|  |
| --- |
| **ZEAL EDUCATION SOCIETY’S** |
| **ZEAL COLLEGE OF ENGINEERING AND RESEARCH** |
| **DEPARTMENT OF COMPUTER ENGINEERING** |
|  |
| **LABORATORY MANUAL** |
| **(As per syllabus of SPPU with effect from 2019)** |
| **LABORATORY PRACTICE-III (410254)**  **Part-A Machine Learning** |
| **Semester- VIII** |
| **(2021-22)** |

**Assignment No**: 01

**1.1 Title**

Assignment based on Linear Regression.

**1.2 Problem Definition:**

The following table shows the results of a recently conducted study on the correlation of the number of hours spent driving with the risk of developing acute backache. Find the equation of the best fit line for this data.

**1.3 Prerequisite:**

Basic of Python, Data Mining Algorithm

**1.4 Software Requirements:**

Anaconda with Python 3.7

**1.5 Hardware Requirement:**

PIV, 2GB RAM, 500 GB HDD, Lenovo A13-4089Model.

**1.6 Learning Objectives:**

Learn Linear Regression for given different Dataset

**1.7 Outcomes:**

After completion of this assignment students are able to understand the How to find the correlation between to Two variable, How to Calculate Accuracy of the Linear Model and how to plot graph using matplotlib.

**1.8 Theory Concepts:**

**1.8.1 Linear Regression**

Regression analysis is used in stats to find trends in data. For example, you might guess that there’s a connection between how much you eat and how much you weight; regression analysis can help you quantify that.

**What is Linear Regression?**

In a cause and effect relationship, the **independent variable** is the cause, and the **dependent variable** is the effect. **Least squares linear regression** is a method for predicting the value of a dependent variable *Y*, based on the value of an independent variable *X*.

## Prerequisites for Regression

Simple linear regression is appropriate when the following conditions are satisfied.

* The dependent variable Y has a linear relationship to the independent variable X. To check this, make sure that the XY [scatterplot](https://stattrek.com/Help/Glossary.aspx?Target=Scatterplot) is linear and that the [residual plot](https://stattrek.com/Help/Glossary.aspx?Target=Residual%20plot) shows a random pattern. For each value of X, the probability distribution of Y has the same standard deviation σ.
* When this condition is satisfied, the variability of the residuals will be relatively constant across all values of X, which is easily checked in a residual plot.
* For any given value of X,
  + The Y values are independent, as indicated by a random pattern on the residual plot.
  + The Y values are roughly normally distributed (i.e., [symmetric](https://stattrek.com/Help/Glossary.aspx?Target=Symmetry) and [unimodal](https://stattrek.com/Help/Glossary.aspx?Target=Unimodal%20distribution)). A little [skewness](https://stattrek.com/Help/Glossary.aspx?Target=Skewness) is ok if the sample size is large. A [histogram](https://stattrek.com/Help/Glossary.aspx?Target=Histogram) or a [dotplot](https://stattrek.com/Help/Glossary.aspx?Target=Dotplot) will show the shape of the distribution.

## The Least Squares Regression Line

Linear regression finds the straight line, called the **least squares regression line** or LSRL, that best represents observations in a [bivariate](https://stattrek.com/Help/Glossary.aspx?Target=Bivariate%20data) data set. Suppose Y is a dependent variable, and X is an independent variable. The population regression line is:

Y = Β0 + Β1X

where Β0 is a constant, Β1 is the regression coefficient, X is the value of the independent variable, and Y is the value of the dependent variable.

Given a random sample of observations, the population regression line is estimated by:

ŷ = b0 + b1x

where b0 is a constant, b1 is the regression coefficient, x is the value of the independent variable, and ŷ is the predictedvalue of the dependent variable.

## How to Define a Regression Line

Normally, you will use a computational tool - a software package (e.g., Excel) or a [graphing calculator](https://stattrek.com/AP/Calculator.aspx) - to find b0 and b1. You enter the X and Y values into your program or calculator, and the tool solves for each parameter.

In the unlikely event that you find yourself on a desert island without a computer or a graphing calculator, you can solve for b0 and b1 "by hand". Here are the equations.

b1 = Σ [ (xi - x)(yi - y) ] / Σ [ (xi - x)2]

b1 = r \* (sy / sx)

b0 = y - b1 \* x

where b0 is the constant in the regression equation, b1 is the regression coefficient, r is the correlation between x and y, xi is the X value of observation i, yi is the Y value of observation i, x is the mean of X, y is the mean of Y, sx is the standard deviation of X, and sy is the standard deviation of Y.

## Properties of the Regression Line

When the regression parameters (b0 and b1) are defined as described above, the regression line has the following properties.

* The line minimizes the sum of squared differences between observed values (the y values) and predicted values (the ŷ values computed from the regression equation).
* The regression line passes through the mean of the X values (x) and through the mean of the Y values (y).
* The regression constant (b0) is equal to the [y intercept](https://stattrek.com/Help/Glossary.aspx?Target=Y%20intercept) of the regression line.
* The regression coefficient (b1) is the average change in the dependent variable (Y) for a 1-unit change in the independent variable (X). It is the [slope](https://stattrek.com/Help/Glossary.aspx?Target=Slope) of the regression line.

The least squares regression line is the only straight line that has all of these properties.

## The Coefficient of Determination

The **coefficient of determination** (denoted by R2) is a key output of regression analysis. It is interpreted as the proportion of the variance in the dependent variable that is predictable from the independent variable.

* The coefficient of determination ranges from 0 to 1.
* An R2 of 0 means that the dependent variable cannot be predicted from the independent variable.
* An R2 of 1 means the dependent variable can be predicted without error from the independent variable.
* An R2 between 0 and 1 indicates the extent to which the dependent variable is predictable. An R2 of 0.10 means that 10 percent of the variance in Y is predictable from X; an R2 of 0.20 means that 20 percent is predictable; and so on.

The formula for computing the coefficient of determination for a linear regression model with one independent variable is given below.

**Coefficient of determination.** The coefficient of determination (R2) for a linear regression model with one independent variable is:

R2 = { ( 1 / N ) \* Σ [ (xi - x) \* (yi - y) ]  
/ (σx \* σy ) }2

where N is the number of observations used to fit the model, Σ is the summation symbol, xi is the x value for observation i, x is the mean x value, yi is the y value for observation i, y is the mean y value, σx is the standard deviation of x, and σy is the standard deviation of y.

If you know the linear correlation (r) between two variables, then the coefficient of determination (R2) is easily computed using the following formula: R2 = r2.

## Standard Error

The **standard error** about the regression line (often denoted by SE) is a measure of the average amount that the regression equation over- or under-predicts. The higher the coefficient of determination, the lower the standard error; and the more accurate predictions are likely to be.

**Example**

Last year, five randomly selected students took a math aptitude test before they began their statistics course. The Statistics Department has three questions.

* What linear regression equation best predicts statistics performance, based on math aptitude scores?
* If a student made an 80 on the aptitude test, what grade would we expect her to make in statistics?
* How well does the regression equation fit the data?

## How to Find the Regression Equation

In the table below, the xi column shows scores on the aptitude test. Similarly, the yi column shows statistics grades. The last two columns show deviations scores - the difference between the student's score and the average score on each test. The last two rows show sums and mean scores that we will use to conduct the regression analysis.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Student** | **xi** | **yi** | **(xi-x)** | **(yi-y)** |
| 1 | 95 | 85 | 17 | 8 |
| 2 | 85 | 95 | 7 | 18 |
| 3 | 80 | 70 | 2 | -7 |
| 4 | 70 | 65 | -8 | -12 |
| 5 | 60 | 70 | -18 | -7 |
| **Sum** | 390 | 385 |  |  |
| **Mean** | 78 | 77 |  |  |

And for each student, we also need to compute the squares of the deviation scores (the last two columns in the table below).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Student** | **xi** | **yi** | **(xi-x)2** | **(yi-y)2** |
| 1 | 95 | 85 | 289 | 64 |
| 2 | 85 | 95 | 49 | 324 |
| 3 | 80 | 70 | 4 | 49 |
| 4 | 70 | 65 | 64 | 144 |
| 5 | 60 | 70 | 324 | 49 |
| **Sum** | 390 | 385 | 730 | 630 |
| **Mean** | 78 | 77 |  |  |

And finally, for each student, we need to compute the product of the deviation scores.

|  |  |  |  |
| --- | --- | --- | --- |
| **Student** | **xi** | **yi** | **(xi-x)(yi-y)** |
| 1 | 95 | 85 | 136 |
| 2 | 85 | 95 | 126 |
| 3 | 80 | 70 | -14 |
| 4 | 70 | 65 | 96 |
| 5 | 60 | 70 | 126 |
| **Sum** | 390 | 385 | 470 |
| **Mean** | 78 | 77 |  |

The regression equation is a linear equation of the form: ŷ = b0 + b1x . To conduct a regression analysis, we need to solve for b0 and b1. Computations are shown below. Notice that all of our inputs for the regression analysis come from the above three tables.

First, we solve for the regression coefficient (b1):

b1 = Σ [ (xi - x)(yi - y) ] / Σ [ (xi - x)2]

b1 = 470/730

b1 = 0.644

Once we know the value of the regression coefficient (b1), we can solve for the regression slope (b0):

b0 = y - b1 \* x

b0 = 77 - (0.644)(78)

b0 = 26.768

Therefore, the regression equation is: ŷ = 26.768 + 0.644x .

## How to Use the Regression Equation

Once you have the regression equation, using it is a snap. Choose a value for the independent variable (x), perform the computation, and you have an estimated value (ŷ) for the dependent variable.

In our example, the independent variable is the student's score on the aptitude test. The dependent variable is the student's statistics grade. If a student made an 80 on the aptitude test, the estimated statistics grade (ŷ) would be:

ŷ = b0 + b1x

ŷ = 26.768 + 0.644x = 26.768 + 0.644 \* 80

ŷ = 26.768 + 51.52 = 78.288

**Warning:** When you use a regression equation, do not use values for the independent variable that are outside the range of values used to create the equation. That is called **extrapolation**, and it can produce unreasonable estimates.

