Coursera – Data Science at Scale – Practical Predictive Analytics: Models and Methods

Kaggle Assignment – Give Me Some Credit

Andre Johnson

Part 1: Problem Description: This paper will cover a Kaggle competition: "Give Me Some Credit" with the following description.

Improve on the state of the art in credit scoring by predicting the probability that somebody will experience financial distress in the next two years.

Banks play a crucial role in market economies. They decide who can get finance and on what terms and can make or break investment decisions. For markets and society to function, individuals and companies need access to credit.

Credit scoring algorithms, which make a guess at the probability of default, are the method banks use to determine whether or not a loan should be granted. This competition requires participants to improve on the state of the art in credit scoring, by predicting the probability that somebody will experience financial distress in the next two years.

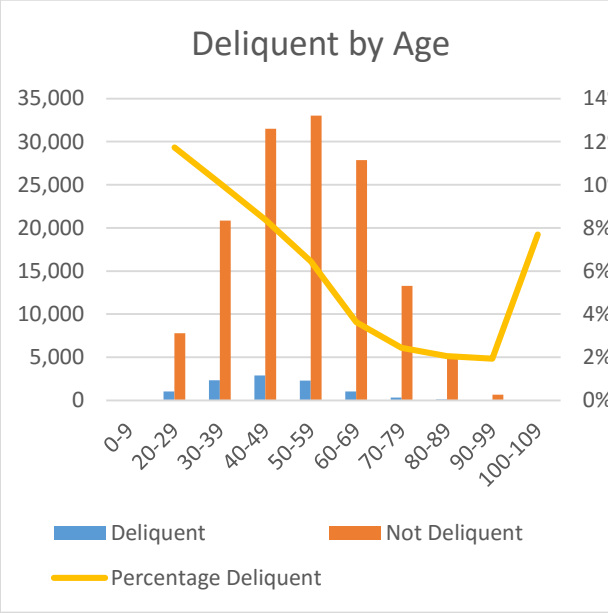
The goal of this competition is to build a model that borrowers can use to help make the best financial decisions.

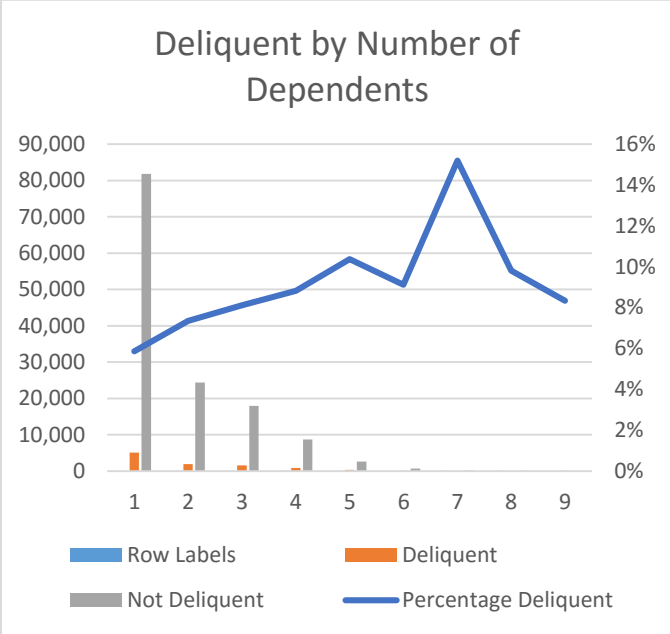
- **Data provided:**
  - cs-training set – 150,000 rows, 12 columns of client data
  - cs-test set - 101,503 rows, 12 columns of client data
  - Data Dictionary:

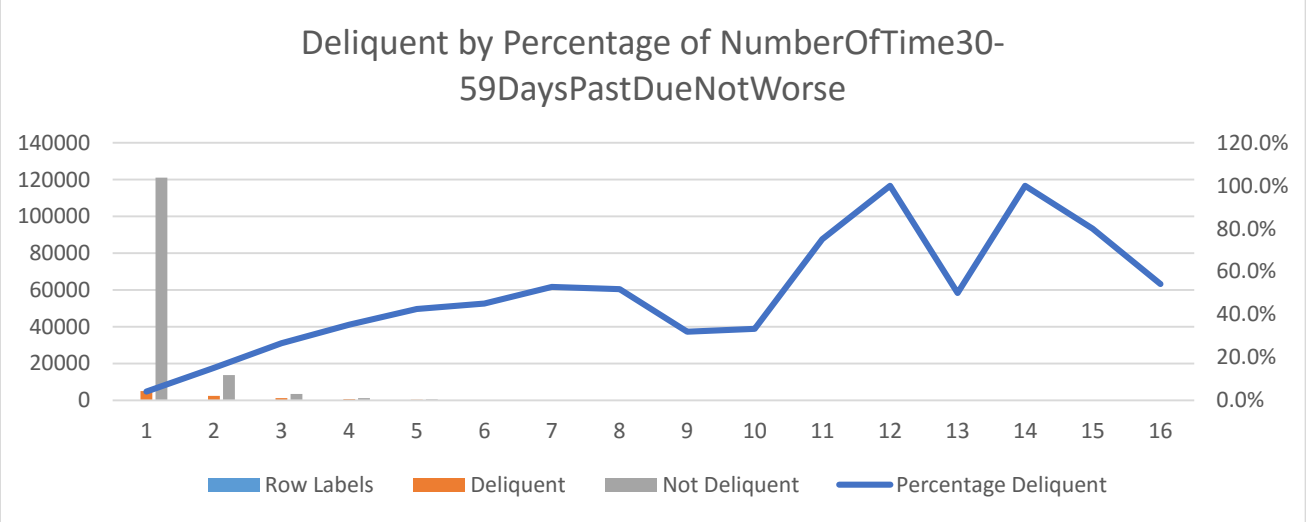
Variable Name	Description	Type
SeriousDlqin2yrs	Person experienced 90 days past due delinquency or worse	Y/N
RevolvingUtilizationOfUnsecuredLines	Total balance on credit cards and personal lines of credit except real estate and no installment debt like car loans divided by the sum of credit limits	percentage
age	Age of borrower in years	integer
NumberOfTime30-59DaysPastDueNotWorse	Number of times borrower has been 30-59 days past due but no worse in the last 2 years.	integer
DebtRatio	Monthly debt payments, alimony,living costs divided by monthy gross income	percentage
MonthlyIncome	Monthly income	real
NumberOfOpenCreditLinesAndLoans	Number of Open loans (installment like car loan or mortgage) and Lines of credit (e.g. credit cards)	integer
NumberOfTimes90DaysLate	Number of times borrower has been 90 days or more past due.	integer
NumberRealEstateLoansOrLines	Number of mortgage and real estate loans including home equity lines of credit	integer
NumberOfTime60-89DaysPastDueNotWorse	Number of times borrower has been 60-89 days past due but no worse in the last 2 years.	integer
NumberOfDependents	Number of dependents in family excluding themselves (spouse, children etc.)	integer

Part 2: Analysis Approach

- **Step 1:** In step 1, after reviewing and cleaning up the data (ie. removing NAs, etc.), I started by visualizing the data to see if there were any apparent trends whether an individual would be "Serious Deliquent" using Excel by plotting "Serious Deliquent" vs other factors before moving on to developing a model for forecasting.







By charting the delinquencies by Age and Number of Dependents clear trend lines started to emerge. The number of delinquencies by Age clearly decreased as age went up (with only an

increase after the age of 100). The number of delinquencies also have a clear increase with the number of dependents. The final plot shows what would be an expected increase in “Serious Deliquent” status as NumberOfTime30-59DaysPastDueNotWorse goes up.

