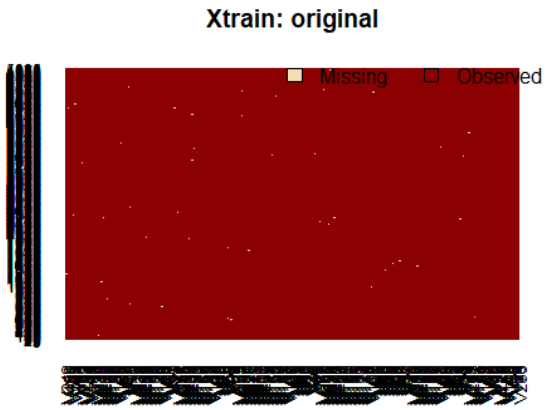
Exam 2: Fraud Detection

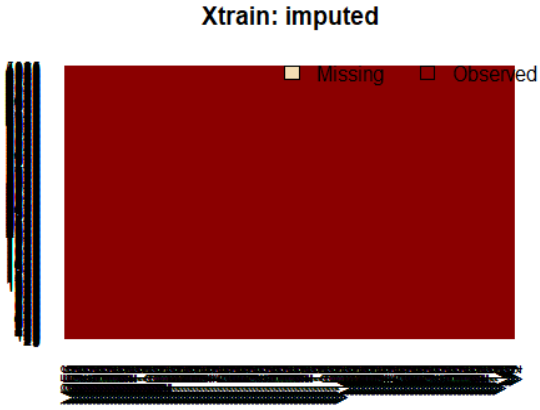
Saahil Deshpande

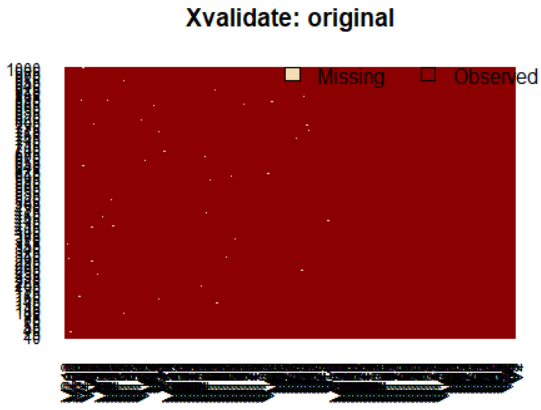
October 28, 2017

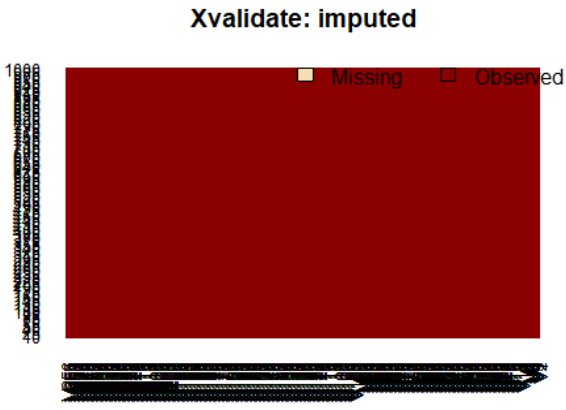
Missing Data:

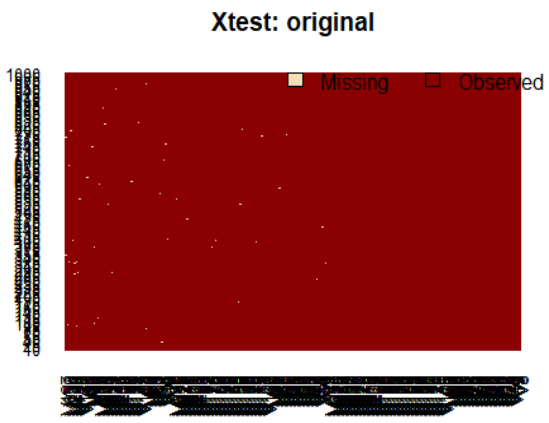
The Xtrain, Xvalidate and Xtest datasets have a few missing values which would create disputes when forming a model. To fix for these disputes, it is necessary to impute the values in place of these missing values. Since the frequency of missing values is not too large for any column or row, i did not exclude a feature or an observation. I imputed the mean of the columns for the numeric feature of each dataset and the mode of the column for the ordinal features of the dataset.

