



**VIT<sup>®</sup>**

**Vellore Institute of Technology**

(Deemed to be University under section 3 of UGC Act, 1956)

**SCHOOL OF COMPUTER ENGINEERING**

**Winter Semester -2021**

**PROJECT REPORT**

**CSE3020 – DATA VISUALIZATION**

BLOOD DONATION PREDICTION	
TEAM NO: 2	
18BCB0145	SIDDHARTHA MONDAL
18BCE2067	PRADYOT SHARMA

Submitted to

**Dr. Archana Tamizharasan**

**SCOPE**

## Project at a glance

<b>No of objectives considered: 3</b> <ol style="list-style-type: none"> <li>1. Exploratory Data-analysis</li> <li>2. Feature Engineering</li> <li>3. Feature Selection</li> </ol>	
<b>Language used: Python</b>	
<b>Statistical Measures used:</b> <ol style="list-style-type: none"> <li>1. Average</li> <li>2. Frequency</li> <li>3. Correlation</li> </ol>	
<b>Library(s) used:</b> <ol style="list-style-type: none"> <li>1. Pandas</li> <li>2. Numpy</li> <li>3. Sklearn (many sublibraries of sklearn have been used)</li> <li>4. Matplotlib</li> <li>5. Seaborn</li> </ol>	
<b>Total no of Visuals created:</b>	<b>11</b>
<b>No of Individual Chart types used:</b>	<b>5</b>
<b>Bar chart</b>	<b>1</b>
<b>Scatter plot</b>	<b>2(in pairplot and regplot)</b>
<b>Line graph</b>	<b>2(in pairplot and regplot)</b>
<b>Histogram</b>	<b>6</b>
<b>Heatmap</b>	<b>2</b>
<b>Time series</b>	<b>8 (Almost all graphs are time dependant)</b>
<b>Total Charts in project</b>	<b>11</b>

## **1.1 Project Statement**

“Predict future blood donations”.

We want to predict whether or not a donor will give blood the next time the opportunity is presented.

## **1.2 Project Objective**

Blood donation has been around for a long time. The first successful recorded transfusion was between two dogs in 1665, and the first medical use of human blood in a transfusion occurred in 1818. Even today, donated blood remains a critical resource during emergencies.

Our dataset is from a mobile blood donation vehicle in Taiwan. The Blood Transfusion Service Center drives to different universities and collects blood as part of a blood drive.

## **1.3 Modules**

### **1.3.1 Import libraries**

All the python libraries required for this project are imported. Some libraries are added in course of the codes.

### **1.3.2 Splitting into training and cross validation dataset**

Like any other machine learning or deep learning project the dataset is split here.

### **1.3.3 Exploratory Data Analysis**

We first check for any missing values in the training and testing data and any necessary edit to the dataframe is done here so that we can proceed with the different statistical analysis, visualization and modelling properly. We also

conduct the plot distribution of the different data in our training set for better understanding and utilization during feature engineering.

#### **1.3.4 Feature Engineering and Feature Selection**

Here we take care of the correlated features and at the same time we add additional features (or columns to the training data) so that our analysis is more accurate prediction, i.e., to find any other good predictor that the already existing features.

#### **1.3.5 Model Fitting**

In this module we conduct feature normalization and then fit the models we chose for this project - Logistic Regression, Random Forest, Support Vector Machine and Gradient Boosting (AdaBoost). Along with this we use an evaluation metric to find out which model fits the best for the given dataset and that is chosen.

#### **1.3.6 Prediction**

This is the final module of our project where we use the fit model to combine the training and test set and estimate the Logistic Regression Model using the hyperparameters found previously and the final predictions are saved in a csv file.

## 1.4 Code with Visuals

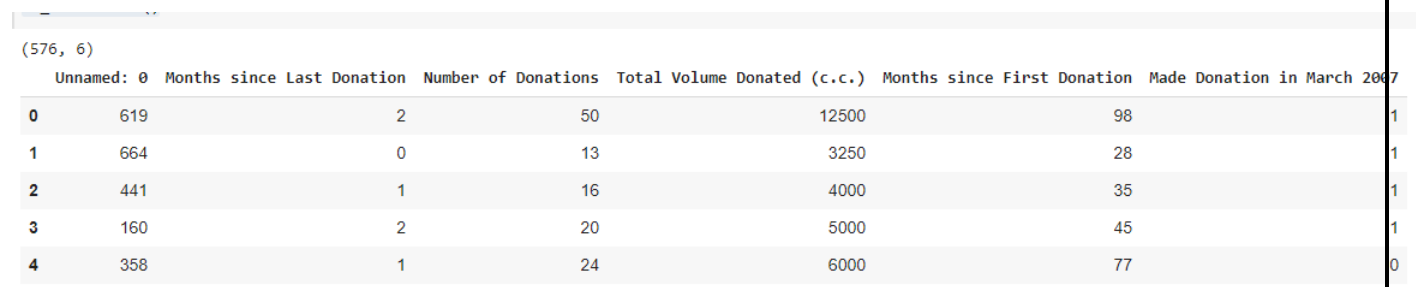
### Importing key libraries and reading dataframes

```
import pandas as pd
import numpy as np
```

```
df_train = pd.read_csv('/content/Warm_Up_Predict_Blood_Donations_-_Traning_Data.csv')
df_test = pd.read_csv('/content/Warm_Up_Predict_Blood_Donations_-_Test_Data.csv')
```

