KLE Society's
KLE Technological University

A DMA Project Report

On

Warm Up : Predict Blood Donation

Bachelor of Engineering in

Computer Science and Engineering

Submitted By

Vineeth Kumar	01FE16BCS250
Rohith Vaidyanathan	01FE16BCS167
Sammed Chougale	01FE16BCS176

Under the guidance of
Prof. Sunitha Hiremath

SCHOOL OF COMPUTER SCIENCE & ENGINEERING
HUBLI – 580 031 (India).

Table of Content

Introduction

In the United States, the American Red Cross is a good resource for information about donating blood. According to their website:

- Every two seconds someone in the U.S. needs blood.
- More than 41,000 blood donations are needed every day.
- A total of 30 million blood components are transfused each year in the U.S.
- The blood used in an emergency is already on the shelves before the event occurs.
- Sickle cell disease affects more than 70,000 people in the U.S. About 1,000 babies are born with the disease each year. Sickle cell patients can require frequent blood transfusions throughout their lives.
- More than 1.6 million people were diagnosed with cancer last year. Many of them will need blood, sometimes daily, during their chemotherapy treatment.
- A single car accident victim can require as many as 100 pints of blood.

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. The 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.

Problem Statement

To predict whether or not a donor will give blood the next time the vehicle comes to campus based on the Data collected from previous Blood donation Vehicle trips to Educational establishments.

Figure 1.0:

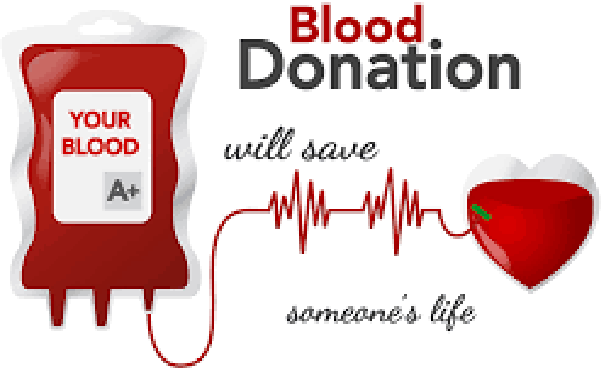


Figure 1.1:

Problem description

Our goal is to predict the binary class, Made Donation in March 2007, which represents whether the donor will donate the blood or not :

- 0 represents the donor will not donate,
- 1 represents the donor will donate.

Data Description

The dataset is from a study of heart disease that has been open to the public for many years. The study collects various measurements on patient health and cardiovascular statistics, and of course makes patient identities anonymous.

Data Set Characteristics: Multivariate
Number of Instances: 576
Area: Business
Attribute Characteristics: Real
Number of Attributes: 5
Associated Tasks: Classification
Missing Values? N/A

Attribute information:

Given is the variable name, variable type, the measurement unit and a brief description. The "Blood Transfusion Service Center" is a classification problem.
The order of this listing corresponds to the order of numerals along the rows of the database.

R (Recency - months since last donation),
F (Frequency - total number of donation),
M (Monetary - total blood donated in c.c.),
T (Time - months since first donation), and
a binary variable representing whether he/she donated blood in March 2007 (1 stand for donating blood; 0 stands for not donating blood).

There are 6 columns in the dataset, where the id column is a unique and random identifier. The remaining 5 features are described in the section below.

The different variables are:

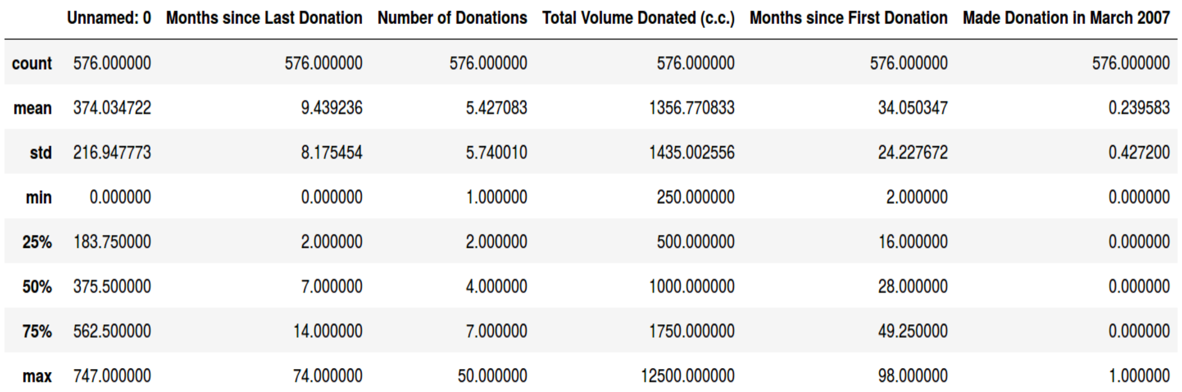
- Months since Last Donation : number of months since the last donation,
- Number of Donations : total number of donations,
- Total Volume Donated (c.c.) : total blood donated in c.c.,
- Months since First Donation : number of months since the first donation,
- Made Donation in March 2007 : a binary variable representing whether he/she donated blood in March 2007.

Additional features add are:

1)Ratio = ‘Months since First Donation’/’Months since Last Donation’
2)Ratio1 = ‘Months since First Donation’/’Number of Donations’
3)Ratio2 = ‘’Months since Last Donation’/’Number of Donations’
4)Average Donation per Month = ‘Total Volume Donated (c.c.)’/‘Months since First Donation’
5)Frequent Donor = ‘Number of Donations’ >= 5

The goal of the challenge is to predict the last column, whether he/she donated blood in March 2007.

Figure 2.0: shows the descriptive statistics of the data. We selected 500 data at random as the training set, and the rest 248 as the testing set.



	Unnamed: 0	Months since Last Donation	Number of Donations	Total Volume Donated (c.c.)	Months since First Donation	Made Donation in March 2007
count	576.000000	576.000000	576.000000	576.000000	576.000000	576.000000
mean	374.034722	9.439236	5.427083	1356.770833	34.050347	0.239583
std	216.947773	8.175454	5.740010	1435.002556	24.227672	0.427200
min	0.000000	0.000000	1.000000	250.000000	2.000000	0.000000
25%	183.750000	2.000000	2.000000	500.000000	16.000000	0.000000
50%	375.500000	7.000000	4.000000	1000.000000	28.000000	0.000000
75%	562.500000	14.000000	7.000000	1750.000000	49.250000	0.000000
max	747.000000	74.000000	50.000000	12500.000000	98.000000	1.000000

Figure 2.0: Descriptive statistics of the data

Objectives

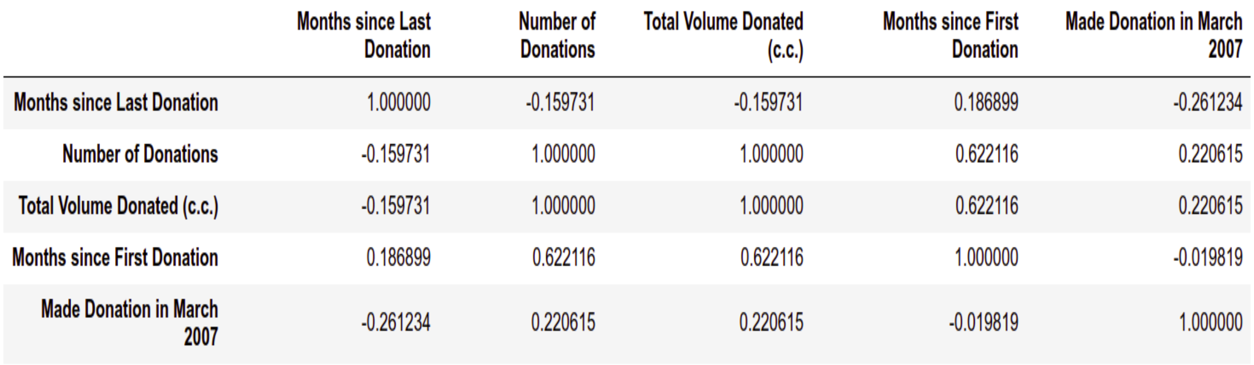
1. **To accurately predict whether the donor will donate the blood or not**
   By using various data mining techniques we can easily predict whether a person will donate the blood or not.

