DATA SCIENCE PROJECT

Student Adaptivity in Online Education

Syed Muhammad Ahmed

023-19-0004 | BS CS VI

syedmuhammad.bscsf19@iba-suk.edu.pk

Abstract – The aim of this project is to visualize the dataset of student adaptivity in online education. The dataset presents the statistics of year 2022 collected from different institutions such as government and non-governmental. The data covers the aspects of gender involvement in the learning, network usage and availability, and the educational background. This project covers to visualize the dataset by mean of different data science metrics such as histogram visualization, and scatter plotting. By the end of this report, Logistics regression model is used to show the ultimate results using accuracy and other metrics.

Table of Contents

1. About the Dataset

The dataset covers the aspects of student adaptivity in online education. The purpose is to gain an insight of how student responded to distant learning in COVID'19 terms. This data can be used to analyse gender-based response, network availability, provision of facilities in governmental and non-governmental institutions and type of device better for distant learning. The dataset needs to be pre-processed to get analytical information using data science and statistical metrics.

2. Pre-processing and cleaning the dataset

2.1. Textual Data

The data presented in the dataset was in textual form. Hence records needed to be transformed to numeric format. Some attributes like gender, institution type, and Self LMS was shifted to binary format of data leaving other data to numeric with three or more values. The transformation is done using `LabelEncoder()` function from Pandas library in Python. This operation was performed separately among all required attributes.

2.2. Data with anomaly

Some attributes in the dataset such as Age and Class duration had invalid values. Since age is meant to be in a valid range i.e., an integer greater than 0, however, it contained information of a data without even a year. Hence the best way found to be to eliminate the attribute. The attribute age plays a very minor role in determining the adaptability.

3. Visualization

3.1. Dependencies

The dataset used libraries such as pandas, sklearn, matplotlib for modelling.

```
import numpy as np
from sklearn.utils import resample
import seaborn as sns
import matplotlib.pylab as plt
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV as cv
from sklearn.metrics import classification_report, roc_auc_score, confusion_matrix
from sklearn.preprocessing import OneHotEncoder, LabelEncoder
from sklearn.svm import SVC
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import BaggingClassifier, HistGradientBoostingClassifier
from sklearn.experimental import enable_hist_gradient_boosting
from sklearn.ensemble import HistGradientBoostingRegressor
from sklearn.neighbors import KNeighborsClassifier,NeighborhoodComponentsAnalysis
from sklearn.pipeline import Pipeline
from imblearn.over_sampling import RandomOverSampler
import plotly.express as px
import io

from google.colab import files


uploaded = files.upload()
```

```
students_a...ducation.csv
• students_adaptability_level_online_education.csv(text/csv) - 101695 bytes, last modified: 5/23/2022 - 100% done
Saving students_adaptability_level_online_education.csv to students_adaptability_level_online_education (1).csv
```

