RAISONI GROUP
a vision beyond

G H Raisoni College of Engineering and Management
Wagholi, Pune
Department of Artificial Intelligence
(An Autonomous Institute Affiliated to SPPU, Pune)

PUNE

LAB MANUAL

CLASS: S.Y. BTech **SEMESTER: IV**

Course Outcomes:

CO1	Recognize the characteristics of machine learning that make it useful to real-world problems.
CO2	Characterize machine learning algorithms as supervised, semi-supervised, and unsupervised.
CO3	Design and implement machine learning solutions to classification, regression, and clustering problems;
CO4	Be able to evaluate and interpret the results of the algorithms
CO5	Effectively use machine learning toolboxes.

Subject Name: Machine Learning Algorithms (UAIP205)

Sr. No.	Name of Experiment	Co's mapping
1	Supervised learning regression: i) Take a dataset and perform preprocessing steps and Split into Training Data set and Test Data set. ii) Perform linear regression analysis. iii) Plot the graphs for Training and test dataset iv) Find out prediction result and check the accuracy v) Improve the accuracy while changing data points.	CO3,CO4,CO5
Content Beyond Syllabus		
2	Implement Logistic Regression using data set of your choice	CO3, CO4, CO5

3	Supervised Learning – Classification: Implement Naïve Bayes Classifier on data set of your choice. Test and Compare for Accuracy and Precision.	CO3,CO5
4	Implement k-neighbours classification using Scikit-learn (sklearn)	CO3,CO4
5	Given the following data, which specify classifications for nine combinations of VAR1 and VAR2 predict a classification for a case where VAR1=0.906 and VAR2=0.606, using the result of k-means clustering with 3 means (i.e., centroids) VAR1 VAR2 CLASS 1.713 1.586 0 0.180 1.786 1 0.353 1.240 1 0.940 1.566 0 1.486 0.759 1 1.266 1.106 0 1.540 0.419 1 0.459 1.799 1 0.773 0.186 1	CO3,CO4,CO5
6	Implement Linear Regression using Scikit-learn (sklearn)	CO4,CO5
7	Implement Naïve Bayes theorem to classify the English text	CO3,CO4,CO5
8	Unsupervised Learning: Implement K-Means Clustering and Hierarchical clustering on proper data set of your choice. Compare their Convergence.	CO3,CO5
9	Implement Support Vector Machine algorithm using suitable dataset.	CO3,CO4,CO5
Open Ended Assignments		
10	Write a program to demonstrate the working of the decision tree based ID3 algorithm. Use an appropriate data set for building the decision tree and apply this knowledge to classify a new sample.	CO3, CO4, CO5
11	Implement movie recommendation system using python.	CO3, CO4, CO5

Assignment No:- 1

Title of Assignment:

Supervised learning - regression

i) Take a dataset and perform preprocessing steps and Split into Training Data set and Test Data set.

ii) Perform linear regression analysis.

iii) Plot the graphs for Training and test dataset

iv) Find out prediction result and check the accuracy

v) Improve the accuracy while changing data points.

During assignment students will be able to:

- Learn Supervised Learning.
- Implementation of Linear Regression.

Prerequisites: Knowledge of python programming and machine learning libraries.

Concepts to be used: Linear Regression.

Input: Any dataset

Output: Successful implementation of Linear Regresion.

Relevant Theory / Literature Survey:

The linear regression algorithm in machine learning is a supervised learning technique to approximate the mapping function to get the best predictions. In this article, we will learn about linear regression for machine learning. The following topics are discussed in this practical.

- What is Regression?
- Types of Regression
- What is Linear Regression?
- Linear Regression Terminologies

- Advantages And Disadvantages Of Linear Regression
- Linear Regression Use Cases
- Use case – Linear Regression Implementation

What is Regression?

The main goal of regression is the construction of an efficient model to predict the dependent attributes from a bunch of attribute variables. A regression problem is when the output variable is either real or a continuous value i.e salary, weight, area, etc.

We can also define regression as a statistical means that is used in applications like housing, investing, etc. It is used to predict the relationship between a dependent variable and a bunch of independent variables. Let us take a look at various types of regression techniques.

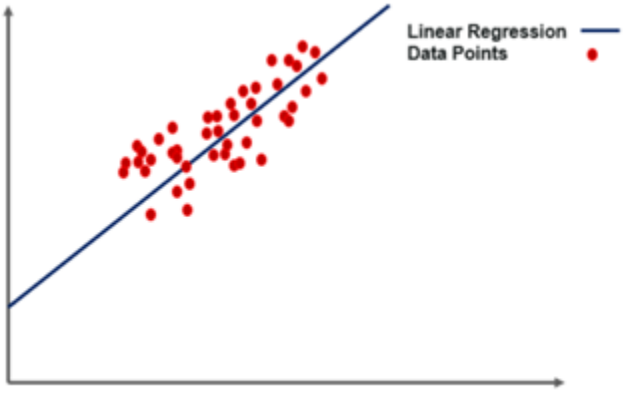
Types Of Regression

The following are types of regression.

1. Simple Linear Regression
2. Polynomial Regression
3. Support Vector Regression
4. Decision Tree Regression
5. Random Forest Regression

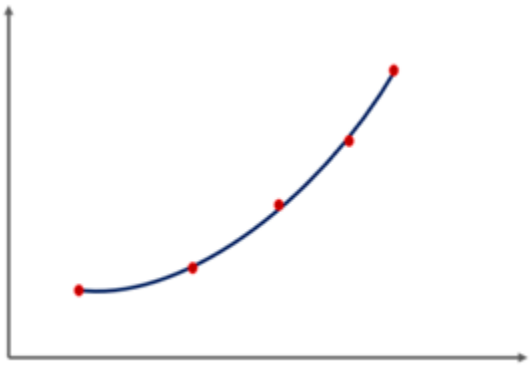
Simple Linear Regression

One of the most interesting and common regression technique is simple linear regression. In this, we predict the outcome of a dependent variable based on the independent variables, the relationship between the variables is linear. Hence, the word linear regression.



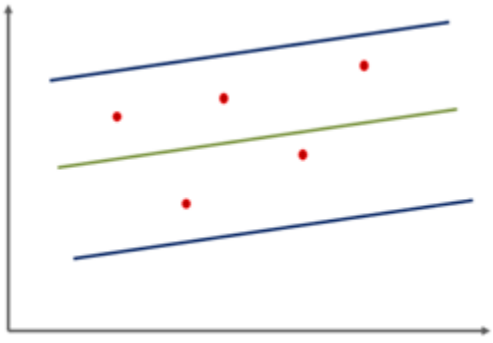
Polynomial Regression

In this regression technique, we transform the original features into polynomial features of a given degree and then perform regression on it.



Support Vector Regression

For support vector machine regression or SVR, we identify a hyperplane with maximum margin such that the maximum number of data points are within those margins. It is quite similar to the support vector machine classification algorithm.



Decision Tree Regression

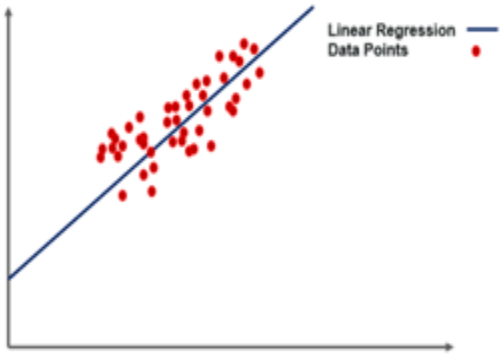
A decision tree can be used for both regression and classification. In the case of regression, we use the ID3 algorithm(Iterative Dichotomiser 3) to identify the splitting node by reducing the standard deviation.

Random Forest Regression

In random forest regression, we ensemble the predictions of several decision tree regressions. Now that we know about different types of regression let us take a look at simple linear regression in detail.

What is Linear Regression?

Simple linear regression is a regression technique in which the independent variable has a linear relationship with the dependent variable. The straight line in the diagram is the best fit line. The main goal of the simple linear regression is to consider the given data points and plot the best fit line to fit the model in the best way possible.



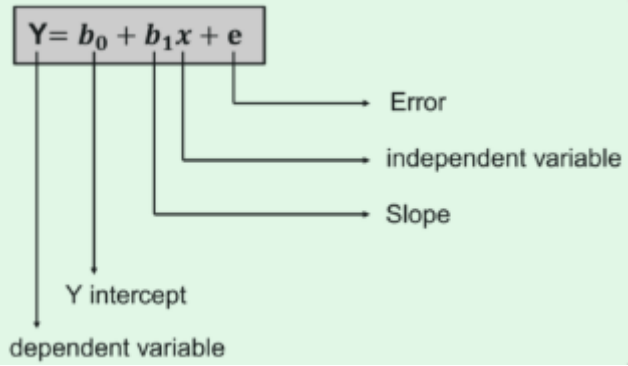
Before moving on to how the linear regression algorithm works, let us take a look at a few important terminologies in simple linear regression.

Linear Regression Terminologies

The following terminologies are important to be familiar with before moving on to the linear regression algorithm.

Cost Function

The best fit line can be based on the linear equation given below.



- The dependent variable that is to be predicted is denoted by Y.
- A line that touches the y-axis is denoted by the intercept b_0.
- b_1 is the slope of the line, x represents the independent variables that determine the prediction of Y.
- The error in the resultant prediction is denoted by e.

The cost function provides the best possible values for b_0 and b_1 to make the best fit line for the data points. We do it by converting this problem into a minimization problem to get the best values for b_0 and b_1. The error is minimized in this problem between the actual value and the predicted value.

$$minimize \frac{1}{n} \sum_{i=1}^{n} (pred_i - y_i)^2$$

$$J = \frac{1}{n} \sum_{i=1}^{n} (pred_i - y_i)^2$$

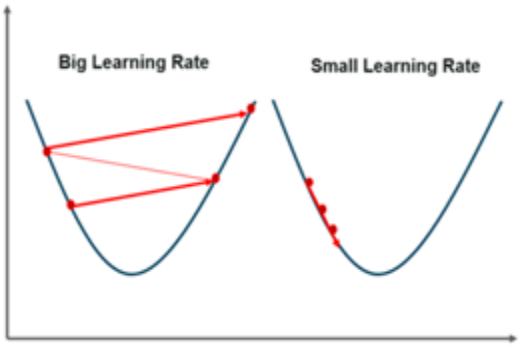
We choose the function above to minimize the error. We square the error difference and sum the error over all data points, the division between the total number of data points. Then, the produced value provides the averaged square error over all data points.

It is also known as MSE(Mean Squared Error), and we change the values of b_0 and b_1 so that the MSE value is settled at the minimum.

Gradient Descent

The next important terminology to understand linear regression is **gradient descent.** It is a method of updating b_0 and b_1 values to reduce the MSE. The idea behind this is to keep iterating the b_0 and b_1 values until we reduce the MSE to the minimum.

To update b_0 and b_1, we take gradients from the cost function. To find these gradients, we take partial derivatives with respect to b_0 and b_1. These partial derivatives are the gradients and are used to update the values of b_0 and b_1.



A smaller learning rate takes closer to the minimum, but it takes more time and in case of a larger learning rate. The time taken is sooner but there is a chance to overshoot the minimum value. Now that we are through with the terminologies in linear regression, let us take a look at a few advantages and disadvantages of linear regression for machine learning.

Advantages And Disadvantages

Advantages	Disadvantages
Linear regression performs exceptionally well for linearly separable data	The assumption of linearity between dependent and independent variables
Easier to implement, interpret and efficient to train	It is often quite prone to noise and overfitting
It handles overfitting pretty well using dimensionally reduction techniques, regularization, and cross-validation	Linear regression is quite sensitive to outliers
One more advantage is the extrapolation beyond a specific data set	It is prone to multicollinearity

Linear Regression Use Cases

- Sales Forecasting
- Risk Analysis
- Housing Applications To Predict the prices and other factors
- Finance Applications To Predict Stock prices, investment evaluation, etc.

The basic idea behind linear regression is to find the relationship between the dependent and independent variables. It is used to get the best fitting line that would predict the outcome with the least error. We can use linear regression in simple real-life situations, like predicting the SAT scores with regard to the number of hours of study and other decisive factors.

With this in mind, let us take a look at a use case.

Algorithm : Linear Regression

1. Loading the Data
2. Exploring the Data
3. Slicing The Data
4. Train and Split Data
5. Generate The Model
6. Evaluate The accuracy

Let us get into the details of each of the steps to implement linear regression.

1. Loading The Data

We can start with the basic diabetes data set that is already present in the sklearn(scikit-learn) data sets module to begin our journey with linear regression.

2. Exploring The Data

After we are done loading the data, we can start exploring by simply checking the labels by using the following code.

after this, we can slice the data so that we can plot the line in the end. We will also use all the data points, for now, we will slice column 2 from the data.

After this step, we will split the data into train and test set.

3. Splitting The Data

The next part involves generating the model, which will include importing linear_model from sklearn.

4. Generating the model

To evaluate the accuracy of the model, we will use the mean squared error from the scikit-learn.

5. Evaluation

To be more clear on how the data points look like on the graph, let us plot the graphs as well.

To get a more accurate model in this scenario, we can use the whole data instead of just the column 2.

FAQ'S

1. What is meant by Regression?
2. What is meant by Linear Regression?
3. What is cost function?
4. What is dependence & independent variable?
5. What is difference between classification and regression?

Outcome:

With the completion of this assignment the students will be able to implement linear regression on given dataset.

Conclusion:

Assignment No:- 2

Title of Assignment:

Implement logistic regression using dataset of your choice

During assignment students will be able to:

- Learn Supervised Learning.
- Implementation of Logistic Regression.

Prerequisites: Knowledge of python programming and machine learning libraries.

Concepts to be used: Logistic Regression.