2. **To fetch customised blood donations statistical report.**
   We can generate customised reports to draw conclusions depending on various parameters and have a better view of people donating blood.

3. **To enhance our knowledge on the subject of Data Mining and Analysis.**
   We were able to gain more knowledge about the subject in depth by implementing data mining techniques on the given dataset.

Data Analysis

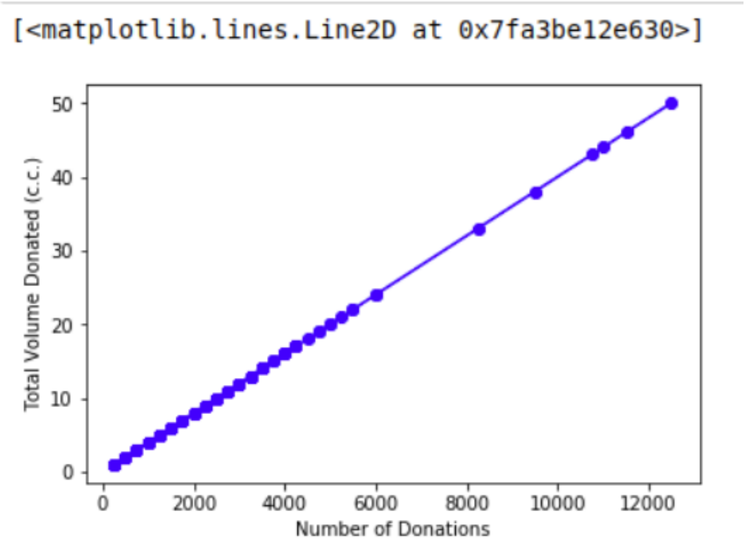
Figure 3.0: shows the correlation between the each attributes.



	Months since Last Donation	Number of Donations	Total Volume Donated (c.c.)	Months since First Donation	Made Donation in March 2007
Months since Last Donation	1.000000	-0.159731	-0.159731	0.186899	-0.261234
Number of Donations	-0.159731	1.000000	1.000000	0.622116	0.220615
Total Volume Donated (c.c.)	-0.159731	1.000000	1.000000	0.622116	0.220615
Months since First Donation	0.186899	0.622116	0.622116	1.000000	-0.019819
Made Donation in March 2007	-0.261234	0.220615	0.220615	-0.019819	1.000000

Figure 3.0: Correlation between the attributes

We observed that there is High degree of correlation between “Total Volume Donated” and “Number of Donations” (1.0) As shown in Figure 3.1, hence we decided to drop this attribute.For all other attributes relatively there was no high degree of correlation.



```
[<matplotlib.lines.Line2D at 0x7fa3be12e630>]
```

Figure 3.1

Correlation Graph for all the attributes is plotted in the figure 3.2 we can infer that variables have highly skewed distributions.

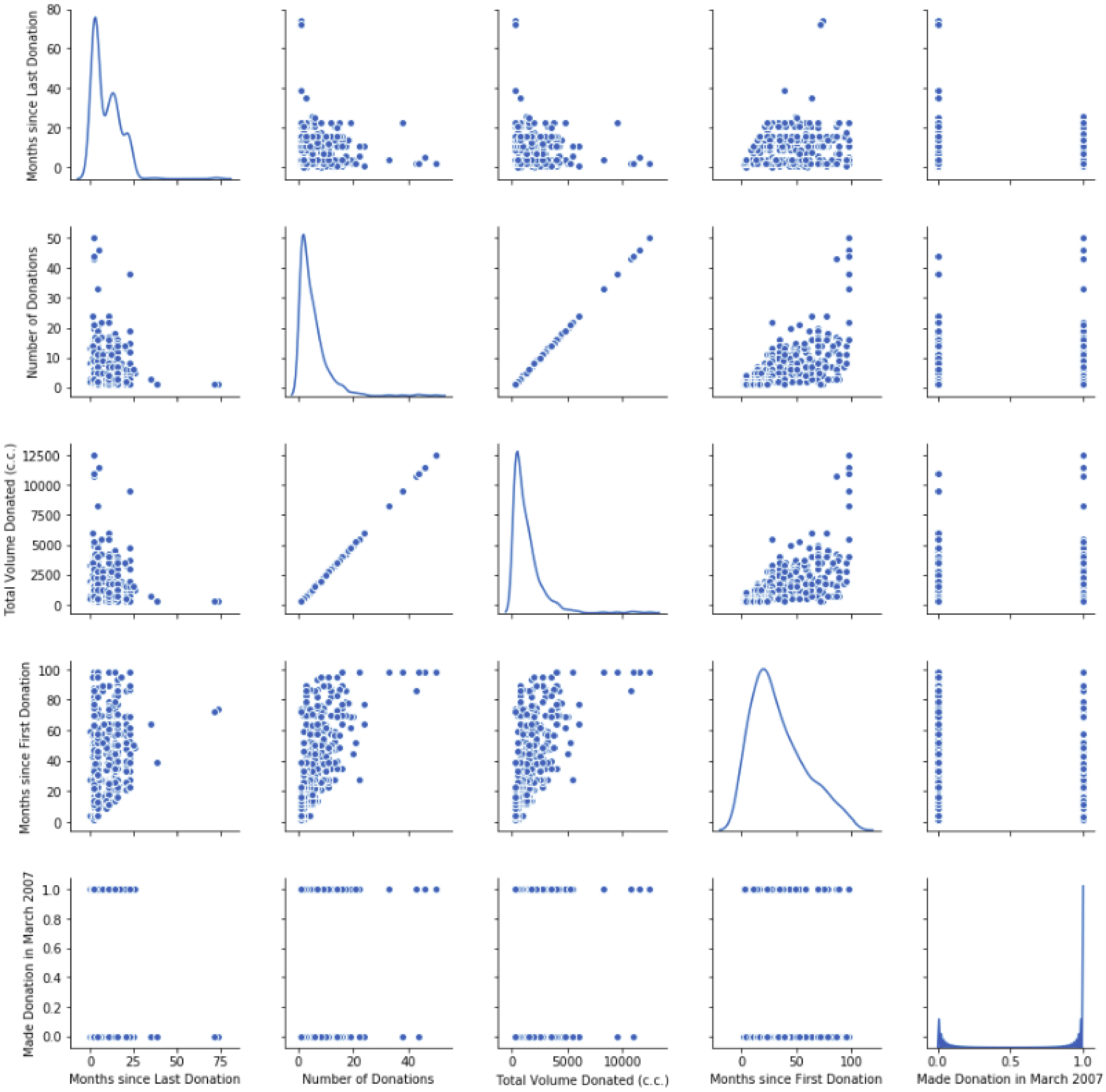
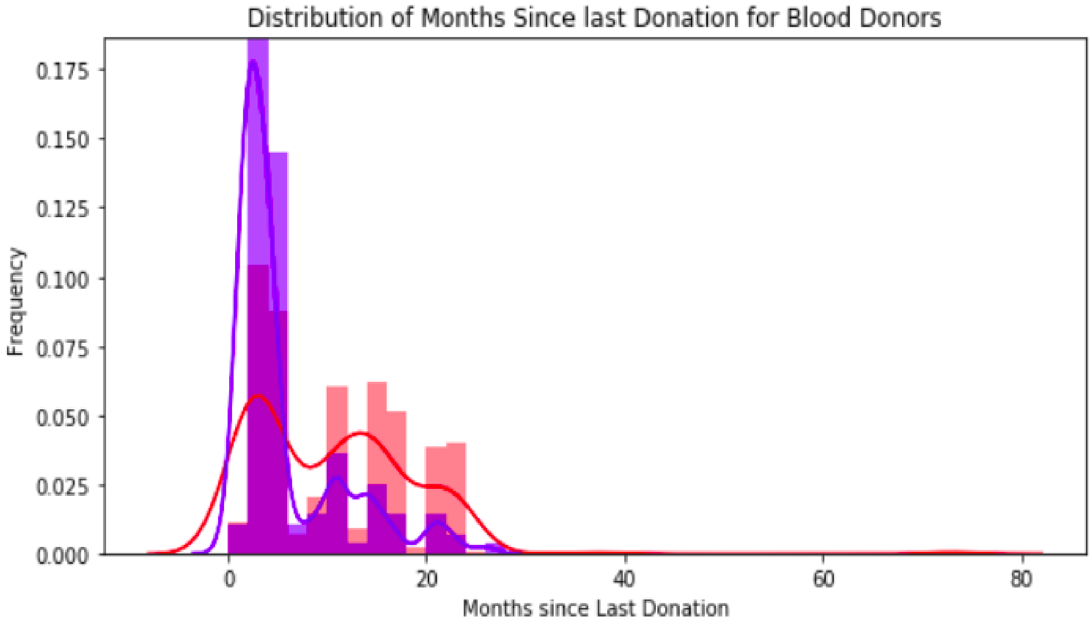