Splitting into training and cross validation dataset

```
print(df_train.shape)
df_train.head()
```



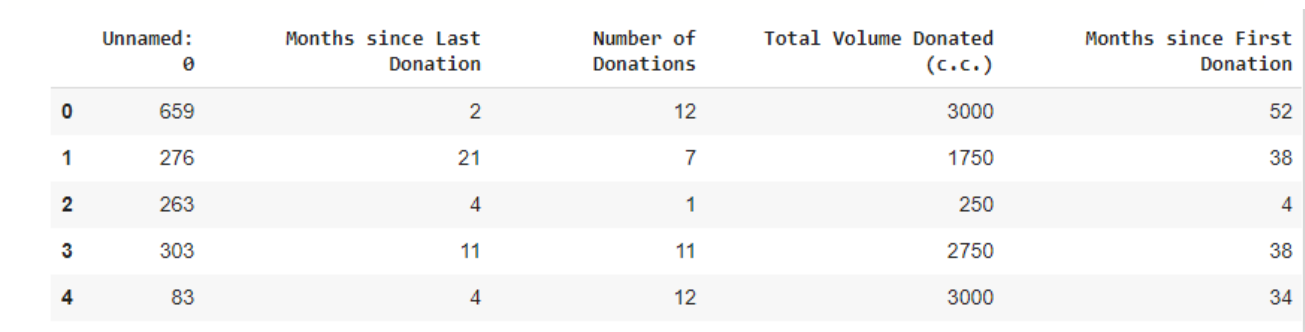
	Unnamed: 0	Months since Last Donation	Number of Donations	Total Volume Donated (c.c.)	Months since First Donation	Made Donation in March 2007
0	619	2	50	12500	98	1
1	664	0	13	3250	28	1
2	441	1	16	4000	35	1
3	160	2	20	5000	45	1
4	358	1	24	6000	77	0

Fig1.4.1

```
import pandas as pd
from sklearn.model_selection import train_test_split
```

```
X = df_train.iloc[:, 1:5]
y = df_train[["Made Donation in March 2007"]]
```

```
X_train, X_cv, y_train, y_cv = train_test_split(X, y, test_size = 0.2, random_state = 0)
df_test.head()
```



	Unnamed: 0	Months since Last Donation	Number of Donations	Total Volume Donated (c.c.)	Months since First Donation	Made Donation in March 2007
0	659	2	12	3000	52	1
1	276	21	7	1750	38	1
2	263	4	1	250	4	1
3	303	11	11	2750	38	1
4	83	4	12	3000	34	1

Fig 1.4.2

```
X_test = df_test.iloc[:, 1:5]
```

```
X_train = X_train.reset_index(drop = True)
```

```
X_cv = X_cv.reset_index(drop = True)
```

```
X_test = X_test.reset_index(drop = True)
```

## Exploratory Data Analysis

We first check for any missing values in the training and testing data.

```
X_train.isnull().sum()
```

```
Months since Last Donation    0
Number of Donations           0
Total Volume Donated (c.c.)   0
Months since First Donation   0
dtype: int64
```

Fig 1.4.3

```
X_cv.isnull().sum()
```

```
Months since Last Donation    0
Number of Donations           0
Total Volume Donated (c.c.)   0
Months since First Donation   0
dtype: int64
```

Fig 1.4.4

```
X_test.isnull().sum()
```

```
Months since Last Donation    0
Number of Donations           0
Total Volume Donated (c.c.)   0
Months since First Donation   0
dtype: int64
```

Fig 1.4.5

We now proceed to plot the distribution of the 4 different columns in the training data to get an idea of the data types that we are dealing with, as this can potentially help with our feature engineering.