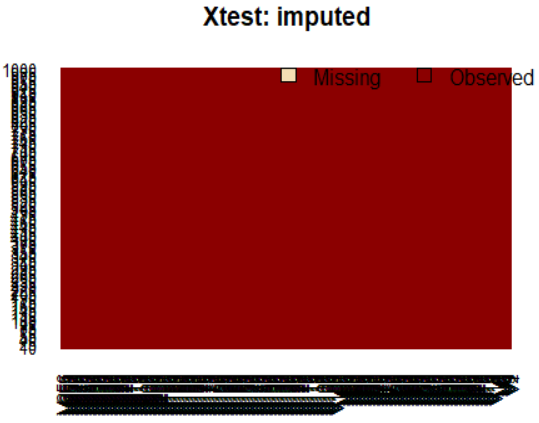












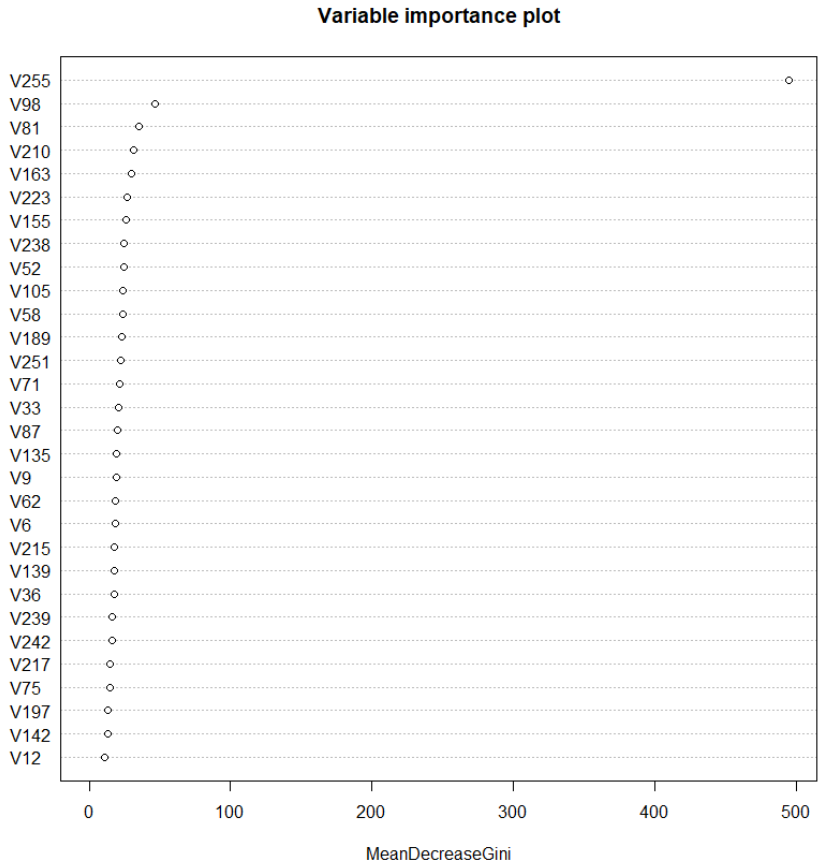
Modelling:

I used three methods to form models and chose the best model based on three metrics. I compared their estimated risk, misclassification rate on the validation set and the recall. The methods I used were random forest, bagging and logistic lasso. From the three methods, bagging provided the least estimate of risk and misclassification rate, and the maximum recall.

