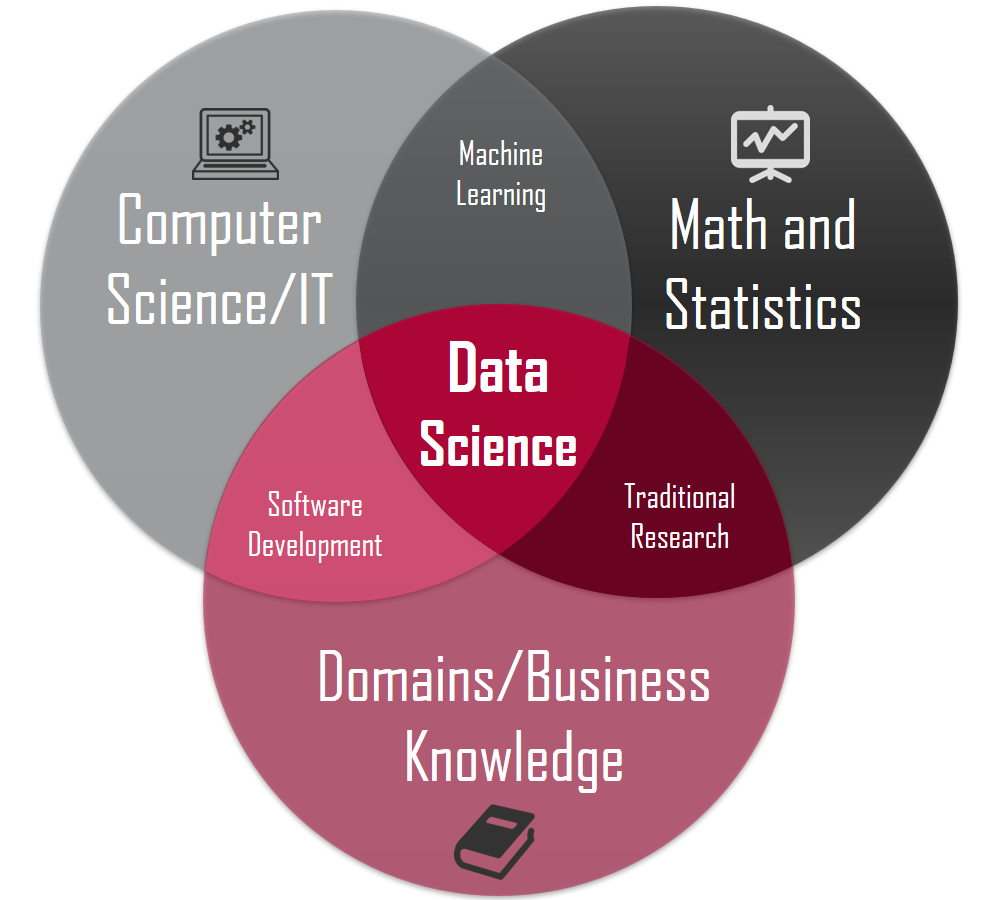
MACHINE LEARNING

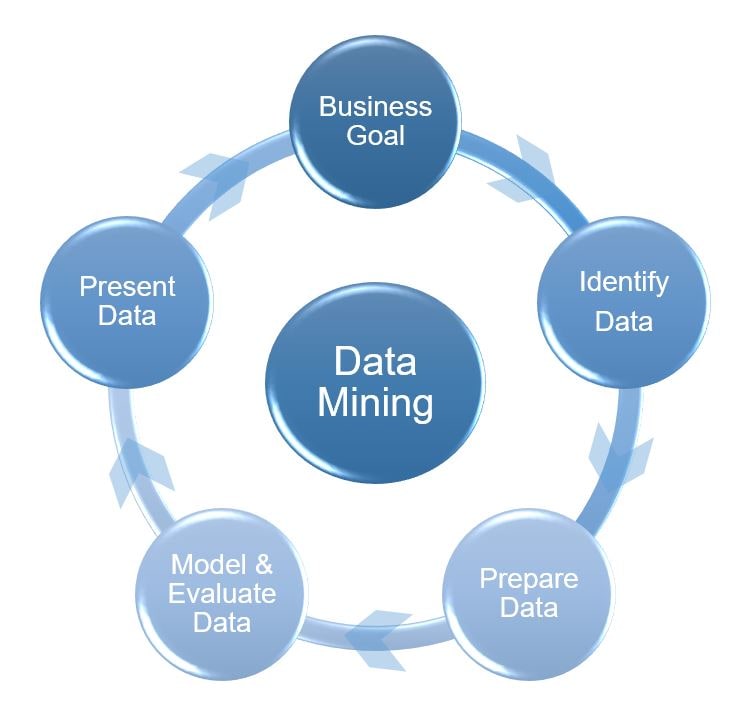
Introduction:

1. **What is data science?**

Data science is an inter-disciplinary field that uses scientific methods, processes, algorithms and systems to extract knowledge and insights from many structural and unstructured data. Data science is related to data mining, deep learning and big data.



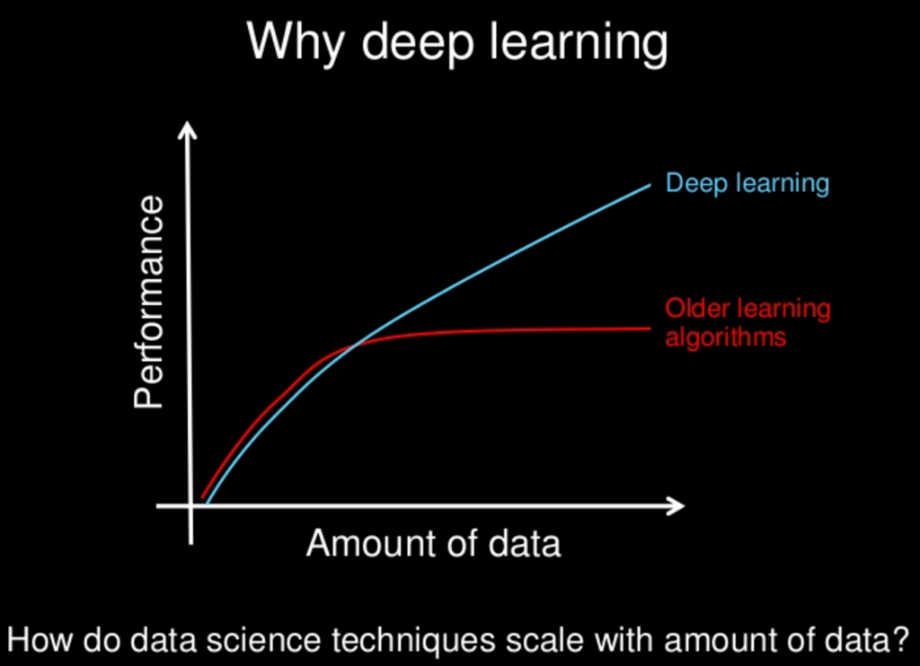
1. **What is data mining, deep learning and big data?**

**Data mining:**

Data mining is the process of finding anomalies, patterns and correlations within large data sets to predict outcomes. Using a broad range of techniques, you can use this information to increase revenues, cut costs, improve customer relationships, reduce risks and more.

**Deep learning:**

**Deep Learning** is a subfield of machine learning concerned with algorithms inspired by the structure and function of the brain called **artificial neural networks.**Deep learning is a class of [machine learning](https://en.wikipedia.org/wiki/Machine_learning) [algorithms](https://en.wikipedia.org/wiki/Algorithm) that uses multiple layers to progressively extract higher level features from the raw input. For example, in [image processing](https://en.wikipedia.org/wiki/Image_processing), lower layers may identify edges, while higher layers may identify the concepts relevant to a human such as digits or letters or faces.



