For this problem we are investigating a classification problem. Specifically, trying to determine the likelihood that a donor will donate blood to a roving blood collection facility in the month of March. This information would be useful in determining resources required/expected as the van collects blood from donors.

We are supplied with a training set and a test set from the website drivendata.org. (https://www.drivendata.org/competitions/2/warm-up-predict-blood-donations/) Both sets contain four independent variables. The training set contains a dependent variable, which we are tasked with trying to predict in the test set. The independent variables are:

- `Months since Last Donation`: this is the number of months since this donor's most recent donation.
- `Number of Donations`: this is the total number of donations that the donor has made.
- `Total Volume Donated`: this is the total amount of blood that the donor has donated in cubic centimeters.
- `Months since First Donation`: this is the number of months since the donor's first donation.

The dependent variable in the training set is:

- `Made Donation in March 2007`: this is the classifier on whether a given donor gives blood or not.

We believe this problem is well within scope for data scientists to be called upon to solve, as it is a classification type problem, with available cross-sectional data. We will be using two techniques as learned in our class to try and forecast the probability that the donors in the test set made a blood donation.

To back our assumption that this is an appropriate problem, and ways to approach, we found the following articles.

Hyndman, Rob et al. (2014) *Forecasting: Principles and Practice*, *Chapter 5*. Otexts.com.

Dayton, Mitchell, "Logistic Regression Analysis", September 1992, *Department of Measurement, Statistics & Evaluation,* University of Maryland

Kuhn, Max, "Building Predictive Models in R Using the caret Package", November 2008, *Journal of Statistical Software*, Volume 28, Issue 5.

Zito, Elena et al, “Adolescents and blood donation: motivation, hurdles and possible recruitment strategies,” January 2012, *Blood Transfus 10*: pp 45-58.

These articles/textbook talk about the predictive ability of different regression techniques on classification problems, which is exactly what we will be doing with these data sets, along with motivation of blood donors.

Data munging for this set was minimal to non-existent, as it is a training set from a data science competition website. No cleaning was required. The training set consists of 576 observations of the five previously discussed variables, the test set is 200 observations of the previously discussed independent variables. Immediately, we notice a strong correlation between amount of blood donated and the number of time that a donor gave blood. Every donation is 250 cc’s of blood, therefore the information is redundant, so we decided to disregard the amount of blood given, and use the number of times as our independent variable. This reduced the number of independent variable from four to three.

Summary statistics as follows:

Training Set:

```
$Months.since.Last.Donation
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  0.000   2.000   7.000   9.439  14.000  74.000 

$Number.of.Donations
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  1.000   2.000   4.000   5.427   7.000  50.000 


$Months.since.First.Donation
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
   2.00   16.00   28.00   34.05   49.25   98.00 

$Made.Donation.in.March.2007
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 0.0000  0.0000  0.0000  0.2396  0.0000  1.0000 
```

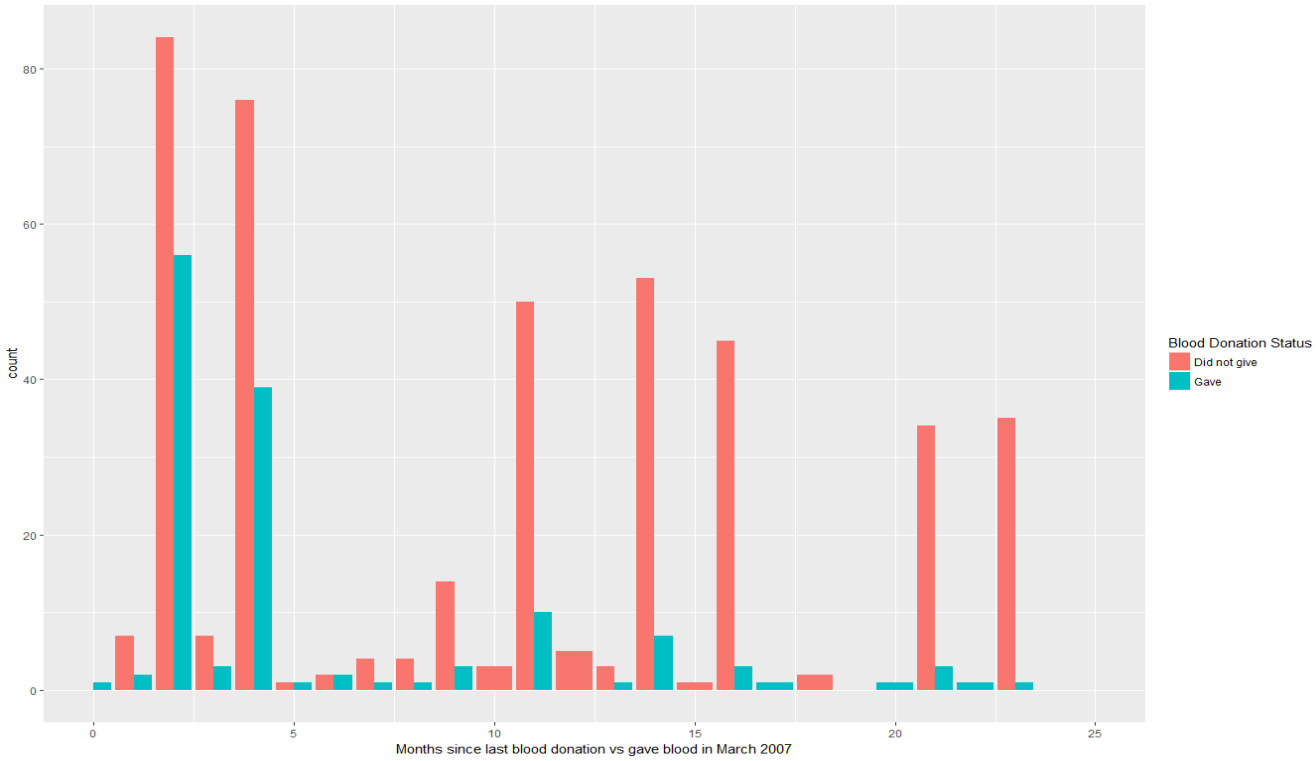
Test Set:

```
$Months.since.Last.Donation
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  0.000   4.000   7.000   9.495  14.000  40.000 

$Number.of.Donations
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  1.000   2.000   4.000   5.935   8.000  41.000 

$Months.since.First.Donation
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
   2.00   14.00   31.00   35.48   52.00   98.00 
```