In this case, the color, Red, represents Non-blood Donors, while the color, Blue, represents Donors.

```
import matplotlib.pyplot as plt
```

```
import seaborn as sns
```

```
plt.figure(figsize = (20, 10))
```

```
plt.subplot(2, 2, 1)
sns.distplot(X_train[y_train.values == 0]['Months since Last Donation'],
             bins = range(0, 81, 2), color = 'red')
plt.ylabel('Frequency')
plt.title('Distribution of Months Since last Donation for Non-blood Donors')

sns.distplot(X_train[y_train.values == 1]['Months since Last Donation'],
             bins = range(0, 81, 2), color = 'blue')
plt.ylabel('Frequency')
plt.title('Distribution of Months Since last Donation for Blood Donors')

plt.subplot(2, 2, 2)
sns.distplot(X_train[y_train.values == 0]['Number of Donations'],
             bins = range(0, 60, 2), color = 'red')
plt.ylabel('Frequency')
plt.title('Distribution of Number of Donations for Non-blood Donors')

sns.distplot(X_train[y_train.values == 1]['Number of Donations'],
             bins = range(0, 60, 2), color = 'blue')
plt.ylabel('Frequency')
plt.title('Distribution of Number of Donations for Blood Donors')

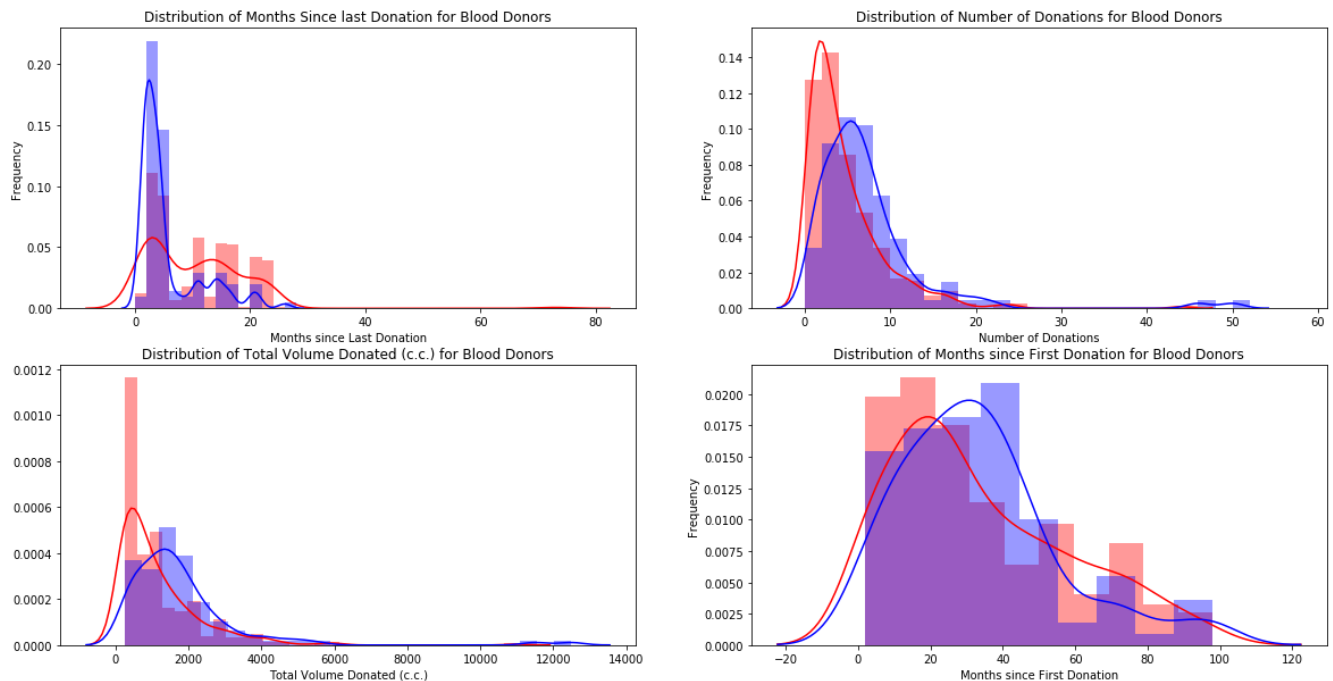
plt.subplot(2, 2, 3)
sns.distplot(X_train[y_train.values == 0]['Total Volume Donated (c.c.)'], color = 'red')
plt.ylabel('Frequency')
plt.title('Distribution of Total Volume Donated (c.c.) for Non-blood Donors')

sns.distplot(X_train[y_train.values == 1]['Total Volume Donated (c.c.)'], color = 'blue')
plt.ylabel('Frequency')
plt.title('Distribution of Total Volume Donated (c.c.) for Blood Donors')

plt.subplot(2, 2, 4)
sns.distplot(X_train[y_train.values == 0]['Months since First Donation'], color = 'red')
plt.ylabel('Frequency')
plt.title('Distribution of Months since First Donation for Non-blood Donors')

sns.distplot(X_train[y_train.values == 1]['Months since First Donation'], color = 'blue')
plt.ylabel('Frequency')
plt.title('Distribution of Months since First Donation for Blood Donors')

plt.show()
```



**Fig 1.4.6**

Intuitively, blood donors who have frequently donated blood in the past are more likely to donate blood. In addition, donors who have donated within 3-6 months are much more likely to donate blood than their counterparts i.e. donors who have not donated in the past 6 months.

On the other hand, the 'Months since First Donation' feature does not seem to be particularly informative of whether a blood donor donated blood in March 2007.

In addition, we note the presence of several outliers in the dataset.

We proceed to examine the relationship across the 4 features in the training dataset, by means of a pairsplot.

```
sns.pairplot(X_train, diag_kind='kde')
plt.show()
```