In this example, the aptitude test scores used to create the regression equation ranged from 60 to 95. Therefore, only use values inside that range to estimate statistics grades. Using values outside that range (less than 60 or greater than 95) is problematic.

## How to Find the Coefficient of Determination

Whenever you use a regression equation, you should ask how well the equation fits the data. One way to assess fit is to check the [coefficient of determination](https://stattrek.com/Help/Glossary.aspx?Target=Coefficient%20of%20determination), which can be computed from the following formula.

R2 = { ( 1 / N ) \* Σ [ (xi - x) \* (yi - y) ] / (σx \* σy ) }2

where N is the number of observations used to fit the model, Σ is the summation symbol, xi is the x value for observation i, x is the mean x value, yi is the y value for observation i, y is the mean y value, σx is the standard deviation of x, and σy is the standard deviation of y.

Computations for the sample problem of this lesson are shown below. We begin by computing the standard deviation of x (σx):

σx = sqrt [ Σ ( xi - x )2 / N ]

σx = sqrt( 730/5 ) = sqrt(146) = 12.083

Next, we find the standard deviation of y, (σy):

σy = sqrt [ Σ ( yi - y )2 / N ]

σy = sqrt( 630/5 ) = sqrt(126) = 11.225

And finally, we compute the coefficient of determination (R2):

R2 = { ( 1 / N ) \* Σ [ (xi - x) \* (yi - y) ] / (σx \* σy ) }2

R2 = [ ( 1/5 ) \* 470 / ( 12.083 \* 11.225 ) ]2

R2 = ( 94 / 135.632 )2 = ( 0.693 )2 = 0.48

A coefficient of determination equal to 0.48 indicates that about 48% of the variation in statistics grades (the[dependent variable](https://stattrek.com/Help/Glossary.aspx?Target=Dependent%20variable)) can be explained by the relationship to math aptitude scores (the [independent variable](https://stattrek.com/Help/Glossary.aspx?Target=Independent%20variable)). This would be considered a good fit to the data, in the sense that it would substantially improve an educator's ability to predict student performance in statistics class.

## Residuals

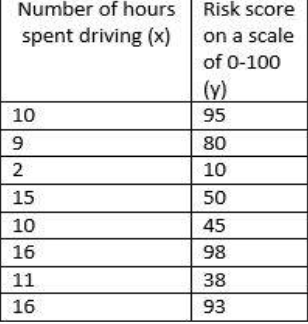
The difference between the observed value of the dependent variable (y) and the predicted value (ŷ) is called the **residual** (e). Each data point has one residual.

Residual = Observed value - Predicted value   
e = y - ŷ

Both the sum and the mean of the residuals are equal to zero. That is, Σ e = 0 and e = 0.

## What is Best Fit Line- A line of best fit  (or "trend" line) is a straight line that best represents the data on a scatter plot.  This line may pass through some of the points, none of the points, or all of the points.

## Given Dataset in Our Definition-



## Algorithm

## Import the Required Packages

## Read Given Dataset

1. Import the Linear Regression and Create object of it

## Find the Accuracy of Model using Score Function

1. Predict the value using Regressor Object

## Take input from user.

## Calculate the value of y

## Draw Scatter Plot

## 

## Important Function Used for Linear Regression

## coef\_ it is used to calculate slope in ML Model

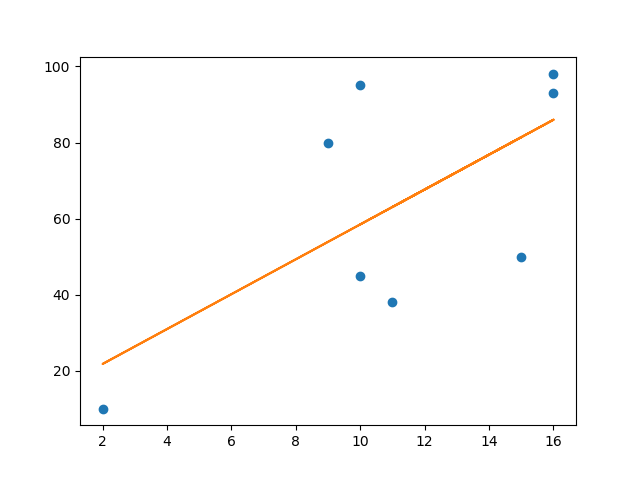
## Intercept\_ it is used to calculate intercept in ML Model

## fit - it shows the relationship between two varraible

## score – it display accuracy score of model

## Scatter Plot generated after Implementation of Code

## 



## 1.9 Conclusion

Thus we learn that to how to find the trend of data using X as Independent Variable and Y is and Dependent Variable by using Linear Regression.

## 1.10 Assignment Questions

* 1. **What is Linear Regression?**
  2. **What is Sigma?**
  3. **What is Mu?**
  4. **What is Probability of Distribution in term of Binominal and Frequency Distribution?**
  5. **What is Standard Deviation?**
  6. **What is Coefficient of Determination?**

Code:

**import** numpy **as** np

**import** pandas **as** pd

**import** matplotlib.pyplot **as** plt

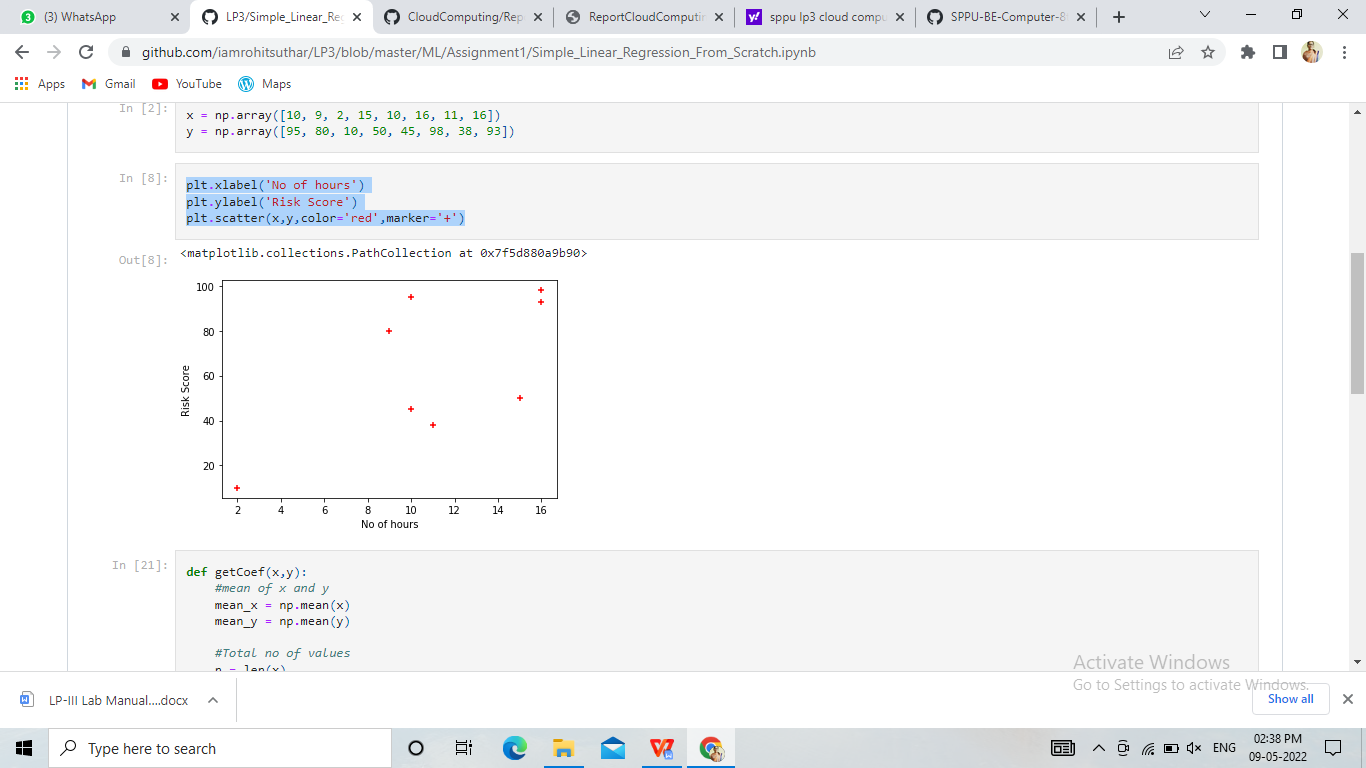
x **=** np**.**array([10, 9, 2, 15, 10, 16, 11, 16])

y **=** np**.**array([95, 80, 10, 50, 45, 98, 38, 93])

plt**.**xlabel('No of hours')

plt**.**ylabel('Risk Score')

plt**.**scatter(x,y,color**=**'red',marker**=**'+')



**def** getCoef(x,y):

*#mean of x and y*

mean\_x **=** np**.**mean(x)

mean\_y **=** np**.**mean(y)

*#Total no of values*

n **=** len(x)

*# formula to calculate b1 and b2*

numer **=** 0

denom **=** 0

**for** i **in** range(n):

numer **+=** (x[i] **-** mean\_x) **\*** (y[i] **-** mean\_y)

denom **+=** (x[i] **-** mean\_x) **\*\*** 2

b1 **=** numer **/** denom

b0 **=** mean\_y **-** (b1 **\*** mean\_x)

**return**(b0, b1)

coefs\_ **=** getCoef(x,y)

print("Coefficients")

print(coefs\_[0])

print(coefs\_[1])

