**MACHINE LEARNING**

CREATED BY VISHNU MALLELA

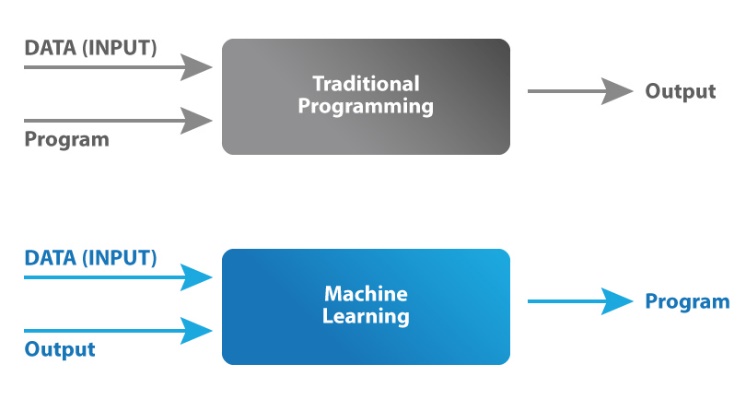
**MODULE-1:**

**(INTRODUCTION TO ML)**

**ML AND APPLICATION, LEARNING PARADIGMS, PERSPECTIVES AND ISSUES, FINITE AND INFINITE HYPOTHESIS, PAC LEARNING, FIND S AND CANDIDATE ELIMINATION**

**Machine Learning:**

Machine learning is defined as use of algorithms and computational statistics to learn from data without being explicitly programmed.



**Applications:**

* Image Recognition

Image Processing, or more specifically, Digital Image Processing, is the process of processing a digital image using a series of algorithms. Digital Image Processing Using Neural Networks has grown in popularity in recent years.

* Speech Recognition

speech recognition is the process of turning spoken commands to text and subsequently classifying, segmenting, and so on. This method is used by a variety of virtual assistants, including Google Assistant, Siri, Cortana, and others.

* Self-driving Automobiles

Machine learning is crucial in self-driving cars. Tesla, the most well-known vehicle manufacturer, is developing a self-driving car. It trains automobile models to recognize people and objects using unsupervised learning and reinforcement learning methods.

* Automatic Language Translation

Machine translation is a task that generally uses machine learning models generated using extremely complex linguistic knowledge and other related data to produce accurate text translation from one language to another. Machine Translations have become an essential aspect of corporate transactions when combined with Natural Language Understanding, which also employs Supervised Learning.

* Online Fraud Detection

Machine learning is proving its potential to make cyberspace a secure place and tracking monetary frauds online is one of its examples. For example: PayPal is using ML for protection against money laundering.

* Product suggestions

Various retail, eCommerce, and entertainment firms can produce suggestions for their users based on various levels of associativity by utilizing various association rule engines. For example, Amazon product recommendations, Netflix, and so on.

* Transportation and Commuting

If you have used an app to book a cab, you are already using Machine Learning to an extent. It provides a personalized application that is unique to you. Automatically detects your location and provides options to either go home or office or any other frequent place based on your History and Patterns.

**Advantages and Challenges:**

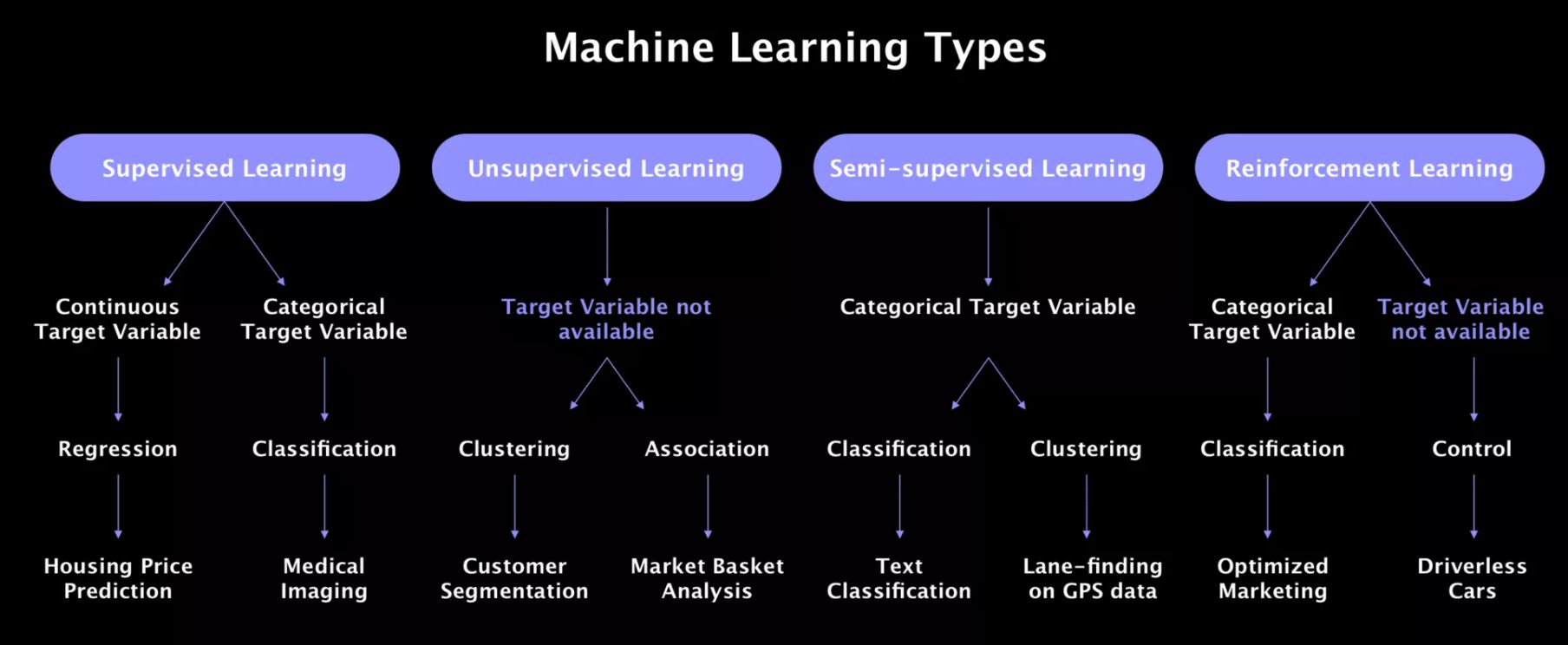
**Advantages:**

1. Easily identifies trends and patterns
2. No human intervention needed (automation)
3. Continuous Improvement
4. Handling multi-dimensional and multi-variety data
5. Wide Applications
6. Solve unprogrammable tasks

**Disadvantages:**

1. Requires Big Data
2. Requires Supervised Data
3. Time, Space and Resources
4. Data Acquisition
5. High error-susceptibility
6. Algorithm Selection

**ML Paradigms:**

****

* **Supervised Learning**

Supervised learning means that the training data that we feed our algorithm with has labels on it. As a result, it maps the input (training data) to output and labels the data accordingly.

Supervised learning is a machine learning task in which a function maps the input to output data using the provided input-output pairs.

1. **Classification:**

The model assigns a category to the target variable. The target variable is the category you want your algorithm to find.

1. **Regression:**

The model assigns a continuous variable to the target variable.

* **Unsupervised Learning**

In unsupervised learning, we don’t have a column for the target variable — we actually don’t really know what we are looking for.

In this type of learning paradigm, the computer is provided with just the input to develop a learning pattern. It is basically Learning from no results!!

1. **Clustering:**

identifies the consumption behavior of each category and identifies how many groups we should have and who should be placed into each group. In summary, our models divide the dataset according to its similarities. This is called Clustering, a subcategory of Unsupervised Learning.

1. **Association:**

An association rule learning problem is where you want to discover rules that describe large portions of your data

* **Reinforcement Learning**

Reinforcement Learning is a type of Machine Learning, and thereby also a branch of Artificial Intelligence. It allows machines and software agents to automatically determine the ideal behavior within a specific context, in order to maximize its performance.

**CONCEPT LEARNING**

‘’ Problem of searching through a predefined space of potential hypotheses for the hypothesis that best fits the training examples”. Concept Learning is a type of Supervised Learning.

Teaching a machine to distinguish between examples and non-examples of ideas such as symphony, anger, beauty, dog, cat etc. is called concept learning. In concept learning, we aim to use data to teach a machine to solve a binary classification problem. That is, to classify a data-point as either belonging to or not belonging to a particular concept or idea.

Feature Space:

a feature space is just the set of all possible values for a chosen set of features from that data. It is always possible to represent feature values and thus a feature space using only numbers, and further to do so in such a way that the feature space can be interpreted as a real space.

**FIND -S ALOGORITHM**

(CONSIDER ONLY +VE EXAMPLES)

The find-S algorithm starts with the most specific hypothesis and generalizes this hypothesis each time it fails to classify an observed positive training data. Hence, the Find-S algorithm moves from the most specific hypothesis to the most general hypothesis.

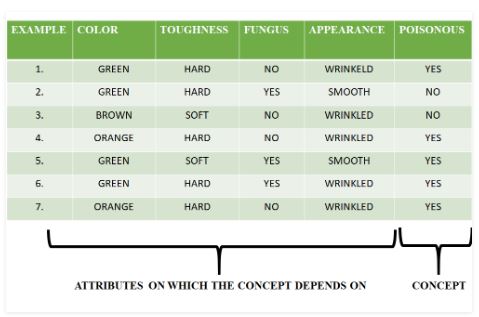
**Most general hypothesis: {?, ?, ?, ?, ?, ?}**

**Most specific hypothesis:** **{ϕ, ϕ, ϕ, ϕ, ϕ, ϕ}**

Steps Involved in Find-S:

1. Start with the most specific hypothesis.   
   **h = {ϕ, ϕ, ϕ, ϕ, ϕ, ϕ}**
2. Take the next example and if it is negative, then no changes occur to the hypothesis.
3. If the example is positive and we find that our initial hypothesis is too specific then we update our current hypothesis to a general condition.
4. Keep repeating the above steps till all the training examples are complete.
5. After we have completed all the training examples, we will have the final hypothesis when can use to classify the new examples.

