|  |
| --- |
| EDWISOR |
| SANTANDER CUSTOMER TRANSACTION PREDICTION USING PYTHON |
| PYTHON REPORT |

|  |
| --- |
| BY : RAJAT AGRAWAL  27-Jul-19 |

**ABSTRACT**

THE DATA GIVEN TO US HAS AN TARGET VARIABLE CATEGORICAL IN NATURE SO ALL THE METHOD WE USE TO PREDICT THE TEST DATASET TARGET VALUE ARE CLASSIFIED MODELLING TECHNIQUE. THE TECHNIQUE WE WILL USE TO PREDICT OUR TEST DATA ARE AS FOLLOWS:

1. LOGISTIC REGRESSION
2. DECISION TREES CLASSIFIER
3. RANDOM FORREST CLASSIFIER
4. NAÏVE BAYES METHOD
5. K-NEAREST NEIGHBOUR

THIS REPORT GIVE US AN IDEA ABOUT HOW TO PREDICT A DATA FROM A GIVEN DATA WHICH IS INCONSISTENT IN NATURE. SO FIRST WE DO DATA EXPLORATORY WORK WHICH WILL AS NAME SUGGEST EXPLORE THE DATA . IN EXPLORING THE DATA WE WILL EXPLORE ALL THE COMPONENT OF DATASET.

AFTER THAT WE WILL DO PREPROCESSING WHICH MAKE THE DATA IN THE FORMAT THAT MODELLING METHOD CAN BE USED ON IT AND THEY GIVE THE CORRECT PREDICTION.

WHEN PREPROCESSING IS DONE OUR DATA IS READY TO FORM A MODEL ON IT SO ON THE BASIS OF CHARACTER OF TARGET VARIABLE VARIOUS METHOD OF MODELLING CAN BE USED.ONCE THE MODEL IS READY ON BASIS OF ERROR BEST MODEL CAN BE DECIDED .

NOW WE HAVE DECIDED OUR BEST MODEL AFTER THAT ON THE BASIS OF TEST DATASET WE CAN PREDICT OUR TARGET VARIABLE AFTER USING BEST METHOD OF PREDICTION

CONTENTS

1. ABOUT…………………………………………………………..*PAGE3*
2. LIBRARY INSTALLATION…………………………………...*PAGE7*
3. DATA EXPLORATION………………………………………..*PAGE8*
4. DATA PREPROCESSING……………………………………..*PAGE14*

* *MISSING VALUE ANALYSIS………………………………… ..PAGE15*
* *OUTLIER ANALYSIS………………………………………….. PAGE15*
* *DATA FEATURING…………………………………………….PAGE18*
* *DATA SCALING……………………………………………….. PAGE20*

1. DATA MODELLING………………………………………… *PAGE21*

* *LOGISTIC REGRESSION………………………………………PAGE24*
* *DECISION TREE……………………………………………….. PAGE26*
* *RANDOM FORRES…………………………………………….PAGE 28*
* *NAÏVE BAYES……………………………………………………PAGE30*
* *K-NEAREST NEIGHBOURS………………………………… PAGE32*

1. SUMMARY OF DATA MODELLING……………………….*PAGE33*
2. CONCLUSION……………………………………………….. *PAGE34*
3. PREDICTION OF OF TEST DATA………………………….. *PAGE35*
4. SAVED FILES DURING PROCESS…………………………. *PAGE37*
5. IMPORTANT NOTE………………………………………*PAGE39*
6. APPENDIX CODING………………………FILE IS ATTACHED

**ABOUT**

***BACKGROUND***

AT SANTANDER , MISSION IS TO HELP PEOPLE AND BUSINESSES PROSPER. WE ARE ALWAYS LOOKING FOR WAYS TO HELP OUR CUSTOMERS UNDERSTAND THEIR FINANCIAL HEALTH AND IDENTIFY WHICH PRODUCTS AND SERVICES MIGHT HELP THEM ACHIEVE THEIR MONETARY GOALS.

OUR DATA SCIENCE TEAM IS CONTINUALLY CHALLENGING OUR MACHINE LEARNING ALGORITHMS,WORKING WITH THE GLOBAL DATA SCIENCE COMMUNITY TO MAKE SURE WE CAN MORE ACCURATELY IDENTIFY NEW WAYS TO SOLVE OUR MOST COMMON CHALLENGE, BINARY CLASSIFICATION PROBLEMSSUCH AS: IS A CUSTOMER SATISFIED? WILL A CUSTOMER BUY THIS PRODUCT? CAN A CUSTOMER PAY THIS LOAN?

***PROBLEM STATEMENT***

IN THIS CHALLENGE, WE NEED TO IDENTIFY WHICH CUSTOMERS WILL MAKE A SPECIFIC TRANSACTION INTHE FUTURE, IRRESPECTIVE OF THE AMOUNT OF MONEY TRANSACTED.

***DATA SET*** :

1) TEST.CSV

2) TRAIN.CSV

***NUMBER OF ATTRIBUTES:***

YOU ARE PROVIDED WITH AN ANONYMIZED DATASET CONTAINING NUMERIC FEATURE VARIABLES, THEBINARY TARGET COLUMN, AND A STRING ID\_CODE COLUMN. THE TASK IS TO PREDICT THE VALUEOF TARGET COLUMN IN THE TEST SET.

***EVALUATION BASIS***

FOR CLASSIFICATION THE ACCURACY METRICS NEED TO BE CONSIDERED ARE AUC, PRECISION & RECALL. YOU ARE FREE TO USE OTHER METRICS IN ADDITION TO THE AFOREMENTIONED METRICS.

