**[CLASSIFICATION AND REGRESSION USING DIFFERENT ALGORITHMS]**



**TECHNOGLOBE, JAIPUR**

**CLASSIFICATION AND REGRESSION USING DIFFERENT ALGORITHMS**

Submitted in partial fulfilment of the requirements for the course in  
**Data Science**  
under  
**[ TechnoGlobe Jaipur ]**

**Submitted by**  
**SHUBHAM SHARMA**  
(RAJ/JPR/001/799)

**Under the guidance of**  
Mr. AJEET KUMAR YADAV

**Approval Certificate**

This is to certify that the project report entitled “**Classification and Regression Using Different Algorithms**” submitted by **SHUBHAM SHARMA** (Roll: **RAJ/JPR/001/799**) in partial fulfillment of the requirements of the course in **DATA SCIENCE** from **TECHNOGLOBE, Jaipur** was carried out in a systematic and procedural manner to the best of our knowledge. It is bona fide work of the candidate and was carried out under our supervision and guidance during the academic session.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Mr. Ajeet Kumar Yadav**

Project Guide

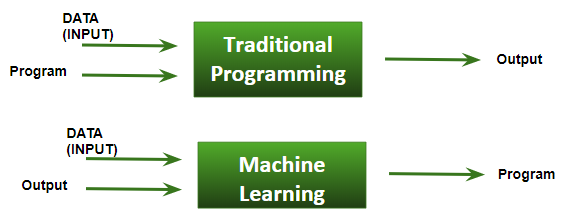
**ACKNOWLEDGEMENT**

The endless thanks go to Lord Almighty for all the blessings he has showered onto me, which has enabled me to write this last note in my research work. During the period of my research, as in the rest of my life, I have been blessed by Almighty with some extraordinary people who have spun a web of support around me. Words can never be enough in expressing how grateful I am to those incredible people in my life who made this thesis possible. I would like an attempt to thank them for making my time during my research in the Institute a period I will treasure. I am deeply indebted to my research supervisor, Sir Ajeet Kumar Yadav. Each meeting with him added in valuable aspects to the implementation and broadened my perspective. He has guided me with his invaluable suggestions, lightened up the way in my darkest times and encouraged me a lot in the academic life.

SHUBHAM SHARMA

**1 What is Machine Learning?**

**Machine Learning is the field of study that gives computers the capability to learn without being explicitly programmed.** Machine Learning (ML) is automating and improving the learning process of computers based on their experiences without being actually programmed i.e. without any human assistance. The process starts with feeding good quality data and then training our machines(computers) by building machine learning models using the data and different algorithms. The choice of algorithms depends on what type of data do we have and what kind of task we are trying to automate.



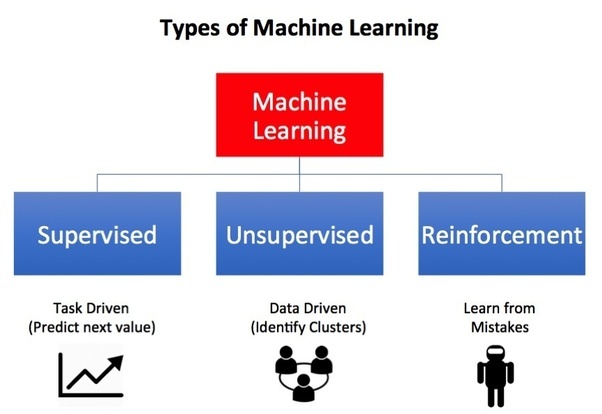
**1.2 What are types of Machine Learning?**

* **Supervised learning:** When an algorithm learns from example data and associated target responses that can consist of numeric values or string labels, such as classes or tags, in order to later predict the correct response when posed with new examples comes under the category of Supervised learning. This approach is indeed similar to human learning under the supervision of a teacher. The teacher provides good examples for the student to memorize, and the student then derives general rules from these specific examples.
* **Unsupervised learning:** Whereas when an algorithm learns from plain examples without any associated response, leaving to the algorithm to determine the data patterns on its own. This type of algorithm tends to restructure the data into something else, such as new features that may represent a class or a new series of un-correlated values. They are quite useful in providing humans with insights into the meaning of data and new useful inputs to supervised machine learning algorithms.

As a kind of learning, it resembles the methods humans use to figure out that certain objects or events are from the same class, such as by observing the degree of similarity between objects. Some recommendation systems that you find on the web in the form of marketing automation are based on this type of learning.

* **Reinforcement learning:** When you present the algorithm with examples that lack labels, as in unsupervised learning. However, you can accompany an example with positive or negative feedback according to the solution the algorithm proposes comes under the category of Reinforcement learning, which is connected to applications for which the algorithm must make decisions (so the product is prescriptive, not just descriptive, as in unsupervised learning), and the decisions bear consequences. In the human world, it is just like learning by trial and error.

Errors help you learn because they have a penalty added (cost, loss of time, regret, pain, and so on), teaching you that a certain course of action is less likely to succeed than others. An interesting example of reinforcement learning occurs when computers learn to play video games by themselves.



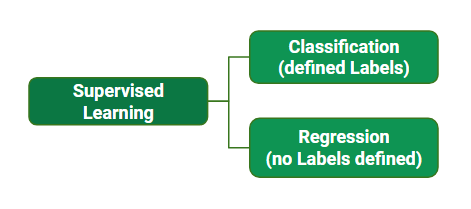
**2. SUPERVISED LEARNING:**

Supervised learning is when the model is getting trained on a labelled dataset. Labelled dataset is one which have both input and output parameters. In this type of learning both training and validation datasets are labelled.

**Training the system:**

While training the model, data is usually split in the ratio of 80:20 i.e. 80% as training data and rest as testing data. In training data, we feed input as well as output for 80% data. The model learns from training data only. We use different machine learning algorithms to build our model. By learning, it means that the model will build some logic of its own.

Once the model is ready then it is good to be tested. At the time of testing, input is fed from remaining 20% data which the model has never seen before, the model will predict some value and we will compare it with actual output and calculate the accuracy.



**2.1 Classification:**

It is a Supervised Learning task where output is having defined labels (discrete value).

The goal here is to predict discrete values belonging to a particular class and evaluate on the basis of accuracy.

It can be either binary or multi class classification. In binary classification, model predicts either 0 or 1; yes or no but in case of multi class classification, model predicts more than one class.

**Example:** Gmail classifies mails in more than one classes like social, promotions, updates, forum.

**2.2 Regression:**

It is a Supervised Learning task where output is having continuous value.

The goal here is to predict a value as much closer to actual output value as our model can and then evaluation is done by calculating error value. The smaller the error the greater the accuracy of our regression model.

**Example of Supervised Learning Algorithms:**

* Linear Regression
* Nearest Neighbor
* Decision Trees
* Support Vector Machine (SVM)
* Random Forest

**3. UNSUPERVISED LEARNING:**

Unsupervised learning is the training of machine using information that is neither classified nor labeled and allowing the algorithm to act on that information without guidance. Here the task of machine is to group unsorted information according to similarities, patterns and differences without any prior training of data.

Unlike supervised learning, no teacher is provided that means no training will be given to the machine. Therefore, machine is restricted to find the hidden structure in unlabeled data by our-self.

For instance, suppose there is given an image having both dogs and cats which have not seen ever.

Thus, the machine has no idea about the features of dogs and cat so we can’t categorize it in dogs and cats. But it can categorize them according to their similarities, patterns, and differences i.e., we can easily categorize the above picture into two parts. First may contain all pics having dogs in it and second part may contain all pics having cats in it. Here you didn’t learn anything before, means no training data or examples.

