Machine Learning & AI Notes

Basics

Artificial intelligence (AI) is currently in still in its conceptual phase. AI is an overarching term many people reference when they are actually talking about machine learning or deep learning.

Artificial Narrow Intelligence (ANG) is the level of intelligence that we are achieving right now. ANG describes intelligence which can only perform one task very well and nothing else. Having an AI play chess, make purchase recommendations or process natural language are all examples of narrow intelligence. Even self driving cars are considered as a type of narrow9 intelligence, using a collection ANGs.

Aritificial General Intelligence (AGI) or human level AI can perceive and reason with its environment similar to how a human does. This type of AI is what people tend to think about when just refering to AI. Some people are of the opinion that this level of intelligence is far off while others think it is just around the corner. Once this intelligence in a machine has been achieved it won't be long after that we hit the level of super intelligence.

Artificial Super Intelligence (ASI) is when the intelligence becomes much smarter than humans in practically every field. Once we reach AGI, the rate at which the machine learns increases exponentially until it reaches super intelligence.

If you want to read into more detail about all of the above, we recommend that you read **Nick Bostrom's Superintelligence: Paths, Dangers, Strategies.**

Machine Learning is a discipline of AI where instead of coding a program to perform in a specific way and return an expected answer, we can give the ability to learn how to complete a task to a computer.

Deep Learning is a subset of machine learning and is inspired by the biology of how human's learn and how each neuron in our brain is connected to the other. Deep learning techniques involve using neural networks to learn. A neural network is made up of multiple **neurons** or **nodes** which take input from x and output a prediction y. Each input x inputs its data through each neuron. The neural network will figure out each nueron's feature itself given x and y in the training set.

Examples to look at that use deep learning:

- Skype real time translation.
- Nueral Doodle.
- Enlitic, used for medical imaging.

QuickDraw
Edges2Cats
RunwayML

Structured data represents the likes of databases. Each feature has a well defined meaning.

Unstructured data covers raw audio, images and text.

This step of capturing patterns from data is called **fitting** or **training** the model. The data used to **fit/train** the model is called the **training data**.

Computations of a neural network are organised in two way:

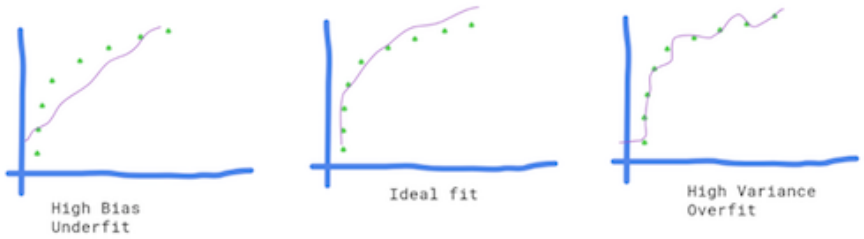
- A forward pass or a forward propagation step where the output of the neural network is computed.
- This is followed by a backward pass or a back propagation step, which is used to compute gradients or derivatives.

Labels or **Classes** are the same thing and it's down to personal preference which one you use. Whenever we need to classify/label text or an image for a our a prediction we train the machine on data that has been correctly labelled or classified.

A **model** is what we train with our data so we are able to get a computer to learn how to complete the task we want.

Bias and Variance

While developing a machine learning model, the two major sources that produce errors are from bias and variances. **Bias** denotes the fluctuations in accuracy with changing training data i.e. Underfitting. **Variance** is the sensitivity of the model to the input data i.e. Overfitting.