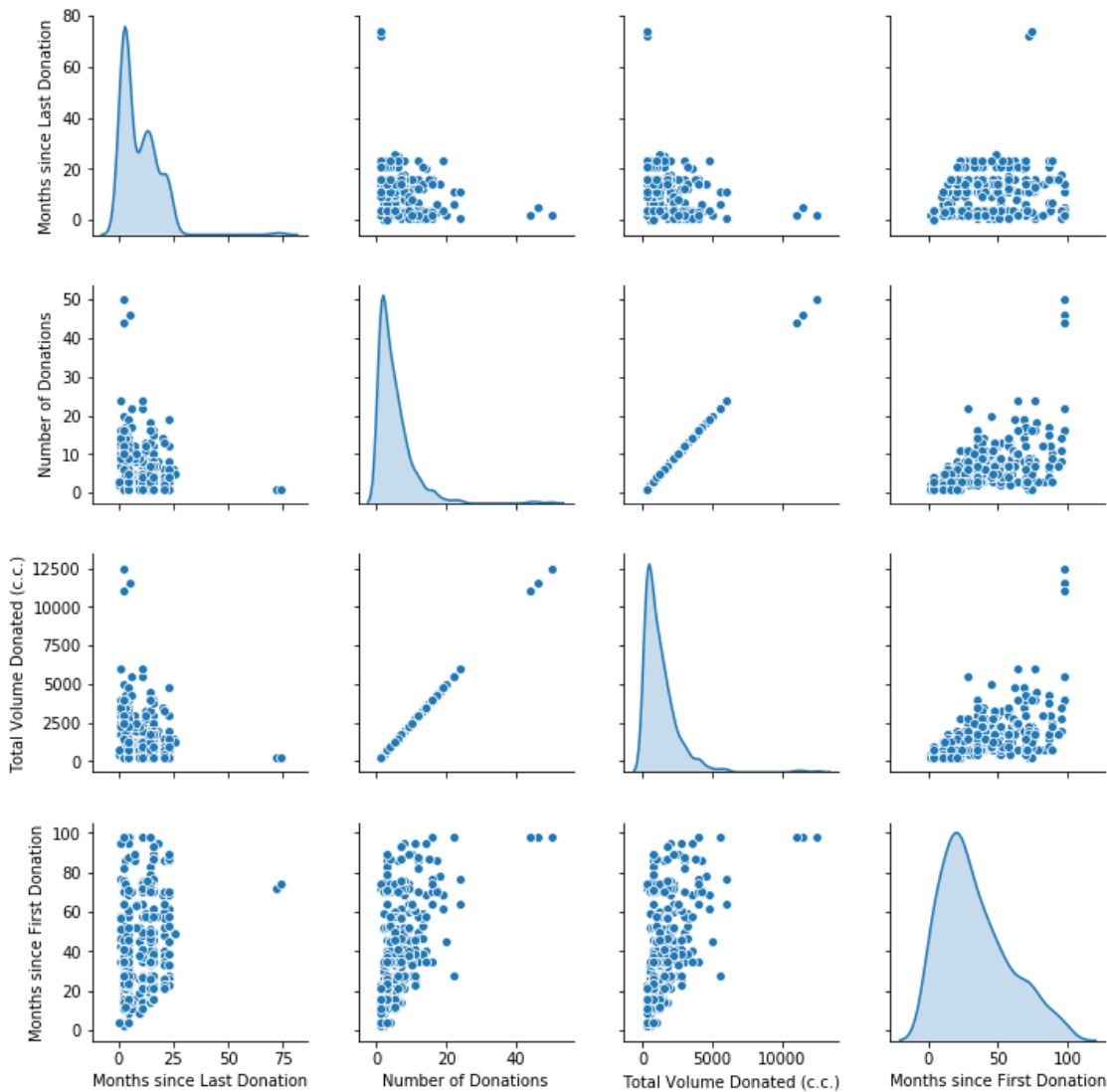


Fig 1.4.7

Both the feature distribution plots and the pairsplot seem to indicate that there are a few anomalies in the dataset. In addition, the pairsplot seem to suggest a linear relationship between the 2 variables, Total Number of Donations and Total Volume Donated.

We check this phenomenon using the correlation between these features, by means of a heatmap.

```
plt.figure(figsize = (20, 10))
X_train_corr = X_train.corr()

sns.heatmap(X_train_corr, annot = True)
plt.show()
```

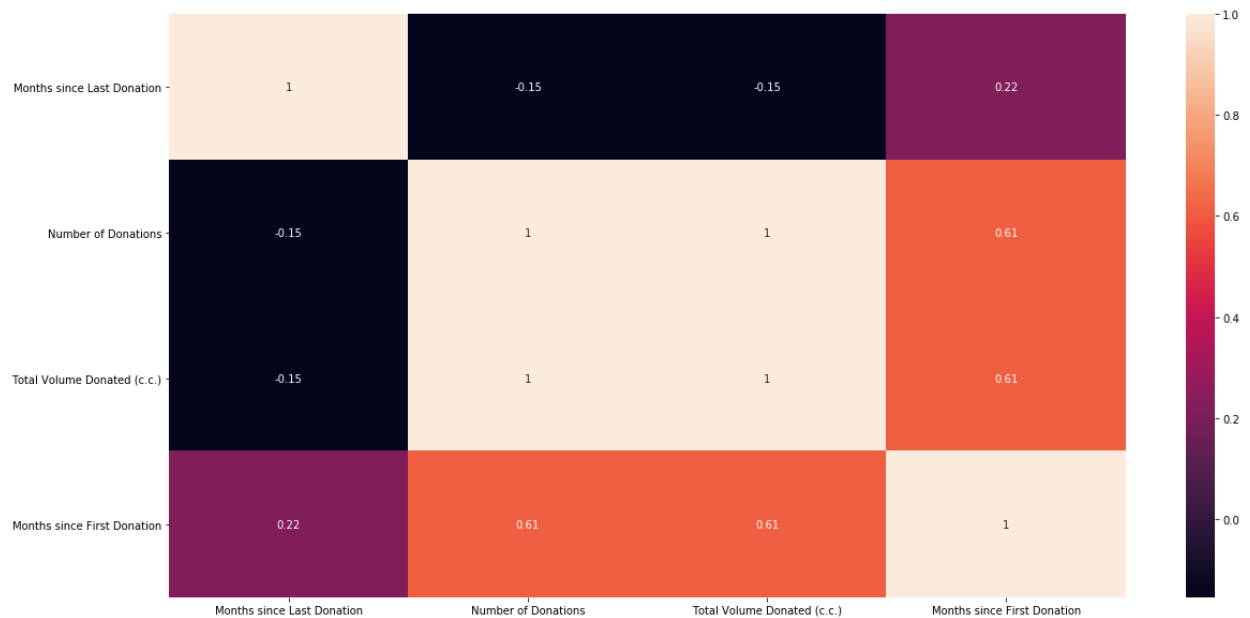


Fig 1.4.8

This confirms our suspicion that the features, Number of Donations and Total Volume Donated, are indeed perfectly correlated.

