Lending Club - Loan Repayment Prediction Model - Final Report

Introduction

The goal of each and every business is to make profit. For a lender, profit depends on whether or not the borrower repays the principal as well as interest. Without repayment the lender will incur a loss and that loss can even potentially be greater than the initial loan amount when lawyer, court and collection fees are taken into consideration. Therefore, it is critically important for a lender to be able to identify whether a potential borrower can and will make all of his or her loan payments. The purpose of this project is to identify the characteristics (limited to those found in the dataset) of persons who are likely to default on their loans and provide a simple framework for borrowers to make such a distinction. In order to make predictions on which potential borrower is or isn't likely to default, binary classification predictive models will be created using a variety of machine learning algorithms. The framework for lenders to make predictions concerning potential borrowers will come in the form of a decision tree.

Literature Review

Benchmarking state-of-the-art classification algorithms for credit scoring: A ten-year update[2]

http://www.business-school.ed.ac.uk/waf/crc_archive/2013/42.pdf (http://www.business-school.ed.ac.uk/waf/crc_archive/2013/42.pdf)

Essentially this paper is a modern update to the landmark benchmarking study of classification algorithms for credit scoring by Baesens, Van Gestel, Viaene, Stepanova, Suykens and Vanthienen in 2003[1]. Since 2003 many new techniques and algorithms have been developed in predictive modeling[2]. This paper builds upon Baesens et al. by including all of the newer state-of-the-art techniques as well as those covered in the previous study. I chose this paper because its goal of comparing of classification algorithms for credit scoring is highly related to the problem this project aims to solve. The paper describes a vector x of m dimensions with each dimension as a feature characterizing an application for a credit products such as a loan[2]. The dataset used in this project is in the same format, where each row (vector) represents an individual loan. The study then goes on to discuss a binary response variable which indicates the existence or non-existence of a default event[2]. The probability of a default event given x is the classification problem being addressed in the study. Finally, a decision maker will take this probability and if it falls under a given threshold the application will be accepted, otherwise it will be rejected. This is essentially the approach I am taking for this capstone project. The Lending Club data set I am using for this project contains over 100 features characterizing the borrower, my vector x. The data set also contains a feature stating the current status of the loan with various possible values, each of which can easily be grouped into default or non-default statuses. This is effectively my binary response variable as described in the study. Finally my goal is to estimate the probability of default given a set of borrower characteristics and use that to determine whether they are likely or unlikely to default. Again, this is parallel to the study. The study considered the following classification algorithms.

TABLE 2: CLASSIFICATION ALGORITHMS CONSIDERED IN THE BENCHMARKING STUDY

	Base model selection	Classification algorithm	Acronym	Number of models[1]
Individual classifier	n.a.	Bayesian Network	B-Net	4
		CART	CART	10
		Extreme learning machine	ELM	120
		Kernalized ELM	ELM-K	200
		k-nearest neighbor	kNN	22
		J4.8	J4.8	36
		Linear discriminant analysis[2]	LDA	1
		Linear support vector machine	SVM-L	29
		Logistic regression[2]	LR	1
		Multilayer perceptron artificial neural network	ANN	171

		Naive Bayes	NB	1
		Quadratic discriminant analysis[2]	QDA	1
		Radial basis function neural network	RbfNN	5
		Regularized logistic regression	LR-R	27
		SVM with radial basis kernel function	SVM- Rbf	300
		Voted perceptron	VP	5
	Classification models from individual classifiers		**16**	**933**
Homogenous ensembles	n.a.	Alternating decision tree	ADT	5
		Bagged decision trees	Bag	9
		Bagged MLP	BagNN	4
		Boosted decision trees	Boost	48
		Logistic model tree	LMT	1
		Random forest	RF	30
		Rotation forest	RotFor	25
		Stochastic gradient boosting	SGB	9
	Classification models from homogeneous ensembles		**8**	**131**
Heterogeneous ensembles	n.a.	Simple average ensemble	AvgS	1
		Weighted average ensemble	AvgW	1
	Static direct	Complementary measure	CompM	4
		Ensemble pruning via reinforcement learning	EPVRL	4
		GASEN	GASEN	4
		Hill-climbing ensemble selection	HCES	12
		HCES with bootstrap sampling	HCES-Bag	16
		Matchting pursuit optimization of ensemble classifiers	MPOCE	1
		Stacking	Stack	6
		Top-*T* ensemble	Top-*T*	12
	Static indirect	Clustering using compound error	CuCE	1
		k-Means clustering	k-Means	1
		Kappa pruning	KaPru	4
		Margin distance minimization	MDM	4
		Uncertainty weighted accuracy	UWA	4
	Dynamic	Probabilistic model for classifier competence	PMCC	1
		k-nearest oracle	kNORA	1
	Classification models from heterogeneous ensembles		**17**	**77**
Overall number of classification algorithms and models			**41**	**1141**

The following table measures the performance of each algorithm in credit scoring classification using Area Under a ROC Curve (AUC). According to the study across all performance measures the top three most accurate classifiers are Random Forests, Bagged (MLP) Neural Networks, and Bagged Decision Trees.

TABLE 5: PERFORMANCE OF INDIVIDUAL CLASSIFIERS AND HOMOGENEOUS ENSEMBLES IN TERMS OF THE AUC

		AC		GC		Bene1		Bene2		UK		PAK		GMC	
Individual classifiers	ANN	.926	(.011)	.791	(.014)	.791	(.009)	.802	(.005)	.742	(.008)	.644	(.004)	.859	(.003)
	B-Net	.922	(.011)	.764	(.015)	.771	(.009)	.786	(.009)	.703	(.023)	.623	(.004)	.860	(.003)
	CART	.856	(.019)	.706	(.031)	.706	(.021)	.713	(.021)	.684	(.012)	.565	(.015)	.797	(.025)
	ELM	.911	(.011)	.778	(.012)	.766	(.010)	.761	(.006)	.650	(.009)	.599	(.003)	.717	(.004)
	ELM-K	.926	(.012)	.794	(.015)	.787	(.007)	.788	(.005)	.734	(.009)	.643	(.004)	.702	(.004)
	J4.8	.915	(.014)	.734	(.020)	.761	(.012)	.747	(.011)	.500	(.000)	.500	(.000)	.500	(.000)
	k-NN	.906	(.016)	.772	(.010)	.765	(.009)	.754	(.007)	.725	(.014)	.600	(.005)	.739	(.004)
	LDA	.929	(.009)	.784	(.012)	.775	(.011)	.779	(.008)	.715	(.010)	.626	(.003)	.692	(.004)
	LR	**.931**	(.011)	.784	(.012)	.773	(.012)	.791	(.006)	.720	(.011)	.626	(.003)	.693	(.005)
	LR-R	.925	(.012)	.778	(.015)	.787	(.007)	.798	(.004)	.690	(.012)	.635	(.004)	.623	(.006)
	NB	.893	(.020)	.777	(.017)	.747	(.013)	.724	(.010)	.701	(.019)	.613	(.006)	.671	(.003)
	RbfNN	.902	(.019)	.762	(.013)	.760	(.009)	.739	(.007)	.701	(.014)	.604	(.003)	.755	(.007)
	QDA	.917	(.018)	.674	(.148)	.765	(.011)	.780	(.006)	.703	(.012)	.612	(.004)	.811	(.003)
	SVM-L	.924	(.013)	.782	(.014)	.786	(.007)	.796	(.003)	.659	(.014)	.636	(.004)	.733	(.017)
	SVM-Rbf	.926	(.012)	.799	(.011)	.786	(.008)	.795	(.004)	.666	(.028)	.630	(.004)	.815	(.009)
	VP	.810	(.030)	.680	(.020)	.698	(.013)	.621	(.017)	.554	(.018)	.567	(.003)	.568	(.024)
Homogeneous ensemble classifiers	ADT	.929	(.010)	.758	(.012)	.786	(.008)	.794	(.010)	.732	(.008)	.641	(.004)	.860	(.004)
	Bag	.930	(.014)	.788	(.014)	.794	(.008)	.805	(.006)	.742	(.007)	.643	(.003)	**.864**	(.003)
	BagNN	.927	(.012)	**.802**	(.010)	.793	(.008)	.802	(.004)	**.745**	(.008)	**.646**	(.004)	.838	(.004)
	Boost	.930	(.010)	.772	(.012)	**.795**	(.007)	**.808**	(.005)	.741	(.010)	.643	(.004)	.860	(.003)
	LMT	.930	(.013)	.747	(.015)	.780	(.007)	.787	(.006)	.720	(.010)	.630	(.004)	.833	(.017)
	RF	.931	(.014)	.789	(.013)	.794	(.008)	.805	(.006)	.742	(.007)	.643	(.003)	**.864**	(.003)
	RotFor	.929	(.013)	.773	(.015)	.788	(.007)	.794	(.007)	.502	(.016)	.635	(.002)	.820	(.005)
	SGB	.928	(.013)	.751	(.015)	.786	(.007)	.797	(.006)	.735	(.012)	.642	(.004)	.860	(.003)

An Empirical Comparison of Supervised Learning Algorithms[3]

https://www.cs.cornell.edu/~caruana/ctp/ct.papers/caruana.icml06.pdf (https://www.cs.cornell.edu/~caruana/ctp/ct.papers/caruana.icml06.pdf)

This study compares the performance of eight machine learning algorithms namely, SVMs, neural nets, logistic regression, naïve bayes, memory based learning, random forests, decision trees, bagged trees, boosted trees, and boosted stumps. The performance metrics used are, accuracy, F-score, Lift, ROC Area, average precision, squared error and cross entropy. The study concludes that bagged trees, random forests and neural nets have the best average performance (prior to calibration) over all the metrics and over all the problems. When calibration is taken into account, the overall best performing algorithm is boosted decision trees (calibrated)[3]. In close second is Random forests, followed by bagged decision trees (uncalibrated).

Table 3. Normalized scores of each learning algorithm by problem (averaged over eight metrics)

MODEL	CAL	COVT	ADULT	LTR.P1	LTR.P2	MEDIS	SLAC	HS	MG	CALHOUS	COD	BACT	MEAN
BST-DT	PLT	**.938**	.857	**.959**	**.976**	.700	.869	**.933**	.855	**.974**	**.915**	.878*	**.896***
RF	PLT	.876	.930	.897	.941	**.810**	.907*	.884	.883	.937	.903*	.847	.892
BAG-DT	–	.878	.944*	.883	.911	.762	.898*	.856	**.898**	.948	.856	**.926**	.887*
BST-DT	ISO	.922*	.865	.901*	.969	.692*	.878	.927	.845	.965	.912*	.861	.885*
RF	–	.876	.946*	.883	.922	.785	.912*	.871	.891*	.941	.874	.824	.884
BAG-DT	PLT	.873	.931	.877	.920	.752	.885	.863	.884	.944	.865	.912*	.882
RF	ISO	.865	.934	.851	.935	.767*	**.920**	.877	.876	.933	.897*	.821	.880
BAG-DT	ISO	.867	.933	.840	.915	.749	.897	.856	.884	.940	.859	.907*	.877
SVM	PLT	.765	.886	.936	.962	.733	.866	.913*	.816	.897	.900*	.807	.862
ANN	–	.764	.884	.913	.901	.791*	.881	.932*	.859	.923	.667	.882	.854
SVM	ISO	.758	.882	.899	.954	.693*	.878	.907	.827	.897	.900*	.778	.852
ANN	PLT	.766	.872	.898	.894	.775	.871	.929*	.846	.919	.665	.871	.846
ANN	ISO	.767	.882	.821	.891	.785*	.895	.926*	.841	.915	.672	.862	.842
BST-DT	–	.874	.842	.875	.913	.523	.807	.860	.785	.933	.835	.858	.828
KNN	PLT	.819	.785	.920	.937	.626	.777	.803	.844	.827	.774	.855	.815
KNN	–	.807	.780	.912	.936	.598	.800	.801	.853	.827	.748	.852	.810
KNN	ISO	.814	.784	.879	.935	.633	.791	.794	.832	.824	.777	.833	.809
BST-STMP	PLT	.644	**.949**	.767	.688	.723	.806	.800	.862	.923	.622	.915*	.791
SVM	–	.696	.819	.731	.860	.600	.859	.788	.776	.833	.864	.763	.781
BST-STMP	ISO	.639	.941	.700	.681	.711	.807	.793	.862	.912	.632	.902*	.780
BST-STMP	–	.605	.865	.540	.615	.624	.779	.683	.799	.817	.581	.906*	.710
DT	ISO	.671	.869	.729	.760	.424	.777	.622	.815	.832	.415	.884	.709
DT	–	.652	.872	.723	.763	.449	.769	.609	.829	.831	.389	.899*	.708
DT	PLT	.661	.863	.734	.756	.416	.779	.607	.822	.826	.407	.890*	.706
LR	–	.625	.886	.195	.448	.777*	.852	.675	.849	.838	.647	.905*	.700
LR	ISO	.616	.881	.229	.440	.763*	.834	.659	.827	.833	.636	.889*	.692
LR	PLT	.610	.870	.185	.446	.738	.835	.667	.823	.832	.633	.895	.685
NB	ISO	.574	.904	.674	.557	.709	.724	.205	.687	.758	.633	.770	.654
NB	PLT	.572	.892	.648	.561	.694	.732	.213	.690	.755	.632	.756	.650
NB	–	.552	.843	.534	.556	.011	.714	-.654	.655	.759	.636	.688	.481

Conclusion

I was fascinated to discover that from both of the studies I researched, ensemble methods were generally the best performers. I was also pleasantly surprised to find that in both studies, the top three performing algorithms were almost identical. According to Benchmarking state-of-the-art classification algorithms for credit scoring: A ten-year update, the top three classifiers were Random Forests, Bagged (MLP) Neural Networks, and Bagged Decision Trees. Meanwhile according to An Empirical Comparison of Supervised Learning Algorithms, the top three are Boosted Decision Trees, Random Forests and Bagged Decision Trees. In both studies Random Forests and Bagged Decision Trees come out on top. It's important to note that An Empirical Comparison of Supervised Learning Algorithms did not include Bagged Neural Networks. Given the agreement among both studies I have decided to use Random Forests and Bagged Decision Trees for this project. For the sake of comparison I am also going to use Support Vector Machines (SVM).
In addition to selecting classification algorithms, metrics for measuring the performance of said algorithms are required. Given the similarities of the goals of my project and the aims of Benchmarking state-of-the-art classification algorithms for credit scoring: A ten-year update, I have decided to use some of its performance measures. Specifically, Percentage Correctly Classified (PCC), and the area under a ROC curve (AUC).