***MISSING VALUE***

NO MISSING VALUE IN DATA.

**IMPORT LIBRARIES**

We have form a python file on jupyter notebook in which various inbuilt functions are present like null function, info function , count function etc. but many external fuctions can only be used by importing libraries.

These libraries contain there own function which will make our works very easy.

We can import library initially or we can import them when we require them. Most Initial step of any data project is to import various libraries which will be use during the course of project as follow:

* Import pandas as pd
* import numpy as np
* import matplotlib.pyplot as plt
* import seaborn as sns
* import os
* from fancyimpute import KNN
* from random import randrange, uniform
* from scipy.stats import chi2\_contingency
* %matplotlib inline

***SET DIRECTORY***

Here we will set the position in our hard disc where all the data will be import and export into the python file. This will be user specific. All the fill we save during the course of project will be save into this folder and python will read only those excel or csv files present in this folder.

Our directory position is "C:/Users/hp/Desktop/abc satender customer" and this is set using function os.chdir . this will change if user is running the file some other computer according to location heor she want to save.

**DATA EXPLORATORY ANALYSIS**

Data exploration is used to explore various aspects of data so that data can be formed in presentable format. The data came to us for modeling has a various glitches on which algorithm can not apply for example sometimes Data type of variable is are mismatched with there values datatype. So will change there data type. Sometimes extra variable are present which are not required so we will remove them. Some time variable columns contain some extra string which is not required so we will remove it.

After importing libraries we will upload train dataset and use head function which will gives top 5 observation for analysis after that we start analyzing the dataset .

The steps we follow in this project for DEA are :

After import data we can show that data contain target value in form of 0 an1 here 0: customer not buy the product

1:customer buy the product.

And the factor on which this target value depends are continuous in nature and are 200 in quantity , named as var\_0,var\_1,var\_2……..

In this way they named upto var\_199.

One variable named as ID\_code are of string datatype which has no use in predicting the model so we can remove it.

Now we can will describe function on dataset which will gives median at 50%, 75%, 25% , mean , number of counts etc.

We will use info function to get the data type of various variable.

During analyzing we have seen a various observation:

1. We use .shape function to get the shape of dataset it gives (200000 ,202) it means dataset contain 2 lakh observation and 202 variables
2. Datset contain null value which we can see by using a function .isnull().sum(). It shows dataset has no null value
3. We use info function which gives following value:

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 200000 entries, 0 to 199999

Columns: 202 entries, ID\_code to var\_199

dtypes: float64(200), int64(1), object(1)

memory usage: 308.2+ MB

It gives us that data contain 200000 entries , 200 variables are of float nature , 1 is integer in nature and 1 is of string nature. And its memory usage. Class of dataframe

1. So if possible we should reduce the dataset variable into necessary variable
2. We used correlation heatmap and principal component analysis to do so
3. Now we will use various method to get missing value as follow:

* trans[trans.isna().any(axis=1)]

it gives us rows having null value. it shows no row has null value.

* trans.isnull().sum().head(5)

ID\_code 0

target 0

var\_0 0

var\_1 0

var\_2 0

dtype: int64

* Now all the missing values are formed used a loop and saves in percentage terms in a new .csv form data set named as "*Missing\_perc.csv*"

1. Now Univariate , bivariate and multivariate analysis is start

***Univariate***

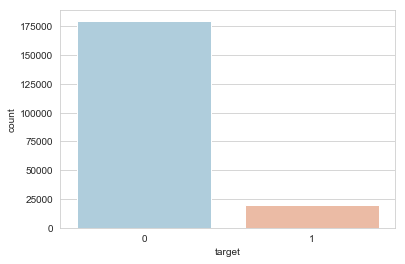
Here the variation in single variable is analyse using the graphical methods here we used histograms , count plot to get the number of observation of each kind

So here we formed the count plot of target variable using function

sns.set\_style('whitegrid')

sns.countplot(x='target',data=trans,palette='RdBu\_r')

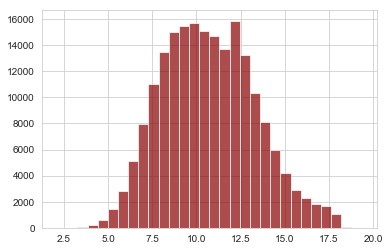
and the graph which gives the count of both the values 0 and 1 is formed.



Here 0 means customer will not buy the product and 1 means he will buy the product, so number of people not buy is much larger the buy.

We have used the function plt.hist to get the histogram of of var\_0 as

trans['var\_0'].hist(bins=30,color= 'darkred',alpha=0.7) and histogram is formed as follow:



***Bivariate***

It gives us the variation between any two variables of dataset so it shows how one variable change with respect other variable. The concept used to check variation are correlation and chi square test.

Here we use function to get the plot

sns.pairplot(trans.iloc [: ,0:10], hue='target',palette='Set1')

Here we have drawn pairplot of initial ten variables to give an anlysis how the variables are depending with each other.