- **Part 3:** Initial Solution: I used R to develop a few models for forecasting severe delinquency rates:
    - **Model 1 –** rpart

```
fol <- formula(SeriousDlqin2yrs ~ RevolvingUtilizationOfUnsecuredLines + age+
NumberOfTime30.59DaysPastDueNotWorse + DebtRatio + MonthlyIncome +
NumberOfOpenCreditLinesAndLoans + NumberOfTimes90DaysLate + NumberRealEstateLoansOrLines +
NumberOfTime60.89DaysPastDueNotWorse + NumberOfDependents)
credit_sample <- credit_noNA[sample(nrow(credit_noNA), size = 2000, replace = FALSE, prob = NULL),]
inTrain <- createDataPartition(y=credit_sample$SeriousDlqin2yrs, p = .7, list = FALSE)
credit_training <- credit_sample[inTrain,]
credit_testing <- credit_sample[-inTrain,]
model <- rpart(fol, method="class", data=credit_training)print(model)
```

```
1) root 1400 104 0 (0.92571429 0.07428571)
   2) NumberOfTimes90DaysLate< 0.5 1323  71 0 (0.94633409 0.05366591) *
   3) NumberOfTimes90DaysLate>=0.5 77  33 0 (0.57142857 0.42857143)
     6) RevolvingUtilizationOfUnsecuredLines< 0.9614853 40  10 0 (0.75000000 0.25000000) *
     7) RevolvingUtilizationOfUnsecuredLines>=0.9614853 37  14 1 (0.37837838 0.62162162)
       14) DebtRatio< 0.3032321 26  13 0 (0.50000000 0.50000000)
         28) NumberOfTimes90DaysLate< 2.5 16   5 0 (0.68750000 0.31250000) *
         29) NumberOfTimes90DaysLate>=2.5 10   2 1 (0.20000000 0.80000000) *
       15) DebtRatio>=0.3032321 11   1 1 (0.09090909 0.90909091) *
```

- The result from the first Model was a tree that used NumberOfTimes90DaysLate as the primary predictor of delinquency (which would make sense) and drilling down from there.
- Validating against the test set resulted in an Accuracy Rate of 94.17%, which is a reasonable start.

```
credit_predict <- predict(model, credit_testing, type="class")
result = credit_testing$SeriousDlqin2yrs==credit_predict
modelAccuracy = mean(result) *100
modelAccuracy
```

```
94.16667
```

- Next I ran a confusion matrix against the rpart model (Model 1).

```
➢ confusionMatrix(credit_testing$Serious
  Dlqin2yrs, credit_predict)
➢
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 560   2
         1  33   5

               Accuracy : 0.9417
                 95% CI : (0.9198, 0.959)
    No Information Rate : 0.9883
    P-Value [Acc > NIR] : 1
```

```
Kappa : 0.2066
 Mcnemar's Test P-Value : 3.959e-07

            Sensitivity : 0.9444
            Specificity : 0.7143
         Pos Pred Value : 0.9964
         Neg Pred Value : 0.1316
             Prevalence : 0.9883
         Detection Rate : 0.9333
   Detection Prevalence : 0.9367
      Balanced Accuracy : 0.8293

       'Positive' Class : 0
```

- **Model 2** – Random Forest

```
➢ RFcontrol <- trainControl(method="cv", number=5)
➢ RFmodel = train(SeriousDlqin2yrs ~ ., method="rf", data=credit_training, trControl=RFcontrol, ntree =
  250)
➢ print(RFmodel)

Random Forest

1400 samples
  11 predictor

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 1120, 1120, 1120, 1120, 1120
Resampling results across tuning parameters:

  mtry  RMSE       Rsquared   RMSE SD     Rsquared SD
   2    0.2392691  0.1727529  0.02084061  0.06637627
   6    0.2418543  0.1665969  0.02037149  0.06753662
  11    0.2449968  0.1548318  0.02121902  0.06780106

RMSE was used to select the optimal model using  the smallest value.
The final value used for the model was mtry = 2.
```

- The resulting model was for a model mtry = 2 with minimized RMSE.
- Validating against the test set resulted in an Accuracy Rate of 94.17% when using .5 as the factor to decide 0 or 1. The Accuracy level for this model was similar to the rpart model (Model 1).

```
➢ credit_RFpredict <- predict(RFmodel, credit_testing)
➢ credit_RFModel_Factor <- function(x){ if(x < .5){return(0)} else return(1)}
➢ credit_RFpredict <- sapply(credit_RFpredict, credit_RFModel_Factor)

➢ 94.16667
```

- Next I ran a confusion matrix against the Random Forest model (Model 2).

```
➢ confusionMatrix(credit_testing$Serious
  Dlqin2yrs, credit_RFpredict)

Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 560   2
         1  33   5

               Accuracy : 0.9417
                 95% CI : (0.9198, 0.959)
    No Information Rate : 0.9883
    P-Value [Acc > NIR] : 1
```

```
Kappa : 0.2066
 Mcnemar's Test P-Value : 3.959e-07

            Sensitivity : 0.9444
            Specificity : 0.7143
         Pos Pred Value : 0.9964
         Neg Pred Value : 0.1316
             Prevalence : 0.9883
         Detection Rate : 0.9333
   Detection Prevalence : 0.9367
      Balanced Accuracy : 0.8293

       'Positive' Class : 0
```