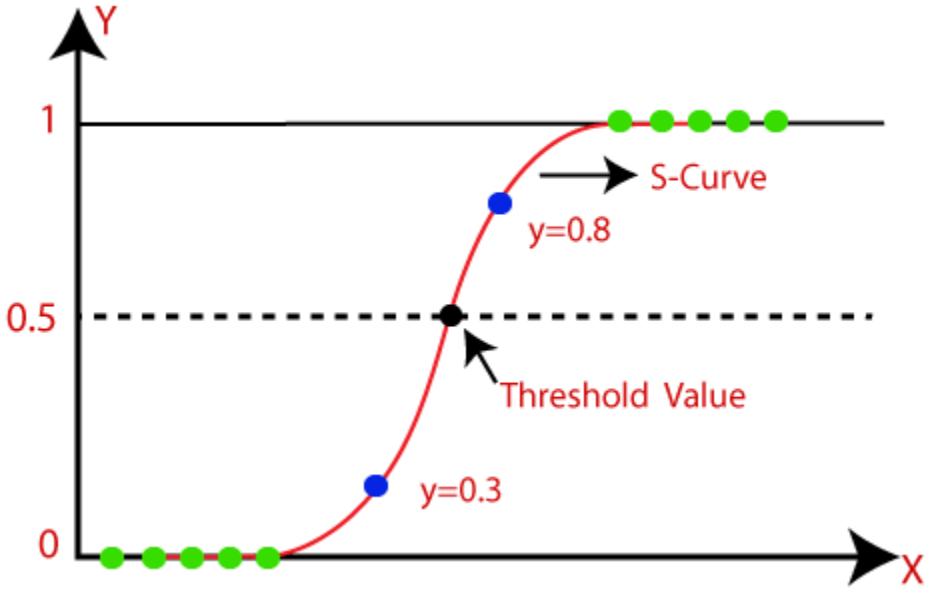
Input: Any dataset

Output: Successful implementation of Logistic Regression.

Relevant Theory / Literature Survey:

- Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.
- Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, **it gives the probabilistic values which lie between 0 and 1**.
- Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas **Logistic regression is used for solving the classification problems**.
- In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).
- The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.

- Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.
- Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification. The below image is showing the logistic function:



Logistic Function (Sigmoid Function):

- The sigmoid function is a mathematical function used to map the predicted values to probabilities.
- It maps any real value into another value within a range of 0 and 1.
- The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the "S" form. The S-form curve is called the Sigmoid function or the logistic function.
- In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0.

Assumptions for Logistic Regression:

- The dependent variable must be categorical in nature.
- The independent variable should not have multi-collinearity.

Logistic Regression Equation:

The Logistic regression equation can be obtained from the Linear Regression equation. The mathematical steps to get Logistic Regression equations are given below:

- We know the equation of the straight line can be written as:

$$y= b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \cdots + b_nx_n$$

- In Logistic Regression y can be between 0 and 1 only, so for this let's divide the above equation by (1-y):

$$\frac{y}{1-y}$$; 0 for y= 0, and infinity for y=1

- But we need range between -[infinity] to +[infinity], then take logarithm of the equation it will become:

$$log\left[\frac{y}{1-y}\right] = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \cdots + b_nx_n$$

The above equation is the final equation for Logistic Regression.

Type of Logistic Regression:

On the basis of the categories, Logistic Regression can be classified into three types:

- **Binomial:** In binomial Logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, Pass or Fail, etc.
- **Multinomial:** In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable, such as "cat", "dogs", or "sheep"
- **Ordinal:** In ordinal Logistic regression, there can be 3 or more possible ordered types of dependent variables, such as "low", "Medium", or "High".

For this problem, we will build a Machine Learning model using the Logistic regression algorithm. The dataset is shown in the below image. In this problem, we will predict the **purchased variable (Dependent Variable)** by using **age and salary (Independent variables)**.

User ID	Gender	Age	EstimatedSalary	Purchased
15624510	Male	19	19000	0
15810944	Male	35	20000	0
15668575	Female	26	43000	0
15603246	Female	27	57000	0
15804002	Male	19	76000	0
15728773	Male	27	58000	0
15598044	Female	27	84000	0
15694829	Female	32	150000	1
15600575	Male	25	33000	0
15727311	Female	35	65000	0
15570769	Female	26	80000	0
15606274	Female	26	52000	0
15746139	Male	20	86000	0
15704987	Male	32	18000	0
15628972	Male	18	82000	0
15697686	Male	29	80000	0
15733883	Male	47	25000	1
15617482	Male	45	26000	1
15704583	Male	46	28000	1
15621083	Female	48	29000	1
15649487	Male	45	22000	1
15736760	Female	47	49000	1

Algorithm in Logistic Regression: To implement the Logistic Regression using Python, the steps are given below:

1. Data Pre-processing
2. Fitting Logistic Regression to the Training set
3. Predicting the test result
4. Test accuracy of the result(Creation of Confusion matrix)
5. Visualizing the test set result.

FAQ'S

1. What is meant by Regression?
2. What is Logistic Regression?
3. What is Logistic function?
4. What are the different types of Logistic Regression?
5. What is the difference between classification and regression?

Outcome:

With the completion of this assignment the students will be able to implement logistic regression on given dataset.

Conclusion:

Assignment No: - 3

Title of Assignment:

Supervised Learning – Classification:

Implement Naïve Bayes Classifier on data set of your choice. Test and Compare for Accuracy and Precision.

During assignment students will be able to:

- Learn Supervised Learning.
- Implementation of Naïve Bayes Classifier.

Prerequisites: Knowledge of python programming and machine learning libraries.

Concepts to be used: Naïve Bayes Classifier

Input: Any dataset

Output: Successful implementation of Naïve Bayes Classifier.

Relevant Theory / Literature Survey:

Naïve Bayes Classifier Algorithm

- Naïve Bayes algorithm is a supervised learning algorithm, which is based on **Bayes theorem** and used for solving classification problems.
- It is mainly used in *text classification* that includes a high-dimensional training dataset.
- Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
- **It is a probabilistic classifier, which means it predicts on the basis of the probability of an object**.
- Some popular examples of Naïve Bayes Algorithm are **spam filtration, Sentimental analysis, and classifying articles**.

Why is it called Naïve Bayes?

The Naïve Bayes algorithm is comprised of two words Naïve and Bayes, Which can be described as:

- **Naïve**: It is called Naïve because it assumes that the occurrence of a certain feature is independent of the occurrence of other features. Such as if the fruit is identified on the bases of color, shape, and taste, then red, spherical, and sweet fruit is recognized as an apple. Hence each feature individually contributes to identify that it is an apple without depending on each other.
- **Bayes**: It is called Bayes because it depends on the principle of Bayes' Theorem.

Bayes' Theorem:

- Bayes' theorem is also known as **Bayes' Rule** or **Bayes' law**, which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.
- The formula for Bayes' theorem is given as:

$$P(A|B)= \frac{P(B|A)P(A)}{P(B)}$$

Where,

P(A|B) is Posterior probability: Probability of hypothesis A on the observed event B.

P(B|A) is Likelihood probability: Probability of the evidence given that the probability of a hypothesis is true.

P(A) is Prior Probability: Probability of hypothesis before observing the evidence.

P(B) is Marginal Probability: Probability of Evidence.

Working of Naïve Bayes' Classifier:

Working of Naïve Bayes' Classifier can be understood with the help of the below example:

Suppose we have a dataset of **weather conditions** and corresponding target variable "**Play**". So using this dataset we need to decide that whether we should play or not on a

particular day according to the weather conditions. So to solve this problem, we need to follow the below steps:

1. Convert the given dataset into frequency tables.
2. Generate Likelihood table by finding the probabilities of given features.
3. Now, use Bayes theorem to calculate the posterior probability.

Problem: If the weather is sunny, then the Player should play or not?

Solution: To solve this, first consider the below dataset:

	Outlook	Play
0	Rainy	Yes
1	Sunny	Yes
2	Overcast	Yes
3	Overcast	Yes
4	Sunny	No
5	Rainy	Yes
6	Sunny	Yes
7	Overcast	Yes
8	Rainy	No
9	Sunny	No
10	Sunny	Yes
11	Rainy	No
12	Overcast	Yes
13	Overcast	Yes

Frequency table for the Weather Conditions:

Weather	Yes	No
Overcast	5	0
Rainy	2	2
Sunny	3	2
Total	10	5

Likelihood table weather condition:

Weather	No	Yes	
Overcast	0	5	5/14= 0.35
Rainy	2	2	4/14=0.29
Sunny	2	3	5/14=0.35
All	4/14=0.29	10/14=0.71	

Applying Bayes'theorem:

P(Yes|Sunny)= P(Sunny|Yes)*P(Yes)/P(Sunny)

P(Sunny|Yes)= 3/10= 0.3

P(Sunny)= 0.35

P(Yes)=0.71

So P(Yes|Sunny) = 0.3*0.71/0.35= **0.60**

P(No|Sunny)= P(Sunny|No)*P(No)/P(Sunny)

P(Sunny|NO)= 2/4=0.5

P(No)= 0.29

P(Sunny)= 0.35

So P(No|Sunny)= 0.5*0.29/0.35 = **0.41**

So as we can see from the above calculation that **P(Yes|Sunny)>P(No|Sunny)**

Hence on a Sunny day, Player can play the game.

Advantages of Naïve Bayes Classifier:

- Naïve Bayes is one of the fast and easy ML algorithms to predict a class of datasets.
- It can be used for Binary as well as Multi-class Classifications.
- It performs well in Multi-class predictions as compared to the other Algorithms.
- It is the most popular choice for **text classification problems**.

Disadvantages of Naïve Bayes Classifier:

- Naive Bayes assumes that all features are independent or unrelated, so it cannot learn the relationship between features.

Applications of Naïve Bayes Classifier:

- It is used for **Credit Scoring**.
- It is used in **medical data classification**.
- It can be used in **real-time predictions** because Naïve Bayes Classifier is an eager learner.
- It is used in Text classification such as **Spam filtering** and **Sentiment analysis**.

Types of Naïve Bayes Model:

There are three types of Naive Bayes Model, which are given below:

- **Gaussian**: The Gaussian model assumes that features follow a normal distribution. This means if predictors take continuous values instead of discrete,

then the model assumes that these values are sampled from the Gaussian distribution.

- **Multinomial**: The Multinomial Naïve Bayes classifier is used when the data is multinomial distributed. It is primarily used for document classification problems, it means a particular document belongs to which category such as Sports, Politics, education, etc.
  The classifier uses the frequency of words for the predictors.
- **Bernoulli**: The Bernoulli classifier works similar to the Multinomial classifier, but the predictor variables are the independent Booleans variables. Such as if a particular word is present or not in a document. This model is also famous for document classification tasks.

Algorithm : Naïve Bayes algorithm:

1. Data Pre-processing step

2. Fitting Naive Bayes to the Training set

3. Predicting the test result

4. Test accuracy of the result(Creation of Confusion matrix)

5. Visualizing the test set result.

1) Data Pre-processing step:

In this step, we will pre-process/prepare the data so that we can use it efficiently in our code. It is similar as we did in data-pre-processing.

2) Fitting Naive Bayes to the Training Set:

After the pre-processing step, now we will fit the Naive Bayes model to the Training set.

3) Prediction of the test set result:

Predict the test set result. For this, we will create a new predictor variable **y_pred**, and will use the predict function to make the predictions.

some predications are different from the real values, which are the incorrect predictions.

4) Creating Confusion Matrix:

Now we will check the accuracy of the Naive Bayes classifier using the Confusion matrix.

5) Visualizing the training set result:

Next we will visualize the training set result using Naïve Bayes Classifier.

6) Visualizing the Test set result:

FAQ'S

1. What is meant by Regression?
2. What is meant by Naïve Bias classification?
3. What is classification?
4. What is dependence & independent variable?
5. What is difference between classification and regression?

Outcome:

With the completion of this assignment the students will be able to implement

Naïve bayes on given dataset

Conclusion:

Assignment No:- 4

Title of Assignment:

Implement K Nearest Neighbor(KNN) classification using Scikit-learn (sklearn)

During assignment students will be able to:

- Learn Classification.
- Implementation of K Nearest Neighbor(KNN) classification.

Prerequisites: Knowledge of python programming and machine learning libraries.

Concepts to be used: Classification.

Input: Any dataset

Output: Successful implementation of K Nearest Neighbor(KNN) classification.

Relevant Theory / Literature Survey:

K Nearest Neighbor(KNN) is a very simple, easy to understand, versatile and one of the topmost machine learning algorithms. KNN used in the variety of applications such as finance, healthcare, political science, handwriting detection, image recognition and video recognition. In Credit ratings, financial institutes will predict the credit rating of customers. In loan disbursement, banking institutes will predict whether the loan is safe or risky. In political science, classifying potential voters in two classes will vote or won't vote. KNN algorithm used for both classification and regression problems. KNN algorithm based on feature similarity approach.

In this tutorial, you are going to cover the following topics:

- K-Nearest Neighbor Algorithm
- How does the KNN algorithm work?
- Eager Vs Lazy learners

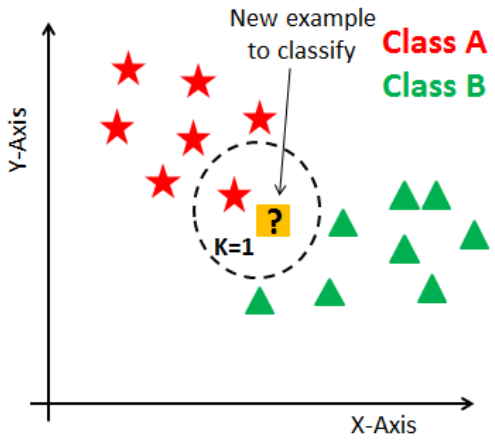
- How do you decide the number of neighbors in KNN?
- Curse of Dimensionality
- Classifier Building in Scikit-learn
- Pros and Cons
- How to improve KNN performance?
- Conclusion

K-Nearest Neighbors

KNN is a non-parametric and lazy learning algorithm. Non-parametric means there is no assumption for underlying data distribution. In other words, the model structure determined from the dataset. This will be very helpful in practice where most of the real world datasets do not follow mathematical theoretical assumptions. Lazy algorithm means it does not need any training data points for model generation. All training data used in the testing phase. This makes training faster and testing phase slower and costlier. Costly testing phase means time and memory. In the worst case, KNN needs more time to scan all data points and scanning all data points will require more memory for storing training data.