(the dependent variable shows that approximately 24 percent of previous blood donors in the training set, gave blood in the month of March)

In the training set, we can gather the following information about the differences between those that donated, and those that did not: The mean of months since last donation for people who did and did not give blood in March is 5.6 months vs 10.6 months respectively. The mean of total number of donations is 7.7 vs 4.7 respectively. This tells us that people who are likely to donate in March, are coming back quicker and have made more donations than those that do not come back to donate. In other words, if it has been a longer time since someone has come in to donate, or if they have not given blood as often, then we expect a lower percentage chance that they will donate in March.

.A visual description of the data:

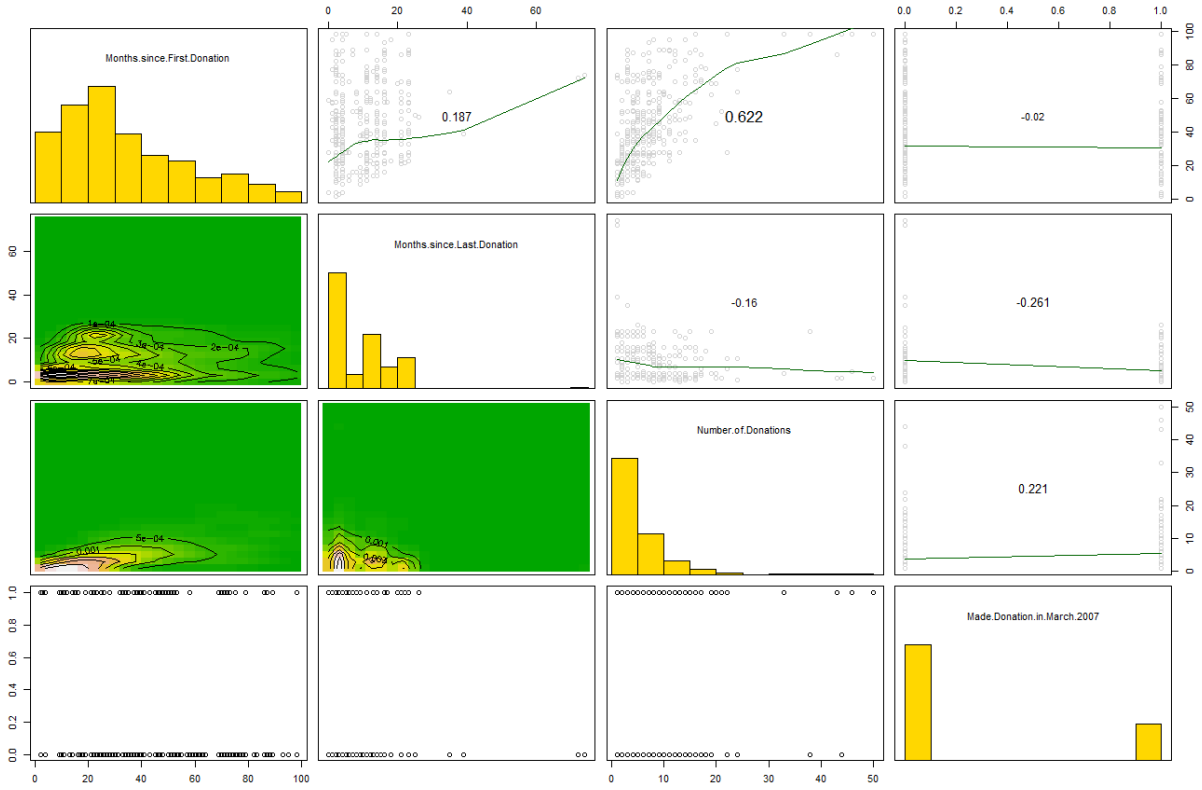


(Figure 1)

Figure 1 shows graphically how many people came back in the training set to give and not give blood, broken up by number of months since their last visit. It shows that for people who come back

within six months are much more likely to give blood than if they go past six months. This graph also seems to show some sort of periodicity that affects how many people are coming in. We hypothesize this is due to promotional strength of the blood drive awareness. (Note, four outliers are not on this chart to aid in readability, it is for donors that did not give blood at 35, 39, 72, 74 months)

(figure 2)



KDEPairs showing correlation between all variables in our dataset

From figure 2, we observe the positive correlation between number of donation and months since first donation (.622) and between months since last donation and months since first donation (.187), and the negative correlation between number of donation and months since last donation (-.16).

For our initial modeling approach, we went with a logistic regression model, as we are trying to predict the likelihood that a given donor will come back in to give blood based on the independent variables.

We initially split the data into training and test sets using random selection. This allowed us to test the viability of the model on a known result set. Although we can determine internal model fit by using various measures, the best way to see how different models handle new information is to test them against data that we know the answers, which isn't possible if we completely train our model on all known data. We went with a 70 to 30 split.