**BIG DATA:**

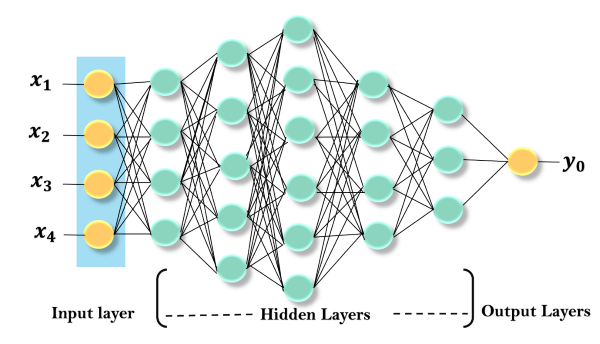
[**Big data** analytics is the process](https://www.educba.com/what-is-big-data-analytics/) of collecting and analyzing the large volume of data sets (called Big Data) to discover useful hidden patterns and other information like customer choices, market trends that can help organizations make more informed and customer-oriented business decisions. Big data is a term that describes the data characterized by 3Vs: **the extreme volume of data, the wide variety of data types and the velocity at which the data must be processed**. Big data can be analyzed for insights that lead to better decisions and [strategic](https://www.educba.com/course/strategic-management/) business moves.

1. **Difference between big data and machine learning?**

[Big data analytics](https://www.educba.com/big-data-analytics-techniques/) as the name suggest is the analysis of big data by discovering hidden patterns or extracting information from it. Machine learning, in simple terms, is teaching a machine how to respond to unknown inputs and give desirable outputs by [using various machine learning models](https://www.educba.com/machine-learning-models/). Big data has large volumes of input and often from varied sources whereas machine learning operates on a smaller scale data and often related to each other. Data mining on the hand is the process used to make a decision while big data is used in dashboards and predictive measures. Data mining is done on big data.

1. **How does deep learning work?**

* Deep Learning Algorithms work on deep neural networks, so it is called deep learning. These deep neural networks are made of multiple layers.
* The first layer is called an Input layer, the last layer is called an output layer, and all layers between these two layers are called hidden layers.
* In the deep neural network, there are multiple hidden layers, and each layer is composed of neurons. These neurons are connected in each layer.
* The input layer receives input data, and the neurons propagate the input signal to its above layers.
* The hidden layers perform mathematical operations on inputs, and the performed data forwarded to the output layer.
* The output layer returns the output to the user.



1. **What are the subdivisions of artificial Intelligence?**

1.Machine Learning

i. Deep Learning

2. Natural Language Processing

i. Text generation

ii.Question Answering

iii.context extraction

iv.classification

v. machine translation

3.Vision

i.Image recognition

ii.machine vision

4.Robotics

5.Planning

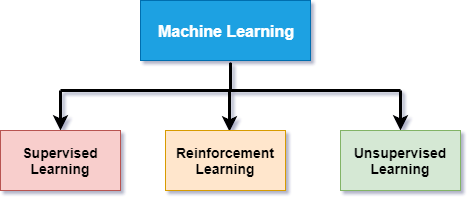
6.Expert systems

1. **Explain the sub-categories of machine learning in detail**

Machine Learning

Machine learning is a part of AI which provides intelligence to machines with the ability to automatically learn with experiences without being explicitly programmed.

Types of Machine Learning



Machine learning can be subdivided in to the main three types:

* **Supervised learning:**  
  Supervised learning is a type of machine learning in which machine learn from known datasets (set of training examples), and then predict the output. A supervised learning agent needs to find out the function that matches a given sample set.  
  Supervised learning further can be classified into two categories of algorithms:
  1. **Classifications**
  2. **Regression**
* **Reinforcement learning:**  
  Reinforcement learning is a type of learning in which an AI agent is trained by giving some commands, and on each action, an agent gets a reward as a feedback. Using these feedbacks, agent improves its performance.  
  Reward feedback can be positive or negative which means on each good action, agent receives a positive reward while for wrong action, it gets a negative reward.  
  Reinforcement learning is of two types:

1. **Positive Reinforcement learning**
2. **Negative Reinforcement learning**

* **Unsupervised learning:**  
  Unsupervised learning is associated with learning without supervision or training. In unsupervised learning, the algorithms are trained with data which is neither labeled nor classified. In unsupervised learning, the agent needs to learn from patterns without corresponding output values.  
  Unsupervised learning can be classified into two categories of algorithms:

1. **Clustering**
2. **Association**

Natural Language processing

Natural language processing is a subfield of computer science and artificial intelligence. NLP enables a computer system to understand and process human language such as English.

NLP plays an important role in AI as without NLP, AI agent cannot work on human instructions, but with the help of NLP, we can instruct an AI system on our language. Today we are all around AI, and as well as NLP, we can easily ask Siri, Google or Cortana to help us in our language.

Natural language processing application enables a user to communicate with the system in their own words directly.

The Input and output of NLP applications can be in two forms:

* **Speech**
* **Text**

Deep Learning

Deep learning is a subset of machine learning which provides the ability to machine to perform human-like tasks without human involvement. It provides the ability to an AI agent to mimic the human brain. Deep learning can use both supervised and unsupervised learning to train an AI agent.

Expert Systems

* An expert system is an application of artificial intelligence. In artificial intelligence, expert systems are the computer programs that rely on obtaining the knowledge of human experts and programming that knowledge into a system.
* Expert systems emulate the decision-making ability of human experts. These systems are designed to solve the complex problem through bodies of knowledge rather than conventional procedural code.
* One of the examples of an expert system is a Suggestion for the spelling error while typing in the Google search box.
* Following are some characteristics of expert systems:
  + High performance
  + Reliable
  + Highly responsive
  + Understandable

Robotics

* Robotics is a branch of artificial intelligence and engineering which is used for designing and manufacturing of robots.
* Robots are the programmed machines which can perform a series of actions automatically or semi-automatically.
* AI can be applied to robots to make intelligent robots which can perform the task with their intelligence. AI algorithms are necessary to allow a robot to perform more complex tasks.
* Nowadays, AI and machine learning are being applied on robots to manufacture intelligent robots which can also interact socially like humans. One of the best examples of AI in robotics is Sophia robot.

Machine Vision

* Machine vision is an application of computer vision which enables a machine to recognize the object.
* Machine vision captures and analyses visual information using one or more video cameras, analog-to-digital conversations, and digital signal processing.
* Machine vision systems are programmed to perform narrowly defined tasks such as counting objects, reading the serial number, etc.

Speech Recognition:

Speech recognition is a technology which enables a machine to understand the spoken language and translate into a machine-readable format. It can also be said as automatic Speech recognition and computer speech recognition. **It is a way to talk with a computer, and on the basis of that command, a computer can perform a specific task**.

There is some speech recognition software which has a limited vocabulary of words and phrase. This software requires unambiguous spoken language to understand and perform specific task. Today's there are various software or devices which contains speech recognition technology such as Cortana, Google virtual assistant, Apple Siri, etc.

There are two types of speech recognition

1. **Speaker Dependent**
2. **Speaker Independent**
3. **What are the steps involved in machine learning?**
   1. Data collection
   2. Data preparation
   3. Choosing a model
   4. Training the model
   5. Evaluating the model
   6. Hyper parameter tuning
   7. Making predictions
4. **What are the steps involved in data preparation?**

**Steps Involved in Data Preprocessing:**

* 1. **Data Cleaning:**  
     The data can have many irrelevant and missing parts. To handle this part, data cleaning is done. It involves handling of missing data, noisy data etc.
     1. **Missing Data:**  
        This situation arises when some data is missing in the data. It can be handled in various ways.  
        Some of them are:
        1. **Remove the rows with null values**
        2. **Fill the Missing values:**  
           Missing values can be filled with mean, median, or by some other means.
     2. **Noisy Data:**  
        Noisy data is a meaningless data that can’t be interpreted by machines. It can be generated due to faulty data collection, data entry errors etc. It can be handled in following ways:
  2. **Data Transformation:**  
     This step is taken in order to transform the data in appropriate forms suitable for mining process. This involves following ways:
     + 1. **Normalization:**  
          It is done in order to scale the data values in a specified range (-1.0 to 1.0 or 0.0 to 1.0)
       2. **Attribute Selection:**  
          In this strategy, new attributes are constructed from the given set of attributes to help the mining process.
       3. **Discretization:**  
          This is done to replace the raw values of numeric attribute by interval levels or conceptual levels.
       4. **Concept Hierarchy Generation:**  
          Here attributes are converted from level to higher level in hierarchy. For Example-The attribute “city” can be converted to “country”.
  3. **Data Reduction:**  
     Since data mining is a technique that is used to handle huge amount of data. While working with huge volume of data, analysis became harder in such cases. In order to get rid of this, we use data reduction technique. It aims to increase the storage efficiency and reduce data storage and analysis costs.