Install all required packages

In [1]:
```
#install.packages("caret", repos='http://cran.us.r-project.org',dependencies =
 TRUE)
install.packages("ggplot2", repos='http://cran.us.r-project.org',dependencies
= TRUE)
install.packages("psych", repos='http://cran.us.r-project.org',dependencies =
TRUE)
install.packages("gmodels", repos='http://cran.us.r-project.org',dependencies
= TRUE)
install.packages("party", repos='http://cran.us.r-project.org',dependencies =
TRUE)
#install.packages("e1071", repos='http://cran.us.r-project.org',dependencies =
 TRUE)
install.packages("unbalanced", repos='http://cran.us.r-project.org',dependenci
es = TRUE)
install.packages("hmeasure", repos='http://cran.us.r-project.org',dependencies
 = TRUE)
install.packages("pROC", repos='http://cran.us.r-project.org',dependencies = T
RUE)
install.packages("adabag", repos='http://cran.us.r-project.org',dependencies =
 TRUE)
library(caret)
library(psych)
library(lattice)
library(party)
library(unbalanced)
#library(hmeasure)
library(pROC)
```

```
package 'ggplot2' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\Ethan\AppData\Local\Temp\RtmpOQ4hA2\downloaded_packages

Warning message:
"dependencies 'graph', 'Rgraphviz' are not available"

package 'psych' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\Ethan\AppData\Local\Temp\RtmpOQ4hA2\downloaded_packages
package 'gmodels' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\Ethan\AppData\Local\Temp\RtmpOQ4hA2\downloaded_packages
package 'party' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\Ethan\AppData\Local\Temp\RtmpOQ4hA2\downloaded_packages
package 'unbalanced' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\Ethan\AppData\Local\Temp\RtmpOQ4hA2\downloaded_packages
package 'hmeasure' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\Ethan\AppData\Local\Temp\RtmpOQ4hA2\downloaded_packages
package 'pROC' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\Ethan\AppData\Local\Temp\RtmpOQ4hA2\downloaded_packages

also installing the dependency 'caret'


package 'caret' successfully unpacked and MD5 sums checked
package 'adabag' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\Ethan\AppData\Local\Temp\RtmpOQ4hA2\downloaded_packages
```

```
Warning message:
"package 'caret' was built under R version 3.3.2"Loading required package: la
ttice
Loading required package: ggplot2
Warning message:
"package 'ggplot2' was built under R version 3.3.2"Warning message:
"package 'psych' was built under R version 3.3.2"
Attaching package: 'psych'

The following objects are masked from 'package:ggplot2':

    %+%, alpha

Warning message:
"package 'party' was built under R version 3.3.2"Loading required package: gr
id
Loading required package: mvtnorm
Warning message:
"package 'mvtnorm' was built under R version 3.3.2"Loading required package:
 modeltools
Warning message:
"package 'modeltools' was built under R version 3.3.2"Loading required packag
e: stats4
Loading required package: strucchange
Warning message:
"package 'strucchange' was built under R version 3.3.2"Loading required packa
ge: zoo

Attaching package: 'zoo'

The following objects are masked from 'package:base':

    as.Date, as.Date.numeric

Loading required package: sandwich
Warning message:
"package 'sandwich' was built under R version 3.3.2"Warning message:
"package 'unbalanced' was built under R version 3.3.2"Loading required packag
e: mlr
Warning message:
"package 'mlr' was built under R version 3.3.2"Loading required package: BBmi
sc
Warning message:
"package 'BBmisc' was built under R version 3.3.2"
Attaching package: 'BBmisc'

The following object is masked from 'package:grid':

    explode

Loading required package: ParamHelpers
Loading required package: stringi

Attaching package: 'mlr'

The following object is masked _by_ '.GlobalEnv':
```

```
    db

The following object is masked from 'package:caret':

    train

Loading required package: foreach
Loading required package: doParallel
Warning message:
"package 'doParallel' was built under R version 3.3.2"Loading required packag
e: iterators
Loading required package: parallel
Warning message:
"package 'pROC' was built under R version 3.3.2"Type 'citation("pROC")' for a
 citation.

Attaching package: 'pROC'

The following objects are masked from 'package:stats':

    cov, smooth, var
```

Dataset

The dataset for this project is the Loan Data dataset from the Lending Club (https://www.lendingclub.com/info/download-data.action (https://www.lendingclub.com/info/download-data.action)). Only 36 month term data from 2007 to February 2013 is being used because it is important that all loans examined are well past their due date. A borrower which defaults on a current loan may do so due to a temporary job loss or a myriad of other reasons. The borrower may then recover and return to good standing once again. Using 36 month term data up until February 2013, ensures that labelling potentially temporary defaults as permanent does not occur.

Loading Raw Data

In [72]:
```
lending_club_2012_2013 <-
read.csv("lending_club_rfe/LoanStats3b_securev1.csv", header=TRUE, skip=1)
lending_club_2007_2011 <-
read.csv("lending_club_rfe/LoanStats3a_securev1.csv", header=TRUE, skip=1)
```

Pruning data set to include only 36 month term loans issued up to February 2013

The reason for this is because all 36 month loans issued February 2013 or earlier have fully come to term.

In [73]:
```
lending_club_jan_2013 <- subset(lending_club_2012_2013,issue_d == "Jan-2013")
lending_club_feb_2013 <- subset(lending_club_2012_2013,issue_d == "Feb-2013")
lending_club_2012 <- subset(lending_club_2012_2013,grepl("2012",issue_d))
lending_club_2012_2013_36 <- subset(rbind(lending_club_2012,lending_club_feb_2
013,lending_club_jan_2013),grepl("36",term))
lending_club_2011 <- subset(lending_club_2007_2011, grepl("2011",issue_d))
lending_club_2007_2010 <- lending_club_2007_2011[-NROW(lending_club_2011),]
lending_club_final <- rbind(lending_club_2007_2010,lending_club_2011,lending_c
lub_2012_2013_36)
```

Number of records in the initial dataset

Please note that each record in the dataset represents an individual loan.

In [74]:
```
nrow(lending_club_final)
```

119276

Number of features (columns) in the initial dataset

In [75]:
```
ncol(lending_club_final)
```

115

Adding label (default) to the data set

default:

1 = true (default has occured)

0 = false (default has not occured)

default is the dependent variable which the predictive model aims to predict

In [76]:
```
# add label (default_status) to dataset
status <- c("Current","Fully Paid","Late (16-30 days)","Does not meet the cred
it policy. Status:Charged Off","Charged Off","Default","In Grace Period","Late
 (31-120 days)","Does not meet the credit policy. Status:Fully Paid")
default_status <- c(0,0,1,1,1,1,1,1,0)
lending_club_default <- data.frame(default = default_status[match(lending_club
_final$loan_status, status)])
lending_club_final <- cbind(lending_club_final,lending_club_default)

# remove all records which have default as NA
lending_club_final <- lending_club_final[-which(is.na(lending_club_final$defau
lt)==TRUE),]
```

Cleaning data set and removing variables/features not known at issuance of loan

The dataset contains variables/features which are only known after the loan is issued. Since the aim of this project is create a predictive model to be used by lenders before the issuance of a loan, the following will be removed:

- total_rec_int
- total_pymnt_inv
- total_pymnt
- total_rec_prncp
- collection_recovery_fee
- recoveries
- last_pymnt_amnt
- total_rec_late_fee
- last_pymnt_d
- last_pymnt_amnt
- next_pymnt_d
- out_prncp
- out_prncp_inv
- issue_d
- initial_list_status
- funded_amnt
- funded_amnt_inv
- id
- pymnt_plan

In [77]:
```
lending_club_final <- lending_club_final[, !(colnames(lending_club_final) %in%
 c("total_rec_int","total_pymnt_inv","total_pymnt"

   ,"total_rec_prncp","collection_recovery_fee",

   "recoveries","last_pymnt_amnt",

   "total_rec_late_fee","last_pymnt_d",

   "last_pymnt_amnt","next_pymnt_d","out_prncp",

   "out_prncp_inv","issue_d",

   "initial_list_status","funded_amnt",

   "funded_amnt_inv","id","pymnt_plan"))]
```

Number of Features in Dataset Before Eliminating 'All NA' Fields

In [78]:
```
length(colnames(lending_club_final))
```

98

Eliminate features which are all NA and populate NA's with column (feature) averages

In [79]:
```
# find which varaibles/features are all NA
na_pct <- sapply(lending_club_final, function(y) sum(is.na(y))/length(y))
na_pct <- data.frame(na_pct)
all_na <- na_pct == 1

# remove variables/features which are all NA
lending_club_final <- lending_club_final[,-which(all_na==TRUE)]

# Find which variables/features have some NA values
na_pct <- sapply(lending_club_final, function(y) sum(is.na(y))/length(y))
na_pct <- data.frame(na_pct)
sig_na <- na_pct > 0
sig_na.df <- as.data.frame(sig_na)
sig_na.df <- subset(sig_na.df, sig_na.df$na_pct==TRUE)
sig_na_col <- row.names(sig_na.df)

# create a data frame of variables/features with some NA values
lending_club_final_na_sig <- lending_club_final[,which(names(lending_club_fina
l) %in% sig_na_col)]

# remove variables/features which have some NA values
lending_club_final <- lending_club_final[,-which(sig_na==TRUE)]

# change the datatypes of the dataframe above to integer
lending_club_final_na_sig_int <- as.data.frame(lapply(lending_club_final_na_si
g,as.integer))

# apply column averages to NA values
for(i in 1:ncol(lending_club_final_na_sig_int)){
  lending_club_final_na_sig_int[is.na(lending_club_final_na_sig_int[,i]), i] <
- mean(lending_club_final_na_sig_int[,i],

          na.rm = TRUE)
}

# bring it all together
lending_club_final <- cbind(lending_club_final_na_sig_int,lending_club_final)
```

Number of Features in Dataset After Eliminating 'All NA' Features

In [80]:
```
length(colnames(lending_club_final))
```

81

Features After Initial Cleansing

In [81]:
```
# columns/features in dataset
colnames(lending_club_final)
```

"emp_title" "annual_inc" "title" "delinq_2yrs" "inq_last_6mths" "mths_since_last_delinq" "mths_since_last_record" "open_acc" "pub_rec" "total_acc" "collections_12_mths_ex_med" "mths_since_last_major_derog" "acc_now_delinq" "tot_coll_amt" "tot_cur_bal" "total_rev_hi_lim" "acc_open_past_24mths" "avg_cur_bal" "bc_open_to_buy" "bc_util" "chargeoff_within_12_mths" "delinq_amnt" "mo_sin_old_il_acct" "mo_sin_old_rev_tl_op" "mo_sin_rcnt_rev_tl_op" "mo_sin_rcnt_tl" "mort_acc" "mths_since_recent_bc" "mths_since_recent_bc_dlq" "mths_since_recent_inq" "mths_since_recent_revol_delinq" "num_accts_ever_120_pd" "num_actv_bc_tl" "num_actv_rev_tl" "num_bc_sats" "num_bc_tl" "num_il_tl" "num_op_rev_tl" "num_rev_accts" "num_rev_tl_bal_gt_0" "num_sats" "num_tl_120dpd_2m" "num_tl_30dpd" "num_tl_90g_dpd_24m" "num_tl_op_past_12m" "pct_tl_nvr_dlq" "percent_bc_gt_75" "pub_rec_bankruptcies" "tax_liens" "tot_hi_cred_lim" "total_bal_ex_mort" "total_bc_limit" "total_il_high_credit_limit" "member_id" "loan_amnt" "term" "int_rate" "installment" "grade" "sub_grade" "emp_length" "home_ownership" "verification_status" "loan_status" "url" "desc" "purpose" "zip_code" "addr_state" "dti" "earliest_cr_line" "fico_range_low" "fico_range_high" "revol_bal" "revol_util" "last_credit_pull_d" "last_fico_range_high" "last_fico_range_low" "policy_code" "application_type" "default"

From looking at the variables, intuitively I would guess the following are significant:

1. int_rate: Interest rate
2. emp_length: Employment length in years. The longer the period of employment the more stable the income.
3. loan_amnt: The loan amount. The larger the loan the more difficult it is to pay off.
4. annual_inc: Annual income.
5. delinq_2yrs: The number of 30+ days past-due incidences of delinquency in the borrower's credit file for the past 2 years. This gives some insight as to the borrower's reliability.
6. fico_range_low: The lower boundary range the borrower’s last FICO pulled belongs to. Credit score is very important.
7. fico_range_high: The upper boundary range the borrower’s last FICO pulled belongs to. Credit score is very important.
8. dti: Debt to income ratio. The higher the ratio, the more likely the borrower is to default.
9. last_fico_range_high: The upper boundary range the borrower’s last FICO pulled belongs to.
10. last_fico_range_low: The lower boundary range the borrower’s last FICO pulled belongs to.