Model 3 – Support Vector Machines

- SVMmodel <- svm(fol, data=credit_training)
- print(SVMmodel)

```
Call:
svm(formula = fol, data = credit_training)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1
      gamma:  0.1

Number of Support Vectors:  10622
```

- Validating against the test set resulted in an Accuracy Rate of 93.7% when using .5 as the factor to decide 0 or 1. The Accuracy level for this model was similar to a model simply predicting all as “Not Serious Deliquent”, in other words a poor forecasting model.

```
➢ credit_SVMpredict <- predict(SVMmodel, credit_testing)
➢ modelAccuracy = mean(credit_SVMpredict == credit_testing$SeriousDlqin2yrs) *1
  00
➢ modelAccuracy
➢
➢ 93.66667
```

- **Part 4: Initial Solution Analysis:**
    - With each of the models the accuracy on the testing sets were over 93% and over 94% for the rpart and Random Forest models which may normally be considered high, however given 93.7% of the testing data set was “Not Serious Deliquent” (0), a model that simply guessed all predictions were “Not Serious Deliquent” (0) would have produced a 93.7% accuracy rate. Hence, the models produced were only incremental improvement to that approach.
    - What becomes clear from the confusion matrices is that the models produced a high percentage of false positives that predicted “Serious Deliquent” (1) when in fact not “Serious Deliquent” (0).
    - Both rpart model and the Random Forest model produced the same results in terms of accuracy and also in terms of false predictions.

- **Part 5: Revised Solution and Analysis:**
    - For Model 2 and 3, the Random Forest and SVM models produces a probability rather than a class of 0 or 1. Initially, the criteria for 0 or 1 was set at x >= .5 = 1 and x < .5 = 0. Adjusting the criteria from up or down from .5 resulted in different results but ultimately did not show significant improvement in the accuracy.
    - Revisiting the data, a few things did stand out for review.
        1. Further cleaning up the data:
            - The levels of RevolvingUtilizationOfUnsecuredLines had a median of 0.17 but a max of 7096, so the vast majority under 1. Further investigation into whether this is inaccurate data or an outlier is warranted.
            - Similarly, DebtRatio had a median of .30 but a maximum of 4483. Again, an investigation on whether this is accurate or there are outliers is warranted.
            - Rerunning the models with outliers removed from RevolvingUtilizationOfUnsecuredLines (<1000) and DebtRatio (<20), the prediction accuracy changed: rpart (from 94.17% to 93.33%), Random Forest (from 94.17% to 94.5%), and SVM (from 93.7% to 93.8%)
        2. Putting various factors into Bins:
            - Cleaning up the data further, I put certain factors in Bins to streamline the analysis in the following formula only keeping NumberOfTimes90DaysLate and NumberOfTime60.89DaysPastDueNotWorse as is.

```
fol <- formula(SeriousDlqin2yrs ~ NumberOfTimes90DaysLate + NumberOfTime60.89Days
PastDueNotWorse + NumberOfTime30.59DaysPastDueNotWorseBins + DebtRatioBins + Mont
hlyIncomeBins + NumberOfOpenCreditLinesAndLoansBins + NumberRealEstateLoansOrLine
sBins + NumberOfDependentsBins)
```

        - Rerunning the models against the as-is and “binned” factors the prediction accuracy changed: rpart (from 94.17% to 94%), Random Forest (from 94.17% to 94%), but **SVM (from 93.7% to 95%)** a significant improvement for the SVM model.
- Opportunities for further improvement:
    1. Further clean-up of the data through eliminating outliers and “binning” showed improvements in the models however the improvements were incremental. The reality is that the selection of criteria for “outliers” and for “binning” as well as the criteria under the Random Forest and SVM models for selecting probability level for selecting a 0 or 1 (originally 0.5%) were educated guesses but somewhat arbitrary. Running further analysis with a variety of different levels of these factors could streamline and enhance the ability to develop a model with a higher level of predictability.
    2. The modelling in this exercise was fairly limited in scope by focusing on 3 models only: rpart, Random Forest, and Support Vector Machines. Future attempts should consider including other models such as K-nearest, other ensemble decision tree models such as SPRINT, ID3, etc. and some neural network models.