3.2. Encoding to numerical values

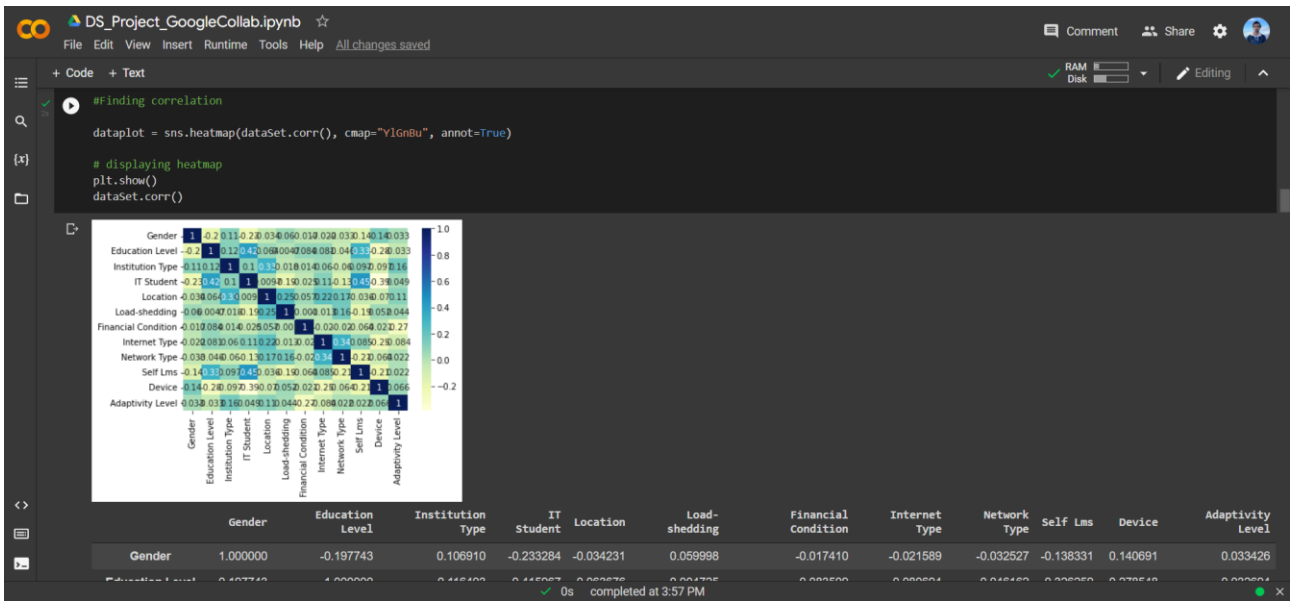
The LabelEncoder() is used to transform values to numbers for training the model.

```
[15] dataSet['Self Lms'] = LabelEncoder().fit_transform(enCode)

     enCode = dataSet['Device'].values
     dataSet['Device'] = LabelEncoder().fit_transform(enCode)

     enCode = dataSet['Adaptivity Level'].values
     dataSet['Adaptivity Level'] = LabelEncoder().fit_transform(enCode)

     #Removing 'Age' and 'Class Duration' attribute as it is not of focus.
     dataSet.pop('Age')
     dataSet.pop('Class Duration')

     #Removing null values if exist.
     dataSet = dataSet.dropna();

     #Display dataframe
     dataSet.head()
```

	Gender	Education Level	Institution Type	IT Student	Location	Load-shedding	Financial Condition	Internet Type	Network Type	Self Lms	Device	Adaptivity Level
0	0	2	1	0	1	1	0	1	2	0	2	2
1	1	2	1	0	1	0	0	0	2	1	1	2
2	1	0	0	0	1	1	0	1	2	0	1	2
3	1	1	1	0	1	1	0	0	2	0	1	2
4	1	1	1	0	1	1	1	0	1	0	1	1

