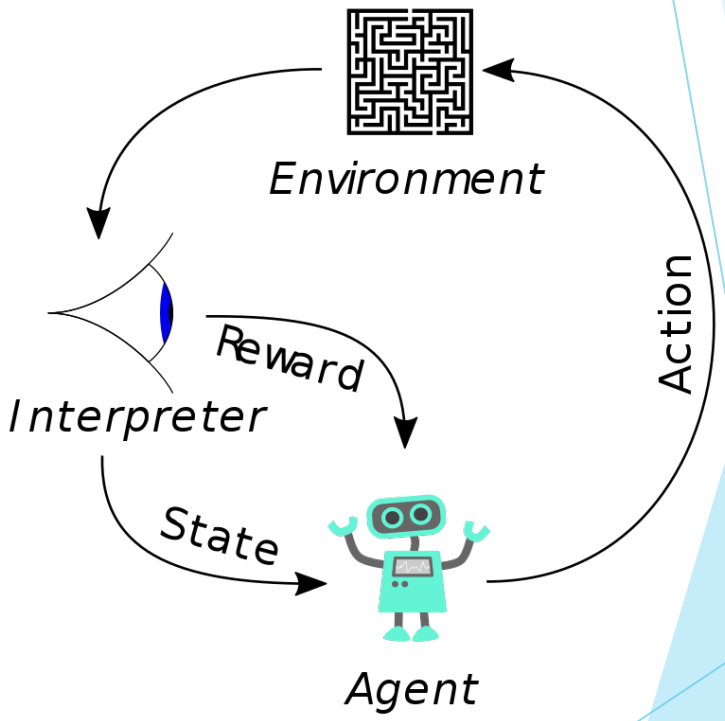


Image Source: https://en.wikipedia.org/wiki/Reinforcement_learning

Reinforcement Learning - Examples

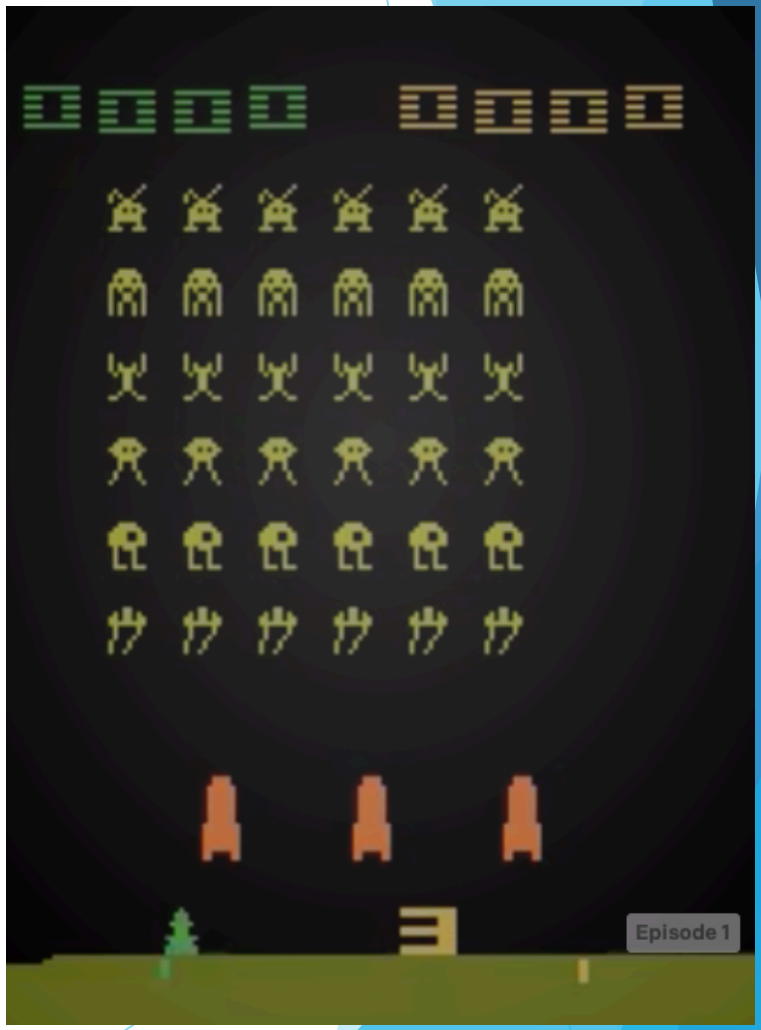
Reinforcement learning is used a lot in game theory and a great place to start is with the OpenAI Website (https://openai.com)

Here you will find resources that you can trial out that focus on reinforcement learning, many include games like Space invaders.

The site was founded by Elon Musk and others.

Gym Docuementation. (http://gym.openai.com/docs/)

Gym Github Source. (https://github.com/openai/gym)



Machine Learning Algorithm’s

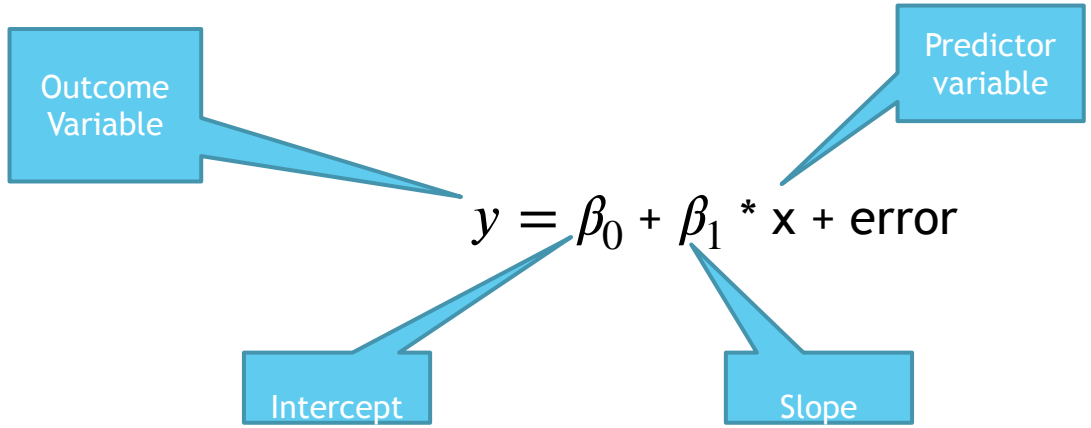
List of Common Machine Learning Algorithms