Supervised Learning

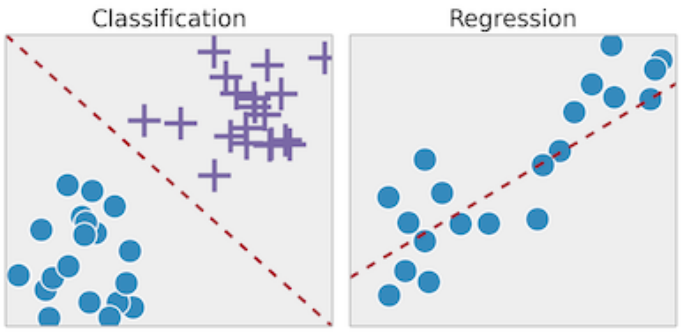
Refers to the process of building a machine learning model that is based on labelled training data. You have a input variable (x) and output variable (y) and you use an algorithm to learn the mapping function from the inout to the output.

f(x) = y

The algorithm iteratively makes predictions on the training data and is corrected by the data representing the answers.

Problems can be grouped in regression and classification:

- **Classification** - The output variable is a category. Eg. “disease” and “no disease”
- **Regression** - The output variable is a real value. Eg. “price” or “weight”



Common algorithms for supervised learning:

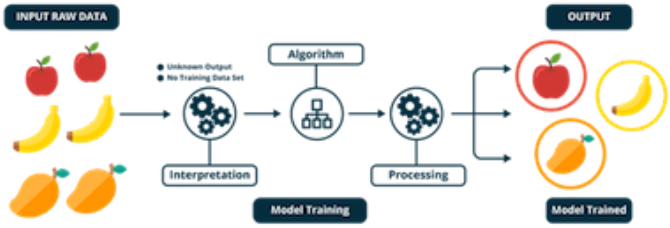
- Logistic regression
- Naive Bayes
- Support Vector Machines
- Artificial Neural Network

For more info: Machine Learning (Breakdow'n)

Unsupervised Learning

Unsupervised machine learning is a class of techniques to find patterns that describe the structure of “unlabled” data (data that has not been classified or categorised).

The most common type of unsupervised learning is clustering. **Clustering** algorithms run through your data and find natural clusters if they exist, e.g K-Means or Hierarchical Clustering, which create clusters based on features of the data.



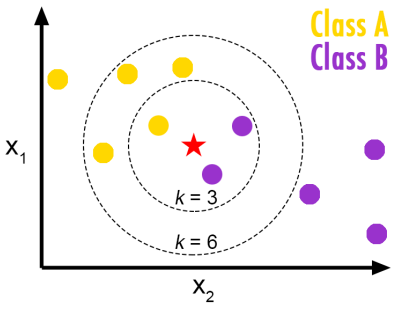
K-Nearest Neighbour

The K-Nearest neighbours Algorithm (KNN) is one of the simplest and most used classification algorithms. Its purpose is to take data points that are separated into several classes to predict the classification of a new sample point. KNN is:

- **Non-parametric:** this means it does not make any assumptions on the underlying data distribution; the model structure is determined from the data.
- **A Lazy Algorithm:** this means there is no explicit training phase and this lack of generalisation means that KNN keeps all the training data which is all (or mostly) needed during the testing phase.

How it works:

1. A positive integer k is specified as well as a new sample.
2. We select the k entires in the database which are closest to the new sample.
3. We then find the most common classification of the entries.
4. The classification is then given to the new sample.



Pros	Cons
No assumptions about data	Computationally expensive - stores all the training data
Simple Algorith	Sensitive to irrelevant features and scale of data
Relatively high accuracy	Stores all or most of training data
Versatile - used for classification or regression	Prediction stage may be slow

Multiple Classification

Mutliple classification happens in two forms; a sample is placed into one of multiple classes or the sample can be given multiple labels.

Multi-class classification means a classfication task with one or more classes where each label is mutually exclusive; it makes the assumption that each sample is assigned to only one label.

Multi-label classification assigns a set of target labels to a sample. This is frequent in most real world applications such as categorising businesses or movies into several genres.

Linear Regression

Linear regression is a supervised learning method. It is used to predict values with a continuous range e.g. 0-100.