In addition, we note that there exists a weak positive correlation between Number of Donations (same for Total Volume Donated) and Months since First Donation. This is likely due to the presence of anomalies present in our training dataset.

## Feature Engineering and Feature Selection

As some of our features are perfectly correlated with one another, including all these features do not provide any additional information as compared to only including 1 of these features. This means that we should remove one of the features which are perfectly correlated with the other feature.

But before we do so, we could generate additional features using existing ones. For example, one of the many features we might be interested in is the average donation of each donor.

```
X_train['Average Donation per Month'] = (X_train['Total Volume Donated (c.c.)']/
                                           X_train['Months since First Donation'])
```

```
X_train.head()
```

	Months since Last Donation	Number of Donations	Total Volume Donated (c.c.)	Months since First Donation	One-time Donor	Average Donation per Month
0	4	4	1000	18	0	55.555556
1	2	1	250	2	1	125.000000
2	20	14	3500	69	0	50.724638
3	2	2	500	11	0	45.454545
4	11	4	1000	58	0	17.241379

Fig 1.4.9

We do the same for the cross-validation dataset and the testing dataset.

```
X_cv['Average Donation per Month'] = X_cv['Total Volume Donated (c.c.)']/X_cv['Months since First Donation']
```

```
X_test['Average Donation per Month'] = X_test['Total Volume Donated (c.c.)']/X_test['Months since First Donation']
```

```
plt.figure(figsize = (20, 10))
```

```
sns.distplot(X_train[y_train.values == 0]['Average Donation per Month'], color = 'red')
plt.ylabel('Frequency')
plt.title('Distribution of Months Since last Donation for Non-blood Donors')
```

```
sns.distplot(X_train[y_train.values == 1]['Average Donation per Month'], color = 'blue')
plt.ylabel('Frequency')
plt.title('Distribution of Months Since last Donation for Blood Donors')
```

```
plt.show()
```

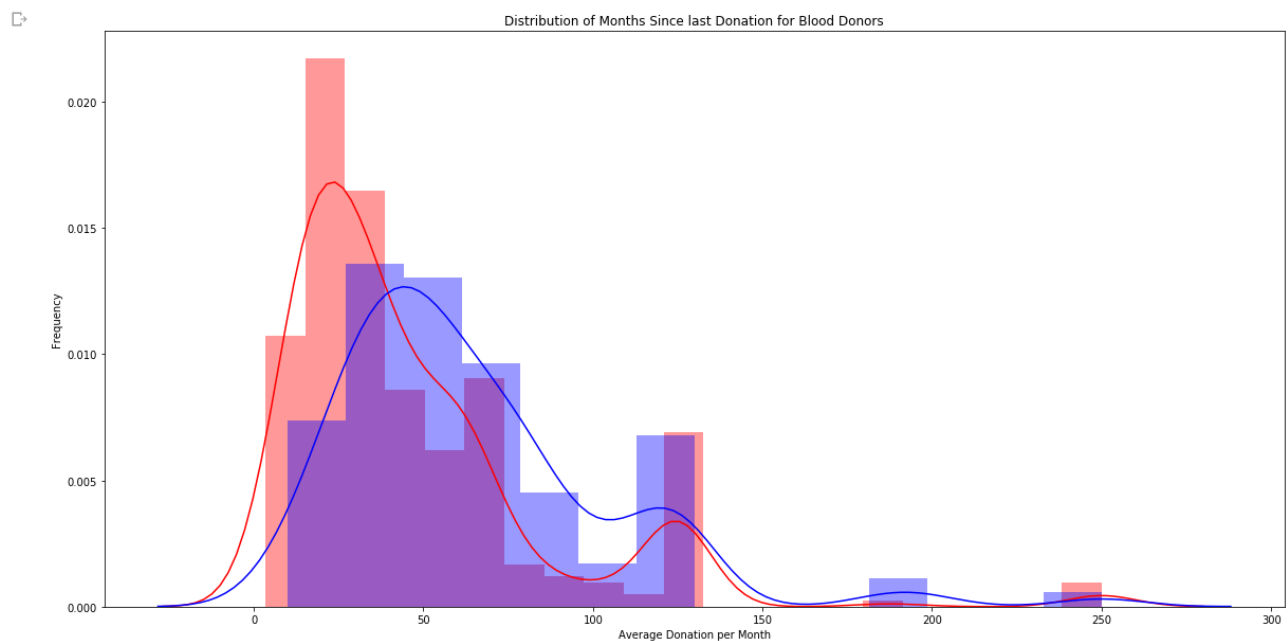


Fig 1.4.10

We include a new variable, average waiting length for donation, to observe the frequency which the donor donates blood.

```
X_train['Waiting Time'] = ((X_train['Months since First Donation'] - X_train['Months since Last Donation'])
                           /X_train['Number of Donations'])
```

```
X_cv['Waiting Time'] = ((X_cv['Months since First Donation'] - X_cv['Months since Last Donation'])
                       /X_cv['Number of Donations'])
```

```
X_test['Waiting Time'] = ((X_test['Months since First Donation'] - X_test['Months since Last Donation'])
                        /X_test['Number of Donations'])
```

```
plt.figure(figsize = (20, 10))
```

```
sns.distplot(X_train[y_train.values == 0]['Waiting Time'], color = 'red')
plt.ylabel('Frequency')
plt.title('Distribution of Waiting Time for Non-blood Donors')
```

```
sns.distplot(X_train[y_train.values == 1]['Waiting Time'], color = 'blue')
plt.ylabel('Frequency')
plt.title('Distribution of Waiting Time for Blood Donors')
```

```
plt.show()
```