Conclusion:

The data provided and the variety of factors included appear to give a good start for developing a forecasting model 1) via visualization we can see the correlation of factors to “serious delinquent” status 2) the models developed give some improvement over a model which simply predicts “Not Serious Delinquent”. Nevertheless, the best model developed, Support Vector Machine, after binning certain factors with 95% correct predictions on the test set is still far from perfect. As discussed above, further enhancements on the models and running other forms of models should be considered. Running various scenarios, with a variety of bins for certain factors, optimizing the selection criteria for Random Forest model (ie. less than or more than 50% benchmark), etc. would make sense for optimizing the predictability of the ultimate model. While more work to be done, the analysis was fruitful and a great start.

Appendix – R Code for Analysis

```
library(caret)
library(ggplot2)
library(rpart)
library(randomForest)
library(kernlab)
library(e1071)

setwd("C:/Users/…. /")
credit <- read.csv("cs-training.csv")
head(credit)
sapply(credit, typeof)

credit_noNA <- subset(credit, !is.na(credit$NumberOfDependents))
credit_noNA <- subset(credit, !is.na(credit$MonthlyIncome))

credit_sample <- credit_noNA[sample(nrow(credit_noNA), size = 2000, replace = FALSE, prob = NULL),]
inTrain <- createDataPartition(y=credit_sample$SeriousDlqin2yrs, p = .7, list = FALSE)
credit_training <- credit_sample[inTrain,]
credit_testing <- credit_sample[-inTrain,]

fol <- formula(SeriousDlqin2yrs ~ RevolvingUtilizationOfUnsecuredLines + age+ NumberOfTime30.59DaysPastDueNotWorse
+ DebtRatio + MonthlyIncome + NumberOfOpenCreditLinesAndLoans + NumberOfTimes90DaysLate +
NumberRealEstateLoansOrLines + NumberOfTime60.89DaysPastDueNotWorse + NumberOfDependents)
model <- rpart(fol, method="class", data=credit_training)
print(model)
credit_predict <- predict(model, credit_testing, type="class")
result = credit_testing$SeriousDlqin2yrs==credit_predict
modelAccuracy = mean(result) *100
modelAccuracy

confusionMatrix(credit_testing$SeriousDlqin2yrs, credit_predict)


RFcontrol <- trainControl(method="cv", number=5)
RFmodel = train(SeriousDlqin2yrs ~ ., method="rf", data=credit_training, trControl=RFcontrol, ntree = 250)
print(RFmodel)
saveRDS(RFmodel, "rfmodel.RDS")

credit_RFpredict <- predict(RFmodel, credit_testing)
credit_RFModel_Factor <- function(x){ if(x < .5){return(0)} else return(1)}
credit_RFpredict <- sapply(credit_RFpredict, credit_RFModel_Factor)
modelAccuracy = mean(credit_RFpredict == credit_testing$SeriousDlqin2yrs) *100
modelAccuracy

credit_RFpredict <- predict(RFmodel, credit_testing)
credit_RFModel_Factor <- function(x){ if(x < .7){return(0)} else return(1)}
credit_RFpredict <- sapply(credit_RFpredict, credit_RFModel_Factor)
modelAccuracy = mean(credit_RFpredict == credit_testing$SeriousDlqin2yrs) *100
modelAccuracy

credit_RFpredict <- predict(RFmodel, credit_testing)
credit_RFModel_Factor <- function(x){ if(x < .3){return(0)} else return(1)}
credit_RFpredict <- sapply(credit_RFpredict, credit_RFModel_Factor)
modelAccuracy = mean(credit_RFpredict == credit_testing$SeriousDlqin2yrs) *100
modelAccuracy

confusionMatrix(credit_testing$SeriousDlqin2yrs, credit_RFpredict)

modelAccuracy = mean(credit_RFpredict == credit_testing$SeriousDlqin2yrs) *100
modelAccuracy
importance(RFmodel)

SVMmodel <- svm(fol, data=credit_training)
credit_SVMpredict <- predict(SVMmodel, credit_testing)
credit_SVMmodel_Factor <- function(x){ if(x < .5){return(0)} else return(1)}
credit_SVMpredict <- sapply(credit_SVMpredict, credit_SVMmodel_Factor)
```

```
modelAccuracy = mean(credit_SVMpredict == credit_testing$SeriousDlqin2yrs) *100
modelAccuracy

confusionMatrix(credit_testing$SeriousDlqin2yrs, credit_SVMpredict)
```