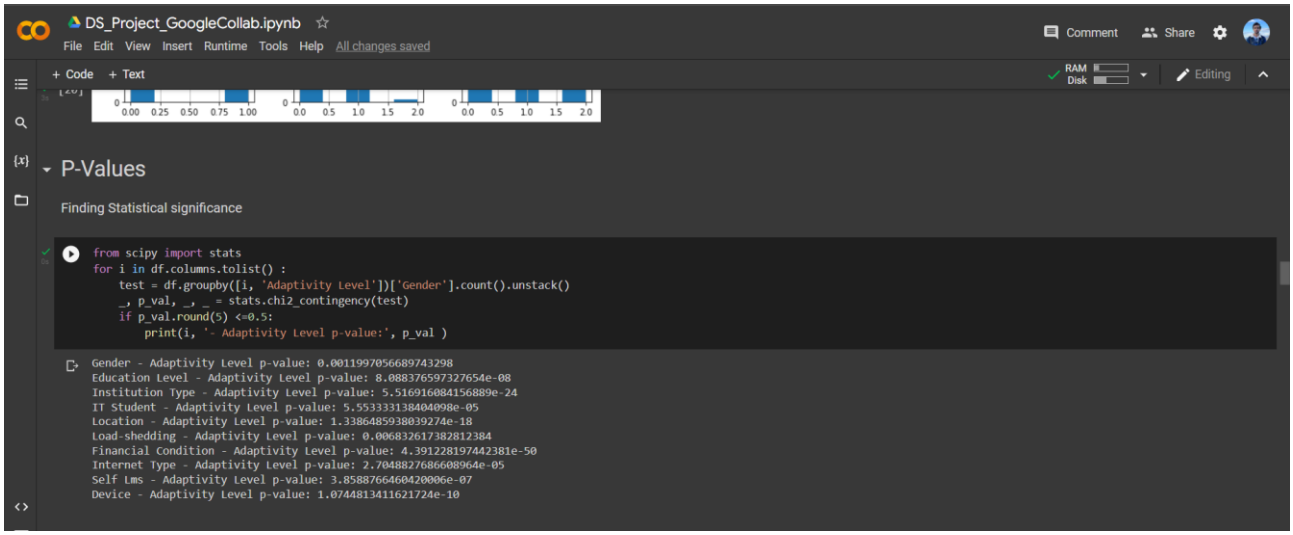
3.3. Correlation

The dataset is visualized using correlation metric. The data is more towards the half of the correlation scale.



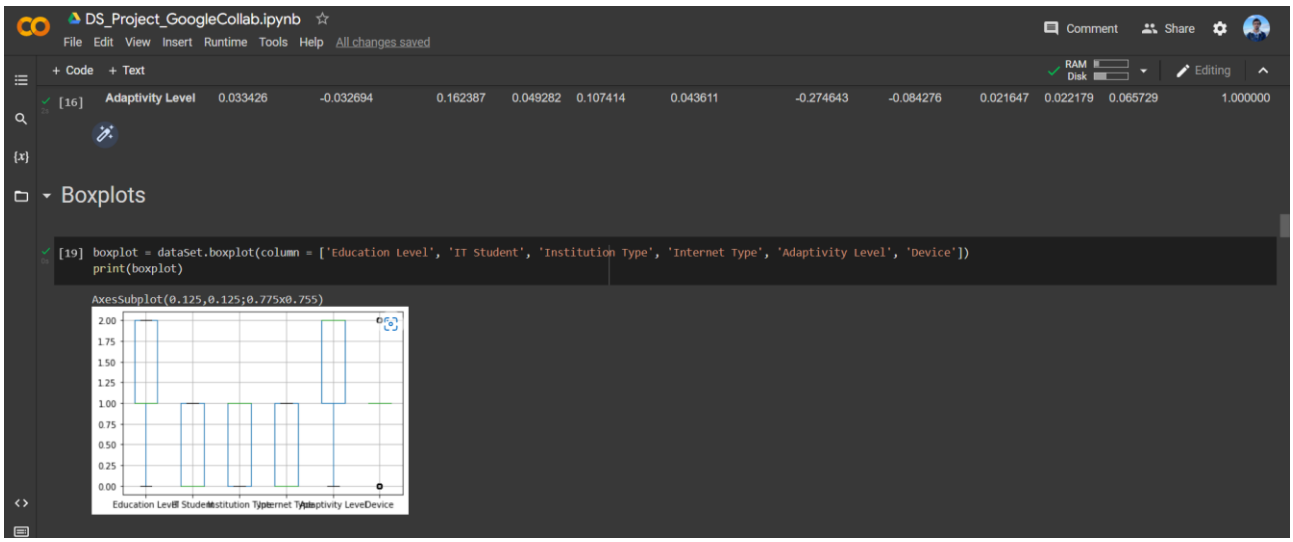
3.4. P-values of adaptivity level

The P-values with respect to adaptivity level is quite less hence the data is statistically significant.