Unsupervised learning classified into two categories of algorithms:

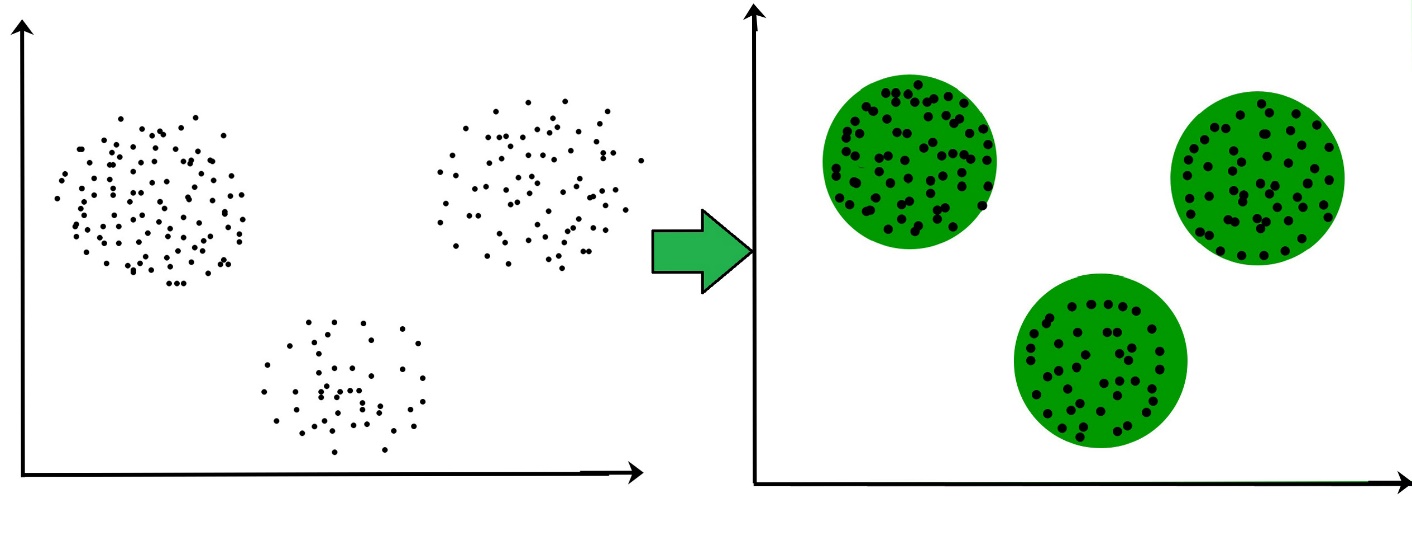
**Clustering:** A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behavior.

**Association:** An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy X also tend to buy Y.

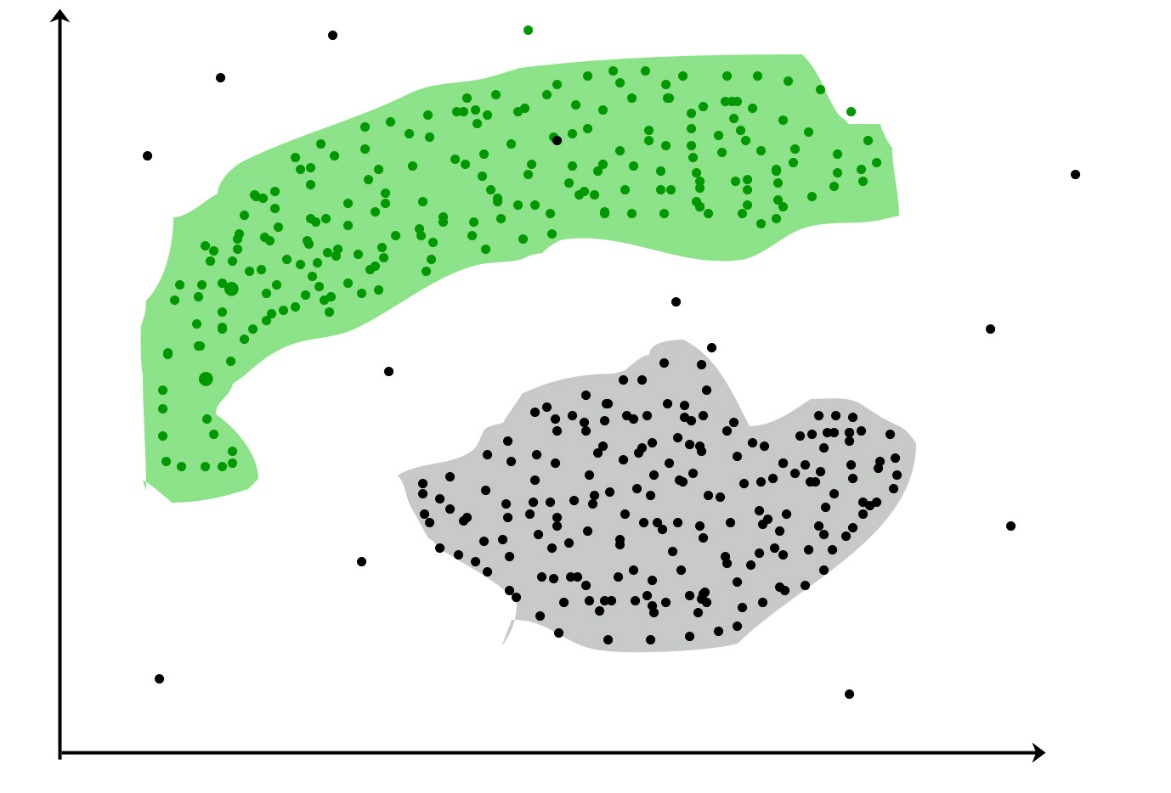
**3.1 Clustering:**

Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group and dissimilar to the data points in other groups. It is basically a collection of objects on the basis of similarity and dissimilarity between them.

**Example**– The data points in the graph below clustered together can be classified into one single group. We can distinguish the clusters, and we can identify that there are 3 clusters in the below picture.



It is not necessary for clusters to be a spherical. Such as:



These data points are clustered by using the basic concept that the data point lies within the given constraint from the cluster center. Various distance methods and techniques are used for calculation of the outliers.

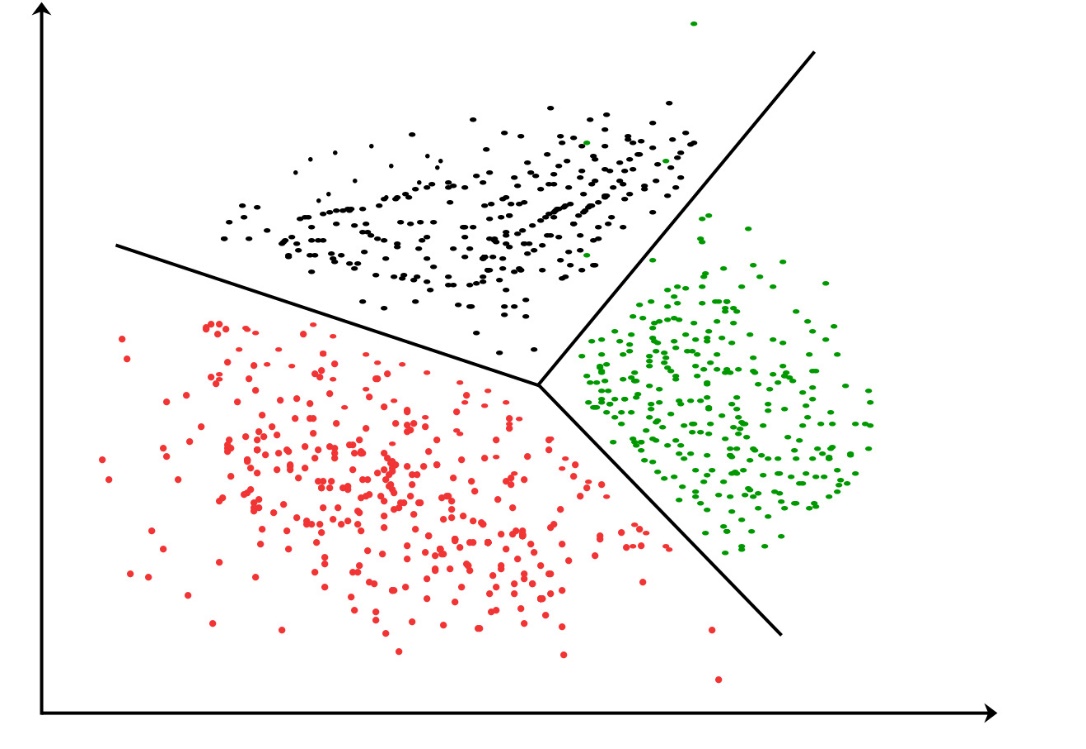
**Why Clustering?**

Clustering is very much important as it determines the intrinsic grouping among the unlabeled data present. There are no criteria for a good clustering. It depends on the user, what is the criteria they may use which satisfy their need.