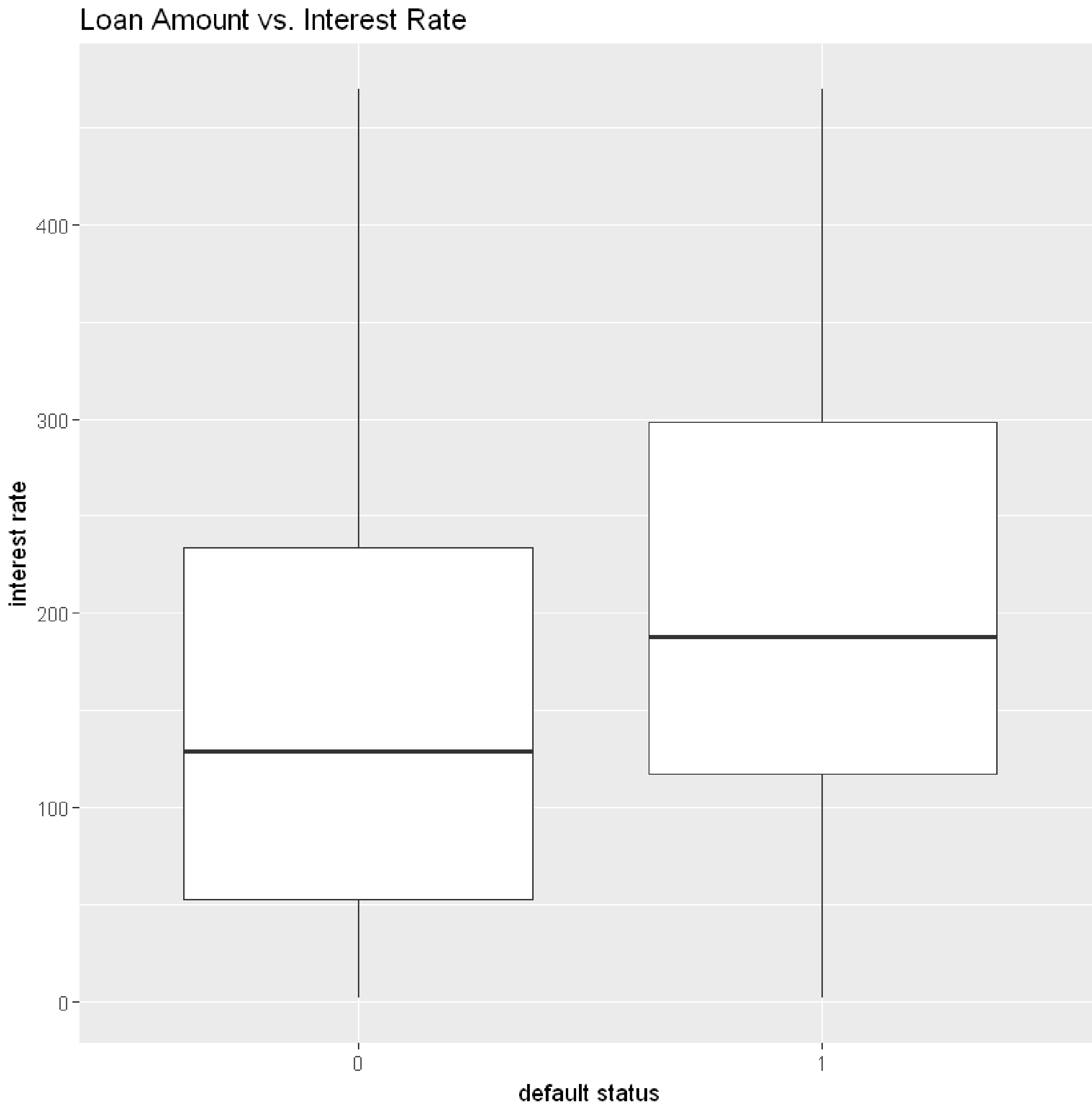
In the next section, each of the features above will be compared to default for correlation.

Please note, for a full description of all of the features in this dataset please refer to Appendix A.

Exploratory Correlation Analysis

int_rate

In [82]:
```
library(ggplot2)
ggplot(lending_club_final, aes(as.factor(default), as.double(int_rate))) + geo
m_boxplot() +
labs(title="Loan Amount vs. Interest Rate", x="default status", y="interest ra
te")
```



Correlation Coefficient - int_rate vs. default

In [83]:
```
# find correlation coefficient
cor(as.numeric(lending_club_final$int_rate), lending_club_final$default, metho
d="pearson")
```

0.134073311733284

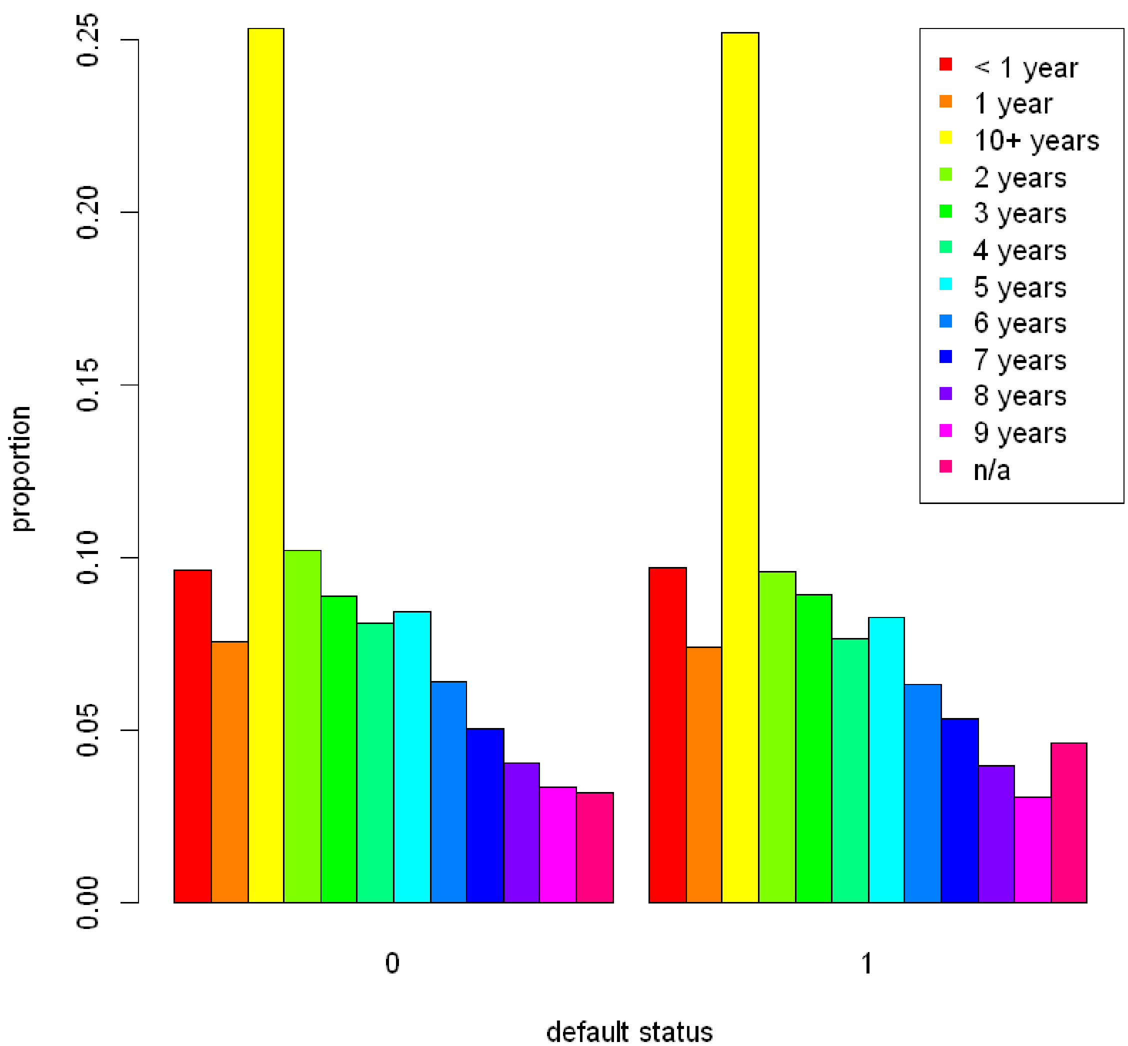
As can be seen from the boxplot above, default is more likely among higher interest rates. This is as expected. Nonetheless, given the correlation coefficient, the interest rate and default status are weakly correlated.

emp_length

In [84]:
```
#install.packages("gmodels", repos='http://cran.us.r-project.org',dependencies
 = TRUE)
library(gmodels)

emp_length.factor <- lending_club_final$emp_length
default.factor <- as.factor(lending_club_final$default)

emp_length.levels <- levels(emp_length.factor)
emp_length.levels <-
emp_length.levels[2:nlevels(lending_club_final$emp_length)]
emp_length.len <- length(emp_length.levels)

joint <- CrossTable(emp_length.factor, default.factor, prop.chisq=FALSE)
joint_counts <- joint$prop.col
barplot(joint_counts, beside=TRUE, ylab="proportion", xlab="default status", c
ol=rainbow(emp_length.len))
legend("topright", emp_length.levels, pch=15, col=rainbow(emp_length.len))
```

```
   Cell Contents
|-------------------------|
|                       N |
|           N / Row Total |
|           N / Col Total |
|         N / Table Total |
|-------------------------|


Total Observations in Table:  119273


                  | default.factor 
emp_length.factor |         0 |         1 | Row Total | 
------------------|-----------|-----------|-----------|
         < 1 year |      9819 |      1665 |     11484 | 
                  |     0.855 |     0.145 |     0.096 | 
                  |     0.096 |     0.097 |           | 
                  |     0.082 |     0.014 |           | 
------------------|-----------|-----------|-----------|
           1 year |      7710 |      1267 |      8977 | 
                  |     0.859 |     0.141 |     0.075 | 
                  |     0.075 |     0.074 |           | 
                  |     0.065 |     0.011 |           | 
------------------|-----------|-----------|-----------|
        10+ years |     25862 |      4312 |     30174 | 
                  |     0.857 |     0.143 |     0.253 | 
                  |     0.253 |     0.252 |           | 
                  |     0.217 |     0.036 |           | 
------------------|-----------|-----------|-----------|
          2 years |     10407 |      1638 |     12045 | 
                  |     0.864 |     0.136 |     0.101 | 
                  |     0.102 |     0.096 |           | 
                  |     0.087 |     0.014 |           | 
------------------|-----------|-----------|-----------|
          3 years |      9087 |      1526 |     10613 | 
                  |     0.856 |     0.144 |     0.089 | 
                  |     0.089 |     0.089 |           | 
                  |     0.076 |     0.013 |           | 
------------------|-----------|-----------|-----------|
          4 years |      8249 |      1309 |      9558 | 
                  |     0.863 |     0.137 |     0.080 | 
                  |     0.081 |     0.076 |           | 
                  |     0.069 |     0.011 |           | 
------------------|-----------|-----------|-----------|
          5 years |      8592 |      1415 |     10007 | 
                  |     0.859 |     0.141 |     0.084 | 
                  |     0.084 |     0.083 |           | 
                  |     0.072 |     0.012 |           | 
------------------|-----------|-----------|-----------|
          6 years |      6550 |      1084 |      7634 | 
                  |     0.858 |     0.142 |     0.064 | 
                  |     0.064 |     0.063 |           | 
                  |     0.055 |     0.009 |           | 
------------------|-----------|-----------|-----------|
          7 years |      5133 |       910 |      6043 | 
```

```
                  |     0.849 |     0.151 |     0.051 |
                  |     0.050 |     0.053 |           |
                  |     0.043 |     0.008 |           |
------------------|-----------|-----------|-----------|
          8 years |      4114 |       681 |      4795 |
                  |     0.858 |     0.142 |     0.040 |
                  |     0.040 |     0.040 |           |
                  |     0.034 |     0.006 |           |
------------------|-----------|-----------|-----------|
          9 years |      3400 |       524 |      3924 |
                  |     0.866 |     0.134 |     0.033 |
                  |     0.033 |     0.031 |           |
                  |     0.029 |     0.004 |           |
------------------|-----------|-----------|-----------|
              n/a |      3226 |       793 |      4019 |
                  |     0.803 |     0.197 |     0.034 |
                  |     0.032 |     0.046 |           |
                  |     0.027 |     0.007 |           |
------------------|-----------|-----------|-----------|
     Column Total |    102149 |     17124 |    119273 |
                  |     0.856 |     0.144 |           |
------------------|-----------|-----------|-----------|
```