Simple Regression:

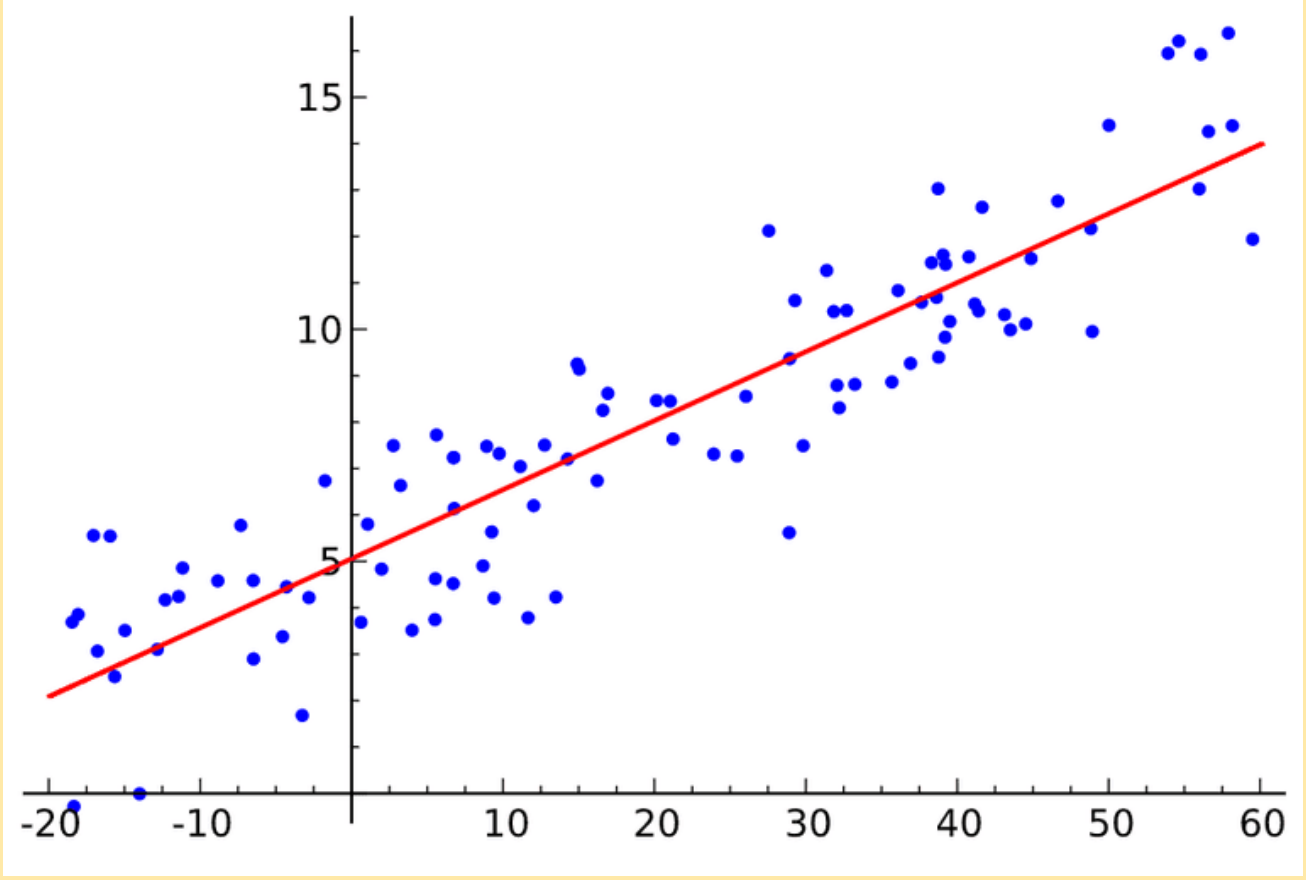
Uses the simple straight line equation

$$y = mx + c$$

y => prediction
x => input data
m => weight
c => Bias

Example:

Sales = (Weight x Radio) + Bias



Multivariable Regression:

Uses a complex linear equation

f(x, y, z) = (w1 * *x) + (w2* * y) + (w3 * z)

x, y, z = input data
w = weights

Example:

Sales = (w1 * Radio) + (w2 * TV) + (w3 * News)

When predicting a continuous value like the price of a house or a stock price we can't use an accuracy metric, so we have to use what is known as the mean squared error (MSE). The MSE ias the average squared difference between the actual and predicted values. We want to minimise this to improve our accuracy.