EXAMPLE:



First, we consider the hypothesis to be a more specific hypothesis. Hence, our hypothesis would be:

h = {ϕ, ϕ, ϕ, ϕ, ϕ, ϕ}

Consider example 1:

The data in example 1 is {GREEN, HARD, NO, WRINKLED}. We see that our initial hypothesis is more specific and we have to generalize it for this example. Hence, the hypothesis becomes:

h = {GREEN, HARD, NO, WRINKLED}

Consider example 2:

Here we see that this example has a negative outcome. Hence, we neglect this example and our hypothesis remains the same.

h = {GREEN, HARD, NO, WRINKLED}

Consider example 3:

Here we see that this example has a negative outcome. Hence, we neglect this example and our hypothesis remains the same.

h = {GREEN, HARD, NO, WRINKLED}

Consider example 4:

The data present in example 4 is {ORANGE, HARD, NO, WRINKLED}. We compare every single attribute with the initial data and if any mismatch is found we replace that particular attribute with a general case (”?”). After doing the process the hypothesis becomes:

h = {?, HARD, NO, WRINKLED }

Consider example 5:

The data present in example 5 is {GREEN, SOFT, YES, SMOOTH}. We compare every single attribute with the initial data and if any mismatch is found we replace that particular attribute with a general case (”?”). After doing the process the hypothesis becomes:

h = { ?, ?, ?, ? }

Since we have reached a point where all the attributes in our hypothesis have the general condition, example 6 and example 7 would result in the same hypothesizes with all general attributes.

h = { ?, ?, ?, ? }

Hence, for the given data the final hypothesis would be:

Final Hypothesis: h = { ?, ?, ?, ? }

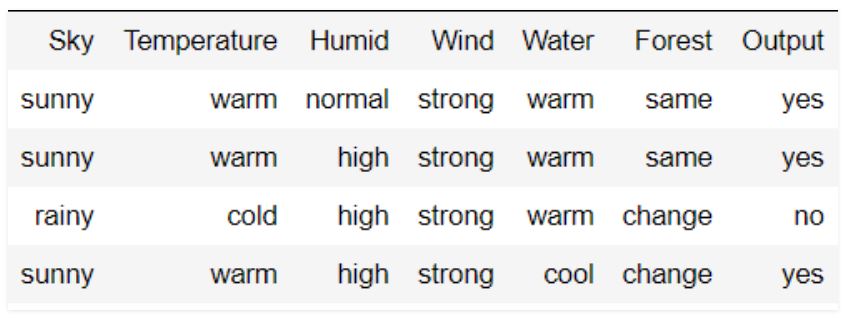
**CANDIDATE ELIMINATION ALGORITHM:**

(CONSIDER BOTH +VE AND -VE EXAMPLES)

The candidate elimination algorithm incrementally builds the version space given a hypothesis space H and a set E of examples. The examples are added one by one; each example possibly shrinks the version space by removing the hypotheses that are inconsistent with the example. The candidate elimination algorithm does this by updating the general and specific boundary for each new example.

* You can consider this as an extended form of Find-S algorithm.
* Consider both positive and negative examples.
* Actually, positive examples are used here as Find-S algorithm (Basically they are generalizing from the specification).
* While the negative example is specified from generalize form**.**

EXAMPLE:



**Initially:** G =[[?, ?, ?, ?, ?, ?], [?, ?, ?, ?, ?, ?], [?, ?, ?, ?, ?, ?],

[?, ?, ?, ?, ?, ?], [?, ?, ?, ?, ?, ?], [?, ?, ?, ?, ?, ?]]

S = [Null, Null, Null, Null, Null, Null]

**For instance 1 :** <'sunny','warm','normal','strong','warm ','same'> and positive output.

G1 = G

S1 = ['sunny','warm','normal','strong','warm ','same']

**For instance 2 :** <'sunny','warm','high','strong','warm ','same'> and positive output.

G2 = G

S2 = ['sunny','warm',?,'strong','warm ','same']

**For instance 3 :** <'rainy','cold','high','strong','warm ','change'> and negative output.

G3 = [['sunny', ?, ?, ?, ?, ?], [?, 'warm', ?, ?, ?, ?], [?, ?, ?, ?, ?, ?],

[?, ?, ?, ?, ?, ?], [?, ?, ?, ?, ?, ?], [?, ?, ?, ?, ?, 'same']]

S3 = S2

**For instance 4 :** <'sunny','warm','high','strong','cool','change'> and positive output.

G4 = G3

S4 = ['sunny','warm',?,'strong', ?, ?]

At last, by synchronizing the G4 and S4 algorithm produce the output.

**MODULE -2**

**(SUPERVISED LEARNING - 1)**

**Learning a Class from Examples, Linear, Non-linear, Multi-class and Multi-label classification, Generalization error bounds: VC Dimension, Decision Trees: ID3, Classification and Regression Trees, Regression: Linear Regression, Multiple Linear Regression, Logistic Regression.**

**REGRESSION:**

Regression is a technique for investigating the relationship between independent variables or features and a dependent variable or outcome. It’s used as a method for predictive modelling in machine learning, in which an algorithm is used to predict continuous outcomes.

**Types of Regression:**

1. Simple Linear Regression:

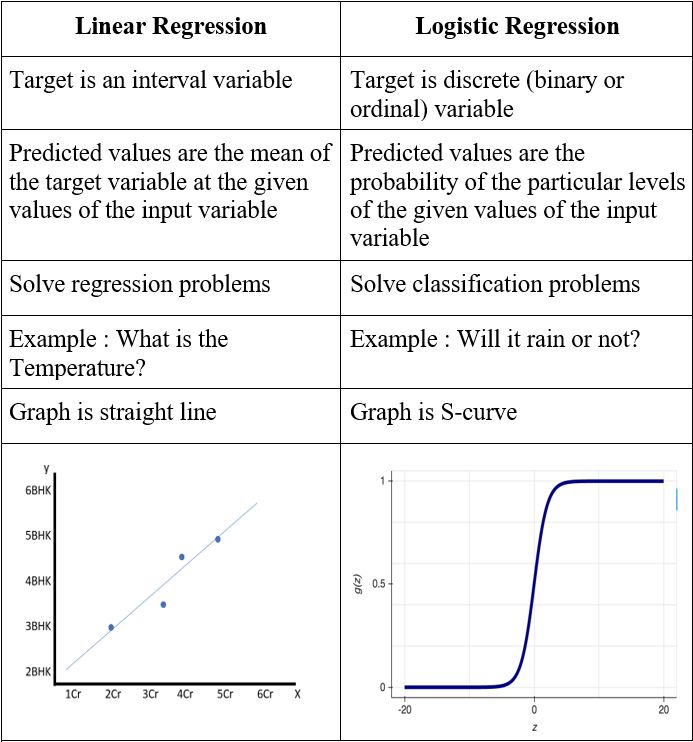
Simple Linear regression is a linear regression technique which plots a straight line within data points to minimise error between the line and the data points.

1. Multiple linear regression:

Multiple linear regression is a technique used when more than one independent variable is used. Polynomial regression is an example of a multiple linear regression technique. It is a type of multiple linear regression, used when there is more than one independent variable. It achieves a better fit in the comparison to simple linear regression when multiple independent variables are involved. The result when plotted on two dimensions would be a curved line fitted to the data points.

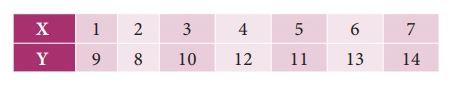
1. Logistic regression:

Logistic regression is used when the dependent variable can have one of two values, such as true or false, or success or failure. Logistic regression models can be used to predict the probability of a dependent variable occurring. Generally, the output values must be binary. A sigmoid curve can be used to map the relationship between the dependent variable and independent variables.

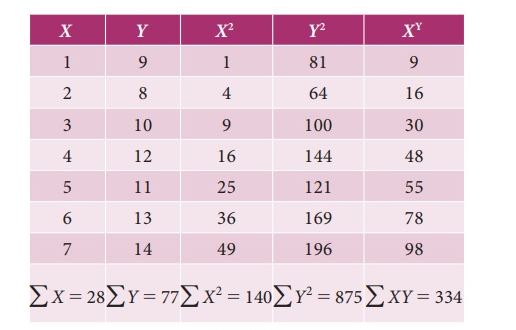


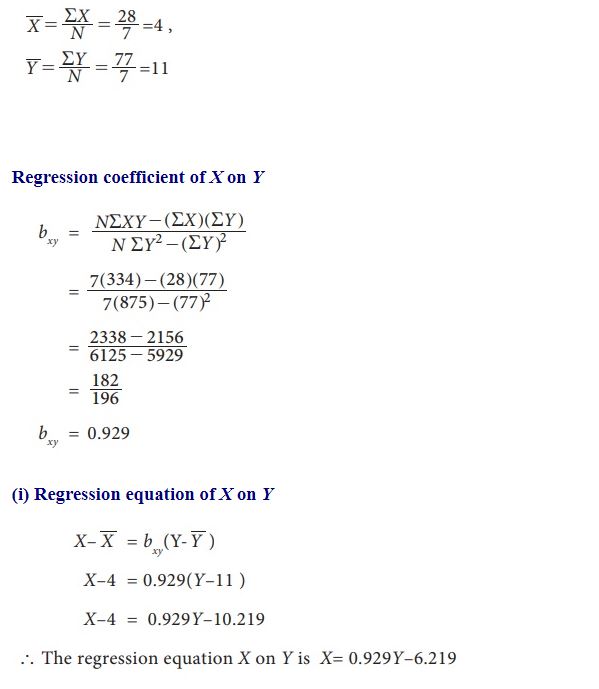
**PROBLEMS:**

1. Calculate the regression coefficient and obtain the lines of regression for the following data (SIMPLE LINEAR REGRESSION)



SOLUTION:





**CLASSIFICATION:**

Classification in machine learning and statistics is a supervised learning approach in which the computer program learns from the data given to it and make new observations or classifications.

**Types of Classification:**

1. **Binary Classification:**

It is a type of classification with two outcomes, for e.g. – either true or false, spam or not.

Algorithms:

1. Logistic Regression
2. k-Nearest Neighbors
3. Decision Trees
4. Support Vector Machine
5. Naive Bayes
6. **Multi-Class Classification:**

The classification with more than two classes, in multi-class classification each sample is assigned to one and only one label or target.

Algorithms:

1. k-Nearest Neighbors.
2. Decision Trees.
3. Naive Bayes.
4. Random Forest.
5. Gradient Boosting.
6. **Multi-label Classification**

This is a type of classification where each sample is assigned to a set of labels or targets.

This is unlike binary classification and multi-class classification, where a single class label is predicted for each example.

Algorithms:

1. Multi-label Decision Trees
2. Multi-label Random Forests
3. Multi-label Gradient Boosting

**DECISION TREES**

A decision tree is a type of supervised machine learning used to categorize or make predictions based on how a previous set of questions were answered. The model is a form of supervised learning, meaning that the model is trained and tested on a set of data that contains the desired categorization.