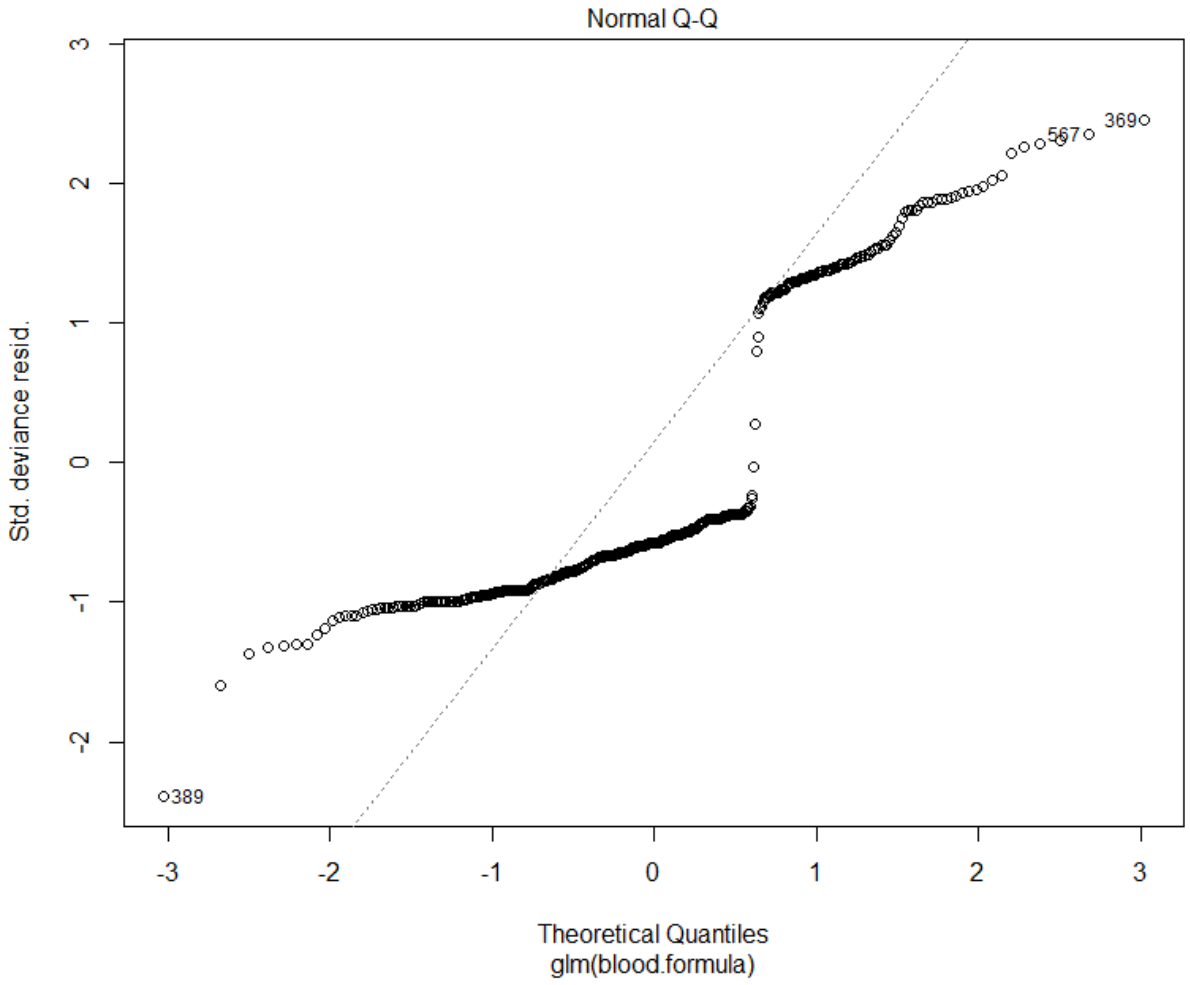
Model 1 using glm in R (glm is used to fit generalized linear models, specified by giving a symbolic description of the linear predictor and a description of the error distribution.)

```
blood.formula <- Made.Donation.in.March.2007 ~
Number.of.Donations + Months.since.Last.Donation + Months.since.First.Donation
blood.Model <- glm(formula = blood.formula, data = bloodTrain, family = "binomial")
blood.prob <- predict(blood.Model, newdata = bloodTest, type = 'response')
result1 <-cor(round(blood.prob), bloodTest$Made.Donation.in.March.2007)
```

Which resulted in:

```
Call:
glm(formula = blood.formula, family = "binomial", data = bloodTrain)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.3188  -0.8485  -0.5708   1.1503   2.4532

Coefficients:
                            Estimate Std. Error z value Pr(>|z|)
(Intercept)                -0.358558   0.231462  -1.549 0.121357
Number.of.Donations         0.109862   0.033022   3.327 0.000878 ***
Months.since.Last.Donation -0.088823   0.020933  -4.243  2.2e-05 ***
Months.since.First.Donation -0.017139   0.007357  -2.330 0.019830 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 468.49  on 402  degrees of freedom
Residual deviance: 421.63  on 399  degrees of freedom
AIC: 429.63

Number of Fisher Scoring iterations: 5
```

This model shows significance for all predictor variables with $p < .05$. Positive coefficients for number of donations and negative for months. Looking at a QQ plot of the residuals:



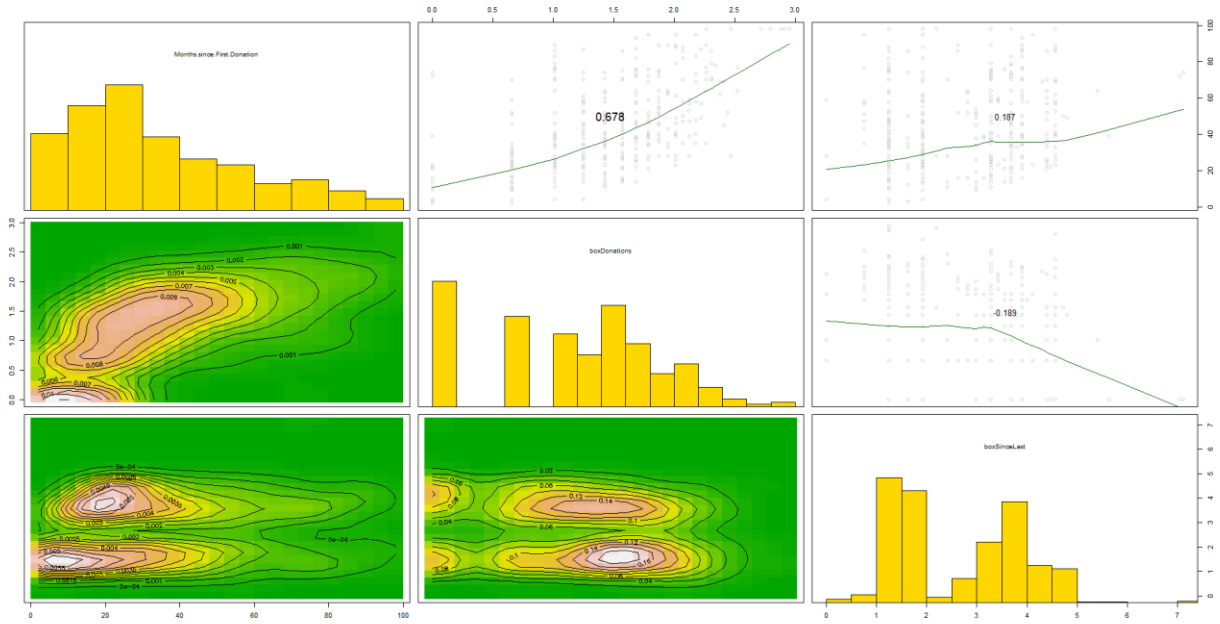
(figure 3)

Residuals QQ plot

This tells us that the residuals are not following a normal distribution. With a correlation of .296 between prediction and actual values.

Our first attempt gave us our benchmark. We then started looking at how to get more information out of the residuals. In looking at the kdepairs, we noticed that the months since first donation are right skewed, and applied a boxcox transformation. This did slightly increase AIC to 434.6, but also increase our correlation to .345. When boxcox transformations were applied to two of the three variables and a dummy variable that was applied to coincide with months since last donation <=

11, the AIC decreased to 414, and correlation jumped to .42. The reason we tried that variable was due to the following correlation:



(Figure 4)

Kdepairs of boxcox transformed independent variables

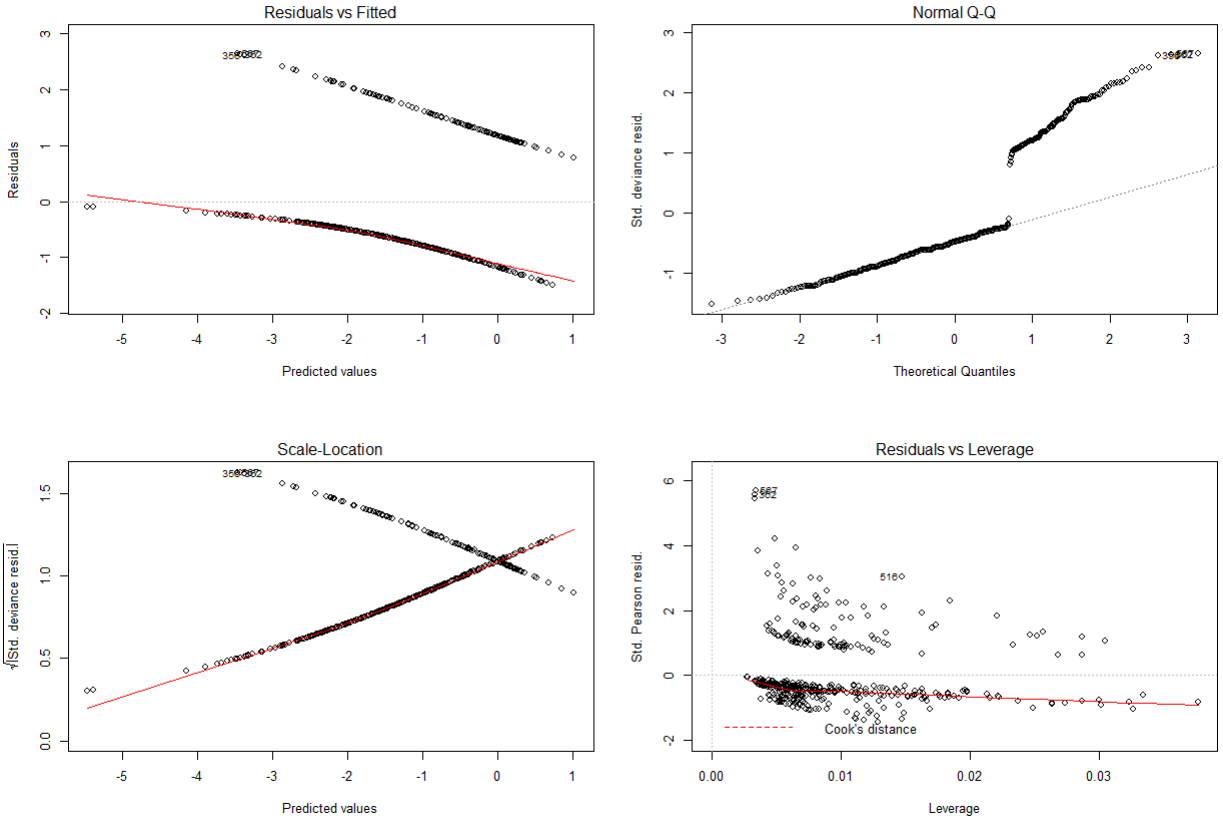
We noticed the pronounced slope change in quadrant (3,2) which is the boxcox transformed correlation between months since last donation and number of donations.