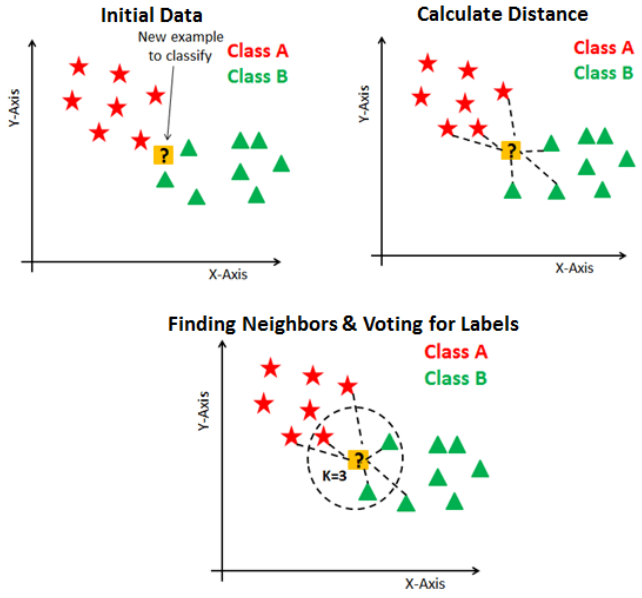
How does the KNN algorithm work?

In KNN, K is the number of nearest neighbors. The number of neighbors is the core deciding factor. K is generally an odd number if the number of classes is 2. When K=1, then the algorithm is known as the nearest neighbor algorithm. This is the simplest case. Suppose P1 is the point, for which label needs to predict. First, you find the one closest point to P1 and then the label of the nearest point assigned to P1.



Suppose P1 is the point, for which label needs to predict. First, you find the k closest point to P1 and then classify points by majority vote of its k neighbors. Each object votes for their class and the class with the most votes is taken as the prediction. For finding closest similar points, you find the distance between points using distance measures such as Euclidean distance, Hamming distance, Manhattan distance and Minkowski distance. KNN has the following basic steps:

1. Calculate distance
2. Find closest neighbors
3. Vote for labels



Eager Vs. Lazy Learners

Eager learners mean when given training points will construct a generalized model before performing prediction on given new points to classify. You can think of such learners as being ready, active and eager to classify unobserved data points.

Lazy Learning means there is no need for learning or training of the model and all of the data points used at the time of prediction. Lazy learners wait until the last minute before classifying any data point. Lazy learner stores merely the training dataset and waits until classification needs to perform. Only when it sees the test tuple does it perform generalization to classify the tuple based on its similarity to the stored training tuples. Unlike eager learning methods, lazy learners do less work in the training phase and more work in the testing phase to make a classification. Lazy learners are also known as instance-based learners because lazy learners store the training points or instances, and all learning is based on instances.

Curse of Dimensionality

KNN performs better with a lower number of features than a large number of features. You can say that when the number of features increases than it requires more data. Increase in dimension also leads to the problem of overfitting. To avoid overfitting, the

needed data will need to grow exponentially as you increase the number of dimensions. This problem of higher dimension is known as the Curse of Dimensionality.

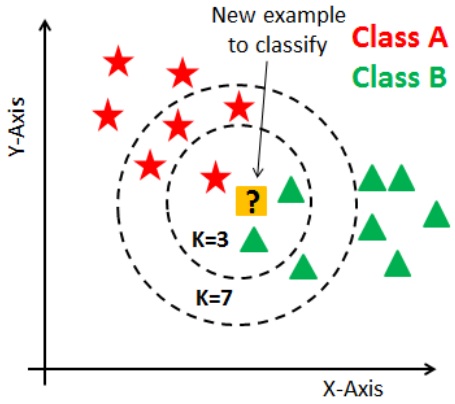
To deal with the problem of the curse of dimensionality, you need to perform principal component analysis before applying any machine learning algorithm, or you can also use feature selection approach. Research has shown that in large dimension Euclidean distance is not useful anymore. Therefore, you can prefer other measures such as cosine similarity, which get decidedly less affected by high dimension.

How do you decide the number of neighbors in KNN?

Now, you understand the KNN algorithm working mechanism. At this point, the question arises that How to choose the optimal number of neighbors? And what are its effects on the classifier? The number of neighbors(K) in KNN is a hyperparameter that you need choose at the time of model building. You can think of K as a controlling variable for the prediction model.

Research has shown that no optimal number of neighbors suits all kind of data sets. Each dataset has it's own requirements. In the case of a small number of neighbors, the noise will have a higher influence on the result, and a large number of neighbors make it computationally expensive. Research has also shown that a small amount of neighbors are most flexible fit which will have low bias but high variance and a large number of neighbors will have a smoother decision boundary which means lower variance but higher bias.

Generally, Data scientists choose as an odd number if the number of classes is even. You can also check by generating the model on different values of k and check their performance. You can also try Elbow method here.



Algorithm : KNN Classifier

1. Defining dataset

Let's first create your own dataset. Here you need two kinds of attributes or columns in your data: Feature and label. The reason for two type of column is "supervised nature of KNN algorithm".

2. Encoding data columns

Various machine learning algorithms require numerical input data, so you need to represent categorical columns in a numerical column.

This process is known as label encoding, and sklearn conveniently will do this for you using Label Encoder.

Here, you imported preprocessing module and created Label Encoder object.

3. Combining Features

Here, you will combine multiple columns or features into a single set of data using "zip" function

4. Generating Model

Let's build KNN classifier model.

First, import the KNeighborsClassifier module and create KNN classifier object by passing argument number of neighbors in KNeighborsClassifier() function.

Then, fit your model on the train set using fit() and perform prediction on the test set using predict().

Algorithm - KNN with Multiple Labels

Till now, you have learned How to create KNN classifier for two in python using scikit-learn. Now you will learn about KNN with multiple classes.

1. Loading Data

Let's first load the required wine dataset from scikit-learn datasets.

2. Exploring Data

After you have loaded the dataset, you might want to know a little bit more about it. You can check feature and target names.

Let's check top 5 records of the feature set.

Let's check records of the target set.

Let's explore it for a bit more. You can also check the shape of the dataset using shape.

3. Splitting Data

To understand model performance, dividing the dataset into a training set and a test set is a good strategy.

Let's split dataset by using function train_test_split(). You need to pass 3 parameters features, target, and test_set size. Additionally, you can use random_state to select records randomly.

4. Generating Model for K=5

Let's build KNN classifier model for k=5.

5. Model Evaluation for k=5

Let's estimate, how accurately the classifier or model can predict the type of cultivars.

Accuracy can be computed by comparing actual test set values and predicted values.

Well, you will get a classification rate .

For further evaluation, you can also create a model for a different number of neighbors.

6. Re-generating Model for K=7

Let's build KNN classifier model for k=7.

7. Model Evaluation for k=7

Let's again estimate, how accurately the classifier or model can predict the type of cultivars for k=7.

Well, you will get a better classification rate.

Here, you have increased the number of neighbors in the model and accuracy got increased. But, this is not necessary for each case that an increase in many neighbors increases the accuracy. For a more detailed understanding of it, you can refer section "How to decide the number of neighbors?" of this tutorial.

Pros

The training phase of K-nearest neighbor classification is much faster compared to other classification algorithms. There is no need to train a model for generalization, That is why KNN is known as the simple and instance-based learning algorithm. KNN can be useful in case of nonlinear data. It can be used with the regression problem. Output value for the object is computed by the average of k closest neighbors value.

Cons

The testing phase of K-nearest neighbor classification is slower and costlier in terms of time and memory. It requires large memory for storing the entire training dataset for prediction. KNN requires scaling of data because KNN uses the Euclidean distance between two data points to find nearest neighbors. Euclidean distance is sensitive to magnitudes. The features with high magnitudes will weight more than features with low magnitudes. KNN also not suitable for large dimensional data.

How to improve KNN?

For better results, normalizing data on the same scale is highly recommended. Generally, the normalization range considered between 0 and 1. KNN is not suitable for the large dimensional data. In such cases, dimension needs to reduce to improve the performance. Also, handling missing values will help us in improving results.

FAQ'S

1. What is meant by classification?
2. What is KNN?
3. What is the pros and cons of KNN?
4. What is dependence & independent variable?
5. What is difference between classification and regression?

Outcome:

With the completion of this assignment the students will be able to implement KNN on given dataset.

Conclusion:

Assignment No:- 5

Title of Assignment:

Given the following data, which specify classifications for nine combinations of VAR1 and VAR2 predict a classification for a case where VAR1=0.906 and VAR2=0.606, using the result of k-means clustering with 3 means (i.e., centroids)

VAR1	VAR2	CLASS
1.713	1.586	0
0.180	1.786	1
0.353	1.240	1
0.940	1.566	0
1.486	0.759	1
1.266	1.106	0
1.540	0.419	1
0.459	1.799	1
0.773	0.186	1

During assignment students will be able to:

- Learn Unsupervised Learning & Clustering.
- Implementation of k-means clustering.

Prerequisites: Knowledge of python programming and machine learning libraries.

Concepts to be used: Unsupervised Learning & Clustering.

Input: given dataset

Output: Successful implementation of k-means clustering.

Relevant Theory / Literature Survey:

K-Means Clustering Algorithm

K-Means Clustering is an unsupervised learning algorithm that is used to solve the clustering problems in machine learning or data science. In this topic, we will learn what

is K-means clustering algorithm, how the algorithm works, along with the Python implementation of k-means clustering.

What is K-Means Algorithm?

K-Means Clustering is an Unsupervised Learning algorithm

, which groups the unlabeled dataset into different clusters. Here K defines the number of pre-defined clusters that need to be created in the process, as if K=2, there will be two clusters, and for K=3, there will be three clusters, and so on.

It is an iterative algorithm that divides the unlabeled dataset into k different clusters in such a way that each dataset belongs only one group that has similar properties.

It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the unlabeled dataset on its own without the need for any training.

It is a centroid-based algorithm, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.

The algorithm takes the unlabeled dataset as input, divides the dataset into k-number of clusters, and repeats the process until it does not find the best clusters. The value of k should be predetermined in this algorithm.

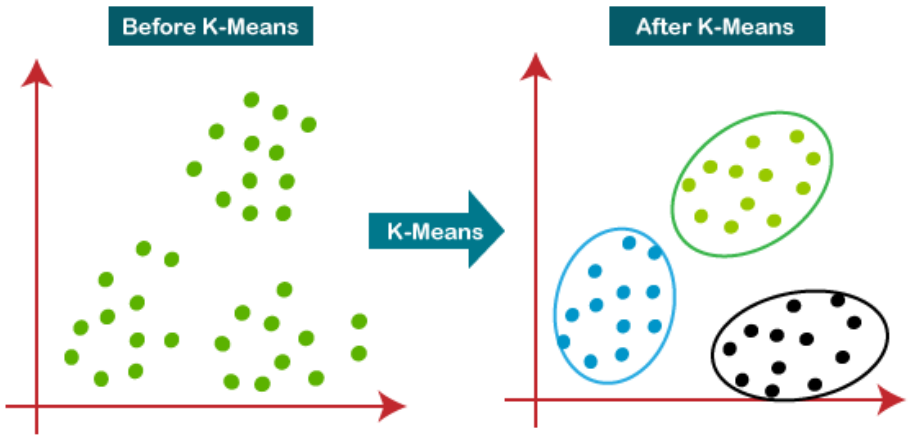
The k-means clustering

algorithm mainly performs two tasks:

- Determines the best value for K center points or centroids by an iterative process.
- Assigns each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster.

Hence each cluster has datapoints with some commonalities, and it is away from other clusters.

The below diagram explains the working of the K-means Clustering Algorithm:



How does the K-Means Algorithm Work?

The working of the K-Means algorithm is explained in the below steps:

Step-1: Select the number K to decide the number of clusters.

Step-2: Select random K points or centroids. (It can be other from the input dataset).

Step-3: Assign each data point to their closest centroid, which will form the predefined K clusters.

Step-4: Calculate the variance and place a new centroid of each cluster.

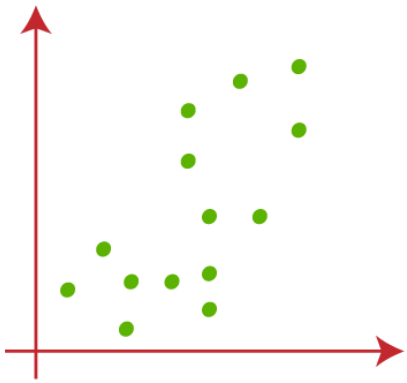
Step-5: Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.

Step-6: If any reassignment occurs, then go to step-4 else go to FINISH.

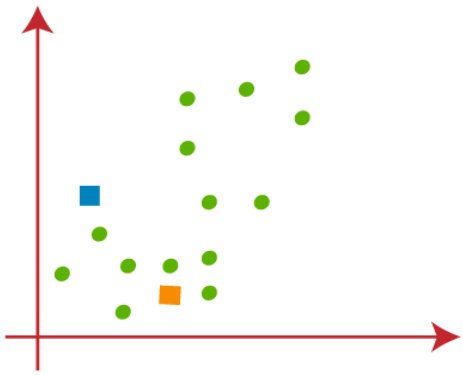
Step-7: The model is ready.