***Multivariate***

It shows the variation of multiple variable with each other all the machine learning algorithm are multivariate in nature because they can use more than 2 variable at a time to get the model,

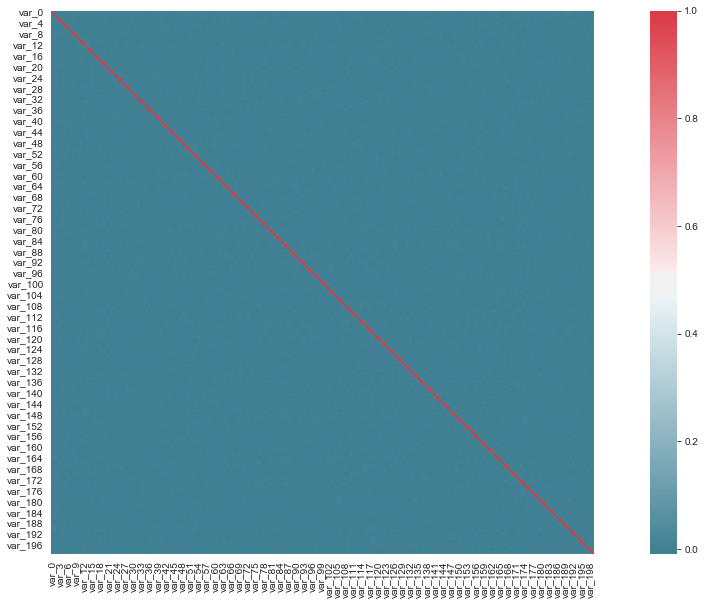
PCA analysis useg to remove less variation variable is also use multivariate analysis

These three analysis are very important part of data exploratory analysis.

1. Our dataset contain very large no number of variables which take lot of time in data modeling process so to save the time of make model simpler we will now try to remove variable which does not have large variation with target variable so to do so we used two method which are :

***Heat map***:

Finding out correlation heat map so that extra variable can be removed out and get dataset contain lesser variable which are easy to handle

  
WHOLE HEATMAP IS SHOWING BLUE COLOR EXCEPT FOR DIAGONAL SO HERE HEAT MAP IS SHOWING THAT NO TWO INDEPENDENT VARIABLES ARE DEPENDING WITH EACH OTHER SO WE CAN NOT REMOVE ANY VARIABLE

***Principle component analysis***

It will remove the variable which showing less variation so will try to take the variable contain aggregate of more than 90% of variation

After using various library we can see that variation of top 100 variabkes is comes out to be .50 so

So we can see that only 50% of variations is covered by 100 variable which is also a 50 % of total variable.so it shows we should use all the variables to form the model as they all are contributing equally to find target variable

**DATA PREPROCESSING**

After EDA we will do preprocessing which make the data in the format that modelling method can be used on it and they give the correct prediction.

When preprocessing is done our data is ready to form a model on it so the basis of character of target variable various method of modelling can be used.once the model is ready on basis of error best model can be decided

Four method used in preprocessing are.

1) missing value analysis

2) outlier analysis

3) data selection

4) data scaling

* ***MISSING VALUE ANALYSIS***

Data does not have any missing value so we are not use missing value imputation

* ***OUTLIER ANALYSIS***

Outliers are values which are inconsistent in nature which make data inappropriate . Which simply check that outlier using the box plot and the values locate at the extreme of the box plot are outlier we can also used the loop to get the outlier.

So once we get outlier we can remove them from data or we can put null value on the place of it and the impute null value by the various method we have learnt in previous section.

Outlier can be shown mathematically as:

Here q75: it is value of a variable at 75 percentile value

q25:it is value of a variable at 25 percentile value

iqr = q75 - q25

min = q25 - (iqr\*1.5)

max = q75 + (iqr\*1.5)

outlier is a observations can comes in data due to incorrect experiment which when used to form the model can effect the models predictability criteria. Sometimes they are correctly filled due to some exceptions in dataset but in that case also they does not give the correct trend.

Loop used to remove the outliers values is as follow:

for i in range(2,202):

#print(i)

* q75, q25 = np.percentile(trans.iloc[:,i], [75 ,25])

iqr = q75 - q25

* min = q25 - (iqr\*1.5)
* max = q75 + (iqr\*1.5)

#print(min)

#print(max)

* trans = trans.drop(trans[trans.iloc[:,i] < min].index)
* trans = trans.drop(trans[trans.iloc[:,i] > max].index)

Loop used for putting the value NaN at the place of outlier and impute it by KNN method:

# #Detect and delete outliers from data

for i in range(2,202):

print(i)

* q75, q25 = np.percentile(trans.iloc[:,i], [75 ,25])
* iqr = q75 - q25
* min = q25 - (iqr\*1.5)
* max = q75 + (iqr\*1.5)

print(min)

print(max)

#Replace with NA

* trans.iloc[:,i]=np.where(trans.iloc[:,i]>max, np.NaN,trans.iloc[:,i])
* trans.iloc[:,i]=np.where(trans.iloc[:,i]<min, np.NaN,trans.iloc[:,i])

# #Calculate missing value

* missing\_val = pd.DataFrame(trans.isnull().sum())

#Reset index

* missing\_val = missing\_val.reset\_index()

#Rename variable

* missing\_val=missing\_val.rename(columns={'index':Variables', 0: 'Missing\_percentage'})

#Calculate percentage

* missing\_val['Missing\_percentage']= = (missing\_val['Missing\_percentage']/len(fare))\*100

#descending order

* missing\_val=missing\_val.sort\_values('Missing\_percentage',ascending = False).reset\_index(drop = True)

#Impute with KNN

* fare=pd.DataFrame(KNN(k=3).complete(fare),columns= fare.columns)

**now these two loops we have written we used first loop which remove outlier as it gives more perfect model.**

**Outliers percentages in each column.csv**

here we have saved the outliers value as null and saved them as separate dataset of missing value named as

“*Outliers percentages in each column.csv*”.

we have also saved the finel dataset after removing the outlier as “*outlier values.csv”*. this dataset is free from outlier and missing values.