For instance, we could be interested in finding representatives for homogeneous groups (data reduction), in finding “natural clusters” and describe their unknown properties (“natural” data types), in finding useful and suitable groupings (“useful” data classes) or in finding unusual data objects (outlier detection). This algorithm must make some assumptions which constitute the similarity of points and each assumption make different and equally valid clusters.

**Clustering Algorithms:**

**K-means clustering algorithm** – It is the simplest unsupervised learning algorithm that solves clustering problem. K-means algorithm partition n observations into k clusters where each observation belongs to the cluster with the nearest mean serving as a prototype of the cluster.



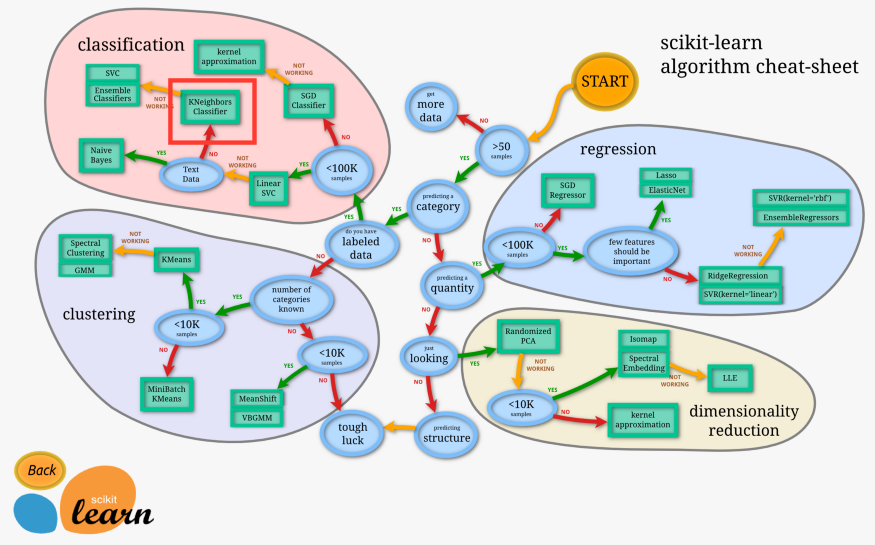
The algorithm will categorize the items into k groups of similarity. To calculate that similarity, we will use the Euclidean distance as measurement.

The algorithm works as follows:

* First, we initialize k points, called means, randomly.
* We categorize each item to its closest mean and we update the mean’s coordinates, which are the averages of the items categorized in that mean so far.
* We repeat the process for a given number of iterations and at the end, we have our clusters.

**4.Rienforecement Learning:**

Reinforcement learning is an area of Machine Learning. Reinforcement. It is about taking suitable action to maximize reward in a particular situation. It is employed by various software and machines to find the best possible behavior or path it should take in a specific situation. Reinforcement learning differs from the supervised learning in a way that in supervised learning the training data has the answer key with it so the model is trained with the correct answer itself whereas in reinforcement learning, there is no answer but the reinforcement agent decides what to do to perform the given task. In the absence of training dataset, it is bound to learn from its experience.



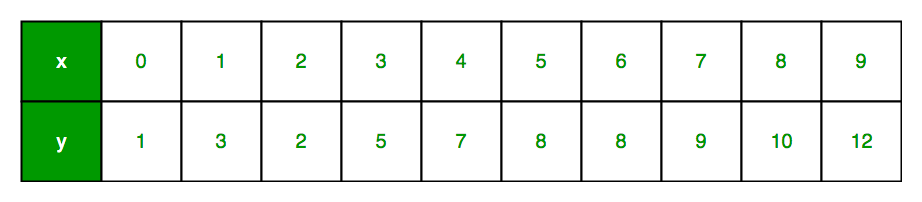
**5. LINEAR REGRESSION – ALGORITHM**

Linear Regression is a machine learning algorithm based on supervised learning. It performs a regression task. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting. Different regression models differ based on – the kind of relationship between dependent and independent variables, they are considering and the number of independent variables being used.

Simple linear regression is an approach for predicting a response using a single feature.

It is assumed that the two variables are linearly related. Hence, we try to find a linear function that predicts the response value(y) as accurately as possible as a function of the feature or independent variable(x).

Let us consider a dataset where we have a value of response y for every feature x:

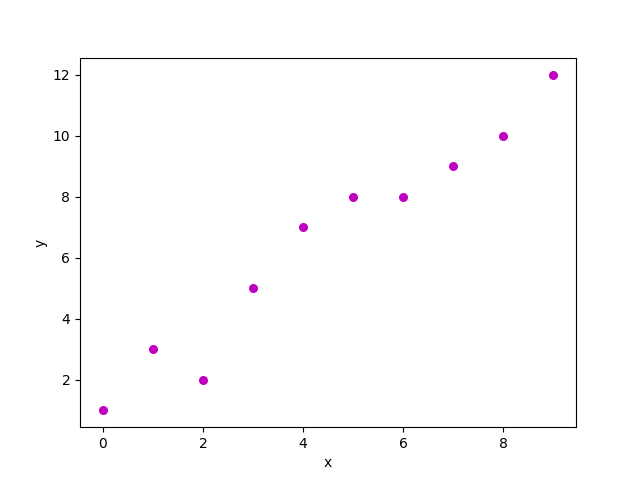


For generality, we define:

* x as feature vector, i.e x = [x\_1, x\_2, …., x\_n],
* y as response vector, i.e y = [y\_1, y\_2, …., y\_n]

for n observations (in above example, n=10).

A scatter plot of above dataset looks like: -



Now, the task is to find a line which fits best in above scatter plot so that we can predict the response for any new feature values. (i.e a value of x not present in dataset).

This line is called regression line.

The equation of regression line is represented as:



Here,

* h(x\_i) represents the predicted response value for ith observation.
* b\_0 and b\_1 are regression coefficients and represent y-intercept and slope of regression line respectively.

To create our model, we must “learn” or estimate the values of regression coefficients b\_0 and b\_1. And once we’ve estimated these coefficients, we can use the model to predict responses!

Now consider:



Here, e\_i is residual error in ith observation.

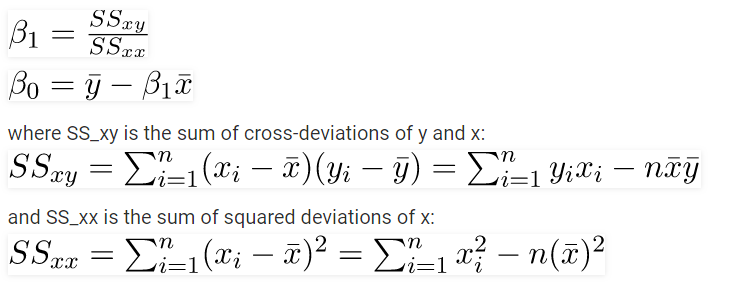
So, our aim is to minimize the total residual error.

We define the squared error or cost function, J as:



and our task is to find the value of b\_0 and b\_1 for which J(b\_0,b\_1) is minimum!

Without going into the mathematical details, we present the result here:



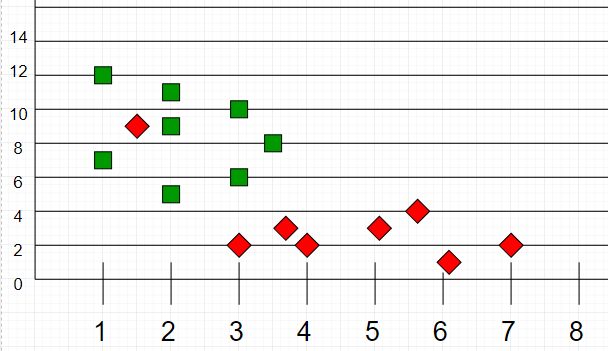
**6. K- NEAREST NEIGHBOR CLASSIFICATION:**

K-Nearest Neighbors is one of the most basic yet essential classification algorithms in Machine Learning. It belongs to the supervised learning domain and finds intense application in pattern recognition, data mining and intrusion detection.