Let's understand the above steps by considering the visual plots:

Suppose we have two variables M1 and M2. The x-y axis scatter plot of these two variables is given below:

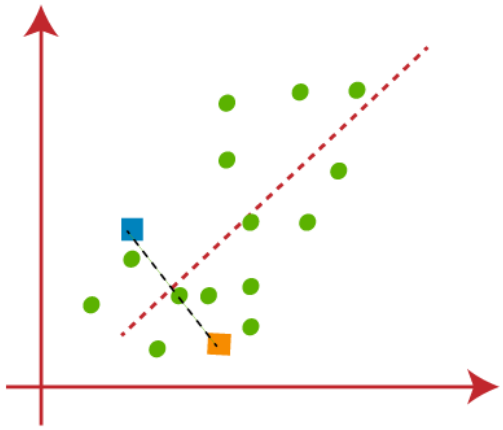


- Let's take number k of clusters, i.e., K=2, to identify the dataset and to put them into different clusters. It means here we will try to group these datasets into two different clusters.
- We need to choose some random k points or centroid to form the cluster. These points can be either the points from the dataset or any other point. So, here we are selecting the below two points as k points, which are not the part of our dataset. Consider the below image:

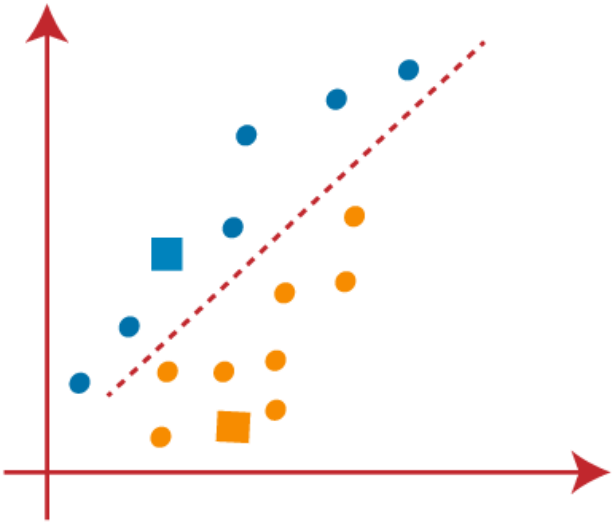


- Now we will assign each data point of the scatter plot to its closest K-point or centroid. We will compute it by applying some mathematics that we have studied to calculate the distance between two points. So, we will draw a median between

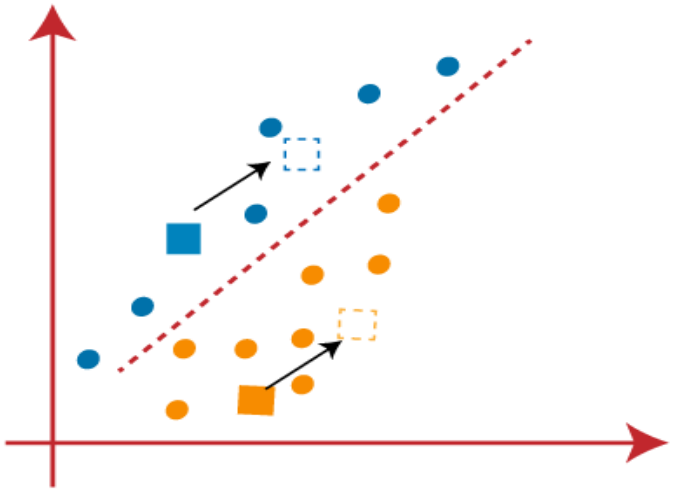
both the centroids. Consider the below image:



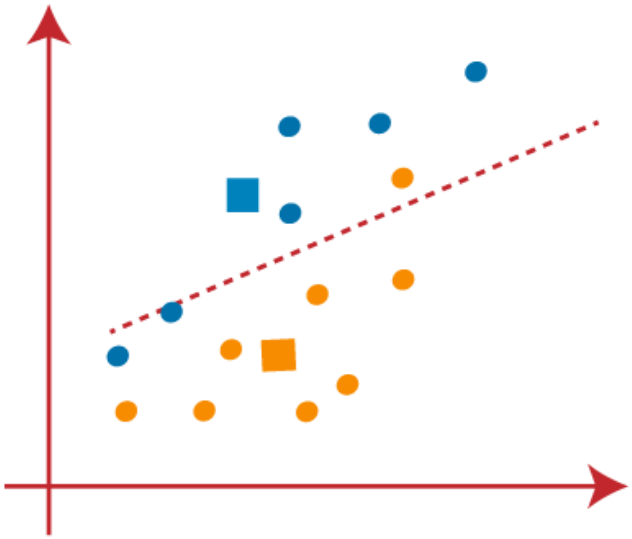
From the above image, it is clear that points left side of the line is near to the K1 or blue centroid, and points to the right of the line are close to the yellow centroid. Let's color them as blue and yellow for clear visualization.



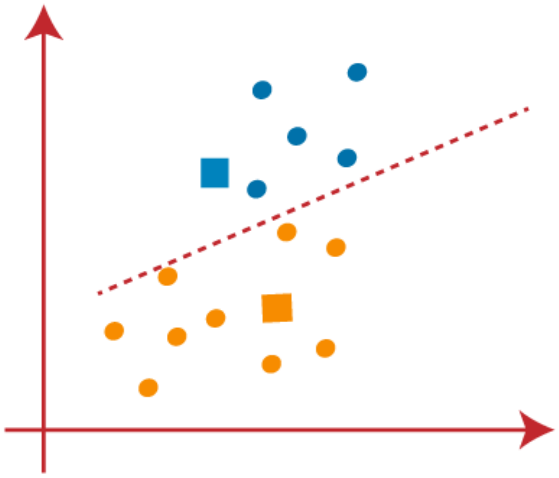
- As we need to find the closest cluster, so we will repeat the process by choosing **a new centroid**. To choose the new centroids, we will compute the center of gravity of these centroids, and will find new centroids as below:



- Next, we will reassign each datapoint to the new centroid. For this, we will repeat the same process of finding a median line. The median will be like below image:



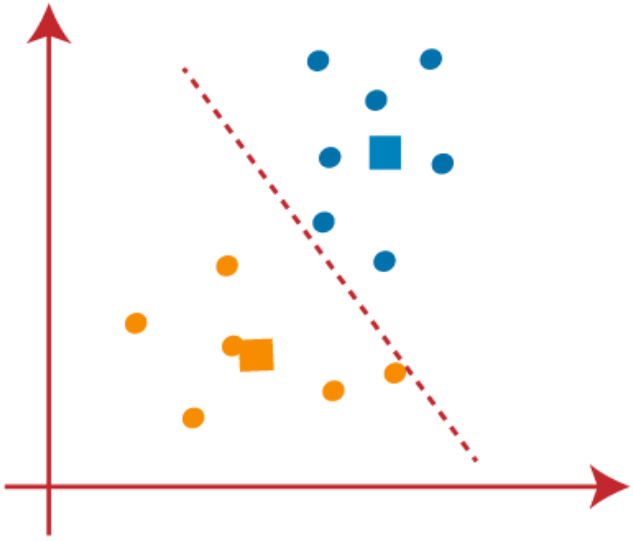
From the above image, we can see, one yellow point is on the left side of the line, and two blue points are right to the line. So, these three points will be assigned to new centroids.



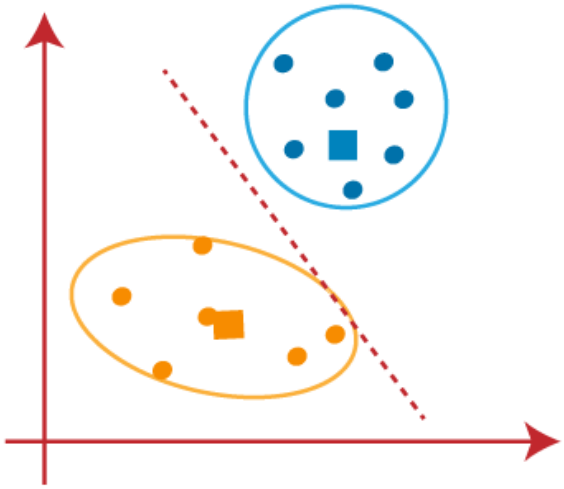
As reassignment has taken place, so we will again go to the step-4, which is finding new centroids or K-points.

- We will repeat the process by finding the center of gravity of centroids, so the new centroids will be as shown in the below image:

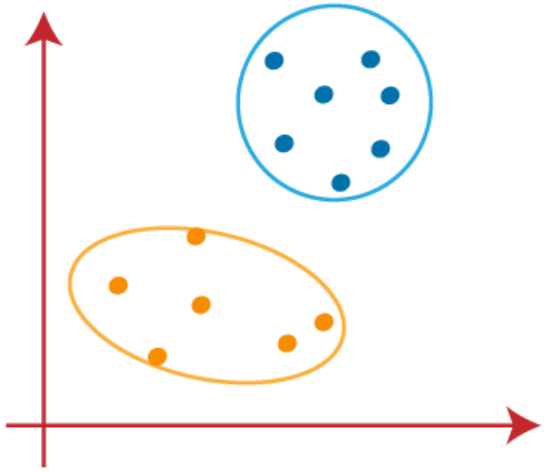
- As we got the new centroids so again will draw the median line and reassign the data points. So, the image will be:



- We can see in the above image; there are no dissimilar data points on either side of the line, which means our model is formed. Consider the below image:



As our model is ready, so we can now remove the assumed centroids, and the two final clusters will be as shown in the below image:



How to choose the value of "K number of clusters" in K-means Clustering?

The performance of the K-means clustering algorithm depends upon highly efficient clusters that it forms. But choosing the optimal number of clusters is a big task. There are some different ways to find the optimal number of clusters, but here we are discussing the most appropriate method to find the number of clusters or value of K. The method is given below:

Elbow Method

The Elbow method is one of the most popular ways to find the optimal number of clusters. This method uses the concept of WCSS value. **WCSS** stands for **Within Cluster Sum of Squares**, which defines the total variations within a cluster. The formula to calculate the value of WCSS (for 3 clusters) is given below:

$$WCSS= \sum_{Pi \text{ in } Cluster1} distance(P_i\ C_1)^2 + \sum_{Pi \text{ in } Cluster2} distance(P_i\ C_2)^2 + \sum_{Pi \text{ in } CLuster3} distance(P_i\ C_3)^2$$

In the above formula of WCSS,

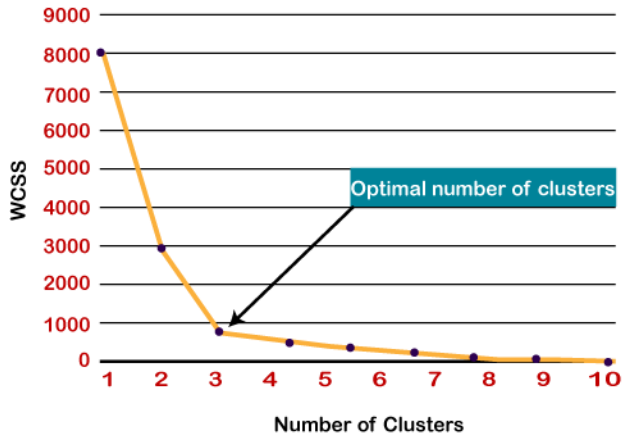
$\sum_{Pi \text{ in } Cluster1} distance(P_i\ C_1)^2$: It is the sum of the square of the distances between each data point and its centroid within a cluster1 and the same for the other two terms.

To measure the distance between data points and centroid, we can use any method such as Euclidean distance or Manhattan distance.

To find the optimal value of clusters, the elbow method follows the below steps:

- It executes the K-means clustering on a given dataset for different K values (ranges from 1-10).
- For each value of K, calculates the WCSS value.
- Plots a curve between calculated WCSS values and the number of clusters K.
- The sharp point of bend or a point of the plot looks like an arm, then that point is considered as the best value of K.

Since the graph shows the sharp bend, which looks like an elbow, hence it is known as the elbow method. The graph for the elbow method looks like the below image:



Algorithm : K-means Clustering Algorithm

The steps to be followed for the implementation are given below:

- **Data Pre-processing**
- **Finding the optimal number of clusters using the elbow method**
- **Training the K-means algorithm on the training dataset**
- **Visualizing the clusters**

Step-1: Data pre-processing

The first step will be the data pre-processing, as we did in our earlier topics of Regression and Classification. But for the clustering problem, it will be different from other models. Let's discuss it:

- **Importing Libraries**
  As we did in previous topics, firstly, we will import the libraries for our model, which is part of data pre-processing.
- numpy - for the performing mathematics calculation
- matplotlib - for plotting the graph
- pandas - for managing the dataset.

Importing the Dataset:
Next, we will import the dataset that we need to use.

Extracting Independent Variables

Step-2: Finding the optimal number of clusters using the elbow method

In the second step, we will try to find the optimal number of clusters for our clustering problem. So, as discussed above, here we are going to use the elbow method for this purpose.

As we know, the elbow method uses the WCSS concept to draw the plot by plotting WCSS values on the Y-axis and the number of clusters on the X-axis. So we are going to calculate the value for WCSS for different k values ranging from 1 to 10.

Next, we have created the **wcss_list** variable to initialize an empty list, which is used to contain the value of wcss computed for different values of k ranging from 1 to 10.

After that, we have initialized the for loop for the iteration on a different value of k ranging from 1 to 10; since for loop in Python, exclude the outbound limit, so it is taken as 11 to include 10^{th} value.

The rest part of the code is similar as we did in earlier topics, as we have fitted the model on a matrix of features and then plotted the graph between the number of clusters and WCSS.

Step- 3: Training the K-means algorithm on the training dataset

As we have got the number of clusters, so we can now train the model on the dataset.

Step-4: Visualizing the Clusters

The last step is to visualize the clusters. As we have 5 clusters for our model, so we will visualize each cluster one by one.

To visualize the clusters will use scatter plot using mtp.scatter() function of matplotlib.

FAQ'S

1. What is meant by Unsupervised learning?
2. What is meant by clustering?
3. What is k-means clustering?

Outcome:

With the completion of this assignment the students will be able to implement k-means clustering on given dataset.

Conclusion:

Assignment No:- 6

Title of Assignment:

Implement Linear Regression using Scikit-learn (sklearn)

During assignment students will be able to:

- Learn Supervised Learning and regression.
- Implementation of Linear Regression.

Prerequisites: Knowledge of python programming and machine learning libraries.

Concepts to be used: Regression.

Input: Any dataset

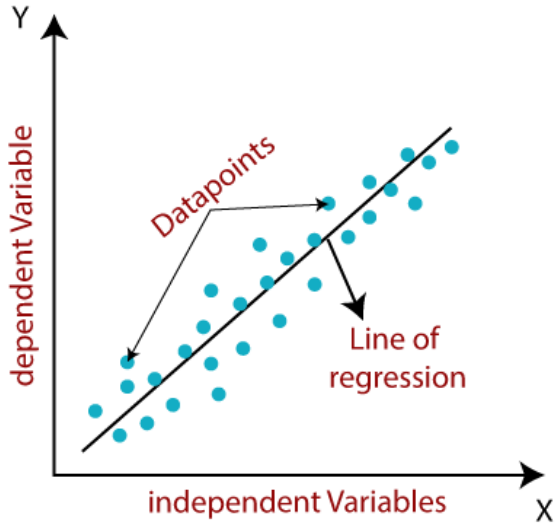
Output: Successful implementation of Linear Regression.

Relevant Theory / Literature Survey:

Linear regression is one of the easiest and most popular Machine Learning algorithms. It is a statistical method that is used for predictive analysis. Linear regression makes predictions for continuous/real or numeric variables such as **sales, salary, age, product price,** etc.

Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (y) variables, hence called as linear regression. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable is changing according to the value of the independent variable.

