BAC Insight Team

Team 2 - Fraud Detection

Tanmay Gupta - Larry Langman
Riya Mokashi - Samarth Baral

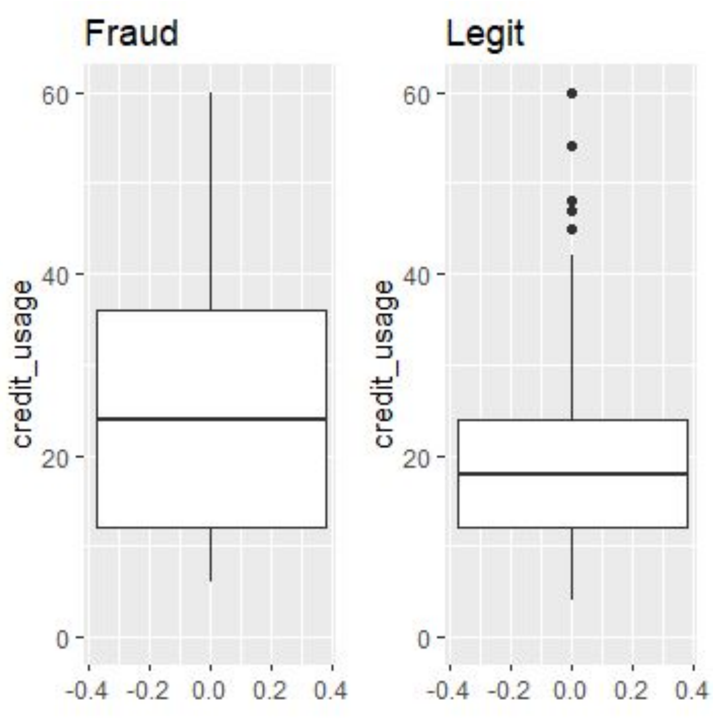
Problem Statement

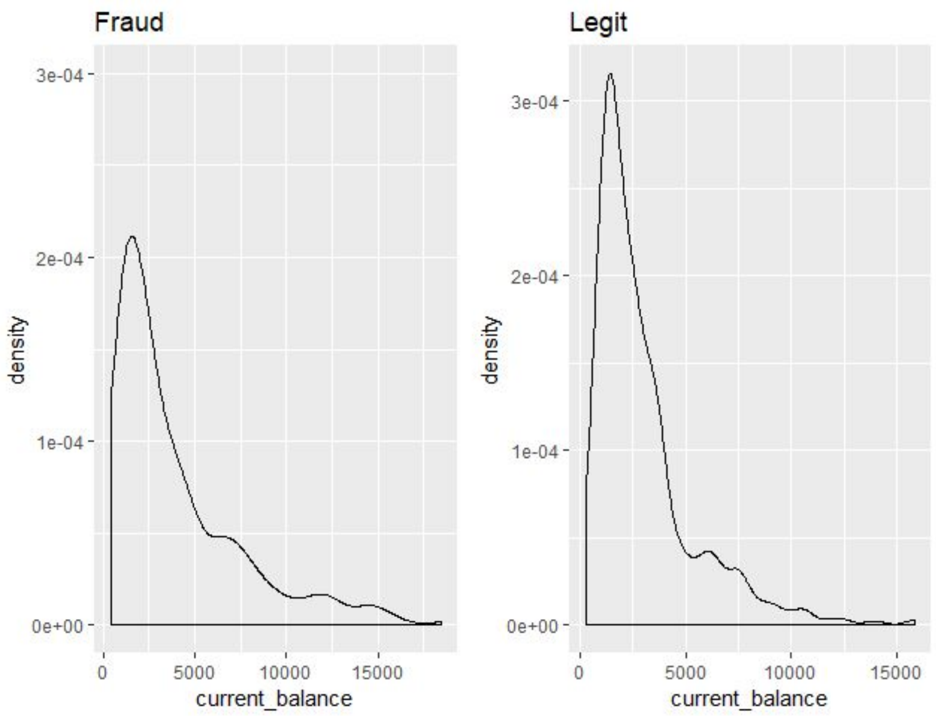
- There is a hidden risk that business face that often goes unnoticed until they tear up the company's finances - Employee Theft
- Costs US companies around $50 Billion
- An important part of that theft comes in the form of Credit Card Fraud - using company funds for personal expenses
- To create an internal analytics tool for a business to classify a financial transaction as fraudulent or legitimate