The various steps to data reduction are:

* + - 1. **Data Cube Aggregation:**  
         Aggregation operation is applied to data for the construction of the data cube.
      2. **Attribute Subset Selection:**  
         The highly relevant attributes should be used, rest all can be discarded. For performing attribute selection, one can use level of significance and p- value of the attribute. The attribute having p-value greater than significance level can be discarded.
      3. **Numerosity Reduction:**  
         This enable to store the model of data instead of whole data, for example: Regression Models.
      4. **Dimensionality Reduction:**  
         This reduce the size of data by encoding mechanisms. It can be lossy or lossless. If after reconstruction from compressed data, original data can be retrieved, such reduction are called lossless reduction else it is called lossy reduction.

The two effective methods of dimensionality reduction are:

* + - * 1. Wavelet transforms
        2. PCA (Principal Component Analysis).

To sum up, data cleaning involves

* 1. remove duplicates
  2. correct errors
  3. deal with missing values
  4. normalization
  5. data type conversions

The procedure:

* + 1. Import necessary library
    2. Import data file
    3. Handling of missing values
    4. Convert categorical data into numeric
    5. Splitting the dataset
    6. Featured scaling

1. **What is feature scaling? Why is it necessary?**

Machine learning algorithm just sees number — if there is a vast difference in the range say few ranging in thousands and few ranging in the tens, and it makes the underlying assumption that higher ranging numbers have superiority of some sort. So these more significant number starts playing a more decisive role while training the model.

The machine learning algorithm works on numbers and does not know what that number represents. A weight of 10 grams and a price of 10 dollars represents completely two different things — which is a no brainer for humans, but for a model as a feature, it treats both as same.

**Machine learning algorithms like**[**linear regression**](https://www.analyticsvidhya.com/blog/2017/05/neural-network-from-scratch-in-python-and-r/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization)**,**[**logistic regression**](https://www.analyticsvidhya.com/blog/2017/05/neural-network-from-scratch-in-python-and-r/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization)**,**[**neural network**](https://www.analyticsvidhya.com/blog/2017/05/neural-network-from-scratch-in-python-and-r/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization)**, etc. that use gradient descent as an optimization technique require data to be scaled.** Take a look at the formula for gradient descent below:

The presence of feature value X in the formula will affect the step size of the gradient descent. The difference in ranges of features will cause different step sizes for each feature. To ensure that the gradient descent moves smoothly towards the minima and that the steps for gradient descent are updated at the same rate for all the features, we scale the data before feeding it to the model.

*Having features on a similar scale can help the gradient descent converge more quickly towards the minima.*

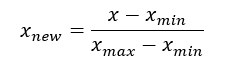
Distance algorithms like [KNN](https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization), [K-means](https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-k-means-clustering/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization), and [SVM](https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization) are most affected by the range of features. This is because behind the scenes **they are using distances between data points to determine their similarity.**

[Tree-based algorithms](https://www.analyticsvidhya.com/blog/2016/04/tree-based-algorithms-complete-tutorial-scratch-in-python/?utm_source=blog&utm_medium=feature-scaling-machine-learning-normalization-standardization), on the other hand, are fairly insensitive to the scale of the features. Think about it, a decision tree is only splitting a node based on a single feature. The decision tree splits a node on a feature that increases the homogeneity of the node. This split on a feature is not influenced by other features.

1. **What are the different methods of feature scaling? Write along with their formulas. Also write the pros and cons of them.**

1) Min Max Scaler  
2) Standard Scaler  
3) Max Abs Scaler  
4) Robust Scaler  
5) Quantile Transformer Scaler  
6) Power Transformer Scaler  
7) Unit Vector Scaler

8) Normalizer

* + - 1. **Min Max Scalar (normalization)**
* Transform features by scaling each feature to a given range. This estimator scales and translates each feature individually such that it is in the given range on the training set, e.g., between zero and one. This Scaler shrinks the data within the range of -1 to 1 if there are negative values.
* 
* Transform features by scaling each feature to a given range. This estimator scales and translates each feature individually such that it is in the given range on the training set, e.g., between zero and one. This Scaler shrinks the data within the range of -1 to 1 if there are negative values.
* This Scaler responds well if the standard deviation is small and when a distribution is **not Gaussian**. This Scaler is **sensitive to outliers**.

2) Standard scalar

**Standardization is another scaling technique where the values are centered around the mean with a unit standard deviation. This means that the mean of the attribute becomes zero and the resultant distribution has a unit standard deviation.**

Here’s the formula for standardization:

[Standardization equation](https://cdn.analyticsvidhya.com/wp-content/uploads/2020/03/Stand_eq.gif)

Centering and scaling happen independently on each feature by computing the relevant statistics on the samples in the training set. If data is not normally distributed, this is not the best Scaler to use.

## 3) Max Abs Scaler

The MaxAbsScaler works very similarly to the MinMaxScaler but automatically scales the data to a [-1,1] range based on the **absolute maximum**. This scaler is meant for**data that is already centered at zero or sparse data**. It does not shift/center the data, and thus does not destroy any sparsity.

x\_scaled = x / max(abs(x))

Let’s once again tackle feature 3 by transforming it using the MaxAbsScaler and compare the output with the original data.

On positive-only data, this Scaler behaves similarly to Min Max Scaler and, therefore, also suffers from the presence of significant**outliers**.

## 4) Robust Scaler

As the name suggests, this Scaler is **robust** to outliers. If our data contains many **outliers**, scaling using the mean and standard deviation of the data won’t work well.

This Scaler removes the median and scales the data according to the quantile range (defaults to IQR: Interquartile Range). The IQR is the range between the 1st quartile (25th quantile) and the 3rd quartile (75th quantile). The centering and scaling statistics of this Scaler are based on percentiles and are therefore not influenced by a few numbers of huge marginal outliers. Note that the outliers themselves are still present in the transformed data.

5) Quantile Transformer Scaler

This method transforms the features to follow a uniform or a normal distribution. Therefore, for a given feature, this transformation tends to spread out the most frequent values. It also reduces the impact of (marginal) outliers: this is therefore a robust preprocessing scheme.

The transformation is applied on each feature independently. First an estimate of the cumulative distribution function of a feature is used to map the original values to a uniform distribution. The obtained values are then mapped to the desired output distribution using the associated quantile function. Features values of new/unseen data that fall below or above the fitted range will be mapped to the bounds of the output distribution. Note that this transform is non-linear. It may distort linear correlations between variables measured at the same scale but renders variables measured at different scales more directly comparable.

## 6) Power Transformer Scaler

The power transformer is a family of parametric, monotonic transformations that are applied to **make data more Gaussian-like**.

The power transform finds the optimal scaling factor in stabilizing variance and minimizing skewness through maximum likelihood estimation. Currently, Sklearn implementation of PowerTransformer supports the Box-Cox transform and the Yeo-Johnson transform. The optimal parameter for stabilizing variance and minimizing skewness is estimated through maximum likelihood. Box-Cox requires input data to be strictly positive, while Yeo-Johnson supports both positive or negative data.