- Linear Regression
- Logistic Regression
- Decision Tree
- SVM (Support Vector Machine)
- Naive Bayes
- kNN (K nearest neighbour)
- K-Means
- Random Forest
- Dimensionality Reduction Algorithms
- Gradient Boosting algorithms
  - GBM
  - XGBoost
  - LightGBM
  - CatBoost

What is Linear Regression

Linear Regression

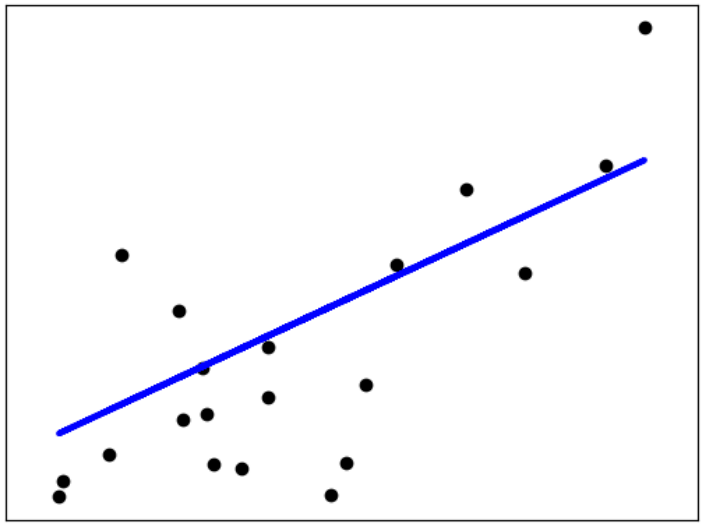
- Linear Regression is the simplest model to use to solve Machine Learning problems.
  - Remember that a regression problem is a predication of a continues output, whereby we are trying to map input variables to some continues function.
  - Linear regression works well when our dataset and all other unknown points lie on a hyperplane and the maximum error is proportional to both the training quality and the adaptability of the original dataset.



What is Linear expanded

In this image we have our line fitting between data points, this line is called our hyperplane or regression line. From this we now have two decision boundaries

What we are wanting to find is the best **fit** for our regression line through the dots



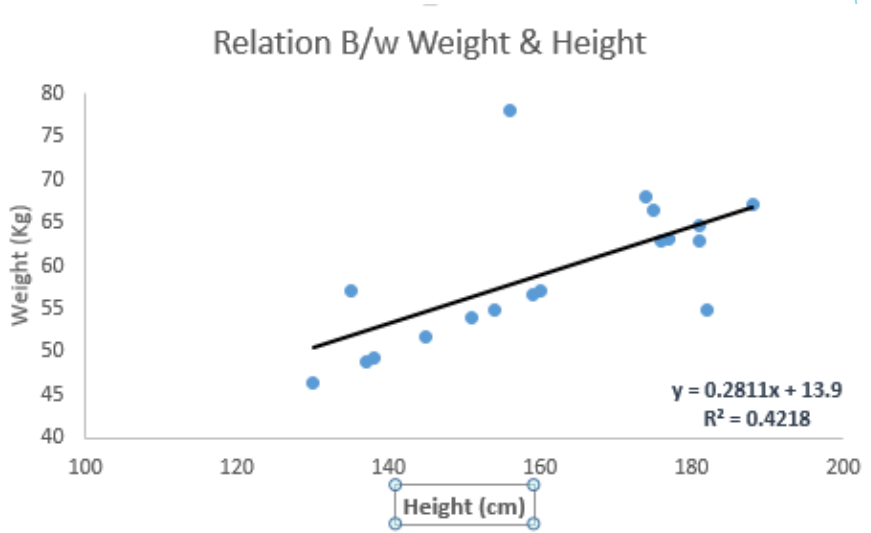
Linear Regression

Linear Regression is used to estimate real values, for example house sales, based on **continuous variable(s** Here, we establish the relationship between independent and dependent variables by fitting a bes line.

This best fit line is known as a regression line and is represented by a linear equation $y = mx + b$.

- In this equation:
- y – Dependent Variable
- m – Slope
- x – Independent variable
- b – Intercept

These coefficients **m** and **b** are derived based on minimising the sum of squared difference of distance between data points and regression line.



What is Linear Regression expanded

- Lets start off with some key definitions for this course

- Dataset = Our training set in this case it will be house prices
- m = is the number of entries in our training set
- x = is our input variable/feature
- y = is our output variable/ our target variable

Square Meters (x)	Price (y)
280	700
240	650
200	550
180	500
150	425

- So if we look at our matrix
- And count the number of rows in the training set, m =5,
- x=square meters and y=price

- Now if we wanted to retrieve a selected entry in our training set we would select the I'th training example.

- For example
- (x^i, y^i) = if subscript I was the second entry in our dataset then (x=240, y=650)
- (x^i, y^i) = if subscript I was the fourth entry in our dataset then (x=180, y=500)