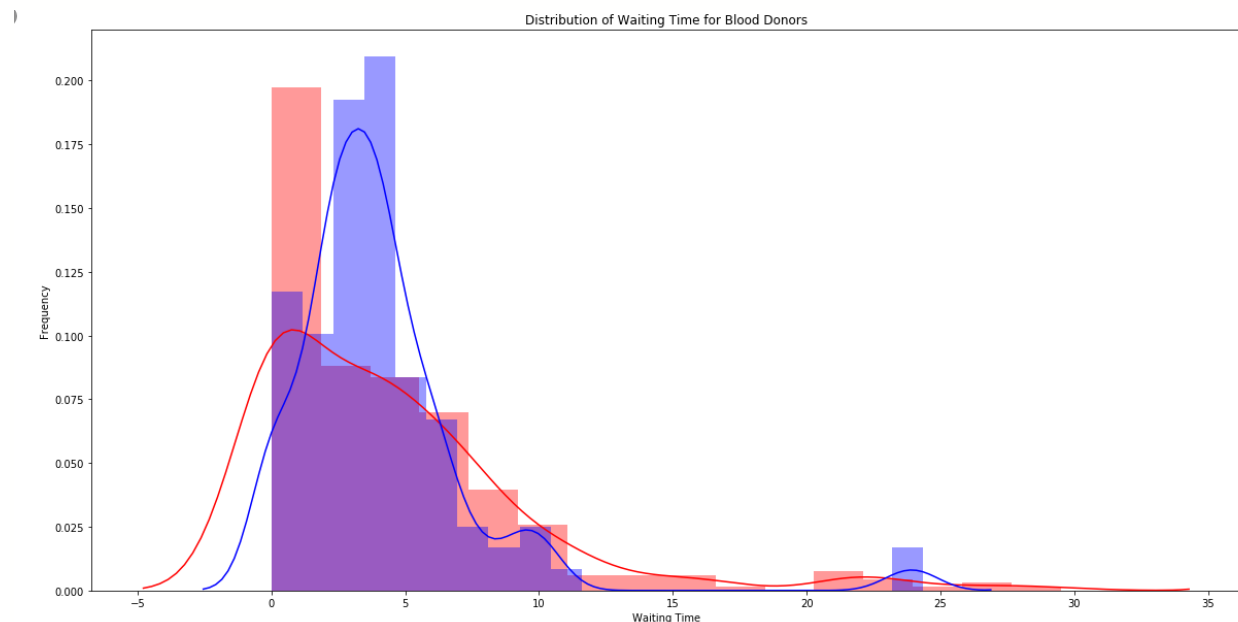


Fig 1.4.11

```
#now checking for one time donors
```

```
X_train['One-time Donor'] = X_train['Number of Donations'] == 1
```

```
tab = pd.crosstab(X_train['One-time Donor'], y_train['Made Donation in March 2007'])
```

```
tab.div(tab.sum(1).astype(float), axis=0)
```

Made Donation in March 2007		0	1
One-time Donor			
False		0.735593	0.264407
True		0.753247	0.246753

Fig 1.4.12

```
plt.figure(figsize = (20, 10))
```

```
sns.regplot(x = y_train.reset_index(drop = True)[X_train['One-time Donor'] == 1],  
            y = X_train[X_train['One-time Donor'] == 1]['Months since First Donation'])
```

```
plt.show()
```