\

## 7) Unit Vector Scaler

https://miro.medium.com/max/165/0*BNgQzjr02S0lRolV.png

Scaling is done considering the whole feature vector to be of unit length. This usually means dividing each component by the Euclidean length of the vector (L2 Norm). In some applications (e.g., histogram features), it can be more practical to use the L1 norm of the feature vector.

Like Min-Max Scaling, the Unit Vector technique produces values of range [0,1]. When dealing with features with hard boundaries, this is quite useful. For example, when dealing with image data, the colors can range from only 0 to 255.

### 8) Normalizer

The normalizer scales each value by dividing each value by its magnitude in nn-dimensional space for nn number of features.

Say your features were x, y and z Cartesian co-ordinates your scaled value for x would be:

xix2i+y2i+z2i−−−−−−−−−−√xixi2+yi2+zi2

Each point is now within 1 unit of the origin on this Cartesian co-ordinate system.

1. How to detect if we need to remove null values or not?

It depends on randomness of data. Try to analyze the situation.

* If more or less equal distribution of nan in categorical values which are of high importance, then removing them might decrease efficiency
* If missing values are random, then see the number of missing rows. If the ration is like below 20% of the total number, deleting them is fine!

1. Write the code for various scaling techniques
   * + 1. Min-Max scalar (Normalization)

from sklearn.preprocessing import MinMaxScaler  
scaler = MinMaxScaler()

* + - 1. Standard scalar

from sklearn.preprocessing import StandardScaler  
scaler = StandardScaler()

* + - 1. Max Abs Scalar

from sklearn.preprocessing import MaxAbsScaler  
scaler = MaxAbsScaler()

* + - 1. Robust scalar

from sklearn.preprocessing import RobustScaler  
scaler = RobustScaler()

* + - 1. Quantile Transformer Scalar

from sklearn.preprocessing import QuantileTransformer  
scaler = QuantileTransformer()

* + - 1. Power Transformer scalar

from sklearn.preprocessing import PowerTransformer  
scaler = PowerTransformer(method='yeo-johnson')

* + - 1. Normalizer

scaler = preprocessing.Normalizer()

1. Write the methods to handle missing values
   * 1. Removies the rows with missing values
     2. Filling it with mean, median, mode or constant

Method: SimpleImputer()

from sklearn.impute import SimpleImputer

imp = SimpleImputer(missing\_values=np.nan, strategy='mean')

* + 1. Using k clustering
    2. Using interpolation( mainly used in deep neural networks)

1. **When should we remove outliers? What are the methods to detect outliers?**

If the outlier in question is:

* A measurement error or data entry error, correct the error if possible. If you can’t fix it, remove that observation because you know it’s incorrect.
* Not a part of the population you are studying (i.e., unusual properties or conditions), you can legitimately remove the outlier.
* A natural part of the population you are studying, you should not remove it.

Methods of outlier detection:

Visual:

Box plot

Scatter plot

Predictive:

IQR

Z-score ( greater than threshold which is usually 3)

1. What is hypothesis testing? Why do we need it? What are the tests available?

The process of hypothesis testing is to draw inferences or some conclusion about the overall population or data by conducting some statistical tests on a sample.

For drawing some inferences, we have to make some assumptions that lead to two terms that are used in the hypothesis testing.

Different Inferential Statistics methods:

1. Z-statistics
2. Hypothesis testing
3. T statistics
4. Normal distribution
5. Central Limit theorem
6. Confidence Interval
7. ANOVA
8. R2
9. Regression

What are the main components of z-test, or t-test?

* Null hypothesis: It is regarding the assumption that there is no anomaly pattern or believing according to the assumption made.
* Alternate hypothesis: Contrary to the null hypothesis, it shows that observation is the result of real effect.

### P value

### It can also be said as evidence or level of significance for the null hypothesis or in machine learning algorithms. It’s the significance of the predictors towards the target.

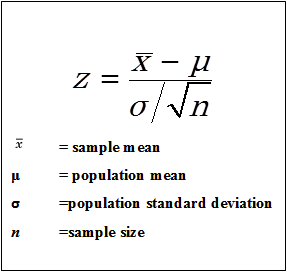
Generally, we select the level of significance by 5 %, but it is also a topic of discussion for some cases. If you have a strong prior knowledge about your data functionality, you can decide the level of significance.

1. What is z-statistics and t-statistics? Where will you use it?

**Z-Test:**

**Example case:** A company says it produces each shipment of 100Kg with a standard deviation of 1. The buyer has doubts in it. He randomly takes 30 shipment goods and weighs. The mean of the sample id 95. Is his claim true or false?

Here we use z-test.

* Z-statistics use z-score
* Z score tells how many standard deviations above or below a data point is from the population mean
* 

# **statsmodels.stats.weightstats.ztest**

**Parameters**

**x1**[array\_like](https://docs.scipy.org/doc/numpy/glossary.html#term-array-like), 1-D or 2-D

first of the two independent samples

**x2**[array\_like](https://docs.scipy.org/doc/numpy/glossary.html#term-array-like), 1-D or 2-D

second of the two independent samples

**value** float

In the one sample case, value is the mean of x1 under the Null hypothesis. In the two sample case, value is the difference between mean of x1 and mean of x2 under the Null hypothesis. The test statistic is x1\_mean - x2\_mean - value.

**alternativi**st

The alternative hypothesis, H1, has to be one of the following

‘two-sided’: H1: difference in means not equal to value (default) ‘larger’: H1: difference in means larger than value ‘smaller’: H1: difference in means smaller than value

**Returns**

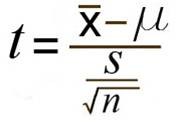
* **tstat**
* **pvalue**
* **T-score:**

we use t-tests when the population standard deviation is not known.

Example scanrio:

Average Intelligence measure is IQ 100. A team of scientists introduce a new medicine and whant to find out if it increases IQ or decreases IQ or has no effect. So they take 30 people who had taken the medication with a mean of 140 and a standard deviation of 20. What is the effect?

We use t-test when the population standard deviation is not known or when sample distribution size is less than 30



\_

X – Sample mean

* - Population mean

S- standard deviation of the sample

n- degree of freedom = no. of observations of sample – 1

# **statsmodels.stats.weightstats.ttest\_ind**

statsmodels.stats.weightstats.ttest\_ind(x1, x2, alternative='two-sided', usevar='pooled', weights=(None, None), value=0)

ttest independent sample

Convenience function that uses the classes and throws away the intermediate results, compared to scipy stats: drops axis option, adds alternative, usevar, and weights option.

**Parameters**

**x1**[array\_like](https://docs.scipy.org/doc/numpy/glossary.html#term-array-like), 1-D or 2-D

first of the two independent samples, see notes for 2-D case

**x2**[array\_like](https://docs.scipy.org/doc/numpy/glossary.html#term-array-like), 1-D or 2-D

second of the two independent samples, see notes for 2-D case

**alternative**[str](https://docs.python.org/3/library/stdtypes.html#str)

The alternative hypothesis, H1, has to be one of the following

* ‘two-sided’ (default): H1: difference in means not equal to value
* ‘larger’: H1: difference in means larger than value
* ‘smaller’: H1: difference in means smaller than value

**usevar**[str](https://docs.python.org/3/library/stdtypes.html#str), ‘pooled’ or ‘unequal’

If pooled, then the standard deviation of the samples is assumed to be the same. If unequal, then Welsh ttest with Satterthwait degrees of freedom is used

**weights**[tuple](https://docs.python.org/3/library/stdtypes.html#tuple) **of** [None](https://docs.python.org/3/library/constants.html#None) or **ndarrays**

Case weights for the two samples. For details on weights see DescrStatsW

**value** float

difference between the means under the Null hypothesis.