Representing our Hypothesis

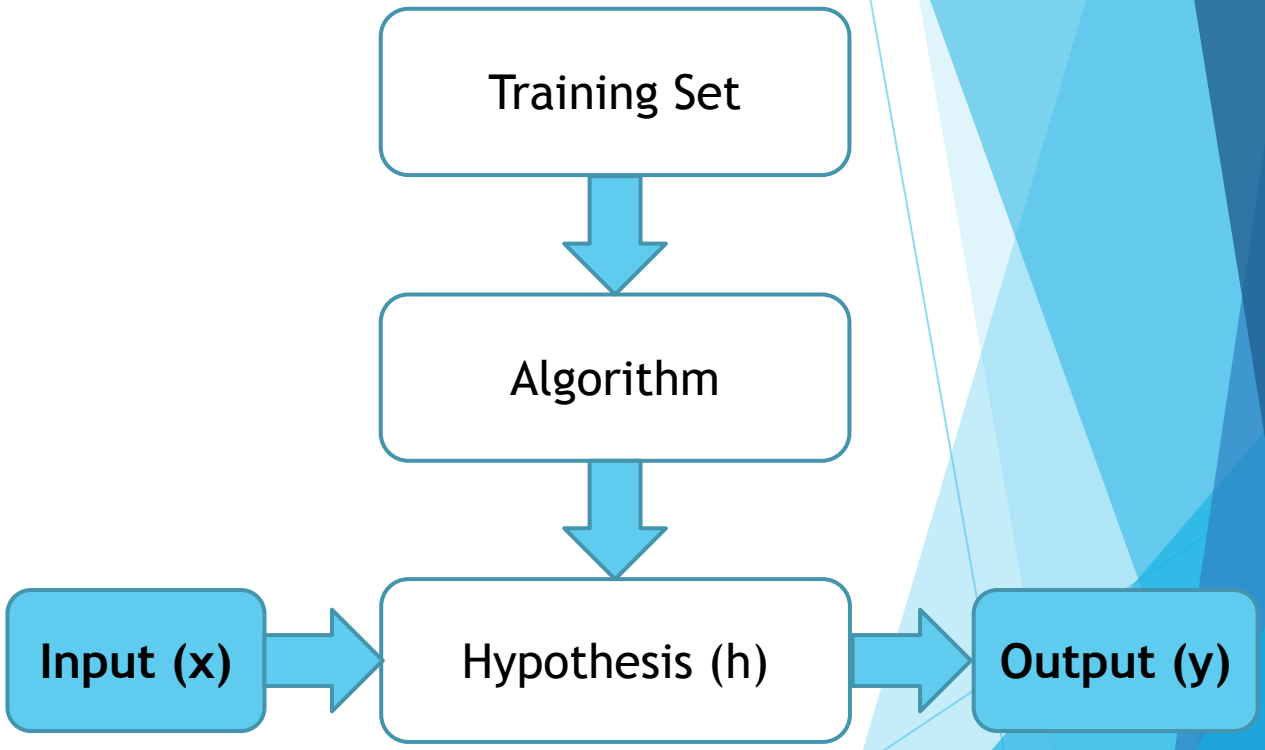
With supervised learning we start off with a training set.

This training set then goes through an algorithm which outputs a hypothesis (denoted by a lowercase **h)**

The hypothesis takes in our input value **x** this is our square meters for a particular house

The Hypothesis then provides an output value **y** this is our predicted house value

So what we are doing is having our Hypothesis (h) map **x** to **y**



Modelling our Hypothesis

We will now model our hypothesis starting with an expression.

What we are wanting to do is find the function

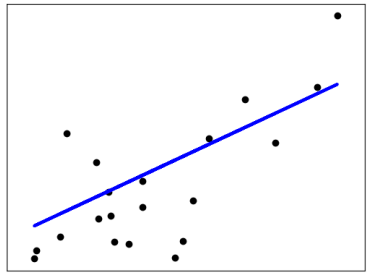
$$\theta_0 + \theta_1 x$$

When we apply our function our graph provides us with a linear regression of predicted house price based on one variable.

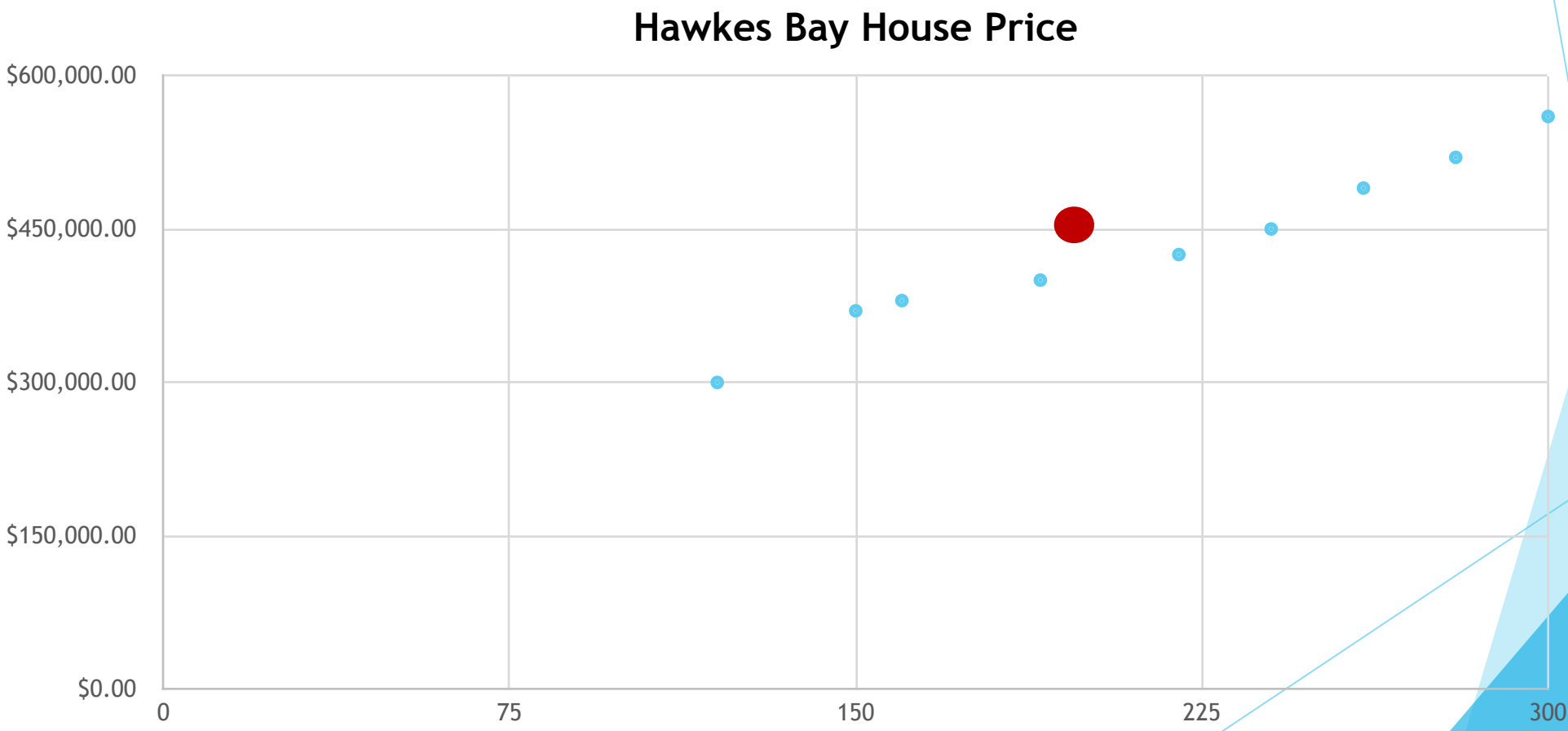
This is will be the bases for our building blocks to more complex Machine Learning Models

