Data Mining and Processing

Yusif Imamverdiyev
Rana Najafzade
Natavan Mammadova

# Project Report

## 1. Dataset Explanation

We have 7 different columns in this dataset with explanations as below:

*Bying* - Bying Rate of the Car

*Maintenance* - Maintenance Rate of the Car

*Doors* - Number of Doors in the Car

*Persons* - Number of Persons that the Car can take

*Luggage Boot* - Size of the Luggage Boot of the Car

*Safety* - Safety Rate of the Car

Class of the Car determining its acceptance which is our target column

:

| | buying | maintainance | doors | persons | lug_boot | safety | class |
|---|---|---|---|---|---|---|---|
| **0** | vhigh | vhigh | 2 | 2 | small | low | unacc |
| **1** | vhigh | vhigh | 2 | 2 | small | med | unacc |
| **2** | vhigh | vhigh | 2 | 2 | small | high | unacc |
| **3** | vhigh | vhigh | 2 | 2 | med | low | unacc |
| **4** | vhigh | vhigh | 2 | 2 | med | med | unacc |
| **...** | ... | ... | ... | ... | ... | ... | ... |
| **1723** | low | low | 5more | more | med | med | good |
| **1724** | low | low | 5more | more | med | high | vgood |
| **1725** | low | low | 5more | more | big | low | unacc |
| **1726** | low | low | 5more | more | big | med | good |
| **1727** | low | low | 5more | more | big | high | vgood |

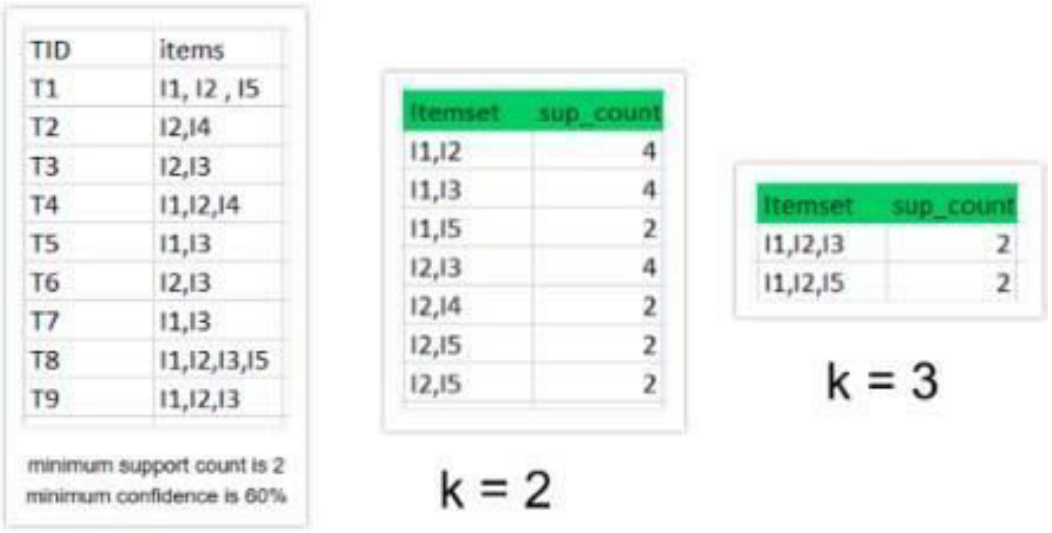
## 2. Problem Description

The main goal of this project is to use other columns to anticipate the target values (column Class). To put it another way, we should be able to forecast whether or not a car would be accepted. This is a classification problem, but before we can create our classifier from scratch or use other Machine Learning models, we must first complete numerous processing stages.

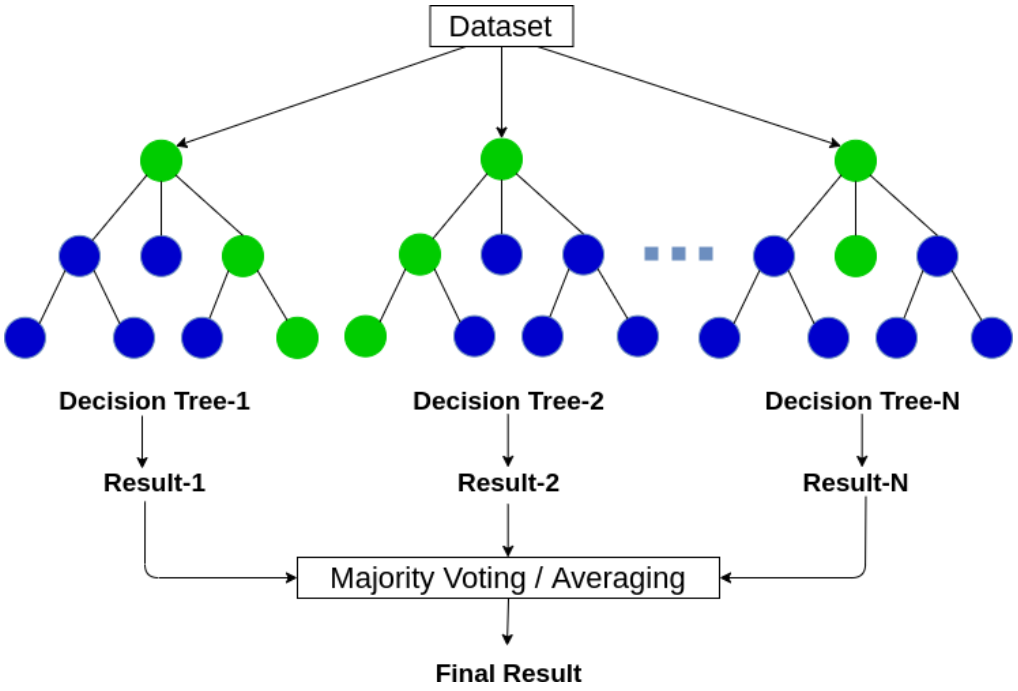
There are a lot of algorithms for doing such classification tasks including Apriori Algorithm, Logistic Regression, K-Nearest Neighbors, Naïve Bayes, Support Vector Classifier, Random Forest, Decision Tree, Neural Networks, and others. In our mandatory task, we choose Apriori Algorithm for several purposes such as counting itemset support counts, finding confidence rates and creating necessary association rules. As an extra tasks, we also tried other Machine Learning or Data Mining technique Random Forest

## 3. Algorithm Explanations

The Apriori algorithm is a well-known Data Mining technique that uses mathematical procedures to find useful associations between distinct elements in a collection. First and foremost, this method counts the number of supporters for each item and itemset. The frequent items are then identified based on the user's threshold. By providing adequate parameters, the SPMF tool can assist in the proper application of the Apriori algorithm. Following the acquisition of a frequent itemset, a classifier can be constructed using confidence rates and association rules. The process of locating frequent itemset is depicted in the diagram below.



For the classification problem, we employed the next supervised learning method, Random Forest. It's similar to the Decision Tree algorithm, except it's built up of numerous decision trees. In other words, in Random Forest, the nodes we saw in the previous method are represented as decision trees. The bagging technique is commonly used to train the tree forest that the algorithm builds. The bagging strategy entails merging numerous learning models to improve the final model's overall accuracy. The example tree forest in the image below can help you better understand the Random Forest Algorithm process.



## 4. Data Visualization and Processing for Modeling

Firstly we collect some basic stats about the data and check data types

```
5]: # some basic stats about the data
    df.describe()
```

5]:

| | buying | maintainance | doors | persons | lug_boot | safety | class |
|---|---|---|---|---|---|---|---|
| **count** | 1728 | 1728 | 1728 | 1728 | 1728 | 1728 | 1728 |
| **unique** | 4 | 4 | 4 | 3 | 3 | 3 | 4 |
| **top** | low | low | 2 | 2 | big | low | unacc |
| **freq** | 432 | 432 | 432 | 576 | 576 | 576 | 1210 |

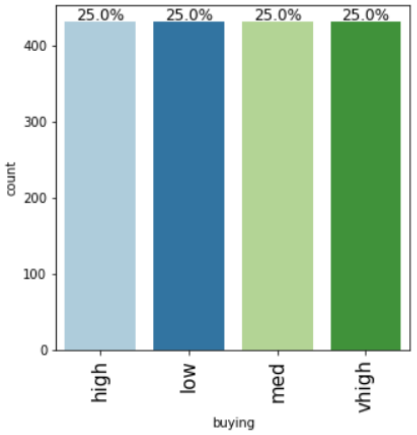
```
[46]: # checking the data types of features
      df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1728 entries, 0 to 1727
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   buying        1728 non-null   object
 1   maintainance  1728 non-null   object
 2   doors         1728 non-null   object
 3   persons       1728 non-null   object
 4   lug_boot      1728 non-null   object
 5   safety        1728 non-null   object
 6   class         1728 non-null   object
dtypes: object(7)
memory usage: 94.6+ KB
```

There are 4 classes in the dataset we have to convert it to binary classification prioblem also we encode the classes by numbers. We can see that the dataset is imbalanced we will deadl with it later in the code. lets analyze other features. And we have some labeled barplots for see data more visually

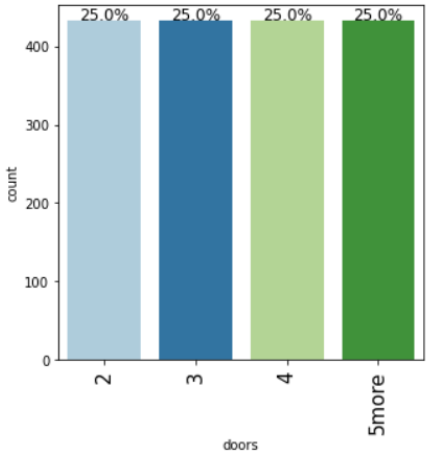
**Observation on buying**

```
In [51]: labeled_barplot(df, "buying", perc=True)
```



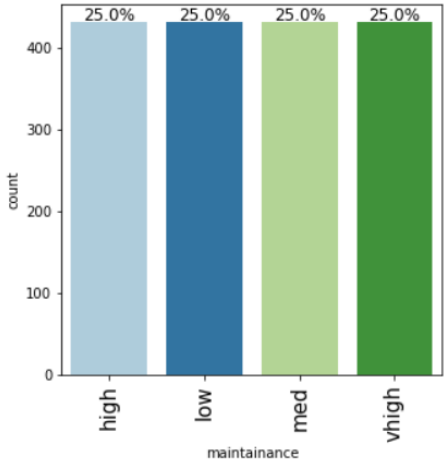

**Observation on doors**

```
In [53]: labeled_barplot(df, "doors", perc=True)
```



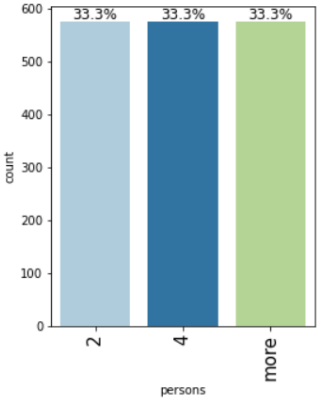

**Observation on maintainance**

```
2]: labeled_barplot(df, "maintainance", perc=True)
```



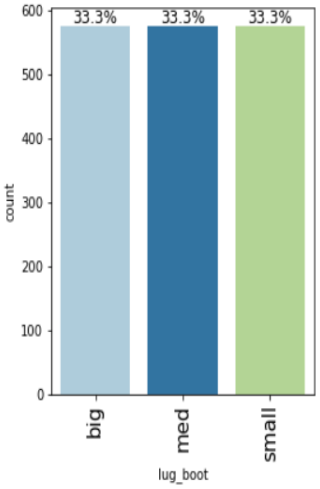

**Observation on persons**

```
In [54]: labeled_barplot(df, "persons", perc=True)
```



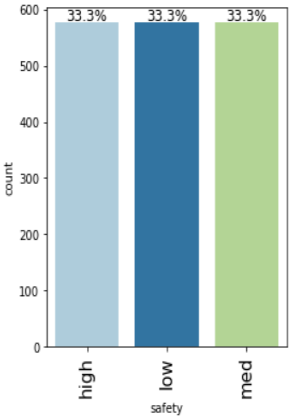

**Observation on lug_boot**

```
In [55]: labeled_barplot(df, "lug_boot", perc=True)
```




**Observation on safety**

```
In [56]: labeled_barplot(df, "safety", perc=True)
```

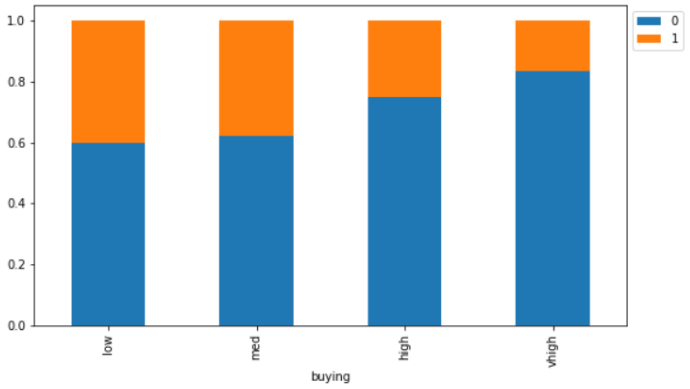



From the all plots above we can see that the dataset has same numbers of all features lets do bi variate analysis

In [59]:
```
for i in X.columns:
    stacked_barplot(df, i, "class")
```

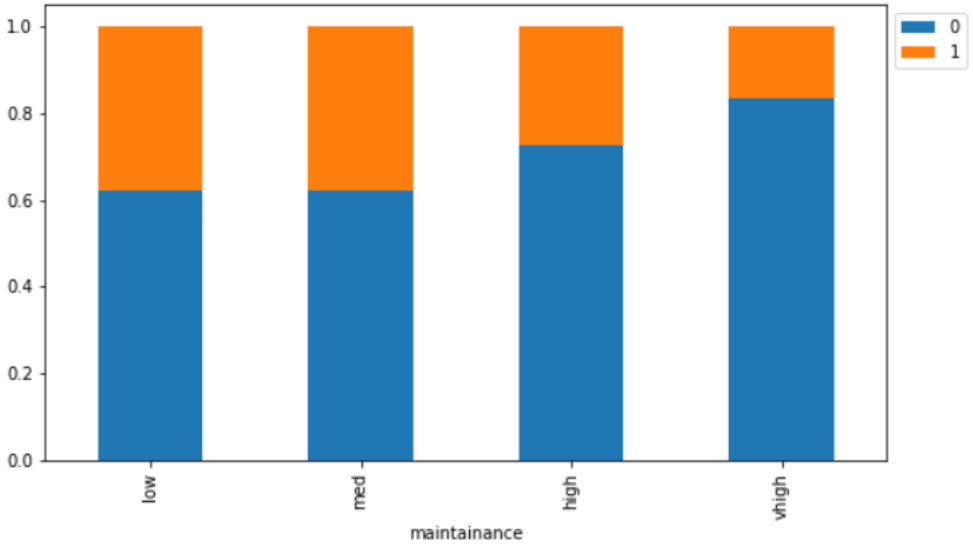
```
class      0    1   All
buying
All     1210  518  1728
low      258  174   432
med      268  164   432
high     324  108   432
vhigh    360   72   432
```

---



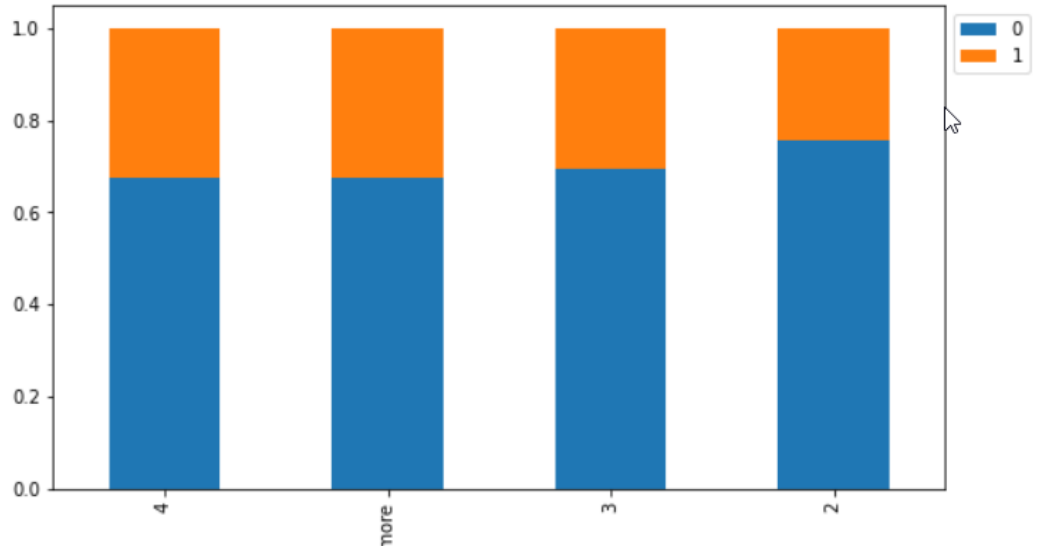

```
class            0    1   All
maintainance
All           1210  518  1728
low            268  164   432
med            268  164   432
high           314  118   432
vhigh          360   72   432
```

---



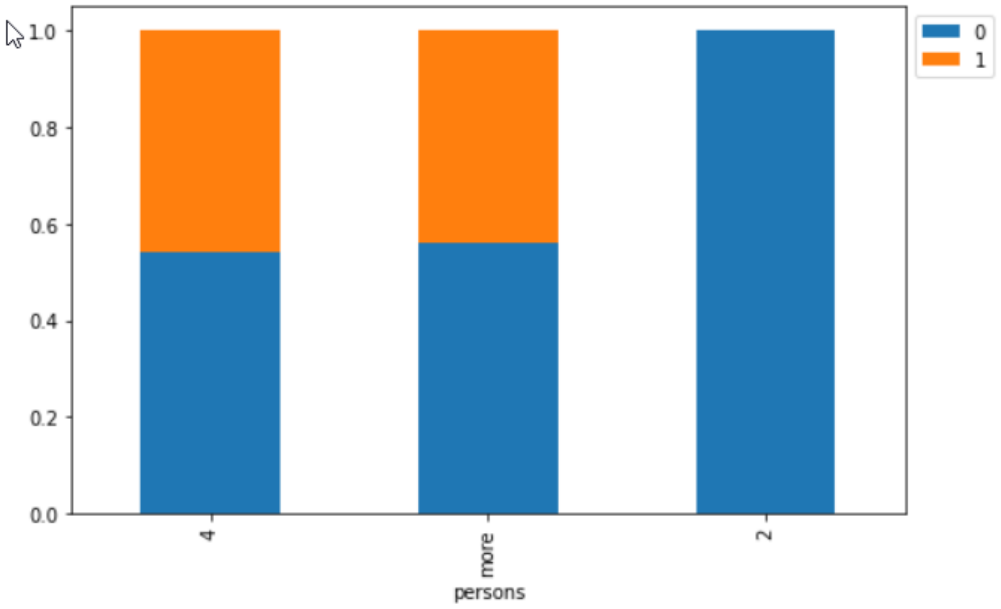
| class | 0 | 1 | All |
|---|---|---|---|
| doors | | | |
| All | 1210 | 518 | 1728 |
| 4 | 292 | 140 | 432 |
| 5more | 292 | 140 | 432 |
| 3 | 300 | 132 | 432 |
| 2 | 326 | 106 | 432 |

---



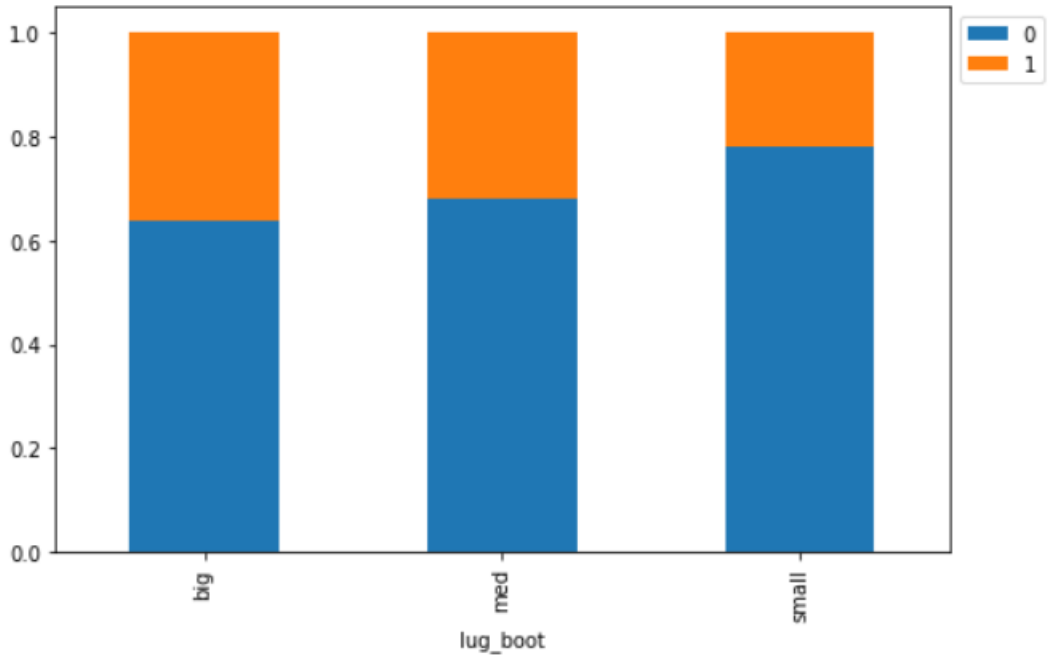

| class | 0 | 1 | All |
|---|---|---|---|
| persons | | | |
| All | 1210 | 518 | 1728 |
| 4 | 312 | 264 | 576 |
| more | 322 | 254 | 576 |
| 2 | 576 | 0 | 576 |

---

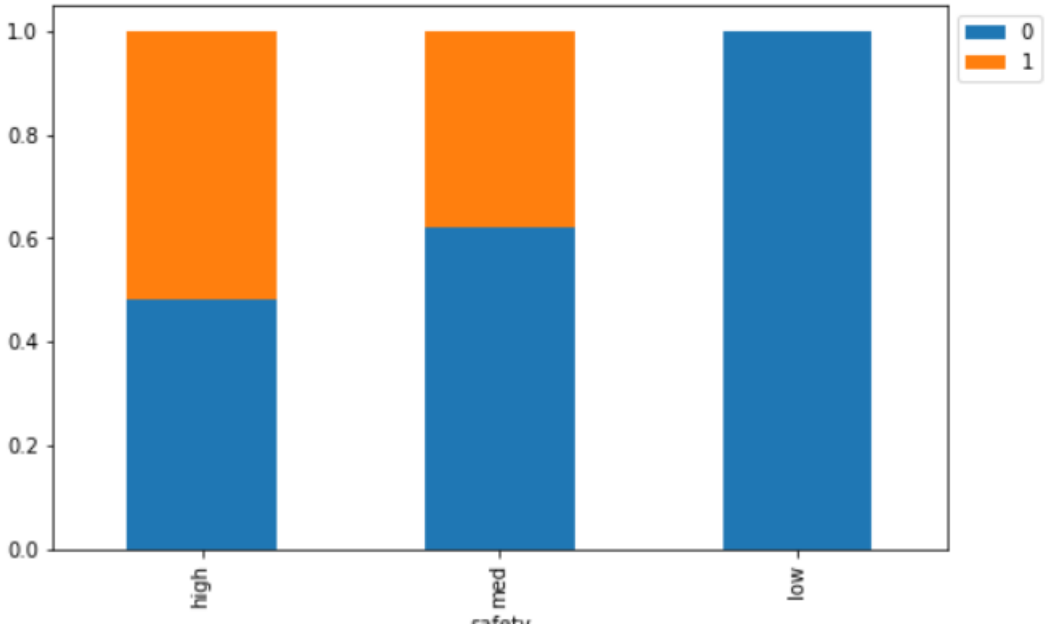


persons

| class | 0 | 1 | All |
|---|---|---|---|
| lug_boot | | | |
| All | 1210 | 518 | 1728 |
| big | 368 | 208 | 576 |
| med | 392 | 184 | 576 |
| small | 450 | 126 | 576 |




| class | 0 | 1 | All |
|---|---|---|---|
| safety | | | |
| All | 1210 | 518 | 1728 |
| high | 277 | 299 | 576 |
| med | 357 | 219 | 576 |
| low | 576 | 0 | 576 |




We can see that if the safety is low and number of persons are 2 there is more chance for the car to

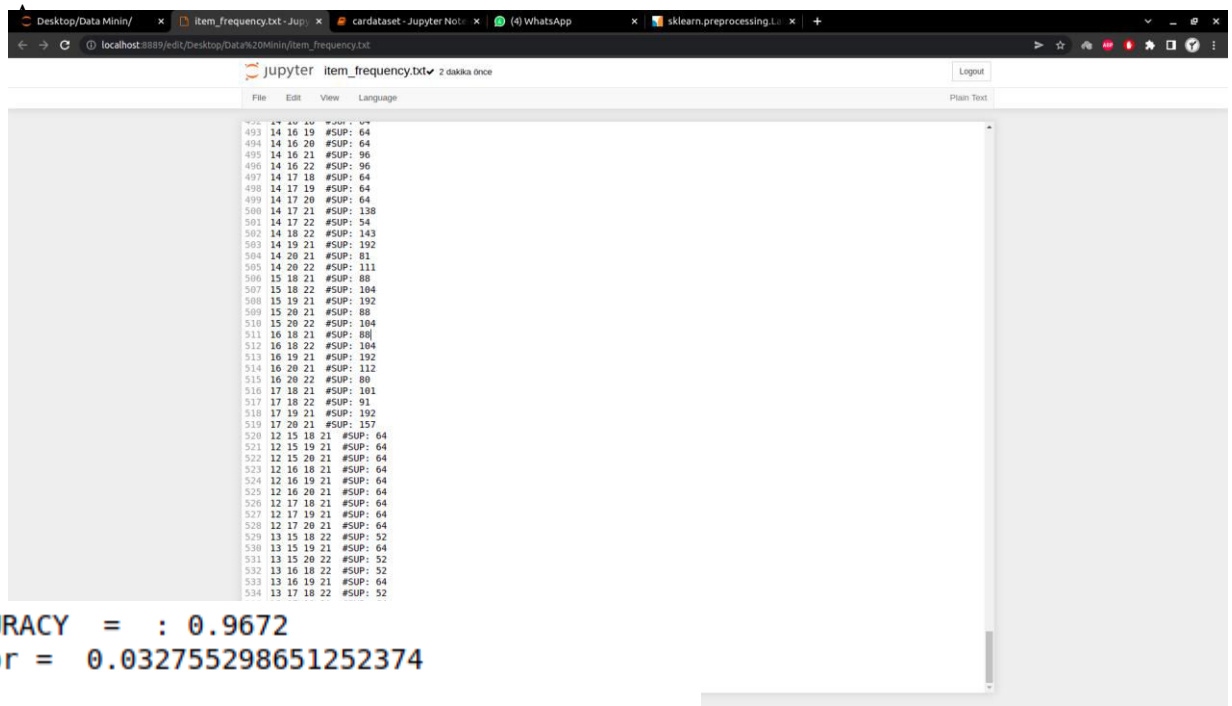
be unacceptable.

```
Before UnderSampling, counts of label '1': 375
Before UnderSampling, counts of label '0': 834

After UnderSampling, counts of label '1': 375
After UnderSampling, counts of label '0': 375

After UnderSampling, the shape of train_X: (750, 21)
After UnderSampling, the shape of train_y: (750,)
```

undersample the data to balance the classes

## 5. Modelling

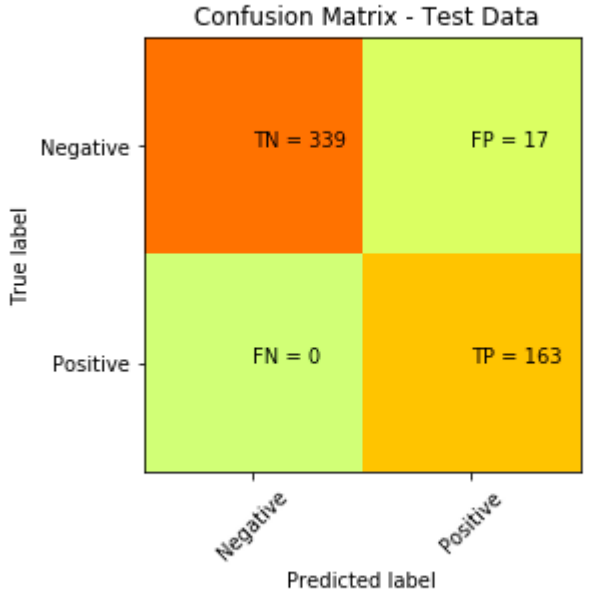
First and foremost, we used the Apriori method to locate frequent itemsets and to complete the categorization task. We utilized SPMF in developer mode to locate frequent itemsets using the Apriori technique. Each item and itemset, as well as their support count, are listed in the SPMF file for common itemsets. We encode our data firsly




look like photo. You can check full list in frequency.txt

# now we train random forest model on undersampled data

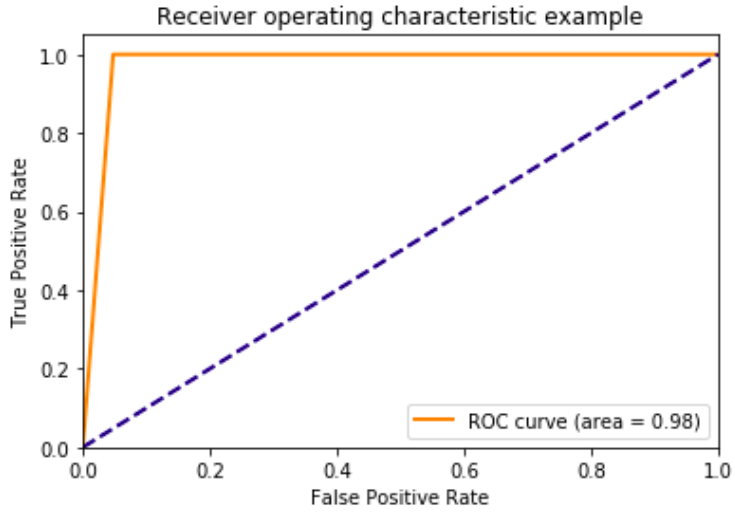
```
ACCURACY   =   : 0.9672
error =  0.032755298651252374
```




```
Sensivity = 0.952247191011236
Specificity = 1.0
```

Our conf matrix

```
Sensivity = 0.952247191011236
Specificity = 1.0
```




And other metrics

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 1.00 | 0.95 | 0.98 | 356 |
| 1 | 0.91 | 1.00 | 0.95 | 163 |
| accuracy | | | 0.97 | 519 |
| macro avg | 0.95 | 0.98 | 0.96 | 519 |
| weighted avg | 0.97 | 0.97 | 0.97 | 519 |