**Returns**

**tstat**[float](https://docs.python.org/3/library/functions.html#float)

test statistic

**pvalue**[float](https://docs.python.org/3/library/functions.html#float)

pvalue of the t-test

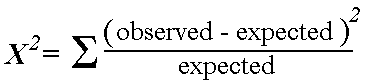
**df**[int](https://docs.python.org/3/library/functions.html#int) or [float](https://docs.python.org/3/library/functions.html#float)

degrees of freedom used in the t-test

If p-value is less than

1. What is chi-square test? Where is it used?

The **Chi Square**statistic is commonly used for testing relationships between categorical variables.  The null hypothesis of the Chi-Square test is that no relationship exists on the categorical variables in the population; they are independent.



Null hypothesis: No relation between the two variables

Alternate hypothesis: relationship exists

Create a cross tab between the two variables

Create the expected value for each cell and prepare a table

Expected value of a cell= (row total) \* (column total)/ (total no. of observations)

Then calculate chi square value

P value determination:

Calculate degree of freedom: (no of rows – 1) \*(no. of columns –1)

Find the value in chi square table

Less than p< significant – relationship exists

Greater than p> non-significant- relationship does not exist

1. What are the machine learning algorithms for regression?

**LOGISTIC REGRESSION:**

It is a linear model that establishes the relationship between a dependent variable y and one or more independent variables denoted by x (inputs)

**Hyper-Parameter:** Learning rate

**GOAL:** To draw a best fit line by determining the values of m and b in the given equation:

**y = mx+b** It uses Gradient Descent Algorithm

**step 1:** ­­­­­Take the derivative of the Loss function for each parameter in it

Note: In machine Learning lingo, the sum of the squared residuals is d type of Loss function

**step 2:** Pick random values for the parameters

**step 3:** Plug the parameter values into the derivatives (Gradient)

**step 4:** Calculate the step sizes

**step 5**: Calculate the New parameter

New parameter= Old parameter – step size

**End:** 1. Step size becomes very small

Pre-defined Maximum no. of steps is reached

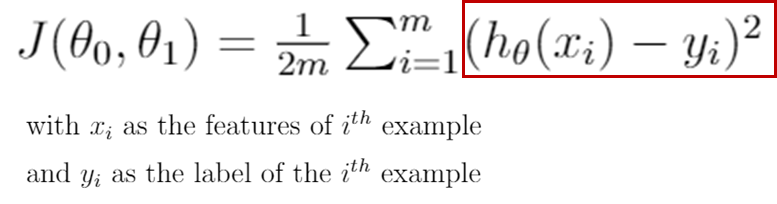
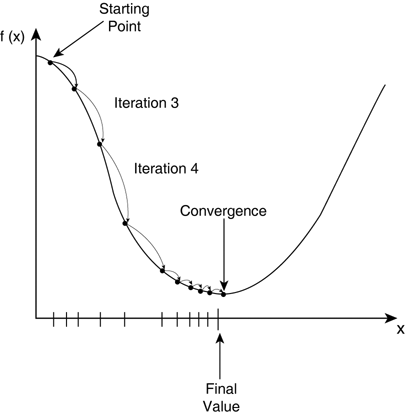
**Note:**

1. If the data set is large we use a method called Stochastic Gradient Descent that uses a randomly selected subset of the data at every step rather than the full data set
2. Learning rate is hit and trial and fulfilled through hyper parameter tuning

**Formulas:**

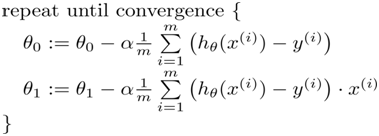
Total no. of observations

Cost function:



True value

Predicted value



Why is gradient descent very effective?

Because it takes big steps when far away from minimal point and small steps when close the minimal point

What is learning rate?

Size of each step is determined by the learning rate. If alpha is very large itmay fail to converge and when alpha is very small, it might take larger no. of steps

When to use linear regression?

Data should be linear in nature

There should not be a relationship between independent variables (if its there, it is called multi collinearity)

What is the way in which the learning rate changes from relatively large to relatively small

Schedule

1. **Multiple regression:**

Formula: y= m1x1 + m2x2 +……+ b

1. **Logistic Regression**:

**Conditions:**

Requires linearly seperable data. Classes are almost/ perfectly linearly seperable (plane/line/hyper-plane)

It is a classification algorithm, it is mainly used to predict nonary outcome [I/O, yes/no, True/False]

It predicts the probability of occurrence of an event by fitting data to logit function (the log of the ratio of the probabilities us called the logit function and forms the basis of logisitic regression.

The logistic regression tries to form a linear equation for the given value of x:

P(x)= ez / (1+ ez) -------------🡪 z= WTX + b (equation of the plane)

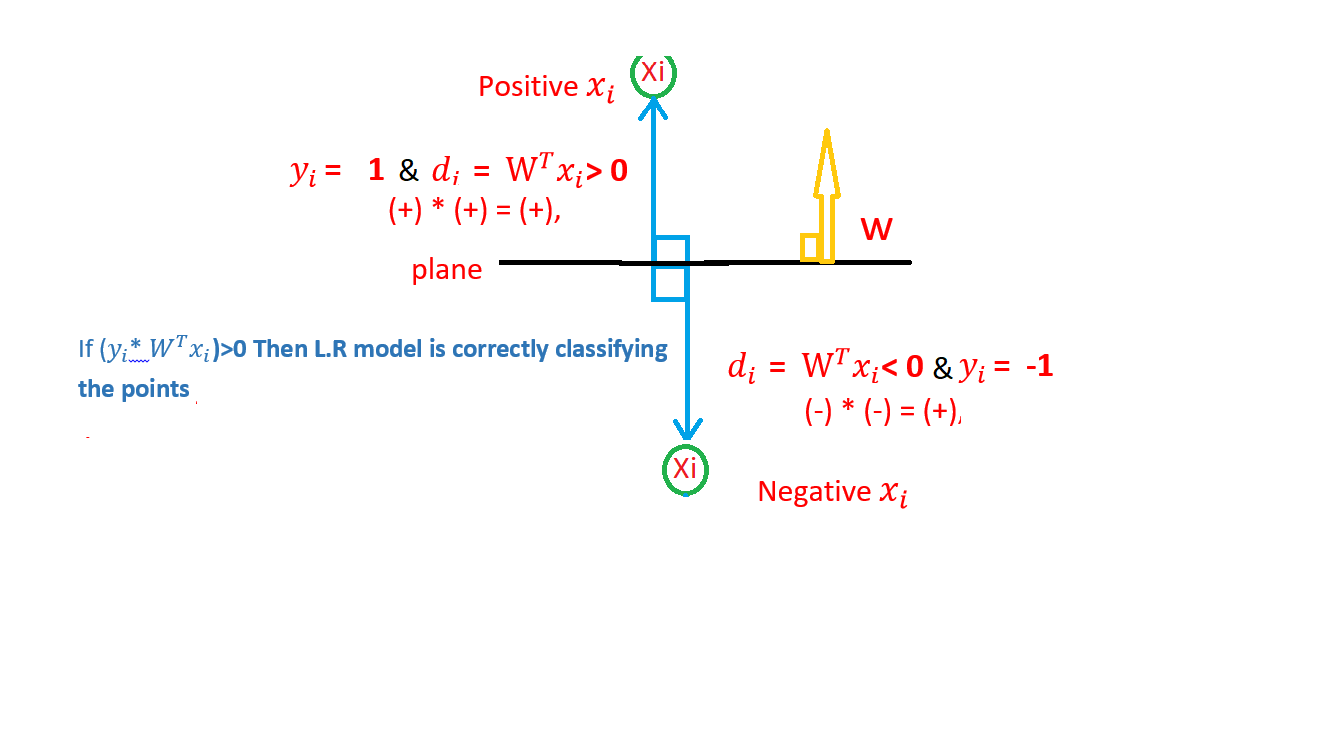
Here y is the probability of the value of x to fall in the class y=1 or class 1 type

P(x)<0.5 -----🡪 class -1

P(x)> 0.5------🡪class 1

We estimate the value of W and b by using maximum likelihood concept.