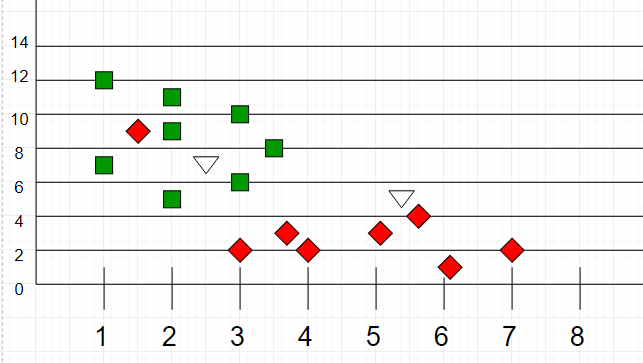
It is widely disposable in real-life scenarios since it is non-parametric, meaning, it does not make any underlying assumptions about the distribution of data (as opposed to other algorithms such as GMM, which assume a Gaussian distribution of the given data).

We are given some prior data (also called training data), which classifies coordinates into groups identified by an attribute.

As an example, consider the following table of data points containing two features:



Now, given another set of data points (also called testing data), allocate these points a group by analyzing the training set. Note that the unclassified points are marked as ‘White’.



**Intuition**

If we plot these points on a graph, we may be able to locate some clusters or groups. Now, given an unclassified point, we can assign it to a group by observing what group its nearest neighbors belong to. This means a point close to a cluster of points classified as ‘Red’ has a higher probability of getting classified as ‘Red’.

Intuitively, we can see that the first point (2.5, 7) should be classified as ‘Green’ and the second point (5.5, 4.5) should be classified as ‘Red’.

**Algorithm**

Let m be the number of training data samples. Let p be an unknown point.

* Store the training samples in an array of data points arr[]. This means each element of this array represents a tuple (x, y).
* for i=0 to m:

Calculate Euclidean distance d(arr[i], p).

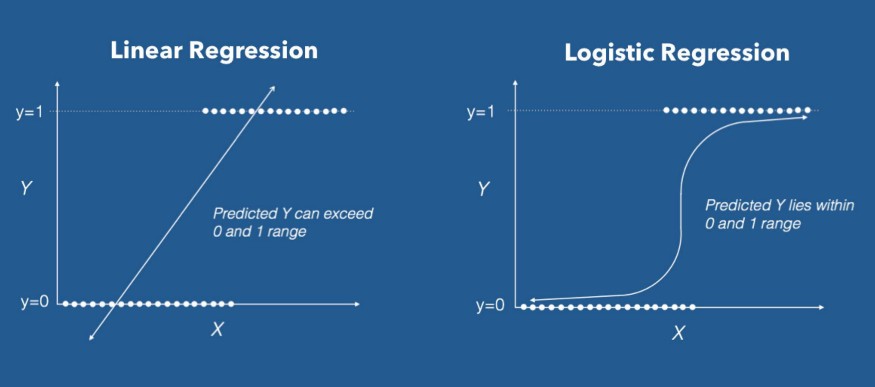
* Make set S of K smallest distances obtained. Each of these distances corresponds to an already classified data point.
* Return the majority label among S.

**7. LOGISTIC REGRESSION ALGORITHM:**

Like all regression analyses, the logistic regression is a predictive analysis. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables.

Here, why it is called regression in classification? because Regression means it predicts the probability i.e. also a value that will use in classification. Ex good or bad, pass or fail etc.

Logistic Regression is a Machine Learning algorithm which is used for the classification problems, it is a predictive analysis algorithm and based on the concept of probability.



We can call a Logistic Regression a Linear Regression model but the Logistic Regression uses a more complex cost function, this cost function can be defined as the **‘Sigmoid function’** or also known as the ‘logistic function’ instead of a linear function.

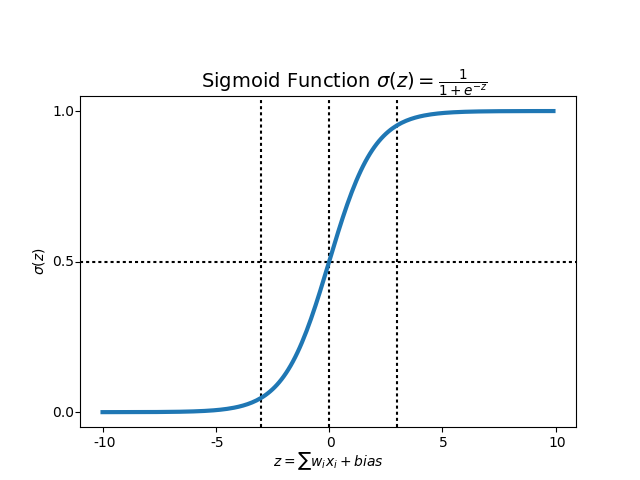
The hypothesis of logistic regression tends it to limit the cost function between 0 and 1. Therefore linear functions fail to represent it as it can have a value greater than 1 or less than 0 which is not possible as per the hypothesis of logistic regression.

https://miro.medium.com/max/279/1*GnceHPIeThNShGSmYzE4eA.png

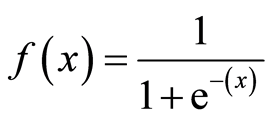
Logistic regression hypothesis expectation

**What is the Sigmoid Function?**

In order to map predicted values to probabilities, we use the Sigmoid function. The function maps any real value into another value between 0 and 1. In machine learning, we use sigmoid to map predictions to probabilities.



Sigmoid Function Graph

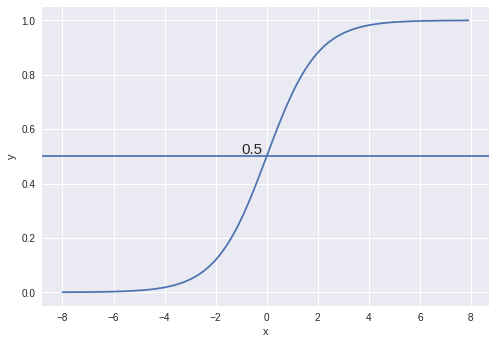


Formula of a sigmoid function

**Decision Boundary**

We expect our classifier to give us a set of outputs or classes based on probability when we pass the inputs through a prediction function and returns a probability score between 0 and 1.

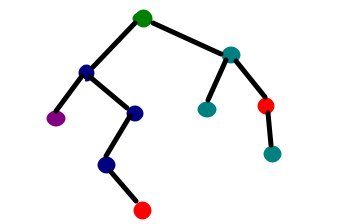
For Example, We have 2 classes, let’s take them like cats and dogs(1 — dog , 0 — cats). We basically decide with a threshold value above which we classify values into Class 1 and of the value goes below the threshold then we classify it in Class 2.



As shown in the above graph we have chosen the threshold as 0.5, if the prediction function returned a value of 0.7 then we would classify this observation as Class 1(DOG). If our prediction returned a value of 0.2 then we would classify the observation as Class 2(CAT).

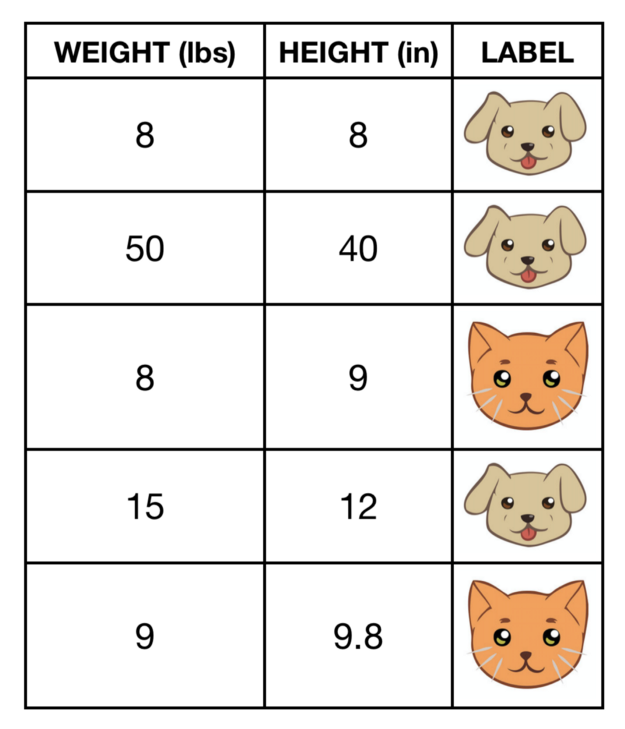
**8. DESCISION TREE ALGORITHM:**

Decision Tree is one of the most powerful and popular algorithms. Decision-tree algorithm falls under the category of supervised learning algorithms. It works for both continuous as well as categorical output variables. Our brain works like a decision tree every time we ask ourselves a question before making a decision. For example: is it cloudy outside? If yes, I will bring an umbrella.