Final R Formula:

```
blood.formula <- Made.Donation.in.March.2007 ~
  BoxCox(Number.of.Donations, -.1509) +
  BoxCox(Months.since.Last.Donation + 1, .2154) +
  Months.since.First.Donation +
  ifelse(BoxCox(Months.since.Last.Donation + 1, .2154) <= 3.2, 1, 0)
```

Model summary:

```
Call:
glm(formula = blood.formula, family = "binomial", data = bloodTrain)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.5512  -0.7644  -0.5232   0.9882   2.5160

Coefficients:
                                                                          Estimate
Std. Error z value Pr(>|z|)
(Intercept)                                                              -1.494727
0.961814  -1.554 0.120168
BoxCox(Number.of.Donations, -0.1509)                                      1.382115
0.268094   5.155 2.53e-07
BoxCox(Months.since.Last.Donation + 1, 0.2154)                           -0.204663
0.240553  -0.851 0.394880
Months.since.First.Donation                                              -0.030146
0.008013  -3.762 0.000168
ifelse(BoxCox(Months.since.Last.Donation + 1, 0.2154) <= 3.2, 1, 0)       0.507007
0.573060   0.885 0.376299

(Intercept)
BoxCox(Number.of.Donations, -0.1509)                                     ***
BoxCox(Months.since.Last.Donation + 1, 0.2154)
Months.since.First.Donation                                              ***
ifelse(BoxCox(Months.since.Last.Donation + 1, 0.2154) <= 3.2, 1, 0)
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 468.49  on 402  degrees of freedom
Residual deviance: 405.01  on 398  degrees of freedom
AIC: 415.01

Number of Fisher Scoring iterations: 4

Confusion Matrix with cutoff value of .5:

Confusion Matrix:
          obs
prediction   0   1
         0 132  16
         1  11  14
.
```



(figure 5)
All plots of residuals from r plot() function

The lower AIC and informational plots told us that our model was doing a fairly decent job of capturing the available information from the training set, however there does appear to be some more information that can be squeezed from the residuals, we just haven't figured that part out yet.

Using this model, we uploaded a prediction to drivendata.org, and the final result was:

Submissions

BEST SCORE	CURRENT RANK	# COMPETITORS	SUBS. TODAY
0.4464	495	3499	2 / 3

EVALUATION METRIC

Log loss $= -\frac{1}{n}\sum_{i=1}^{n}\left[y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)\right]$

The metric used for this competition is logarithmic loss. $\hat{y}$ is the probability that $y = 1$. Logarithmic loss provides a steep penalty for predictions that are both confident and wrong. The goal is to minimize the log loss.

SUBMISSIONS

Score	Submitted by	Timestamp
0.4632	nfoboy	Oct. 28, 2017, 12:28 a.m.

Our next model that we attempted to pump information out of the data was the caret package in R. Below is the R code in how we split our data into test and training sets, (as before with glm)

```
set.seed(513)
train_index <- createDataPartition(y=data$md, p = 0.75, list=FALSE)
train_data <- data[train_index,]
test_data <- data[-train_index,]
```

In addition to the logistic regression models we have implemented as discussed above, we attempted to increase the accuracy of predictions using the Boosting method. In this method, we are building multiple models (typically of the same type), each of which learns to fix the prediction errors of a prior model in the chain.

For creating such a predictive model we have used the caret package (short for _C_lassification _A_nd _RE_gression _T_raining).

The train function is used to evaluate, using resampling, the effect of model tuning parameters on performance and choose the “optimal” model across these parameters. By default, simple bootstrap resampling is used in the train function. However, the function trainControl can be used to specifiy the type of resampling and here we are cross-validating the data 5 times, i.e. training 5 times on different portions of data before setting the best tuning parameters.

```
formula <- md ~ log.mld + log.nod + log.mfd + log.avgd

dataCt <- trainControl(method='cv', number=5, returnResamp='none', summaryFunction =
twoClassSummary, classProbs = TRUE)
```

With the above control parameters for train function, we are fitting the predictive model using the gradient boosting machine (GBM) model and preprocessing the data by centre and scale.

```
newm <- train(formula, data = train_data, method = "gbm",
              trControl=dataCt,
              metric = "ROC",
              preProc = c("center", "scale"))

print(newm)
```

```
## Stochastic Gradient Boosting
##
## 433 samples
##   4 predictor
##   2 classes: 'Donated', 'Not.Donated'
##
## Pre-processing: centered (4), scaled (4)
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 348, 346, 346, 346, 346
## Resampling results across tuning parameters:
##
##   interaction.depth  n.trees  ROC        Sens       Spec
##   1                   50      0.7249693  0.1919048  0.9483450
##   1                  100      0.7133441  0.2490476  0.9422844
##   1                  150      0.7091339  0.2690476  0.9270862
##   2                   50      0.7184794  0.2209524  0.9362704
##   2                  100      0.7095011  0.3357143  0.9180420
##   2                  150      0.7019115  0.3357143  0.9118881
##   3                   50      0.7026325  0.2966667  0.9301632
##   3                  100      0.7133579  0.3366667  0.9179021
##   3                  150      0.7007391  0.3357143  0.9240093
##
## Tuning parameter 'shrinkage' was held constant at a value of 0.1
##
## Tuning parameter 'n.minobsinnode' was held constant at a value of 10
## ROC was used to select the optimal model using  the largest value.
## The final values used for the model were n.trees = 50, interaction.depth
##  = 1, shrinkage = 0.1 and n.minobsinnode = 10.
```