A decision tree is a graphical representation of all possible solutions to a decision based on certain conditions.

Decision Tree models are created using 2 steps: Induction and Pruning.

1. **CLASSIFICATION TREES (Discrete/categorical):**

where the outcome was a variable like ‘fit’ or ‘unfit’. Here the decision variable is Categorical ex: CART algorithm

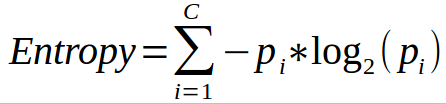
1. **REGRESSION TREES (Continuous):**

Here the decision or the outcome variable is Continuous, there are many algorithms out there which construct Decision Trees.

ex: Iterative Dichotomiser 3.

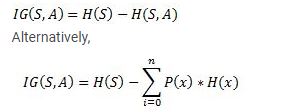
**ENTROPY (Predictability of outcome /Randomness):**

Entropy, also called as Shannon Entropy is denoted by H(S) for a finite set S, is the measure of the amount of uncertainty or randomness in data



**INFORMATION GAIN:**

Information gain is also called as Kullback-Leibler divergence denoted by IG(S,A) for a set S is the effective change in entropy after deciding on a particular attribute A. It measures the relative change in entropy with respect to the independent variables



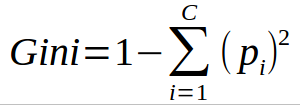
Information gain is calculated after node splitting.

**Information gain = Entropy(s) – [ (average weight) \* Entropy (each feature)]**

Information Gain = Entropy before splitting - Entropy after splitting

**Gini Index:**

Gini Index is the measure of impurity or the purity that is used in building a decision tree in the CART Algorithm.



**CART ALGORITHM:**

**ONLY BINARY TREES**

**USE “GINI INDEX” FOR BEST ATTRIBUTE MEASUREMENT**

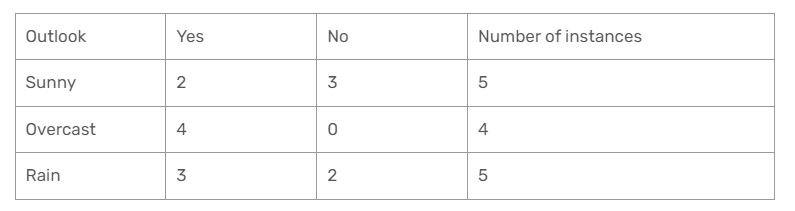
The CART stands for Classification and Regression Trees is a greedy algorithm that greedily searches for an optimum split at the top level, then repeats the same process at each of the subsequent levels.

PROBLEM:



Gini index is a metric for classification tasks in CART.

1. **OUTLOOK**

****

**Gini(Temp=Hot) = 1 – (2/4)2 – (2/4)2 = 0.5**

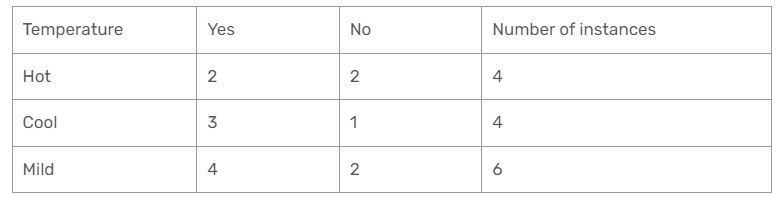
**Gini(Temp=Cool) = 1 – (3/4)2 – (1/4)2 = 1 – 0.5625 – 0.0625 = 0.375**

**Gini(Temp=Mild) = 1 – (4/6)2 – (2/6)2 = 1 – 0.444 – 0.111 = 0.445**

**We’ll calculate weighted sum of gini index for temperature feature**

**Gini(Temp) = (4/14) x 0.5 + (4/14) x 0.375 + (6/14) x 0.445 = 0.142 + 0.107 + 0.190 = 0.439**

1. **Temperature**

****

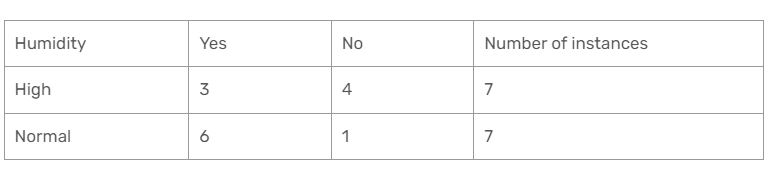
**Gini(Humidity=High) = 1 – (3/7)2 – (4/7)2 = 1 – 0.183 – 0.326 = 0.489**

**Gini(Humidity=Normal) = 1 – (6/7)2 – (1/7)2 = 1 – 0.734 – 0.02 = 0.244**

**Weighted sum for humidity feature will be calculated next**

**Gini(Humidity) = (7/14) x 0.489 + (7/14) x 0.244 = 0.367**

1. **Humidity**

****

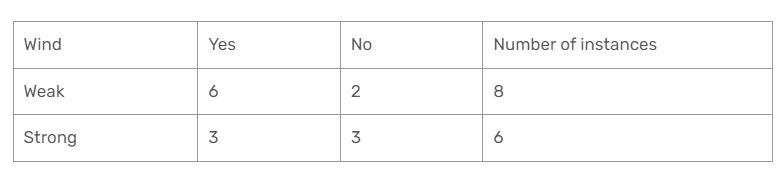
**Gini(Humidity=High) = 1 – (3/7)2 – (4/7)2 = 1 – 0.183 – 0.326 = 0.489**

**Gini(Humidity=Normal) = 1 – (6/7)2 – (1/7)2 = 1 – 0.734 – 0.02 = 0.244**

**Weighted sum for humidity feature will be calculated next**

**Gini(Humidity) = (7/14) x 0.489 + (7/14) x 0.244 = 0.367**

1. **WIND**

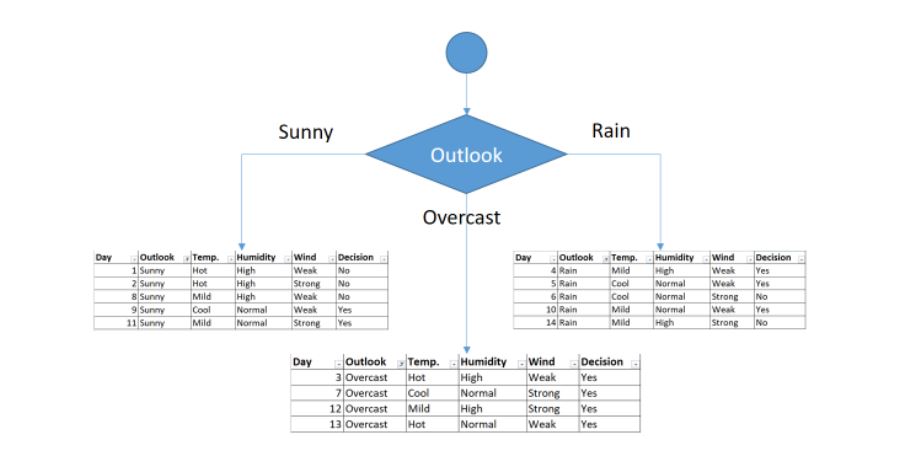
****

**Gini(Wind=Weak) = 1 – (6/8)2 – (2/8)2 = 1 – 0.5625 – 0.062 = 0.375**

**Gini(Wind=Strong) = 1 – (3/6)2 – (3/6)2 = 1 – 0.25 – 0.25 = 0.5**

**Gini(Wind) = (8/14) x 0.375 + (6/14) x 0.5 = 0.428**

**Now, Lowest Gini attribute will highest priority.**

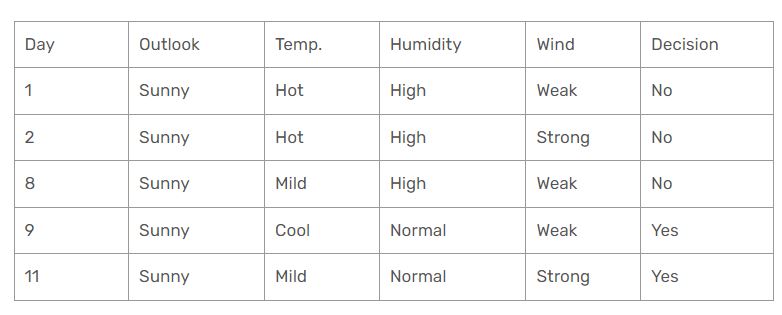
****

You might realize that sub dataset in the overcast leaf has only yes decisions. This means that overcast leaf is over.

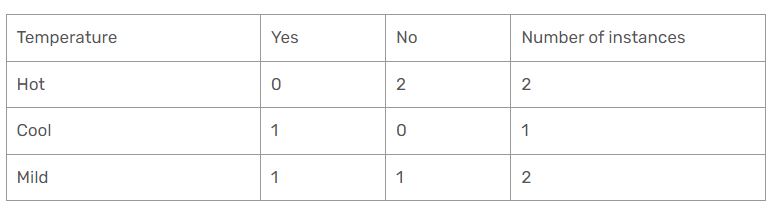
**We will apply same principles to those sub datasets in the following steps.**

**Focus on the sub dataset for sunny outlook. We need to find the gini index scores for temperature, humidity and wind features respectively.**