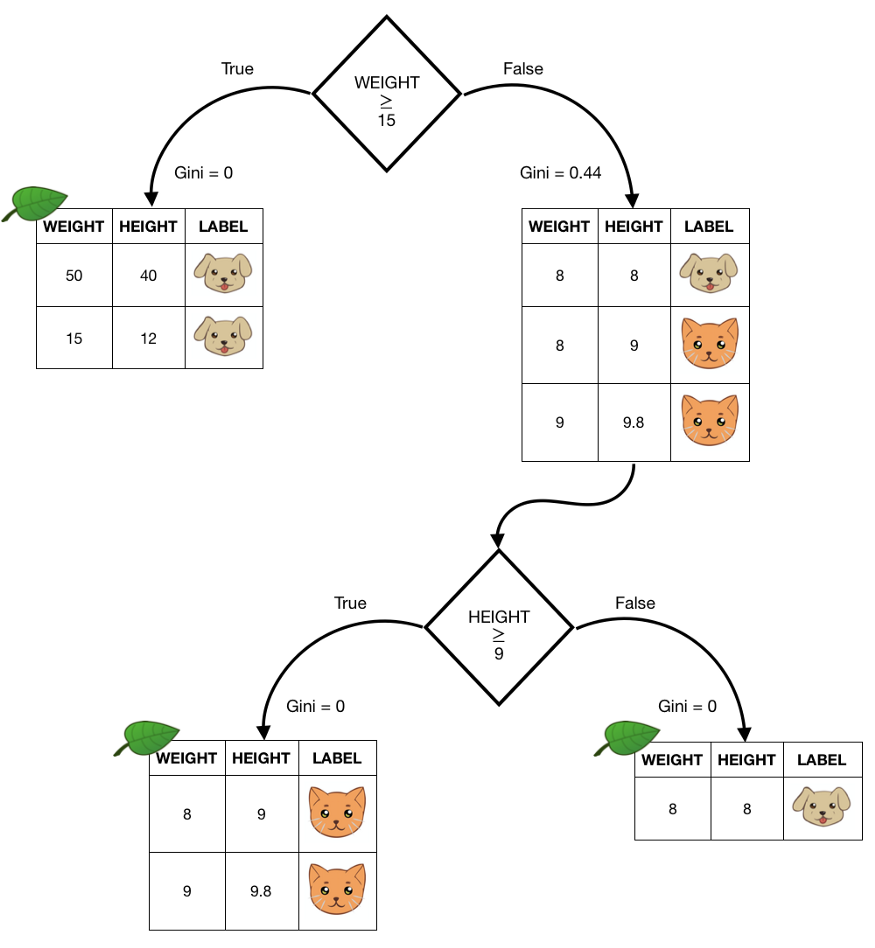


When training a dataset to classify a variable, the idea of the Decision Tree is to divide the data into smaller datasets based on a certain feature value until the target variables all fall under one category. While the human brain decides to pick the “splitting feature” based on the experience (i.e. the cloudy sky), a computer splits the dataset based on the maximum information gain. Let’s define a simple problem and jump into some calculations to see exactly what this means!

Let’s say we want to build a decision tree to determine whether a pet is a cat or a dog based on weight and height. We could divide these data points based on certain values of one of the two characteristics, for example: for a weight greater than 15 lbs, I am sure that the pet is a dog, at least based on this very simplistic dataset. But if the weight is less than that, my subset will contain two cats and one dog, so I will need to split the subset again, until there is only one class left. In other words, until all my subsets are pure. I actually drew a decision tree for these data values, here is what it looks like:

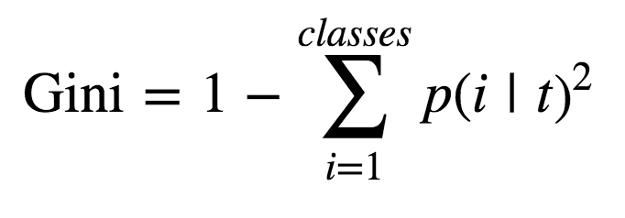


Cats and dogs dataset



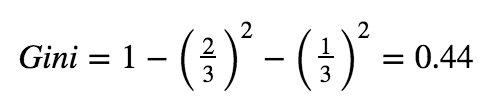
Decision tree example.

Every tree starts with a root node, i.e. the first split. Without thinking too much, we can easily come up with different types of root nodes to split the data reaching a solution in the same number of steps. But how would a computer decide how to define the nodes? For sure, it is going to look for the most efficient way and in order to understand what that is, we need to introduce Gini: “the most commonly used measurement of inequality”. This inequality refers to the target classes in each subset following a node. For this reason, it can be calculated after each split, and depending on how the inequality changes after a node, we can also define the “information gain”.



Definition of Gini

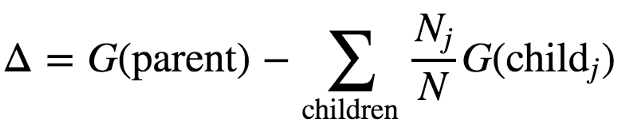
To calculate Gini, we consider the probability of finding each class after a node, we sum the square of those values and we subtract this amount from 1. For this reason, when a subset is pure (i.e. there is only one class in it), Gini will be 0, because the probability of finding that class is 1 indeed! And in that case, we say we have reached a leaf, because there is no need to split anymore as we achieved our goal. But if we look at the picture above, after the root node in the False case, we have a subset with 3 observations, 2 of which are cats and 1 is dog. If we want to calculate the Gini for that subset, we have:



Gini of the resulting dataset for weight greater or equal to 15 lbs

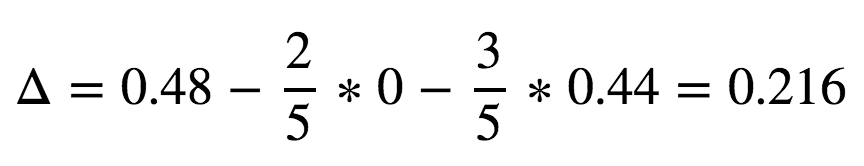
Another metric, instead of Gini, that can be used to calculate the inequality of the classes, is Entropy. They have the same purpose but Entropy varies on a slightly different scale; however, for this purpose we will only be using Gini.

Depending on the splitting strategy that we choose, we will have different values of Gini for each subset, and depending on the Gini value after a node, we can define the Information Gain:



Definition of Information Gain

this is defined as the difference between the Gini of the parent, and the weighted average of the children’s Gini. If we refer to the example above, knowing that the Gini of the initial dataset is equal to 0.48 by simply applying the definition, we can calculate the information gain after the root node (splitting at weight value of 15 lbs’):



Information gain after root node

The decision tree will then consider all the possible splits and choose the one with the highest information gain.

**9. SUPPORT VECTOR MACHINE ALGORITHM:**

“Support Vector Machine” (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiate the two classes very well.



Support Vectors are simply the co-ordinates of individual observation. Support Vector Machine is a frontier which best segregates the two classes (hyper-plane/ line).

**How does it work?**

Now the burning question is “How can we identify the right hyper-plane?”

* **Identify the right hyper-plane (Scenario-1):** Here, we have three hyper-planes (A, B and C). Now, identify the right hyper-plane to classify star and circle.



Remember a thumb rule to identify the right hyper-plane: “Select the hyper-plane which segregates the two classes better”. In this scenario, hyper-plane “B” has excellently performed this job.