Decision Trees

A **Decision Tree** is a structure that allows us to split the dataset into branches and then make simple decisions at each level. The point at the bottom of the tree where we make the prediction is called a **leaf.**

The algorithms used construct the rules of the tree based on the relationship between the input data and the target labels in the training data. We need algorithms that can construct the optimal tree based on our data. These algorithms make use the concept of entropy to achieve this.

Entropy is basically a measure of uncertainty. As we move from top of the tree towards the bottom, we go from complete uncertainty to complete certainty. So a good way to construct the decision tree would be to reduce the uncertainty at each level. In other words, we need to reduce the entropy with each passing level in the tree.

Sentiment Analysis

Sentiment Analysis is the process of determine whether a piece of text is positive, neutral or negative; this is done through the use of variables such as context, tone and emotion. Within machine learning sentiment analysis is being used by companies to explore customers attitudes towards the company, product and understanding customer satisfaction.

Raw text data cannot be fed into machine learning models, hence we turn test into a numerical representation using vectorisation.

- **Tokenisation**: When we convert the text into a list of words and assign each individual string an id value. In Keras the tokenisation works by giving the value 1 to the most common word, 2 to the second most common and so on; this is useful as we would want to ignore rare words.

Text is often passed through a model that contains an LSTM layer (Long Short-Term Memory) which is a type of RNN (see below), this is because the next words in a sentence depend on the words that come around it. You may also see a CNN (see below) layer incorporated which is used to simplify the words into smaller grouped blocks that can be processed easily.

CNN (Convolutional Neural Networks)

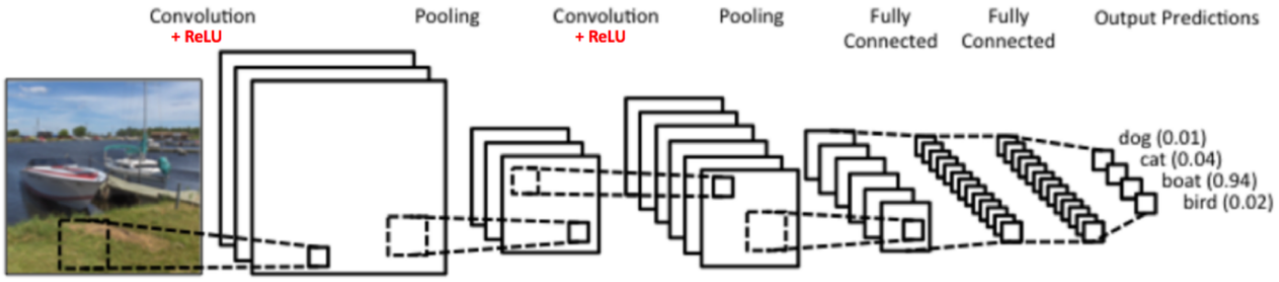
Convolutional Neural Networks are a subset of Neural Networks that have proven to be effective in areas of image recognition and classfication. Recently CNN's have been used within NLP tasks

such as sentence classification. CNN's are made up of input and output layers and beneath these we have multiple hidden layers, which are convolutional and subsampling layers commonly followed by a fully connected layer.

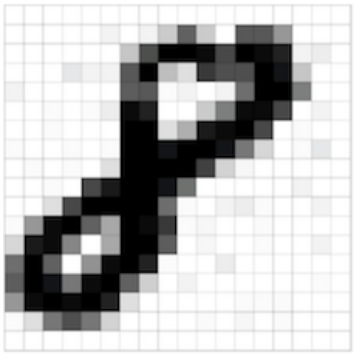
How does a CNN work?