1. **SUNNY**

****

**Gini of temperature for sunny outlook:**

****

**Gini(Outlook=Sunny and Temp.=Hot) = 1 – (0/2)2 – (2/2)2 = 0**

**Gini(Outlook=Sunny and Temp.=Cool) = 1 – (1/1)2 – (0/1)2 = 0**

**Gini(Outlook=Sunny and Temp.=Mild) = 1 – (1/2)2 – (1/2)2 = 1 – 0.25 – 0.25 = 0.5**

**Gini(Outlook=Sunny and Temp.) = (2/5)x0 + (1/5)x0 + (2/5)x0.5 = 0.2**

**Gini of humidity for sunny outlook**

**Gini(Outlook=Sunny and Humidity=High) = 1 – (0/3)2 – (3/3)2 = 0**

**Gini(Outlook=Sunny and Humidity=Normal) = 1 – (2/2)2 – (0/2)2 = 0**

**Gini(Outlook=Sunny and Humidity) = (3/5)x0 + (2/5)x0 = 0**

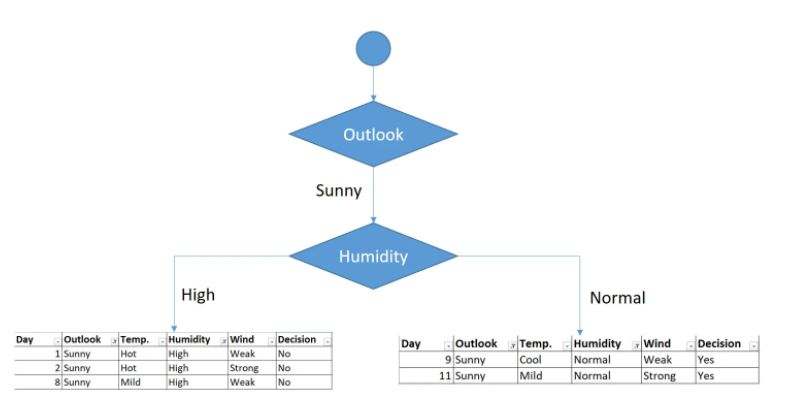
**Gini of wind for sunny outlook**

**Gini(Outlook=Sunny and Wind=Weak) = 1 – (1/3)2 – (2/3)2 = 0.266**

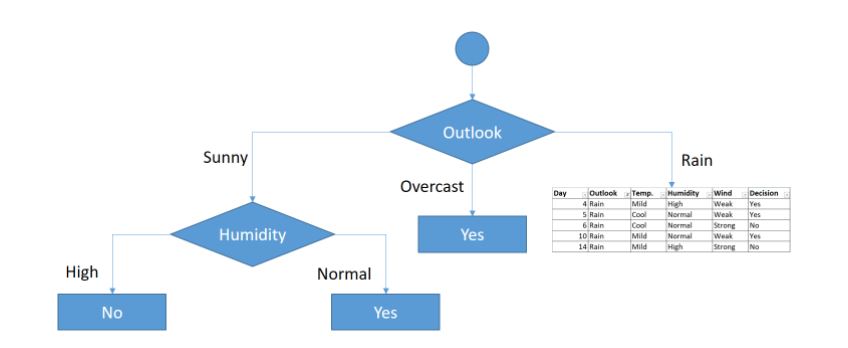
**Gini(Outlook=Sunny and Wind=Strong) = 1- (1/2)2 – (1/2)2 = 0.2**

**Gini(Outlook=Sunny and Wind) = (3/5)x0.266 + (2/5)x0.2 = 0.46**

**We’ll put humidity check at the extension of sunny outlook.**

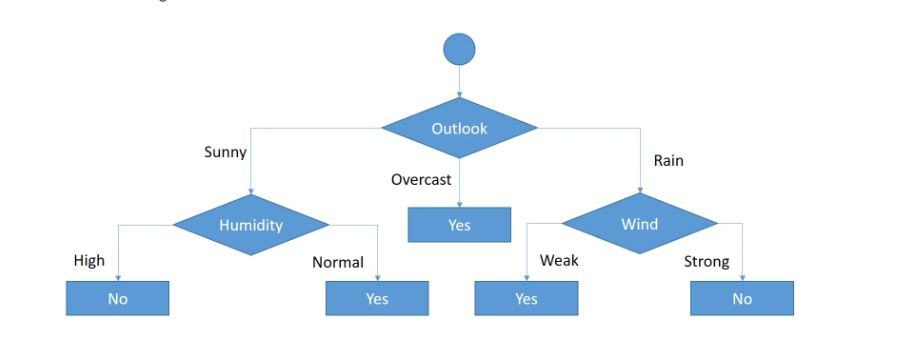
****

As seen, decision is always no for high humidity and sunny outlook. On the other hand, decision will always be yes for normal humidity and sunny outlook. This branch is over.



Now , repeating the steps for Rain.

Finally we get,



**ID-3 ALGORITHM (uses Entropy and information gain)**

Decision tree algorithms transform raw data to rule based decision-making trees. Herein, ID3 is one of the most common decision tree algorithms. Firstly, It was introduced in 1986 and it is acronym of Iterative Dichotomiser**.**

**First of all, dichotomization means dividing into two completely opposite things. That’s why, the algorithm iteratively divides attributes into two groups which are the most dominant attribute and others to construct a tree. Then, it calculates the entropy and information gains of each attribute. In this way, the most dominant attribute can be founded. After then, the most dominant one is put on the tree as decision node. Thereafter, entropy and gain scores would be calculated again among the other attributes. Thus, the next most dominant attribute is found. Finally, this procedure continues until reaching a decision for that branch. That’s why, it is called Iterative Dichotomiser.**

PROBLEM:



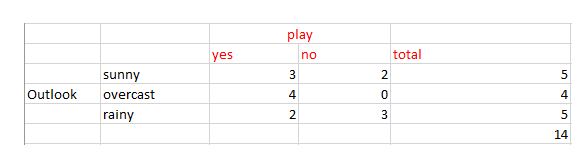
Here There are for independent variables to determine the dependent variable. The independent variables are Outlook, Temperature, Humidity, and Wind. The dependent variable is whether to play football or not.

As the first step, we have to find the parent node for our decision tree. For that follow the steps:

***Find the entropy of the class variable.***

E(S) = -[(9/14)log(9/14) + (5/14)log(5/14)] = 0.94

note: Here typically we will take log to base 2. Here total there are 14 yes/no. Out of which 9 yes and 5 no. Based on it we calculated probability above.



***Now we have to calculate average weighted entropy***. ie, we have found the total of weights of each feature multiplied by probabilities.

E(S, outlook) = (5/14)\*E(3,2) + (4/14)\*E(4,0) + (5/14)\*E(2,3) = (5/14)(-(3/5)log(3/5)-(2/5)log(2/5))+ (4/14)(0) + (5/14)((2/5)log(2/5)-(3/5)log(3/5)) = 0.693

***The next step is to find the information gain***. It is the difference between parent entropy and average weighted entropy we found above.

IG(S, outlook) = 0.94 - 0.693 = 0.247

Similarly find Information gain for Temperature, Humidity, and Windy.

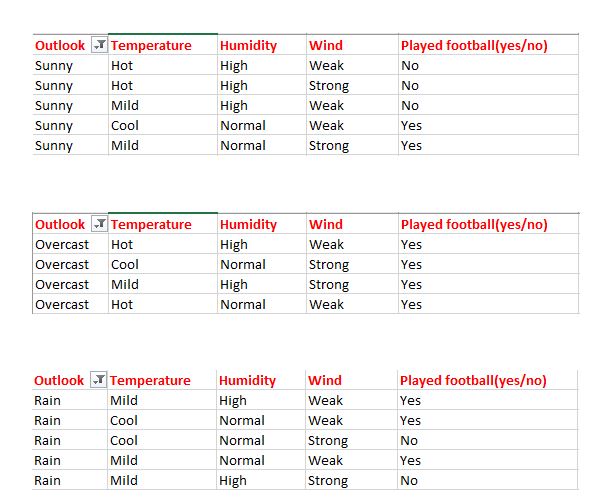
IG(S, Temperature) = 0.940 - 0.911 = 0.029

IG(S, Humidity) = 0.940 - 0.788 = 0.152

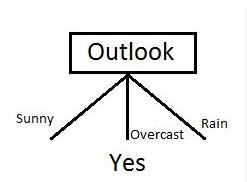
IG(S, Windy) = 0.940 - 0.8932 = 0.048

***Now select the feature having the largest entropy gain***. Here it is Outlook. So it forms the first node(root node) of our decision tree.

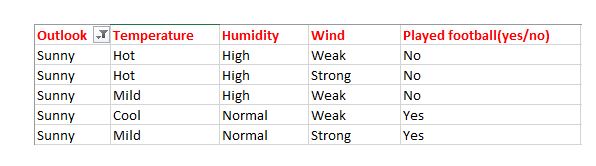
Now our data look as follows,



Since overcast contains only examples of class ‘Yes’ we can set it as yes. That means If outlook is overcast football will be played. Now our decision tree looks as follows.



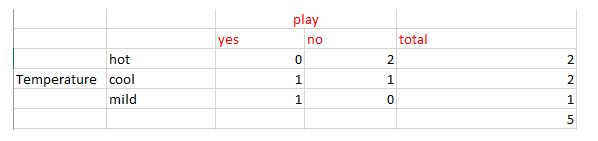
The next step is to find the next node in our decision tree. Now we will find one under sunny. We have to determine which of the following Temperature, Humidity or Wind has higher information gain.



Calculate parent entropy E(sunny)

E(sunny) = (-(3/5)log(3/5)-(2/5)log(2/5)) = 0.971.

Now Calculate the information gain of Temperature. IG(sunny, Temperature)



E(sunny, Temperature) = (2/5)\*E(0,2) + (2/5)\*E(1,1) + (1/5)\*E(1,0)=2/5=0.4

Now calculate information gain.

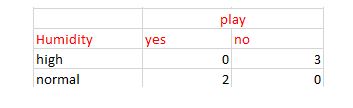
IG(sunny, Temperature) = 0.971–0.4 =0.571

Similarly we get

IG(sunny, Humidity) = 0.971

IG(sunny, Windy) = 0.020

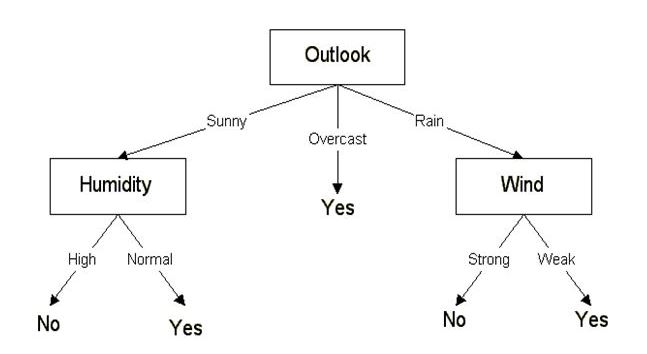
Here IG(sunny, Humidity) is the largest value. So Humidity is the node that comes under sunny.



For humidity from the above table, we can say that play will occur if humidity is normal and will not occur if it is high. Similarly, find the nodes under rainy.