Data

- Simulated data on German Credit Card Transactions
- Each transaction was labelled as Legit or Fraud
- Have 1000 transactions - 700 legit and 300 fraud
- Have 21 predictors for the data

Data Exploration





Logistic Regression Model

Definition (from google): In statistics, the logistic model is a widely used statistical model that, in its basic form, uses a logistic function to model a binary dependent variable

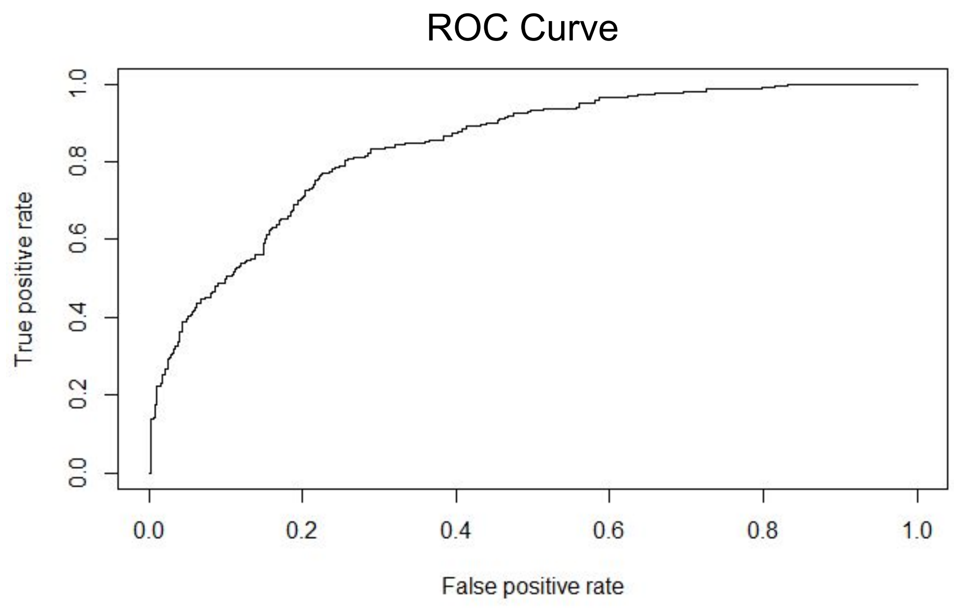
In this case, key variables were: Current Balance, Credit Usage, Age (of borrower), and location (one of 4 zones the person is from).

3 Steps:

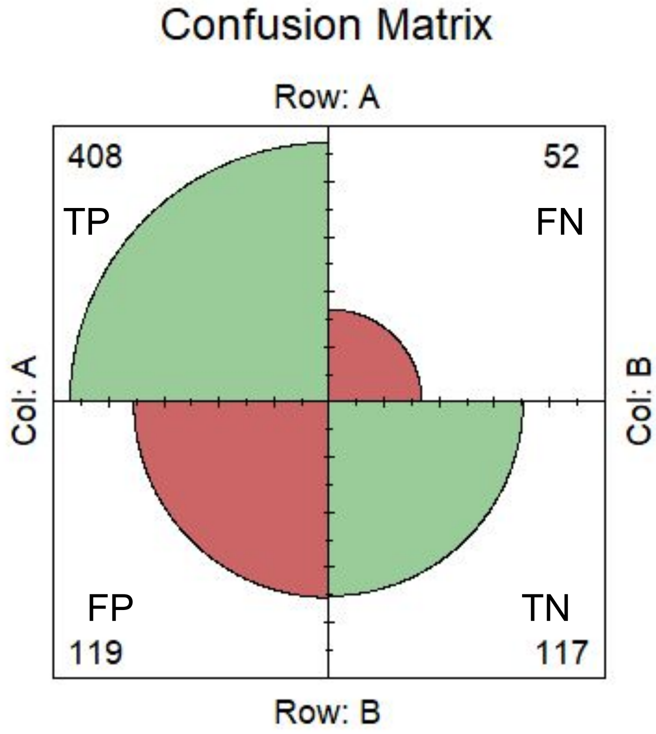
1. Determine key variables
2. Determine an accuracy cutoff point
3. Create a confusion matrix to display the model’s accuracy

