

MERCEDEs-BENZ PROJECT

Thobelani Mkhize



November 28, 2019

**# Importing the Reliable libraries**

import pandas as pd

import numpy as np

import statistics as s

from sklearn import model\_selection, preprocessing

from sklearn.ensemble import AdaBoostClassifier

from sklearn import svm

from xgboost import XGBClassifier

from sklearn.model\_selection import train\_test\_split

import seaborn as sns

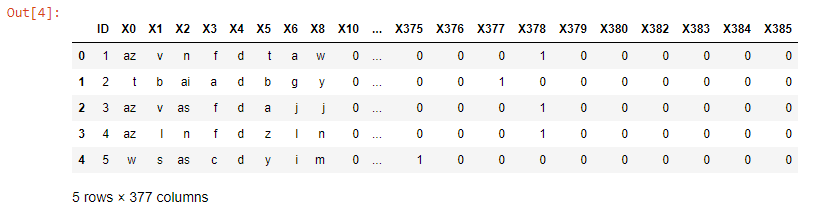
**# Loading the Datasets 'Test and Train' using the pandas DataFrame**

MecTest = pd.read\_csv('test.csv')

MecTrain = pd.read\_csv('train.csv')

MecTest.head()

**Output:**



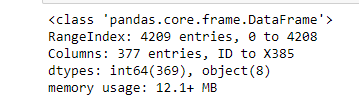
**# Checking the structure of the dataset and if is there any null values**

MecTest.info()

MecTest.describe()

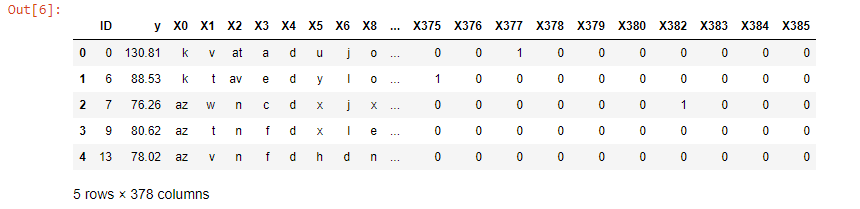
#MecTest.isnull().sum()

**Output:**



MecTrain.head()

**Output:**



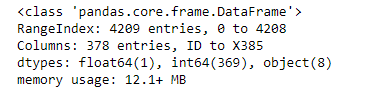
**# Checking the structure of the dataset and if is there any null values**

MecTrain.info()

MecTrain.describe()

#MecTrain.isnull().sum()

**Output:**



**#Data Conversion for the Test Dataset**

for f in MecTest.columns:

if MecTest[f].dtype == 'object':

lbl = preprocessing.LabelEncoder()

lbl.fit(list(MecTest[f].values))

MecTest[f] = lbl.transform(list(MecTest[f].values))

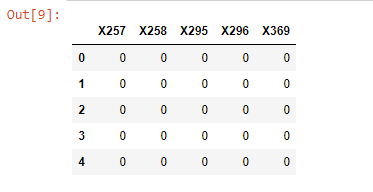
**Q One**

**# Checking the variance for the Test data set**

variance = MecTest.loc[:,MecTest.var() == 0.0]

variance.head()

**Output:**



**# Removing the variable where the variance equal to zero**

MecTest.drop(['X257','X258','X295','X296','X369'], inplace = True, axis = 1)

MecTest.shape

**#Data Conversion for the Train Dataset**

for f in MecTrain.columns:

if MecTrain[f].dtype == 'object':

lbl = preprocessing.LabelEncoder()

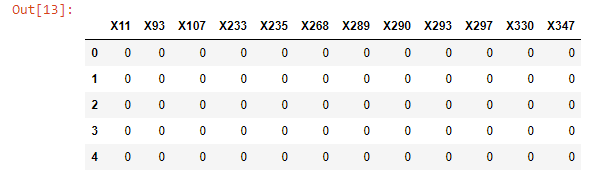
lbl.fit(list(MecTrain[f].values))

MecTrain[f] = lbl.transform(list(MecTrain[f].values))

variance = MecTrain.loc[:,MecTrain.var() == 0.0]

variance.head()

**Output:**



**# Removing the variable where the variance equal to zero**

MecTrain.drop(['X11','X93','X107','X233','X235','X268','X289','X290','X293','X297','X330','X347'], inplace = True, axis = 1)

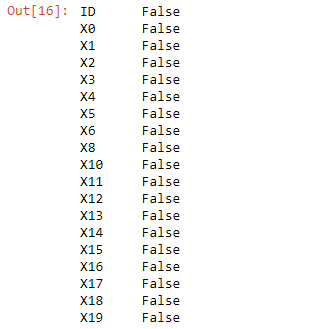
MecTrain.shape

**Q Two**

**# Checking Null Values for the Test and Train sets**

MecTest.isnull().any()

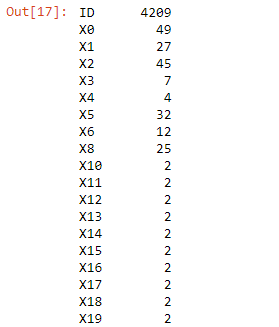
**Output:**



**# Checking for unique values for the Test Set**

MecTest.nunique()

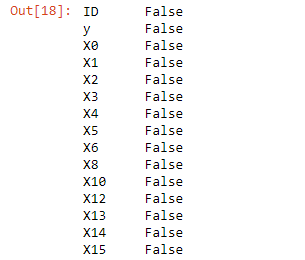
**Output:**



**# Checking for the null and unique values in Train set**

MecTrain.isnull().any()