***Note: A branch with entropy more than 0 needs further splitting.***

Finally, our decision tree will look as below:



**PROBLEMS LINK** : <https://medium.datadriveninvestor.com/decision-tree-algorithm-with-hands-on-example-e6c2afb40d38>

**MODULE -3**

**(SUPERVISED LEARNING -2)**

**Neural Networks: Introduction, Perceptron, Multilayer Perceptron, Support vector machines: Linear and Non-Linear, Kernel Functions, K-Nearest Neighbors**

**The art and science of Deep Learning is built on the foundation of Neural Networks and how they work.**

**Neural Networks:**

It is essentially a naive implementation of how our brains might work. It’s not a very accurate representation but it tries to replicate some of the methods our brain uses to learn from it’s mistakes

Artificial neural networks, usually simply called neural networks, are computing systems inspired by the biological neural networks that constitute animal brains. An ANN is based on a collection of connected units or nodes called artificial neurons, which loosely model the neurons in a biological brain

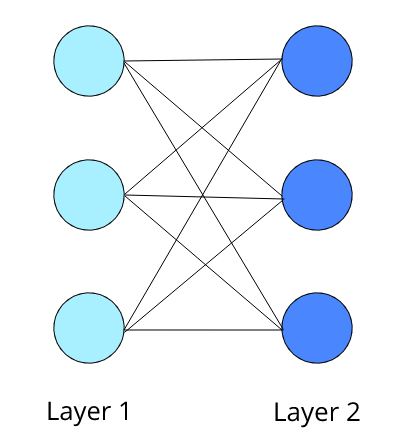
**SINGLE LAYER NEURAL NETWOR:**

**Perceptron: (Linear Binary Classifier)**

Perceptron is a single layer neural network and a multi-layer perceptron is called Neural Networks. decision made by the perceptron is then passed onto the next layer for the next perceptron to use in their decision.

Functionality:

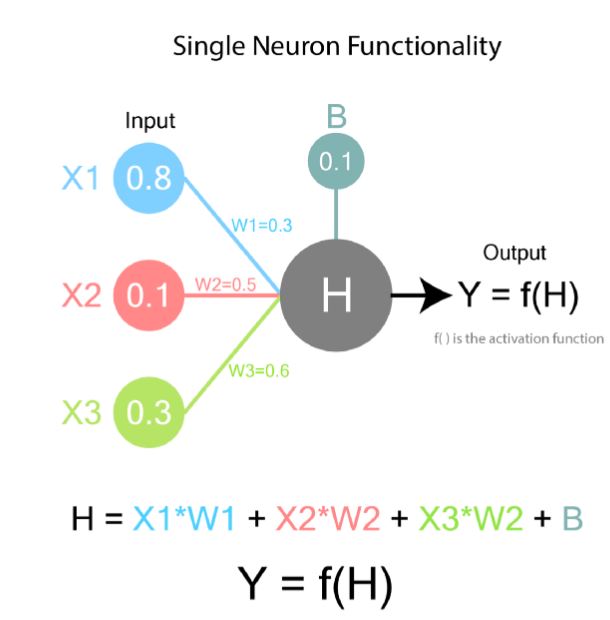
A neural network consists of many **Nodes**(Neurons) in many **layers.**Each layer can have *any number* of nodes and a neural network can have *any number* of layers. Let’s have a closer look at a couple of layers.

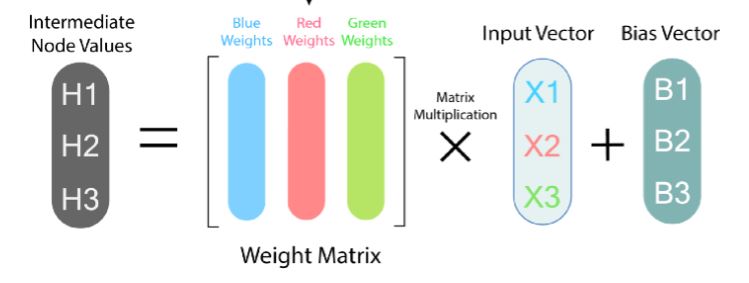


Now as you can see, there are many interconnections between both the layers. These interconnections exist between **each node** in the first layer with **each and every node** in the second layer. These are also called the **weights** between two layers.

**Weights** shows the strength of the particular node.

***A bias*** value allows you to shift the activation function curve up or down.





When we feed the inputs into the first layer, the values of the nodes will be calculated layer by layer using the matrix multiplications and activation functions till we get the final values at the output layer. That is how we get an output from a neural network.

**Role of Activation Function:**

The activation function is the most important factor in a neural network which decided whether or not a neuron will be activated or not and transferred to the next layer. This simply means that it will decide whether the neuron's input to the network is relevant or not in the process of prediction

**Types of Activation functions:**

1. **LINEAR**

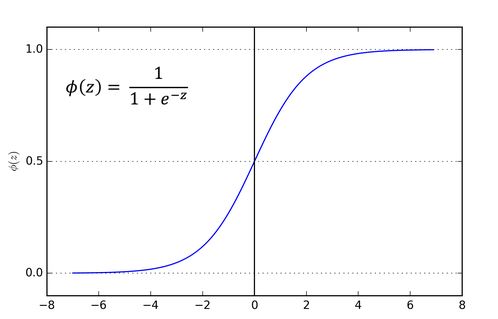
As you can see the function is a line or linear. Therefore, the output of the functions will not be confined between any range.

F(X) =X

RANGE: (-Infinity to infinity)

1. **NON-LINEAR**

**1.SIGMOID:**

****

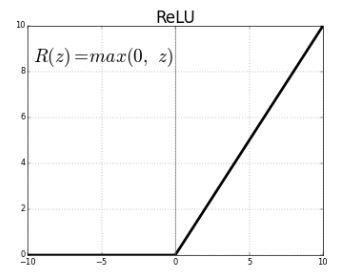
Therefore, it is especially used for models where we have to predict the probability as an output. Since probability of anything exists only between the range of 0 and 1, sigmoid is the right choice.

1. **Tanh**

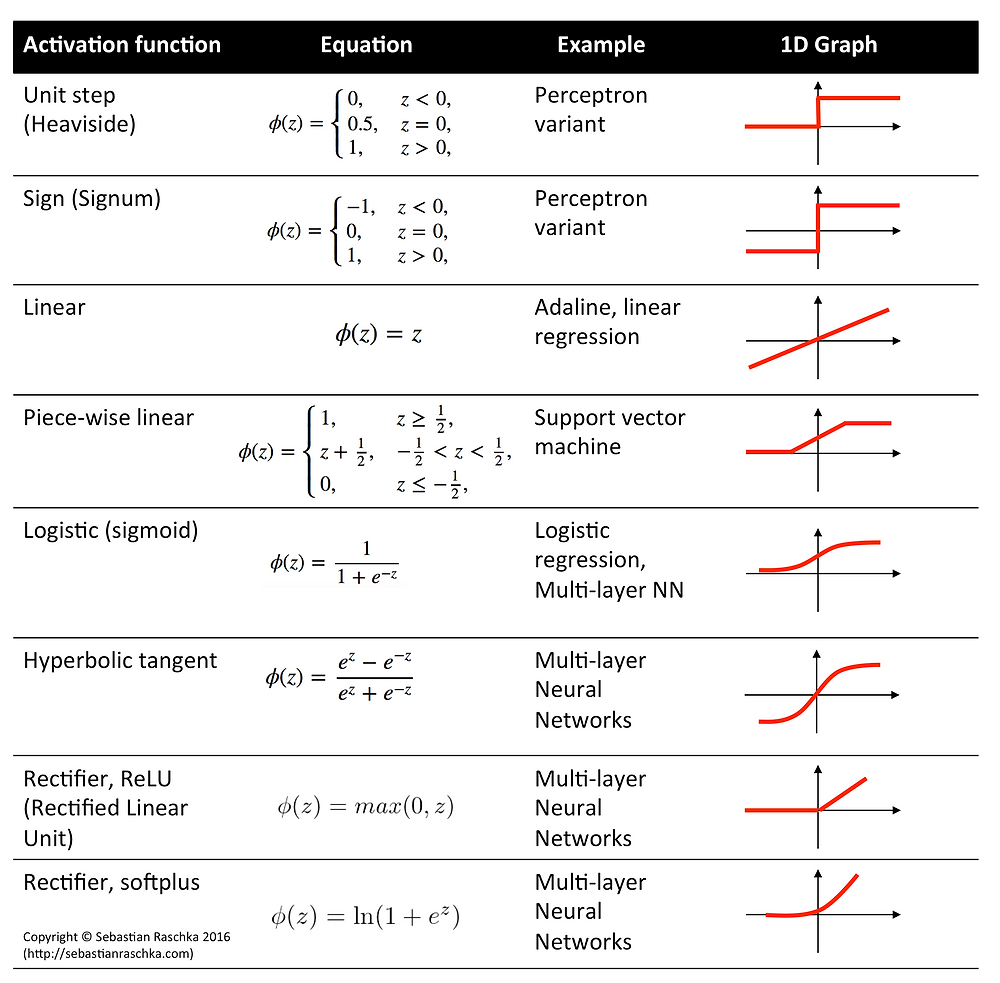
tanh is also like logistic sigmoid but better. The range of the tanh function is from (-1 to 1). tanh is also sigmoidal (s - shaped).

1. **ReLu**

The ReLU is the most used activation function in the world right now. Since, it is used in almost all the convolutional neural networks or deep learning.



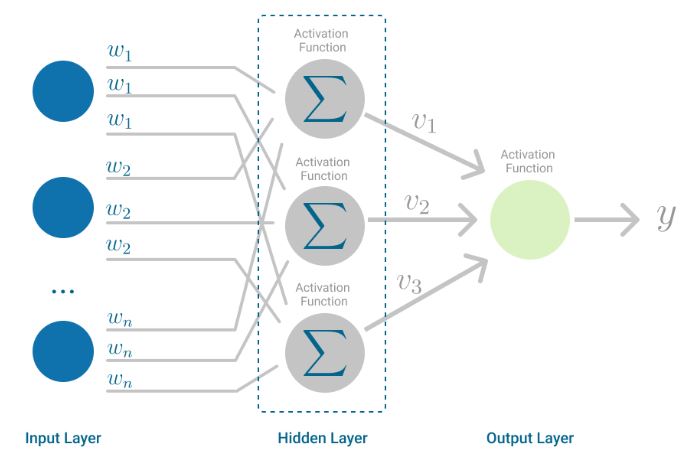
**CHEET SHEET FOR ACTIVATION FUNCTIONS:**

****

MULTI LAYER NEURAL NETWORK:

The**Multilayer Perceptron** was developed to tackle this limitation of linear data. It is a neural network where the mapping between inputs and output is non-linear.

A Multilayer Perceptron has input and output layers, and one or more **hidden layers** with many neurons stacked together. And while in the Perceptron the neuron must have an activation function that imposes a threshold, like ReLU or sigmoid, neurons in a Multilayer Perceptron can use any arbitrary activation function.