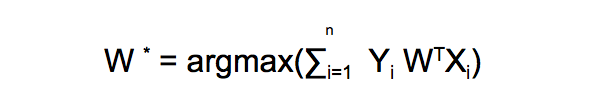
Say y can be 1 or -1 representing the classes.

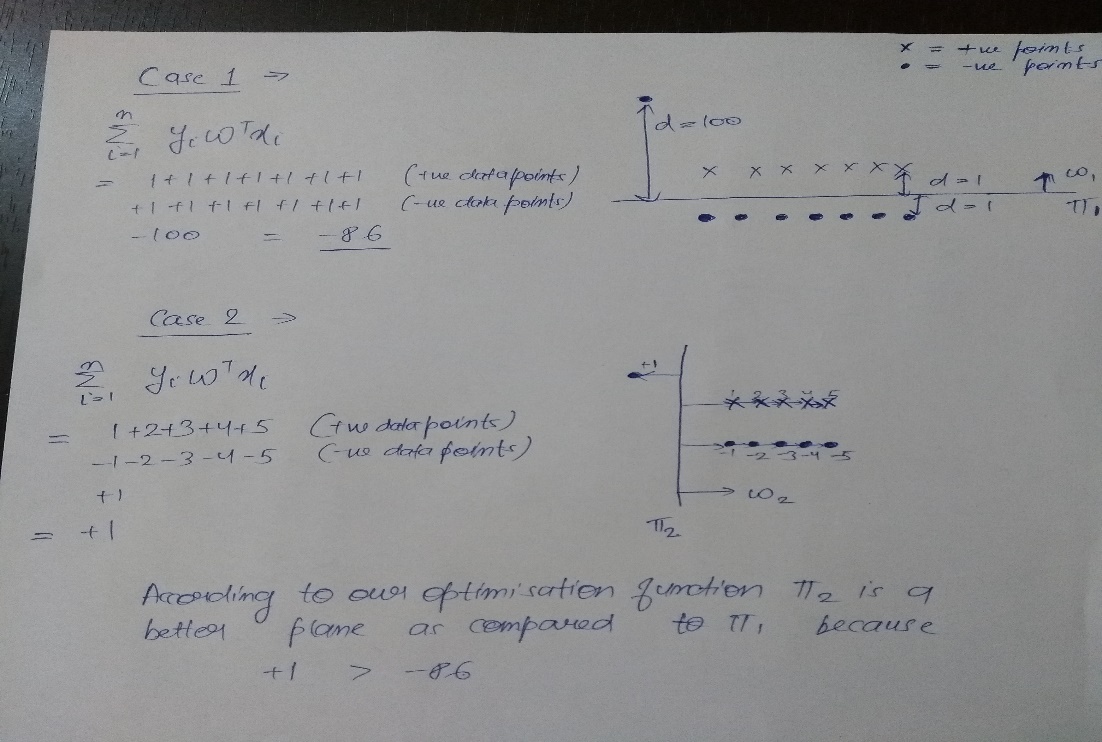


Yi \* WTX = +ve if correctly classified

-ve if wrongly classified

So we need to maximize the ∑ Yi \* WTX

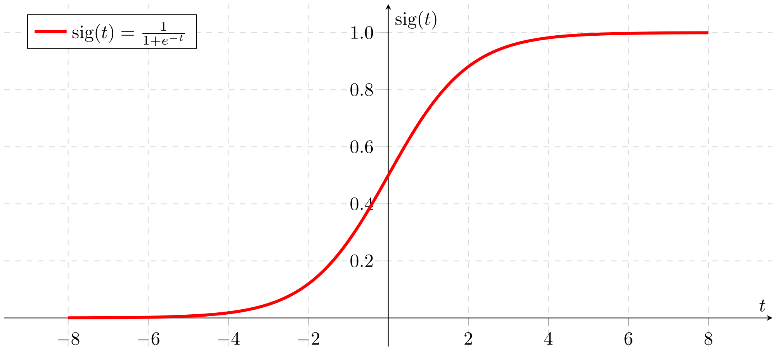


However , we have a problem

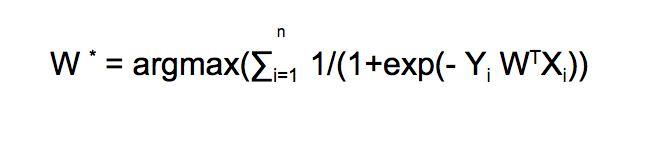
Now we have to modify **W\*=(arg max(w)(i=1 to n):(Yi\* W(transpose) Xi))**byTechnique called**Squshing.**

# SQUASHING →

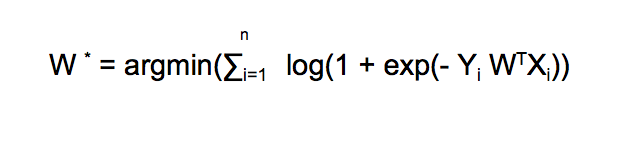
So instead of using simple signed- distance we will use: If signed- distance is small: Use as it is. If signed- distance is large: Make it a small value. So we want a function: When its value is small: increasing linearly. When its value becomes large: Tapper it off. Now, One such function we have is **SIGMOID FUNCTION**



So the new optimisation function becomes

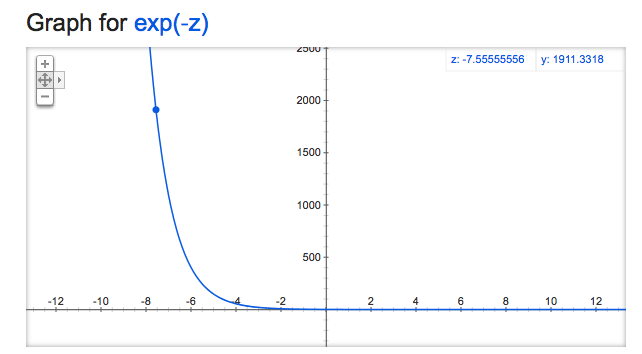


On taking log and simplyfing we get,



## Strategy for minimisation

n   
W` = argmin(∑i=1 log(1 + exp(- Yi W^T Xi)) //Optimisation Function  
Let Z = Yi W^T Xi n   
W` = argmin(∑i=1 log(1 + exp(- Zi))



From the above graph exp(-Zi) will always be positive. We are looking to minimise our optimisation function and the smallest value for exp(-Zi) is 0 .

n   
W` = argmin(∑i=1 log(1 + exp(- Zi)) >= 0

The minimum value for our optimisation function is 0, which occurs when exp(-Zi) is 0 as log(1+0) = 0.

So the overall minimum value for our optimisation function will occur when

Zi -> +∞ for all i

Let’s take a closer look at the term Zi .

Z = Yi W^T Xi

Since it is a supervised learning algorithm therefore we are given the values of X and Y.

X — Features on the basis of which we predict the correct class label

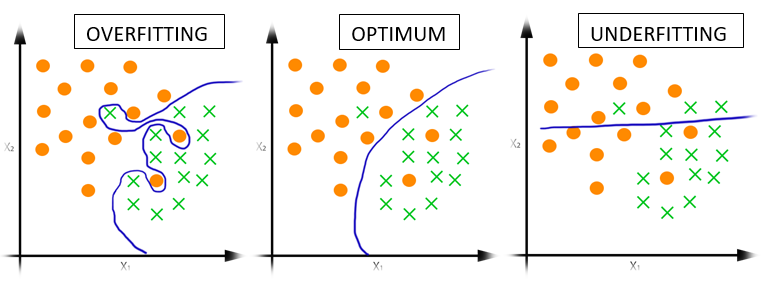
Y — The correct class label

So we can’t change Xi or Yi hence the only term left to manipulate is “W”. You can get a slight intuition that if we pick a really large value of W then only our Z will move closer to infinity.

In order to move the value of Zi to infinity we will pick a very large value (either + or -) for W.