**After removing outlier we get 175073 observation from 200000 observations so 24927 observation are exist as an outlier in trans dataset**

* ***FEATURE SELECTION***

It is method in which we remove the continuous variable which are dependent on each other so that no two variables denoted same variation . this method used the concept of collinearity to find the dependency between the independent variable . most of the time we used heat maps to find there collinearity. If collinearity between two variables is equal to 1 or -1 we remove one variable from them.

For categorical variable we used chi-square test to find dependency between target and independent variable. So if the p value due to chi-square test between target variable and continuous variable is less than .005 it shows that two variables are dependent than we take that variable for modeling but those variables whose value comes out to be greater than .05 will remove from dataset.

**CHECK FOR COLLINEARITY**

In heat map the variables which shows perfectly red are dependent and those which are perfectly blue are independent.

So in python script we can see that all variables in heat map shows perfectly dark red colour except for the diagonal ones so all them are independent with each other.

In data exploratory analysis we have shown earlier that no variable is dependent with each other correlation value is comes out to be 0

So for script we have fetch out initial 10 variables just for example to show initial 10 varible dependence heatmap and show that no two variable are depend on each other

Heat map script for dataset is as follow:

* trans\_corr = trans.iloc[:,2:10]

#Set the width and hieght of the plot

* f, ax = plt.subplots(figsize=(20, 10))

#Generate correlation matrix

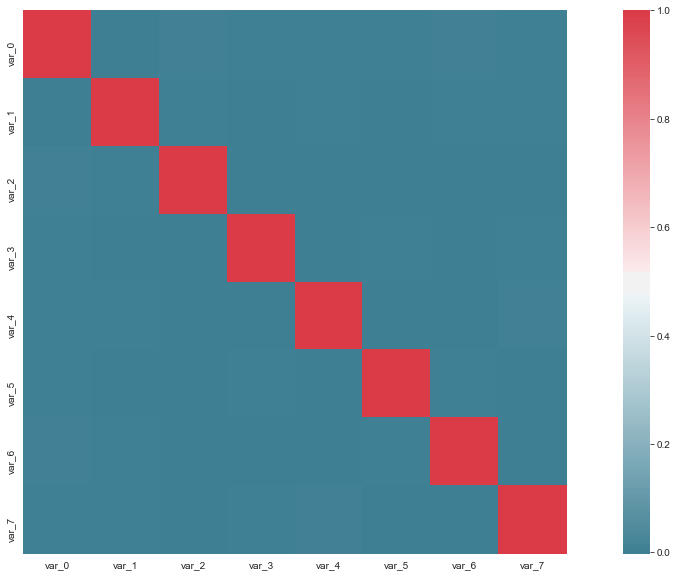
* corr = trans\_corr.corr()

#Plot using seaborn library

* sns.heatmap(corr,mask=np.zeros\_like(corr,dtype=np.bool), cmap=sns.diverging\_palette(220, 10, as\_cmap=True),

square=True, ax=ax)

**HEAT MAP OF INITIAL TEN VARIABLES WE GET**

**  
CHECK FOR CHI SQUARE TEST**

No check for chi square test as all the independent variables are continuous in nature. The one variable which is string type named ID\_code has no use for predicting data.

* ***FEATURE SCALING***

It used when modeling method used for prediction require distance between two variable as algorithm. Variations between two variable plays an important role so make them equally variate we will use standardization and normalization for each continuous independent variable.

Standardization is prefer when distribution of target variable is normal in nature otherwise normalization is used:

**But as our data is categorical in nature we will always use standardization method.**

Formulae:

Standardization:

http://ci.columbia.edu/ci/premba_test/c0331/images/s6/5836240103.gif

Here µ is mean

Normalization :

https://www.statisticshowto.datasciencecentral.com/wp-content/uploads/2015/11/normalize-data.png

We will used feature scaling according to the algorithm we used:

For example: if we used KNN we used normalization as our data is skew symmetric but when we used random forest or decision tree there is no requirement of feature scaling.

So method used to standardize the dataset is:

* from sklearn.preprocessing import StandardScaler
* scaler = StandardScaler()
* scaler.fit(trans.drop('target',axis=1))
* scaled\_features = scaler.transform(trans.drop('target',axis=1))
* df\_feat = pd.DataFrame(scaled\_features,columns=trans.columns[:-1])
* df\_feat.head()

**PREDICTING OUR MODEL**

Model is start to build when all the work on data is completed and we have get the data in neat and clean format without any missing value, without any outlier , no dependency of independent variable with each other. Data is in standardize form if require for algorithm.

The data given to us has an target variable categorical in nature so all the method we use to predict the test dataset target value are classified modelling technique. The technique we will use to predict our test data are as follows:

1. Logistic regression
2. Decision trees classifier
3. Random forrest classifier
4. Naïve bayes method
5. K-nearest neighbor

All the machine learning algorithm used following steps to

Get the prediction in this model:

1. first we remove ID\_code variable column as it is of string data type it create problem during the modeling of machine learning algorithm and it also has no use in predicting the data as it contain simple numbers so we will remove it.
2. In all the algorithm before starting anything first we import their respective library .
3. Then we import the libraries to split the data into training and test set . we usually take 40% of data as test set. command used for this purpose is “From sklearn.model\_selection import train\_test\_split”
4. we set the train data into algorithm using respective command as algorithm.
5. Than we predict the data using .predict function and putting the test independent variable.
6. Now according to question for classification the accuracy metrics need to be considered are auc, precision & recall.
7. After that we will import classification\_report and confusion\_metrics from sklearn which used to get Precision, recall values which used to get the best model
8. Now from predictions we get from x\_test is used with y\_test values to get the 2\*2 confusion metrics
9. The four position of confusion metrics is denoted as

TN = CM.iloc[0,0]

FN = CM.iloc[1,0]