There are four main parts to a CNN :

1. Convolution
2. Non-Linearity (ReLU)
3. Pooling or Sub-Sampling
4. Classification (Fully Connected Layer)

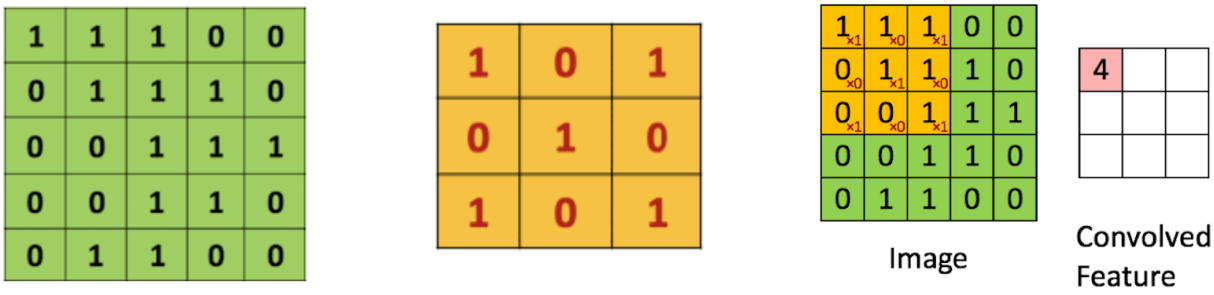


Before using a CNN we need to understand how images are composed. Images are represented as a matrix of pixel values between 0 and 255. A channel is a term used to refer to a component of an image; typical full colour images have 3 channels (RGB) so that is three 2D matrices stacked on top of each other. Grayscale images have only one 2D channel.



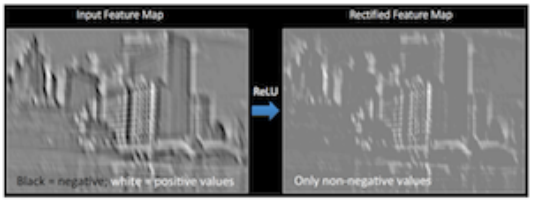
Step 1: Convolutional Step

The primary purpose of convolution is to extract features from the data passed in. Imagine we have the 5x5 image (green) and that it is a special case where the the pixels are only 1 or 0. We will take the 3x3 matrix and slide it over the original image by 1 pixel at a time (this is our stride) and multiply the elements and add them to get a final value for the feature map.



Step 2: Non-Linearity (ReLU)

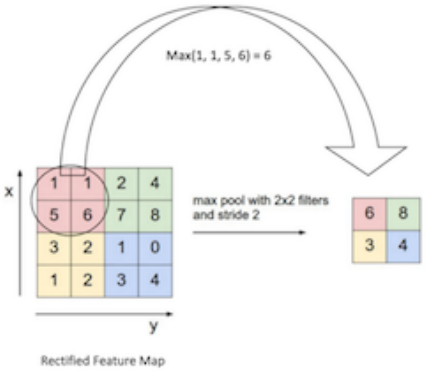
ReLU (Rectified Linear Unit) steps are normally introduced after convolution operations in CNN's. ReLU is applied per pixel and replaces any negative values with a 0 value; this is due to wanting to introduce non-linearity. The purpose of introducing non-linearity is due to the fact that most real world data is not linear. There are other Non-Linearity function that you can apply such as tanh or sigmoid but it is proven that ReLU has good performance



Step 3: Pooling Stage

The pooling stage of the process is called spatial pooling and reduces the dimensionality of the feature maps while retaining the important information. There are different types of pooling: Max, Average, Sum etc. and pooling can also be referred to as subsampling or downsampling. We perform pooling to redact the spatial size of the input, this means pooling makes the input representations smaller, reduces network computations which reduces overfitting and makes the network invariant to small transformations or distortions.