The linear regression model provides a sloped straight line representing the relationship between the variables. Consider the below image:



Mathematically, we can represent a linear regression as:

$$y= a_0+a_1x+ \varepsilon$$

Here,

Y= Dependent Variable (Target Variable)
X= Independent Variable (predictor Variable)
a0= intercept of the line (Gives an additional degree of freedom)
a1 = Linear regression coefficient (scale factor to each input value).
ε = random error

The values for x and y variables are training datasets for Linear Regression model representation.

Types of Linear Regression

Linear regression can be further divided into two types of the algorithm:

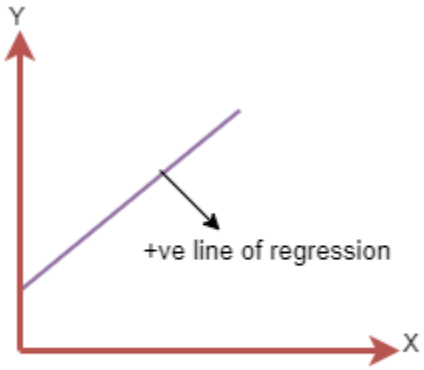
- **Simple Linear Regression:**
  If a single independent variable is used to predict the value of a numerical dependent variable, then such a Linear Regression algorithm is called Simple Linear Regression.
- **Multiple Linear regression:**
  If more than one independent variable is used to predict the value of a numerical

dependent variable, then such a Linear Regression algorithm is called Multiple Linear Regression.

Linear Regression Line

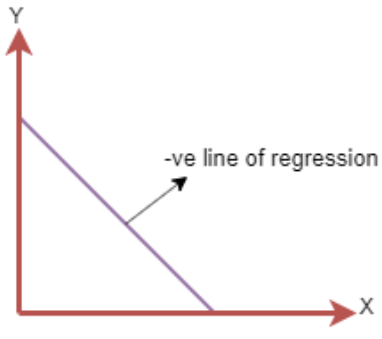
A linear line showing the relationship between the dependent and independent variables is called a **regression line**. A regression line can show two types of relationship:

- **Positive Linear Relationship:**
  If the dependent variable increases on the Y-axis and independent variable increases on X-axis, then such a relationship is termed as a Positive linear relationship.



The line equation will be: $Y= a_0+a_1x$

- **Negative Linear Relationship:**
  If the dependent variable decreases on the Y-axis and independent variable increases on the X-axis, then such a relationship is called a negative linear relationship.



The line of equation will be: $Y= -a_0+a_1x$

Finding the best fit line:

When working with linear regression, our main goal is to find the best fit line that means the error between predicted values and actual values should be minimized. The best fit line will have the least error.

The different values for weights or the coefficient of lines (a_0, a_1) gives a different line of regression, so we need to calculate the best values for a_0 and a_1 to find the best fit line, so to calculate this we use cost function.

Cost function-

- The different values for weights or coefficient of lines (a_0, a_1) gives the different line of regression, and the cost function is used to estimate the values of the coefficient for the best fit line.
- Cost function optimizes the regression coefficients or weights. It measures how a linear regression model is performing.
- We can use the cost function to find the accuracy of the **mapping function**, which maps the input variable to the output variable. This mapping function is also known as **Hypothesis function**.

For Linear Regression, we use the **Mean Squared Error (MSE)** cost function, which is the average of squared error occurred between the predicted values and actual values. It can be written as:

For the above linear equation, MSE can be calculated as:

$$MSE = 1\frac{1}{N}\sum_{i=1}^{n}(y_i - (a_1x_i + a_0))^2$$

Where,

N=Total number of observation
Yi = Actual value
$(a1x_i+a_0)$= Predicted value.

Residuals: The distance between the actual value and predicted values is called residual. If the observed points are far from the regression line, then the residual will be high, and so cost function will high. If the scatter points are close to the regression line, then the residual will be small and hence the cost function.

Gradient Descent:

- Gradient descent is used to minimize the MSE by calculating the gradient of the cost function.
- A regression model uses gradient descent to update the coefficients of the line by reducing the cost function.
- It is done by a random selection of values of coefficient and then iteratively update the values to reach the minimum cost function.

Model Performance:

The Goodness of fit determines how the line of regression fits the set of observations. The process of finding the best model out of various models is called **optimization**. It can be achieved by below method:

1. R-squared method:

- R-squared is a statistical method that determines the goodness of fit.
- It measures the strength of the relationship between the dependent and independent variables on a scale of 0-100%.
- The high value of R-square determines the less difference between the predicted values and actual values and hence represents a good model.
- It is also called a **coefficient of determination,** or **coefficient of multiple determination** for multiple regression.
- It can be calculated from the below formula:

$$\text{R-squared} = \frac{\text{Explained variation}}{\text{Total Variation}}$$

Assumptions of Linear Regression

Below are some important assumptions of Linear Regression. These are some formal checks while building a Linear Regression model, which ensures to get the best possible result from the given dataset.

- **Linear relationship between the features and target:**
  Linear regression assumes the linear relationship between the dependent and independent variables.
- **Small or no multicollinearity between the features:**
  Multicollinearity means high-correlation between the independent variables. Due to multicollinearity, it may difficult to find the true relationship between the predictors and target variables. Or we can say, it is difficult to determine which predictor variable is affecting the target variable and which is not. So, the model assumes either little or no multicollinearity between the features or independent variables.
- **Homoscedasticity Assumption:**
  Homoscedasticity is a situation when the error term is the same for all the values of independent variables. With homoscedasticity, there should be no clear pattern distribution of data in the scatter plot.
- **Normal distribution of error terms:**
  Linear regression assumes that the error term should follow the normal distribution pattern. If error terms are not normally distributed, then confidence intervals will become either too wide or too narrow, which may cause difficulties in finding coefficients.
  It can be checked using the **q-q plot**. If the plot shows a straight line without any deviation, which means the error is normally distributed.
- **No autocorrelations:**
  The linear regression model assumes no autocorrelation in error terms. If there will be any correlation in the error term, then it will drastically reduce the accuracy of the model. Autocorrelation usually occurs if there is a dependency between residual errors.

FAQ'S

1. What is meant by Regression?
2. What is meant by Linear Regression?
3. What is cost function?
4. What is standard deviation and normalization?
5. What is difference between classification and regression?

Outcome:

With the completion of this assignment the students will be able to implement linear regression on given dataset.

Conclusion:

Assignment No:- 7

Title of Assignment:

Implement Naïve Bayes theorem to classify the English text

During assignment students will be able to:

- Learn Supervised Learning.
- Implementation of Naïve Bayes Classifier.

Prerequisites: Knowledge of python programming and machine learning libraries.

Concepts to be used: Naïve Bayes Classifier

Input: Text dataset

Output: Successful implementation of Naïve Bayes Classifier.

Relevant Theory / Literature Survey:

Naive Bayes

- Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.
- It is mainly used in *text classification* that includes a high-dimensional training dataset.
- Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
- It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.
- Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

Why is it called Naïve Bayes?

The Naïve Bayes algorithm is comprised of two words Naïve and Bayes, Which can be described as:

- **Naïve**: It is called Naïve because it assumes that the occurrence of a certain feature is independent of the occurrence of other features. Such as if the fruit is identified on the bases of color, shape, and taste, then red, spherical, and sweet fruit is recognized as an apple. Hence each feature individually contributes to identify that it is an apple without depending on each other.
- **Bayes**: It is called Bayes because it depends on the principle of Bayes' Theorem

.

Bayes' Theorem:

- Bayes' theorem is also known as **Bayes' Rule** or **Bayes' law**, which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.
- The formula for Bayes' theorem is given as:

$$P(A|B)= \frac{P(B|A)P(A)}{P(B)}$$

Where,

P(A|B) is Posterior probability: Probability of hypothesis A on the observed event B.

P(B|A) is Likelihood probability: Probability of the evidence given that the probability of a hypothesis is true.

P(A) is Prior Probability: Probability of hypothesis before observing the evidence.

P(B) is Marginal Probability: Probability of Evidence.

Working of Naïve Bayes' Classifier:

Working of Naïve Bayes' Classifier can be understood with the help of the below example:

Suppose we have a dataset of **weather conditions** and corresponding target variable "**Play**". So using this dataset we need to decide that whether we should play or not on a particular day according to the weather conditions. So to solve this problem, we need to follow the below steps:

1. Convert the given dataset into frequency tables.
2. Generate Likelihood table by finding the probabilities of given features.
3. Now, use Bayes theorem to calculate the posterior probability.

Problem: If the weather is sunny, then the Player should play or not?

Solution: To solve this, first consider the below dataset:

	Outlook	Play
0	Rainy	Yes
1	Sunny	Yes
2	Overcast	Yes
3	Overcast	Yes
4	Sunny	No
5	Rainy	Yes
6	Sunny	Yes
7	Overcast	Yes
8	Rainy	No
9	Sunny	No
10	Sunny	Yes
11	Rainy	No
12	Overcast	Yes
13	Overcast	Yes

Frequency table for the Weather Conditions:

Weather	Yes	No

Overcast	5	0
Rainy	2	2
Sunny	3	2
Total	10	5

Likelihood table weather condition:

Weather	**No**	**Yes**	
Overcast	0	5	5/14= 0.35
Rainy	2	2	4/14=0.29
Sunny	2	3	5/14=0.35
All	4/14=0.29	10/14=0.71	

Applying Bayes'theorem:

P(Yes|Sunny)= P(Sunny|Yes)*P(Yes)/P(Sunny)

P(Sunny|Yes)= 3/10= 0.3

P(Sunny)= 0.35

P(Yes)=0.71

So P(Yes|Sunny) = 0.3*0.71/0.35= **0.60**

P(No|Sunny)= P(Sunny|No)*P(No)/P(Sunny)

P(Sunny|NO)= 2/4=0.5

P(No)= 0.29

P(Sunny)= 0.35

So P(No|Sunny)= 0.5*0.29/0.35 = **0.41**

So as we can see from the above calculation that **P(Yes|Sunny)>P(No|Sunny)**

Hence on a Sunny day, Player can play the game.

Advantages of Naïve Bayes Classifier:

- Naïve Bayes is one of the fast and easy ML algorithms to predict a class of datasets.
- It can be used for Binary as well as Multi-class Classifications.
- It performs well in Multi-class predictions as compared to the other Algorithms.
- It is the most popular choice for **text classification problems**.

Disadvantages of Naïve Bayes Classifier:

- Naive Bayes assumes that all features are independent or unrelated, so it cannot learn the relationship between features.

Applications of Naïve Bayes Classifier:

- It is used for **Credit Scoring**.
- It is used in **medical data classification**.
- It can be used in **real-time predictions** because Naïve Bayes Classifier is an eager learner.
- It is used in Text classification such as **Spam filtering** and **Sentiment analysis**.

Types of Naïve Bayes Model:

There are three types of Naive Bayes Model, which are given below:

- **Gaussian**: The Gaussian model assumes that features follow a normal distribution. This means if predictors take continuous values instead of discrete, then the model assumes that these values are sampled from the Gaussian distribution.
- **Multinomial**: The Multinomial Naïve Bayes classifier is used when the data is multinomial distributed. It is primarily used for document classification problems, it means a particular document belongs to which category such as Sports, Politics,

education, etc.
The classifier uses the frequency of words for the predictors.

- **Bernoulli**: The Bernoulli classifier works similar to the Multinomial classifier, but the predictor variables are the independent Booleans variables. Such as if a particular word is present or not in a document. This model is also famous for document classification tasks.

How to use Naive Bayes for Text?

In our case, we can't feed in text directly to our classifier. Texts are huge, they have lots of words and various combinations so we use occurrences of words in a single text data(i.e. term frequency-tf) and how many times that word comes across all text data (i.e. inverse document frequency-idf) and then we can generate feature vectors.

Feature vectors representing text contains the probabilities of appearance of the words of the text within the texts of a given category so that the algorithm can compute the likelihood of that text's belonging to the category.

Feature Vector

Document ID	Word 1	Word 2
1	2	0
2	0	1
3	3	1
4	1	3

ID	Class
1	1
2	0
3	0
4	1

1- belongs
0- Does not belong

Once we are done finding likelihoods of various categories we find the category having maximum likelihood and there we have categorized our text!

For example, let's say you have a data set as :

Data

DOCUMENT ID	CONTENT	CLASS

DOCUMENT ID	CONTENT	CLASS
1	Nvidia GPU is the best in the world.	computer graphics
2	Nvidia is giving tough competition to AMD.	computer graphics
3	We were running our application with GTX 1050(High end GPU) still it didn't work then we realized the problem was with the OS.	computer graphics
4	GPU, Ganpat Pandey University, is located in Maharashtra.	not computer graphics
5	Please buy GPU from our store.	?

We can predict the class of last data by using Naive Bayes by considering the probability of important words,

Important words for com.graphics from the data can be considered as,
{Nvidia, GPU, AMD, GTX 1050}

All words will have feature vectors of theirs representing their number of occurences in each record,
Nvidia -> [1; 1; 0; 0]
and so on.
P(computer graphics | GPU) = P(GPU | computer graphics) * (P(computer graphics) / P(GPU))

P(computer graphics) =
Number of records having class as computer graphics / Number of total records = 3/4 = 0.75

P(GPU) = Number of records having GPU / Total number of reccords = 3/4 = 0.75

P(GPU | computer graphics) =
Number of records having computer graphics with GPU in it / Total number of computer graphics' records
= 2 /3 = 0.66

P(computer graphics | GPU) = 0.66 * (0.75/0.75) = 0.66
This is greater than 0.5 so we can predict that this text data will be belonging to computer graphics.

Here, we have considered only one word GPU that is important in our test data but in real data we'll consider feature vectors of all words and then compute the probability for the class based on the occurences of all the non primitive words.

The main advantage is that you can get really good results when data available is not much and computational resources are scarce.

Algorithm :

1. Data Pre-processing step

2. Fitting Naive Bayes to the Training set

3. Predicting the test result

4. Test accuracy of the result(Creation of Confusion matrix)

5. Visualizing the test set result.

FAQ's :

1. What is meant by Naïve Bias classification?
2. What is classification?
3. What is dependence & independent variable?
4. What is difference between classification and regression?

Outcome:

With the completion of this assignment the students will be able to implement

Naïve bayes on given dataset.

Conclusion:

Assignment No:- 8

Title of Assignment:

Unsupervised Learning: Implement K-Means Clustering and Hierarchical clustering on proper data set of your choice. Compare their Convergence

During assignment students will be able to:

- Learn Unsupervised Learning & Clustering.
- Implementation of k-means clustering.