Further clean-up

1. Get rid of outliers for RevolvingUtilizationOfUnsecuredLines and DebtRatio.

```
credit_clean <- credit_noNA[credit_noNA$RevolvingUtilizationOfUnsecuredLines<999, ]
credit_clean <- credit_clean[credit_noNA$DebtRatio<20, ]

credit_sample <- credit_clean[sample(nrow(credit_clean), size = 2000, replace = FALSE, prob = NULL),]
inTrain <- createDataPartition(y=credit_sample$SeriousDlqin2yrs, p = .7, list = FALSE)
credit_training <- credit_sample[inTrain,]
credit_testing <- credit_sample[-inTrain,]

fol <- formula(SeriousDlqin2yrs ~ RevolvingUtilizationOfUnsecuredLines + age+ NumberOfTime30.59DaysPastDueNotWorse
+ DebtRatio + MonthlyIncome + NumberOfOpenCreditLinesAndLoans + NumberOfTimes90DaysLate +
NumberRealEstateLoansOrLines + NumberOfTime60.89DaysPastDueNotWorse + NumberOfDependents)
model <- rpart(fol, method="class", data=credit_training)
print(model)
credit_predict <- predict(model, credit_testing, type="class")
result = credit_testing$SeriousDlqin2yrs==credit_predict
modelAccuracy = mean(result) *100
modelAccuracy

confusionMatrix(credit_testing$SeriousDlqin2yrs, credit_predict)


RFcontrol <- trainControl(method="cv", number=5)
RFmodel = train(SeriousDlqin2yrs ~ ., method="rf", data=credit_training, trControl=RFcontrol, ntree = 250)
print(RFmodel)
saveRDS(RFmodel, "rfmodel.RDS")


credit_RFpredict <- predict(RFmodel, credit_testing)
credit_RFModel_Factor <- function(x){ if(x < .5){return(0)} else return(1)}
credit_RFpredict <- sapply(credit_RFpredict, credit_RFModel_Factor)
modelAccuracy = mean(credit_RFpredict == credit_testing$SeriousDlqin2yrs) *100
modelAccuracy

confusionMatrix(credit_testing$SeriousDlqin2yrs, credit_RFpredict)

modelAccuracy = mean(credit_RFpredict == credit_testing$SeriousDlqin2yrs) *100
modelAccuracy
importance(RFmodel)

SVMmodel <- svm(fol, data=credit_training)
credit_SVMpredict <- predict(SVMmodel, credit_testing)
credit_SVMmodel_Factor <- function(x){ if(x < .5){return(0)} else return(1)}
credit_SVMpredict <- sapply(credit_SVMpredict, credit_SVMmodel_Factor)

modelAccuracy = mean(credit_SVMpredict == credit_testing$SeriousDlqin2yrs) *100
modelAccuracy

confusionMatrix(credit_testing$SeriousDlqin2yrs, credit_SVMpredict)
```