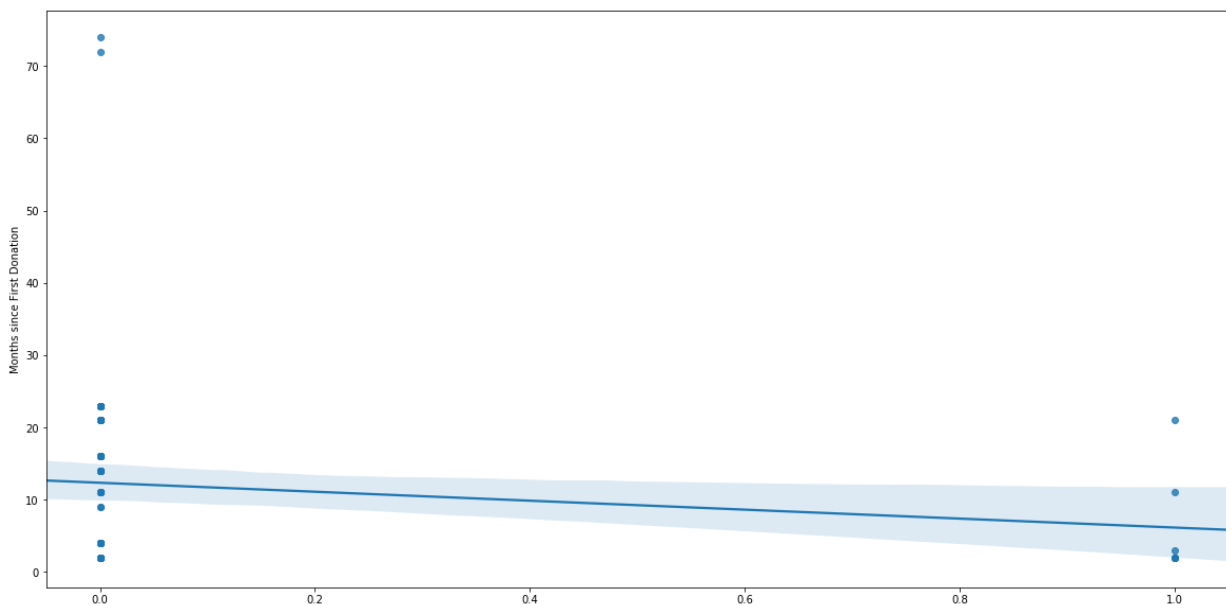


Fig 1.4.13

```
del X_train['One-time Donor']
```

From the plots above, we figured that donor who donated 3-6 months prior and donors who have donated more than 5 times ever since their first donation are much more likely to donate blood than their peers. Let's create new features based on this set of information.

In addition, it is important to note that donors who donated 3 months prior are not much likely to donate blood than their counterparts. As a result, the feature should not include these donors.

```
X_train['Donated in the past 3-6 months'] = ((X_train['Months since Last Donation'] >= 3) &
                                              (X_train['Months since Last Donation'] <= 6))
```

```
X_cv['Donated in the past 3-6 months'] = ((X_cv['Months since Last Donation'] >= 3) &
                                           (X_cv['Months since Last Donation'] <= 6))
```

```
X_test['Donated in the past 3-6 months'] = ((X_test['Months since Last Donation'] >= 3) &
                                             (X_test['Months since Last Donation'] <= 6))
```

```
X_train['Frequent Donor'] = (X_train['Number of Donations'] >= 5)
```

```
X_cv['Frequent Donor'] = (X_cv['Number of Donations'] >= 5)
```

```
X_test['Frequent Donor'] = (X_test['Number of Donations'] >= 5)
```

Next, we proceed to remove the Total Volume Donated (c.c.) feature from the 3 datasets.

```
cols_to_keep = ['Months since Last Donation', 'Number of Donations',
                'Months since First Donation', 'Average Donation per Month',
                'Waiting Time', 'Donated in the past 3-6 months', 'Frequent Donor']
X_train = X_train[cols_to_keep]; X_cv = X_cv[cols_to_keep]; X_test = X_test[cols_to_keep]
```

```
X_train.head()
```

	Months since Last Donation	Number of Donations	Months since First Donation	Average Donation per Month	Waiting Time	Donated in the past 3-6 months	Frequent Donor
0	4	4	18	55.555556	3.50	True	False
1	2	1	2	125.000000	0.00	False	False
2	20	14	69	50.724638	3.50	False	True
3	2	2	11	45.454545	4.50	False	False
4	11	4	58	17.241379	11.75	False	False

Fig 1.4.14

```
plt.figure(figsize = (20, 10))
train_ = X_train.copy(); train_['y'] = y_train.reset_index(drop = True)
train_corr = train_.corr()

sns.heatmap(train_corr, annot = True)
plt.show()
```