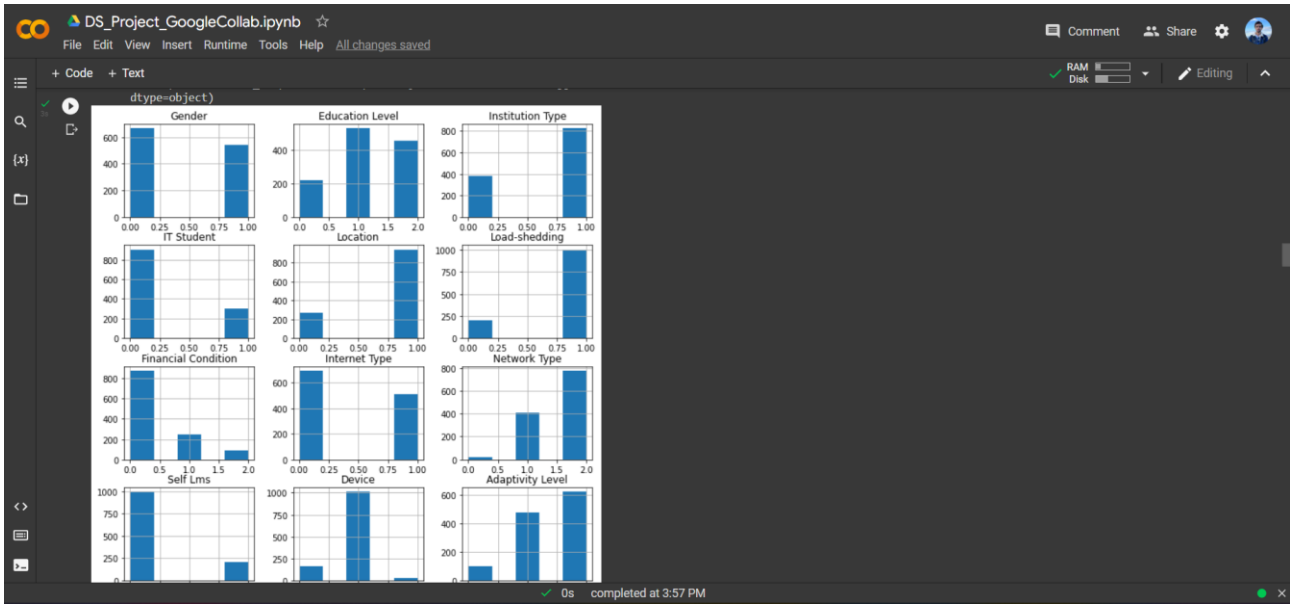
3.5. Boxplots

Since the data set is based on binary values hence the boxplot does not show much metric for these values.



3.6. Histograms

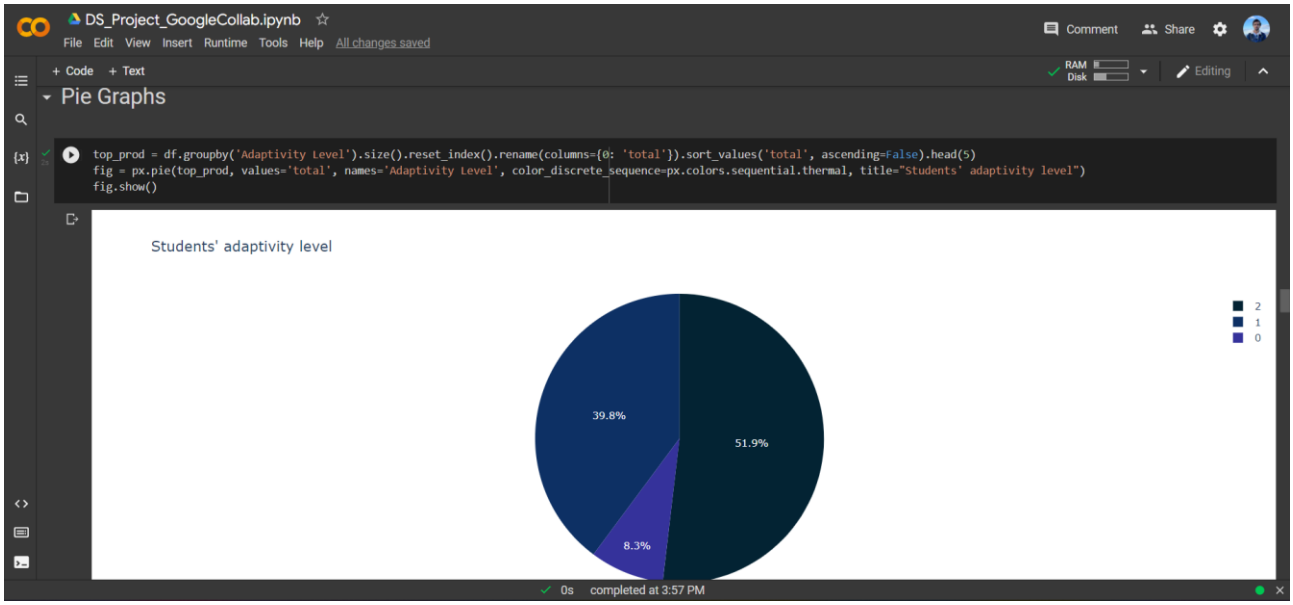
The histogram of all attributes shows balance among all. However, only “Adaptivity Level” shows imbalance which may hinder the model training, hence more work is done later.