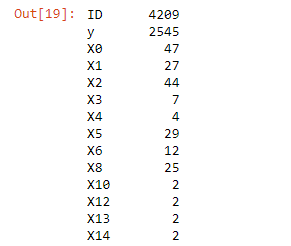
**Output:**



**# Checking for unique values for the Train Set**

MecTrain.nunique()

**Output:**



**Q Tree**

**#Applying Label Encoder**

**#Data Conversion for the Test Dataset**

for f in MecTest.columns:

if MecTest[f].dtype == 'object':

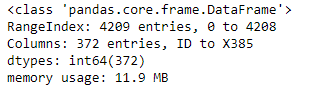
lbl = preprocessing.LabelEncoder()

lbl.fit(list(MecTest[f].values))

MecTest[f] = lbl.transform(list(MecTest[f].values))

MecTest.info()

**Output:**



**#Applying Label Encoder**

**#Data Conversion for the Train Dataset**

for f in MecTrain.columns:

if MecTrain[f].dtype == 'object':

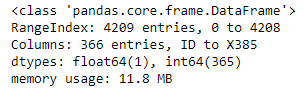
lbl = preprocessing.LabelEncoder()

lbl.fit(list(MecTrain[f].values))

MecTrain[f] = lbl.transform(list(MecTrain[f].values))

MecTrain.info()

**Output:**



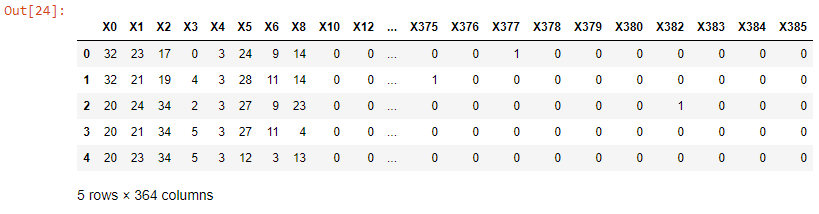
**# Separating x and y**

**#Removing the ID and the Y column from the train data**

x\_train = MecTrain.iloc[:,2:379]

x\_train.head()

**Output:**

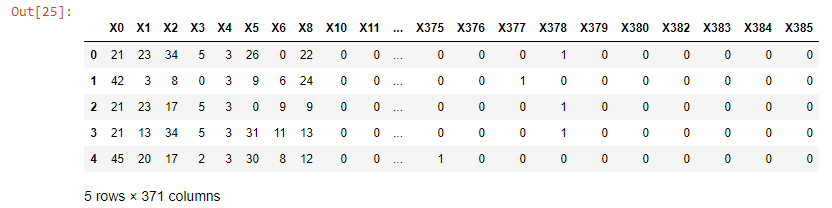


**# Removing the Y column from the test dataset**

x\_test = MecTest.iloc[:,1:]

x\_test.head()

**Output:**



print(x\_test.shape)

print(x\_train.shape)

print(y\_train.shape)

x\_train.info()

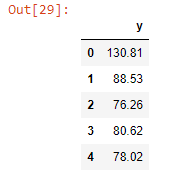
y\_train2 = MecTrain['y']

y\_train2.shape

y\_train = MecTrain.iloc[:,1:2]

y\_train.head()

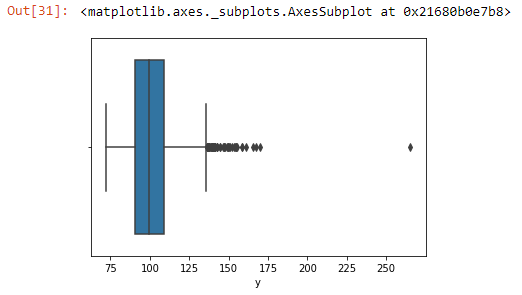
**Output:**



**# checking the outliers using Boxploting**

sns.boxplot(y\_train2)

**Output:**



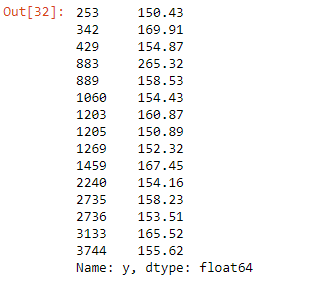
**# Removing outliers from y\_train**

y\_train3 = filter = y\_train2.values > 150

Outl\_Rem = y\_train2[filter]

Outl\_Rem

**Output:**



print(y\_train3.shape)

**QFour**

**# Perform dimensionality reduction PCA**

from sklearn.decomposition import PCA

**# initialising the model**

modelPCA = PCA(n\_components = 8)

# Fitting the model for x\_train

modelPCA.fit(x\_train)

**Output:**



**# Tranform the data in to 8 dimentions**

dim = modelPCA.transform(x\_train)

dim.shape

**# Fitting the model f0r the x\_test**

modelPCA.fit(x\_test)

**Output:**



**# Tranform the data in to 8 dimentions**

dimTest = modelPCA.transform(x\_test)

dimTest.shape

**Q Five**

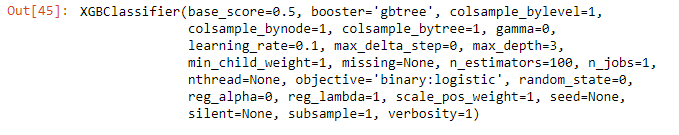
**# Predict the test data frame values using XGBoost**

import xgboost as xgb

xgb\_classifier = xgb.XGBClassifier()

xgb\_classifier.fit(dim,y\_train3)

**Output:**



pred = xgb\_classifier.predict(dimTest)

pred

**Output:**



**# finding the predictions**

from sklearn.metrics import accuracy\_score

print('Accuracy of the model:', accuracy\_score(y\_train3,pred))

**Output:**

# The accuracy is 99%