Prerequisites: Knowledge of python programming and machine learning libraries.

Concepts to be used: Unsupervised Learning & Clustering.

Input: given dataset

Output: Successful implementation of k-means clustering.

Relevant Theory / Literature Survey:

1. What is Clustering?

Clustering analysis or Clustering is the task of grouping a set of an object in such a way object in the same group(called cluster) are more similar(in some sense or another to each other than to those in another group (clusters). It is the main task of exploratory data mining & a common technique for statistical data analysis used in many fields including Machine Learning, Pattern Recognition, Image Analysis, Information Retrieval, Bioinformatics, Data Compression, and Computer Graphics. Example: Library

Clustering algorithms:

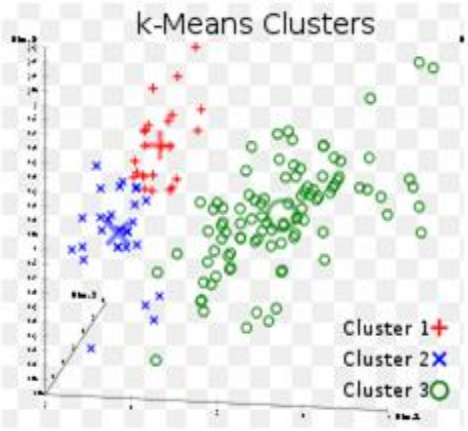
1. K-means clustering
2. Hierarchical Clustering

2. How does Clustering Algorithm work?

2.1: k-Mean Cluster.

2.1.1: What is k-Mean?

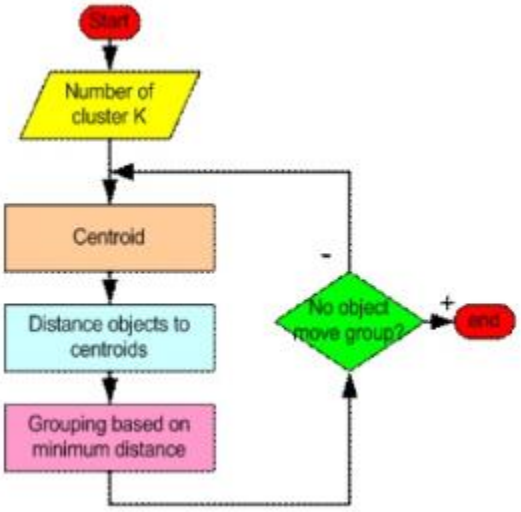
K-Means clustering aims to partition n observation into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster.



2.1.2: How k-Mean Cluster work?

The k-Means clustering algorithm attempt to split a given anonymous data set(a set of containing information as to class identity into a fixed number (k) of the cluster.

Initially, k number of the so-called centroid is chosen. A centroid is a data point (imaginary or real) at the center of the cluster.



Dataset Description:

This dataset has three attributes first is an item which is our target to make a cluster of similar items second and the third attribute is the informatics value of that item.

I	X1	X2
A	1	1
B	1	0
C	0	2
D	2	4
E	3	5

Now In the first step take any random row to let's suppose I take row 1 and row 3.

	X1	X2	Centroid Value
C1	1	1	(1, 1)
C2	0	2	(0, 2)

Now take these above centroid values to compare with observing the value of the respected row of our data by using the Euclidean Distance formula.

$$\text{Euclidean distance} = \sqrt{(\text{observe value} - \text{centroid value})^2 + (\text{observe value} - \text{centroid value})^2}$$

$$\text{Euclidean distance} = \sqrt{((X_x - X_1))^2 + (X_y - Y_1)^2}$$

Now let's solve one by one

Row 1 (A)

$C1=\sqrt{(1-1)^2+(1-1)^2}=1$

$C2=\sqrt{(1-0)^2+(1-2)^2}=1.4$

Row 2 (B)

$C1=\sqrt{(1-1)^2+(0-1)^2}=1$

$C2=\sqrt{(1-0)^2+(0-2)^2}=2.2$

Row 3 (C)

$C1=\sqrt{(0-1)^2+(2-1)^2}=1.4$

$C2=\sqrt{(0-0)^2+(2-2)^2}=0$

Row 4 (D)

$C1=\sqrt{(2-1)^2+(4-1)^2}=3.2$

$C2=\sqrt{(2-0)^2+(4-2)^2}=2.8$

Row 5 (E)

$C1=\sqrt{(3-1)^2+(5-1)^2}=4.5$

$C2=\sqrt{(3-0)^2+(5-2)^2}=4.2$

Now

I	C1=1	C2=2
A	0	1.4
B	1	2.2
C	1.4	0
D	3.2	2.8
E	4.5	4.2

Let’s say A and B are belong the Cluster 1 and C, D and E.

As we show in below table

I	C1=1	C2=2	Cluster
A	0	1.4	1
B	1	2.2	1
C	1.4	0	2
D	3.2	2.8	2
E	4.5	4.2	2

Now calculate the centroid of cluster 1 and 2 respectively and again calculate the closest mean until calculate when our centroid is repeated previous one.

I	X1	X2
A	1	1
B	1	0
C	0	2
D	2	4
E	3	5

Now find the Centroid of respected Cluster 1 and Cluster 2

$X1 = \frac{1+1}{2} = 1 \ , \frac{1+0}{2} = 2$

$X1 = (1, 0.5)$

$X2 = \frac{0+2+3}{3} = 1.7 \ , \frac{2+4+5}{3} = 3.7$

$X2 = (1.7,3.7)$

New Centroid

X1 = (1, 0.5)

X2 = (1.7,3.7)

Previous Centroid

X1 = (1, 1)

X2 = (0, 2)

If New Centroid Value is equal to previous Centroid Value then our cluster is final otherwise if not equal then repeat the step until new Centroid value is equal to previous Centroid value.

So in our case new centroid value is not equal to previous centroid.

Now recalculate cluster having the closest mean.

So

X1 = (1, 0.5)

X2 = (1.7,3.7)

Similarly procedure as we calculate above

I	X1	X2
A	0.5	2.7
B	0.5	3.7
C	1.8	2.4
D	3.6	0.5
E	4.9	1.9

So based on based one, A B and C belongs to cluster 1 & D and E from cluster 2.

I	X1	X2	Cluster
A	0.5	2.7	1
B	0.5	3.7	1
C	1.8	2.4	1
D	3.6	0.5	2
E	4.9	1.9	2

So mean of Cluster I and 2

X1(Cluster 1) = (0.7 , 1)

X1(Cluster 2) = (2.5 , 4.5)

New Centroid

X1 = (0.7 , 1)

X2 = (2.5 , 4.5)

Previous Centroid

X1 = (1, 0.5)

X2 = (1.7,3.7)

If New Centoid Value is equal to previous Centroid Value then our cluster is final otherwise if not equal then repeat the step until new Centroid value is equal to previous Centroid value .

So in our case new centroid value is not equal to previous centroid.

Now recalculate cluster having a closest mean similar step

I	X1	X2
A	0.5	2.7
B	0.5	3.7
C	1.8	2.4
D	3.6	0.5
E	4.9	1.9

So based on closest distance, A B and C belongs to cluster 1 & D and E from cluster 2.

I	X1	X2	Cluster
A	0.5	2.7	1
B	0.5	3.7	1
C	1.8	2.4	1
D	3.6	0.5	2
E	4.9	1.9	2

So mean of Cluster I and 2

X1(Cluster 1) = (0.7 , 1)

X1(Cluster 2) = (2.5 , 4.5)

New Centroid

X1 = (1, 0.5)

X2 = (1.7,3.7)

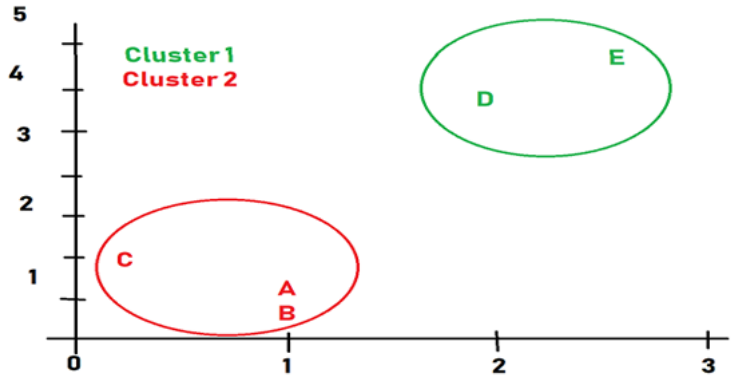
Previous Centroid

X1 = (1, 0.5)

X2 = (1.7,3.7)

So here we have New Centroid values is Equal to previous value and Hence our cluster are final. A, B and C are belong to cluster 1 and D and E are belong to Cluster 2.

As shown in fig:



Note: If you want this article check out my **academia.edu** profile.

2.2: Hierarchical Cluster.

2.2.1: What is Hierarchical?

Similar working like K-Mean clustering but the difference is that we create a tree structure.So

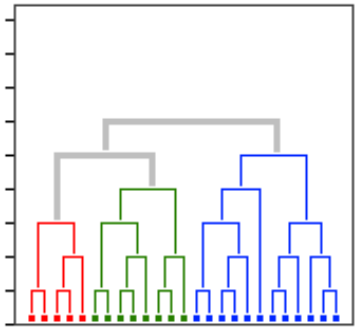
Bottom-up:

Initially, each point is a cluster.

Repeatedly combine the two “nearest” clusters into one.

Top-Down:

Start with one cluster and recursively split it.



Used for clustering similar things join and make a hierarchical clustering.

2.2.2: How Hierarchical Cluster work?

Agglomerative with Dendogram

Item	E	A	C	B	D
E	0	1	2	2	3
A	1	0	2	5	3
C	2	2	0	1	6
B	2	5	1	0	3
D	3	3	6	3	0

Consider only one part (lower triangle part)

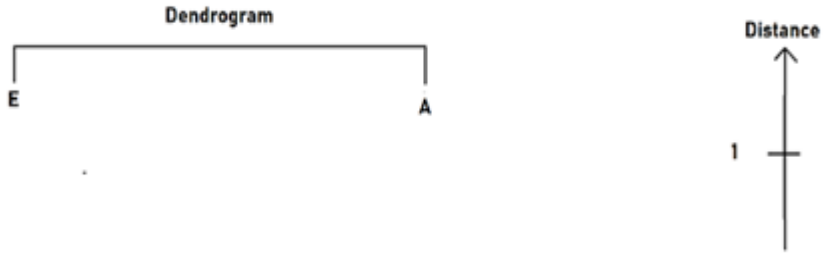
Item	E	A	C	B	D
E	0				
A	1	0			
C	2	2	0		
B	2	5	1	0	3
D	3	3	6	3	0

Now find the minimum distance

Minimum distance = 1 for **E & A**

Hence merge EA

So



Now Find the minimum distance from EA to other

C = min [dist { (E, A), C}]

= min [dist(E, C) , dist(A, C)]

= min [2, 2]

= 2

B = min [dist { (E, A), B}]

= min [dist(E, B) , dist(A, C)]

= min [2, 5]

= 2

D = min [dist { (E, A), D}]

= min [dist(E, D) , dist(A, D)]

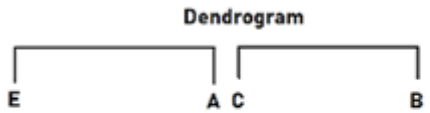
= min [3, 3]

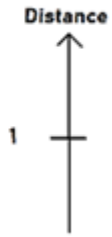
= 3

Item	EA	C	B	D
EA	0			
C	2	0		
B	2	1	0	3
D	3	6	3	0

In this minimum distance = 1

So for CB





Now again find the minimum distance from CB to another point as we find above So

EA = min [dist { (E, A), (C, B}]

= min [dist { (E, C) , (E, B) , (A, C), (A, B) }]

= min dist[2, 2, 2, 5]

= 2

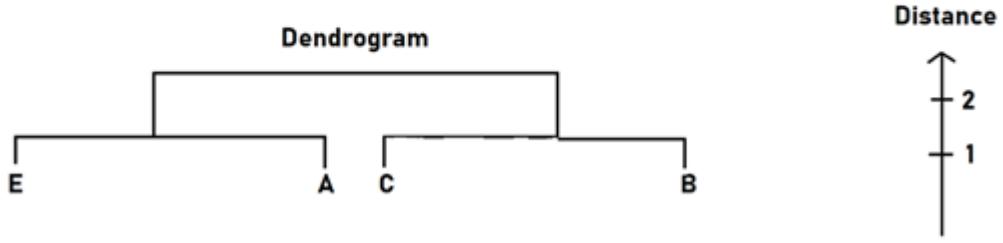
D = min [dist { (C, B), D}]

= min [dist{(C, D) , (B, D)}]

= min dist[6, 3]

= 3

Item	EA	CB	D
EA	0		
CB	2	0	
D	3	6	0



Now minum distance = 2 for EA , and CB from above table

D = min(dist [(E, A , C, B) , D]

= min dist [(E, D), (A, D). (C, D) , (B, D)

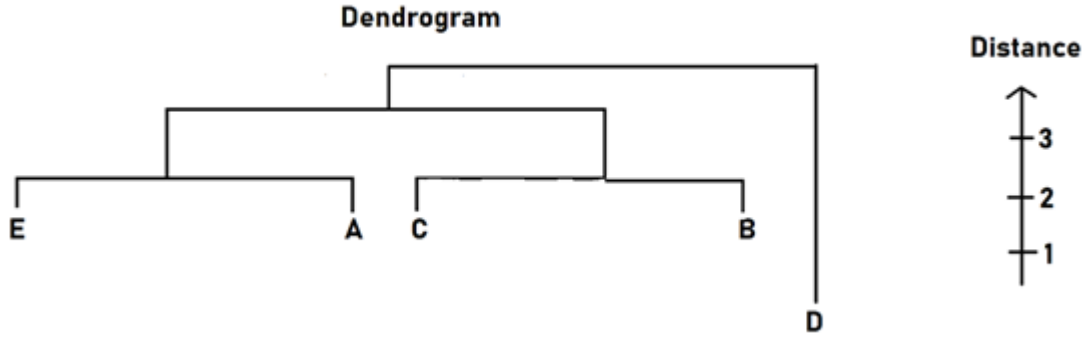
= min dist [3, 3, 6, 3]

= 3

So

Item	EA, CB	D
EA	0	
D	3	0

So finally are dendogram are final



Algorithm : K- Means

1. Data Preprocessing

2. Import the Libraries

3. Import the dataset

4. Elbow method to find the optimal number of clusters

5. Applying kMeans to the mall dataset and Visualizing the Clusters.

6. Init the Model

7. Fit and Predict the Dataset

8. Visualizing the Result

Algorithm : Hierarchical clustering

1. Data Preprocessing

2. Import the Libraries

3. Import the dataset

4. Use the dendogram method to find the optimal number of clusters

5. Applying Hierarchical to the and Visualizing the Clusters

6. Import and Init the model

7. Fit and Predict the Dataset

8. Visualizing the Result

FAQ'S

1. What is meant by Unsupervised learning?
2. What is meant by clustering?
3. What is k-means clustering?

Outcome:

With the completion of this assignment the students will be able to implement k-means clustering on given dataset.