$$h_\theta(x) = \theta_0 + \theta_1 x$$

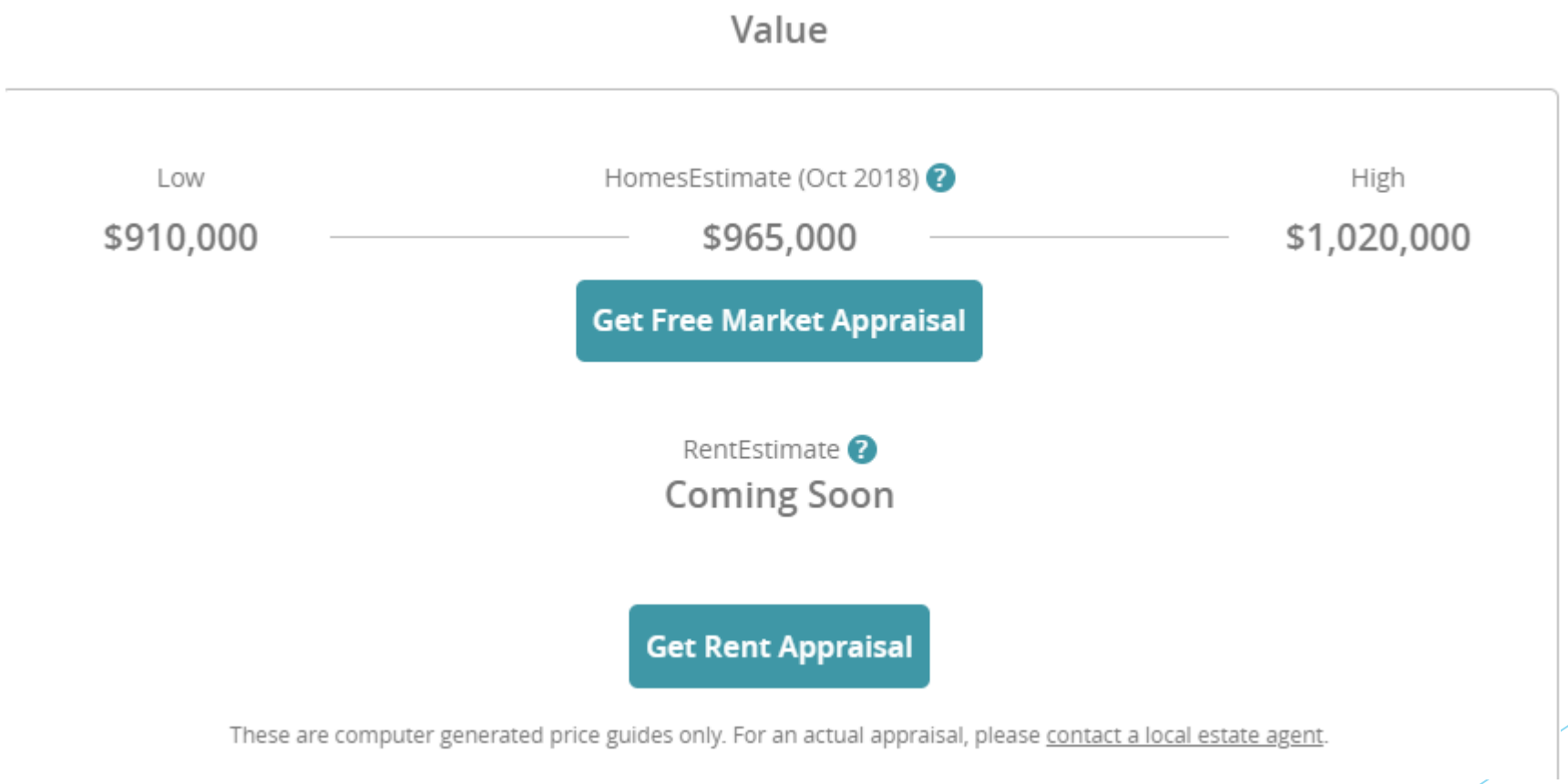
$$h(x) = \theta_0 + \theta_1 x$$



Linear Regression – Housing Example



Linear Regression – Housing Example 2



Linear Regression With Multiple Features

- Let's start off with some key definitions for this course.
- Linear regression with multiple variables is also known as "multivariate linear regression".
- We will now introduce notation for equations where we can have any number of input variables.
  - $x_j^{(i)}$ = value of feature *j* in the i^{th} training example
  - $x^{(i)}$ = The input (features) of the i^{th} training example
  - *m* = The number of training examples
  - *n* = The number of features

Linear Regression with Multiple Features

Size square meters (x1)	Number of bedrooms (x2)	Number of Floors (x3)	Age of home (x4)	Price (y)
240	4	1	10	690
160	3	2	15	500
180	3	2	20	550
120	2	1	25	425
…	…	…	…	…

$n = 4$
$m = 47$

$$x^{(2)} = \begin{bmatrix} 160 \\ 3 \\ 2 \\ 15 \end{bmatrix}$$

$\epsilon \mathbb{R}^4$ $x^{(2)}$ *is a 4* dimensional vector

$\mathbb{R}^n$ *n* dimensional feature vector

$x_3^{(2)} = 2$

Multiple Features

- The multivariable form of the hypothesis function accommodating these multiple features is as follows:

  $$h(\theta) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \cdots + \theta_n x_n$$

- In order to develop intuition about this function, we can think about θ_0 as the basic price of a house, θ_1 as the price per square meter, θ_2 as the price per floor, etc. x_1 will be the number of square meters in the house, x_2 the number of floors, etc.

Multiple Features Continued

Using the definition of matrix multiplication, our multivariable hypothesis function can be concisely represented as:

$$h_\theta(x) = \begin{bmatrix}\theta_0 & \theta_1 & \theta_n\end{bmatrix} = \theta^t x \begin{bmatrix}x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5\end{bmatrix}$$

This is a vectorisation of our hypothesis function for one training example; see the lessons on vectorisation to learn more.