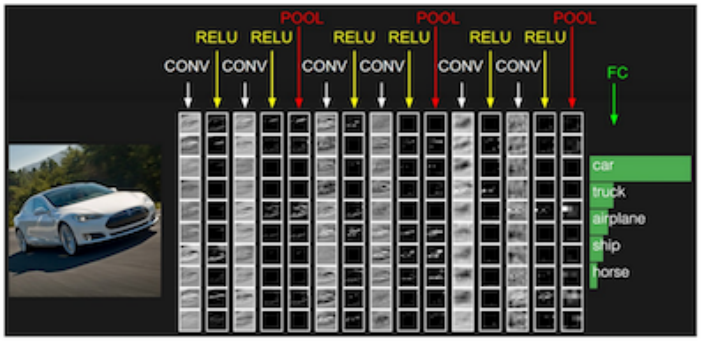
Example: the image below shows one feature map where Max Pooling is being applied.



The process of Pooling gets applied to every feature map that is outputted in the convolution stage; if we have 3 feature maps after convolution we will have 3 layers after pooling.

Step 4: Fully Connected Layer

The Fully Connected layer implies that every neutron in the previous layer is connected to every neutron in the next layer. The most popular activation function in the output layer is Softmax however other classifiers such as SVM can be used. The output from the previous convolutional and pooling layers represent high-level features for classifying the image based on the training data set. The sum of the output probabilities from a fully connected layer is equal to 1 which is ensured by the Softmax function.



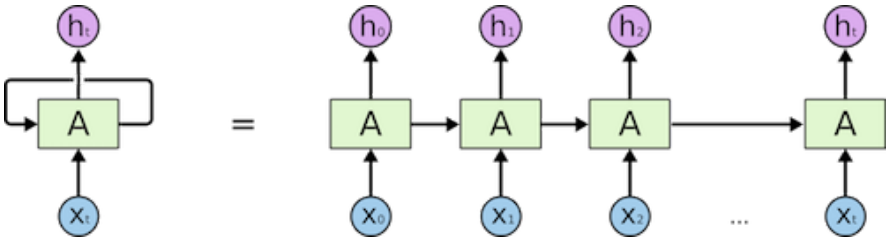
Explain visually CNNs (Image Kernels)

RNNs (Recurrent Neural Networks)

A RNN is a function that applies the same transformation to every element of a sequence, the transformation is known as the RNN cell or step. The output of the RNN layer is the output of the

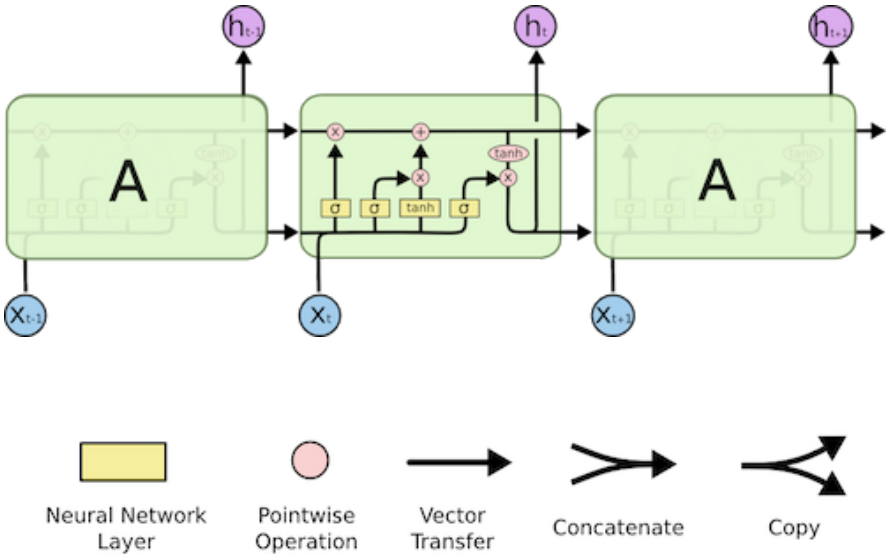
RNN cell applied on every element in the sequence; for example in the case of text the elements are usually successive words or characters. A RNN cell also holds an internal memory that summarises the history of the sequence it has seen so far.