Figure 3.2

Visualisation (Recent first donation)

Figure 4.0 shows that on average, people that donated tend to have donated had fewer months since their last donation than those who did not donate.

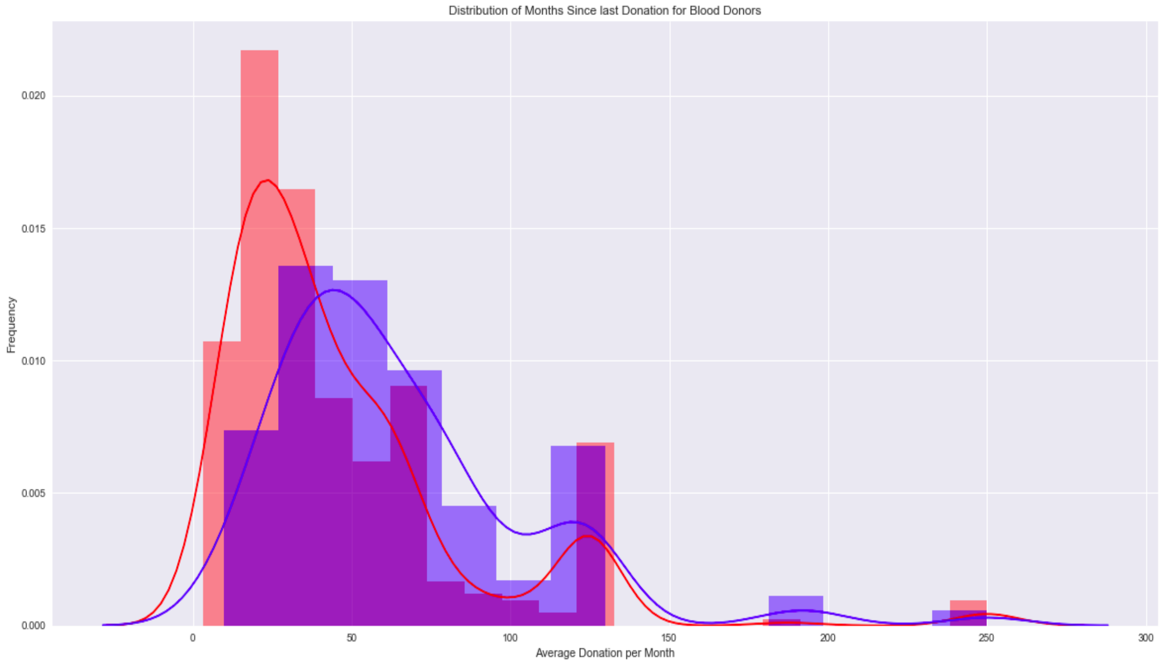


🟥 Does not Donate Blood

🟦 Donates Blood

Figure 4.0

By analysing the Figure 4.1 we can infer that Donors who, on average, donate more than 50 ml of blood per month are more likely to donate blood.



Does not Donate Blood

Donates Blood

Figure 4.1

Train vs Test

By analysing the Figure 4.2 we can infer that, the test set is very similar to the training dataset, Some outliers which have to be normalised