Conclusion:

Assignment No:- 9

Title of Assignment:

Implement Support Vector Machine algorithm using suitable dataset.

During assignment students will be able to:

- Learn Classification.
- Implementation of Support Vector Machine.

Prerequisites: Knowledge of python programming and machine learning libraries.

Concepts to be used: Classification.

Input: Any dataset

Output: Successful implementation of Support Vector Machine (SVM)

Relevant Theory / Literature Survey:

SVM offers very high accuracy compared to other classifiers such as logistic regression, and decision trees. It is known for its kernel trick to handle nonlinear input spaces. It is used in a variety of applications such as face detection, intrusion detection, classification of emails, news articles and web pages, classification of genes, and handwriting recognition.

SVM is an exciting algorithm and the concepts are relatively simple. The classifier separates data points using a hyperplane with the largest amount of margin. That's why an SVM classifier is also known as a discriminative classifier. SVM finds an optimal hyperplane which helps in classifying new data points.

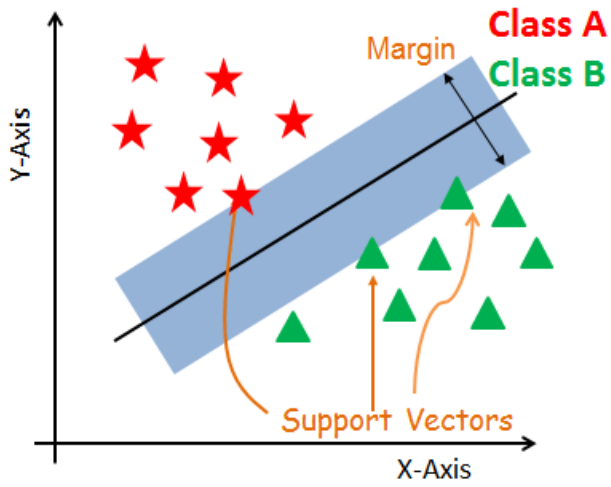
In this practical, you are going to cover following topics:

- Support Vector Machines
- How does it work?
- Kernels
- Classifier building in Scikit-learn

- Tuning Hyperparameters
- Advantages and Disadvantages

Support Vector Machines

Generally, Support Vector Machines is considered to be a classification approach, it but can be employed in both types of classification and regression problems. It can easily handle multiple continuous and categorical variables. SVM constructs a hyperplane in multidimensional space to separate different classes. SVM generates optimal hyperplane in an iterative manner, which is used to minimize an error. The core idea of SVM is to find a maximum marginal hyperplane(MMH) that best divides the dataset into classes.



Support Vectors

Support vectors are the data points, which are closest to the hyperplane. These points will define the separating line better by calculating margins. These points are more relevant to the construction of the classifier.

Hyperplane

A hyperplane is a decision plane which separates between a set of objects having different class memberships.

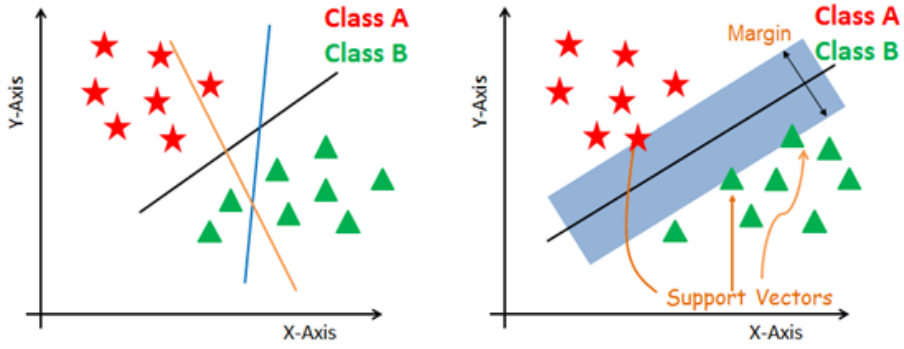
Margin

A margin is a gap between the two lines on the closest class points. This is calculated as the perpendicular distance from the line to support vectors or closest points. If the margin is larger in between the classes, then it is considered a good margin, a smaller margin is a bad margin.

How does SVM work?

The main objective is to segregate the given dataset in the best possible way. The distance between the either nearest points is known as the margin. The objective is to select a hyperplane with the maximum possible margin between support vectors in the given dataset. SVM searches for the maximum marginal hyperplane in the following steps:

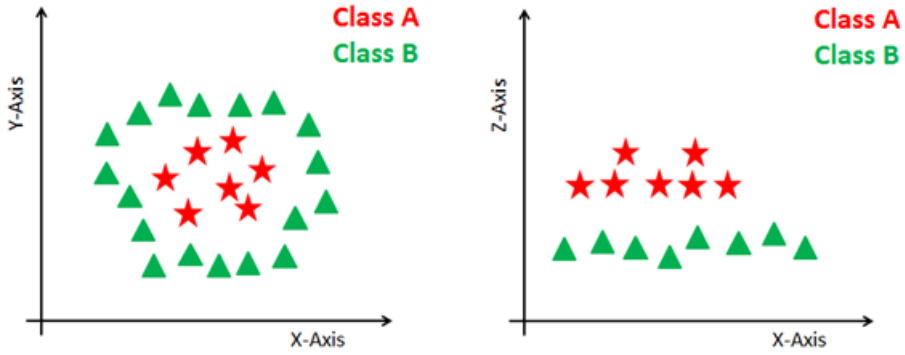
1. Generate hyperplanes which segregates the classes in the best way. Left-hand side figure showing three hyperplanes black, blue and orange. Here, the blue and orange have higher classification error, but the black is separating the two classes correctly.
2. Select the right hyperplane with the maximum segregation from the either nearest data points as shown in the right-hand side figure.



Dealing with non-linear and inseparable planes

Some problems can't be solved using linear hyperplane, as shown in the figure below (left-hand side).

In such situation, SVM uses a kernel trick to transform the input space to a higher dimensional space as shown on the right. The data points are plotted on the x-axis and z-axis (Z is the squared sum of both x and y: z=x^2=y^2). Now you can easily segregate these points using linear separation.



SVM Kernels

The SVM algorithm is implemented in practice using a kernel. A kernel transforms an input data space into the required form. SVM uses a technique called the kernel trick. Here, the kernel takes a low-dimensional input space and transforms it into a higher dimensional space. In other words, you can say that it converts nonseparable problem to separable problems by adding more dimension to it. It is most useful in non-linear separation problem. Kernel trick helps you to build a more accurate classifier.

- **Linear Kernel** A linear kernel can be used as normal dot product any two given observations. The product between two vectors is the sum of the multiplication of each pair of input values.

  F(x, xj) = sum(x.xj)

- **Polynomial Kernel** A polynomial kernel is a more generalized form of the linear kernel. The polynomial kernel can distinguish curved or nonlinear input space.

  F(x, xj) = (x.xj+1)^d

Where d is the degree of the polynomial. d=1 is similar to the linear transformation. The degree needs to be manually specified in the learning algorithm.

- **Radial Basis Function Kernel** The Radial basis function kernel is a popular kernel function commonly used in support vector machine classification. RBF can map an input space in infinite dimensional space.

  F(x, xj) = exp(-gamma * ||x - xj||^2)

Here gamma is a parameter, which ranges from 0 to 1. A higher value of gamma will perfectly fit the training dataset, which causes over-fitting. Gamma=0.1 is considered to be a good default value. The value of gamma needs to be manually specified in the learning algorithm.

Algorithm :

1. Loading Data

First load the required dataset you will use.

2. Exploring Data

After you have loaded the dataset, you might want to know a little bit more about it. You can check feature and target names.
You can also check the shape of the dataset using shape.
Check top 5 records of the feature set.

Let's take a look at the target set.

3. Splitting Data

Understand model performance, dividing the dataset into a training set and a test set is a good strategy. Split the dataset by using the function train_test_split(). you need to pass 3 parameters features, target, and test_set size. Additionally, use random_state to select records randomly.

4. Generating Model

Let's build support vector machine model. First, import the SVM module and create support vector classifier object by passing argument kernel as the linear kernel in SVC() function.

Then, fit your model on train set using fit() and perform prediction on the test set using predict().

5. Evaluating the Model

Let's estimate how accurately the classifier or model can predict the class .

Accuracy can be computed by comparing actual test set values and predicted values.

For further evaluation, check precision and recall of model.

6. Tuning Hyperparameters

- **Kernel**: The main function of the kernel is to transform the given dataset input data into the required form. There are various types of functions such as linear, polynomial, and radial basis function (RBF). Polynomial and RBF are useful for non-linear hyperplane. Polynomial and RBF kernels compute the separation line in the higher dimension. In some of the applications, it is suggested to use a more complex kernel to separate the classes that are curved or nonlinear. This transformation can lead to more accurate classifiers.

- **Regularization**: Regularization parameter in python's Scikit-learn C parameter used to maintain regularization. Here C is the penalty parameter, which represents misclassification or error term. The misclassification or error term tells the SVM optimization how much error is bearable. This is how you can control the trade-off between decision boundary and misclassification term. A smaller value of C creates a small-margin hyperplane and a larger value of C creates a larger-margin hyperplane.

- **Gamma**: A lower value of Gamma will loosely fit the training dataset, whereas a higher value of gamma will exactly fit the training dataset, which causes over-fitting. In other words, you can say a low value of gamma considers only nearby points in calculating the separation line, while the a value of gamma considers all the data points in the calculation of the separation line.

Advantages

SVM Classifiers offer good accuracy and perform faster prediction compared to Naïve Bayes algorithm. They also use less memory because they use a subset of training points in the decision phase. SVM works well with a clear margin of separation and with high dimensional space.

Disadvantages

SVM is not suitable for large datasets because of its high training time and it also takes more time in training compared to Naïve Bayes. It works poorly with overlapping classes and is also sensitive to the type of kernel used.

FAQ'S

1. What is meant by Support Vector?
2. What is meant by Hyperplane?
3. What is kernal?
4. What is gamma in SVM?
5. Discuss pros and cons of SVM?

Outcome:

With the completion of this assignment the students will be able to implement Support Vector Machine on given dataset.

Conclusion:

Assignment No:- 10

Title of Assignment:

Write a program to demonstrate the working of the decision tree based ID3 algorithm. Use an appropriate data set for building the decision tree and apply this knowledge to classify a new sample.

During assignment students will be able to:

- Learn Classification.
- Implementation of Decision tree algorithm .

Prerequisites: Knowledge of python programming and machine learning libraries.

Concepts to be used: Classification.

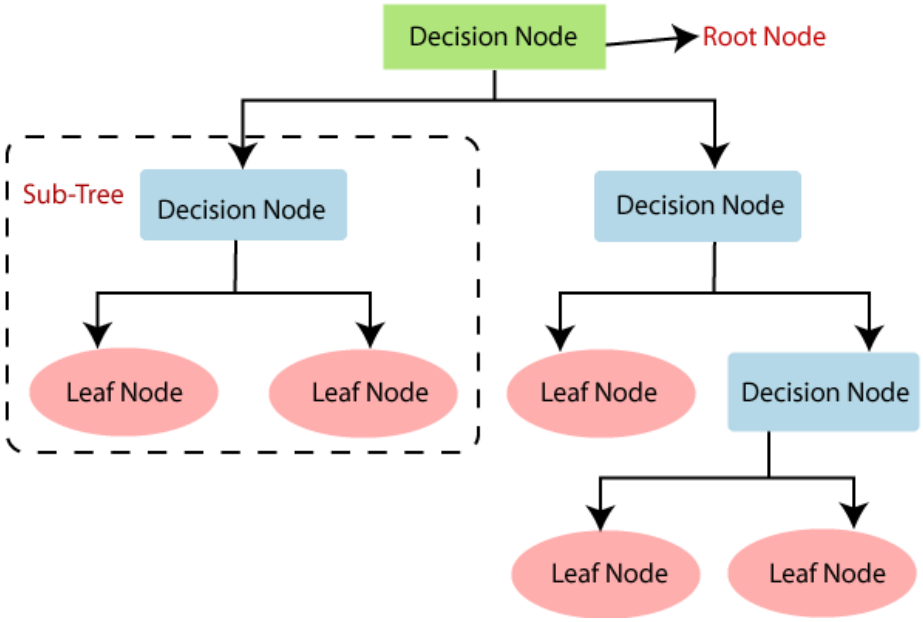
Input: Any dataset

Output: Successful implementation of Decision Tree Algorithm (ID3)

Relevant Theory / Literature Survey:

- Decision Tree is a **Supervised learning technique** that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where **internal nodes represent the features of a dataset, branches represent the decision rules** and **each leaf node represents the outcome.**
- In a Decision tree, there are two nodes, which are the **Decision Node** and **Leaf Node.** Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
- The decisions or the test are performed on the basis of features of the given dataset.
- **It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.**
- It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.

- In order to build a tree, we use the **CART algorithm,** which stands for **Classification and Regression Tree algorithm.**
- A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.
- Below diagram explains the general structure of a decision tree:



Why use Decision Trees?

There are various algorithms in Machine learning, so choosing the best algorithm for the given dataset and problem is the main point to remember while creating a machine learning model. Below are the two reasons for using the Decision tree:

- Decision Trees usually mimic human thinking ability while making a decision, so it is easy to understand.
- The logic behind the decision tree can be easily understood because it shows a tree-like structure.

Decision Tree Terminologies

Root Node: Root node is from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.