RNN's struggle to learn with long sequences and to solve this the RNN cell is often replaced with a gated recurrent unit (GRU) or long short-term memory cell (LSTM).



LSTM

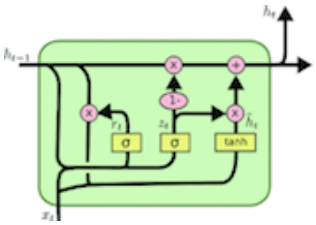
Long Short Term Memory is an RNN that is trained using Back-propagation through time and instead of neurons a LSTM network has memory blocks that are connected through the layers. The memory blocks are made up several gates that manage the block's state and output as well as a memory for recent sequences. Traditional neural networks usually throw away what they've learned previously and start over again. Recurrent Nueral Networks (RNN) are different, what they learn persists through each layer. A typical RNN can struggle to identify and learn the long term dependecies of the data. This is where a LSTM comes in as it is capable of learning long term dependencies.



More detailed explanation on LSTMs

GRU

A GRU can be considered a variation on a LSTM and often provide the same performance. They have been shown to exhibit better performance on smaller datasets.



Python/Jupyter Notebook Tips

TAB : shows function list
SHIFT-TAB (x1) : function parameters
SHIFT-TAB (x2) OR ? : Documentation
?? : Gets the source code for the function

Python Modules

Numpy:
Numpy uses vectorisation/vectorization to perform faster calculations when compared to using explicit for loops to complete the same task.

Install using pip:

```
pip install numpy
```

```
python -m pip install numpy
```

Install using anaconda:

```
conda install numpy
```

Pandas:
Pandas is an open-source data analysis and manipulation tool that provides flexible data-structures and is designed to work with both labeled and
unlabelled data.

Install using pip:

```
pip install pandas
```

```
python -m pip install pandas
```

Install using anaconda:

```
conda install pandas
```

Tensorflow:
Tensorflow is an open-source low level machine learning library for python developed by google.

Install using pip:

```
pip install tensorflow
```

```
python -m pip install tensorflow
```

Install using anaconda:

```
conda install tensorflow
```

Keras:
Keras is a high level API written in Python which can be run on top of Tensoflow or Theano. The standar library is developed to run off a CPU but there is a keras package which can be downloaded to run off Nvidia CUDA GPUs for much faster training.

Install using pip:

```
pip install keras
```

```
python -m pip install keras
```

Install using anaconda:

```
conda install keras
```

Scikit-learn:
Scikit-learn provides a fast and simple machine learning library made accessible to both new comers and experts.

Install using pip:

```
python -m pip install scikit-learn
```

Install using anaconda:

```
conda install scikit-learn
```

Matplotlib:
Matplotlib is a 2D plotting library which produces high quality figures which can be used in Python scripts, IPython shells and jupyter notebooks.

Install using pip:

```
pip install matplotlib
```

```
python -m pip install matplotlib
```

Install using anaconda:

```
conda install matplotlib
```

Scipy:
Scipy is open-source Python software for mathematics, science and engineering.

Install using pip:

```
pip install scipy
```

```
python -m pip install scipy
```

†

Install using anaconda:

```
conda install scipy
```

Pillow:
Pillow is a fork of an old image processing library called PIL and is the library that is recommended that you use this library over PIL as it is more frequently updated.

Install using pip:

```
pip install Pillow
```

```
python -m pip install Pillow
```

Install using anaconda:

```
conda install pillow
```

OpenCV:
Opencv is an open source computer vision and machine learning software library.