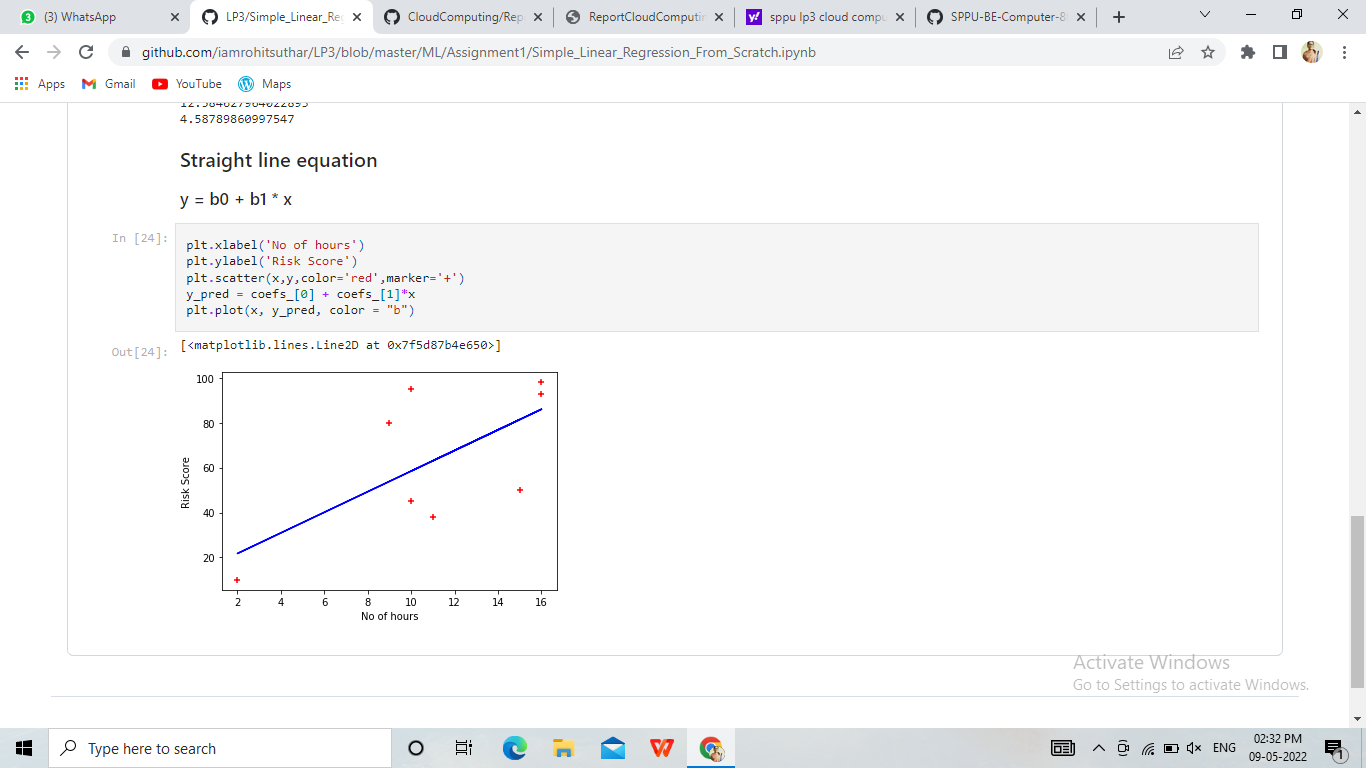
plt**.**xlabel('No of hours')

plt**.**ylabel('Risk Score')

plt**.**scatter(x,y,color**=**'red',marker**=**'+')

y\_pred **=** coefs\_[0] **+** coefs\_[1]**\***x

plt**.**plot(x, y\_pred, color **=** "b")



**Assignment No**: 02

**2.1 Title**

Assignment based on Decision Tree Classfier

**2.2 Problem Definition:**

A dataset collected in a cosmetics shop showing details of customers and whether or not they responded to a special offer to buy a new lip-stick is shown in table below. Use this dataset to build a decision tree, with Buys as the target variable, to help in buying lip-sticks in the future. Find the root node of decision tree. According to the decision tree you have made from previous training data set, what is the decision for the test data: [Age < 21, Income = Low, Gender = Female, Marital Status = Married]?

**2.3 Prerequisite:**

Basic of Python, Data Mining Algorithm, Concept of Decision Tree Classifier

**2.4 Software Requirements:**

Anaconda with Python 3.7

**2.5 Hardware Requirement:**

PIV, 2GB RAM, 500 GB HDD, Lenovo A13-4089Model.

**2.6 Learning Objectives:**

Learn How to Apply Decision Tree Classifier to find the root node of decision tree. According to the decision tree you have made from previous training data set.

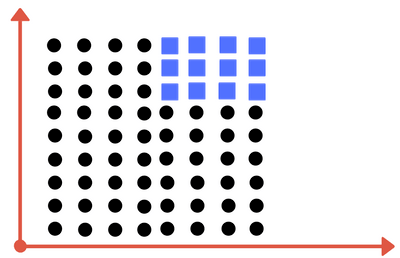
**2.7 Outcomes:**

After completion of this assignment students are able Implement code for Create Decision tree for given dataset and find the root node for same based on the given condition.

**2.8 Theory Concepts:**

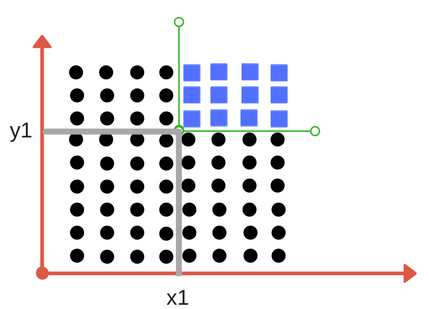
**2.8.1 Motivation**

Suppose we have following plot for two classes represented by black circle and blue squares. Is it possible to draw a single separation line ? Perhaps no.

****

## Can you draw single division line for these classes?

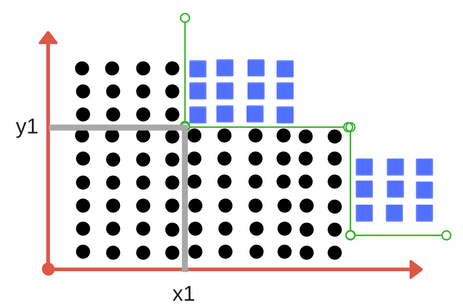
## We will need more than one line, to divide into classes. Something similar to following image:



We need two lines one for threshold of x and threshold for y.

We need two lines here one separating according to threshold value of ***x***and other for threshold value of ***y.***

*Decision Tree Classifier, repetitively divides the working area(plot) into sub part by identifying lines.*(repetitively because there may be two distant regions of same class divided by other as shown in image below).



## So when does it terminate?

## Either it has divided into classes that are pure (only containing members of single class )

## Some criteria of classifier attributes are met.

## Impurity-

## In above division, we had clear separation of classes. But what if we had following case?

## Impurity is when we have a traces of one class division into other. This can arise due to following reason

## We run out of available features to divide the class upon.

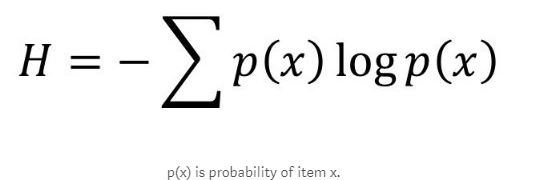
## We tolerate some percentage of impurity (we stop further division) for faster performance. (There is always trade off between accuracy and performance).

## For example in second case we may stop our division when we have x number of fewer number of elements left. This is also known as gini impurity.

## 

## 2. Entropy

## Entropy is degree of randomness of elements or in other words it is measure of impurity. Mathematically, it can be calculated with the help of probability of the items as:



## 

## It is negative summation of probability times the log of probability of item x.

## For example, if we have items as number of dice face occurrence in a throw event as 1123, the entropy is p(1) = 0.5 p(2) = 0.25 p(3) = 0.25 entropy = - (0.5 \* log(0.5)) - (0.25 \* log(0.25)) -(0.25 \* log(0.25) = 0.45

## 3. Information Gain

## Suppose we have multiple features to divide the current working set. What feature should we select for division? Perhaps one that gives us less impurity.

## Suppose we divide the classes into multiple branches as follows, the information gain at any node is defined as

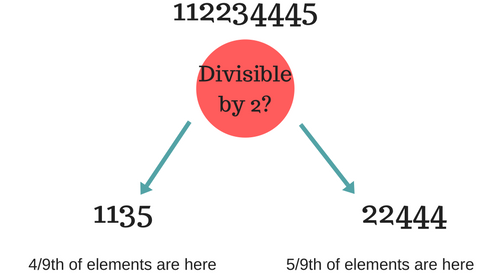
## Information Gain (n) = Entropy(x) — ([weighted average] \* entropy(children for feature))

## This need a bit explanation!

## Suppose we have following class to work with intially

## 112234445

## Suppose we divide them based on property: divisible by 2



Entropy at root level : 0.66  
Entropy of left child : 0.45 , weighted value = (4/9) \* 0.45 = 0.2  
Entropy of right child: 0.29 , weighted value = (5/9) \* 0.29 = 0.16

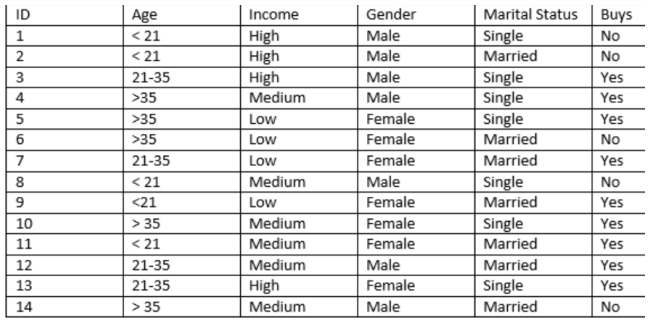
**Information Gain** = 0.66 - [0.2 + 0.16] = ***0.3***

*Check what information gain we get if we take decision as***prime number instead of divide by 2.**Which one is better for this case?