* **Identify the right hyper-plane (Scenario-2):** Here, we have three hyper-planes (A, B and C) and all are segregating the classes well. Now, How can we identify the right hyper-plane?



Here, maximizing the distances between nearest data point (either class) and hyper-plane will help us to decide the right hyper-plane. This distance is called as Margin.



Above, you can see that the margin for hyper-plane C is high as compared to both A and B. Hence, we name the right hyper-plane as C. Another lightning reason for selecting the hyper-plane with higher margin is robustness. If we select a hyper-plane having low margin then there is high chance of miss-classification.

* **Identify the right hyper-plane (Scenario-3):** Hint: Use the rules as discussed in previous section to identify the right hyper-plane

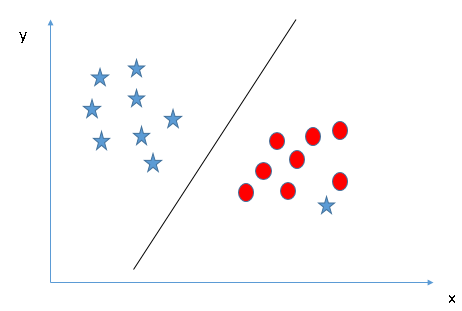


Some of you may have selected the hyper-plane B as it has higher margin compared to A. But, here is the catch, SVM selects the hyper-plane which classifies the classes accurately prior to maximizing margin. Here, hyper-plane B has a classification error and A has classified all correctly. Therefore, the right hyper-plane is A.

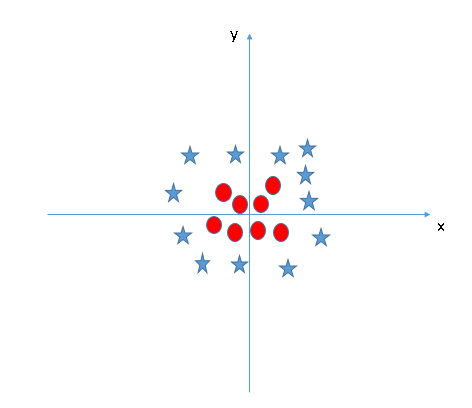
* **Can we classify two classes (Scenario-4)?:** Below, I am unable to segregate the two classes using a straight line, as one of star lies in the territory of other(circle) class as an outlier.



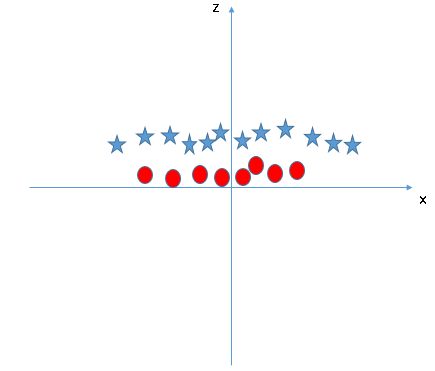
As I have already mentioned, one star at other end is like an outlier for star class. SVM has a feature to ignore outliers and find the hyper-plane that has maximum margin. Hence, we can say, SVM is robust to outliers.



* **Find the hyper-plane to segregate to classes (Scenario-5):** In the scenario below, we can’t have linear hyper-plane between the two classes, so how does SVM classify these two classes? Till now, we have only looked at the linear hyper-plane.



SVM can solve this problem. Easily! It solves this problem by introducing additional feature. Here, we will add a new feature z=x^2+y^2. Now, let’s plot the data points on axis x and z:

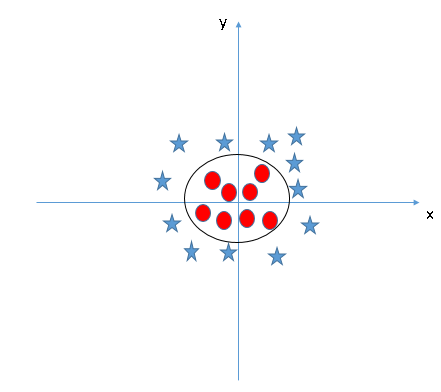


In above plot, points to consider are:

* All values for z would be positive always because z is the squared sum of both x and y
* In the original plot, red circles appear close to the origin of x and y axes, leading to lower value of z and star relatively away from the origin result to higher value of z.

In SVM, it is easy to have a linear hyper-plane between these two classes. But, another burning question which arises is, should we need to add this feature manually to have a hyper-plane. No, SVM has a technique called the kernel trick. These are functions which takes low dimensional input space and transform it to a higher dimensional space i.e. it converts not separable problem to separable problem, these functions are called kernels. It is mostly useful in non-linear separation problem. Simply put, it does some extremely complex data transformations, then find out the process to separate the data based on the labels or outputs you’ve defined.

When we look at the hyper-plane in original input space it looks like a circle:



**10. K-FOLD CROSS VALIDATION ALGORITHM:**

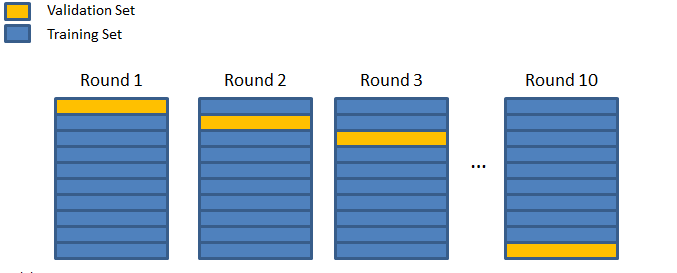
There are different types of Cross Validation Techniques but the overall concept remains the same,

• To partition the data into a number of subsets

• Hold out a set at a time and train the model on remaining set

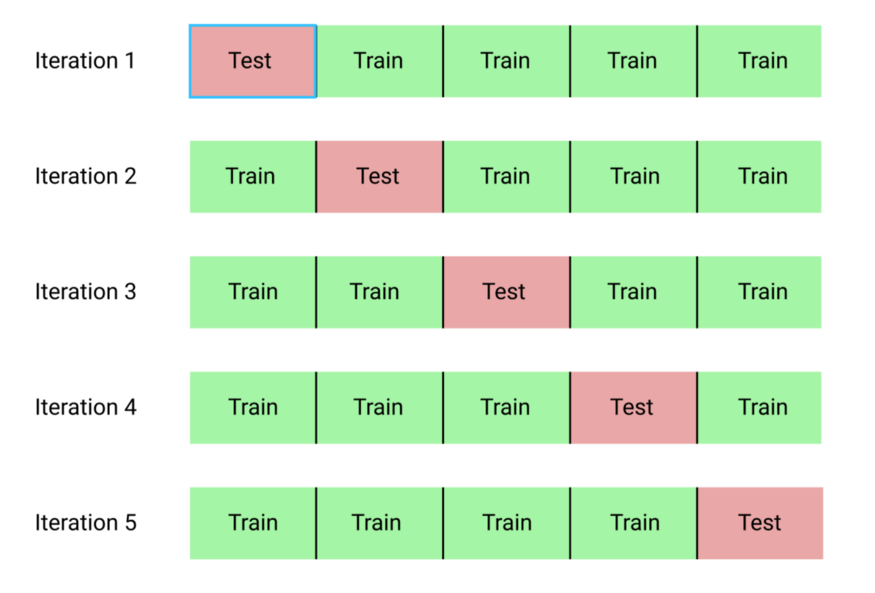
• Test model on hold out set

Repeat the process for each subset of the dataset



the process of cross validation in general

**k-Fold Cross Validation:**



The procedure has a single parameter called k that refers to the number of groups that a given data sample is to be split into. As such, the procedure is often called k-fold cross-validation. When a specific value for k is chosen, it may be used in place of k in the reference to the model, such as k=10 becoming 10-fold cross-validation.

If k=5 the dataset will be divided into 5 equal parts and the below process will run 5 times, each time with a different holdout set.

1. Take the group as a holdout or test data set

2. Take the remaining groups as a training data set

3. Fit a model on the training set and evaluate it on the test set

4. Retain the evaluation score and discard the model

At the end of the above process Summarize the skill of the model using the sample of model evaluation scores.

**How to decide the value of k?**

The value for k is chosen such that each train/test group of data samples is large enough to be statistically representative of the broader dataset.