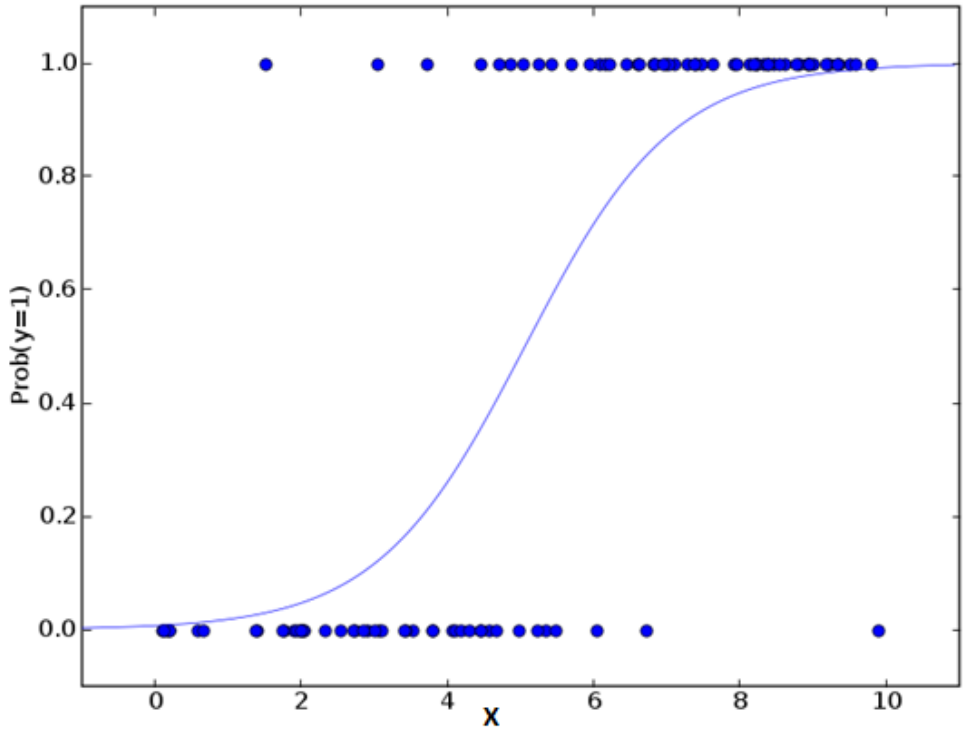
Logistic Regression

Logistic Regression is a classification algorithm and not a regression algorithm. It is used to estimate discrete values:

- Binary values like 0/1
- Yes/No
- True/False)
- based on a given set of independent variable(s).

In simple words, it predicts the probability of occurrence, of an event by fitting data to a **logit function**. Hence, it is also known as **logit regression**.

Since, it predicts the probability, its output values lies between 0 and 1 (as expected).



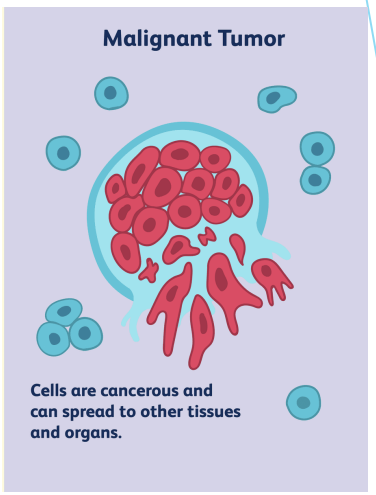
Logistic Regression - Example

These are some examples were Logistic regression can be used:

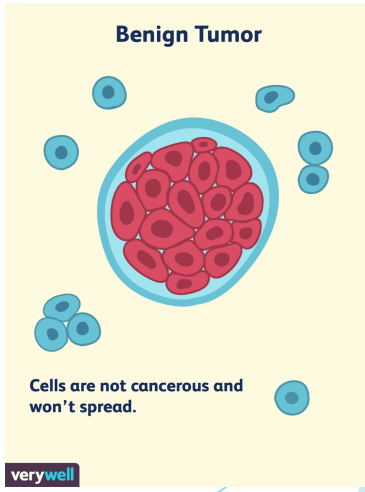
- Cat/ non-Cat image classification
- Malignant/ Non-Malignant

Cat = 1

non-Cat = 0



Malignant = 1



non-Malignant = 0

Decision Tree

- A Decision Tree is a type of supervised learning algorithm that is mostly used for classification problems. Surprisingly, it works for both categorical and continuous dependent variables.