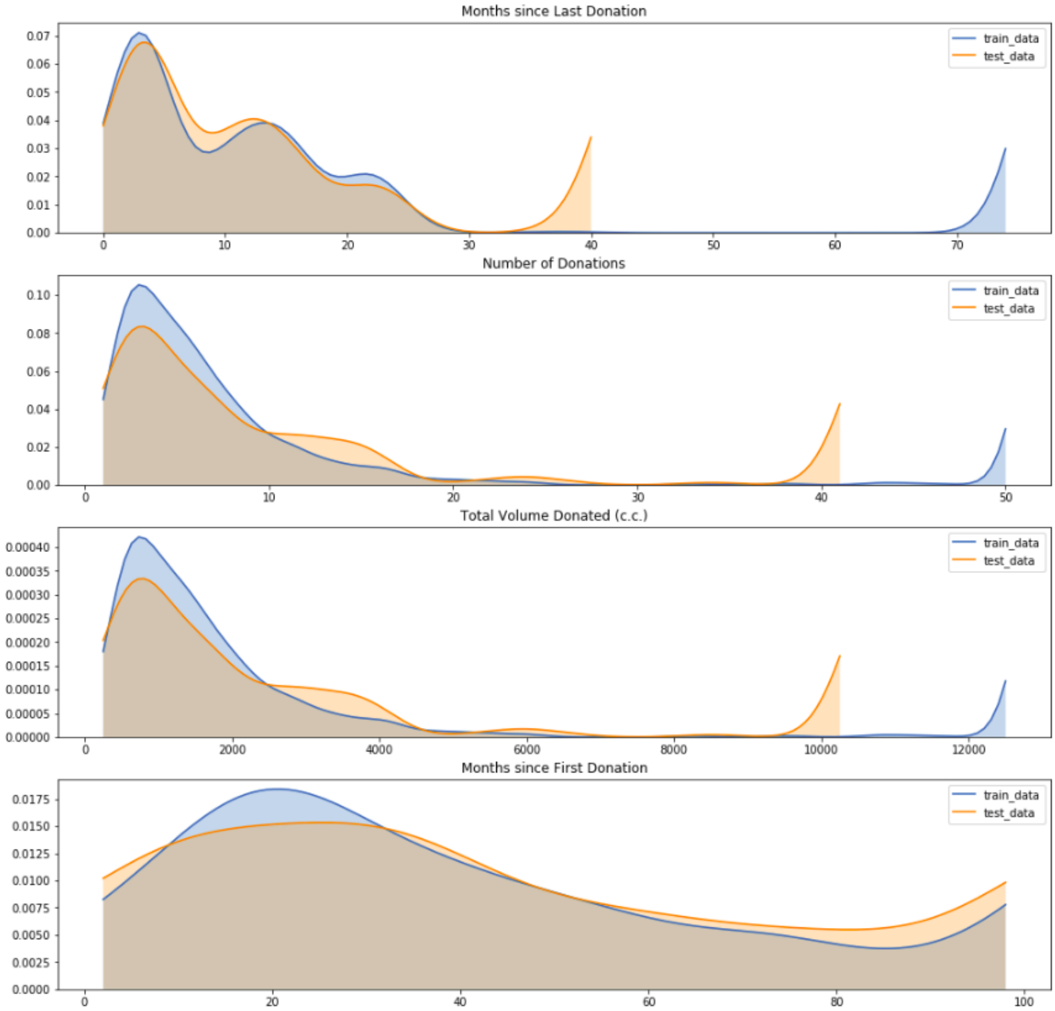


Figure 4.2

Box plot is used to find the outliers from the given dataset hence we used box plots to find the outliers in each attribute.
By observing the Figure 4.3(a) we can infer that, in the attribute Months since Last Donation has 6 outliers.

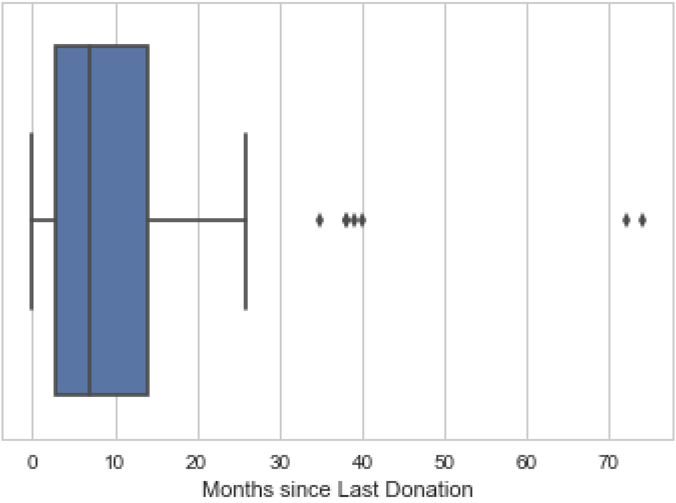


Figure 4.3(a)

By observing the Figure 4.3(b) we can infer that, in the attribute Number of Donations has 19 outliers.

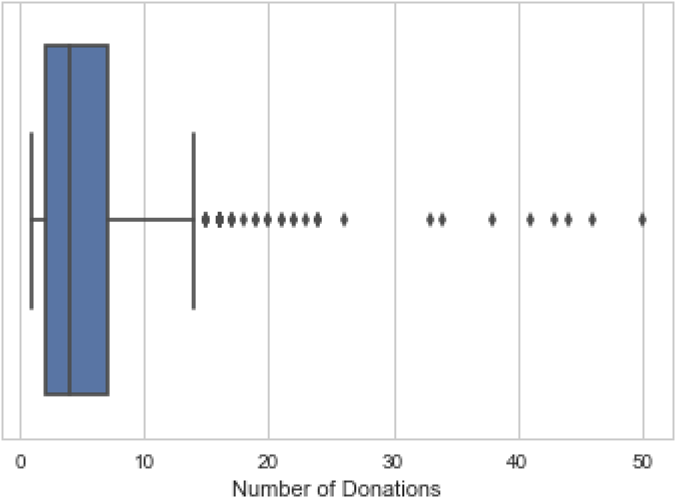


Figure 4.3(b)

By observing the Figure 4.3(c) we can infer that, in the attribute Total Volume Donated(c.c.) has 19 attributes.

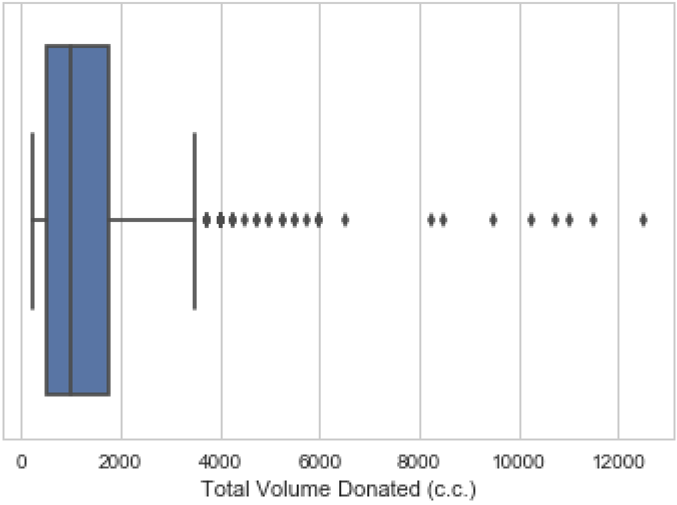


Figure 4.3(c)

By observing the Figure 4.3(d) we can infer that, in the attribute Months since First Donation has 0 attributes.

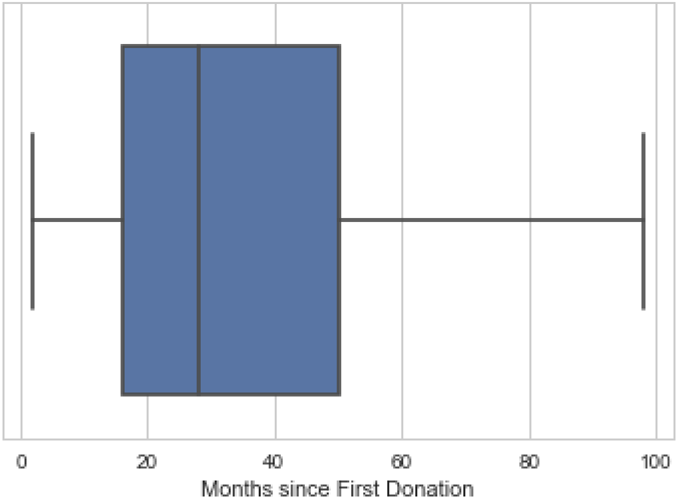
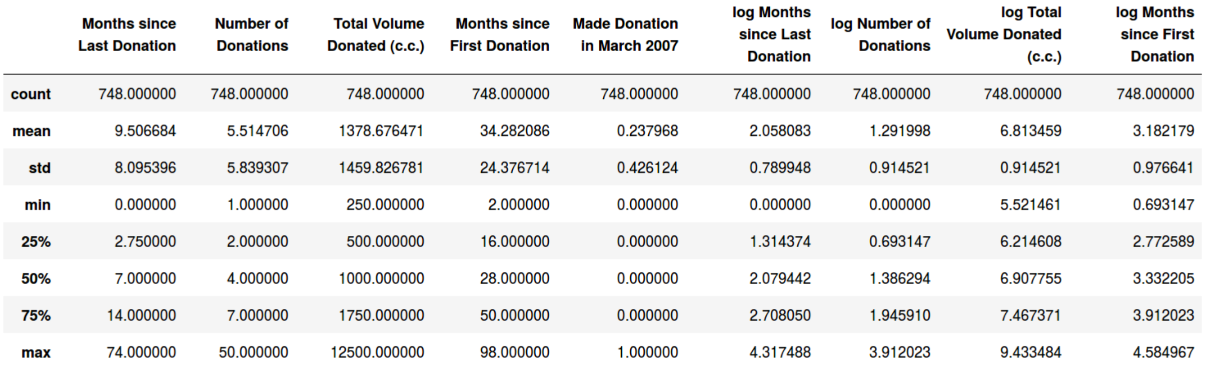


Figure 4.3(d)

Table 3 shows the descriptive statistics of the data after applying the log to the attributes. We selected 500 data at random as the training set, and the rest 248 as the testing set.