Multilayer Perceptron falls under the category of free-forward algorithms**,** because inputs are combined with the initial weights in a weighted sum and subjected to the activation function, just like in the Perceptron. But the difference is that each linear combination is propagated to the next layer.

Each layer is *feeding* the next one with the result of their computation, their internal representation of the data. This goes all the way through the hidden layers to the output layer.

But it has more to it.

If the algorithm only computed the weighted sums in each neuron, propagated results to the output layer, and stopped there, it wouldn’t be able to *learn* the weights that minimize the cost function. If the algorithm only computed one iteration, there would be no actual learning.

This is where [**Backpropagation**](https://en.wikipedia.org/wiki/Backpropagation) comes into play.

**Backpropagation:**

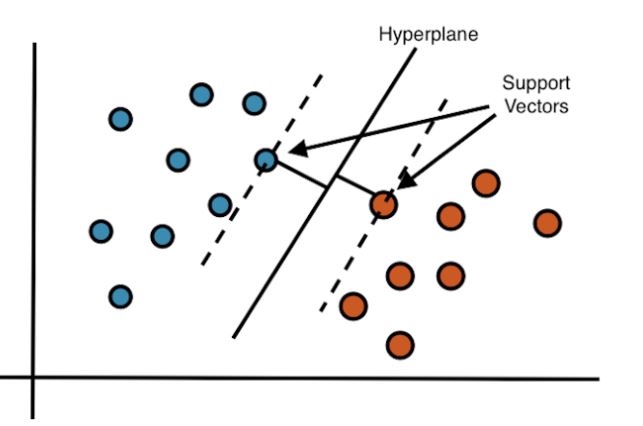
Backpropagation is the learning mechanism that allows the Multilayer Perceptron to iteratively adjust the weights in the network, with the goal of minimizing the cost function.

**SUPPORT VECTOR MACHINE:**

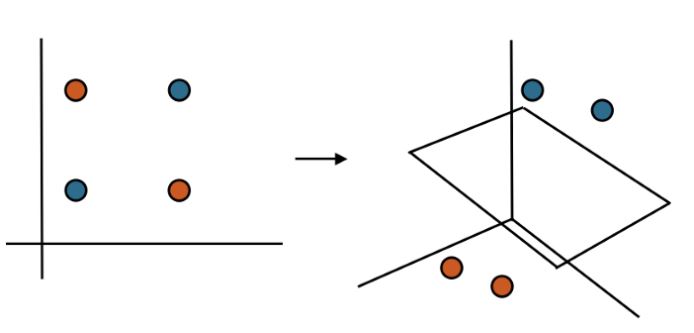
**Support Vector Machine is a supervised learning algorithm which identifies the best hyperplane to divide the dataset.**

SVMs are supervised machine learning models that are usually employed for classification (SVC — Support Vector Classification) or regression (SVR — Support Vector Regression) problems.

* **Support Vectors** — the points which are closest to the hyperplane
* **Hyperplane** — a subspace with dimension 1 lower than its ambient space. It serves to divide the space into multiple sections.  
  Given a 3-dimensional space, the subsequent hyperplane would be 2-dimensional plane. Similarly, in a 2-dimensional plane, the hyperplane would be a 1-dimensional line.
* **Margin** — the distance between the hyperplane and the nearest data point from either side
* **Kernel** — a mathematical function used to transform input data into a different form. Common kernel functions include *linear, nonlinear, polynomial, etc.*



We would want to choose a hyperplane with the greatest margin between the hyperplane and all points. This would yield the greatest likelihood of new data being correctly classified.



**When we can easily separate data with hyperplane by drawing a straight line is Linear SVM. When we cannot separate data with a straight line we use Non – Linear SVM. In this, we have Kernel functions. They transform non-linear spaces into linear spaces. It transforms data into another dimension so that the data can be classified.**

**It transforms two variables x and y into three variables along with z. Therefore, the data have plotted from 2-D space to 3-D space. Now we can easily classify the data by drawing the best hyperplane between them**.

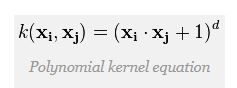
**A kernel helps to form the hyperplane in the higher dimension without raising the complexity.**

**KERNEL FUNCTIONS:**

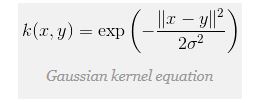
In machine learning, a kernel refers to a method that allows us to apply linear classifiers to non-linear problems by mapping non-linear data into a higher-dimensional space without the need to visit or understand that higher-dimensional space

IMPORTANT KERNEL FUNCTIONS:

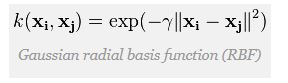
1.Polynomial



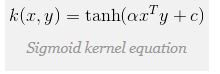
2.Gaussian



3.Laplace RBF

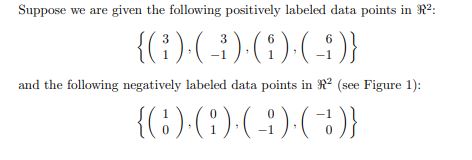


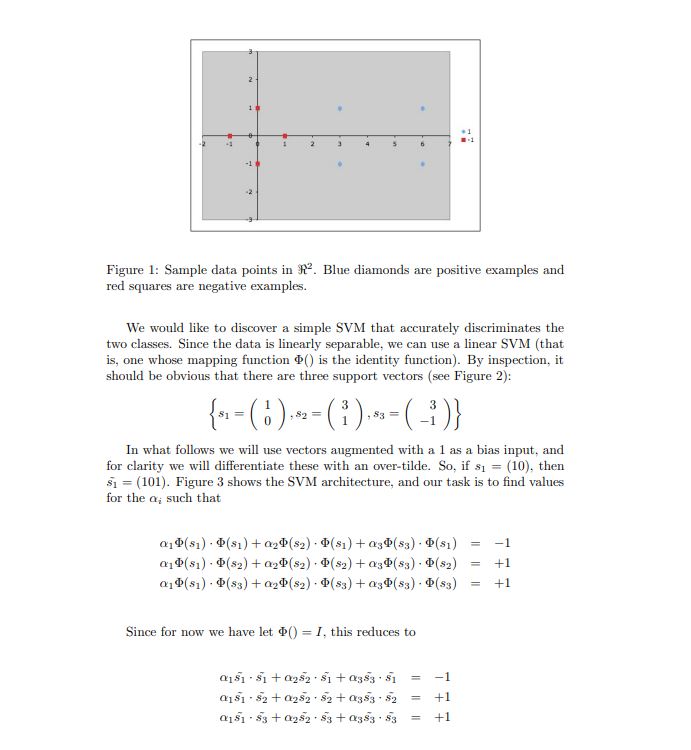
4.Sigmoid

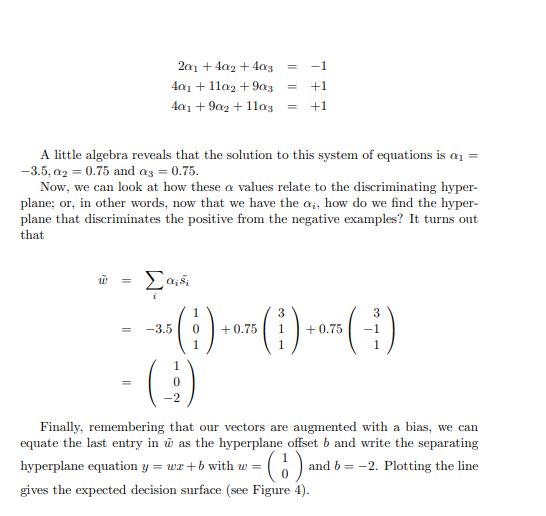


**PROBLEMS:**

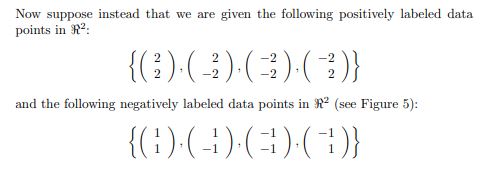
1. **LINEAR**

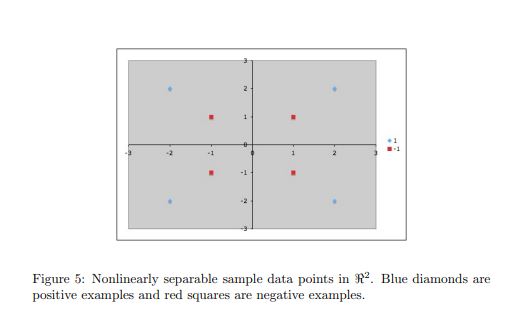


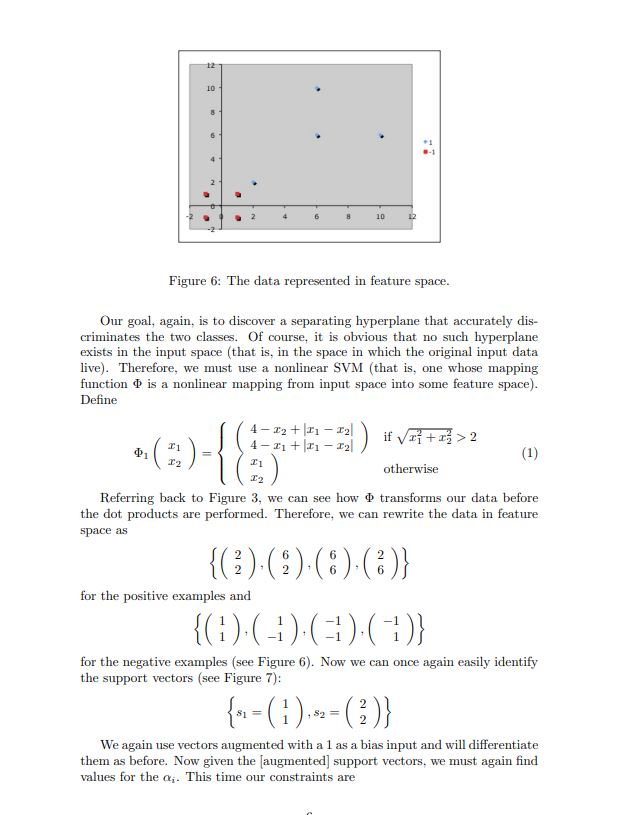


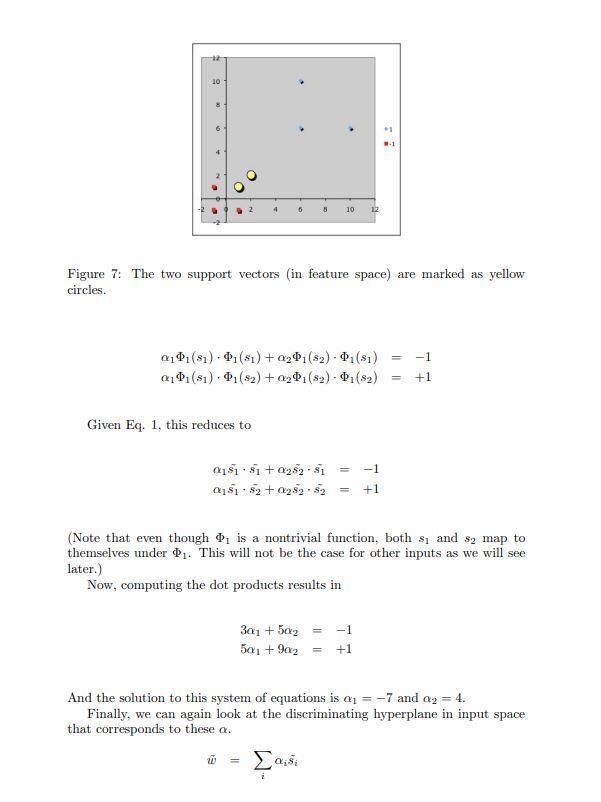


1. **NON-LINEAR**

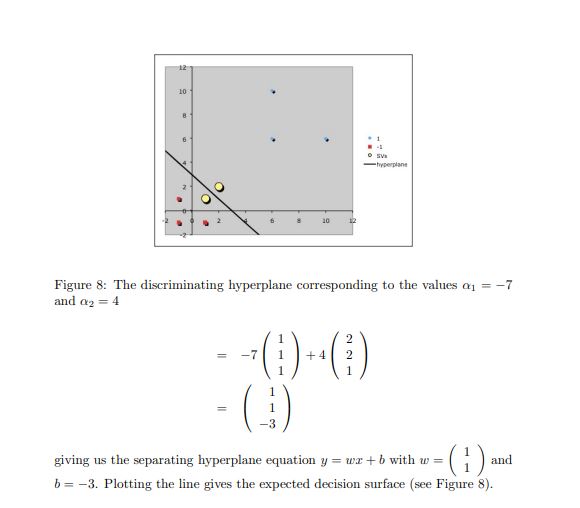








**NAÏVE BAYIES ALGORITHM:**



**K-NEAREST NEIGHBORS:**

(SOLVES BOTH REGRESSION AND CLASSIFICATION PROBLEMS)

**The KNN algorithm assumes that similar things exist in close proximity. In other words, similar things are near to each other.**

The K-NN working can be explained on the basis of the below algorithm:

* **Step-1:** Select the number K of the neighbors
* **Step-2:** Calculate the Euclidean distance of **K number of neighbors**
* **Step-3:** Take the K nearest neighbors as per the calculated Euclidean distance.
* **Step-4:** Among these k neighbors, count the number of the data points in each category.
* **Step-5:** Assign the new data points to that category for which the number of the neighbor is maximum.
* **Step-6:** Our model is ready.

**MODULE -4**

**(ENSEMBLE LEARNING)**

**Ensemble Learning Model Combination Schemes, Voting, Error-Correcting Output Codes, Bagging: Random Forest Trees, Boosting: Adaboost, Stacking**

**Ensemble Learning:**

**Ensemble learning is a general meta approach to machine learning that seeks better predictive performance by combining the predictions from multiple models.**

The three main classes of ensemble learning methods are **bagging**, **stacking**, and **boosting**,

* Bagging involves fitting many decision trees on different samples of the same dataset and averaging the predictions.
* Stacking involves fitting many different models’ types on the same data and using another model to learn how to best combine the predictions.
* Boosting involves adding ensemble members sequentially that correct the predictions made by prior models and outputs a weighted average of the predictions.

**BIAS:**

**Bias is the difference between the average prediction of our model and the correct value which we are trying to predict. Model with high bias pays very little attention to the training data and oversimplifies the model. It always leads to high error on training and test data.**

**VARIANCE:**

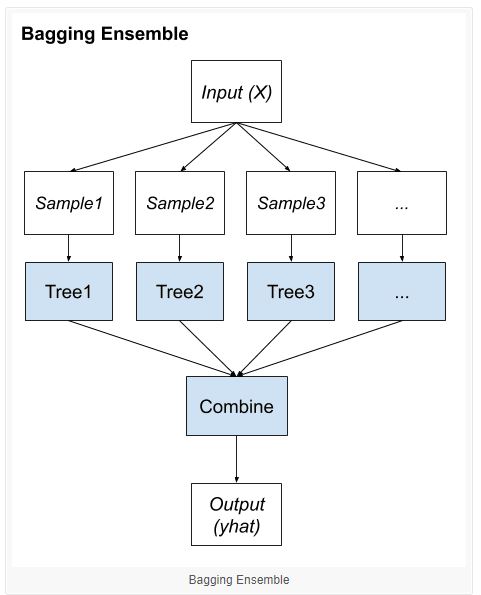
**Variance is the variability of model prediction for a given data point or a value which tells us spread of our data. Model with high variance pays a lot of attention to training data and does not generalize on the data which it hasn’t seen before. As a result, such models perform very well on training data but has high error rates on test data.**

**Bagging: (Bootstrap Aggregating)**

ensemble learning method that seeks a diverse group of ensemble members by varying the training data.

**Here the objective is to create several subsets of data from training sample chosen randomly with replacement. Each collection of subset data is used to train their decision trees.**

Ex: Random Forest



**As you can expect this helps us to reduce the variance error.**

**This technique is effective on models which tend to overfit on the dataset (high variance models). Bagging reduces the variance without making the predictions biased.**

**Bagging is a way to decrease the variance in the prediction by generating additional data for training from dataset using combinations with repetitions to produce multi-sets of the original data.**

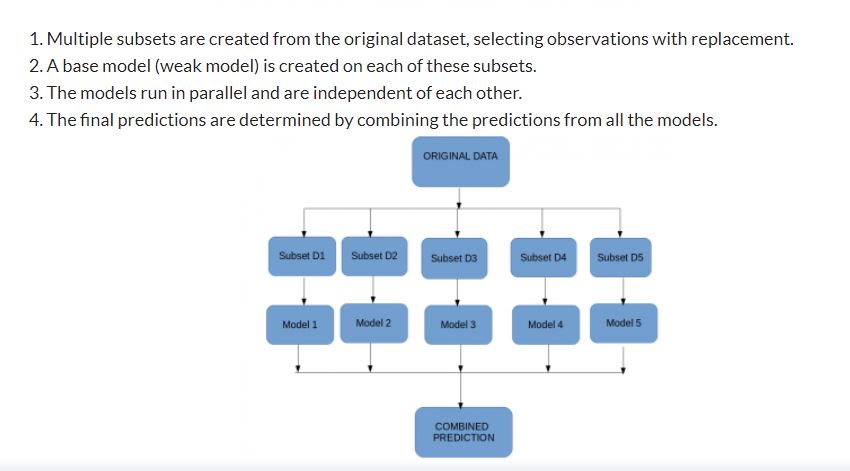
**Advantages:**

* **Reduces over-fitting of the model.**
* **Handles higher dimensionality data very well.**
* **Maintains accuracy for missing data.**

**Disadvantages:**

* **Since final prediction is based on the mean predictions from subset trees, it won’t give precise values for the classification and regression model.**

**STEPS:**

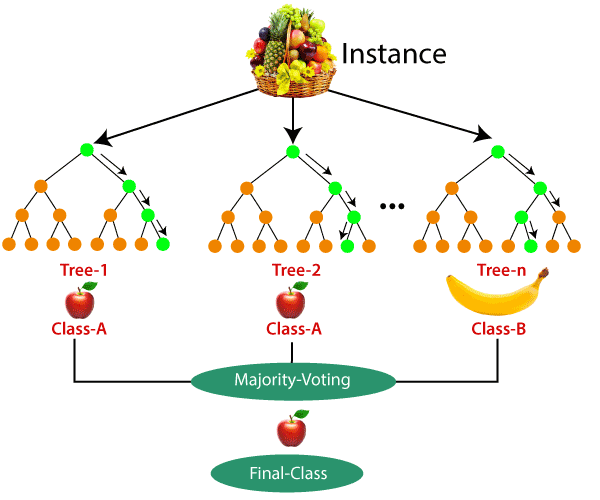
****

**RANDOM FOREST:**

**Random forest, like its name implies, consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model’s prediction.**

**Random Forests is a Machine Learning algorithm that tackles one of the biggest problems with**[**Decision Trees**](https://towardsdatascience.com/decision-tree-classifier-explained-in-real-life-picking-a-vacation-destination-6226b2b60575)**: variance**

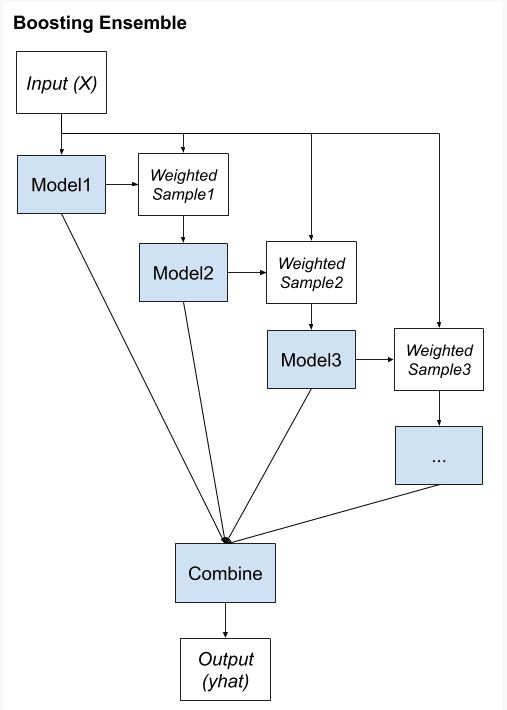
**Even though Decision Trees is simple and flexible, it is**[**greedy algorithm**](https://en.wikipedia.org/wiki/Greedy_algorithm)**. It focuses on optimizing for the node split at hand, rather than taking into account how that split impacts the entire tree. A greedy approach makes Decision Trees run faster, but makes it prone overfitting**.