TP = CM.iloc[1,1]

FP = CM.iloc[0,1]

1. The loop we used to get confusion metrics is:

#build confusion matrix

* from sklearn.metrics import confusion\_matrix
* CM = confusion\_matrix(y\_test, predictions)
* CM = pd.crosstab(y\_test, predictions)

#let us save TP, TN, FP, FN

* TN = CM.iloc[0,0]
* FN = CM.iloc[1,0]
* TP = CM.iloc[1,1]
* FP = CM.iloc[0,1]

#check accuracy of model

* print("recall", "=", (TP\*100)/(TP+FN))

#precision

* print("precision", "=" ,(TP\*100)/(TP+FP))

1. To form a classification report following command will be used.

* print(classification\_report(y\_test,predictions))

and the report contain all measuring quantity form like this

precision recall f1-score support

0 0.93 0.99 0.96 47405

1 0.69 0.26 0.38 5117

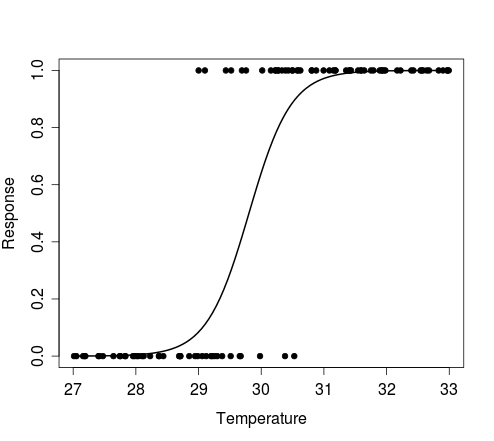
accuracy 0.92 52522

macro avg 0.81 0.63 0.67 52522

weighted avg 0.90 0.92 0.90 52522

1. now finel step is to calculate AUC which can be calculate using the command metrics.roc\_auc\_score(y\_test, predictions) and before that we have to import library metrics from sklearn.
2. Now based on the combination of precision, recall , AUC best machine learning algorithm will be choose.
3. For the betterment of model all the three values should be maximize.

* ***LOGISTIC REGRESSION METHOD:***

Logistic regression is a part of Generalised Linear Model. It uses log of odd to form a linear line . and a line which gives maximum log of likehood will be used to predict the data .so for binary model based on the probability we decide the value for example: if probability is less than .5 we takes 0 value if probability is more than .5 we take 1 as a target value.

the graph of logistic curve looks like this.

Logistic algorithm works by using following commands one after other.

* from sklearn.linear\_model import LogisticRegression
* logmodel = LogisticRegression()
* logmodel.fit(x\_train,y\_train)
* predictions = logmodel.predict(x\_test)

EVALUATION

For evaluation of error or accuracy various library should be needed. So the following commands Are used to get an different accuracy:

* from sklearn.metrics import classification\_report
* from sklearn.metrics import confusion\_matrix
* print(classification\_report(y\_test,predictions))

now the classification report form

now build the confusion metrics and get the precision and recall error:

#build confusion matrix

* from sklearn.metrics import confusion\_matrix
* CM = confusion\_matrix(y\_test, predictions)
* CM = pd.crosstab(y\_test, predictions)

#let us save TP, TN, FP, FN

* TN = CM.iloc[0,0]
* FN = CM.iloc[1,0]
* TP = CM.iloc[1,1]
* FP = CM.iloc[0,1]

#check accuracy of model

#recall

* print("recall", "=", (TP\*100)/(TP+FN))

#precision

* print("precision", "=" ,(TP\*100)/(TP+FP))

now get the AUC using metrics library the command used for this is

* metrics.roc\_auc\_score(y\_test, predictions)