The bagging provides the out of bag error, which is similar to k-fold cross validation and can be used as an estimate for the risk. My model provided the out of bag error rate as 16.12%. I feel that the misclassification rate is not the best metric to decide what the best model is, since for an unbalance in the classes, the misclassification rate can be misleading. The recall would be a much better metric for our purpose. Hence, I would prefer model which would have a trade-off between the misclassification rate and the recall. A higher recall would indicate that the model has the capability to identify majority of the frauds while a lower misclassification rate at the same time would mean that the model isn't classifying too many of the not frauds as frauds.

To form such a model I repeated the procedure for various number of trees and various cut off values. The best result I obtained was for 500 trees with the cut off value at 0.485. The predictions with probabilities greater than 0.485 were classified as Fraud, while all other predictions were classified as Not Fraud.

The model includes all the 256 features. The important features can be observed in the variance importance plot.



V255 is strongly associated with the detection of fraud. Apart from that, V98, V81 and V210 are also associated with detecting a fraud. The importance of the features is based on the Gini index.

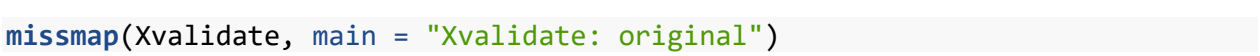
Using the validation set I found the following metrics for the model:

```
## [1] "miss-class"
## [1] 0.165
## [1] "Sensitivity"
## [1] 0.7777778
## [1] "Specificity"
## [1] 0.8602305
## [1] "Precision"
## [1] 0.7104478
## [1] "Recall"
## [1] 0.7777778
## [1] "F1 score"
## [1] 0.7425897
## [1] "confusion mat"
##            true.class
## pred.class  fraud not fraud
##   fraud       238        97
##   not fraud    68       597
```

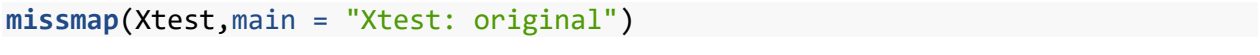
The model as a misclassification rate of 16.5% and recall of 78%. This means that 78% of the time the model will identify a Fraud and also it does not misclassify too many of the Not Fraud as Fraud.

Appendix:

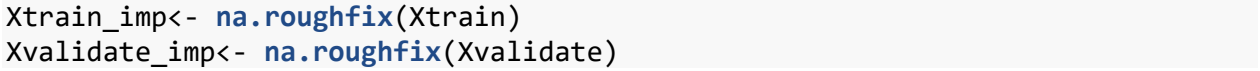
```
yourName = 'SaahilDeshpande'#fill in your name, no spaces, leave quotes
load('C:/Users/saahi/OneDrive/Documents/Stat 6306/exam 2/Ytrain.Rdata')
load('C:/Users/saahi/OneDrive/Documents/Stat 6306/exam 2/Xtrain.Rdata')
load('C:/Users/saahi/OneDrive/Documents/Stat 6306/exam 2/Xtest.Rdata')
load('C:/Users/saahi/OneDrive/Documents/Stat 6306/exam 2/Xvalidate.Rdata')
load('C:/Users/saahi/OneDrive/Documents/Stat 6306/exam 2/Yvalidate.Rdata')

Ytrain<- factor(Ytrain)
Yvalidate<- factor(Yvalidate)

require(Amelia)

## Loading required package: Amelia

## Warning: package 'Amelia' was built under R version 3.4.2

## Loading required package: Rcpp

## ##
## ## Amelia II: Multiple Imputation
## ## (Version 1.7.4, built: 2015-12-05)
## ## Copyright (C) 2005-2017 James Honaker, Gary King and Matthew Blackwell
## ## Refer to http://gking.harvard.edu/amelia/ for more information
## ##

require(randomForest)

## Loading required package: randomForest

## Warning: package 'randomForest' was built under R version 3.4.2

## randomForest 4.6-12

## Type rfNews() to see new features/changes/bug fixes.

missmap(Xtrain, main = "Xtrain: original")
```



```
missmap(Xvalidate, main = "Xvalidate: original")
```



```
missmap(Xtest,main = "Xtest: original")
```



```
Xtrain_imp<- na.roughfix(Xtrain)
Xvalidate_imp<- na.roughfix(Xvalidate)
```

```r
Xtest_imp<- na.roughfix(Xtest)
missmap(Xtrain_imp,main = "Xtrain: imputed")
```

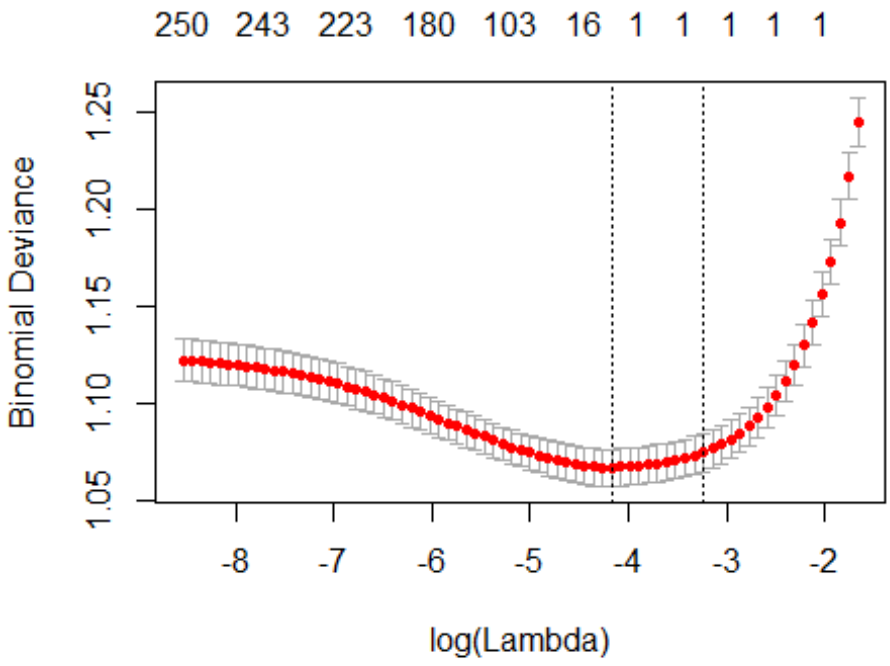
```r
missmap(Xvalidate_imp, main = "Xvalidate: imputed")
```

```r
missmap(Xtest_imp, main = "Xtest: imputed")
```

```r
## This code is taken from Homework 3 and Homework 4 for STAT 6306 and modifi
ed
misClass =function(pred.class,true.class,produceOutput=FALSE){
  confusion.mat = table(pred.class,true.class)
  if(produceOutput){
    return(1-sum(diag(confusion.mat))/sum(confusion.mat))
  }
  else{
    print('miss-class')
    print(1-sum(diag(confusion.mat))/sum(confusion.mat))
    print('Sensitivity')
    print(confusion.mat[1,1]/sum(confusion.mat[,1]))
    print('Specificity')
    print(confusion.mat[2,2]/sum(confusion.mat[,2]))
    print('Precision')
    print(confusion.mat[1,1]/sum(confusion.mat[1,]))
    print('Recall')
    print(confusion.mat[1,1]/sum(confusion.mat[,1]))
    print('F1 score')
    print(2*(confusion.mat[1,1]/sum(confusion.mat[1,])*confusion.mat[1,1]/sum
(confusion.mat[,1]))/(confusion.mat[1,1]/sum(confusion.mat[1,])+confusion.mat
[1,1]/sum(confusion.mat[,1])))
    print('confusion mat')
    print(confusion.mat)
  }
}

out.rf<- randomForest(Xtrain_imp,Ytrain)
class.rf<-predict(out.rf,Xvalidate_imp)
out.bag<- randomForest(Xtrain_imp,Ytrain, importance = T, mtry = ncol(Xtrain)
,cutoff = c(0.486,1-0.486))
class.bag<- predict(out.bag,Xvalidate_imp)

varImpPlot(out.bag,type = 2, main = "Variable importance plot")
```

```r
xtrain<- Xtrain_imp
xvalidate<- Xvalidate_imp
```

```
xtrain$V255<- as.numeric(xtrain$V255)
xtrain$V256<- as.numeric(xtrain$V256)
xvalidate$V255<- as.numeric(xvalidate$V255)
xvalidate$V256<- as.numeric(xvalidate$V256)

require(glmnet)

## Loading required package: glmnet

## Warning: package 'glmnet' was built under R version 3.4.1

## Loading required package: Matrix

## Warning: package 'Matrix' was built under R version 3.4.1

## Loading required package: foreach

## Warning: package 'foreach' was built under R version 3.4.1

## Loaded glmnet 2.0-12

Xmat = as.matrix(xtrain,dimnames = NULL)
Ynum = as.numeric(Ytrain)-1
lasso.cv.glmnet = cv.glmnet(Xmat,Ynum,alpha=1,family='binomial',nfolds = 10)
sum.out<- summary(lasso.cv.glmnet)

plot(lasso.cv.glmnet)
```



```r
betaHat.lasso<-coef(lasso.cv.glmnet,s='lambda.min')[-1]
S.lasso<- which(abs(betaHat.lasso)> 1e-16)

Xmat_0 = as.matrix(xvalidate,dimnames = NULL)
Ynum_0 = as.numeric(Yvalidate)-1
Yhat.lasso   = predict(lasso.cv.glmnet, Xmat_0,
                    s='lambda.min',type='class')

misClass(class.rf,Yvalidate)
```

```
## [1] "miss-class"
## [1] 0.257
## [1] "Sensitivity"
## [1] 0.1699346
## [1] "Specificity"
## [1] 0.9956772
## [1] "Precision"
## [1] 0.9454545
## [1] "Recall"
## [1] 0.1699346
## [1] "F1 score"
## [1] 0.2880886
## [1] "confusion mat"
##            true.class
## pred.class  fraud not fraud
##   fraud        52         3
##   not fraud   254       691
```

```r
misClass(class.bag, Yvalidate)
```

```
## [1] "miss-class"
## [1] 0.165
## [1] "Sensitivity"
## [1] 0.7777778
## [1] "Specificity"
## [1] 0.8602305
## [1] "Precision"
## [1] 0.7104478
## [1] "Recall"
## [1] 0.7777778
## [1] "F1 score"
## [1] 0.7425897
## [1] "confusion mat"
##            true.class
## pred.class  fraud not fraud
##   fraud       238        97
##   not fraud    68       597
```

```r
misClass(Yhat.lasso,Ynum_0)
```

```
## [1] "miss-class"
## [1] 0.344
## [1] "Sensitivity"
## [1] 0.4640523
## [1] "Specificity"
## [1] 0.740634
## [1] "Precision"
## [1] 0.4409938
## [1] "Recall"
## [1] 0.4640523
## [1] "F1 score"
## [1] 0.4522293
## [1] "confusion mat"
##           true.class
## pred.class   0   1
##          0 142 180
##          1 164 514
```

```
Ypred<- predict(out.bag,Xtest_imp)
### get preds:
Yhat = data.frame('Yhat' = Ypred)
#write.table
if(yourName == 'firstLast'){
  print('fill in your name!')
}else{
  fName = paste(c(yourName,'_Predictions.txt'),collapse='')
  write.table(Yhat,file=fName,row.names=FALSE,col.names=FALSE)
}
```