Decision tree at every stage selects the one that gives best information gain.

***When information gain is 0 means the feature does not divide the working set at all.***

## Given Data set in Our Definition



## What is the decision for the test data: [Age < 21, Income = Low, Gender = Female, Marital Status = Married]?

## Answer is Whether Yes or No??

1. Which of the attributes would be the root node.  
A. Age  
B. Income  
C. Gender  
D. Marital Status  
**Solution: C**

2. What is the decision for the test data [Age < 21, Income = Low, Gender = Female, Marital Status  
= Married]?  
A. Yes  
B. No  
**Solution: A**  
[Hints: construct the decision tree to answer these questions]

## Algorithm

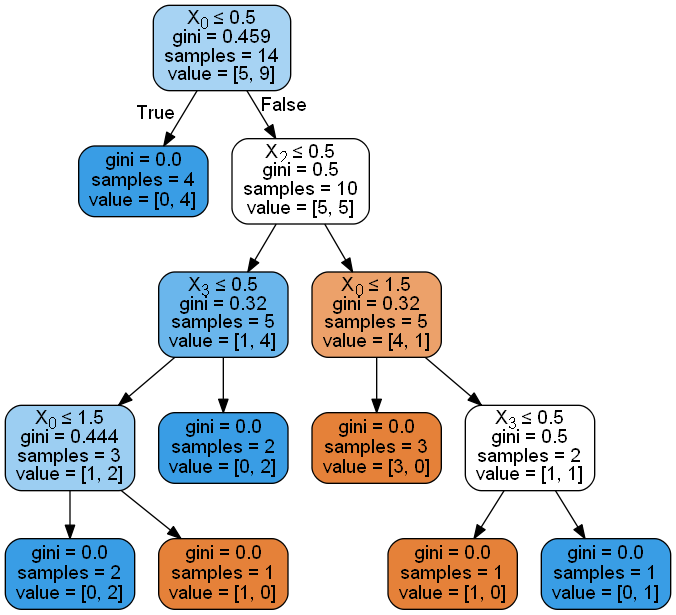
## Import the Required Packages

## Read Given Dataset

1. Perform the label Encoding Mean Convert String value into Numerical values
2. Import and Apply Decision Tree Classifier
3. Predict value for the given Expression like [Age < 21, Income = Low, Gender = Female, Marital Status = Married]? In encoding Values [1,1,0,0]
4. Import the packages for Create Decision Tree.
5. Check the Decision Tree Created based on Expression.

## 

## Decision Tree Generated after Implementation of Code



## 2.12 Conclusion

In this way we learn that to how to create Decision Tree based on given decision, Find the Root Node of the tree using Decision tree Classifier.

## 2.13 Assignment Questions

* 1. **What is gini index? What is the Formula for Calculate same?**
  2. **What is label Encoder? How it differ from One Hot Encoder?**
  3. What is Overfitting Problem while building a decision tree model?
  4. What is Pre-pruning and Post pruning Approach in decision tree model?
  5. **What are different advantages and disadvantages of decision tree algorithm?**

Code:

**import** numpy **as** np

**import** pandas **as** pd

**from** sklearn.preprocessing **import** LabelEncoder

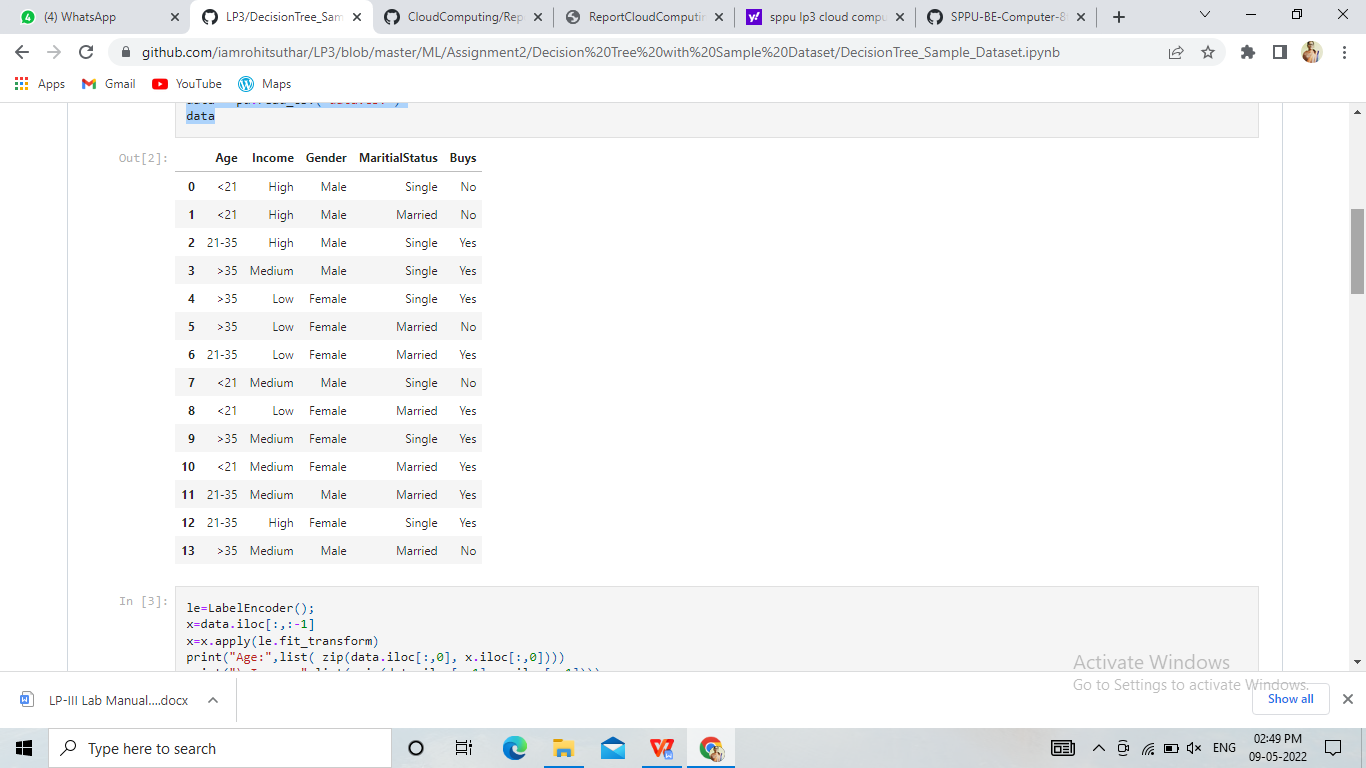
**from** sklearn.tree **import** DecisionTreeClassifier

**from** sklearn.tree **import** export\_graphviz

**from** IPython.display **import** Image

data **=** pd**.**read\_csv("data.csv")

data



le**=**LabelEncoder();x**=**data**.**iloc[:,:**-**1]

x**=**x**.**apply(le**.**fit\_transform)

print("Age:",list( zip(data**.**iloc[:,0], x**.**iloc[:,0])))

print("\nIncome:",list( zip(data**.**iloc[:,1], x**.**iloc[:,1])))

print("\nGender:",list( zip(data**.**iloc[:,2], x**.**iloc[:,2])))

print("\nmaritialStatus:",list( zip(data**.**iloc[:,3], x**.**iloc[:,3])))

y**=**data**.**iloc[:,**-**1]

dt**=**DecisionTreeClassifier()

dt**.**fit(x,y)

*#[Age < 21, Income = Low,Gender = Female, Marital Status = Married]*

query**=**np**.**array([1,1,0,0])

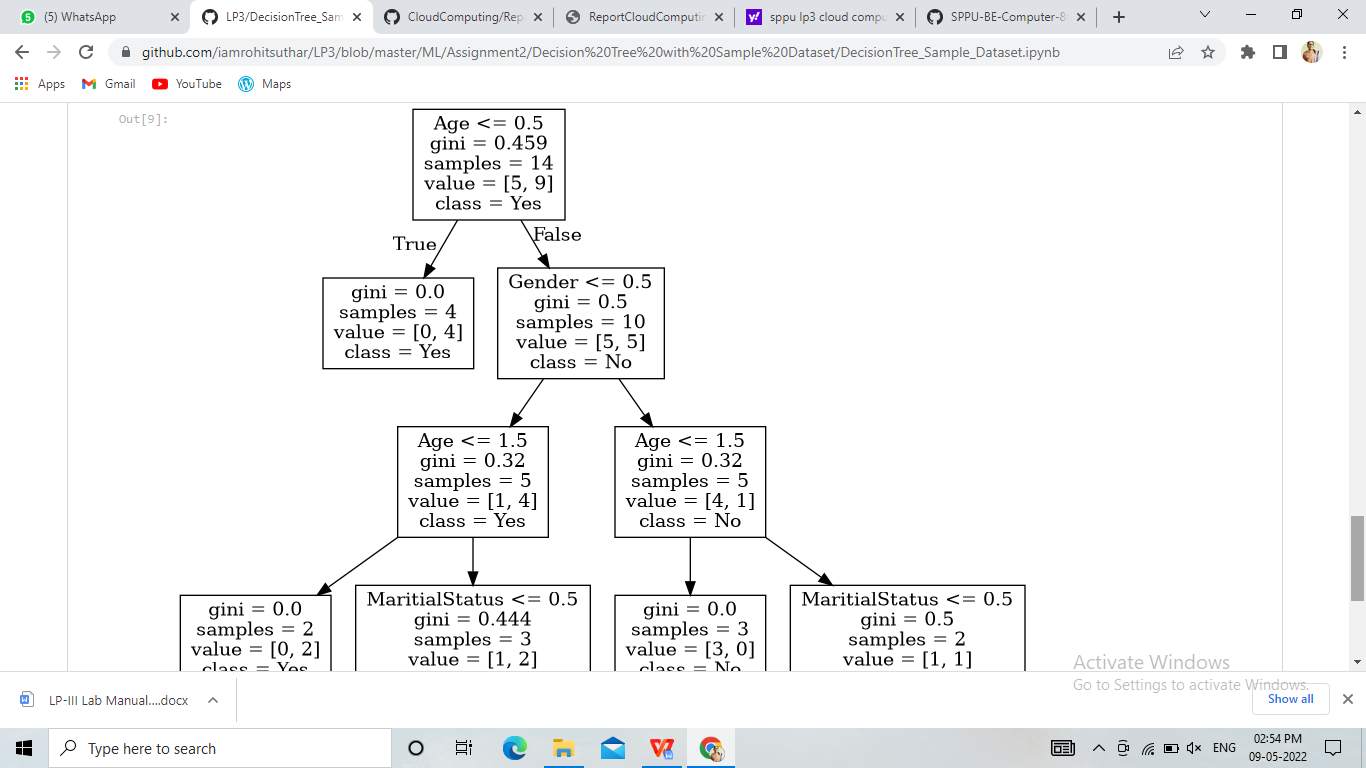
pred**=**dt**.**predict([query])

pred[0]

export\_graphviz(dt,out\_file**=**"data.dot",feature\_names**=**x**.**columns,class\_names**=**["No","Yes"])