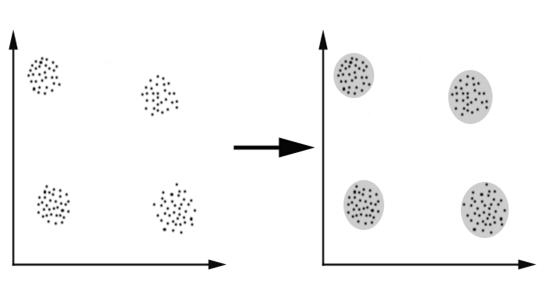
A value of k=10 is very common in the field of applied machine learning, and is recommend if you are struggling to choose a value for your dataset.

If a value for k is chosen that does not evenly split the data sample, then one group will contain a remainder of the examples. It is preferable to split the data sample into k groups with the same number of samples, such that the sample of model skill scores are all equivalent.

**11. CLUSTERING ALGORITHM:**

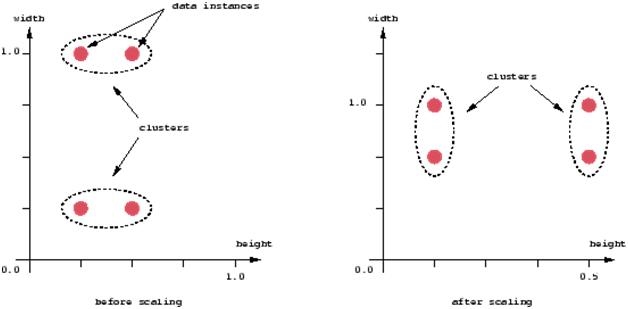
Clustering analysis has been an emerging research issue in data mining due its variety of applications. With the advent of many data clustering algorithms in the recent few years and its extensive use in wide variety of applications, including image processing, computational biology, mobile communication, medicine and economics, has led to the popularity of this algorithms. Main problem with the data clustering algorithms is that it cannot be standardized. Algorithm developed may give best result with one type of data set but may fail or give poor result with data set of other types. Although there have been many attempts for standardizing the algorithms which can perform well in all case of scenarios but till now no major accomplishment has been achieved. Many clustering algorithms have been proposed so far. However, each algorithm has its own merits and demerits and cannot work for all real situations.

Clustering is a process which partitions a given data set into homogeneous groups based on given features such that similar objects are kept in a group whereas dissimilar objects are in different groups. It is the most important unsupervised learning problem. It deals with finding structure in a collection of unlabeled data.



For clustering algorithm to be advantageous and beneficial some of the conditions need to be satisfied.

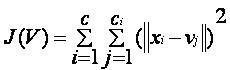
1. Scalability - Data must be scalable otherwise we may get the wrong result.



1. Clustering algorithm must be able to deal with different types of attributes.
2. Clustering algorithm must be able to find clustered data with the arbitrary shape.
3. Clustering algorithm must be insensitive to noise and outliers.
4. Interpret-ability and Usability - Result obtained must be interpretable and usable so that maximum knowledge about he input parameters can be obtained.
5. Clustering algorithm must be able to deal with data set of high dimensionalities.

**k-means clustering algorithm**

k-means is one of the simplest unsupervised learning algorithms that solve the well-known clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed apriori. The main idea is to define k centers, one for each cluster. These centers should be placed in a cunning way because of different location causes different result. So, the better choice is to place them as much as possible far away from each other. The next step is to take each point belonging to a given data set and associate it to the nearest center. When no point is pending, the first step is completed and an early group age is done. At this point we need to re-calculate k new centroids as barycenter of the clusters resulting from the previous step. After we have these k new centroids, a new binding has to be done between the same data set points and the nearest new center. A loop has been generated. As a result of this loop we may notice that the k centers change their location step by step until no more changes are done or in other words centers do not move any more. Finally, this algorithm aims at minimizing an objective function know as squared error function given by:



where,

‘||xi - vj||’ is the Euclidean distance between xi and vj.

‘ci’ is the number of data points in ith cluster.

‘c’ is the number of cluster centers.

**PROJECT**

In the project I have used following python libraries:

• Pandas

• Numpy

• Sk-Learn

The dataset given was of Sales Prediction Dataset in which there are columns contains:

* TV promotion budget (in million)
* Social Media promotion budget (in million)
* Radio promotion budget (in million)
* Influencer: Whether the promotion collaborate with Mega, Macro, Nano, Micro influencer
* Sales (in million)

are taken as FEATURES for the model building

This data can be used for simple tasks:

* Data preprocessing
* Exploratory Data Analysis
* Visualization
* Prediction using Linear Regression and Model Evaluation

In the project, I have built 2 models which are:

• LinearRegression

• KNN (K-Nearest Neighbors)

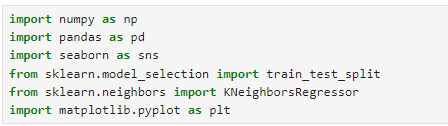
At each model building step, the dataset was split into test and train data set using test\_train\_split class of sklearn module.

According to model, accuracy score was generated and test data was verified.

Even the predictions are made after the model building.

**PROJECT ON SALES PREDICTIONS**

**1.Importing libraries**



**2.Load the dataset**

**A screenshot of a computer

Description automatically generated**

**3.Handling Missing and duplicated data**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**4.Statistical Summary**

**A screenshot of a computer

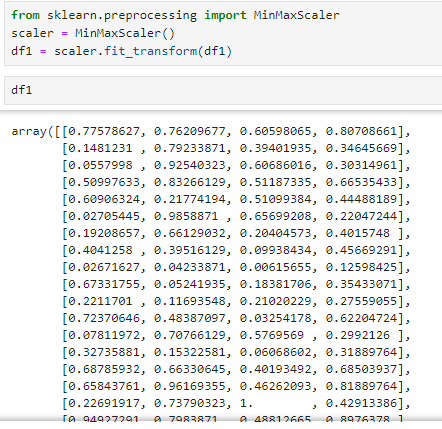
Description automatically generated**

|  |
| --- |
| **df.describe().transpose():** |
| * This command calculates summary statistics for all numerical columns in the df DataFrame. |
| * The describe() function returns metrics like count, mean, standard deviation, minimum, and maximum values, as well as the 25th, 50th (median), and 75th percentiles for each numeric column. |
| * The transpose() method flips the rows and columns so that the statistics become rows, and the column names become the index. |
| **sns.color\_palette("viridis", as\_cmap=True):** |
| * This line creates a color palette using the "viridis" color map from the Seaborn library, which is a gradient of colors designed to be visually appealing and easily interpreted in plots or heatmaps. |
| * as\_cmap=True ensures that it is returned as a continuous color map rather than a discrete palette. |
| **numerical\_summary.style.background\_gradient(cmap=palette):** |
| * This applies a background color gradient to the summary statistics DataFrame (numerical\_summary). |
| * The gradient is based on the values in the DataFrame: higher values will have a more intense color, while lower values will have lighter shades. |
| * The palette (which uses the "viridis" colormap) is applied as the gradient. |