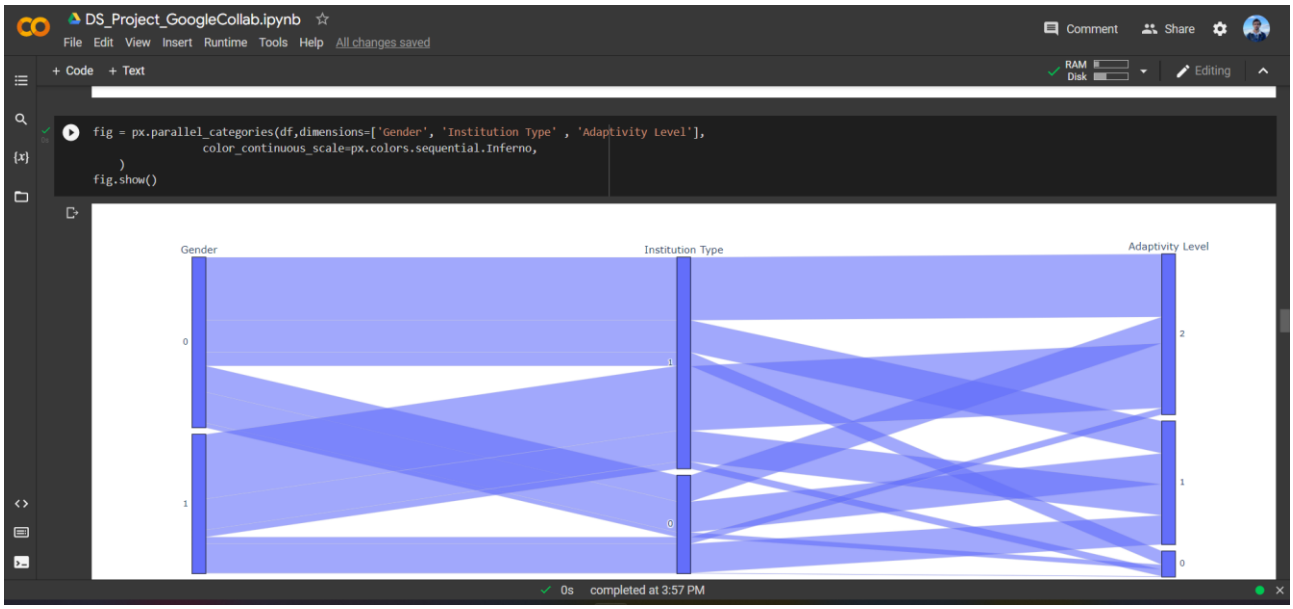
3.7. Scatterplot

Since the dataset is based on binary data, so scatterplots does not work well.

3.8. Pie-Graphs

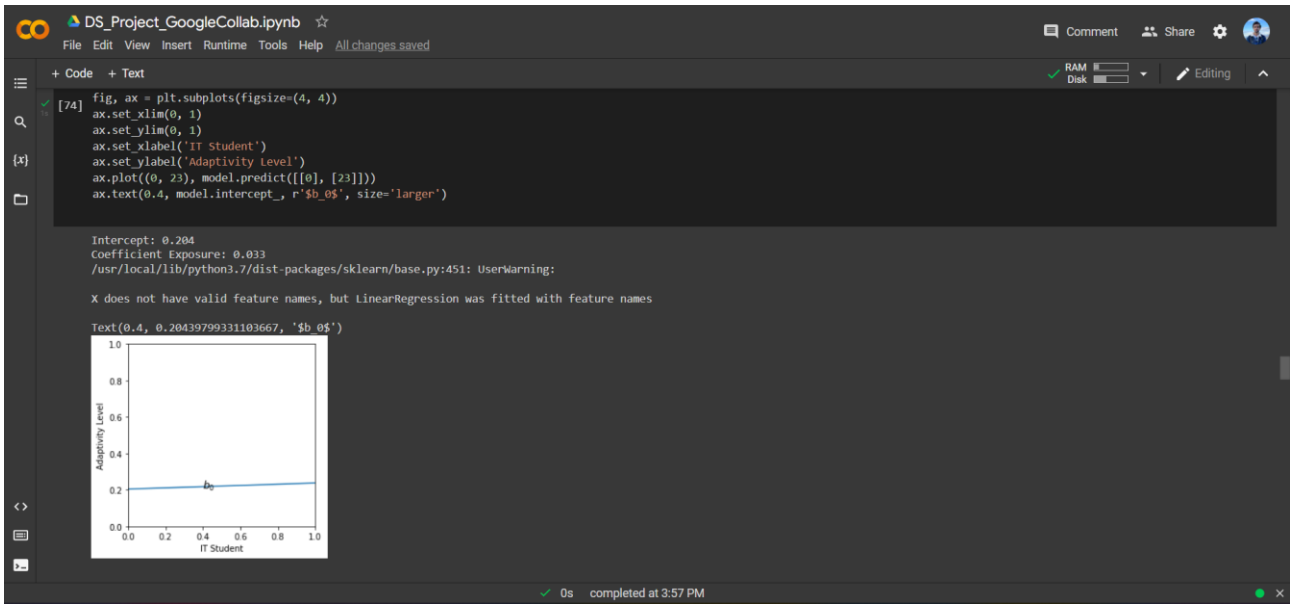


3.9. Parallel Graphs

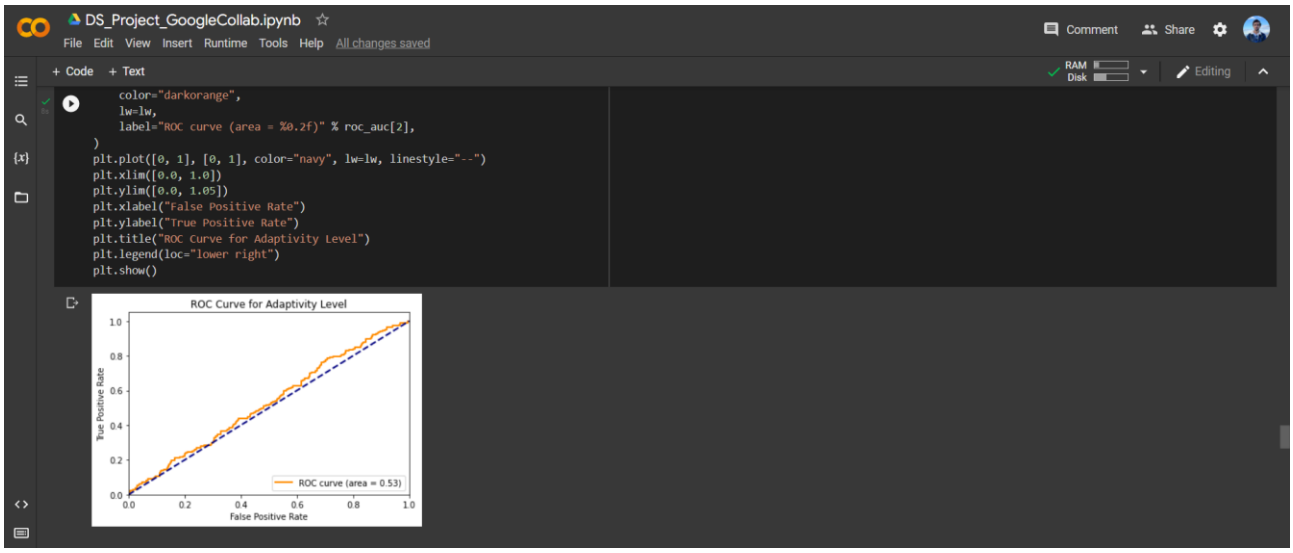


3.10. Linear Regression Model

The model is built on attributes “IT Student” vs. “Adaptivity Level”.



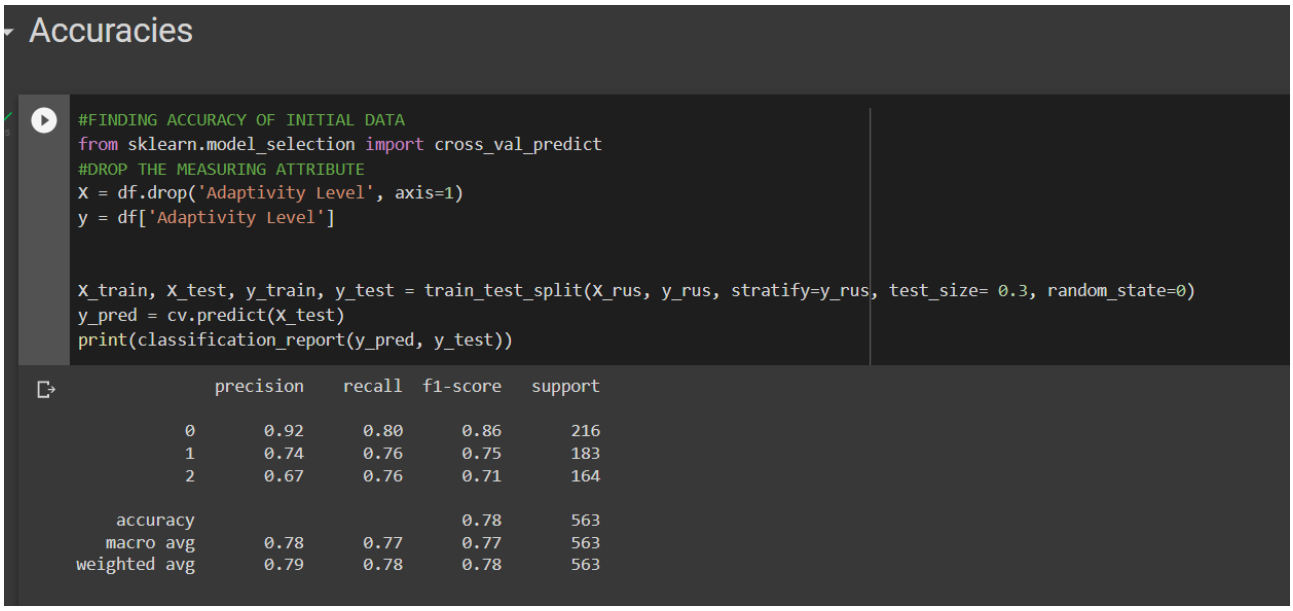
3.11. ROC Curve



3.12. Accuracy metric on initial data

3.12.1. Classification Report

Accuracy: 0.78



3.13. KNN Classifier

Accuracy: 0.81

DS_Project_GoogleCollab.ipynb

File Edit View Insert Runtime Tools Help

+ Code + Text

```
[81] #USING KNN CLASSIFIER TO IMPROVE ACCURACY

     knc = KNeighborsClassifier(n_neighbors=3).fit(X_train, y_train)
     y_pred = knc.predict(X_test)
     print(classification_report(y_pred, y_test))
```

```
              precision    recall  f1-score   support

           0       0.83      0.92      0.87       169
           1       0.84      0.80      0.82       197
           2       0.78      0.74      0.76       197

    accuracy                           0.81       563
   macro avg       0.81      0.82      0.81       563
weighted avg       0.81      0.81      0.81       563
```