	Months since Last Donation	Number of Donations	Total Volume Donated (c.c.)	Months since First Donation	Made Donation in March 2007	log Months since Last Donation	log Number of Donations	log Total Volume Donated (c.c.)	log Months since First Donation
count	748.000000	748.000000	748.000000	748.000000	748.000000	748.000000	748.000000	748.000000	748.000000
mean	9.506684	5.514706	1378.676471	34.282086	0.237968	2.058083	1.291998	6.813459	3.182179
std	8.095396	5.839307	1459.826781	24.376714	0.426124	0.789948	0.914521	0.914521	0.976641
min	0.000000	1.000000	250.000000	2.000000	0.000000	0.000000	0.000000	5.521461	0.693147
25%	2.750000	2.000000	500.000000	16.000000	0.000000	1.314374	0.693147	6.214608	2.772589
50%	7.000000	4.000000	1000.000000	28.000000	0.000000	2.079442	1.386294	6.907755	3.332205
75%	14.000000	7.000000	1750.000000	50.000000	0.000000	2.708050	1.945910	7.467371	3.912023
max	74.000000	50.000000	12500.000000	98.000000	1.000000	4.317488	3.912023	9.433484	4.584967

Figure 4.4

By analysing the Figure 4.5 we can infer that, dealing with these outliers by taking the log of variables and reducing the ranges,
.apply(lambda x: np.log(x)),
.apply(lambda x: np.log(x+1))
(x+1) is applied because
The log of the variables look much better distributed than the initial distributions.

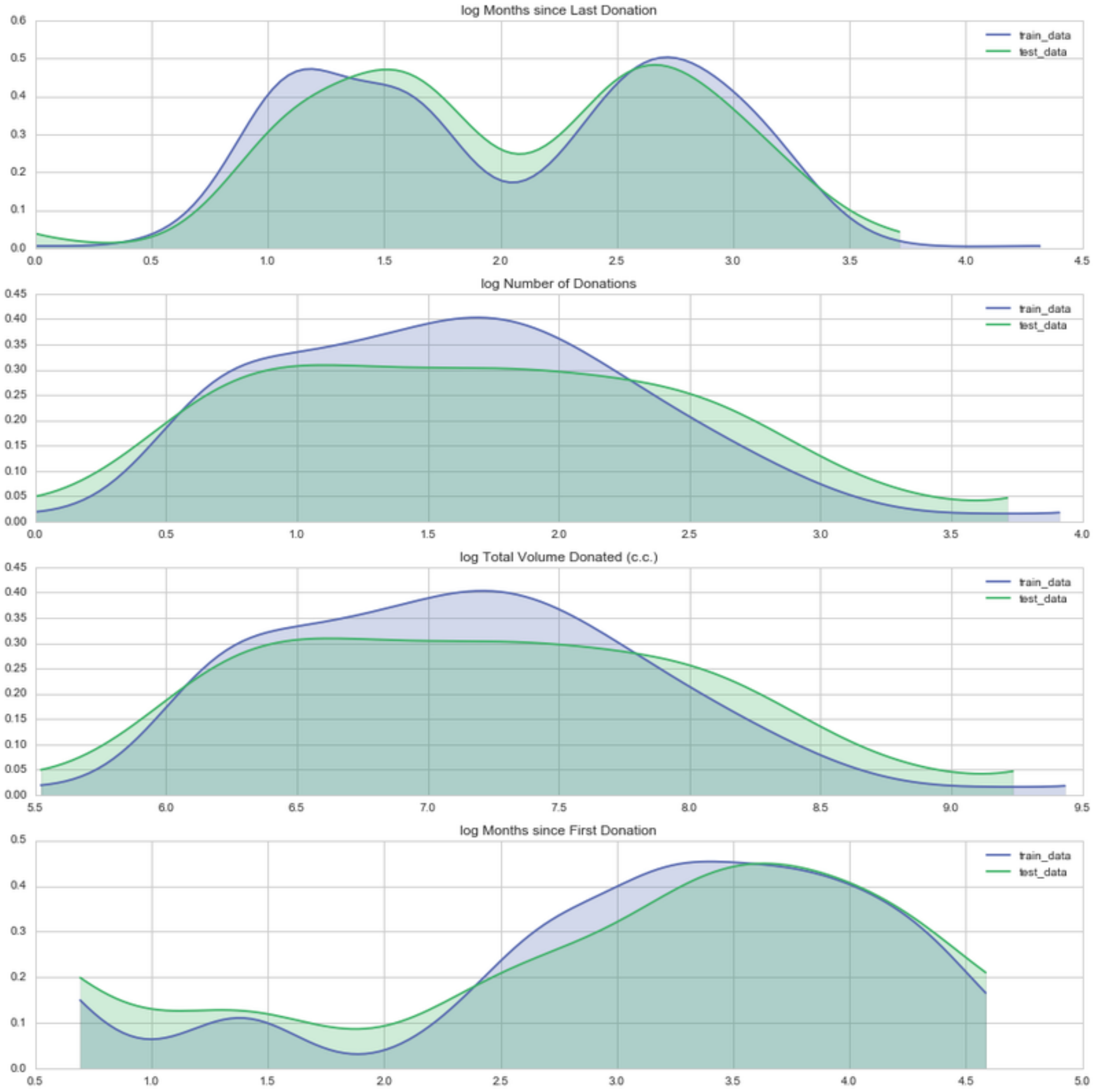
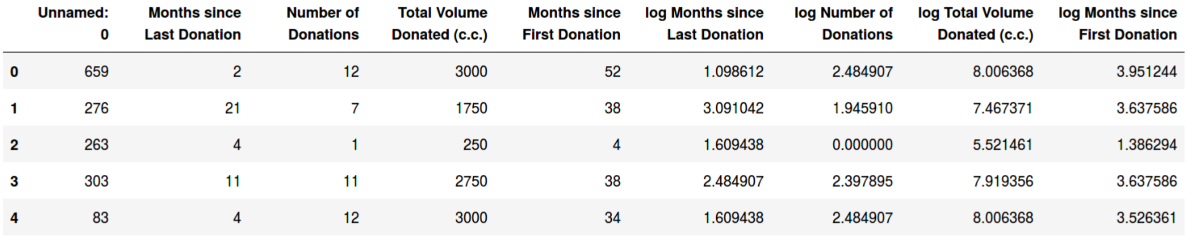


Figure 4.5

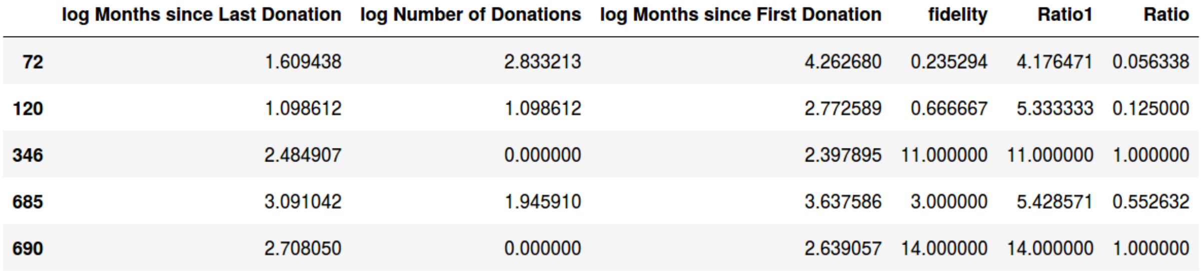
By analysing the FIgure 4.6 we can infer that, values of the attributes after applying the log.