**5. Data Manipulation**

****

**6.Feature Scaling**

****

**7.Splitting Data**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

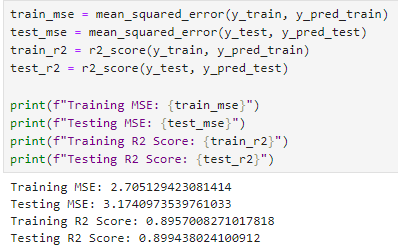
Description automatically generated**

**8.Training and evaluation of LinearRegression**

**LinearRegression( )  
A screenshot of a computer

Description automatically generated**

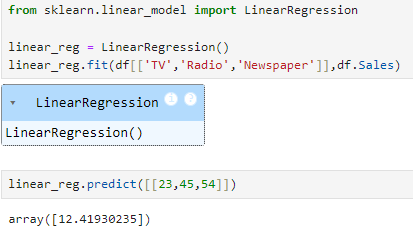
**#evaluate the model**

****

|  |
| --- |
| 1. train\_mse = mean\_squared\_error(y\_train, y\_pred\_train): |
| * + This calculates the Mean Squared Error (MSE) for the training set. |
| * + y\_train contains the actual values, and y\_pred\_train contains the predicted values from the model. |
| * + MSE measures how close the predictions are to the actual values. A lower MSE indicates better performance. |
| 1. test\_mse = mean\_squared\_error(y\_test, y\_pred\_test): |
| * + Similarly, this calculates the MSE for the test set using y\_test (actual values) and y\_pred\_test (predicted values). |
| * + This helps evaluate the model's performance on unseen data (testing data). |
| 1. train\_r2 = r2\_score(y\_train, y\_pred\_train): |
| * + This calculates the R² Score (also called the coefficient of determination) for the training set. |
| * + The R² score tells you how well the predictions explain the variance in the actual data. An R² score of 1 means perfect predictions, while an R² score of 0 means the model does no better than predicting the mean. |
| 1. test\_r2 = r2\_score(y\_test, y\_pred\_test): |
| * + This calculates the R² score for the test set, measuring how well the model generalizes to new, unseen data. |
| 1. print(f"Training MSE: {train\_mse}"): |
| * + This prints the MSE for the training set to the console. |
| 1. print(f"Testing MSE: {test\_mse}"): |
| * + This prints the MSE for the test set. |
| 1. print(f"Training R2 Score: {train\_r2}"): |
| * + This prints the R² score for the training set. |
| 1. print(f"Testing R2 Score: {test\_r2}"): |
| * + This prints the R² score for the test set. |
| In summary: |
| * The MSE values tell you how well the model is predicting, with lower values indicating better accuracy. |
| * The R² values tell you how much of the variance in the actual data is explained by the predictions, with values closer to 1 indicating a better fit. |
| * The code compares both training and test data to assess how well the model is performing during training and how well it generalizes to unseen data (testing). |

**Prediction of sales**

**A white rectangular object with red text

Description automatically generated**

**9. KNN (K-Nearest Neighbors)**

**A screenshot of a computer code

Description automatically generated**

**A graph with a line

Description automatically generated**

This code implements the Elbow Method to find the optimal number of neighbors (k) for a K-Nearest Neighbors (KNN) regression model, and plots the Mean Squared Error (MSE) against different values of k. Explanation of each part:

|  |
| --- |
| 1. mse\_values = [] and r2\_values = []: |
| * + These are empty lists where the Mean Squared Error (MSE) and R² scores for each value of k (number of neighbors) will be stored. |
| 1. max\_neighbors = 10: |
| * + This sets the maximum value for k to 10. The loop will try values of k from 1 to 10. |
| 1. for k in range(1, max\_neighbors + 1): |
| * + This loop iterates over possible values of k from 1 to 10. |
| 1. knn = KNeighborsRegressor(n\_neighbors=k): |
| * + In each iteration, a new K-Nearest Neighbors (KNN) regression model is created with the current value of k (i.e., the number of neighbors considered for prediction). |
| 1. knn.fit(X\_train, y\_train): |
| * + The model is trained on the training data (X\_train and y\_train). |
| 1. y\_pred = knn.predict(X\_test): |
| * + The model is used to predict the target values for the test set (X\_test), and the predicted values are stored in y\_pred. |
| 1. mse = mean\_squared\_error(y\_test, y\_pred): |
| * + The Mean Squared Error (MSE) between the actual values (y\_test) and the predicted values (y\_pred) is calculated. |
| 1. r2 = r2\_score(y\_test, y\_pred): |
| * + The R² score is also calculated for the test set to see how well the model explains the variance in the data. |
| 1. mse\_values.append(mse) and r2\_values.append(r2): |
| * + The calculated MSE and R² score for each value of k are added to the mse\_values and r2\_values lists, respectively. |
| 1. Plotting the Elbow Curve: |
| * + plt.plot(range(1, max\_neighbors + 1), mse\_values, marker='o'): This line plots the MSE values on the y-axis and the number of neighbors (k) on the x-axis, marking each point with a circle (marker='o'). |
| * + plt.xlabel('Number of Neighbors (k)'): Labels the x-axis as "Number of Neighbors (k)". |
| * + plt.ylabel('Mean Squared Error (MSE)'): Labels the y-axis as "Mean Squared Error (MSE)". |
| * + plt.title('Elbow Method for Optimal k'): Sets the plot title to "Elbow Method for Optimal k". |
| * + plt.grid(True): Adds a grid to the plot for better readability. |
| 1. plt.show(): |
| * + Displays the plot showing the MSE values for different k values. |
| Purpose of the Code: |
| * The Elbow Method is used to find the optimal value of k (number of neighbors) by observing where the MSE significantly drops and then stabilizes. This "elbow" point is often the best value of k for the model, balancing accuracy and simplicity. |

**Predictions made By KNN**

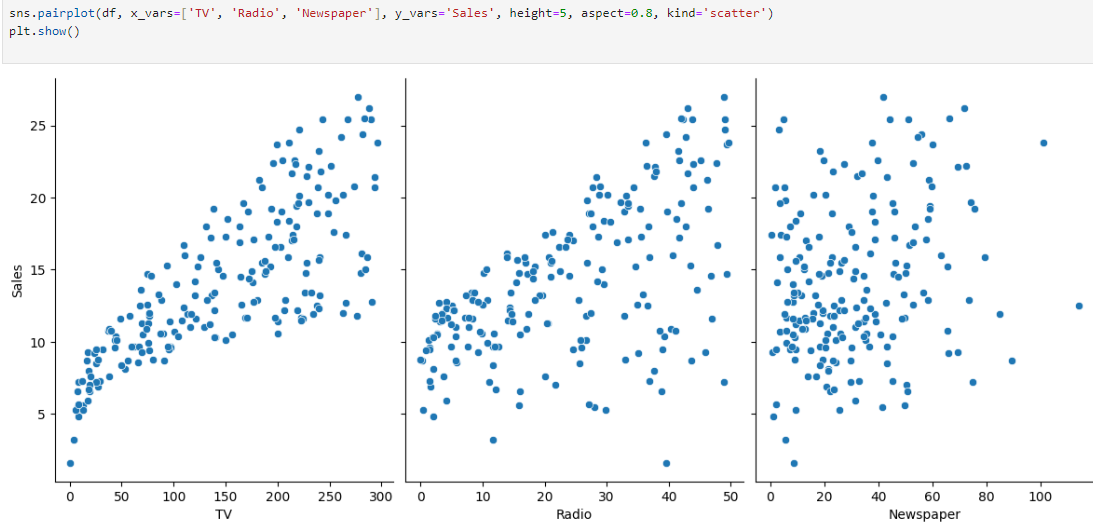
**A screenshot of a computer

Description automatically generated**

**A close up of a radio

Description automatically generated**

**10. Plotting the Columns**

****

Here we used the seaborn library. Here pairplot is a function of this library that can plot different different columns respective to one target column. At the same time we can plot multiple columns.

Here we can see that the KNeighborRegressor has the score value higher than the LinearRegression. So for this data set KNeighborRegressor algorithm is good because it has score value over 93%.

**References:**

1. Platform used -> [Jupyter NoteBook](https://jupyter.org/)

2. Dataset used from -> [Kaggle](https://www.kaggle.com/)

3. More:-

* [https://www.geeksforgeeks.org/](https://www.geeksforgeeks.org/  www.datacamp.com/https://scikit-learn.org/   https://towardsdatascience.com/  https://machinelearningmastery.com/)
* [www.datacamp.com/](https://www.geeksforgeeks.org/  www.datacamp.com/https://scikit-learn.org/   https://towardsdatascience.com/  https://machinelearningmastery.com/)
* [https://scikit-learn.org/](https://www.geeksforgeeks.org/  www.datacamp.com/https://scikit-learn.org/   https://towardsdatascience.com/  https://machinelearningmastery.com/)
* [https://towardsdatascience.com/](https://www.geeksforgeeks.org/  www.datacamp.com/https://scikit-learn.org/   https://towardsdatascience.com/  https://machinelearningmastery.com/)
* [https://machinelearningmastery.com/](https://www.geeksforgeeks.org/  www.datacamp.com/https://scikit-learn.org/   https://towardsdatascience.com/  https://machinelearningmastery.com/)