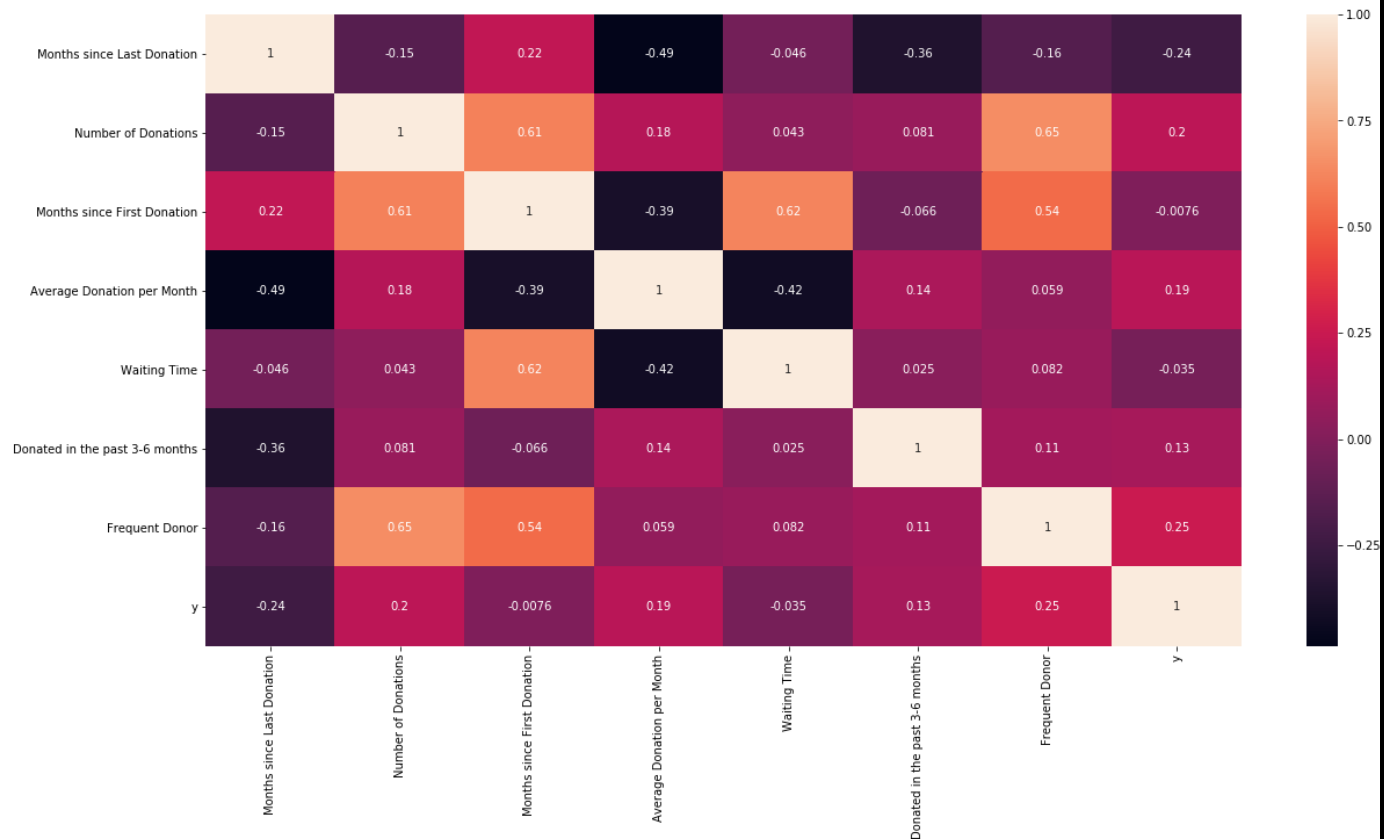
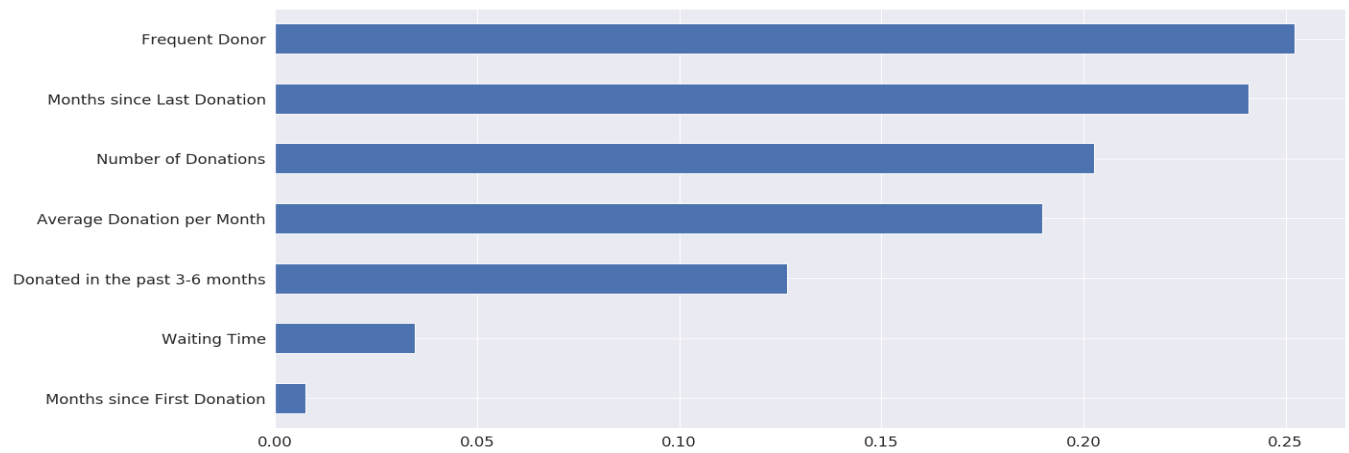


Fig 1.4.15

```
plt.figure(figsize = (20, 10))
sns.set(font_scale = 1.5)
(abs(train_corr)
 .y
 .drop('y')
 .sort_values()
 .plot
 .barh())
plt.show()
```



**Fig 1.4.16**



## **Conclusion**

Using this project, we could actually predict the blood donation probability with respect to the given dataset. Considering that a simple dataset is received for any particular area of a particular time and scenario, this program could actually help in cases of an emergency. With the prediction values, people can be motivated to donate blood and hence increasing the odds of losing a life due to unavailability of blood in situations of war, natural disaster or pandemic.

Further work on this project could lead to finding of a good Machine learning or even a deep learning model with high accuracy for prediction of donors' donation probability. Along with this we would like to add that this project helped us learn different aspects of data visualization from python perspective and hence helped us explore into pandas, numpy, matplotlib, etc.

## List of figures

S. no.	Title of visual	Page no.
1.	<b>Histogram (Histplot)</b> Each of the histplots here are the plots of frequency of donation w.r.t features in the training dataset.	8
2.	<b>Histogram (Histplot)</b> Frequency of donation w.r.t average donation per month	11
3.	<b>Histogram (Histplot)</b> Frequency of donation w.r.t waiting time for donation	12
4.	<b>Pairplot</b> To examine the relationship between the different features (columns) of the training set.	9
5.	<b>Heatmap</b> Checking correlation coefficient values between the features of the training dataset	10
6.	<b>Heatmap</b> Checking correlation coefficient values between the features of the training dataset along with the additional features from feature engineering.	15
7.	<b>Regplot</b> To check the significance of the one-time-donors values from the dataset and whether it can be used as a feature.	13
8.	<b>Barchart</b> The final features that shall be used after the feature engineering is completed and feature selection is performed.	16