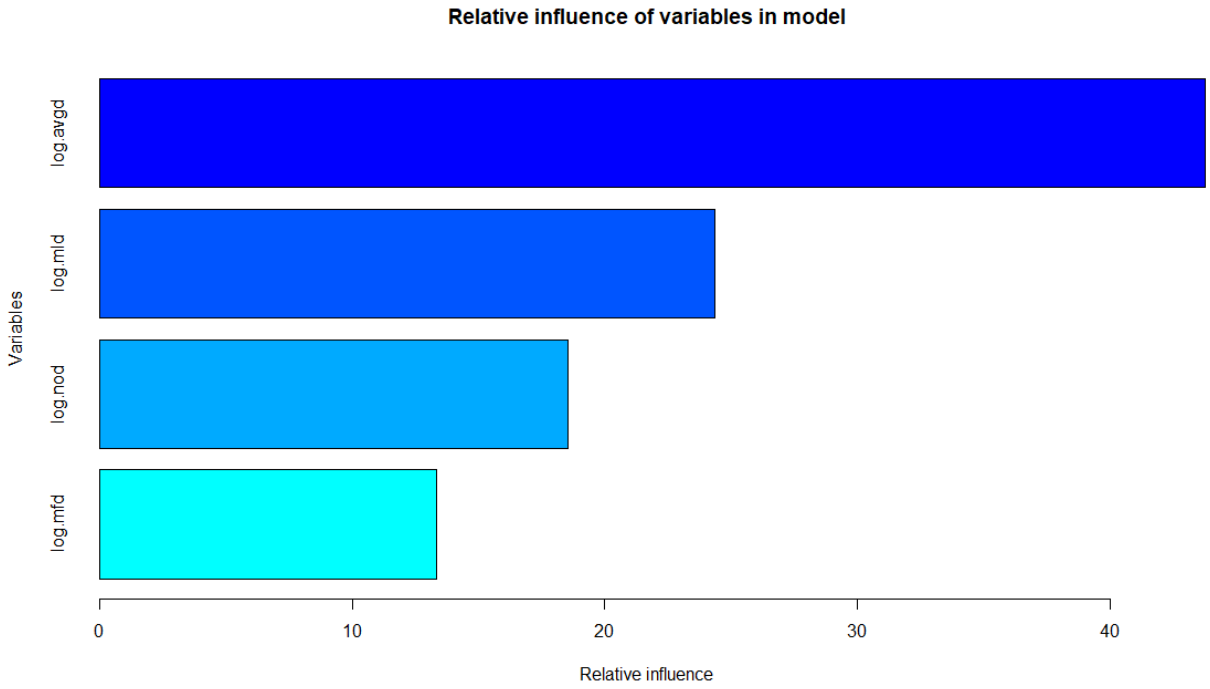
We observe that ROC was used to select the optimal model using the largest value. The final values used for the model were:

- number of iterations, i.e. trees, called n.trees = 50
- complexity of the tree, called interaction.depth = 1
- learning rate: how quickly the algorithm adapts, called shrinkage = 0.1
- the minimum number of training set samples in a node to commence splitting called n.minobsinnode = 10

Let us observe the summary statistics for the final model:

```
summary(newm)
```

```
##                var  rel.inf
## log.avgd log.avgd 43.78253
## log.mld   log.mld 24.34423
## log.nod   log.nod 18.53259
## log.mfd   log.mfd 13.34066
```



(figure 6)

From figure 6 summary statistics, we can see the relative influence of the variables in prediction of the "Made Donation in March 2007". We observe a 44 percent influence avgd on the prediction variable and similarly 24 percent, 19 percent, 13 percent of relative influence of variables mld, nod, mfd on the predictive variable respectively.

The varImp function gives the importance of variables in the model.

```
varImp(newm)
```

```
## gbm variable importance
##
##          Overall
## log.avgd  100.00
## log.mld    36.15
## log.nod    17.06
## log.mfd     0.00
```

We observe that avgd (Months since first donated/Number of Donations) is the most important variable for our prediction model.

Let us make the predictions for test_data as follows:

```
preds<- predict(newm,test_data)
predprob <- predict(newm,test_data,type="prob")
predprob <- predprob['Donated']
print(postResample(pred=preds, obs=test_data$md))

##  Accuracy     Kappa
## 0.8041958 0.2669352
```

We can observe an accuracy of 80 percent in our Blood donation predictions, which represents a fairly good model.

The Kappa statistic compares an Observed Accuracy with an Expected Accuracy (random chance). The kappa statistic is used not only to evaluate a single classifier, but also to evaluate classifiers amongst themselves. It considers random chance (agreement with a random classifier), which generally means it is less misleading than simply using accuracy as a metric for model validation. Kappa Statistics ranges from 0 to 1 (the higher the better). In this case Kappa Statistic is 0.267, signifies a fair model.