**Case 1 —** [Yi = +1]Yi \* W^T \* Xi = +1 \* (very large +ve value of W ) \* Xi = Very large +ve value  
**Case 2—** [Yi = -1] Yi \* W^T \* Xi = -1 \* (very large -ve value of W ) \* Xi = Very large +ve value

So as you can see if we pick a large value for W then we can accomplish our goal of making Zi -> +∞

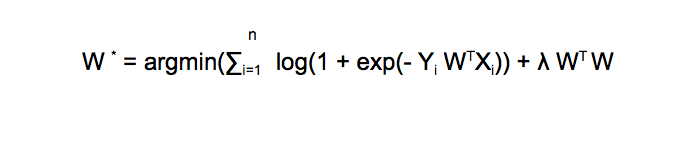
The values of W and B are found using gradient descent. **The only problem with this strategy is that we can successfully minimise our optimisation function for all values of ‘ i’.** The main problem here is that we are **overfitting** our model. 

There are two major types of regularisation —

1. L2 regularisation
2. L1 regularisation

**L2 regularisation —**

In L2 regularisation we introduce an additional term called the regularisation term in order to prevent overfitting.



Here ‘λ’ is a hyperparameter which will play an important role in our classification model but first let’s focus on the affect of this regularisation term.

If you remember that our goal was to make Zi -> +∞ and since Xi and Yi are fixed therefore we could only tweak the value of W and here you can see that we are multiplying W^TW with λ.

So earlier we were increasing the value of W to make it +∞ or -∞ but now if we try to do that then although the value of our loss term will move towards 0 but the value of our regularisation term will be very very large.So there is essentially a trade-off between loss term and regularisation term.

*The regularisation term essentially penalises our model for choosing very large values of W, hence avoiding overfitting.*

**Role of λ**

λ plays a key role in optimising our function.

* If we significantly decrease the value of λ then the model overfits as the affect of regularisation term becomes negligible.
* If we significantly increase the value of λ then our model underfits as the loss term becomes negligible and the regularisation term doesn’t contain any training data.

## L1 regularisation

The purpose of L1 regularisation is same as that of L2 i.e. avoiding overfitting in this case.

n  
W \* = argmin(∑i=1 log(1 + exp(- Yi W^TXi)) - Loss termλ ||W|| - Regularisation Term n   
Here ||W|| = ∑i=1 |Wi| where n is the number of data points and |Wi| represents the absolute value of W.

The main difference between L1 and L2 Regularisation is that L1 Regularisation creates sparse vectors.

**K-NN classification**

**The KNN Algorithm**

1. Load the data
2. Initialize K to your chosen number of neighbors

3. For each example in the data

3.1 Calculate the distance between the query example and the current example from the data.

3.2 Add the distance and the index of the example to an ordered collection

4. Sort the ordered collection of distances and indices from smallest to largest (in ascending order) by the distances

5. Pick the first K entries from the sorted collection

6. Get the labels of the selected K entries

7. If regression, return the mean of the K labels

8. If classification, return the mode of the K labels

## Choosing the right value for K

Here are some things to keep in mind:

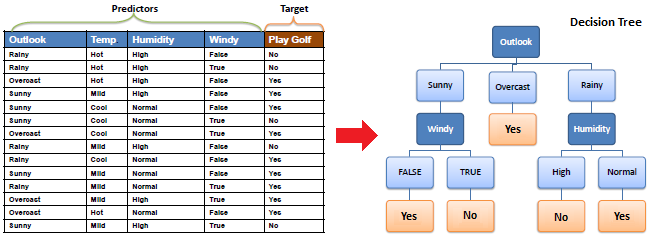
1. As we decrease the value of K to 1, our predictions become less stable. Just think for a minute, imagine K=1 and we have a query point surrounded by several reds and one green (I’m thinking about the top left corner of the colored plot above), but the green is the single nearest neighbor. Reasonably, we would think the query point is most likely red, but because K=1, KNN incorrectly predicts that the query point is green.
2. Inversely, as we increase the value of K, our predictions become more stable due to majority voting / averaging, and thus, more likely to make more accurate predictions (up to a certain point). Eventually, we begin to witness an increasing number of errors. It is at this point we know we have pushed the value of K too far.
3. In cases where we are taking a majority vote (e.g. picking the mode in a classification problem) among labels, we usually make K an odd number to have a tiebreaker.

**Advantages**

1. The algorithm is simple and easy to implement.
2. There’s no need to build a model, tune several parameters, or make additional assumptions.
3. The algorithm is versatile. It can be used for classification, regression, and search (as we will see in the next section).

**Disadvantages**

1. The algorithm gets significantly slower as the number of examples and/or predictors/independent variables increase.
2. **Decision Tree**