Logistic Regression Model

77.2% Accurate

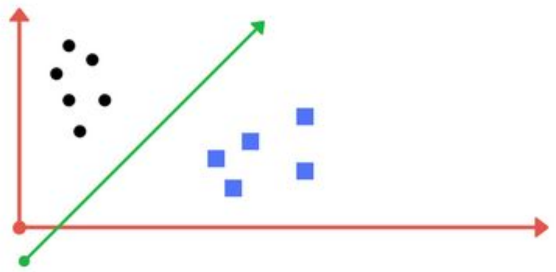



```
sensitivity 0.7713004
specificity 0.7741935
cutoff      0.3176704
```



Support Vector Machine

Definition: A Support Vector Machine is a discriminative classifier formally defined by a a separating hyperplane. In simple words, an SVM creates a hyperplane (or simply a line) that separates classes. In order to make an SVM work, one needs to pass through a set of training data, with each data point belonging in one of the categories; the SVM will then classify the data points accordingly.



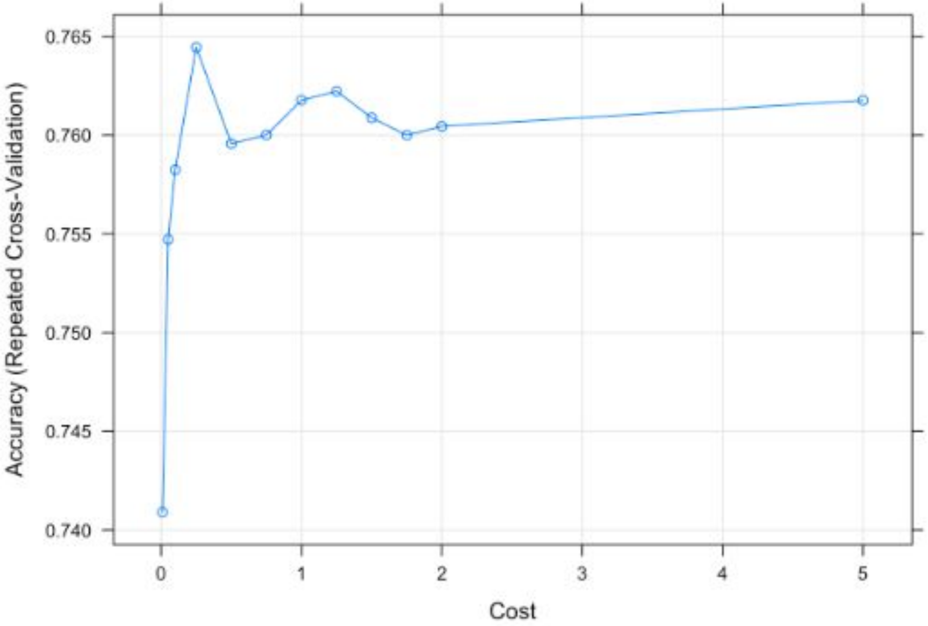
Patel, Savan. “Sample cut to divide into two classes” *Medium*, May 3, 2017

SVM Sample Code and Explanation

```
inTrain<- createDataPartition(y=data$class,p=0.75, list=FALSE)

train_set <- data[inTrain,]
test_set <- data[-inTrain,]

grid <- expand.grid(C = c(0.01, 0.05, 0.1, 0.25, 0.5, 0.75, 1, 1.25, 1.5, 1.75, 2,5))
trctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 3)

set.seed(3233)
svm_Linear <- train(class ~., data = train_set, method = "svmLinear",
                    trControl=trctrl,
                    preProcess = c("center", "scale"),
                    tuneGrid = grid,
                    tuneLength = 10)
plot(svm_Linear)
```