2. Create Bins for Various Factors

```
RevolvingUtilizationOfUnsecuredLines_Cut <- function(x) {cut(x, c(0, 0.5, 1.0, 10.0, Inf), labels=1:4, include.lowest=TRUE)}
credit_noNA$RevolvingUtilizationOfUnsecuredLinesBins <- lapply(credit_noNA$RevolvingUtilizationOfUnsecuredLines,
RevolvingUtilizationOfUnsecuredLines_Cut)

Age_Cut <- function(x) {cut(x, c(0, 20, 30, 40, 50, 60, 70, 80, 90, Inf), include.lowest=FALSE)}
credit_noNA$ageBins <- lapply(credit_noNA$age, Age_Cut)

NumberOfTime30.59DaysPastDueNotWorse_Cut <- function(x) {cut(x, c(0, 1, 5, 10, 20, 50, Inf), labels=1:6,
include.lowest=TRUE)}
```

```
credit_noNA$NumberOfTime30.59DaysPastDueNotWorseBins <-
lapply(credit_noNA$NumberOfTime30.59DaysPastDueNotWorse, NumberOfTime30.59DaysPastDueNotWorse_Cut)

DebtRatio_Cut <- function(x) {cut(x, c(0, 0.25, 0.5, 0.75, 1.0, 5.0, 10.0, Inf), labels=1:7, include.lowest=TRUE)}
credit_noNA$DebtRatioBins <- lapply(credit_noNA$DebtRatio, DebtRatio_Cut)

MonthlyIncome_Cut <- function(x) {cut(x, c(0, 3000, 5500, 10000, 20000, 50000, Inf), labels=1:6, include.lowest=TRUE)}
credit_noNA$MonthlyIncomeBins <- lapply(credit_noNA$MonthlyIncome, MonthlyIncome_Cut)

NumberOfOpenCreditLinesAndLoans_Cut <- function(x) {cut(x, c(0, 2, 5, 10, 20, 50, Inf), labels=1:6, include.lowest=TRUE)}
credit_noNA$NumberOfOpenCreditLinesAndLoansBins <- lapply(credit_noNA$NumberOfOpenCreditLinesAndLoans,
NumberOfOpenCreditLinesAndLoans_Cut)

NumberRealEstateLoansOrLines_Cut <- function(x) {cut(x, c(0, 2, 5, 10, 20, 50, Inf), labels=1:6, include.lowest=TRUE)}
credit_noNA$NumberRealEstateLoansOrLinesBins <- lapply(credit_noNA$NumberRealEstateLoansOrLines,
NumberRealEstateLoansOrLines_Cut)

NumberOfDependents_Cut <- function(x) {cut(x, c(0, 1, 2, 3, 4, 5, 6, 7, 8,  Inf), labels=1:9, include.lowest=TRUE)}
credit_noNA$NumberOfDependentsBins <- lapply(credit_noNA$NumberOfDependents, NumberOfDependents_Cut)


credit_sample <- credit_noNA[sample(nrow(credit_noNA), size = 2000, replace = FALSE, prob = NULL),]
inTrain <- createDataPartition(y=credit_sample$SeriousDlqin2yrs, p = .7, list = FALSE)
credit_training <- credit_sample[inTrain,]
credit_testing <- credit_sample[-inTrain,]

fol <- formula(SeriousDlqin2yrs ~ NumberOfTimes90DaysLate + NumberOfTime60.89DaysPastDueNotWorse +
NumberOfTime30.59DaysPastDueNotWorseBins + DebtRatioBins + MonthlyIncomeBins +
NumberOfOpenCreditLinesAndLoansBins + NumberRealEstateLoansOrLinesBins + NumberOfDependentsBins)
model <- rpart(fol, method="class", data=credit_training)
print(model)
credit_predict <- predict(model, credit_testing, type="class")
result = credit_testing$SeriousDlqin2yrs==credit_predict
modelAccuracy = mean(result) *100
modelAccuracy

confusionMatrix(credit_testing$SeriousDlqin2yrs, credit_predict)


RFcontrol <- trainControl(method="cv", number=5)
RFmodel = train(SeriousDlqin2yrs ~ ., method="rf", data=credit_training, trControl=RFcontrol, ntree = 250)
print(RFmodel)
saveRDS(RFmodel, "rfmodel.RDS")


credit_RFpredict <- predict(RFmodel, credit_testing)
credit_RFModel_Factor <- function(x){ if(x < .5){return(0)} else return(1)}
credit_RFpredict <- sapply(credit_RFpredict, credit_RFModel_Factor)
modelAccuracy = mean(credit_RFpredict == credit_testing$SeriousDlqin2yrs) *100
modelAccuracy

confusionMatrix(credit_testing$SeriousDlqin2yrs, credit_RFpredict)

modelAccuracy = mean(credit_RFpredict == credit_testing$SeriousDlqin2yrs) *100
modelAccuracy


SVMmodel <- svm(fol, data=credit_training)
credit_SVMpredict <- predict(SVMmodel, credit_testing)
credit_SVMmodel_Factor <- function(x){ if(x < .5){return(0)} else return(1)}
credit_SVMpredict <- sapply(credit_SVMpredict, credit_SVMmodel_Factor)

modelAccuracy = mean(credit_SVMpredict == credit_testing$SeriousDlqin2yrs) *100
modelAccuracy

confusionMatrix(credit_testing$SeriousDlqin2yrs, credit_SVMpredict)
```