**!**dot -Tpng data.dot -o tree.png

Image("tree.png")



**Assignment No**: 03

**3.1 Title**

Assignment based on k-NN Classification

**3.2 Problem Definition:**

In the following diagram let blue circles indicate positive examples and orange squares indicate negative examples. We want to use k-NN algorithm for classifying the points. If k=3, find-the class of the point (6,6). Extend the same example for Distance-Weighted k-NN and Locally weighted Averaging.

**3.3 Prerequisite:**

Basic of Python, Data Mining Algorithm, Concept of KNN Classification

**3.4 Software Requirements:**

Anaconda with Python 3.7

**3.5 Hardware Requirement:**

PIV, 2GB RAM, 500 GB HDD, Lenovo A13-4089Model.

**3.6 Learning Objectives:**

Learn How to Apply KNN Classification for Classify Positive and Negative Points in given example.

**3.7 Outcomes:**

After completion of this assignment students are able Implement code for KNN Classification for Classify Positive and Negative Points in given example also  Extend the same example for Distance-Weighted k-NN and locally weighted Averaging

**3.8 Theory Concepts:**

**3.8.1 Motivation**

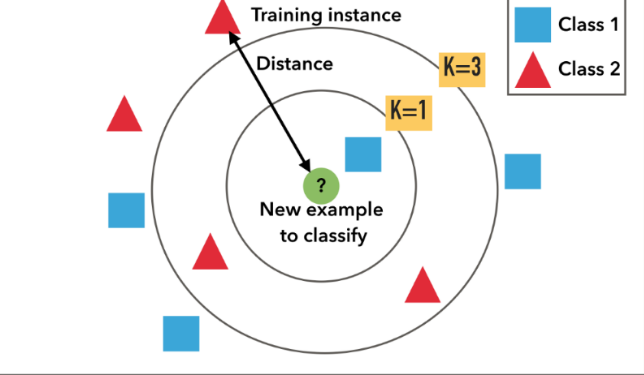
## K-Nearest Neighbors (kNN) Algorithm-

KNN is an non parametric lazy learning algorithm. That is a pretty concise statement. When you say a technique is non parametric , it means that it does not make any assumptions on the underlying data distribution. This is pretty useful , as in the real world , most of the practical data does not obey the typical theoretical assumptions made (eg gaussian mixtures, linearly separable etc) . Non parametric algorithms like KNN come to the rescue here.

It is also a lazy algorithm. What this means is that it does not use the training data points to do any generalization. In other words, there is no explicit training phase or it is very minimal. This means the training phase is pretty fast . Lack of generalization means that KNN keeps all the training data. More exactly, all the training data is needed during the testing phase. (Well this is an exaggeration, but not far from truth). This is in contrast to other techniques like SVM where you can discard all non support vectors without any problem.  Most of the lazy algorithms – especially KNN – makes decision based on the entire training data set (in the best case a subset of them).

The dichotomy is pretty obvious here – There is a non existent or minimal training phase but a costly testing phase. The cost is in terms of both time and memory. More time might be needed as in the worst case, all data points might take point in decision. More memory is needed as we need to store all training data.

KNN Algorithm is based on **feature similarity**: How closely out-of-sample features resemble our training set determines how we classify a given data point:



Example of k-NN classification. The test sample (inside circle) should be classified either to the first class of blue squares or to the second class of red triangles. If k = 3 (outside circle) it is assigned to the second class because there are 2 triangles and only 1 square inside the inner circle. If, for example k = 5 it is assigned to the first class (3 squares vs. 2 triangles outside the outer circle).

KNN can be used for **classification** — the output is a class membership (predicts a class — a discrete value). An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors. It can also be used for **regression** — output is the value for the object (predicts continuous values). This value is the average (or median) of the values of its k nearest neighbors.

### Assumptions in KNN

Before using KNN, let us revisit some of the assumptions in KNN.

KNN assumes that the data is in a feature space. More exactly, the data points are in a metric space. The data can be scalars or possibly even multidimensional vectors. Since the points are in feature space, they have a notion of distance – This need not necessarily be Euclidean distance although it is the one commonly used.

Each of the training data consists of a set of vectors and class label associated with each vector. In the simplest case , it will be either + or – (for positive or negative classes). But KNN , can work equally well with arbitrary number of classes.

We are also given a single number "k" . This number decides how many neighbors (where neighbors is defined based on the distance metric) influence the classification. This is usually a odd number if the number of classes is 2. If k=1 , then the algorithm is simply called the nearest neighbor algorithm.

### KNN for Classification

Lets see how to use KNN for classification. In this case, we are given some data points for training and also a new unlabelled data for testing. Our aim is to find the class label for the new point. The algorithm has different behavior based on k.

### Case 1 : k = 1 or Nearest Neighbor Rule

This is the simplest scenario. Let x be the point to be labeled . Find the point closest to x . Let it be y. Now nearest neighbor rule asks to assign the label of y to x. This seems too simplistic and some times even counter intuitive. If you feel that this procedure will result a huge error , you are right – but there is a catch. This reasoning holds only when the number of data points is not very large.

If the number of data points is very large, then there is a very high chance that label of x and y are same. An example might help – Lets say you have a (potentially) biased coin. You toss it for 1 million time and you have got head 900,000 times. Then most likely your next call will be head. We can use a similar argument here.

Let me try an informal argument here -  Assume all points are in a D dimensional plane . The number of points is reasonably large. This means that the density of the plane at any point is fairly high. In other words , within any subspace there is adequate number of points. Consider a point x in the subspace which also has a lot of neighbors. Now let y be the nearest neighbor. If x and y are sufficiently close, then we can assume that probability that x and y belong to same class is fairly same – Then by decision theory, x and y have the same class.

The book "Pattern Classification" by Duda and Hart has an excellent discussion about this Nearest Neighbor rule. One of their striking results is to obtain a fairly tight error bound to the Nearest Neighbor rule. The bound is

P^* \leq P \leq P^* ( 2 - \frac{c}{c-1} P^*)

Where P^* is the Bayes error rate, c is the number of classes and P is the error rate of Nearest Neighbor. The result is indeed very striking (atleast to me) because it says that if the number of points is fairly large then the error rate of Nearest Neighbor is less that twice the Bayes error rate. Pretty cool for a simple algorithm like KNN.

### Case 2 : k = K or k-Nearest Neighbor Rule

This is a straightforward extension of 1NN. Basically what we do is that we try to find the k nearest neighbor and do a majority voting. Typically k is odd when the number of classes is 2. Lets say k = 5 and there are 3 instances of C1 and 2 instances of C2. In this case , KNN says that new point has to labeled as C1 as it forms the majority. We follow a similar argument when there are multiple classes.

One of the straight forward extension is not to give 1 vote to all the neighbors. A very common thing to do is weighted kNN where each point has a weight which is typically calculated using its distance. For eg under inverse distance weighting, each point has a weight equal to the inverse of its distance to the point to be classified. This means that neighboring points have a higher vote than the farther points.

It is quite obvious that the accuracy \*might\* increase when you increase k but the computation cost also increases.

### Some Basic Observations

1. If we assume that the points are d-dimensional, then the straight forward implementation of finding k Nearest Neighbor takes O(dn) time.   
2. We can think of KNN in two ways  – One way is that KNN tries to estimate the posterior probability of the point to be labeled (and apply bayesian decision theory based on the posterior probability). An alternate way is that KNN calculates the decision surface (either implicitly or explicitly) and then uses it to decide on the class of the new points.   
3. There are many possible ways to apply weights for KNN – One popular example is the Shephard’s method.   
4. Even though the naive method takes O(dn) time, it is very hard to do better unless we make some other assumptions. There are some efficient data structures like **[KD-Tree](http://en.wikipedia.org/wiki/Kd_tree)**  which can reduce the time complexity but they do it at the cost of increased training time and complexity.   
5. In KNN, k is usually chosen as an odd number if the number of classes is 2.   
6. Choice of k is very critical – A small value of k means that noise will have a higher influence on the result. A large value make it computationally expensive and kinda defeats the basic philosophy behind KNN (that points that are near might have similar densities or classes ) .A simple approach to select k is set  k=\sqrt{n}   
7. There are some interesting data structures and algorithms when you apply KNN on graphs – See **[Euclidean minimum spanning tree](http://en.wikipedia.org/wiki/Euclidean_minimum_spanning_tree)** and **[Nearest neighbor graph](http://en.wikipedia.org/wiki/Nearest_neighbor_graph)** .

8. There are also some nice techniques like condensing, search tree and partial distance that try to reduce the time taken to find the k nearest neighbor.