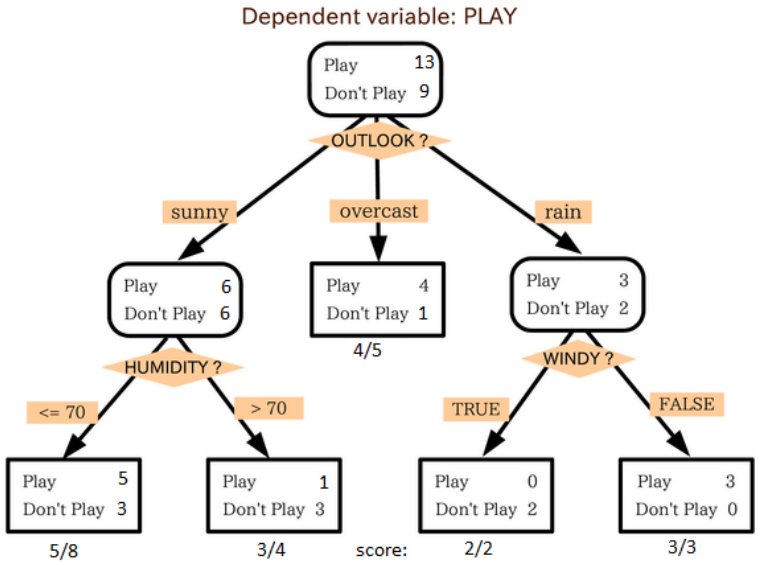
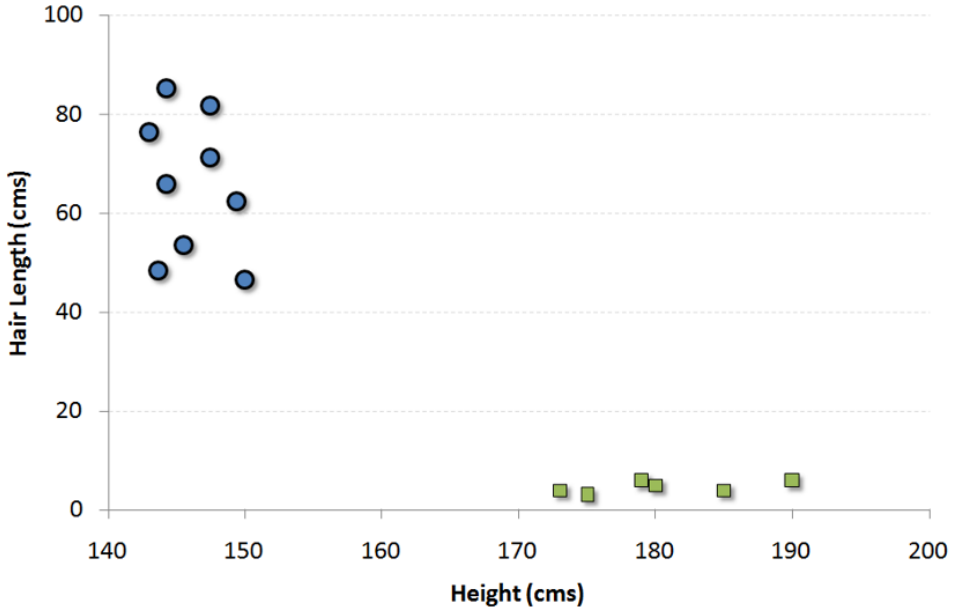
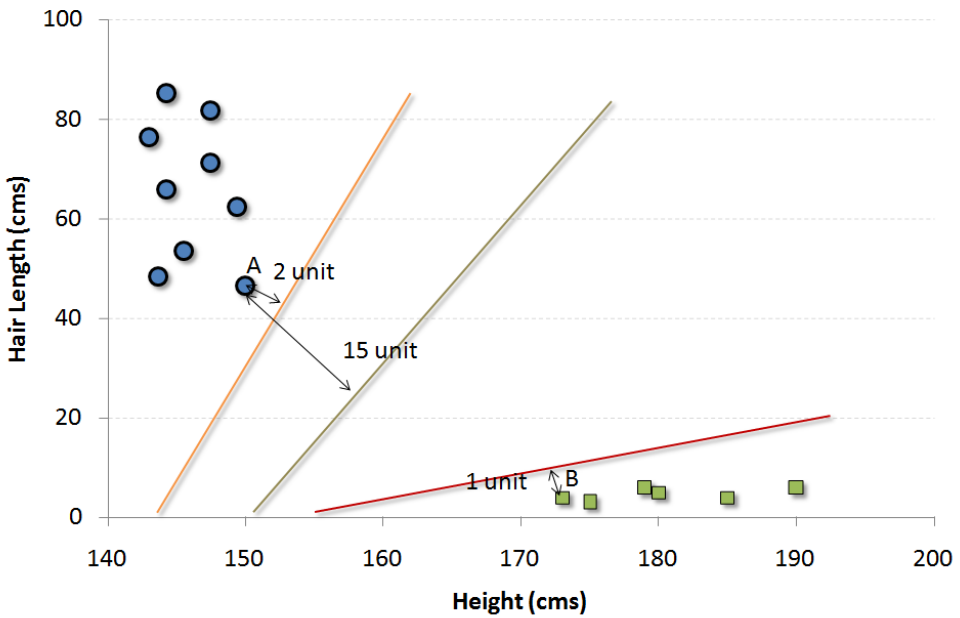


Image Source: https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/

SVM (Support Vector Machine)

SVM is a classification method. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate.





SVM (Support Vector Machine)

An SVM model is a representation of the examples as points in space.

- Mapped so that the examples of the separate categories are divided by a clear gap, that is as wide as possible.
- New examples are then mapped into that same space.
- And predicted to belong to a category based on the side of the gap on which they fall.

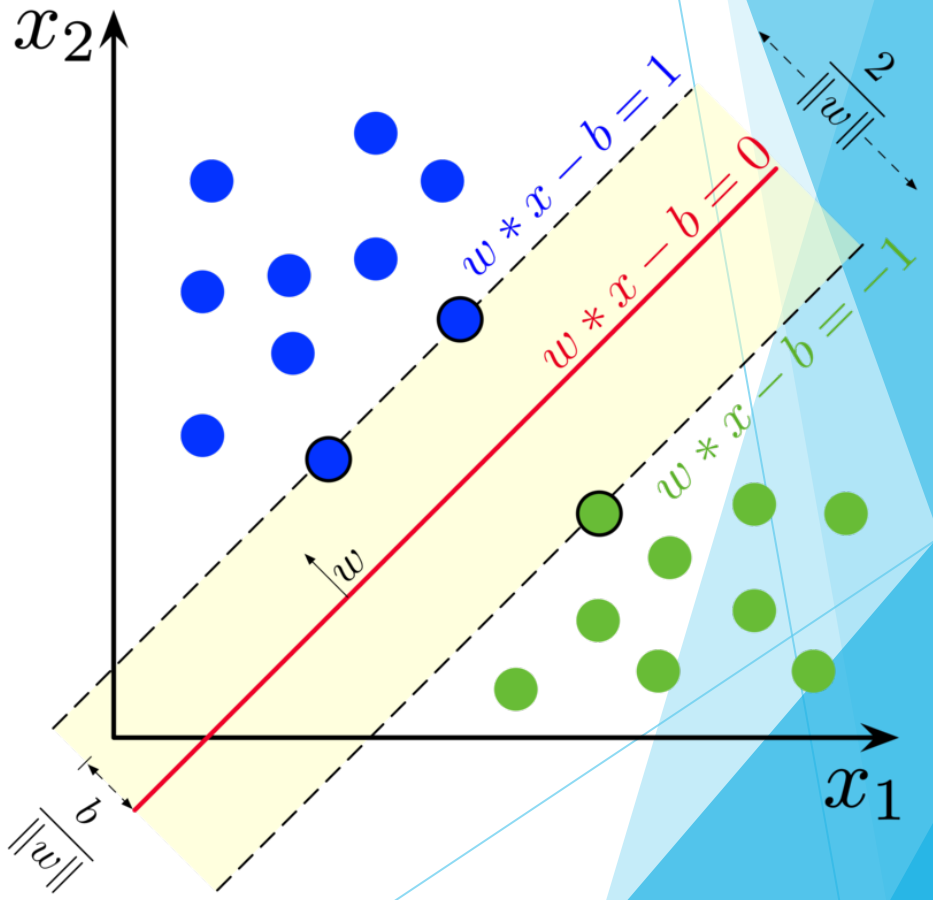


Image Sourced: https://en.wikipedia.org/wiki/Support-vector_machine

SVM (Support Vector Machine)

However, sometimes data is not so well placed.

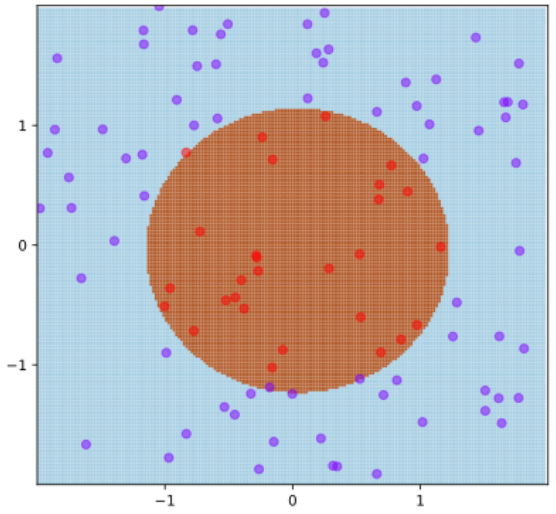
For example, here we have a line with data points represented as green and blue dots.

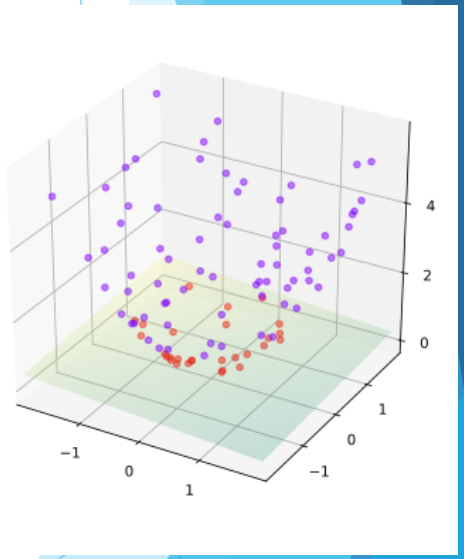
These dots are not linearly separable.

However, with SVM there is a trick we can use, which makes SVM a very resourceful algorithm for Machine Learning and Deep Learning.

SVM (Support Vector Machine)

- SVM can perform both Linear Classification and Non-Linear Classification using the **kernel trick(kernel method).**
- The **kernel methods** are used for pattern analysis, that can operate in high dimensional, implicit feature space without ever computing coordinates of the data in that space.
- In simplistic terms we can bend and manipulate our data until we can split our data.
- There are many other algorithms that operated with **kernels(to name a few):**
  - **PCA(Principle component analysis)**
  - **Kernel Perceptron**
  - **Ridge Regression**





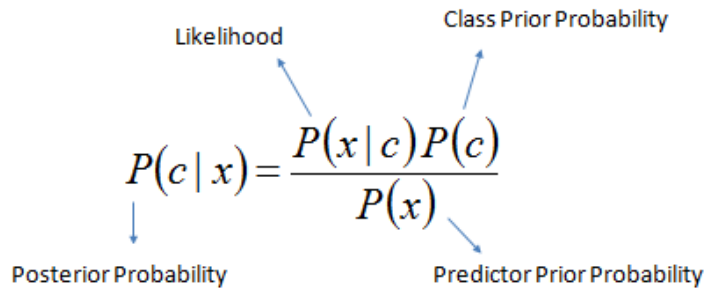
SVM (Support Vector Machine)

- Classification of images can be performed using SVMs
- Image segmentation systems can make use of SVMs
- Handwritten characters can be recognised with SVMs
- Uses in biological and other sciences

Data with lots of error.
    Discriminator depends entirely on the few nearest data points
Choosing the wrong kernel
    kernel selection is trial and error
Large datasets
    Calculating the kernel is expensive
Each of these requires a human in the loop to make judgement calls.

Naïve Bayes

- It is a classification technique based on Bayes' theorem with an assumption of independence between predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. For example, a fruit may be considered to be an apple if it is red, round, and about 3 inches in diameter. Even if these features depend on each other or upon the existence of the other features, a naive Bayes classifier would consider all of these properties to independently contribute to the probability that this fruit is an apple.
- Naive Bayesian model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.
- Bayes theorem provides a way of calculating posterior probability P(c|x) from P(c), P(x) and P(x|c). Look at the equation below:
  - *P*(*c*|*x*) is the posterior probability of *class* (*target*) given *predictor* (*attribute*).
  - *P*(*c*) is the prior probability of *class*.
  - *P*(*x*|*c*) is the likelihood which is the probability of *predictor* given *class*.
  - *P*(*x*) is the prior probability of *predictor*.



$$P(c \mid \mathrm{X}) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

Naïve Bayes - Example

- **Example:** Let's understand it using an example. Below I have a training data set of weather and corresponding target variable 'Play'. Now, we need to classify whether players will play or not based on weather condition. Let's follow the below steps to perform it.
- **Problem:** Players will play if the weather is sunny, is this statement correct?
- We can solve this by using the previous equation, so P(Yes | Sunny) = P(Sunny | Yes) * P(Yes) / P (Sunny)
- Here we have P (Sunny |Yes) = 3/9 = 0.33, P(Sunny) = 5/14 = 0.36, P(Yes)= 9/14 = 0.64
- Now, P (Yes | Sunny) = 0.33 * 0.64 / 0.36 = 0.60, which has higher probability.