SVM Sample Code and Confusion Matrix

Graph to calculate the optimal cost that will result in the highest accuracy



76% Accuracy

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 148  33
         1  27  42

               Accuracy : 0.76
                 95% CI : (0.7021, 0.8116)
    No Information Rate : 0.7
    P-Value [Acc > NIR] : 0.02104
```

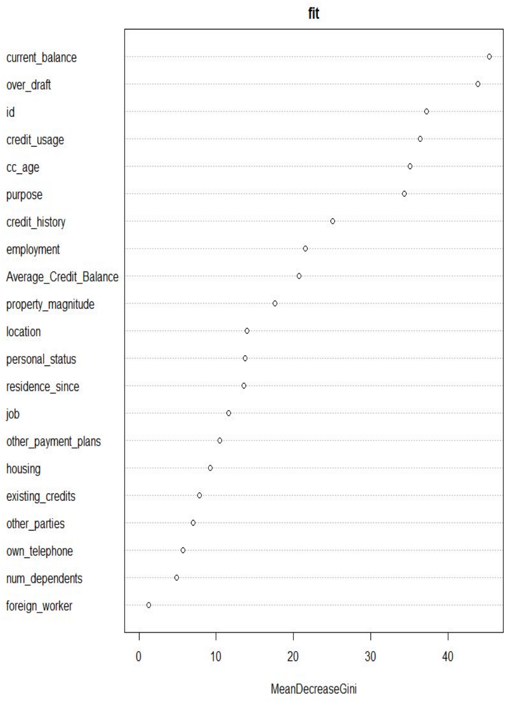
Decision Tree Model

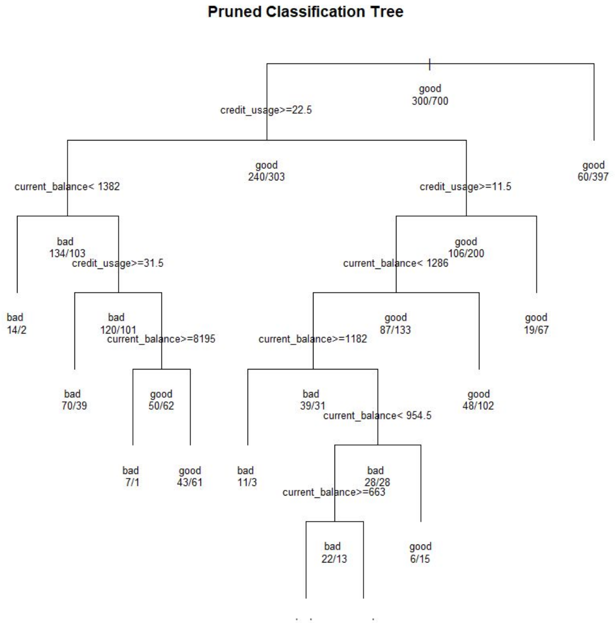
A sequence of branching operations based on comparisons of some quantities, the comparisons being assigned the unit computational cost

```
pred <- ifelse(pred<0.5, 0, 1)

confusionMatrix(pred, FraudData$classnum)
```

```
##       [,1] [,2]
## [1,]   597  103
## [2,]   166  134
```

73% Success Rate





Random Forest Model

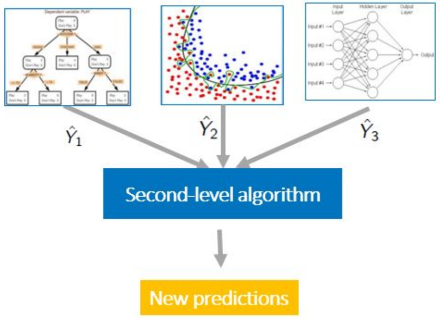
```
library(caret)
library(randomForest)
train_set <- sample(1:nrow(FraudData), 750)
train<-FraudData[train_set,]
test<-FraudData[-train_set,]

RandForest=randomForest(class ~., data=train, ntree=5, mtry=6, importance=TRUE)
RandForest

output <- predict(RandForest, newdata = test)
confusionMatrix(output, test$class)
```

```
> caret::confusionMatrix(output, test$class)
Confusion Matrix and Statistics

          Reference
Prediction bad good
      bad   75    0
      good   0  175

               Accuracy : 1
                 95% CI : (0.9854, 1)
    No Information Rate : 0.7
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 1
 Mcnemar's Test P-Value : NA

            Sensitivity : 1.0
            Specificity : 1.0
         Pos Pred Value : 1.0
         Neg Pred Value : 1.0
             Prevalence : 0.3
         Detection Rate : 0.3
   Detection Prevalence : 0.3
      Balanced Accuracy : 1.0

       'Positive' Class : bad
```

Virtually 100% Success Rate

Going Forward

- Choosing the Appropriate Model
- Creating an Ensemble Model



- Working towards prevention over detection

Questions?