**RESULTS OF LOGISTIC REGRESSION**

**Recall = 26.34356068008599%**

**Precision = 69.05737704918033%**

**AUC =62.53%**

**Accuracy =92%**

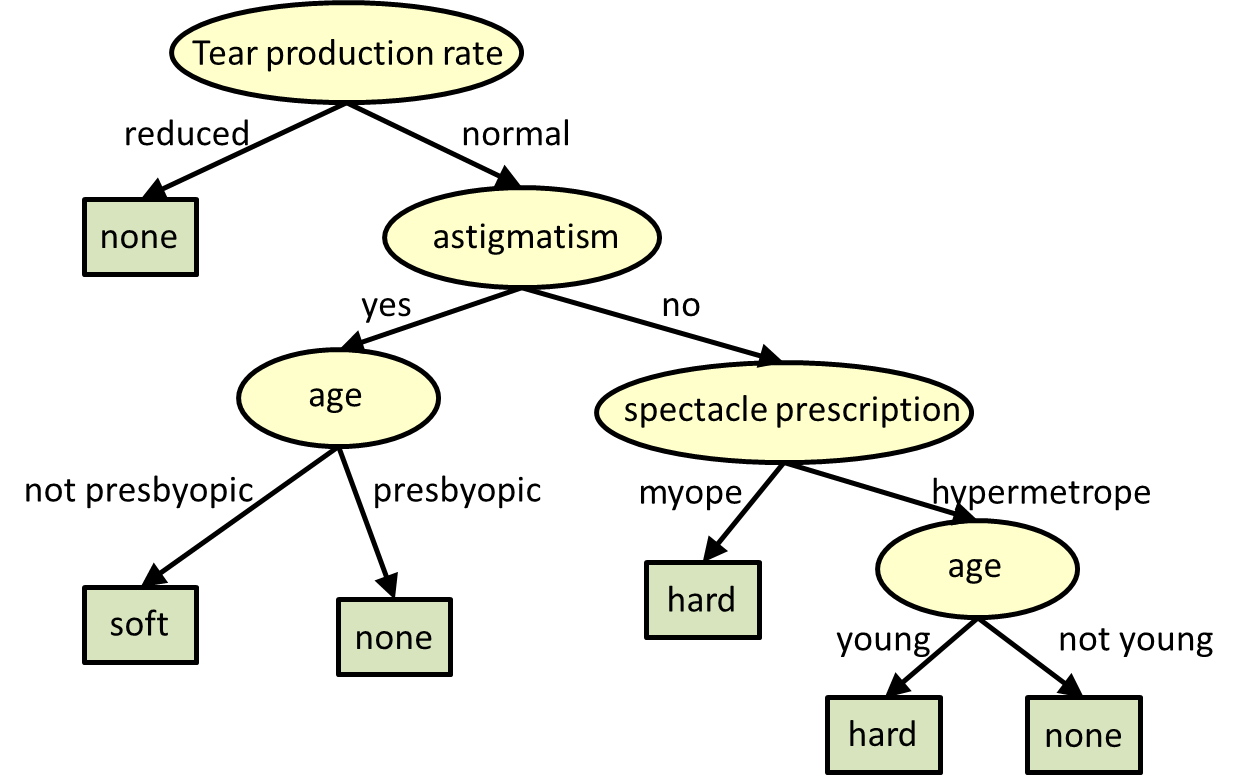
* ***DECISION TREE CLASSIFIER***

It is a classification technique used C5.0 method to assigning a variable at every node from top to bottom. So the variable gives max information gain during the split will be prefer according to that every variable is decided . the variable gives maximum information gain will be present at the top of tree

Information Gain = Entropy of system before split – Entropy of system after split

Entropy = uncertainity in the data / measure of impurity.

Decision tree look like this example:



Decision tree algorithm form by using the following commands:

#Import library

* from sklearn.tree import DecisionTreeClassifier
* dtree = DecisionTreeClassifier(criterion="entropy")
* dtree.fit(x\_train,y\_train)

#PREDICTION AND EVALUATION

* predictions\_dt = dtree.predict(x\_test)
* print(classification\_report(y\_test,predictions\_dt))

precision recall f1-score support

0 0.91 0.91 0.91 53980

1 0.19 0.20 0.19 6020

accuracy 0.84 60000

macro avg 0.55 0.55 0.55 60000

weighted avg 0.84 0.84 0.84 60000

#build confusion matrix

* from sklearn.metrics import confusion\_matrix
* CM = confusion\_matrix(y\_test, predictions\_dt)
* CM = pd.crosstab(y\_test, predictions\_dt)

#let us save TP, TN, FP, FN

* TN = CM.iloc[0,0]
* FN = CM.iloc[1,0]
* TP = CM.iloc[1,1]
* FP = CM.iloc[0,1]

#check accuracy of model

#recall

* print("recall", "=", (TP\*100)/(TP+FN))

#precision

* print("precision", "=" ,(TP\*100)/(TP+FP))

Now import library metrics to the AUC of decision tree

* metrics.roc\_auc\_score(y\_test, predictions\_dt)

**RESULT OF DECISION TREE**

**Recall = 19.53488372093023%**

**Precision = 19.174955160606554%**

**Auc =54.89%**

**Accuracy =84%**

* ***RANDOM FORREST CLASSIFIER***

As name suggest it uses many numbers of decision trees to form the forrest and that forrest uses to predict the value it uses essemble technique on that way error will be distributed.

It form many number number of trees by using the concept of bagging and boosting based on mode of result of that groups of trees we decide the result .

In bagging we form the bag of n variable from N total variable and m observation from total M observation

Where **n** **< N** and

**m<M**

In boosting we take out one variable at a time with replacement than take out second variable and do this process till we get n variable. Two variable in this process can be same. Same process is done to choose an observation.

It uses Gini Impurity to split at node, lesser the gini impurity more chances are for variable preferability

Form the random forest algorithm using the following command:

#import library

* from sklearn.ensemble import RandomForestClassifier
* rfc = RandomForestClassifier(n\_estimators=100)
* rfc.fit(x\_train, y\_train)

#predict function

* predictions\_rfc = rfc.predict(x\_test)
* print(classification\_report(y\_test,predictions\_rfc)

precision recall f1-score support

0 0.90 1.00 0.95 47405

1 1.00 0.00 0.00 5117

accuracy 0.90 52522

macro avg 0.95 0.50 0.47 52522

weighted avg 0.91 0.90 0.86 52522

#build confusion matrix

* CM = confusion\_matrix(y\_test, predictions\_rfc)
* CM = pd.crosstab(y\_test, predictions\_rfc)

#let us save TP, TN, FP, FN

* TN = CM.iloc[0,0]
* FN = CM.iloc[1,0]
* TP = CM.iloc[1,1]
* FP = CM.iloc[0,1]

#check accuracy of model

#recall

* print("recall", "=", (TP\*100)/(TP+FN))

#precision

* print("precision", "=" ,(TP\*100)/(TP+FP))

**RESULT OF RANDOM FORREST**

**n\_estimators=100**

**Recall = 0.019542700801250732**

**Precision = 100.0**

**Auc =50%**

**Accuracy =90%**

* ***NAÏVE BAYES***

It uses bayes laws of probability to predict the data . it assumes that all the variables are perfectly independent with each other , so we find probability

For each target variable on the bases of condition given. The target variable which gives max probability is considered as a result ,for example if probability of yes is more than no than no will be the result.