Leaf Node: Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.

Splitting: Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.

Branch/Sub Tree: A tree formed by splitting the tree.

Pruning: Pruning is the process of removing the unwanted branches from the tree.

Parent/Child node: The root node of the tree is called the parent node, and other nodes are called the child nodes.

How does the Decision Tree algorithm Work?

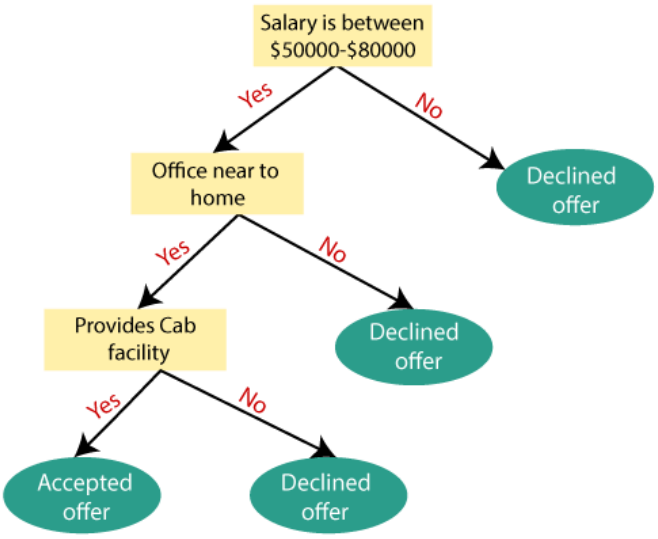
In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the values of root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node.

For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further. It continues the process until it reaches the leaf node of the tree. The complete process can be better understood using the below algorithm:

1. **Step-1:** Begin the tree with the root node, says S, which contains the complete dataset.
2. **Step-2:** Find the best attribute in the dataset using **Attribute Selection Measure (ASM).**
3. **Step-3:** Divide the S into subsets that contains possible values for the best attributes.
4. **Step-4:** Generate the decision tree node, which contains the best attribute.
5. **Step-5:** Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

Example: Suppose there is a candidate who has a job offer and wants to decide whether he should accept the offer or Not. So, to solve this problem, the decision tree starts with the root node (Salary attribute by ASM). The root node splits further into the next

decision node (distance from the office) and one leaf node based on the corresponding labels. The next decision node further gets split into one decision node (Cab facility) and one leaf node. Finally, the decision node splits into two leaf nodes (Accepted offers and Declined offer). Consider the below diagram:



Attribute Selection Measures

While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes. So, to solve such problems there is a technique which is called as **Attribute selection measure or ASM.** By this measurement, we can easily select the best attribute for the nodes of the tree. There are two popular techniques for ASM, which are:

- **Information Gain**
- **Gini Index**

1. Information Gain:

- Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an attribute.
- It calculates how much information a feature provides us about a class.

- According to the value of information gain, we split the node and build the decision tree.
- A decision tree algorithm always tries to maximize the value of information gain, and a node/attribute having the highest information gain is split first. It can be calculated using the below formula:

1. Information Gain= Entropy(S)- [(Weighted Avg) *Entropy(each feature)

Entropy: Entropy is a metric to measure the impurity in a given attribute. It specifies randomness in data. Entropy can be calculated as:

Entropy(s)= -P(yes)log2 P(yes)- P(no) log2 P(no)

Where,

- **S= Total number of samples**
- **P(yes)= probability of yes**
- **P(no)= probability of no**

2. Gini Index:

- Gini index is a measure of impurity or purity used while creating a decision tree in the CART(Classification and Regression Tree) algorithm.
- An attribute with the low Gini index should be preferred as compared to the high Gini index.
- It only creates binary splits, and the CART algorithm uses the Gini index to create binary splits.
- Gini index can be calculated using the below formula:

Gini Index= $1- \sum_j P_j^2$

Pruning: Getting an Optimal Decision tree

Pruning is a process of deleting the unnecessary nodes from a tree in order to get the optimal decision tree.

A too-large tree increases the risk of overfitting, and a small tree may not capture all the important features of the dataset. Therefore, a technique that decreases the size of the learning tree without reducing accuracy is known as Pruning. There are mainly two types of tree **pruning** technology used:

- **Cost Complexity Pruning**
- **Reduced Error Pruning.**

Advantages of the Decision Tree

- It is simple to understand as it follows the same process which a human follow while making any decision in real-life.
- It can be very useful for solving decision-related problems.
- It helps to think about all the possible outcomes for a problem.
- There is less requirement of data cleaning compared to other algorithms.

Disadvantages of the Decision Tree

- The decision tree contains lots of layers, which makes it complex.
- It may have an overfitting issue, which can be resolved using the **Random Forest algorithm.**
- For more class labels, the computational complexity of the decision tree may increase.

FAQ'S

1. What is the decision tree algorithm?
2. List down some popular algorithms used for deriving decision trees along with their attribute selection measures.
3. List down the attribute selection measures used by the Id3 algorithm to construct a decision tree.
4. How does a decision tree handle missing attributes?

Outcome:

With the completion of this assignment the students will be able to implement Support Vector Machine on given dataset.

Conclusion:

Assignment No:- 11

Title of Assignment:

Implement movie recommendation system using python.

During assignment students will be able to:

- Learn Supervised Learning.
- Implementation of Recommendation System.

Prerequisites: Knowledge of python programming and machine learning libraries.

Concepts to be used: Recommendation

Input: Any dataset

Output: Successful implementation of Recommendation Systems.

Relevant Theory / Literature Survey:

What Is Recommendation System?

A recommendation system is a subclass of Information filtering Systems that seeks to predict the rating or the preference a user might give to an item. In simple words, it is an algorithm that suggests relevant items to users. Eg: In the case of Netflix which movie to watch, In the case of e-commerce which product to buy, or In the case of kindle which book to read, etc.

Use-Cases Of Recommendation System

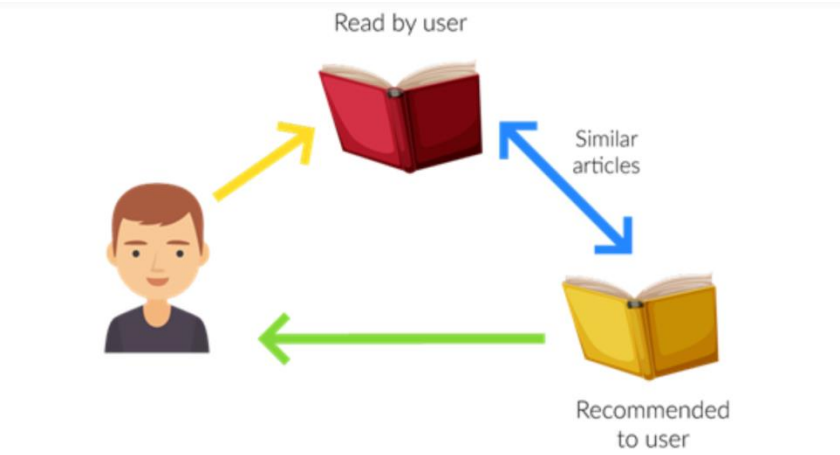
There are many use-cases of it. Some are

A. **Personalized Content:** Helps to Improve the on-site experience by creating dynamic recommendations for different kinds of audiences like Netflix does.

B. **Better Product search experience**: Helps to categories the product based on their features. Eg: Material, Season, etc.

Types of Recommendation System:

1. Content-Based Filtering:



In this type of recommendation system, relevant items are shown using the content of the previously searched items by the users. Here content refers to the attribute/tag of the product that the user like. In this type of system, products are tagged using certain keywords, then the system tries to understand what the user wants and it looks in its database and finally tries to recommend different products that the user wants.

Let us take an example of the movie recommendation system where every movie is associated with its genres which in the above case is referred to as tag/attributes. Now let assume user A comes and initially system don't have any data about user A. so initially, the system tries to recommend the popular movies to the users or the system tries to get some information of the user by getting a form filled by the user. After some time, users might have given a rating to some of the movies like it gives a good rating to movies based on the action genre and a bad rating to the movies based on the anime genre. So here system

recommends action movies to the users. But here you can't say that the user dislikes animation movies because maybe the user dislikes that movie due to some other reason like acting or story but actually likes animation movies and needs more data in this case.

Advantage

- Model doesn't need data of other users since recommendations are specific to a single user.
- It makes it easier to scale to a large number of users.
- The model can Capture the specific Interests of the user and can recommend items that very few other users are interested in.

Disadvantage

- Feature representation of items is hand-engineered to some extent, this tech requires a lot of domain knowledge.
- The model can only make recommendations based on the existing interest of a user. In other words, the model has limited ability to expand on the user's existing interests.

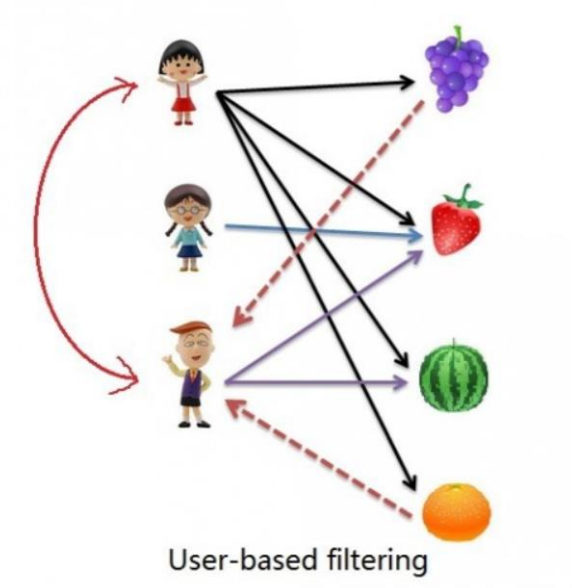
2. Collaborative Based Filtering

Recommending the new items to users based on the interest and preference of other similar users is basically collaborative-based filtering. For eg:- When we shop on Amazon it recommends new products saying *"Customer who brought this also brought"* as shown below.

the disadvantage of content-based filtering as it will use the user Interaction instead of content from the items used by the users. For this, it only needs the historical performance of the users. Based on the historical data, with the assumption that user who has agreed in past tends to also agree in future.

There are 2 types of collaborative filtering:-

A. User-Based Collaborative Filtering

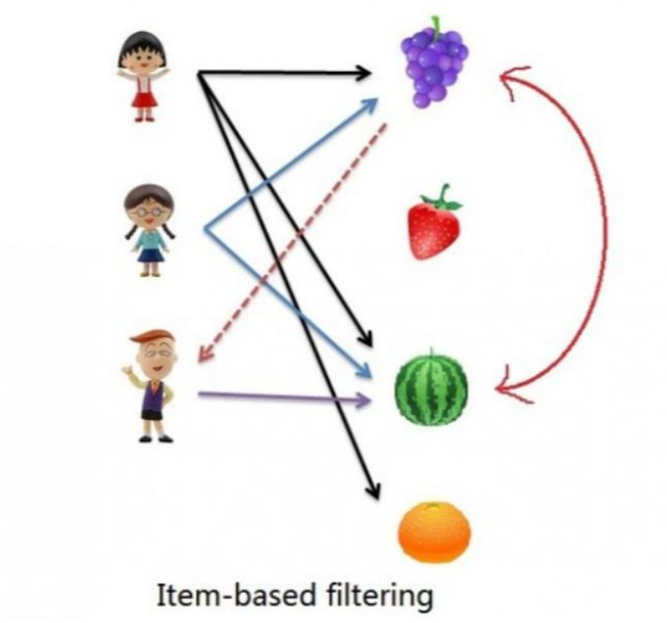


Rating of the item is done using the rating of neighbouring users. In simple words, It is based on the notion of users' similarity.

Let see an example. On the left side, you can see a picture where 3 children named A, B, C, and 4 fruits i.e, grapes, strawberry, watermelon, and orange respectively.

Based on the image let assume A purchased all 4 fruits, B purchased only strawberry and C purchased strawberry as well as watermelon. Here A & C are similar kinds of users because of this C will be recommended Grapes and Orange as shown in dotted line.

B. Item-Based Collaborative Filtering



The rating of the item is predicted using the user's own rating on neighbouring items. In simple words, it is based on the notion of item similarity.

Let us see with an example as told above about users and items. Here the only difference is that we see similar items, not similar users like if you see grapes and watermelon you will realize that watermelon is purchased by all of them but grapes are purchased by Children A & B. Hence Children C is being recommended grapes.

Now after understanding both of them you may be wondering which to use when. Here is the solution if No. of items is greater than No. of users go with user-based collaborative filtering as it will reduce the computation power and If No. of users is greater than No. of items go with item-based collaborative filtering. For Example, Amazon has lakhs of items to sell but has billions of customers. Hence Amazon uses item-based collaborative filtering because of less no. of products as compared to its customers.

Advantage

- It works well even if the data is small.
- This model helps the users to discover a new interest in a given item but the model might still recommend it because similar users are interested in that item.
- No need for Domain Knowledge

Disadvantage

- It cannot handle new items because the model doesn't get trained on the newly added items in the database. This problem is known as Cold Start Problem.
- Side Feature Doesn't have much importance. Here Side features can be actor name or releasing year in the context of movie recommendation.

Implementation:

Collaborative filtering makes automatic predictions (filtering) about a user's interests by collecting preferences or taste information from many users. Recommendations have existed for a long time now with their models based on various techniques like weighted averages, correlation, machine-learning, deep-learning, etc.

The Movielens 20M dataset has over 20 Million Movie Ratings and Tagging Activities Since 1995. In this experiment , we'll retrieve information from movie.csv & rating.csv files. We train our model using Python libraries Pandas, Seaborn, Scikit-learn, and SciPy.

Step-by-step Algorithm:

1. Importing our datasets & merging them to create a Pandas DataFrame
2. Adding necessary features to analyze data
3. Visualizing data to build on our analysis using Seaborn
4. Eliminating data by setting a threshold
5. Creating a Pivot Table with users as indices and movies as columns
6. Creating a kNN model to print 5 recommendations similar to each movie

FAQ'S

1. What are recommendation systems ?
2. What are knowledge based recommender systems different from collaborative and content based recommender systems?
3. What are the basic components of content based systems?
4. What is a model based collaborative approach?
5. What is the difference between collaborative and content based recommender systems?

Outcome:

With the completion of this assignment the students will be able to implement recommender system on given dataset.

Conclusion: