CLASSIFICATION OF BLOOD TRANSFUSION DATA

Abstract

Classification modelling of the data from the blood transfusion service center in Hsin-Chu City in Taiwan.

Subbiah Soundarapandian & Sudershan Ravi

s3825012 & s3829895

Conents

Student ID: s3825012 & s3829895
Student Name: Subbiah Soundarapandian & Sudershan Ravi

We certify that this is all our own original work. If we took any parts from elsewhere, then they were non-essential parts of the assignment, and they are clearly attributed in our submission. We will show we agree to this honor code by typing "Yes": *Yes*.

INTRODUCTION:

Blood Transfusion is a process of inducing blood into your body following an illness or injury. Healthy blood is composed of several essential components such as, red and white blood cells, plasma and platelets, among others. And blood transfusion would aid in supplementing one or more of those components that constitute healthy blood in the case of deficiencies. The duration of the process depends on the quantity of blood required and could vary between 1 to 4 hours. Blood Transfusion has been growing to be a relatively common medical procedure and is usually safe.

The data used for the purpose of this research is acquired from Blood Transfusion Service Centre in Hsin-Chu City in Taiwan, available in the UCI Machine Learning Repository. This is a classification problem to demonstrate the RFMTC (Recency, Frequency, Monetary Value, Time, Churn Rate) marketing model, an augmented RFM model. The dataset explores the donor database maintained at the Blood Transfusion Service Centre in Hsin-Chu City in Taiwan, comprising donor data of 748 volunteers selected at random. The data included R (Recency- months since last donation), F (Frequency- number of donations), M (Monetary- blood donated in c.c.), T (Time- months since first donation) and whether a person donated blood in March 2007, which takes binary inputs (1- donated blood, 0- have not donated blood).

DATA PREPARATION:

As the selected dataset is a classification problem, there is a requirement for python libraries such as pandas, numpy, seaborn and matplotlib among others, to perform the data modelling process. Hence, the aforementioned libraries are imported into the kernel. Following which, the csv (comma-separated values) dataset, is read into the console as a data frame and inspected for any obvious anomalies.

	Recency (months)	Frequency (times)	Monetary (c.c. blood)	Time (months)	whether he/she donated blood in March 2007
0	2	50	12500	98	1
1	0	13	3250	28	1
2	1	16	4000	35	1
3	2	20	5000	45	1
4	1	24	6000	77	0
5	4	4	1000	4	0
6	2	7	1750	14	1
7	1	12	3000	35	0
8	2	9	2250	22	1
9	5	46	11500	98	1

DATA EXPLORATION:

The basic information about the dataset is examined, in the intentions of gaining insight about the shape, values and data type of the features available in the dataset. On, examining the output it is learned that while all the features are of integer data type, none of them contain non-null values.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 748 entries, 0 to 747
Data columns (total 5 columns):
 #   Column                                            Non-Null Count  Dtype
---  ------                                            --------------  -----
 0   Recency (months)                                  748 non-null    int64
 1   Frequency (times)                                 748 non-null    int64
 2   Monetary (c.c. blood)                             748 non-null    int64
 3   Time (months)                                     748 non-null    int64
 4   whether he/she donated blood in March 2007        748 non-null    int64
dtypes: int64(5)
memory usage: 29.3 KB
```

Then, the descriptive statistics of the data frame are calculated to explore the count, mean, standard deviation and the percentile values. On inspecting the statistical data, it was evident that there were no negative values present in the data frame.

	Recency (months)	Frequency (times)	Monetary (c.c. blood)	Time (months)	whether he/she donated blood in March 2007
count	748.000000	748.000000	748.000000	748.000000	748.000000
mean	9.506684	5.514706	1378.676471	34.282086	0.237968
std	8.095396	5.839307	1459.826781	24.376714	0.426124
min	0.000000	1.000000	250.000000	2.000000	0.000000
25%	2.750000	2.000000	500.000000	16.000000	0.000000
50%	7.000000	4.000000	1000.000000	28.000000	0.000000
75%	14.000000	7.000000	1750.000000	50.000000	0.000000
max	74.000000	50.000000	12500.000000	98.000000	1.000000

The target feature of the data frame is the one containing binary values representing whether a person has donated blood in March 2007. The distribution of the values in this feature is examined by calculating the counts of values. The result makes it clear that most of the population have donated blood in the month of March.

```
0    570
1    178
Name: whether he/she donated blood in March 2007, dtype: int64
```

Before we proceed further to check for outliers, the statistical function is imported from the SciPy library.

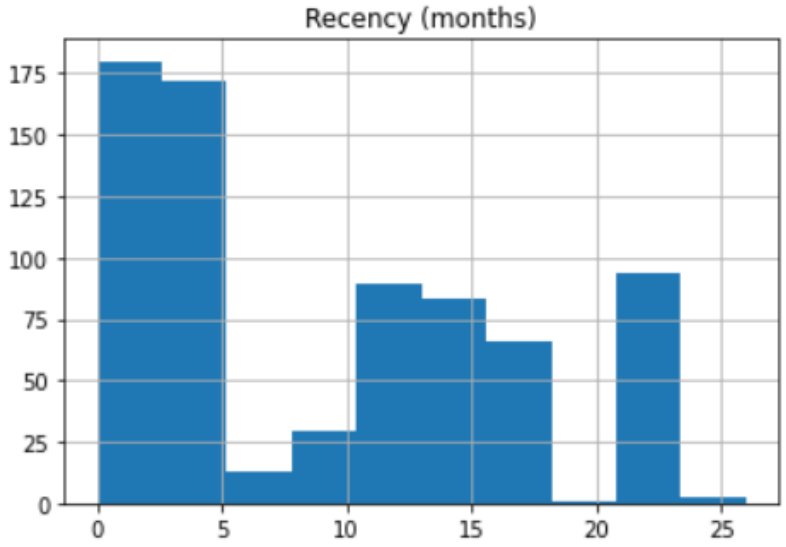
Further, the data frame needs to be checked for outliers as they affect the assumptions and hence, it is preferred to remove them. To achieve this, the z-scores of the values in the data frame are calculated and values with a z-score greater than 3, are dropped from the data frame. The resultant data frame contains 729 observations of 5 features.

VISUALIZATION:

To understand the features of the data better, each of the features is visualized. To begin with, the frequency of values in 'Recency (months)' is plotted against a histogram to determine the most frequent values in the features.

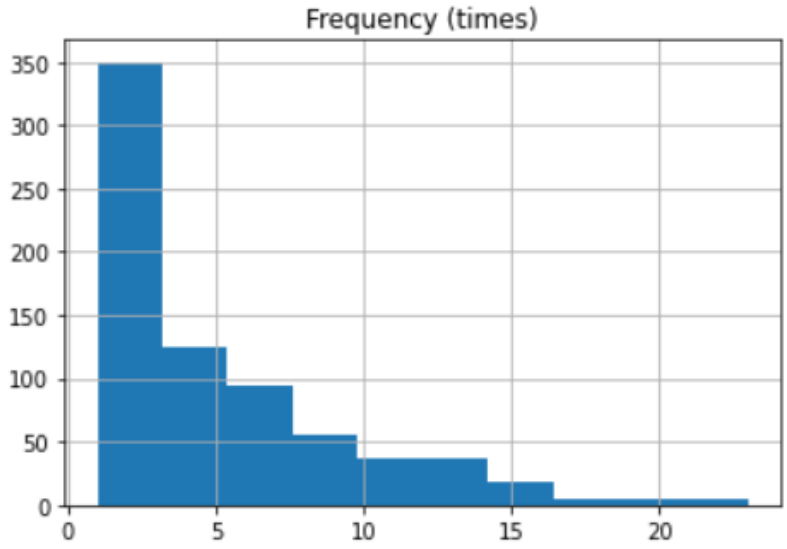
The output graph informs that a major chunk of the population has preferred to donate blood once in every 1-5 month(s) while many people have donated blood once in every 10-18 months and others every 21-23 months.

```
array([[<AxesSubplot:title={'center':'Recency (months)'}>]], dtype=objec
t)
```



The frequencies of blood donations by the donors are explored by plotting the values against a histogram.

```
array([[<AxesSubplot:title={'center':'Frequency (times)'}>]], dtype=obje
t)
```



The resultant graph makes it clear that most of the donors preferred to donate 1-3 time(s) while there is a declining trend as the frequency increases.

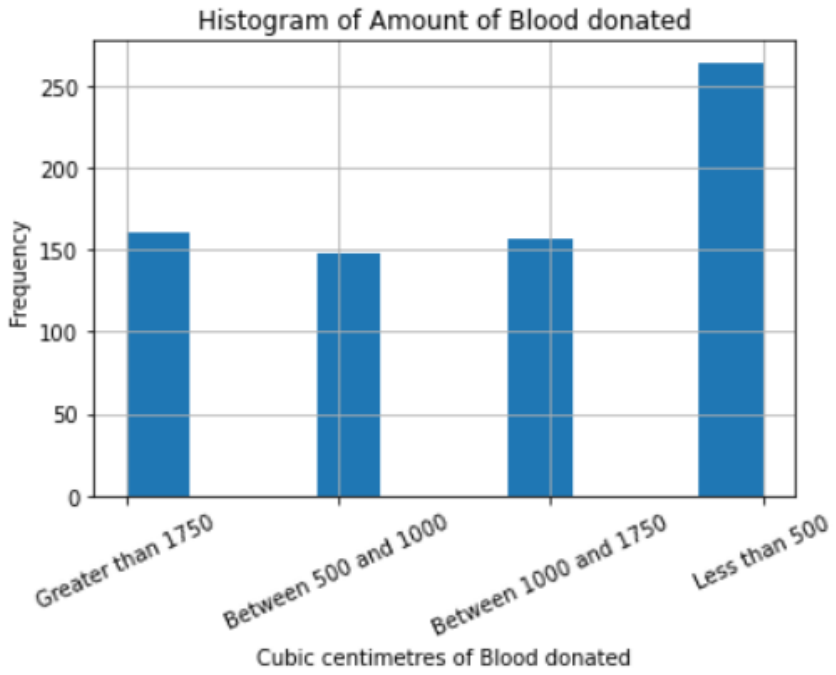
The monetary values are observed to be too high in number and hence, the values are categorized into 4 labels of equal quantiles of 500. The values are then plotted against a histogram to determine the frequency of values in each category.

```
Monetary = pd.qcut(assignds['Monetary (c.c. blood)'], q=4, labels = ["Less
Monetary.value_counts()
```

```
Less than 500            264
Greater than 1750        160
Between 1000 and 1750    157
Between 500 and 1000     148
Name: Monetary (c.c. blood), dtype: int64
```

```
Monetary.hist(xrot = 25)
plt.xlabel("Cubic centimetres of Blood donated")
plt.ylabel("Frequency")
plt.title("Histogram of Amount of Blood donated")
```

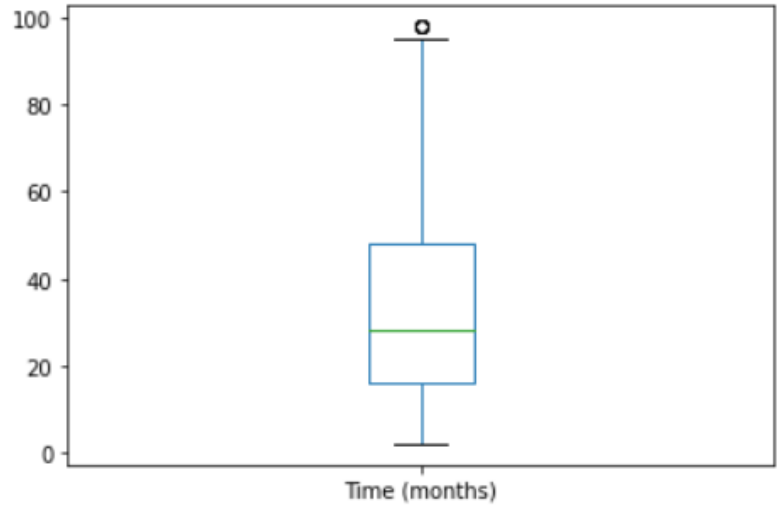
```
Text(0.5, 1.0, 'Histogram of Amount of Blood donated')
```



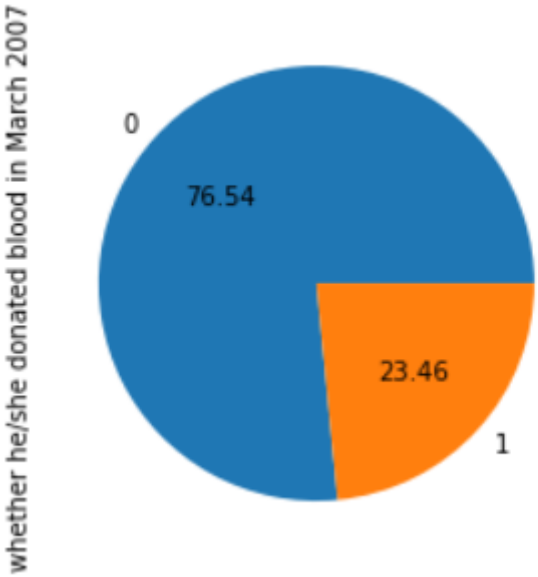
On inspecting the plot, it is evident that most of the donors preferred to donate less than 500 cubic centimetres of blood while the other 3 categories were of the same frequencies.

Then, the time since the first donation is visualized by plotting the values against a box plot to display the quartile statistics of the feature.

```
<AxesSubplot:>
```



And finally, the composition of both the variables in the target feature is examined by plotting a pie chart.



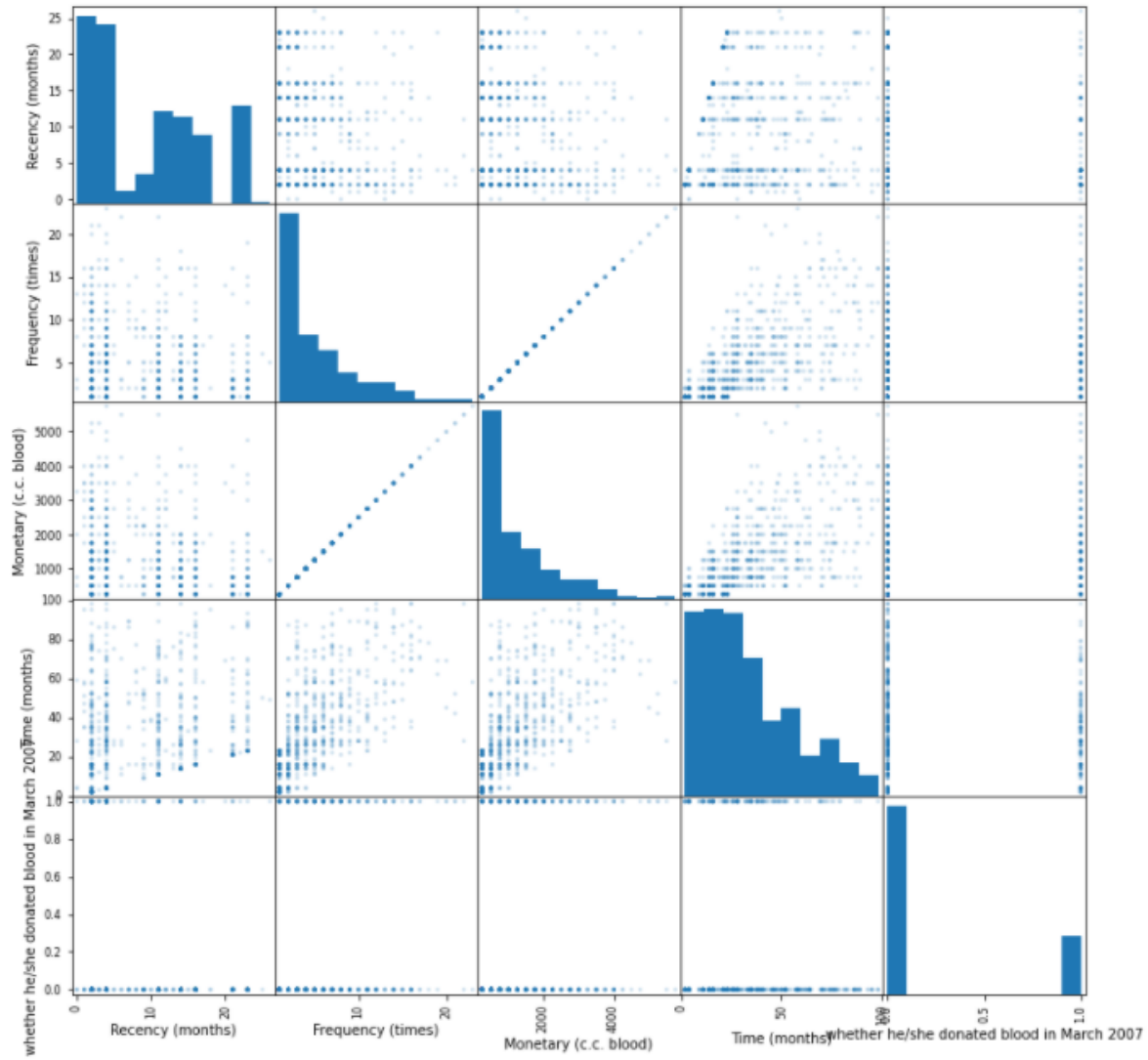
It is evident from the output plot that two-thirds of the population has not donated blood in March 2007.

INITIAL HYPOTHESES:

The initial hypotheses proposed upon exploring the data frame are:

- There is a positive correlation between the frequency and monetary values. As the number of donations by a donor increases, the monetary value of that donor increases.
- There is a positive correlation between the monetary values and time since the first donation by donor.
- There is no statistically significant correlation between other features in the data frame.

The correlation between the features in the data frame is explored by plotting a scatter matrix. The output provides a matrix of scatter plots depicting correlations between the features while the histograms of the values in the individual features.



It is evident from the scatter matrix that the there is a strongly linear positive correlation between frequency which proves our hypothesis true, while there is no strong linear correlation between monetary value and the time since the first donation which proves our hypothesis false.

Furthermore, on observing the scatter plots there are no other evident correlations between the features which again, proves our hypothesis true again.

DATA MODELLING:

Since the target variable of our dataset is a binary variable, we opt for classification. There are many classification algorithms under the 'sci-kit learn' (sklearn) library. The ones which we use in this report are K-Nearest Neighbours(k-NN) and Decision Tree Classifier.

TRAIN-TEST SPLIT:

For any model to be fit, it is always recommended to split the dataset into train and test set. Train set is used for model to learn and the test set is used to validate the predicted values against the observed values. While

splitting the dataset, we pass the parameter 'stratify' and set to target variable since the target variable unique values are not equally distributed.

Training set consist of 80% of total samples while test set consist of 20% of total samples. The resulting dimensions of train and test set are as follows,

```
The Shape of training descriptive data is  (583, 4)
The Shape of training target data is  (583,)
The Shape of test descriptive data is  (146, 4)
The Shape of test target data is  (146,)
```

FEATURE SCALING:

Feature Scaling is an important step in the process of model fitting. To get all the values in the fixed range, we do feature scaling. There are many scaling methods under the sklearn library such as Standard Scaler, Robust Scaler etc. The one which we use here is a MinMax Scaler which converts all the datapoints between 0 and 1.

Now let us look at the afore-mentioned classification models fit to our dataset and its results.

K-NEAREST NEIGHBORS CLASSIFIER:

K-NN Classifier is one of the famous supervised machine learning methods. It works based on the Minskowski (Euclidean and Manhattan) distances calculated between the new datapoint to be predicted and the already existing data points and decides upon to which group, the new datapoint suits the best and classifies accordingly.

HYPERTUNING:

In our analysis, we go with grid search[1] to hyper tune the model and find which model has the best accuracy and more reliable.

Grid search algorithm is one of the famous hyper-tuning algorithms. It iterates different K-NN models with the parameters that are passed to the 'param_grid' variable and gives us the model with optimal parameters to fit.

In this report, we pass the values of 'n_neighbors', 'p' and 'weights' to the parameter grid of the grid search algorithm. Also, we pass the Stratified k-Fold cross validation function as a method of validating the scores of the different models.

Once the parameters are set, the grid search algorithm is fit into the whole of the data and the target variable. The 'best_params_' attribute of the Grid Search Algorithm gives us the best model with respective 'n_neighbors', 'p' and 'weights' parameters.

FITTING K-NN MODEL:

```
gs_KNN.best_params_
```

```
{'n_neighbors': 9, 'p': 5, 'weights': 'uniform'}
```

As mentioned earlier, the 'best_params_' attribute fetches us the optimal parameters to fit the model with. Hence, we fit the model with the above parameters and get the classification metrics that helps us decide if the model is accurate and reliable to a good extent.

Below are the validations of the above model.

VALIDATION:

Let us look at the confusion matrix after validating with the test set which is the 20% of the total sample.

	y_pred_No	y_pred_Yes
y_test_No	109	3
y_test_Yes	26	8

The above pic shows that, our model has predicted negative response variable for 109 instances and falsely predicted the same for 3 instances, while predicted the positive response variable for 8 instances and falsely predicted for 26 instances. This is due to the inconsistency in the distribution of target instances (imbalanced dataset).

Now let us have a look at the classification metrics.

	precision	recall	f1-score	support
0	0.81	0.97	0.88	112
1	0.73	0.24	0.36	34
accuracy			0.80	146
macro avg	0.77	0.60	0.62	146
weighted avg	0.79	0.80	0.76	146

- Precision shows how precise our model is. In this case, our model has truly predicted the negative response variable for 81% of the time and positive response variable for 73% of the time.
- Recall score gives us the percentage of time the model has correctly identified the true positives and true negatives. In this case, our model has correctly identified the true negatives for 97% of the time but however only 24% of the time the model has correctly identified the true negatives.
- F1-Score gives us the harmonic mean of precision and recall score. Our model has the f1-score of 88% and 36% for negative and positive responses, respectively.
- Our model has good characteristics for Negative response instances and average to bad characteristics for positive response instance due to the imbalance in the dataset.
- As we can see, the support for negative response '0' is way more than that of the positive response.
- And finally, the accuracy of the 9 K-NN model turns out to be 80% which is a decent score for any classification model.

DECISION TREE CLASSIFIER:

Decision Tree Classifier is another type of supervised Machine Learning Classification algorithm. It follows a tree like structure and learns to split based on each of the attributes' values.

HYPERTUNING:

As dealt with K nearest neighbours algorithm, here also we go with Grid Search[1] algorithm. Here we use grid search among the parameters namely criterion, max_depth, and splitter.

Once we pass the parameters, we fit the model to the total data. As mentioned earlier, best_params_ attribute fetches us the decision tree model with higher accuracy among the lot.

FITTING THE BEST DECISION TREE CLASSIFIER:

```
gs_DT.best_params_
```

```
{'criterion': 'gini', 'max_depth': 4, 'splitter': 'best'}
```

The above picture shows the best parameters that can be used to fit the decision tree classifier. Decision tree with gini criterion with maximum depth of 4 branches and best splitter turns out to be a model with higher accuracy.

VALIDATION:

Let us have a look at the confusion matrix obtained after fitting the decision tree model.

	y_pred1_No	y_pred1_Yes
y_test_No	106	6
y_test_Yes	17	17

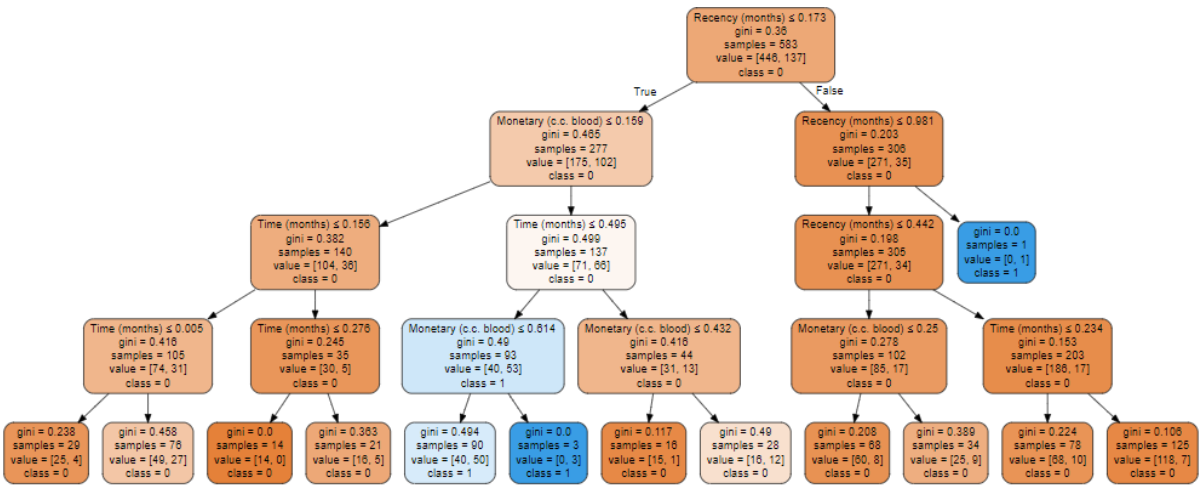
The above pic shows that the model has correctly predicted that the donor will not donate blood for 106 instances and falsely predicted that donor who donated, will not donate for 6 instances. It has predicted that donor will donate blood for 17 instances and will not donate blood for 17 instances.

Looking at the classification report of the fitted decision tree classifier model,

	precision	recall	f1-score	support
0	0.86	0.95	0.90	112
1	0.74	0.50	0.60	34
accuracy			0.84	146
macro avg	0.80	0.72	0.75	146
weighted avg	0.83	0.84	0.83	146

- Our decision tree model has truly predicted the negative response variable for 86% of the time and positive response variable for 74% of the time.
- Decision tree Classifier has correctly identified the true negatives for 95% of the time and 50% of the time the model has correctly identified the true negatives.
- F1-Score gives us the harmonic mean of precision and recall score. Our model has the f1-score of 90% and 60% for negative and positive responses, respectively.
- And finally, the accuracy of the Decision tree classifier model turns out to be 84% which is a better score than the K-NN model.

We can also visualize the tree with the help of an online visualization tool 'GraphViz'.



MODEL COMPARISON:

As a result of this report, we can compare two models and decide on which model is more accurate and reliable in predicting whether the donor will donate blood or not. It is more evident from the classification reports of the two models that the decision tree classifier performs much better than the K-Nearest Neighbour

model. Decision tree model has good metrics compared to K-NN model which struggled due to imbalanced dataset. Decision Tree model tackled the data imbalance to a good extent. Decision Tree model's accuracy is more than the K-NN model and hence we can rely on this model.

CONCLUSION:

To conclude, the decision tree classifier provides the most accurate model with an accuracy of 84%.

REFERENCES:

[1] Okamura, S. (2020). GridSearchCV for Beginners. Retrieved 22 May 2021, from https://towardsdatascience.com/gridsearchcv-for-beginners-db48a90114ee