Correlation Coefficient - emp_length vs. default

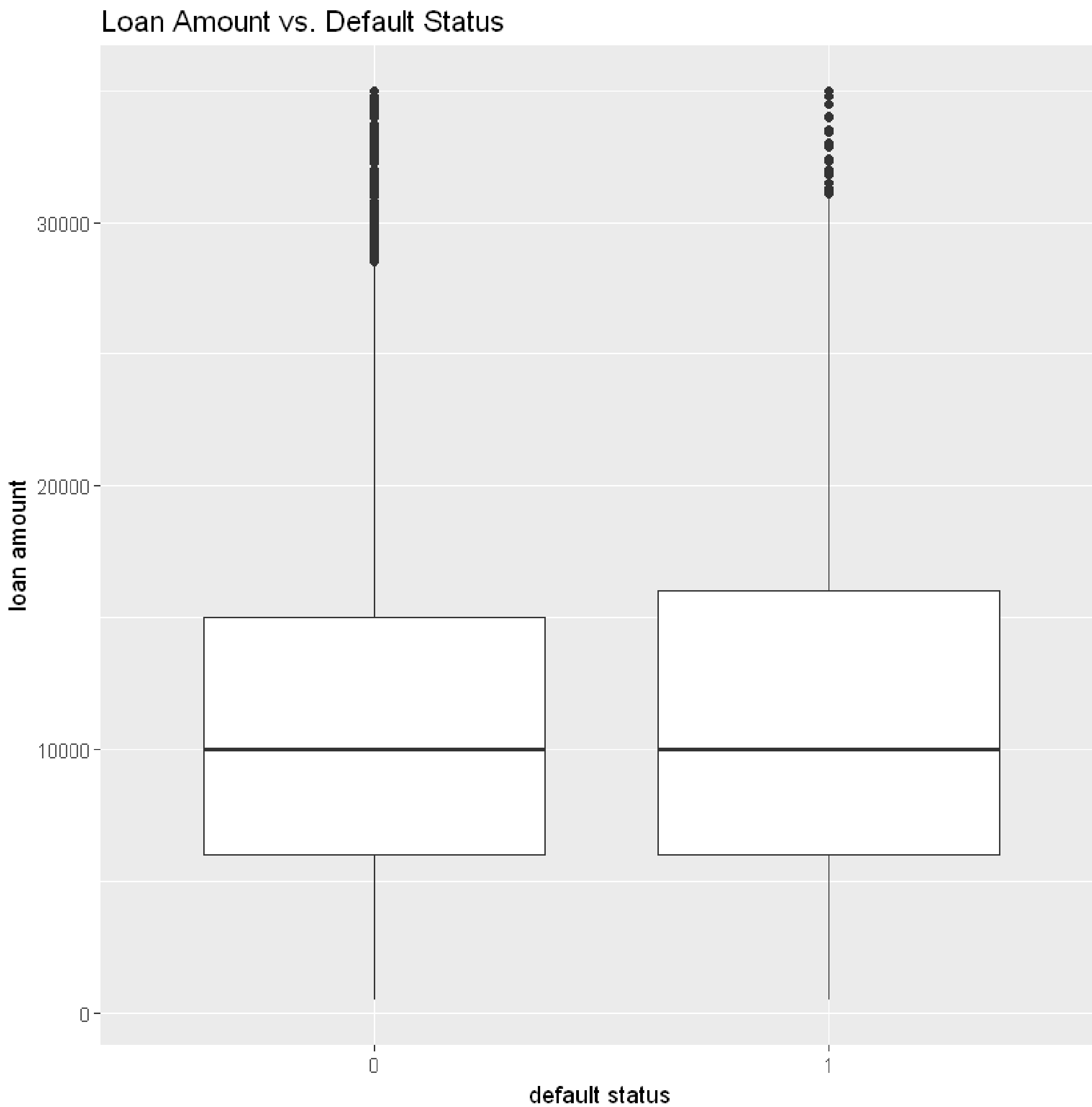
In [85]:
```
# find correlation coefficient
cor(as.numeric(lending_club_final$emp_length), lending_club_final$default, met
hod="pearson")
```

0.0111569730555627

As can be seen from the diagram above, the distribution of loans by employment length is essentially the same for those who defaulted and those who did not. Therefore, emp_length does seem to have much of a bearing in whether a borrower is likely or not to default. This corresponds to the fact that the correlation coefficient indicates a very weak correlation.

Loan Amount (loan_amnt)

In [86]:
```
ggplot(lending_club_final, aes(as.factor(default), loan_amnt)) +
geom_boxplot() +
labs(title="Loan Amount vs. Default Status", x="default status", y="loan amoun
t")
```



Correlation Coefficient - loan amount vs. default

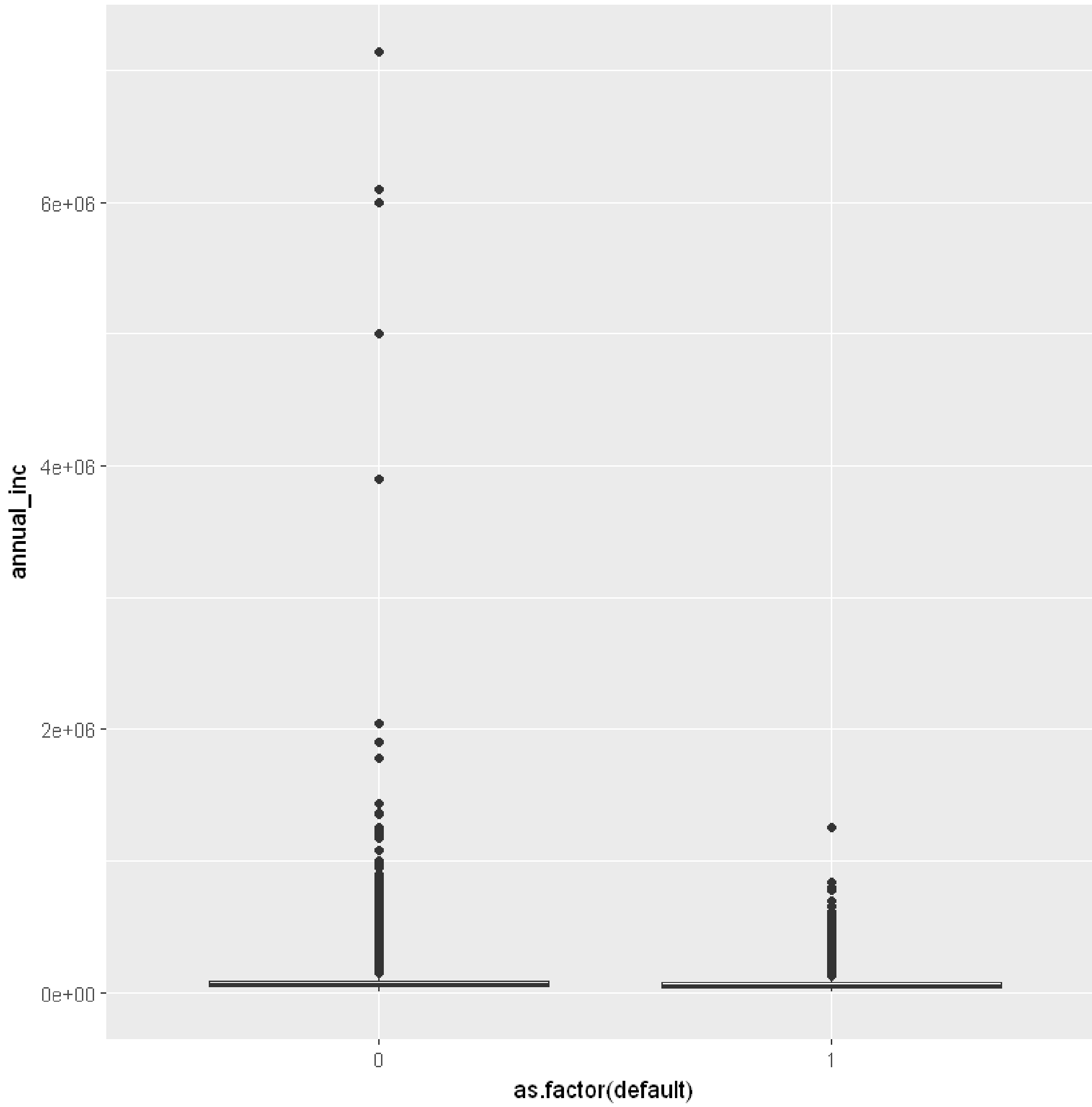
In [87]:
```
# find correlation coefficient
cor(as.numeric(lending_club_final$loan_amnt), lending_club_final$default, meth
od="pearson")
```

0.0222278832849699

As can be seen from the boxplot above, the loan amount has little bearing on whether or not a borrower is going to default. This is further supported by a small correlation coefficient. Therefore we can conclude that loan_amnt and default are weakly correlated.

Annual Income (annual_inc)

In [88]:
```
ggplot(lending_club_final, aes(as.factor(default), annual_inc)) +
geom_boxplot()
```



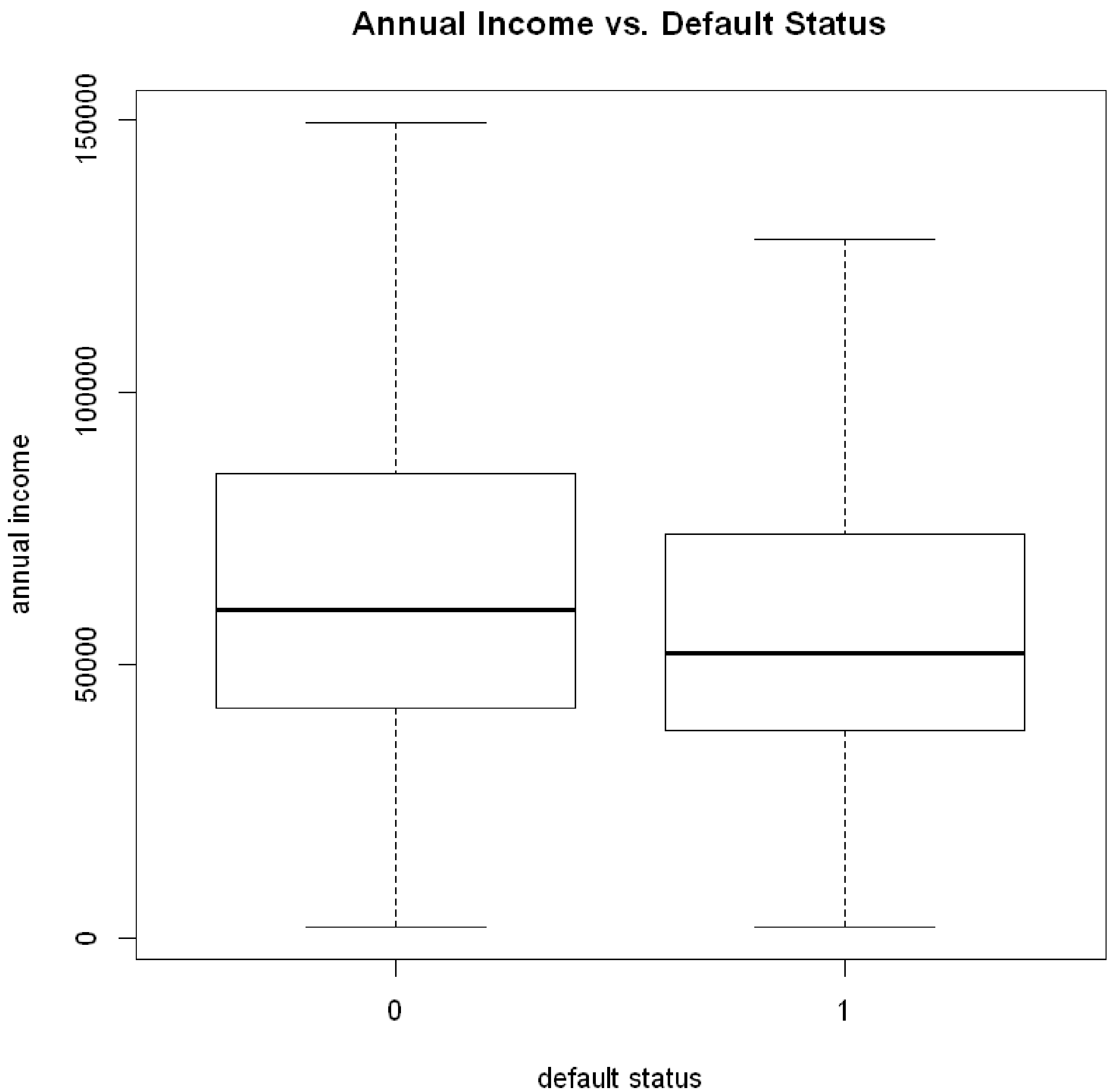
The boxplot is not visible due to extreme outliers. I will redo boxplot without outliers.