Install using pip:

```
pip install opencv-python
```

```
python -m pip install opencv-python
```

Install using anaconda:

```
conda install opencv
```

Tqdm:
Tqdm is a useful library that visualises the progress of any loops you are using within your program.

Install using pip:

```
pip install tqdm
```

```
python -m pip install tqdm
```

Install using anconda:

```
conda install tqdm
```

Flask:
The flask library is a web framework written in Python.

Install using pip:

```
pip install Flask
```

```
python -m pip install Flask
```

Install using anaconda:

```
conda install flask
```

Node Modules

Node.js:
Node.js is an open-source JavaScript runtime environment that runs JavaScript outside a browser.

Install :

- You will need to have a version of home-brew installed in order to install through terminal.

```
brew install node
```

Install using the GUI:
https://nodejs.org/en/download/

Restify:
Restify is a Node.js module, which is a web-service framework optimised for building semantically correct RESTful web-services.

Install using npm:

```
npm install restify
```

Express:
Express is a minimal and flexible Node.js web application framework that provides features for web and mobile applications.

```
npm install express
```

Axios:

Axios is a promise-based HTTP client that works both in the browser and in a node.js environment.

```
npm install axios
```

Request:
Request is a Node.js package for making HTTP requests.

```
npm install request
```

FS:
FS is a Node.js module that provides an API for interacting with the file system.

```
npm install fs
```

Express-fileupload:
Express-fileupload is an express middleware for uploading files.

```
npm install express-fileupload
```

BotBuilder:
Bot Builder SDK for Node.js is a powerful, easy-to-use framework that provides a way for Node.js developers to write bots.

```
npm install -- save botbuilder
```

Bluebird:
Bluebird is a JavaScript library for promises.

```
npm install --save bluebird
```

Learning Resources:

Website Links	Description
https://developers.google.com/machine-learning/crash-course/	
https://ai.google/education/	
https://selfdrivingcars.mit.edu/	Videos on

	Deep Learning.
https://software.intel.com/en-us/ai-academy/students/kits/machine-learning-501	
https://elitedatascience.com/keras-tutorial-deep-learning-in-python	
https://www.kaggle.com/learn/machine-learning	
https://tensorflowkorea.files.wordpress.com/2016/09/bay-area-deep-learning-school-presentation.pdf	Good slides detailing convolutional networks explained by andrej kaparthy.
https://medium.com/@14prakash/transfer-learning-using-keras-d804b2e04ef8	Transfer learning article
http://www.r2d3.us/visual-intro-to-machine-learning-part-1/	Good visual description to ML
https://jovianlin.io/keras-models-sequential-vs-functional/	A decription of sequential vs functional models
http://www.fast.ai/	A great source to learn about machine learning and deep learning.
https://cs.nyu.edu/~fergus/papers/zeilerECCV2014.pdf	A detailed paper which breaks down CNNs and describes how it learns the features of different images.

https://www.coursera.org/specializations/deep-learning	A good course which breaks down the maths behind the theory for deep learning.
https://github.com/tensorflow/lucid	A good tool for use in research allowing you to visualise a neural network.
https://medium.com/tensorflow/neural-style-transfer-creating-art-with-deep-learning-using-tf-keras-and-eager-execution-7d541ac31398	Good medium article with step by step code for learning about Neural style transfer.
http://www.fast.ai/2018/08/10/fastai-diu-imagenet/	A fast.ai article detailing how they beat google's ml code for the imagenet data.
http://deeplearning.net/software/theano_versions/dev/tutorial/conv_arithmetic.html	Detailed description of convolutions in a convolutional layer and how padding

	can change the output.
https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6	A breakdown of the different activation functions.
http://ml-cheatsheet.readthedocs.io/en/latest/index.html	ML cheatsheet.
https://hackernoon.com/gradient-descent-aynk-7cbe95a778da	Breakdown of gradient descent.