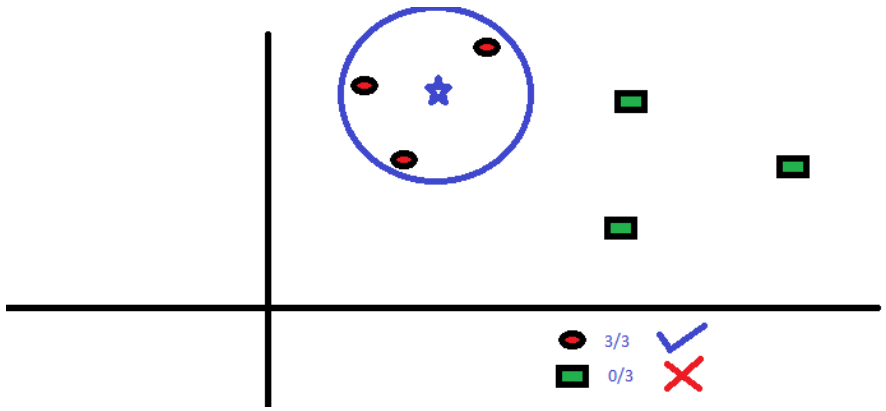
Weather	Play
Sunny	No
Overcast	Yes
Rainy	Yes
Sunny	Yes
Sunny	Yes
Overcast	Yes
Rainy	No
Rainy	No
Sunny	Yes
Rainy	Yes
Sunny	No
Overcast	Yes
Overcast	Yes
Rainy	No

Frequency Table		
Weather	No	Yes
Overcast		4
Rainy	3	2
Sunny	2	3
Grand Total	5	9

Likelihood table				
Weather	No	Yes		
Overcast		4	=4/14	0.29
Rainy	3	2	=5/14	0.36
Sunny	2	3	=5/14	0.36
All	5	9		
	=5/14	=9/14		
	0.36	0.64		

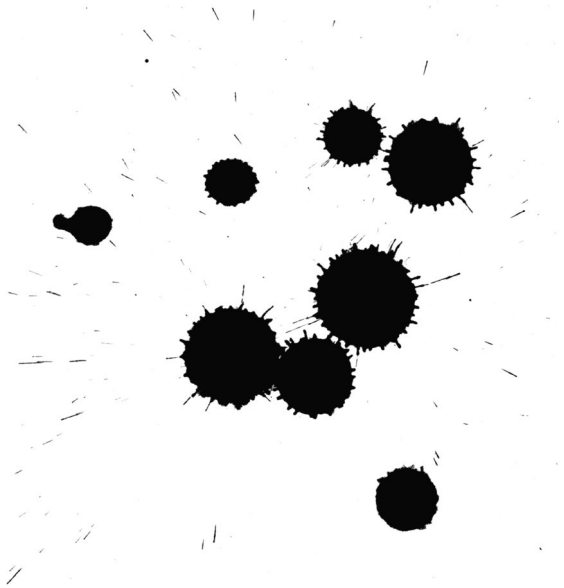
kNN (k_ Nearest Neighbors)

- It can be used for both classification and regression problems. However, it is more widely used in classification problems in the industry. K nearest neighbors is a simple algorithm that stores all available cases and classifies new cases by a majority vote of its k neighbors. The case being assigned to the class is most common amongst its K nearest neighbors measured by a distance function.
- These distance functions can be Euclidean, Manhattan, Minkowski and Hamming distance. First three functions are used for continuous function and fourth one (Hamming) for categorical variables. If K = 1, then the case is simply assigned to the class of its nearest neighbor. At times, choosing K turns out to be a challenge while performing kNN modeling.



K-Means

- It is a type of unsupervised algorithm which solves the clustering problem. Its procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters). Data points inside a cluster are homogeneous and heterogeneous to peer groups.
- Remember figuring out shapes from ink blots? k means is somewhat similar to this activity. You look at the shape and spread, to decipher how many different clusters / population are present!



Preprocessing of Data

Working with Raw Data

When working with **Raw Data** it is often not in an optimised format for being used with Machine Learning/Deep Learning algorithms. The process of getting our **Raw Data** into a usable format is the **preprocessing** stage.

Data **preprocessing** is an integral step in Machine Learning/ Deep Learning, as the quality of data and the useful information that can be derived from it directly affects the ability of our model to learn; therefore, it is extremely important that we preprocess our data before feeding it into our model.

Step 1 - Data Selection

It’s import to know what subset of data you will need to be selecting to work with. It may be that you need only a small subset of data to work with or the complete data set.

More data may give you better results however may increase your training time. Likewise, a small subset of data that is feature selected well may have better results with a faster training time.

It is import to understand the problem you are trying to solve.

Step 2 - Preprocessing Data

This step looks at getting our data in a form that we can use.

- **Formatting**: The data you have selected may not be in a format that is suitable for you to work with. The data may be in a relational database and you would like it in a flat file, or the data may be in a proprietary file format and you would like it in a relational database or a text file.

- **Cleaning**: Cleaning data is the removal or fixing missing data. There may be data instances that are incomplete and do not carry the data you believe you need to address the problem. These instances may need to be removed. Additionally, there may be sensitive information in some of the attributes and these attributes may need to be anonymised or removed from the data entirely.

- **Sampling**: There may be far more selected data available than you need to work with. More data can result in much longer running times for algorithms and larger computational and memory requirements. You can take a smaller representative sample of the selected data that may be much faster for exploring and prototyping solutions before considering the whole dataset.

Step 3 - Transforming Data

There are three common data transformations scaling, attribute decompositions and attribute aggregations. This step is also referred to as feature engineering.

- **Scaling**: The preprocessed data may contain attributes with a mixtures of scales for various quantities such as dollars, kilograms and sales volume. Many machine learning methods like data attributes to have the same scale such as between 0 and 1 for the smallest and largest value for a given feature. Consider any feature scaling you may need to perform.

- **Decomposition**: There may be features that represent a complex concept that may be more useful to a machine learning method when split into the constituent parts. An example is a date that may have day and time components that in turn could be split out further. Perhaps only the hour of day is relevant to the problem being solved. consider what feature decompositions you can perform.

- **Aggregation**: There may be features that can be aggregated into a single feature that would be more meaningful to the problem you are trying to solve. For example, there may be a data instances for each time a customer logged into a system that could be aggregated into a count for the number of logins allowing the additional instances to be discarded. Consider what type of feature aggregations could perform.

Activities This Week

In this weeks activity you will look at the following preprocessing and transforming tools:

- Mean Removal
- Scaling
- Normalization
- Binarization
- One Hot Encoding