In [89]:
```
summary(lending_club_final$annual_inc)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1896   41000   60000   68910   83000 7142000
```

The outlier problem is further illustrated by the summary table above. The minimum annual income is $1896 and the maximum is $7,142,000. That's a range of over 7 million!

In [90]:
```
boxplot(annual_inc~as.factor(default),outline=FALSE, data=lending_club_final,
ylab="annual income", xlab="default status")
title("Annual Income vs. Default Status")
```



Correlation Coefficient - annual income vs. default status

In [91]:
```
# find correlation coefficient
cor(as.numeric(lending_club_final$annual_inc), lending_club_final$default, met
hod="pearson")
```

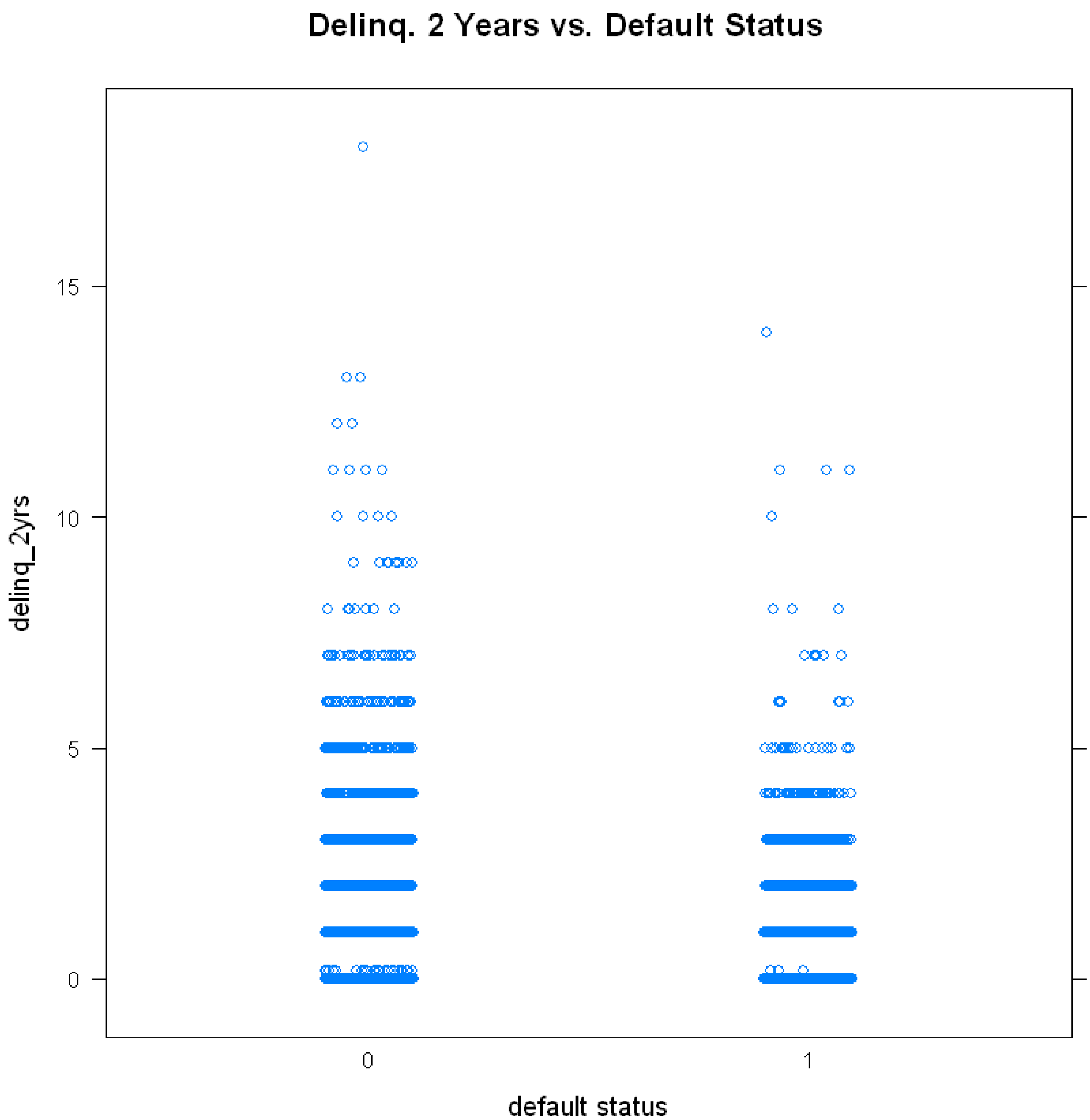
-0.0493465883580697

From the boxplot above we can see that those who defaulted were more likely to have a lower income. Nonetheless, given the correlation coefficient, the correlation is weak.

Number of delinquencies in the last 2 years (delinq_2yrs)

In [92]:
```
summary(lending_club_final$delinq_2yrs)
delinq_2.table <- table(delinq_2yrs=lending_club_final$delinq_2yrs,default=as.
factor(lending_club_final$default))
prop.table(delinq_2.table,2)

#install.packages("psych", repos='http://cran.us.r-project.org',dependencies =
 TRUE)
#library(psych)
describeBy(lending_club_final$delinq_2yrs, as.factor(lending_club_final$defaul
t))

#library(lattice)
stripplot(lending_club_final$delinq_2yrs~as.factor(lending_club_final$default)
 jitter=TRUE,
          main="Delinq. 2 Years vs. Default Status", xlab="default status", yl
ab="delinq_2yrs")
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 0.0000  0.0000  0.0000  0.1726  0.0000 18.0000 

                   default
delinq_2yrs                    0            1
  0                 8.803219e-01 8.685471e-01
  0.172637616987018 2.545301e-04 1.751927e-04
  1                 8.841007e-02 9.530484e-02
  2                 2.056799e-02 2.359262e-02
  3                 6.324095e-03 7.066106e-03
  4                 2.065610e-03 2.803083e-03
  5                 1.047489e-03 1.167951e-03
  6                 4.601122e-04 5.255781e-04
  7                 2.545301e-04 3.503854e-04
  8                 6.852735e-05 1.751927e-04
  9                 8.810659e-05 0.000000e+00
  10                3.915848e-05 5.839757e-05
  11                4.894811e-05 1.751927e-04
  12                1.957924e-05 0.000000e+00
  13                1.957924e-05 0.000000e+00
  14                0.000000e+00 5.839757e-05
  18                9.789621e-06 0.000000e+00

$`0`
   vars      n mean   sd median trimmed mad min max range skew kurtosis se
X1    1 102149 0.17 0.56      0    0.02   0   0  18    18 5.83    61.52  0

$`1`
   vars     n mean   sd median trimmed mad min max range skew kurtosis se
X1    1 17124 0.19 0.61      0    0.04   0   0  14    14 5.85    60.51  0

attr(,"call")
by.default(data = x, INDICES = group, FUN = describe, type = type)
```



Correlation Coefficient - delinq_2yrs vs. default

In [93]:
```
# find correlation coefficient
cor(lending_club_final$delinq_2yrs, lending_club_final$default, method="pearso
n")
```

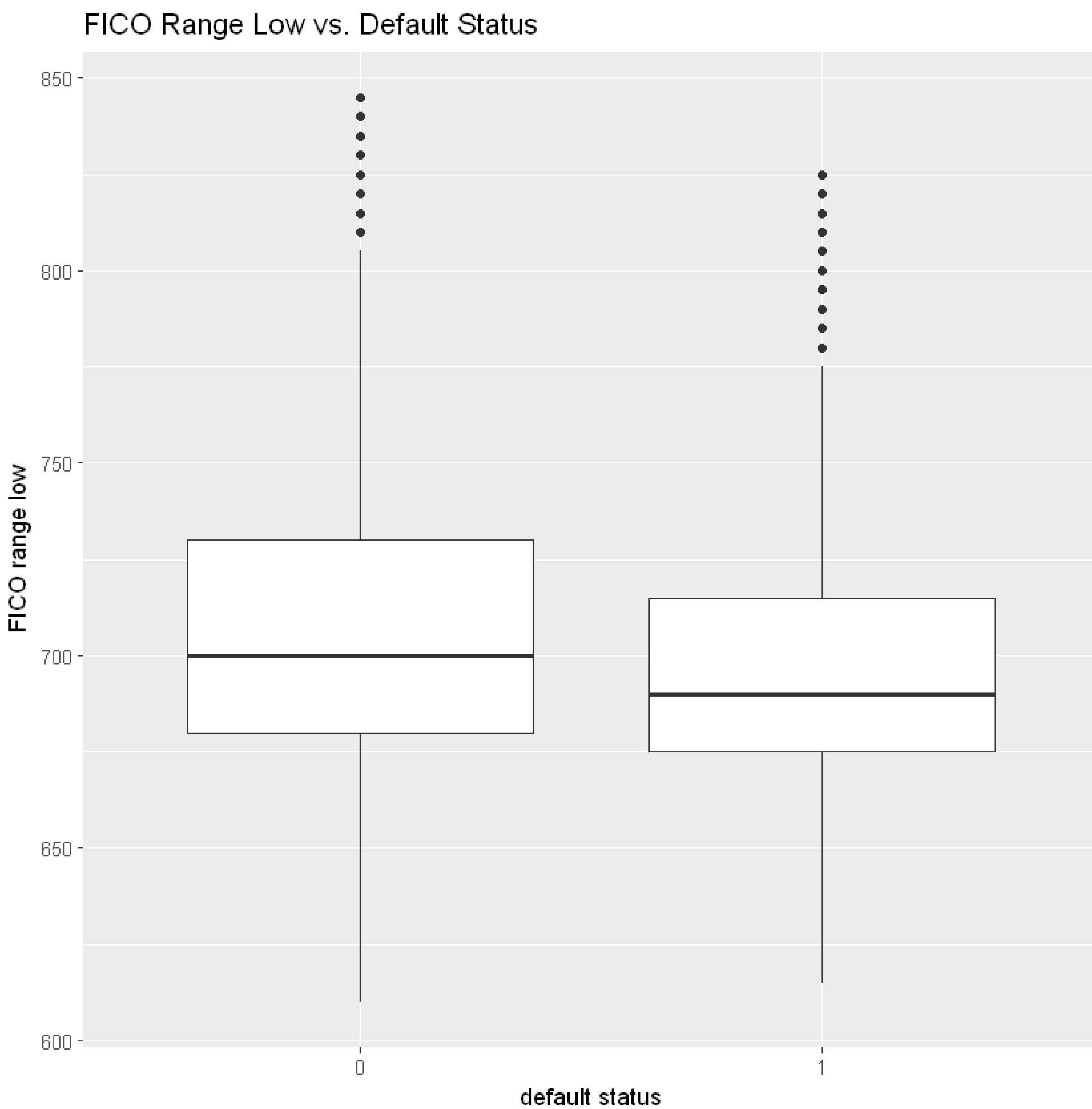
0.0132384016452998

The distributions for the number of delinquencies in the last 2 years by default status are very similar. Combined with a very small coefficient, we can conclude delinq_2yrs and default are weakly correlated.

fico_range_low

In [94]:
```
describeBy(lending_club_final$fico_range_low, as.factor(lending_club_final$def
ault))
ggplot(lending_club_final, aes(as.factor(default), fico_range_low)) + geom_box
plot() +
    labs(title="FICO Range Low vs. Default Status", x="default status", y="FIC
O range low")
```

```
$`0`
   vars      n   mean    sd median trimmed   mad min max range skew kurtosis
X1    1 102149 708.84 35.11    700  705.68 37.06 610 845   235 0.74    -0.06
     se
X1 0.11

$`1`
   vars     n  mean    sd median trimmed   mad min max range skew kurtosis
 se
X1    1 17124 696.8 30.02    690  693.49 29.65 615 825   210 0.96     0.64 0.
23

attr(,"call")
by.default(data = x, INDICES = group, FUN = describe, type = type)
```