### Applications of KNN

KNN is a versatile algorithm and is used in a huge number of fields. Let us take a look at few uncommon and non trivial applications.

**1. Nearest Neighbor based Content Retrieval**This is one the fascinating applications of KNN – Basically we can use it in Computer Vision for many cases – You can consider handwriting detection as a rudimentary nearest neighbor problem. The problem becomes more fascinating if the content is a video – given a video find the video closest to the query from the database – Although this looks abstract, it has lot of practical applications – Eg : Consider **[ASL](http://en.wikipedia.org/wiki/Asl)** (American Sign Language)  . Here the communication is done using hand gestures.

So lets say if we want to prepare a dictionary for ASL so that user can query it doing a gesture. Now the problem reduces to find the (possibly k) closest gesture(s) stored in the database and show to user. In its heart it is nothing but a KNN problem. One of the professors from my dept , Vassilis Athitsos , does research in this interesting topic – See **[Nearest Neighbor Retrieval and Classification](http://vlm1.uta.edu/~athitsos/nearest_neighbors/)** for more details.

**2. Gene Expression**This is another cool area where many a time, KNN performs better than other state of the art techniques . In fact a combination of KNN-SVM is one of the most popular techniques there. This is a huge topic on its own and hence I will refrain from talking much more about it.

## 3. Protein-Protein interaction and 3D structure prediction  Graph based KNN is used in protein interaction prediction. Similarly KNN is used in structure prediction.

## 4. Credit ratings — collecting financial characteristics vs. comparing people with similar financial features to a database. By the very nature of a credit rating, people who have similar financial details would be given similar credit ratings. Therefore, they would like to be able to use this existing database to predict a new customer’s credit rating, without having to perform all the calculations.

## Should the bank give a loan to an individual? Would an individual default on his or her loan? Is that person closer in characteristics to people who defaulted or did not default on their loans?

## In political science — classing a potential voter to a “will vote” or “will not vote”, or to “vote Democrat” or “vote Republican”.

1. More advance examples could include handwriting detection (like OCR), image recognition and even video recognition.

**Pros**:

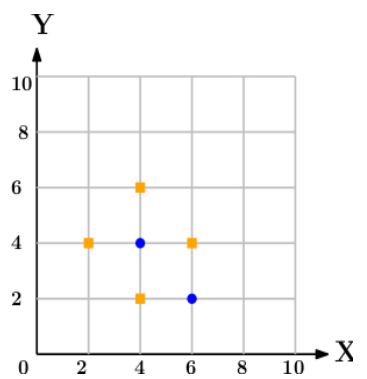
* No assumptions about data — useful, for example, for nonlinear data
* Simple algorithm — to explain and understand/interpret
* High accuracy (relatively) — it is pretty high but not competitive in comparison to better supervised learning models
* Versatile — useful for classification or regression

**Cons**:

* Computationally expensive — because the algorithm stores all of the training data
* High memory requirement
* Stores all (or almost all) of the training data
* Prediction stage might be slow (with big N)
* Sensitive to irrelevant features and the scale of the data

## Given Diagram Represent Positive and Negative Point with Color

## In the following diagram let blue circles indicate positive examples and orange squares indicate negative examples. We want to use k-NN algorithm for classifying the points. If k=3, find-the class of the point (6,6). Extend the same example for Distance-Weighted k-NN and Locally weighted Averaging



## Algorithm

## Import the Required Packages

## Read Given Dataset

## Import KNeighborshood Classifier and create object of it.

## Predict the class for the point(6,6) w.r.t to General KNN.

## Predict the class for the point(6,6) w.r.t to Distance Weighted KNN.

## 3.11 Conclusion

In this way we learn KNN Classification to predict the General and Distance Weighted KNN for Given data point in term of Positive or Negative.

## 3.12 Assignment Questions

* 1. **How KNN Different from Kmean Clustering Algorithm?**
  2. **What is the Formula for Euclidean Distance?**
  3. **What is Hamming Distance?**
  4. **What is formula for Manhatten and Minkowski Distnace?**
  5. **What are the application of KNN?**

Code:

**import** math

**def** euclidean\_distance(row1, row2):

distance **=** 0.0

**for** i **in** range(len(row1)**-**1):

distance **+=** (row1[i] **-** row2[i])**\*\***2

**return** sqrt(distance)

**def** get\_neighbors(train, test\_row, num\_neighbors):

distances **=** list()

**for** train\_row **in** train:

dist **=** euclidean\_distance(train\_row, test\_row)

distances**.**append((train\_row, dist))

distances**.**sort(key**=lambda** tup: tup[1])

neighbors **=** list()

**for** i **in** range(num\_neighbors):

neighbors**.**append(distances[i][0])

**return** neighbors

train\_data **=** [[2, 4, "Orange"], [4, 4, "Blue"], [4, 6, "Orange"], [4, 2, "Orange"], [6, 2, "Blue"], [6, 4, "Orange"]]test\_data **=** [[6, 6]]

**for** item **in** test\_data:

print("Data point : ", item)

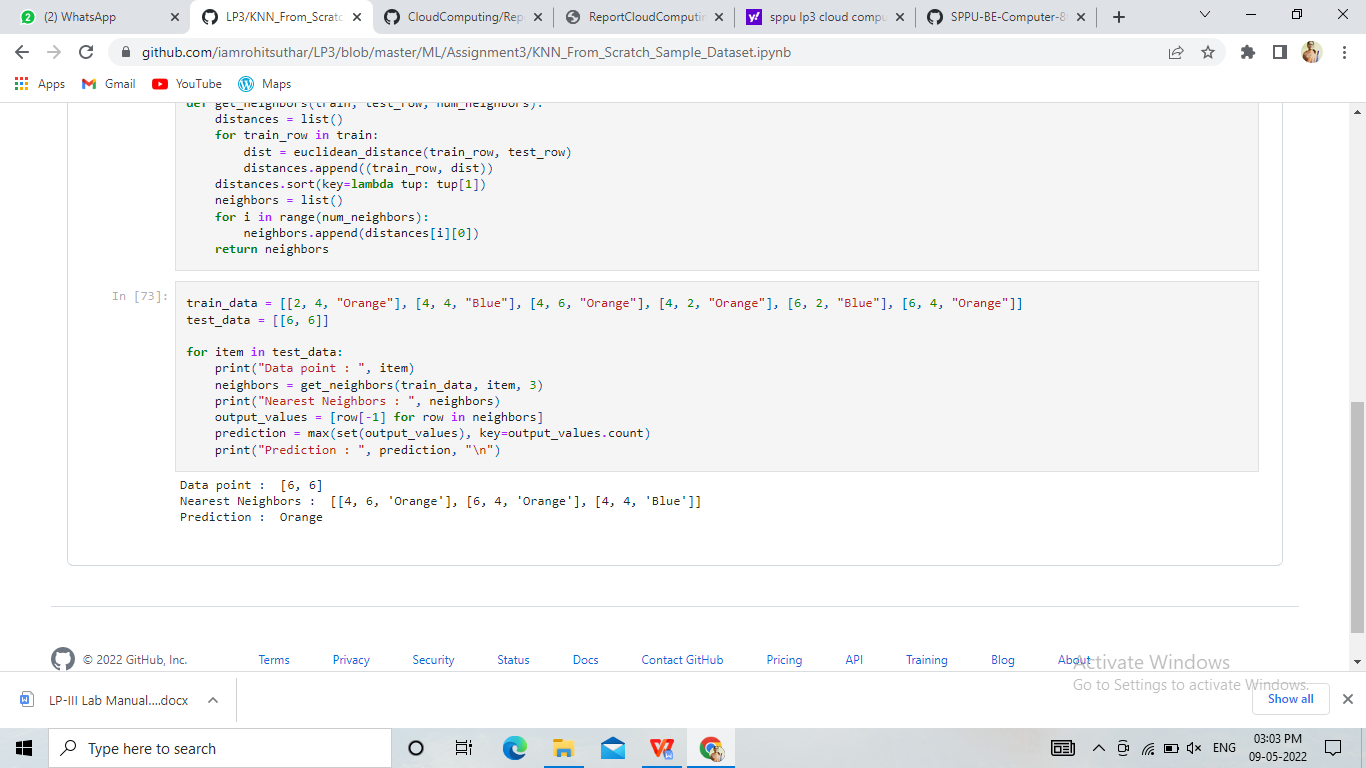
neighbors **=** get\_neighbors(train\_data, item, 3)

print("Nearest Neighbors : ", neighbors)

output\_values **=** [row[**-**1] **for** row **in** neighbors]

prediction **=** max(set(output\_values), key**=**output\_values**.**count)

print("Prediction : ", prediction, "\n")



# Assignment No:

**4.1 Title**

Assignment based on k-mean Clustering

**4.2 Problem Definition:**

We have given a collection of 8 points. P1=[0.1,0.6] P2=[0.15,0.71] P3=[0.08,0.9] P4=[0.16,0.85] P5=[0.2,0.3] P6=[0.25,0.5] P7=[0.24,0.1] P8=[0.3,0.2]. Perform the k-mean clustering with initial centroids as m1=P1 =Cluster#1=C1 and m2=P8=cluster#2=C2. Answer the following  
1] Which cluster does P6 belongs to?  
2] What is the population of cluster around m2?  
3] What is updated value of m1 and m2?

**4.3 Prerequisite:**

Basic of Python, Data Mining Algorithm, Concept of K-mean Clustering

**4.4 Software Requirements:**

Anaconda with Python 3.7

**4.5 Hardware Requirement:**

PIV, 2GB RAM, 500 GB HDD, Lenovo A13-4089Model.

**4.6 Learning Objectives:**

Learn How to Apply Kmean Clustering for given datapoints

**4.7 Outcomes:**

After completion of this assignment students are able Implement code for the k-mean clustering with initial centroids

**4.8 Theory Concepts:**

* + 1. **Motivation**

A Hospital Care chain wants to open a series of Emergency-Care wards within a region. We assume that the hospital knows the location of all the maximum accident-prone areas in the region. They have to decide the number of the Emergency Units to be opened and the location of these Emergency Units, so that all the accident-prone areas are covered in the vicinity of these Emergency Units.