**Boosting:**

[**Boosting**](https://machinelearningmastery.com/gradient-boosting-machine-ensemble-in-python/)**is an ensemble method that seeks to change the training data to focus attention on examples that previous fit models on the training dataset have gotten wrong.**

**In this technique, learners are learned sequentially with early learners fitting simple models to the data and then analysing data for errors. Consecutive trees (random sample) are fit and at every step, the goal is to improve the accuracy from the prior tree. When an input is misclassified by a hypothesis, its weight is increased so that next hypothesis is more likely to classify it correctly. This process converts weak learners into better performing model**

****

 The objective is to develop a so-called “strong-learner” from many purpose-built “weak-learners.”

Boosting is an iterative technique which adjusts the weight of an observation based on the last classification.

EXAMPLE: ADABOOST, XGBOOST.

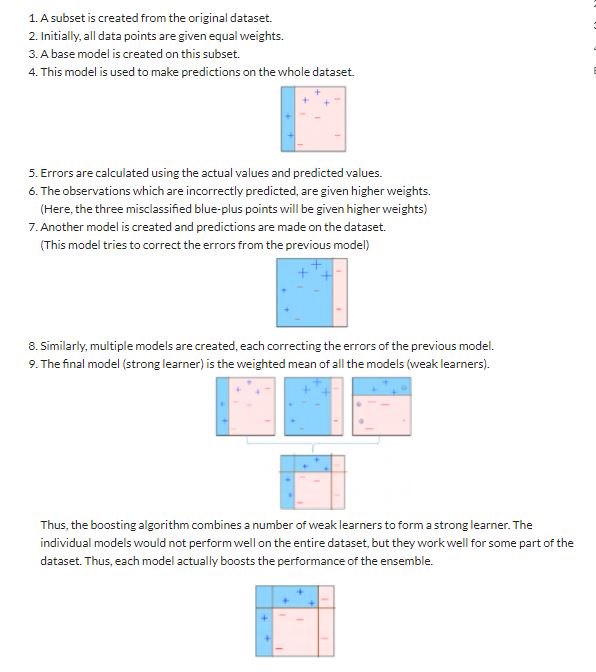
**Advantages:**

* Supports different loss function (we have used ‘binary: logistic’ for this example).
* Works well with interactions.

**Disadvantages:**

* Prone to over-fitting.
* Requires careful tuning of different hyper-parameters.

**STEPS:**

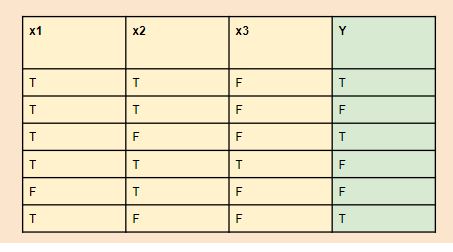


**PERFORMANCE:**

* Single Decision Tree: 82.843%
* Random Forest: 83.202%
* AdaBoost: 83.033%

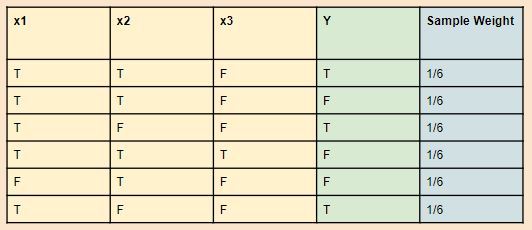
While the performance can be slightly improved with some additional hyperparameter optimization, the similarity of the results above tells us that we are fairly close to extracting the maximum information contained within the features used.

**ADABOOST – (PROBLEM)**



**Step 1: Assign a sample weight for each sample**

Using the equation above, calculate the sample weight for each sample. For the first round, the sample weight will be equal. In this example, the sample weight for each sample will be equal to 1/6.

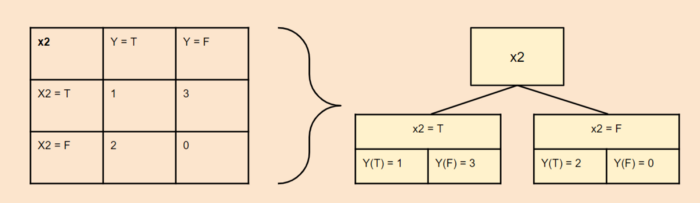


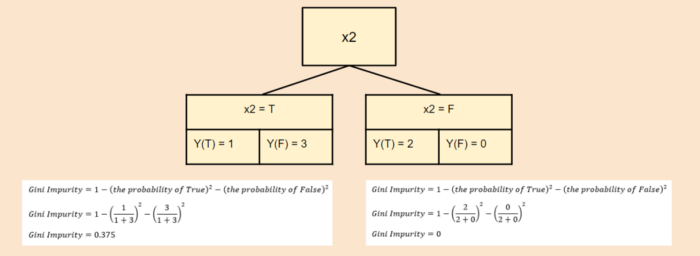
**Step 2: Calculate the Gini Impurity for each variable**

The next step is to calculate the Gini Impurity for each variable. The formula to calculate the Gini Impurity of each node is as follows:

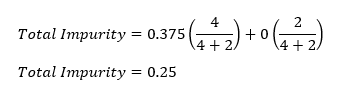


Once you calculate the Gini Impurity of each node, the total Gini Impurity for each variable is the weighted average of the impurities of each node.





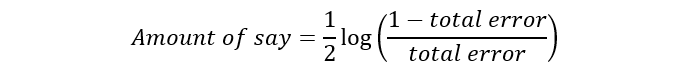
Once the Gini Impurity is calculated for each leaf node, the Total Gini Impurity can be calculated by taking the weighted average of the two individual impurities.



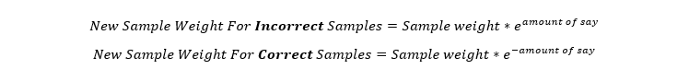
If you do that for each variable, you’ll get that x2 has the lowest Gini Impurity, so x2 will be used to create the first stump.

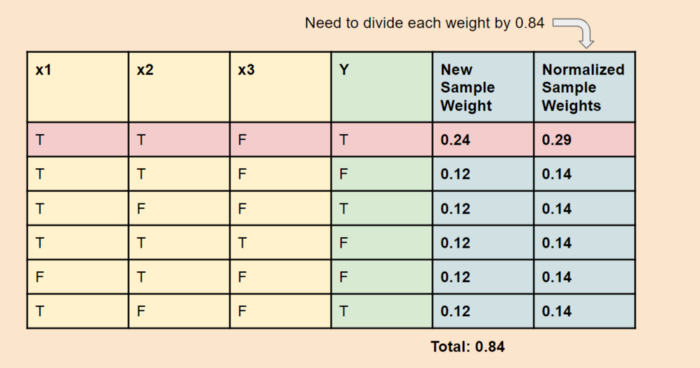
**Step 3: Calculate the Amount of Say for the stump that was created**

Total Error is equal to the sum of the weights of the incorrectly classified samples. Since one of the samples was incorrectly classified for x2, the total error is equal to 1/6.



**Step 4: Calculate the new sample weights for the next stump**

****

****

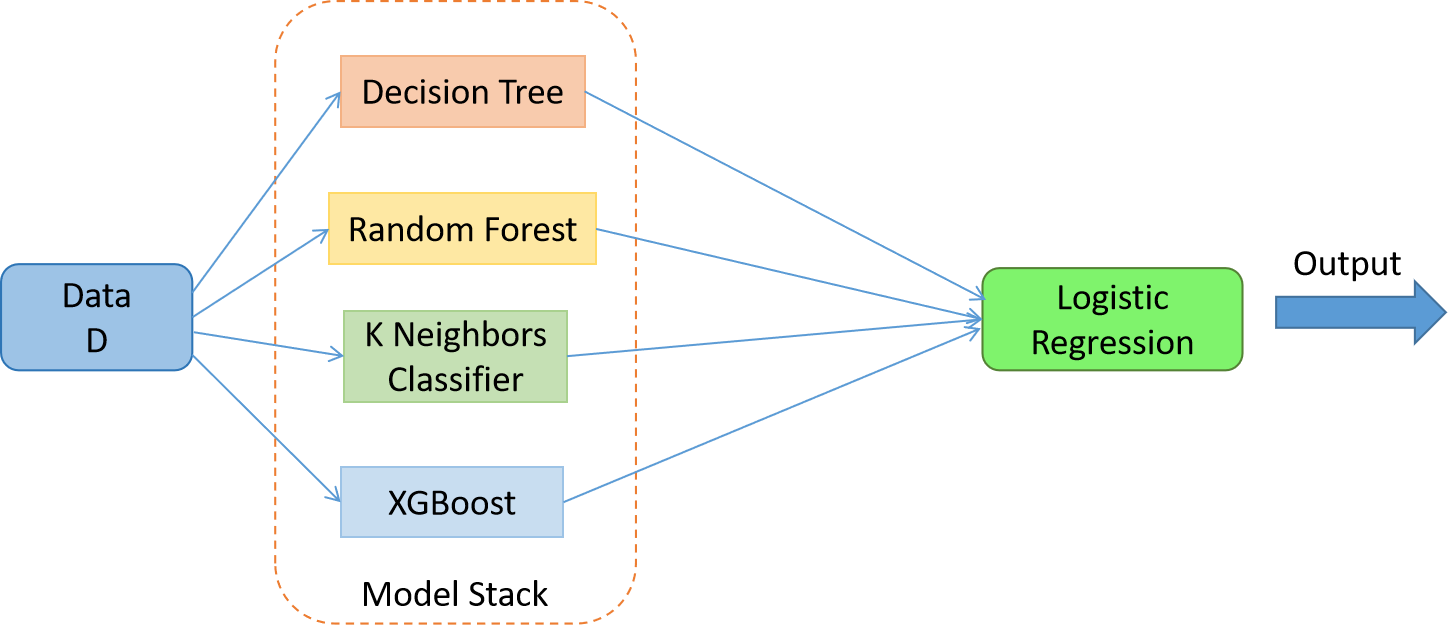
**Step 5: Create a bootstrapped dataset with the odds of each sample being chosen based on their new sample weights.**

**In this step, we’re going to randomly choose 6 samples with replacement from the dataset, with the odds of picking each based on their new sample weight.**

**Notice how the one that was incorrectly classified has a weight that’s more than double the others. This means that it is more likely to be selected multiple times, and thus, the next stump will focus more on classifying the misclassified sample correctly. This is the power of AdaBoost!**

**STACKING:**

**In stacking, an algorithm takes the outputs of sub-models as input and attempts to learn how to best combine the input predictions to make a better output prediction.**

****

**MODULE-5**

**(UNSUPERVISED LEARNING -1)**

**Introduction to clustering, Hierarchical: AGNES, DIANA, Partitional: K-means clustering, K-Mode Clustering, Self-Organizing Map, Expectation Maximization, Gaussian Mixture Models**