|  |  |  |
| --- | --- | --- |
| Decision Tree - Classification |  |  |
| Decision tree builds classification or regression models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with **decision nodes** and **leaf nodes**. A decision node (e.g., Outlook) has two or more branches (e.g., Sunny, Overcast and Rainy). Leaf node (e.g., Play) represents a classification or decision. The topmost decision node in a tree which corresponds to the best predictor called **root node**. Decision trees can handle both categorical and numerical data. |  |  |
|  |  |  |
| Algorithm |  |  |
| The core algorithm for building decision trees called **ID3** by J. R. Quinlan which employs a top-down, greedy search through the space of possible branches with no backtracking. ID3 uses *Entropy* and *Information Gain* to construct a decision tree. In ZeroR model there is no predictor, in OneR model we try to find the single best predictor, naive Bayesian includes all predictors using Bayes' rule and the independence assumptions between predictors but decision tree includes all predictors with the dependence assumptions between predictors. |  |  |
|  |  |  |
| **Entropy** |  |  |
| A decision tree is built top-down from a root node and involves partitioning the data into subsets that contain instances with similar values (homogenous). ID3 algorithm uses entropy to calculate the homogeneity of a sample. If the sample is completely homogeneous the entropy is zero and if the sample is an equally divided it has entropy of one. |  |  |
|  |  |  |
|  |  |  |
| To build a decision tree, we need to calculate two types of entropy using frequency tables as follows: |  |  |
|  |  |  |
| a) Entropy using the frequency table of one attribute: |  |  |
| https://www.saedsayad.com/images/Entropy_3.png |  |  |
| https://www.saedsayad.com/images/Entropy.pngb) Entropy using the frequency table of two attributes: https://www.saedsayad.com/images/Entropy_2.png |  |  |
|  |  |  |
|  |  |  |
| **Information Gain** |  |  |
| The information gain is based on the decrease in entropy after a dataset is split on an attribute. Constructing a decision tree is all about finding attribute that returns the highest information gain (i.e., the most homogeneous branches). |  |  |
|  |  |  |
| *Step 1*: Calculate entropy of the target. |  |  |
| https://www.saedsayad.com/images/Entropy_target.png |  |  |
| *Step 2*: The dataset is then split on the different attributes. The entropy for each branch is calculated. Then it is added proportionally, to get total entropy for the split. The resulting entropy is subtracted from the entropy before the split. The result is the Information Gain, or decrease in entropy. |  |  |
| https://www.saedsayad.com/images/Entropy_attributes.png |  |  |
| https://www.saedsayad.com/images/Entropy_gain.png |  |  |
| *Step 3*: Choose attribute with the largest information gain as the decision node, divide the dataset by its branches and repeat the same process on every branch. |  |  |
| https://www.saedsayad.com/images/Entropy_attribute_best.png |  |  |
| https://www.saedsayad.com/images/decision_tree_slices.png |  |  |
| *Step 4a*: A branch with entropy of 0 is a leaf node. |  |  |
| https://www.saedsayad.com/images/Entropy_overcast.png |  |  |
| *Step 4b*: A branch with entropy more than 0 needs further splitting. |  |  |
| https://www.saedsayad.com/images/Entropy_Sunny.png |  |  |
| *Step 5*: The ID3 algorithm is run recursively on the non-leaf branches, until all data is classified. |  |  |
| Decision Tree – OverfittingOverfitting is a significant practical difficulty for decision tree models and many other predictive models. Overfitting happens when the learning algorithm continues to develop hypotheses that reduce training set error at the cost of an increased test set error. There are several approaches to avoiding overfitting in building decision trees.  * **Pre-pruning** that stop growing the tree earlier, before it perfectly classifies the training set. * **Post-pruning** that allows the tree to perfectly classify the training set, and then post prune the tree.   Practically, the second approach of post-pruning overfit trees is more successful because it is not easy to precisely estimate when to stop growing the tree. The important step of tree pruning is to define a criterion be used to determine the correct final tree size using one of the following methods:   1. Use a distinct dataset from the training set (called validation set), to evaluate the effect of post-pruning nodes from the tree. 2. Build the tree by using the training set, then apply a statistical test to estimate whether pruning or expanding a particular node is likely to produce an improvement beyond the training set.    * Error estimation    * Significance testing (e.g., Chi-square test) 3. Minimum Description Length principle : Use an explicit measure of the complexity for encoding the training set and the decision tree, stopping growth of the tree when this encoding size (size(tree) + size(misclassifications(tree)) is minimized.   The first method is the most common approach. In this approach, the available data are separated into two sets of examples: a training set, which is used to build the decision tree, and a validation set, which is used to evaluate the impact of pruning the tree. The second method is also a common approach. Here, we explain the error estimation and Chi2 test.   **Post-pruning using Error estimation**  https://www.saedsayad.com/images/Decision_tree_error.pngError estimate for a sub-tree is weighted sum of error estimates for all its leaves. The error estimate (*e*) for a node is:  In the following example we set *Z* to 0.69 which is equal to a confidence level of 75%.  https://www.saedsayad.com/images/Decision_tree_prune.png  The error rate at the parent node is 0.46 and since the error rate for its children (0.51) increases with the split, we do not want to keep the children. **Post-pruning using Chi2 test**In [Chi2 test](https://www.saedsayad.com/categorical_categorical.htm) we construct the corresponding frequency table and calculate the Chi2 value and its probability.     |  |  |  |  | | --- | --- | --- | --- | |  | Bronze | Silver | Gold | | Bad | 4 | 1 | 4 | | Good | 2 | 1 | 2 |   Chi2 = 0.21          Probability = 0.90         degree of freedom=2   If we require that the probability has to be less than a limit (e.g., 0.05), therefore we decide not to split the node. |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| Decision Tree - Regression |  |  |
| Decision tree builds regression or classification models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with **decision nodes** and **leaf nodes**. A decision node (e.g., Outlook) has two or more branches (e.g., Sunny, Overcast and Rainy), each representing values for the attribute tested. Leaf node (e.g., Hours Played) represents a decision on the numerical target. The topmost decision node in a tree which corresponds to the best predictor called **root node**. Decision trees can handle both categorical and numerical data. |  |  |
| https://saedsayad.com/images/Decision_tree_r1.png |  |  |
|  |  |  |
| Decision Tree Algorithm |  |  |
| The core algorithm for building decision trees called **ID3** by J. R. Quinlan which employs a top-down, greedy search through the space of possible branches with no backtracking. The ID3 algorithm can be used to construct a decision tree for regression by replacing Information Gain with *Standard Deviation* *Reduction*. |  |  |
|  |  |  |
| **Standard Deviation** |  |  |
| A decision tree is built top-down from a root node and involves partitioning the data into subsets that contain instances with similar values (homogenous). We use standard deviation to calculate the homogeneity of a numerical sample. If the numerical sample is completely homogeneous its standard deviation is zero. |  |  |
|  |  |  |
| a) Standard deviation for **one** attribute: |  |  |
| https://saedsayad.com/images/Decision_tree_r2.png |  |  |
|  |  |  |
| * Standard Deviation (**S**) is for tree building (branching). * Coefficient of Deviation (**CV**) is used to decide when to stop branching. We can use Count (**n**) as well. * Average (**Avg**) is the value in the leaf nodes. |  |  |
| b) Standard deviation for **two** attributes (target and predictor): |  |  |
| https://saedsayad.com/images/Decision_tree_r3.png |  |  |
|  |  |  |
| **Standard Deviation Reduction** |  |  |
| The standard deviation reduction is based on the decrease in standard deviation after a dataset is split on an attribute. Constructing a decision tree is all about finding attribute that returns the highest standard deviation reduction (i.e., the most homogeneous branches). |  |  |
|  |  |  |
| ***Step 1***: The standard deviation of the target is calculated. |  |  |
|  |  |  |
| **Standard deviation (Hours Played) = 9.32** |  |  |
|  |  |  |
| ***Step 2***: The dataset is then split on the different attributes. The standard deviation for each branch is calculated. The resulting standard deviation is subtracted from the standard deviation before the split. The result is the standard deviation reduction. |  |  |
| https://saedsayad.com/images/Decision_tree_r4.png |  |  |
| https://saedsayad.com/images/Decision_tree_r5.png |  |  |
| ***Step 3***: The attribute with the largest standard deviation reduction is chosen for the decision node. |  |  |
| https://saedsayad.com/images/Decision_tree_r6.png |  |  |
| ***Step 4a***: The dataset is divided based on the values of the selected attribute. This process is run recursively on the non-leaf branches, until all data is processed. |  |  |
| https://saedsayad.com/images/Decision_tree_r7.png |  |  |
| In practice, we need some termination criteria. For example, when coefficient of deviation (**CV**) for a branch becomes smaller than a certain threshold (e.g., 10%) and/or when too few instances (**n**) remain in the branch (e.g., 3). |  |  |
|  |  |  |
| ***Step 4b***: "Overcast" subset does not need any further splitting because its CV (8%) is less than the threshold (10%). The related leaf node gets the average of the "Overcast" subset. |  |  |
| https://saedsayad.com/images/Decision_tree_r9.png |  |  |
|  |  |  |
| ***Step 4c***: However, the "Sunny" branch has an CV (28%) more than the threshold (10%) which needs further splitting. We select "Windy" as the best best node after "Outlook" because it has the largest SDR. |  |  |
| https://saedsayad.com/images/Decision_tree_r10.png |  |  |
| Because the number of data points for both branches (FALSE and TRUE) is equal or less than 3 we stop further branching and assign the average of each branch to the related leaf node. |  |  |
| https://saedsayad.com/images/Decision_tree_r8.png |  |  |
|  |  |  |
| ***Step 4d***: Moreover, the "rainy" branch has an CV (22%) which is more than the threshold (10%). This branch needs further splitting. We select "Windy" as the best best node because it has the largest SDR. |  |  |
| https://saedsayad.com/images/Decision_tree_r11.png |  |  |
| Because the number of data points for all three branches (Cool, Hot and Mild) is equal or less than 3 we stop further branching and assign the average of each branch to the related leaf node. |  |  |
| https://saedsayad.com/images/Decision_tree_r12.png |  |  |
|  |  |  |
| When the number of instances is more than one at a *leaf node* we calculate the *average* as the final value for the target. **Random Forest**Introduction Random forest is a supervised learning algorithm which is used for both classification as well as regression. But however, it is mainly used for classification problems. As we know that a forest is made up of trees and more trees means more robust forest. Similarly, random forest algorithm creates decision trees on data samples and then gets the prediction from each of them and finally selects the best solution by means of voting. It is an ensemble method which is better than a single decision tree because it reduces the over-fitting by averaging the result. Working of Random Forest Algorithm We can understand the working of Random Forest algorithm with the help of following steps −   * **Step 1** − First, start with the selection of random samples from a given dataset. * **Step 2** − Next, this algorithm will construct a decision tree for every sample. Then it will get the prediction result from every decision tree. * **Step 3** − In this step, voting will be performed for every predicted result. * **Step 4** − At last, select the most voted prediction result as the final prediction result.   For regression we take its average! |  |  |