The challenge is to decide the location of these Emergency Units so that the whole region is covered. Here is when K-means Clustering comes to rescue!

A cluster refers to a small group of objects. Clustering is grouping those objects into clusters. In order to learn clustering, it is important to understand the scenarios that lead to cluster different objects. Let us identify a few of them.

**What is Clustering?**

Clustering is dividing data points into homogeneous classes or clusters:

Points in the same group are as similar as possible

Points in different group are as dissimilar as possible

When a collection of objects is given, we put objects into group based on similarity.

**Application of Clustering:**

Clustering is used in almost all the fields. You can infer some ideas from Example 1 to come up with lot of clustering applications that you would have come across.

Listed here are few more applications, which would add to what you have learnt.

* Clustering helps marketers improve their customer base and work on the target areas. It helps group people (according to different criteria’s such as willingness, purchasing power etc.) based on their similarity in many ways related to the product under consideration.
* Clustering helps in identification of groups of houses on the basis of their value, type and geographical locations.
* Clustering is used to study earth-quake. Based on the areas hit by an earthquake in a region, clustering can help analyse the next probable location where earthquake can occur.

**Clustering Algorithms:**

A Clustering Algorithm tries to analyse natural groups of data on the basis of some similarity. It locates the centroid of the group of data points. To carry out effective clustering, the algorithm evaluates the distance between each point from the centroid of the cluster.

The goal of clustering is to determine the intrinsic grouping in a set of unlabelled data.



**What is K-means Clustering?**

K-means (Macqueen, 1967) is one of the simplest unsupervised learning algorithms that solve the well-known clustering problem. K-means clustering is a method of vector quantization, originally from signal processing, that is popular for cluster analysis in data mining.

**K-means Clustering – Example 1:**

A pizza chain wants to open its delivery centres across a city. What do you think would be the possible challenges?

* They need to analyse the areas from where the pizza is being ordered frequently.
* They need to understand as to how many pizza stores has to be opened to cover delivery in the area.
* They need to figure out the locations for the pizza stores within all these areas in order to keep the distance between the store and delivery points minimum.

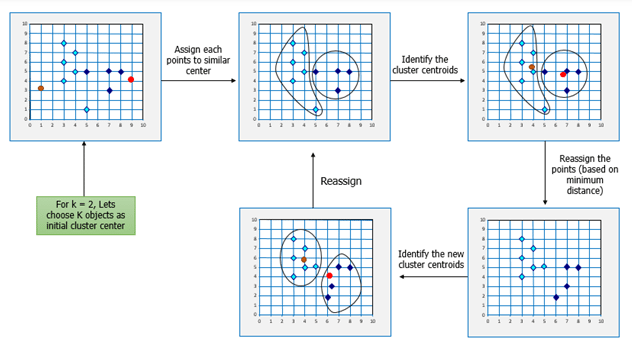
Resolving these challenges includes a lot of analysis and mathematics. We would now learn about how clustering can provide a meaningful and easy method of sorting out such real life challenges. Before that let’s see what clustering is.

**K-means Clustering Method:**

If k is given, the K-means algorithm can be executed in the following steps:

* Partition of objects into k non-empty subsets
* Identifying the cluster centroids (mean point) of the current partition.
* Assigning each point to a specific cluster
* Compute the distances from each point and allot points to the cluster where the distance from the centroid is minimum.
* After re-allotting the points, find the centroid of the new cluster formed.

**The step by step process:**



Now, let’s consider the problem in Example 1 and see how we can help the pizza chain to come up with centres based on K-means algorithm.

**Similarly, for opening Hospital Care Wards:**

K-means Clustering will group these locations of maximum prone areas into clusters and define a cluster center for each cluster, which will be the locations where the Emergency Units will open. These Clusters centers are the centroids of each cluster and are at a minimum distance from all the points of a particular cluster, henceforth, the Emergency Units will be at minimum distance from all the accident prone areas within a cluster.

Here is another example for you, try and come up with the solution based on your understanding of K-means clustering.

**K-means Clustering – Example 2:**

Let’s consider the data on drug-related crimes in Canada. The data consists of crimes due to various drugs that include, Heroin, Cocaine to prescription drugs, especially by underage people. The crimes resulted due to these substance abuse can be brought down by starting de-addiction centres in areas most afflicted by this kind of crime. With the available data, different objectives can be set. They are:

* Classify the crimes based on the abuse substance to detect prominent cause.
* Classify the crimes based on age groups.
* Analyze the data to determine what kinds of de-addiction centre is required.
* Find out how many de-addiction centres need to be setup to reduce drug related crime rate.

The K-means algorithm can be used to determine any of the above scenarios by analyzing the available data.

Following the K-means Clustering method used in the previous example, we can start off with a given k, following by the execution of the K-means algorithm.

**Mathematical Formulation for K-means Algorithm:**

|  |  |  |
| --- | --- | --- |
| K-Means clustering intends to partition *n* objects into *k* clusters in which each object belongs to the cluster with the nearest mean. This method produces exactly *k* different clusters of greatest possible distinction. The best number of clusters *k* leading to the greatest separation (distance) is not known as a priori and must be computed from the data. The objective of K-Means clustering is to minimize total intra-cluster variance, or, the squared error function: |  |  |
|  |  |  |
|  |  |  |

## 4.9 Algorithm

## Import the Required Packages

## Create dataset using DataFrame

## Find centroid points

## plot the given points

## for i in centroids():

## plot given elements with centroid elements

## import KMeans class and create object of it

## using labels find population around centroid

## Find new centroids

## 4.10 Conclusion

In this way we learn Kmean Clustering Algorithm.

## 4.11 Assignment Questions

### **[Why does k-means clustering algorithm use only Euclidean distance metric?](https://stats.stackexchange.com/questions/81481/why-does-k-means-clustering-algorithm-use-only-euclidean-distance-metric)**

### **[How to decide on the correct number of clusters?](https://stats.stackexchange.com/questions/23472/how-to-decide-on-the-correct-number-of-clusters)**

1. What are the Different Application of K Mean Clustering?
2. What is K-medoids?
3. What is *[k](https://en.wikipedia.org/wiki/K-medians_clustering" \o "K-medians clustering)*[-medians clustering](https://en.wikipedia.org/wiki/K-medians_clustering" \o "K-medians clustering)?

Code:

**from** copy **import** deepcopy

**import** numpy **as** np

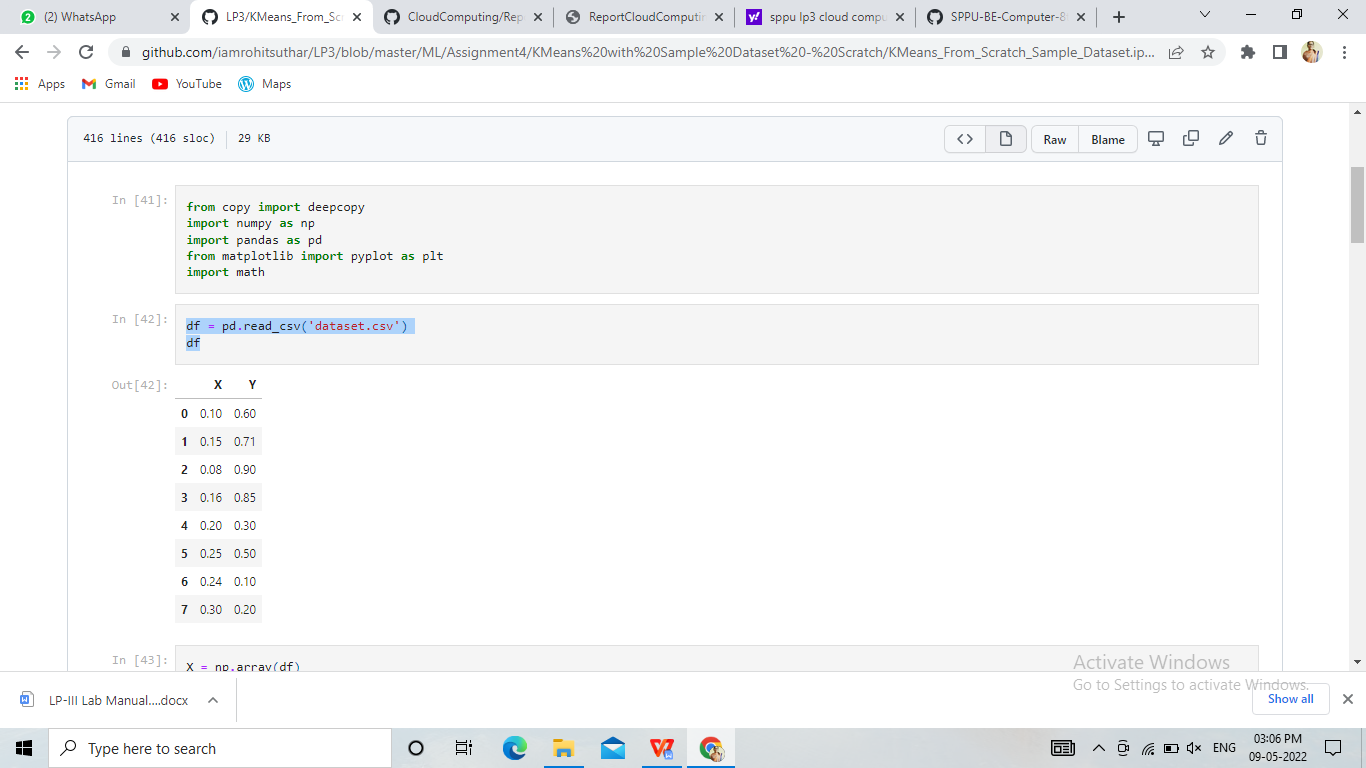
**import** pandas **as** pd

**from** matplotlib **import** pyplot **as** plt

**import** math

df **=** pd**.**read\_csv('dataset.csv')

df



X **=** np**.**array(df)

c\_x **=** np**.**array([0.1,0.3])

c\_y **=** np**.**array([0.6,0.2])

centroids **=** np**.**array(list(zip(c\_x,c\_y)))