3.13.1. Bagging Classifier and Histogram Gradient Boosting Classifier

Accuracy by Bagging Classifier: 0.76

Accuracy by Histogram Gradient Boosting Classifier: 0.87

DS_Project_GoogleCollab.ipynb

File Edit View Insert Runtime Tools Help

+ Code + Text

```
[85] bagg = BaggingClassifier(base_estimator= knc)
     params = {'max_features':np.linspace(0.1, 1, 10), 'max_samples':np.linspace(0.1, 0.2, 10)}
     cv = GridSearchCV(bagg, params, cv=5)
     cv.fit(X_train, y_train)
     y_pred = cv.predict(X_test)
     print(classification_report(y_pred, y_test))
```

```
              precision    recall  f1-score   support

           0       0.92      0.82      0.87       211
           1       0.77      0.71      0.74       205
           2       0.60      0.76      0.67       147

    accuracy                           0.76       563
   macro avg       0.76      0.76      0.76       563
weighted avg       0.78      0.76      0.77       563
```

```
hgbc = HistGradientBoostingClassifier(loss='categorical_crossentropy')
hgbc.fit(X_train, y_train)
y_pred = hgbc.predict(X_test)
print(classification_report(y_pred, y_test))
```

```
              precision    recall  f1-score   support

           0       0.95      0.88      0.92       203
           1       0.87      0.84      0.86       193
           2       0.79      0.88      0.83       167

    accuracy                           0.87       563
   macro avg       0.87      0.87      0.87       563
weighted avg       0.87      0.87      0.87       563
```

✓ 1s completed at 4:09 PM

3.14. Model Fitting

3.14.1. Logistic Regression Classifier

Accuracy: 0.62 on train data

Accurate: 0.67 on test data

```
DS_Project_GoogleCollab.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text

[90] lr = LogisticRegression(solver='newton-cg').fit(X_train,y_train)
     print('train data report')
     print(classification_report(lr.predict(X_train),y_train))
     print('test data report')
     print(classification_report(lr.predict(X_test),y_test))

     train data report
                   precision    recall  f1-score   support

                0       0.63      0.67      0.65       409
                1       0.59      0.59      0.59       435
                2       0.63      0.59      0.61       468

         accuracy                           0.62      1312
        macro avg       0.62      0.62      0.62      1312
     weighted avg       0.62      0.62      0.62      1312

     test data report
                   precision    recall  f1-score   support

                0       0.61      0.75      0.67       152
                1       0.69      0.62      0.65       209
                2       0.67      0.62      0.64       202

         accuracy                           0.65       563
        macro avg       0.65      0.66      0.65       563
     weighted avg       0.66      0.65      0.65       563
```

3.14.2. KNN Classifier

Accuracy on training: 0.86

Accuracy on testing: 0.88

```
DS_Project_GoogleCollab.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text

KNN = KNeighborsClassifier(n_neighbors=40,weights="distance").fit(X_train,y_train)
print('Train')
print(classification_report(KNN.predict(X_train),y_train))
print('Test')
print(classification_report(KNN.predict(X_test),y_test))

Train
              precision    recall  f1-score   support

           0       0.96      0.83      0.89       503
           1       0.92      0.85      0.88       471
           2       0.71      0.92      0.80       338

    accuracy                           0.86      1312
   macro avg       0.86      0.87      0.86      1312
weighted avg       0.88      0.86      0.87      1312

Test
              precision    recall  f1-score   support

           0       0.95      0.87      0.91       205
           1       0.90      0.85      0.88       200
           2       0.78      0.92      0.84       158

    accuracy                           0.88       563
   macro avg       0.88      0.88      0.88       563
weighted avg       0.89      0.88      0.88       563

✓ 0s completed at 4:12 PM
```

4. Conclusion

The model is mainly based on binary values and found to be give more accuracy under KNN-classifier. Although, the imbalance among one attribute was affecting results on initial results. Therefore, the dataset was trained and tested under different classifiers. Some of the classifiers are beyond book study just to confirm the results.