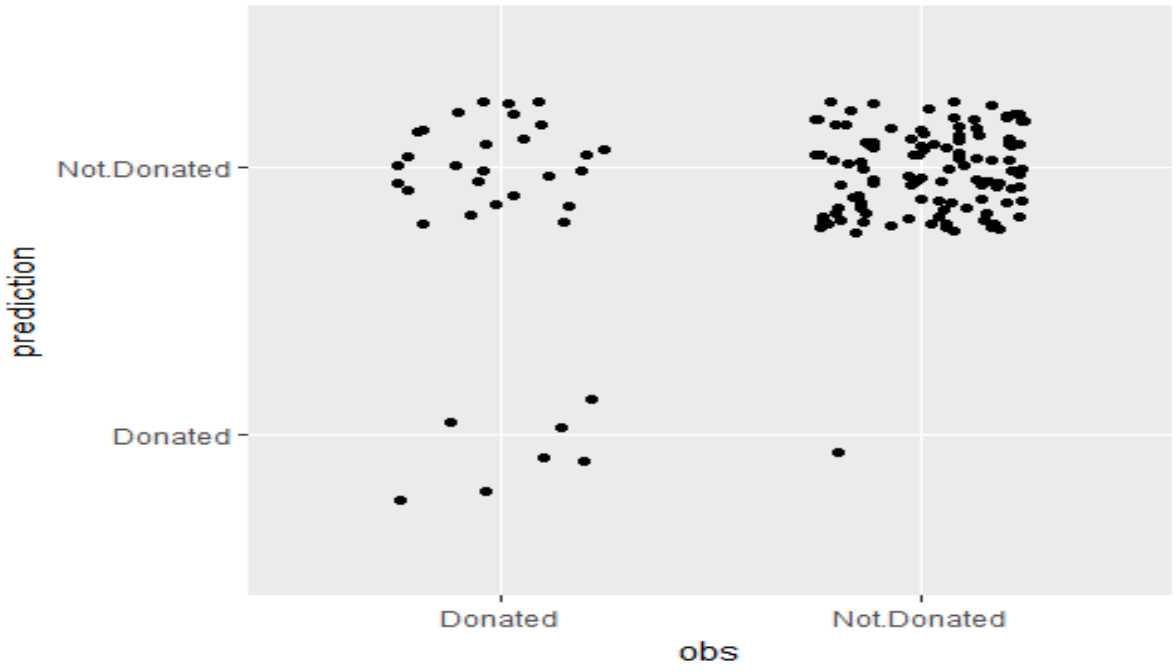
Following is the confusion matrix, which is a cross-tabulation of the observed and predicted values.

```
confusionMatrix(preds,test_data$md)

## Confusion Matrix and Statistics
##
##              Reference
## Prediction    Donated Not.Donated
##   Donated           7           1
##   Not.Donated      27         108
##
##                Accuracy : 0.8042
##                  95% CI : (0.7296, 0.8658)
##     No Information Rate : 0.7622
##     P-Value [Acc > NIR] : 0.1392
##
##                   Kappa : 0.2669
##  Mcnemar's Test P-Value : 2.306e-06
##
##             Sensitivity : 0.20588
##             Specificity : 0.99083
##          Pos Pred Value : 0.87500
##          Neg Pred Value : 0.80000
##              Prevalence : 0.23776
##          Detection Rate : 0.04895
##    Detection Prevalence : 0.05594
##       Balanced Accuracy : 0.59835
##
##        'Positive' Class : Donated
##
```

From the above we observe, 115 (108 + 7) correct predictions out of the 143 observations we predicted. We have a sensitivity (true positive rate) of 0.205, which indicates the probability that the model will predict a “Donated” result given that the result is true and a specificity (false positive rate) of 0.99, which indicates the probability that the model will predict a “Not.Donated” result given that the result is true.

Following is the graphical representation of the confusion matrix.

```
x <- data.frame(prediction=preds,obs=test_data$md)
table(x)

##              obs
## prediction    Donated Not.Donated
##   Donated           7           1
##   Not.Donated      27         108

ggplot(x, aes(x=obs,y=prediction))+geom_jitter(position =  position_jitter(width = 0.
25, height = 0.25))
```



(figure 7)

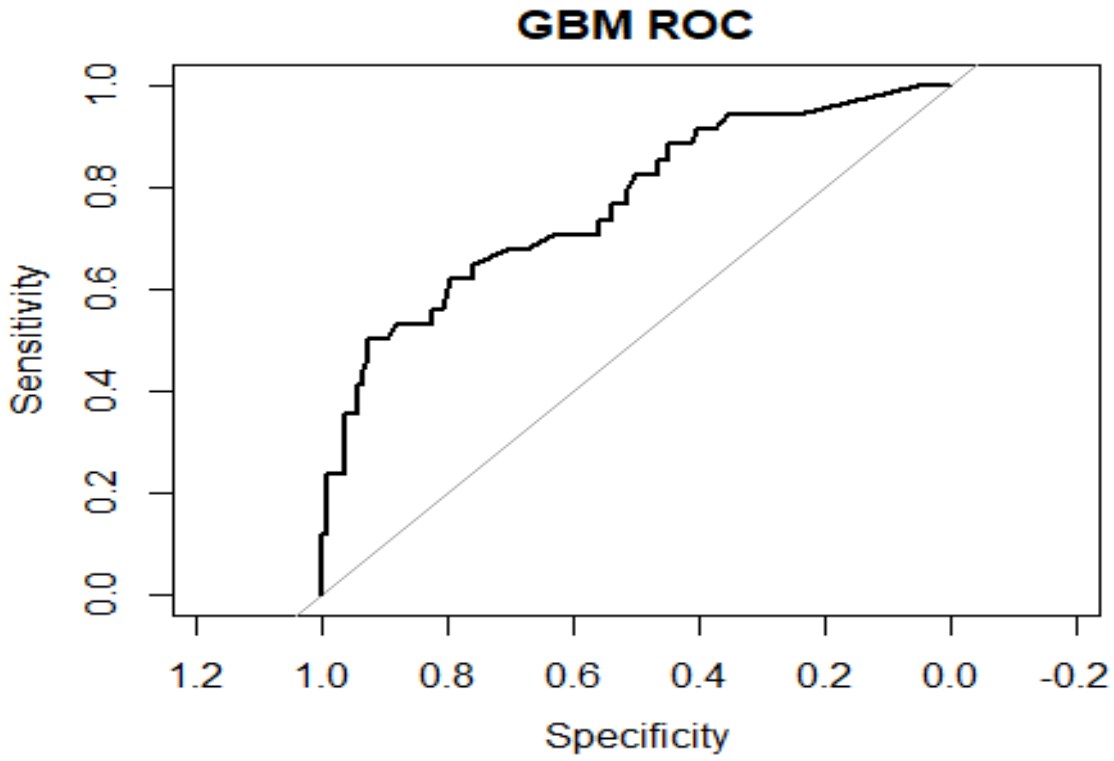
Confusion matrix of gbm model on test set

Our model suffers from poor sensitivity, as we have predicted correct only 7 out of the 34 donations made.

We use the Receiver Operating Characteristic (ROC) curves to characterize model performance. With two classes the Receiver Operating Characteristic (ROC) curve can be used to estimate performance using a combination of sensitivity and specificity.

The area under the ROC curve is a common metric of performance.

```
gbm.ROC <- roc(predictor=predprob$Donated,
               response=test_data$md,
               levels=rev(levels(test_data$md)))
gbm.ROC$auc
## Area under the curve: 0.7674
plot(gbm.ROC,main="GBM ROC")
```



(figure 8)

The ROC curve plots the pairs (sensitivity, 1-specificity) as the cutoff value increases from 0 and 1. For our model we have taken cut off value as 0.5 (which is given), so 1-specificity is 0.00917 and sensitivity is 0.20588. Better performance is reflected by curves that are closer to the top left corner. The comparison curve is the diagonal, which reflects the performance of the

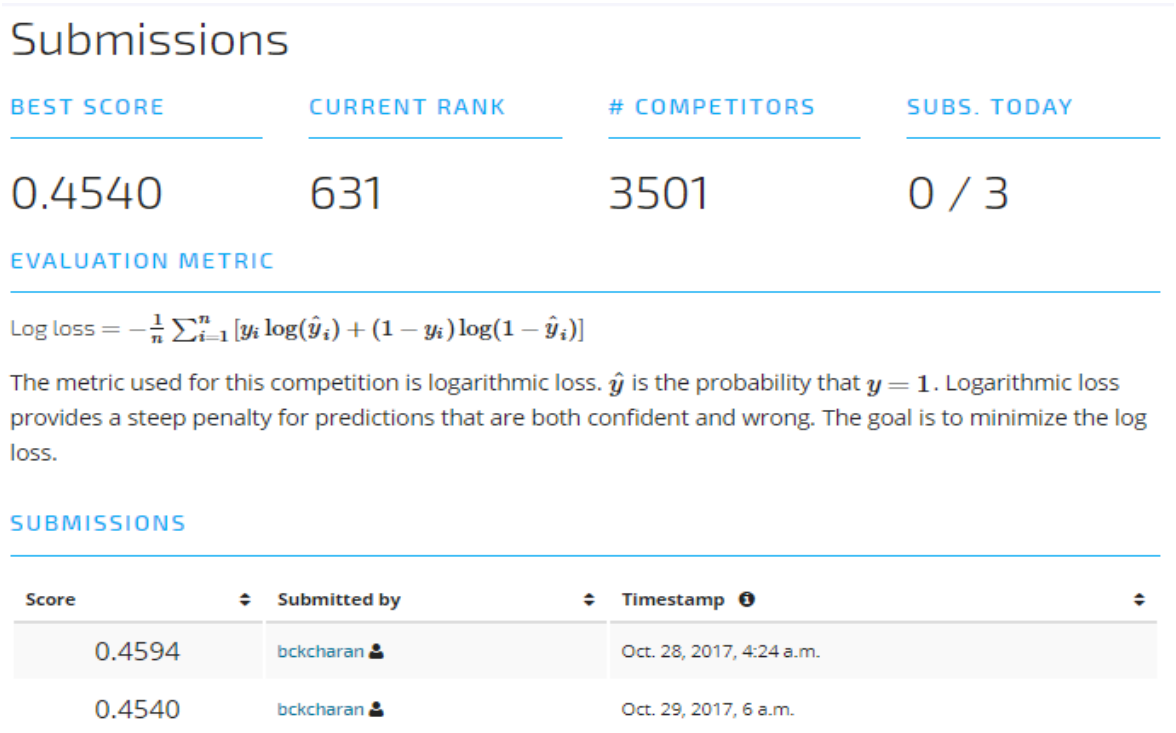
naïve rule, using varying cutoff values. The closer the curve follows the left-hand border and then the top border of the ROC space, the more accurate the test is. The closer the curve comes to the 45-degree diagonal of the ROC space, the less accurate the test is. So the ROC curve (using validation data) in figure 8 is not perfect, more variability can be explained for the model.

The accuracy of the model depends on how well the test separates the group being verified into those donate and do not donate blood. Accuracy is measured by the area under the ROC curve. An area of 1 represents a perfect test; an area of 0.5 represents a worthless test. An area of is 0.7674 under ROC curve signifies that model is good enough to classify who donates the blood and who does not.

We submitted the prediction using this model to drivendata.org for the following result.

Final Model:

```
formula <- md ~ log.mld + log.nod + log.mfd + log.avgd

dataCt <- trainControl(method='cv', number=5, returnResamp='none', summaryFunction = twoClassS
ummary, classProbs = TRUE)

newm <- train(formula, data = data, method = "gbm",
              trControl=dataCt,
              metric = "ROC",
              preProc = c("center", "scale"))

predprob <- predict(newm,original_test_data,type="prob")
predprob <- predprob['Donated']
```



Submissions

BEST SCORE	CURRENT RANK	# COMPETITORS	SUBS. TODAY
0.4540	631	3501	0 / 3

EVALUATION METRIC

Log loss $= -\frac{1}{n}\sum_{i=1}^{n}\left[y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)\right]$

The metric used for this competition is logarithmic loss. $\hat{y}$ is the probability that $y = 1$. Logarithmic loss provides a steep penalty for predictions that are both confident and wrong. The goal is to minimize the log loss.

SUBMISSIONS

Score	Submitted by	Timestamp
0.4594	bckcharan	Oct. 28, 2017, 4:24 a.m.
0.4540	bckcharan	Oct. 29, 2017, 6 a.m.

After calculating the accuracy rate of the glm model at .844 to allow us to compare the two different models submitted to drivendata, we clearly see that not only does the glm model do better in the competition, it performs better against the associated training sets.

For a recommendation to our customers, we would tell them that they could use the glm model to get approximately 80 percent accuracy on being able to predict the return rate of current blood donors based on the given predictors. If the company can start to track additional predictor variables (i.e. month specificity of return donors, age, gender, membership in health organizations, occupation, etc) a better model might be able to be developed. Also, the pattern associated with overall donor participation indicated that better promotional awareness for all potential blood collection dates would keep the donor participation at a higher level.