The formula used by it is:



The algorithm formed by using the following command:

#Naive Bayes

* from sklearn.naive\_bayes import GaussianNB

#Naive Bayes implementation

* NB\_model = GaussianNB().fit(x\_train, y\_train)

#predict test cases

* predictions\_nb = NB\_model.predict(x\_test)
* print(classification\_report(y\_test,predictions\_nb))

precision recall f1-score support

0 0.93 0.98 0.96 47405

1 0.72 0.36 0.48 5117

accuracy 0.92 52522

macro avg 0.83 0.67 0.72 52522

weighted avg 0.91 0.92 0.91 52522

#build confusion matrix

* CM = confusion\_matrix(y\_test, predictions\_nb)
* CM = pd.crosstab(y\_test, predictions\_nb)

#let us save TP, TN, FP, FN

* TN = CM.iloc[0,0]
* FN = CM.iloc[1,0]
* TP = CM.iloc[1,1]
* FP = CM.iloc[0,1]

#check accuracy of model

#recall

* print("recall", "=", (TP\*100)/(TP+FN))

#precision

* print("precision", "=" ,(TP\*100)/(TP+FP))

#import AUC

* metrics.roc\_auc\_score(y\_test, predictions\_nb)

**RESULT OF NAÏVE BAYES**

**Recall = 35.86085597029509%**

**Precision = 72.04554377699255%**

**Auc =67.08%**

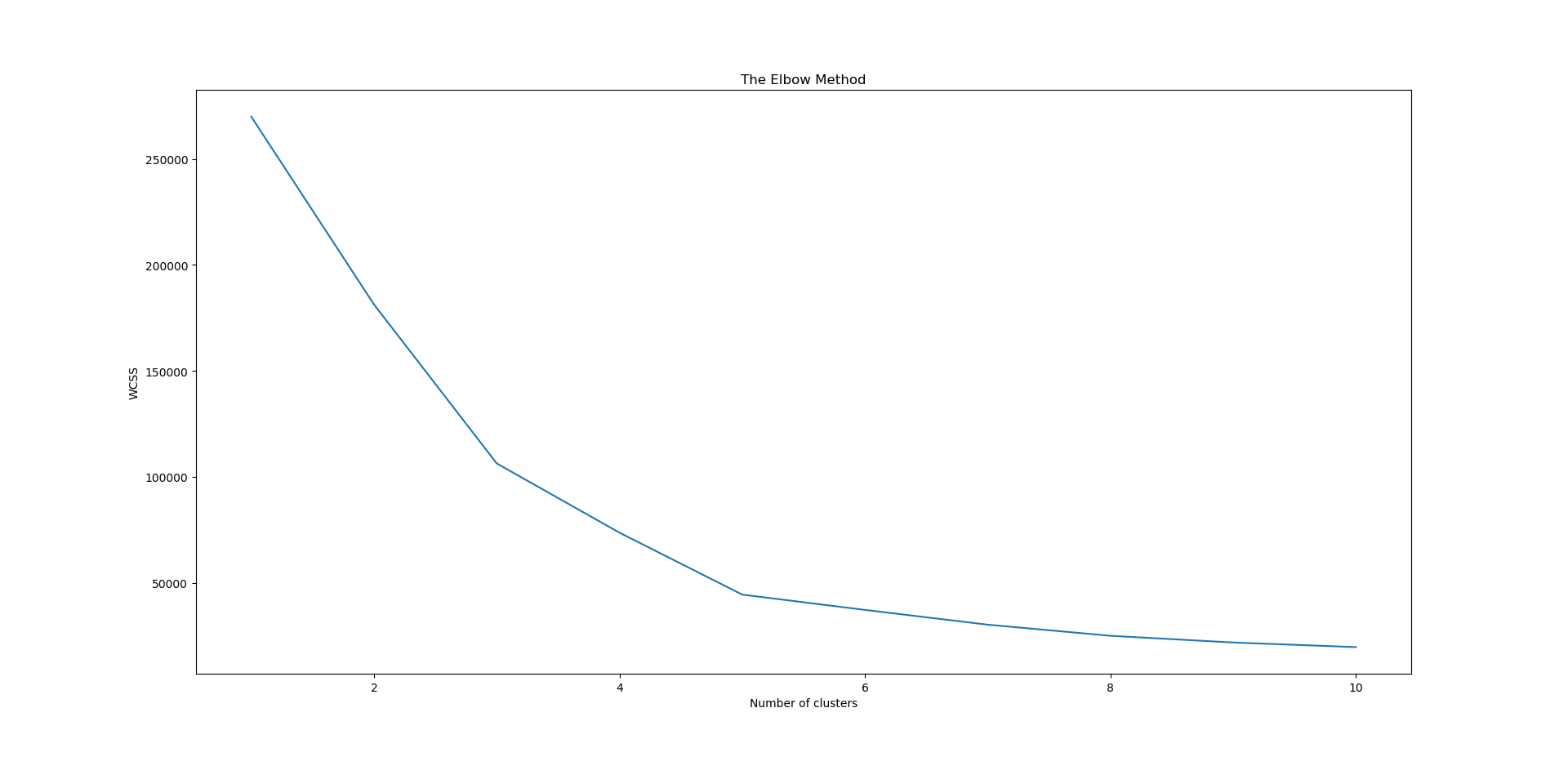
**Accuracy =92%**

* ***K-NEAREST NEIGHBOUR***

This method used Euclidean distance method to predict the value for categorical variable used mode of K nearest neighbor and for continuous variable use mode of K nearest neighbor.

We form a graph between the different K values and error to get most appropriate value of K it should form elbow curve least point of elbow curve gives us K value.