Centroids

Output:

array([[0.1, 0.6],

[0.3, 0.2]])

**class** K\_Means:

**def** \_\_init\_\_(self, k**=**2, tol**=**0.001, max\_iter**=**300):

self**.**k **=** k

self**.**tol **=** tol

self**.**max\_iter **=** max\_iter

**def** fit(self,data,centroids):

self**.**centroids **=** {}

**for** i **in** range(self**.**k):

self**.**centroids[i] **=** centroids[i]

**for** i **in** range(self**.**max\_iter):

self**.**classifications **=** {}

**for** i **in** range(self**.**k):

self**.**classifications[i] **=** []

**for** featureset **in** data:

distances **=** [np**.**linalg**.**norm(featureset**-**self**.**centroids[centroid]) **for** centroid **in** self**.**centroids]

classification **=** distances**.**index(min(distances))

self**.**classifications[classification]**.**append(featureset)

prev\_centroids **=** dict(self**.**centroids)

**for** classification **in** self**.**classifications:

self**.**centroids[classification] **=** np**.**average(self**.**classifications[classification],axis**=**0)

optimized **=** **True**

**for** c **in** self**.**centroids:

original\_centroid **=** prev\_centroids[c]

current\_centroid **=** self**.**centroids[c]

**if** np**.**sum((current\_centroid**-**original\_centroid)**/**original\_centroid**\***100.0) **>** self**.**tol:

print(np**.**sum((current\_centroid**-**original\_centroid)**/**original\_centroid**\***100.0))

optimized **=** **False**

**if** optimized:

**break**

**def** predict(self,data):

distances **=** [np**.**linalg**.**norm(data**-**self**.**centroids[centroid]) **for** centroid **in** self**.**centroids]

classification **=** distances**.**index(min(distances))

**return** classification

model **=** K\_Means()

model**.**fit(X, centroids)

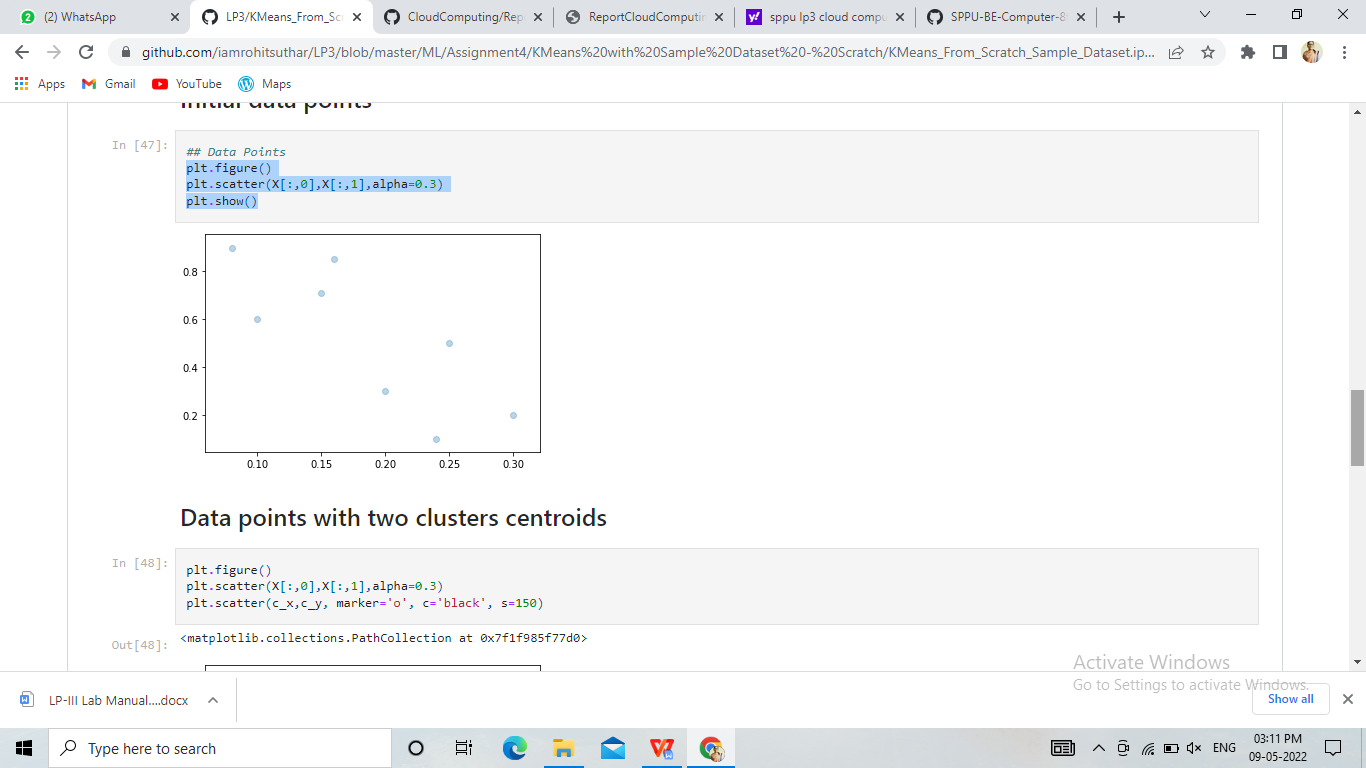
O/p:

66.66666666666666

plt**.**figure()

plt**.**scatter(X[:,0],X[:,1],alpha**=**0.3)

plt**.**show()



colors **=** ['r','b']

**for** centroid **in** model**.**centroids:

plt**.**scatter(model**.**centroids[centroid][0], model**.**centroids[centroid][1],

marker**=**"o", color**=**"k", s**=**150, linewidths**=**5)

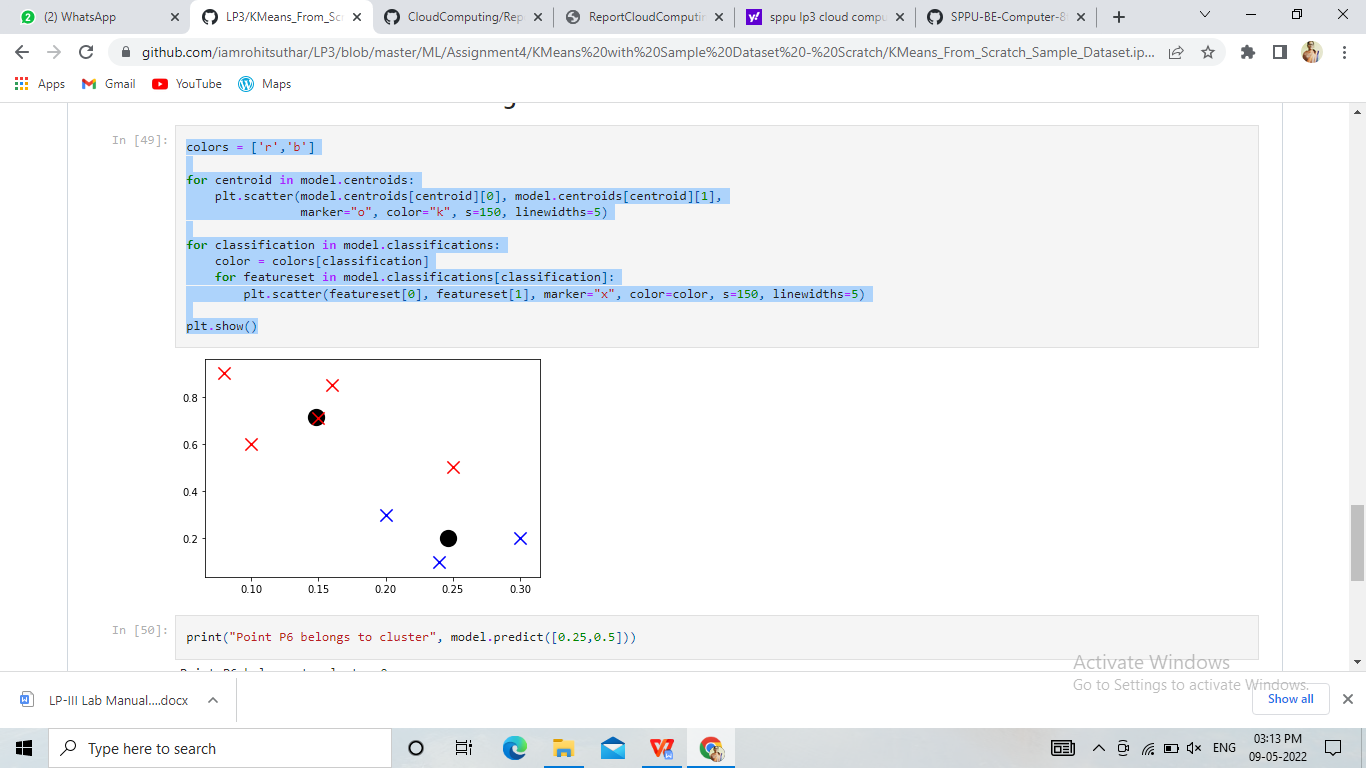
**for** classification **in** model**.**classifications:

color **=** colors[classification]

**for** featureset **in** model**.**classifications[classification]:

plt**.**scatter(featureset[0], featureset[1], marker**=**"x", color**=**color, s**=**150, linewidths**=**5)

plt**.**show()



print("Point P6 belongs to cluster", model**.**predict([0.25,0.5]))

O/p:

Point P6 belongs to cluster 0