Correlation Coefficient - fico_range_low vs. default

In [95]:
```
# find correlation coefficient
cor(lending_club_final$fico_range_low, lending_club_final$default, method="pea
rson")
```

-0.121732212983032

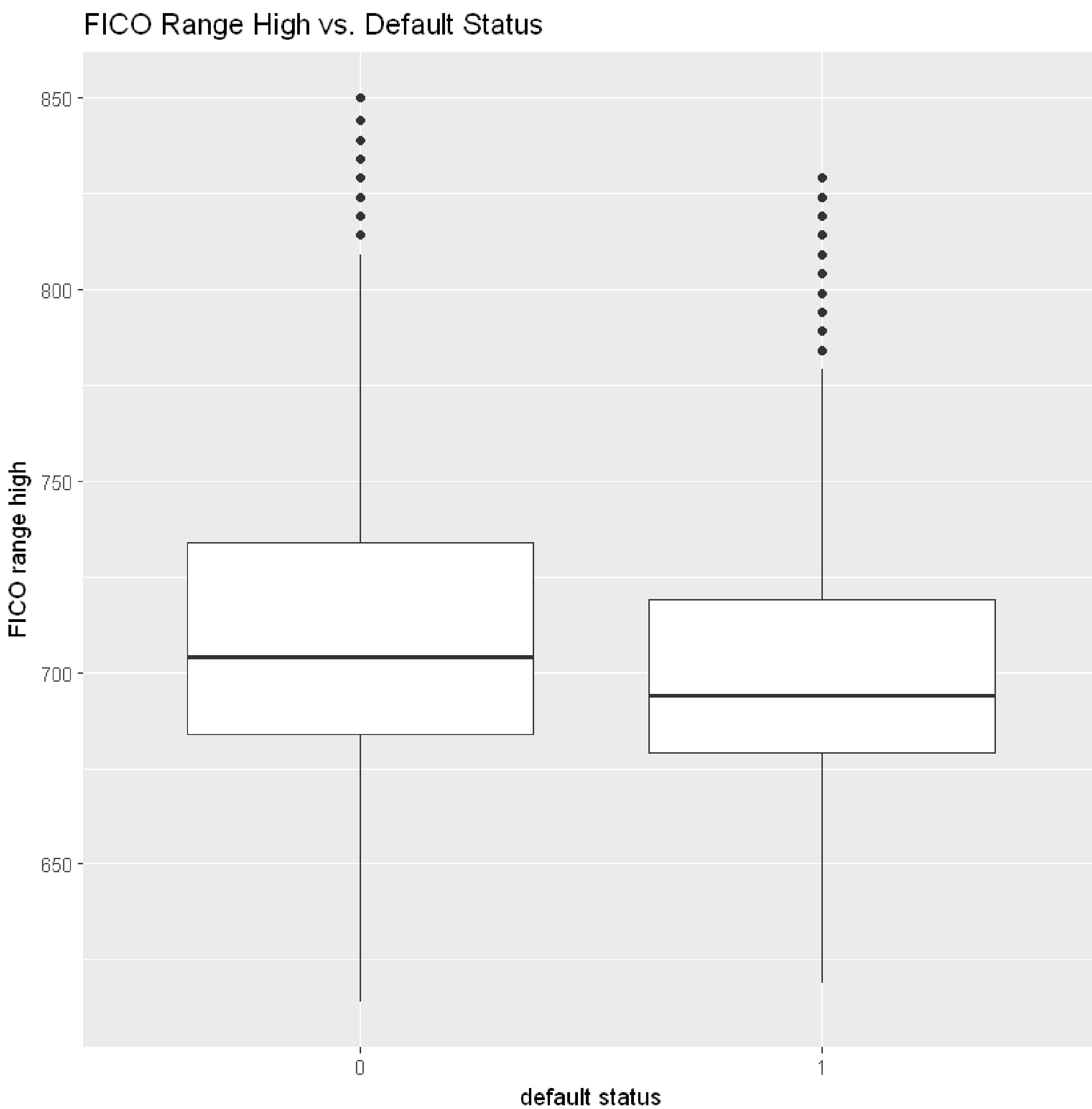
From the boxplots we can see that those who defaulted were more likely to have a lower fico_range_low. Despite this, given the correlation coefficient, we can conclude that fico_range_low and default are weakly correlated.

fico_range_high

In [96]:

```
describeBy(lending_club_final$fico_range_high, as.factor(lending_club_final$de
fault))
ggplot(lending_club_final, aes(as.factor(default), fico_range_high)) + geom_bo
xplot() +
    labs(title="FICO Range High vs. Default Status", x="default status", y="FI
CO range high")
```

```
$`0`
   vars      n   mean    sd median trimmed   mad min max range skew kurtosis
X1    1 102149 712.84 35.12    704  709.68 37.06 614 850   236 0.74    -0.06
     se
X1 0.11

$`1`
   vars     n  mean    sd median trimmed   mad min max range skew kurtosis
 se
X1    1 17124 700.8 30.02    694  697.49 29.65 619 829   210 0.96     0.64 0.
23

attr(,"call")
by.default(data = x, INDICES = group, FUN = describe, type = type)
```



Correlation Coefficient - fico_range_high vs. default

In [97]:
```
# find correlation coefficient
cor(lending_club_final$fico_range_high, lending_club_final$default, method="pe
arson")
```

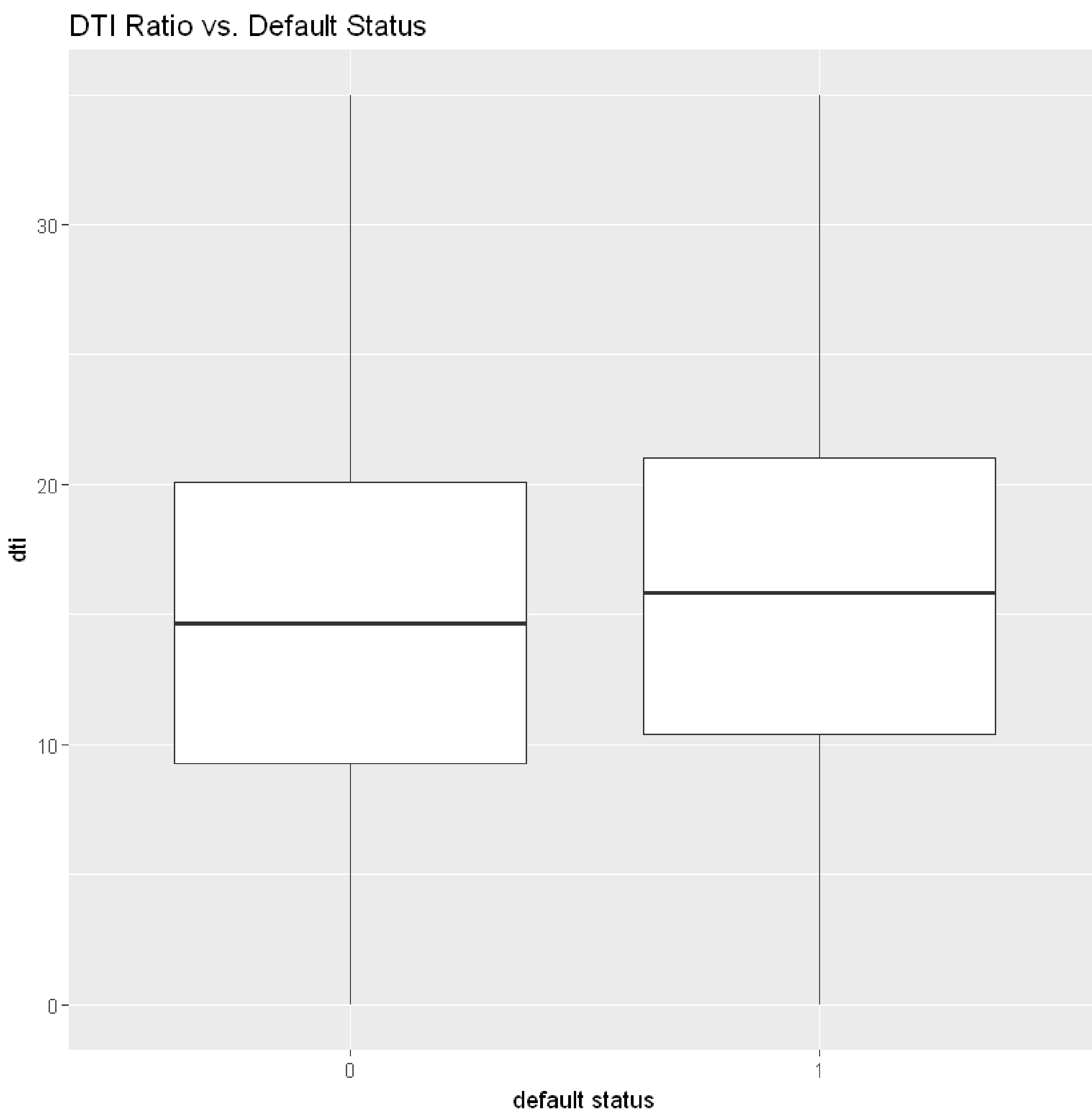
-0.121732047202564

From the boxplots we can see that those who defaulted were more likely to have a lower fico_range_high, but given that the correlation coefficient is so small we can conclude that fico_range_high and default are weakly correlated.

Debt-to-Income Ratio (dti)

In [98]:
```
describeBy(lending_club_final$dti, as.factor(lending_club_final$default))

ggplot(lending_club_final, aes(as.factor(default), dti)) + geom_boxplot() +
    labs(title="DTI Ratio vs. Default Status", x="default status", y="dti")
```

```
$`0`
   vars      n  mean   sd median trimmed  mad min   max range skew kurtosis
X1    1 102149 14.82 7.31  14.65   14.69 8.02   0 34.99 34.99 0.16    -0.54
     se
X1 0.02

$`1`
   vars     n  mean   sd median trimmed  mad min   max range skew kurtosis
 se
X1    1 17124 15.78 7.32  15.86   15.74 7.84   0 34.95 34.95 0.05    -0.51 0.
06

attr(,"call")
by.default(data = x, INDICES = group, FUN = describe, type = type)
```



Correlation Coefficient - dti vs. default

In [99]:
```
# find correlation coefficient
cor(lending_club_final$dti, lending_club_final$default, method="pearson")
```

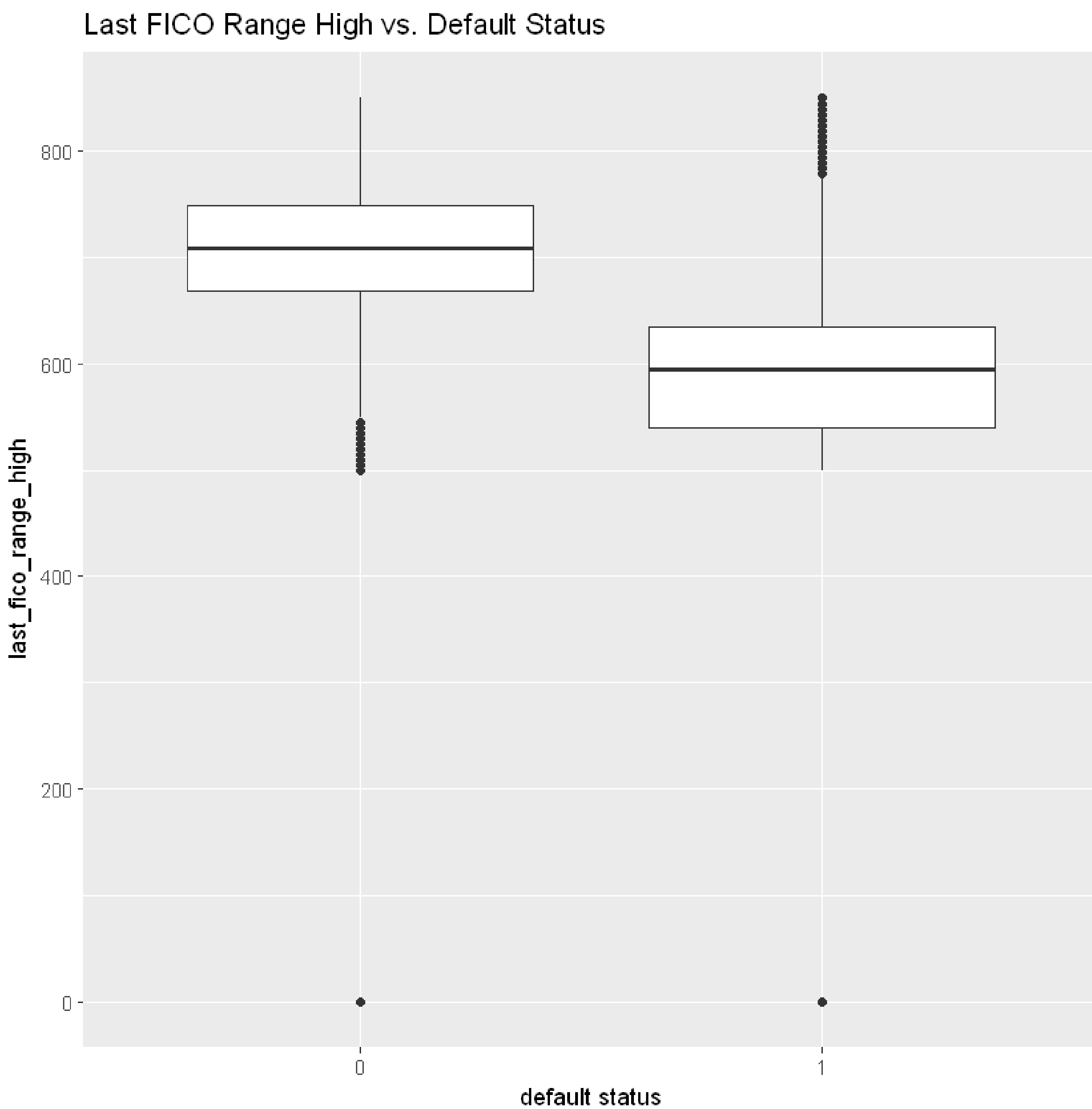
0.0460055401734663

The boxplots above demonstrate that those who defaulted were more likely have a higher DTI (Debt-to-Income) ratio, but given that the small correlation coefficient we can conclude that dti and default are very weakly correlated.