	Unnamed: 0	Months since Last Donation	Number of Donations	Total Volume Donated (c.c.)	Months since First Donation	log Months since Last Donation	log Number of Donations	log Total Volume Donated (c.c.)	log Months since First Donation
0	659	2	12	3000	52	1.098612	2.484907	8.006368	3.951244
1	276	21	7	1750	38	3.091042	1.945910	7.467371	3.637586
2	263	4	1	250	4	1.609438	0.000000	5.521461	1.386294
3	303	11	11	2750	38	2.484907	2.397895	7.919356	3.637586
4	83	4	12	3000	34	1.609438	2.484907	8.006368	3.526361

Figure 4.6:

By analysing the Figure 4.7 we can infer that, values of the attributes after applying the log and other new features introduced.



	log Months since Last Donation	log Number of Donations	log Months since First Donation	fidelity	Ratio1	Ratio
72	1.609438	2.833213	4.262680	0.235294	4.176471	0.056338
120	1.098612	1.098612	2.772589	0.666667	5.333333	0.125000
346	2.484907	0.000000	2.397895	11.000000	11.000000	1.000000
685	3.091042	1.945910	3.637586	3.000000	5.428571	0.552632
690	2.708050	0.000000	2.639057	14.000000	14.000000	1.000000

Figure 4.7

Feature selection

By analysing the Figure 4.8 we can infer that, importance of the features or feature selection.

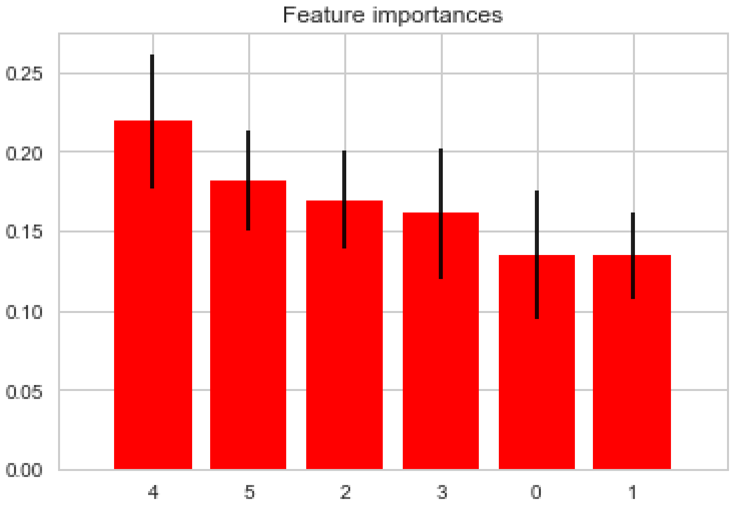


Figure 4.8

Data Preprocessing

We don’t have any NaN value attributes. As their were outliers in the attributes as shown in the figure 4.2 to remove the outliers we applied the log to each attribute. Figure 4.5 shows the attributes without the outliers. We have have done scaling.

Data Transformation

No data transformation process has been applied since we had the normalized data.

Metrics Used

The Error Metric – Log Loss

The metric in this competition is logarithmic loss, or *log loss*, which uses the *probabilities* of class predictions and the true class labels to generate a number that is closer to zero for better models, and exactly zero for a perfect model.
The formula for log loss that highly *confident* (probability close to one) *wrong* answers will contribute more to the total log loss number. This property of log loss makes it more informative alternative to accuracy.

The formula for log loss is:

$-\log P(yt|yp) = -(yt \log(yp) + (1 - yt) \log(1 - yp))$

Log Loss vs Accuracy

- **Accuracy** is the count of predictions where your predicted value equals the actual value. Accuracy is not always a good indicator because of its yes or no nature.
- **Log Loss** takes into account the uncertainty of your prediction based on how much it varies from the actual label. This gives us a more nuanced view into the performance of our model

Validation techniques

k-Fold Cross-Validation

Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample.
The procedure has a single parameter called k that refers to the number of groups that a given data sample is to be split into. As such, the procedure is often called k-fold cross-validation. When a specific value for k is chosen, it may be used in place of k in the reference to the model, such as k=10 becoming 10-fold cross-validation.
Cross-validation is primarily used in applied machine learning to estimate the skill of a machine learning model on unseen data.
It generally results in a less biased or less optimistic estimate of the model skill than other methods, such as a simple train/test split.

Machine learning models.

1)Random Forest:

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set.

2)Adaboost:

AdaBoost, short for *Adaptive Boosting*, is a machine learning meta-algorithm. It can be used in conjunction with many other types of learning algorithms to improve performance. The output of the other learning algorithms ('weak learners') is combined into a weighted sum that represents the final output of the boosted classifier. AdaBoost is adaptive in the sense that subsequent weak learners are tweaked in favor of those instances misclassified by previous classifiers. AdaBoost is sensitive to noisy data and outliers. In some problems it can be less susceptible to the overfitting problem than other learning algorithms. The individual learners can be weak, but as long as the performance of each one is slightly better than random guessing, the final model can be proven to converge to a strong learner.

3)Xg boost:

XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements machine learning algorithms under the Gradient Boosting framework. XGBoost provides a parallel tree boosting (also known as GBDT, GBM) that solve many data science problems in a fast and accurate way. The same code runs on major distributed environment (Hadoop, SGE, MPI) .

4)SVM:

In machine learning, support vector machines (SVMs, also support vector networks) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier (although methods such as Platt scaling exist to use SVM in a probabilistic classification setting). An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

5)Logistic Regression:

The logistic model (or logit model) is a widely used statistical model that, in its basic form, uses a logistic function to model a binary dependent variable; many more complex extensions exist. In regression analysis, logistic regression (or logit regression) is estimating the parameters of a logistic model; it is a form of binomial regression. Mathematically, a binary logistic model has a dependent variable with two possible values, such as pass/fail, win/lose, alive/dead or healthy/sick; these are represented by an indicator variable, where the two values are labeled "0" and "1". In the logistic model, the log-odds (the logarithm of the odds) for the value labeled "1" is a linear combination of one or more independent variables ("predictors"); the independent variables can each be a binary variable (two classes, coded by an indicator variable) or a continuous variable (any real value)

Comparison of the learning models:

Models	Parameters	Accuracy,log loss
1)Random Forest:	Number of estimators: 1000, minimum sample leaf =9,minimum sample split =3,learning rate= 0.01	0.7950, 04502
2)Adaboost:	Number of estimators: 25, random state =12345,learning rate= 0.05	0.7932, 0.4687
3)Xg boost:	Base score=0.5,learning rate=0.01,number of estimators=200,max depth=3	0.7967, 0.5002
4)SVM:	Learning rate=0.01,cache size=200,random state=12345	0.7266, 0.4936
5)Logistic Regression	Learning rate=0.1,kernel=linear	0.8121, 0.4487

Bias variance tradeoff for logistic regression

We got the best results for the logistic regression so we want to check the bias and the variance to see how we can improve.

In figure 5.0 we plotted the Training error and validation error to illustrate the above tradeoff. The difference between the training error curve and validation error curve is called as Variance. The training error curve depicts the bias.

These are the bias and variance so we explored the deep learning methods to improve the model.

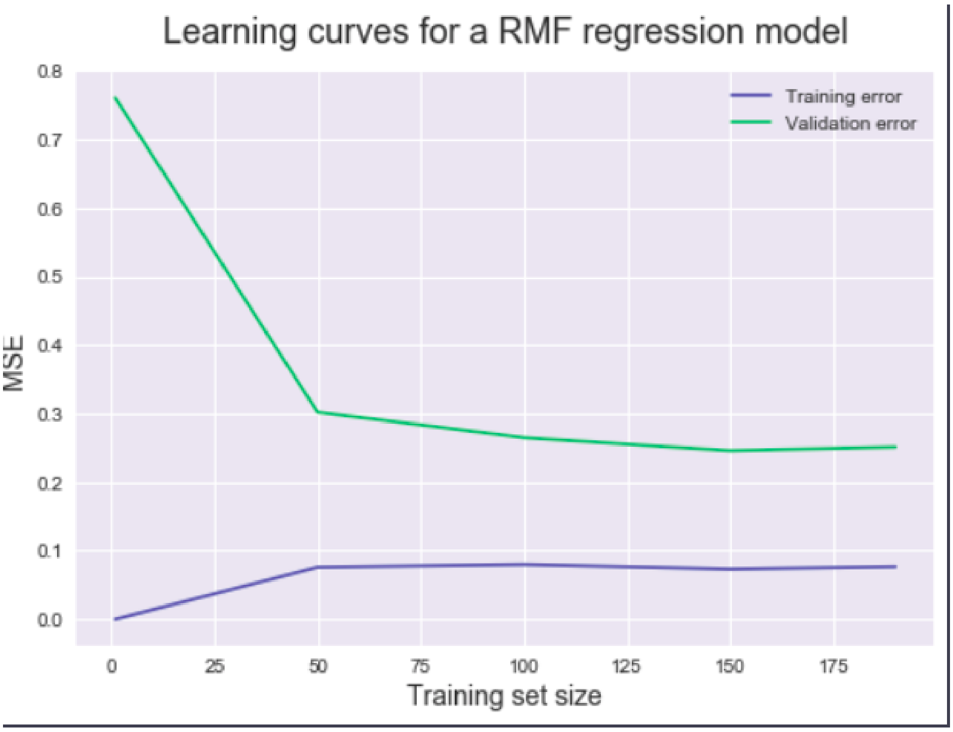
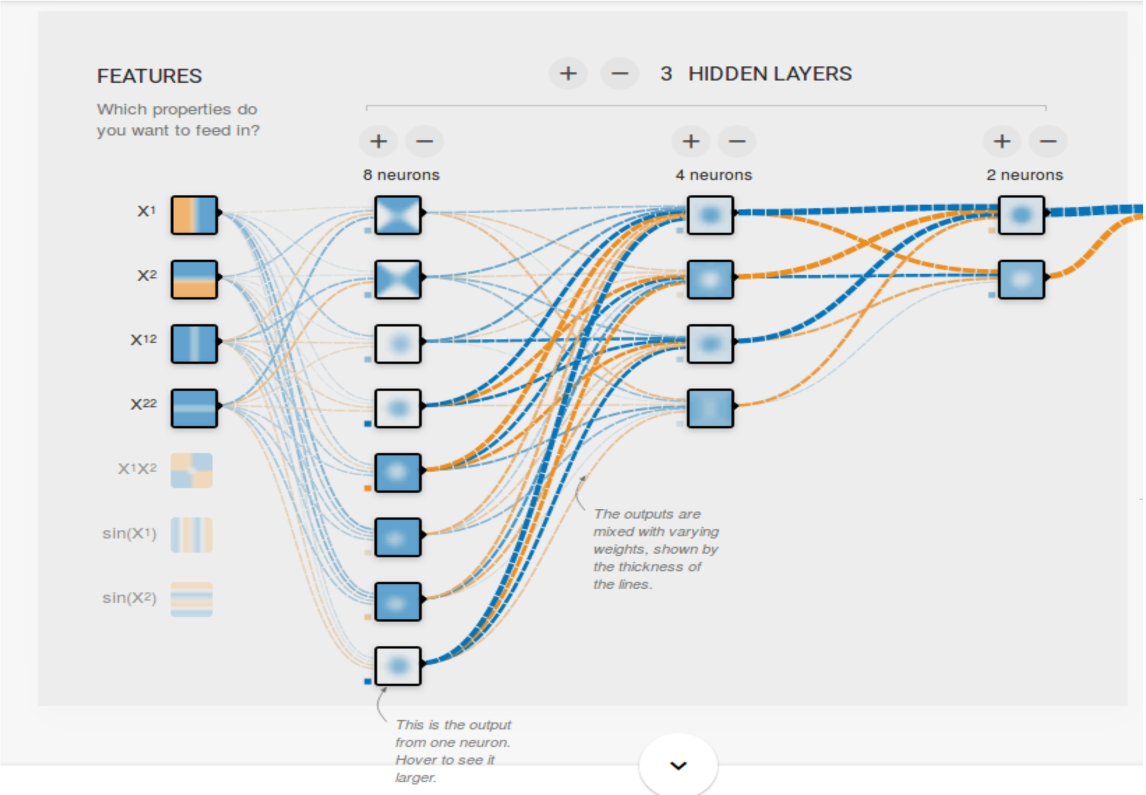


Figure 5.0

Deep learning methods

Neural Networks:

Artificial neural networks (ANN) or connectionist systems are computing systems vaguely inspired by the biological neural networks that constitute animal brains.[1] The neural network itself is not an algorithm, but rather a framework for many different machine learning algorithms to work together and process complex data inputs.[2] Such systems "learn" to perform tasks by considering examples, generally without being programmed with any task-specific rules. For example, in image recognition, they might learn to identify images that contain cats by analyzing example images that have been manually labeled as "cat" or "no cat" and using the results to identify cats in other images. They do this without any prior knowledge about cats, for example, that they have fur, tails, whiskers and cat-like faces. Instead, they automatically generate identifying characteristics from the learning material that they process.



The outputs are mixed with varying weights, shown by the thickness of the lines.

This is the output from one neuron. Hover to see it larger.

Figure 6.0

Input Dimension : 4
Hidden Layer: 3
D layer 1: 32
D layer 2: 16
D layer 3: 4

Output Layer: 1
Epchs: 500
Log Loss :0.27

Preprocessing for Neural Network

Scaling of data

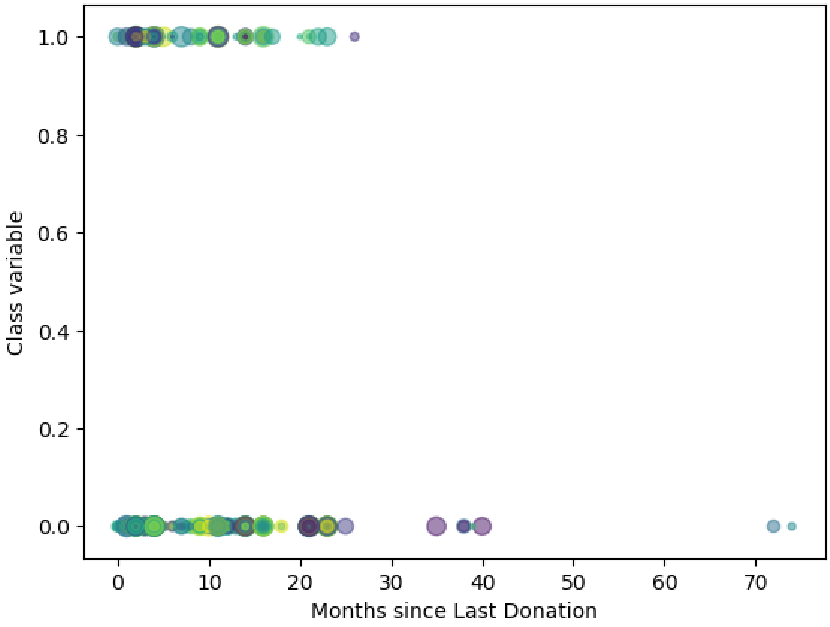


Figure 6.1(a)

Before scaling

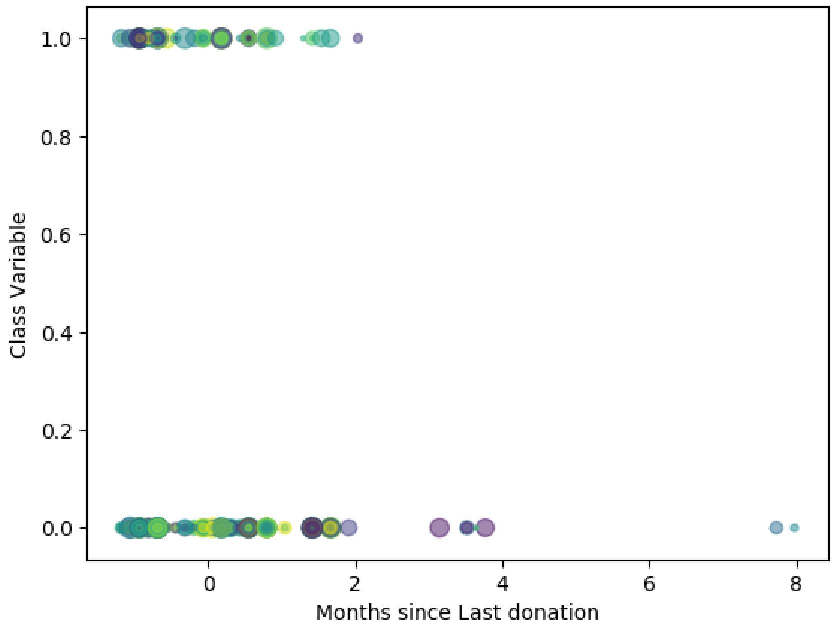


Figure 6.1(b)

After scaling

Did Scaling Matter?

- Yes Scaling did matter to us

Input Dimension : 4

Dense Layer: 4
D layer 1: 256
D layer 2: 128
D layer 3: 64
D layer 4: 32

Output Layer: 1
Epchs: 500

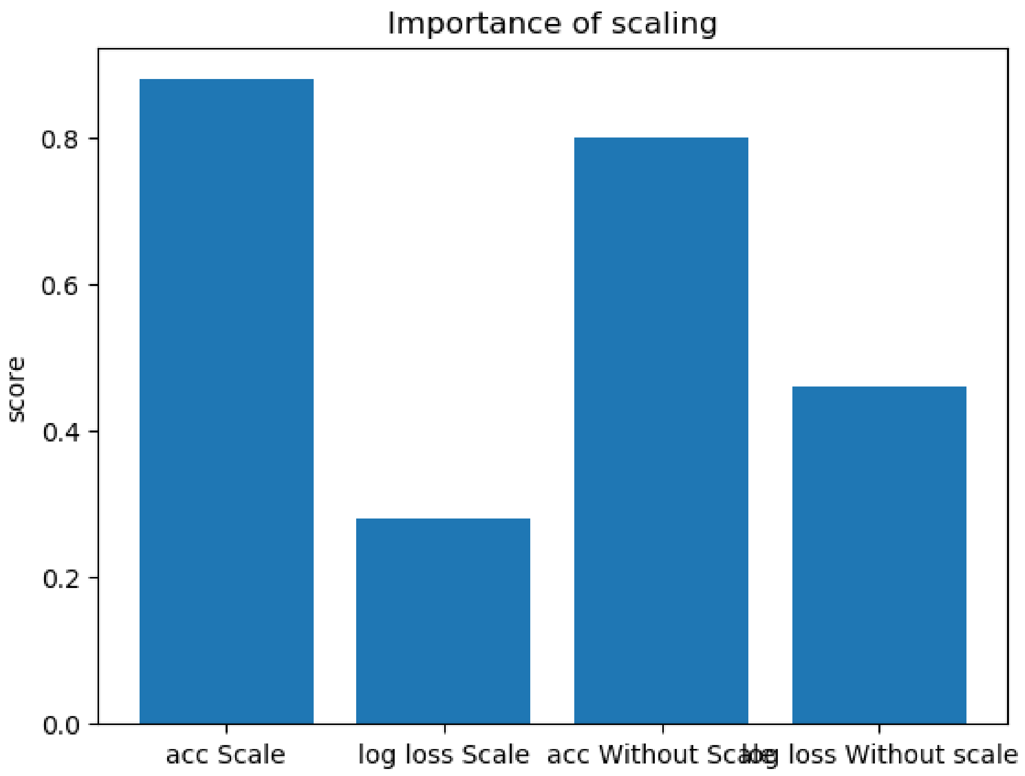


Figure 6.1(c)

<u>The Journey</u>: 38 submissions , more to come …….

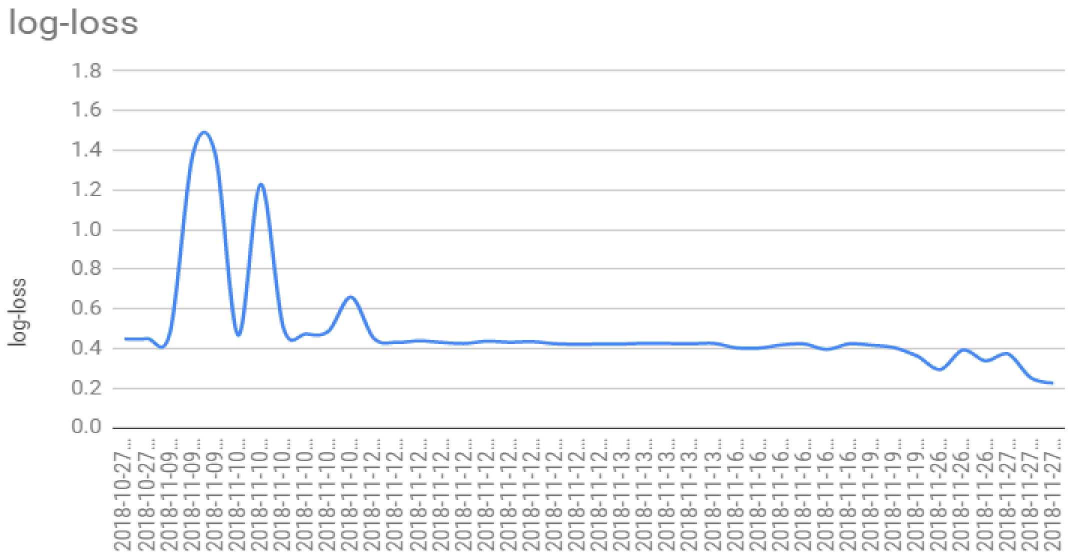


Figure 7.0

With log loss 0.2259
Today we Stand at 36th position out of 5125+ participants on leader board

Conclusion

- The problem was to predict whether the donor will donate the blood or not.
- There was a considerable difference in the scores of machine learning algorithms and neural networks, for us Neural networks worked better than machine learning algorithms so we fined tuned the Neural Networks to get better results.
- Log Loss of Neural Network is 0.2259
- We still have 3 months time and we are trying to get a rank with single digit

References:

https://www.drivendata.org/competitions/2/warm-up-predict-blood-donations/

https://stackoverflow.com/

https://pandas.pydata.org/pandas-docs/stable/tutorials.html

https://matplotlib.org/users/pyplot_tutorial.html