The elbow curve look like this:



**RESULT OF KNN**

HERE WE CANNOT USE K-NEAREST NEIGHBOUR METHOD BECAUSE KNN IS A LAZZY METHOD AND OUR DATA SET CONTAIN ABOUT 2 LAKH OBSERVATION AND 202 VARIABLES SO IT TAKES A VERY LONG TIME TO PREDICT THE DATA ALMOST INFINITE TIME.

**SUMMARY OF THE MODEL**

*ERRORS*

1. LOGISTIC REGRESSION:Results

* recall = 26.34356068008599
* precision = 69.05737704918033
* AUC =62.53%
* accuracy =92%

1. DECISION TREES:Results

* recall = 19.53488372093023%
* precision = 19.174955160606554%
* AUC =54.89%
* accuracy =84%

1. RANDOM FORREST:Results

For n\_estimators=100

* recall = 0.019542700801250732
* precision = 100.0
* AUC =50%
* accuracy =90%

1. NAIVE BAYES:Results

* recall = 35.86085597029509%
* precision = 72.04554377699255%
* AUC =67.08%
* accuracy =92%

1. KNN:Results

taking a very very long time.. can not use it to predict the result.

**CONCLUSION OF MODELLING**

**CRITERION OF PREDICTING THE MODEL**

PRECISION CRITERION AS FOLLOW:

RF> NAÏVE BAYES>LOGISTIC REGRESSION>DECISION TREE

RECALL CRITERION AS FOLLOW:

NAÏVE BAYES>LOGISTIC REGRESSION>DECISION TREE>RF

AUC CRITERION AS FOLLOW:

NAÏVE BAYES>LOGISTIC REGRESSION> DECISION TREE>RF

"To get the most accurate model values the recall, precision, AUC should be high".

According to the question we have to predict the result based on recall , precision and AUC. so after analysing the results of all four machine learning algorithm we can see that "Naive Bayes" is giving all the three parameters equally good. In Random Forest precision is high but recall is very low.

**WE USE NAÏVE BAYES TO PREDICT OUR MODEL BECAUSE THE COMBINATION OF ALL THREE MEASURING QUANTITY IS GOOD**

**PREDICTING OUR TEST DATA**

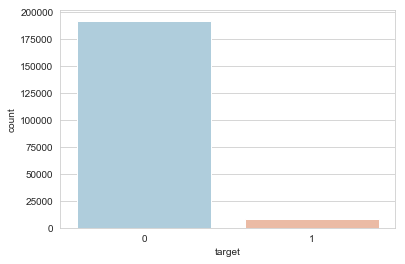
After forming the machine learning algorithm and test it for test data we have fetch out from train dataset given. And the get most accurate method

Now we are ready to predict any external data given to us.

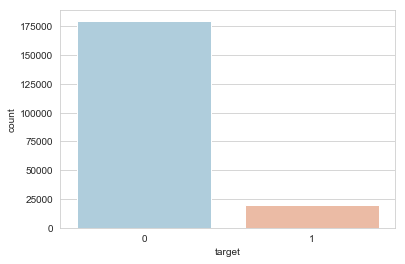
Now in the external data named as test.csv is given to us contain all variables of train data except “target” variable. Our task is to predict the target value in form of 0 and 1 where using the most appropriate algorithm we have formed in previous section.

As we have developed in our previous section that **NAÏVE BAYES** gives most accurate model so we will use **NAÏVE BAYES** to predict the target value.

1. So to do that first we upload test dataset into python
2. Now get the head of the test dataset to check the values it contain
3. Now form a vector contain the value of ID\_code which later join to dataset.
4. Now drop ID\_code column from dataset as it has no use and our naïve bayes model can not read this variable
5. Now predict the target variable using the this test dataset remains using the command **predictions\_test = NB\_model.predict(test).**
6. Now new datset is form using vector prediction\_test and id\_code having the variable name target and ID\_code respectively.
7. Now this dataset is join with the test dataset formed to predict the model
8. Now it upto us we can change 0 into “not buy” and 1 into “buy” string which is easy to understand for layman.
9. Now after predicting anf forming the finel dataset we will upload it as .csv file in our hard disc named as ***"test\_prediction result.csv"***
10. So the number of target values 0 and 1 formed as a prediction can be calculated using the function . count. The image formed is as follow:

******

the count present in train data is as follow:



**SAVED FILES DURING PROCESS :**

The file saved during the program is written with Italian font and blue in colour in this report at the place where it is formes. The test.csv and train.csv file is given in question and following files are formed during the course of project.

* ***Missing\_perc.csv:*** contain the number of missing value percentage in each column for a variable from maximum to minimum sort value. It show all the percentage are zero
* ***Outliers percentages in each column.csv:*** shows the percentage of outliers values present in each column with there variable name.
* ***Outliers number in each column.csv: :*** shows the number of outliers values present in each column with there variable name.
* ***outlier values.csv:***this dataset contain the train value dataset form after removing the outliers from it. So it is the outlier free dataset.
* ***test\_prediction result.csv***: this file formed after predicting the value from test.csv and add the target column contain predicted value as a new column of test.csv dataset

**IMPORTANT NOTE**

INSTRUCTION TO RUN AND DEPLOY THE PYTHON CODE IS WRITTEN WITH THE CODE ITSELF.

ONLY THE FILE LOCATION SHOULD BE CHANGED IN OS.CHDIR FUNCTION ACCORDING TO THE USER FILE LOCATION.

ALL THE SAVED FILE DURING THE PROCESS ARE GIVEN WITH CODE.

USER CAN SIMPLY USE SHIFT + ENTER TO GET ALL COMMAND RUN.