last_fico_range_high

In [100]:
```
describeBy(lending_club_final$last_fico_range_high, as.factor(lending_club_fin
al$default))

ggplot(lending_club_final, aes(as.factor(default), last_fico_range_high)) + ge
om_boxplot() +
    labs(title="Last FICO Range High vs. Default Status", x="default status", 
"last_fico_range_high")
```

```
$`0`
   vars      n   mean    sd median trimmed  mad min max range  skew kurtosis
X1    1 102149 704.46 68.24    709  708.89 59.3   0 850   850 -1.36     8.83
     se
X1 0.21

$`1`
   vars     n   mean    sd median trimmed   mad min max range  skew kurtosis
X1    1 17124 590.82 58.77    594  590.23 66.72   0 850   850 -0.03     0.56
     se
X1 0.45

attr(,"call")
by.default(data = x, INDICES = group, FUN = describe, type = type)
```



Correlation Coefficient - last_fico_range_high vs. default

In [101]:
```
# find correlation coefficient
cor(lending_club_final$last_fico_range_high, lending_club_final$default, method="pearson")
```

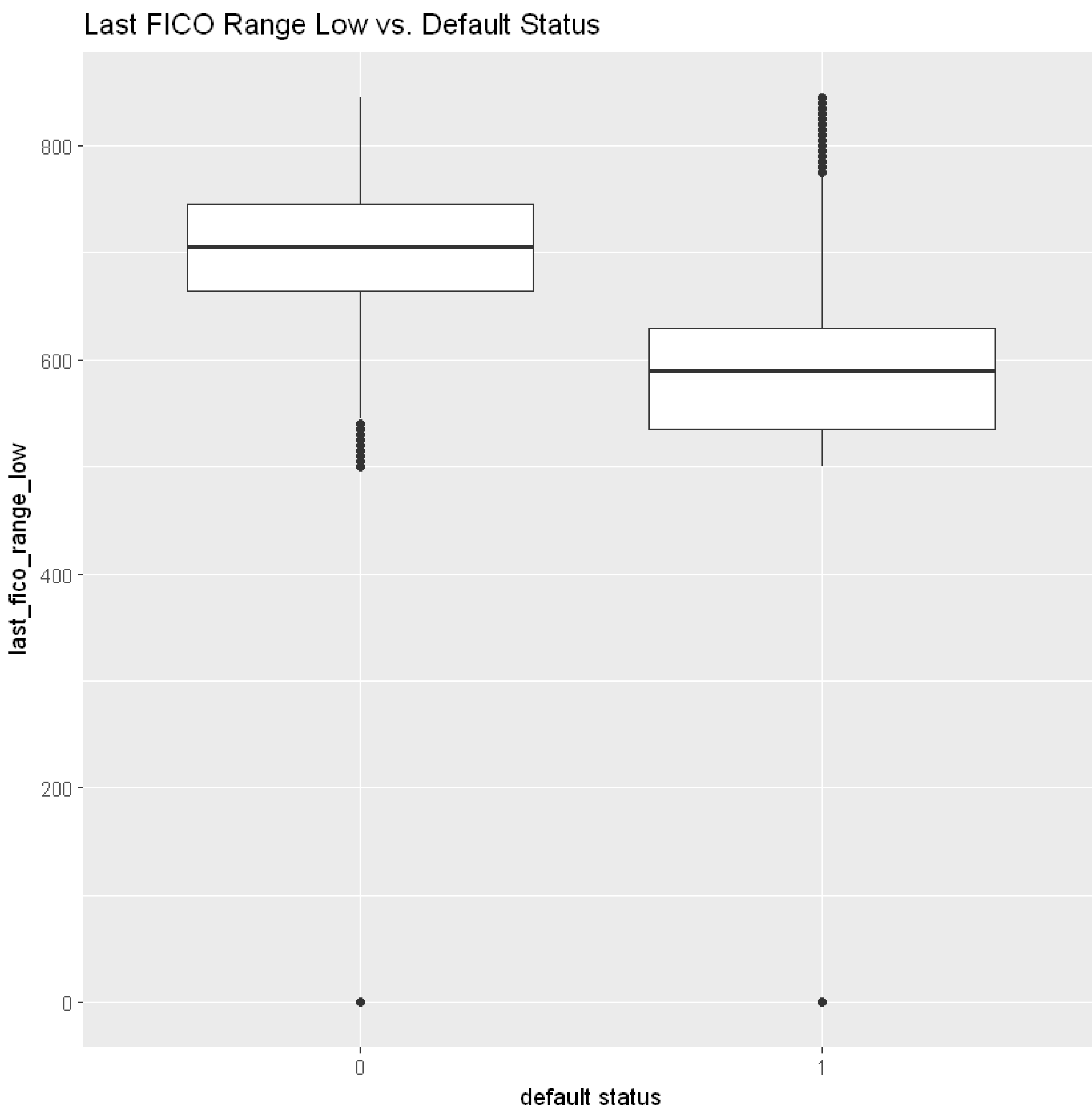
-0.51135373559302

As expected, borrowers who default are more likely to have a lower low range FICO score. The interesting thing about this feature is that it seems to have a much larger impact on default status than the other low range FICO score (fico_range_low) and all other features examined so far, including other FICO scores. I imagine this is because last_fico_range_low is a more recent FICO score and better reflects the borrowers current financial situation. The significant correlation seen in the boxplots is confirmed by the correlation coefficient. We can conclude that last_fico_range_high and default have a medium correlation, the strongest so far.

last_fico_range_low

In [102]:
```
describeBy(lending_club_final$last_fico_range_low, as.factor(lending_club_fina
l$default))

ggplot(lending_club_final, aes(as.factor(default), last_fico_range_low)) + geo
m_boxplot() +
    labs(title="Last FICO Range Low vs. Default Status", x="default status", y
ast_fico_range_low")
```

```
$`0`
   vars      n   mean    sd median trimmed  mad min max range skew kurtosis
X1    1 102149 698.19 81.92    705  704.89 59.3   0 845   845 -3.5    26.62
     se
X1 0.26

$`1`
   vars     n   mean     sd median trimmed   mad min max range  skew kurtosis
X1    1 17124 543.72 175.63    590  586.23 66.72   0 845   845 -2.49     5.08
     se
X1 1.34

attr(,"call")
by.default(data = x, INDICES = group, FUN = describe, type = type)
```



Correlation Coefficient - last_fico_range_low vs. default

In [103]:
```
# find correlation coefficient
cor(lending_club_final$last_fico_range_low, lending_club_final$default, method
earson")
```

-0.473050990490747

As expected having a lower, high range recent FICO score is more likely among borrowers who default. Again similar to last_fico_range_high, last_fico_range_low has a significant impact on whether the borrower is likel to default for similar reasons. Given the correlation coefficient we can conclude that last_fico_range_low and default have a medium correlation.

Conclusion - Exploratory Correlation Analysis

Overall the only features which were significantly correlated with default were last_fico_range_low and last_fico_range_high. This is not a surprise considering FICO is a credit score. Aside from these, two the only other features which any kind of significant correlation were fico_range_high, fico_range_low, and int_rate. I was surprised that most of the features I chose were not highly correlated. It will be interesting to see which features will be chosen by Recursive Feature Elimination (RFE).

Converting Data Types

Some data types are not accepted by various machine learning algorithms.

In [104]:
```
# convert int_rate to numeric
lending_club_final$int_rate <- as.character(lending_club_final$int_rate)
lending_club_final$int_rate <- as.numeric(substr(lending_club_final$int_rate,1
har(lending_club_final$int_rate)-1))

# convert earliest_cr_line to POSIX
lending_club_final$earliest_cr_line <- as.vector(sapply(lending_club_final$ear
liest_cr_line, function(x) paste0(x,"-01")))
lending_club_final$earliest_cr_line <- as.Date(lending_club_final$earliest_cr_
line,"%b-%Y-%d")
lending_club_final$earliest_cr_line <- as.numeric(as.POSIXct(lending_club_fina
l$earliest_cr_line, format="%Y-b%-%d"))

# convert revol_util to numeric
lending_club_final$revol_util <- as.character(lending_club_final$revol_util)
lending_club_final$revol_util <- as.numeric(substr(lending_club_final$revol_ut
il,1,nchar(lending_club_final$revol_util)-1))

# convert last_credit_pull_d to POSIX because as factors they have too many le
vels!
lending_club_final$last_credit_pull_d <- as.vector(sapply(lending_club_final$l
ast_credit_pull_d, function(x) paste0(x,"-01")))
lending_club_final$last_credit_pull_d <- as.Date(lending_club_final$last_credi
t_pull_d,"%b-%Y-%d")
lending_club_final$last_credit_pull_d <- as.numeric(as.POSIXct(lending_club_fi
nal$last_credit_pull_d, format="%Y-b%-%d"))

# remove useless variables
to_remove <- c("url","desc","title","emp_title","id","loan_status","zip_code")
lending_club_final <- lending_club_final[ , !(names(lending_club_final) %in% t
o_remove)]
```

Remove incomplete records

Incomplete records should be removed as long as they are a small portion of the data set < 3%

In [105]:
```
nrows <- nrow(lending_club_final)
ncomplete <- sum(complete.cases(lending_club_final))
print(1-(ncomplete/nrows))
```

```
[1] 0.001224083
```

Since incomplete rows represent < 3% of data set, remove them.

Number of records after incomplete records removed

In [106]:
```
lending_club_final <- lending_club_final[complete.cases(lending_club_final),]

# how many records in data set so far
lcf_before_na_rm <- nrow(lending_club_final)
lcf_before_na_rm
```

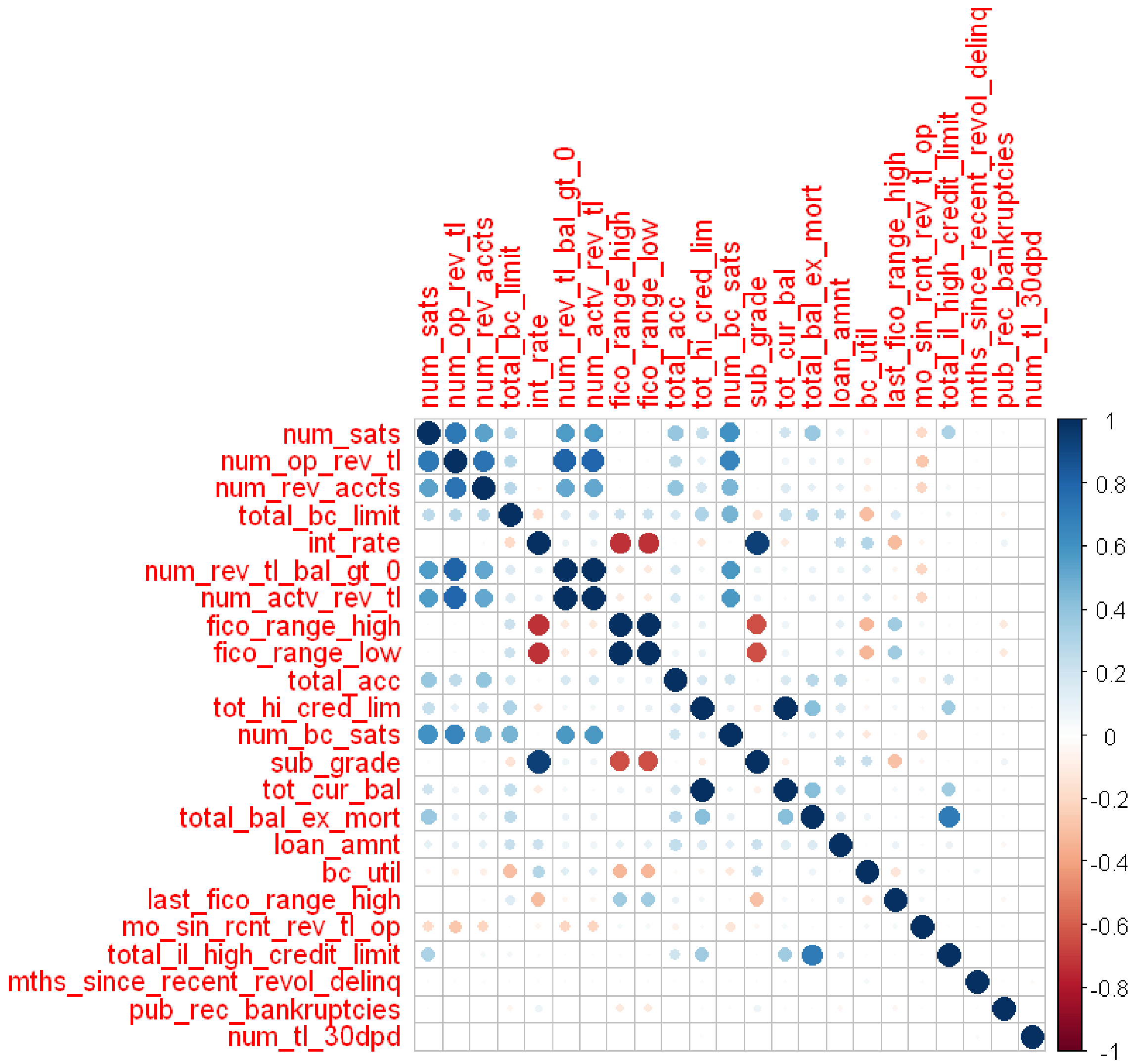
119127

Pair-Wise Correlation

In this section I will find pair-wise correlations. Highly correlated features are redundant an can possibly diminish the performance of a prediction model.

In [107]:

```
lending_club_final_fact_as_num <- lending_club_final

# change factor columns/ features to numeric
indx <- sapply(lending_club_final_fact_as_num, is.factor)
lending_club_final_fact_as_num[indx] <-
lapply(lending_club_final_fact_as_num[indx], function(x) seq_along(levels(x))[


# create subset of dataset with only numeric features
num_feat <- sapply(lending_club_final_fact_as_num, is.numeric)
lending_club_final_num_only <- lending_club_final_fact_as_num[,num_feat]

# eliminate columns/features which are all the same value
unilength <- sapply(lending_club_final_num_only,function(x) length(unique(x)))
lending_club_final_num_only <- subset(lending_club_final_num_only, select=unil
ength>1)

# calculate correlation matrix
correlationMatrix <- cor(lending_club_final_num_only)

# summarize the correlation matrix
#print(correlationMatrix)

# find features that are highly correlated
highlyCorrelated <- findCorrelation(correlationMatrix, cutoff=0.6)
highCorrelationMatrix <- correlationMatrix[highlyCorrelated,highlyCorrelated]

# plot correlation of highly correlated features
library(corrplot)
corrplot(highCorrelationMatrix, method="circle")
```



Dataset Balance

In this section I am going to determine if the dataset is balanced. A balanced dataset would be one where the proportion of loans which defaulted was similar to those who did not, and vice versa for an unbalanced dataset. People normally pay off their loans so the data should be unbalanced in favour of non-default.

How many records per class? 0 = non-default, 1 = default

In [108]:
```
table(lending_club_final$default)
```

```
     0      1 
102039  17088 
```

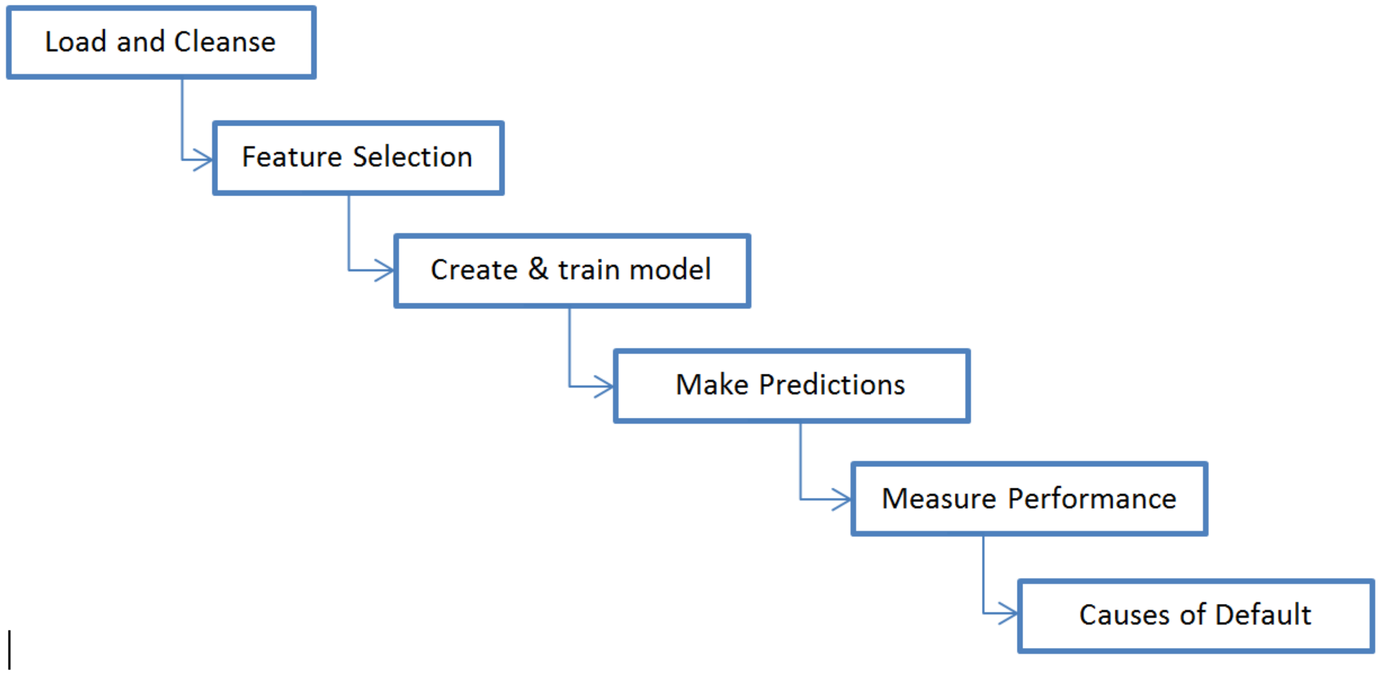
What is the proportion per class? 0 = non-default, 1 = default

In [109]:
```
prop.table(table(lending_club_final$default))
```

```
        0         1 
0.8565564 0.1434436 
```

As we can see the dataset is unbalanced and will have to be balanced before creating our predictive models.

Approach



Step 1: Load, Cleanse and Balance

Load dataset and perform initial cleansing. Cleansing will entail removing features which only contain NA values, imputing values into features which have a large proportion of NA values, and changing feature data types in order to perform correlation analysis. Imputing will be done by populating NA's with the column average. Finally, the dataset will tested for imbalance and corrected if necessary.

Step 2: Feature Selection

Determine which features are significant and which will be used in the predictive models. Some feature analysis will be performed in order to understand the data but Recursive Feature Elimination (RFE) will be relied upon to determine the optimal feature subset.

Step 3: Create and Train Model

Using the pruned dataset consisting of features determined from steps 1 and 2, create a test and training dataset. Create predictive models on the training dataset for the following algorithms: Random Forests, Bagged Decision Trees, and Support Vector Machines (SVM). Note: Each model is created using the same training dataset.

Step 4: Make Predictions

Use the models on the test dataset and record the results.

Step 5: Measure Performance

This is a sub-step of step 5. For each result set in step 5, measure the Percentage Correctly Classified (PCC), the area under a ROC curve (AUC), and accuracy, then compare the performance of each of the classification algorithms.

Step 6: Causes of Default

Using the knowledge gained from all of the previous steps, determine the causes of loan default.

Predictive Models

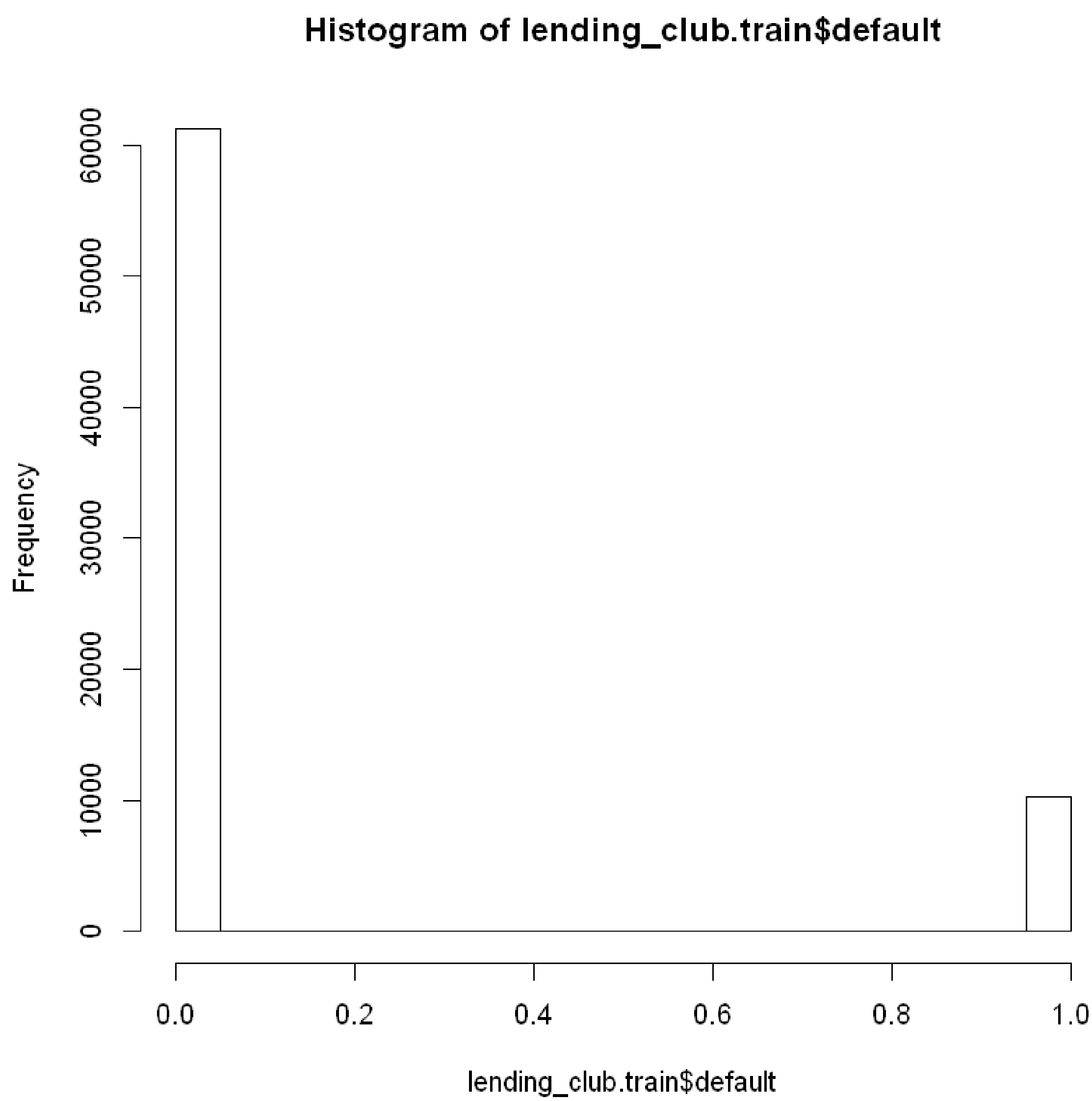
Create training and test data sets

In [110]:
```
train_rows <- sample(nrow(lending_club_final),(nrow(lending_club_final)*0.6))
lending_club.train <- lending_club_final[train_rows,]
lending_club.test <- lending_club_final[-train_rows,]
```

Before balancing the dataset

In [111]:
```
# How many records per class?
table(lending_club.train$default)

# Plot histogram of class distribution
hist(lending_club.train$default)

# Percentage of total dataset per class (0= non-default, 1= default)
prop.table(table(lending_club.train$default))

hist(lending_club.train$default)
```

```
    0     1 
61252 10224 

       0        1 
0.856959 0.143041 
```



Balancing the Training Dataset - Using SMOTE

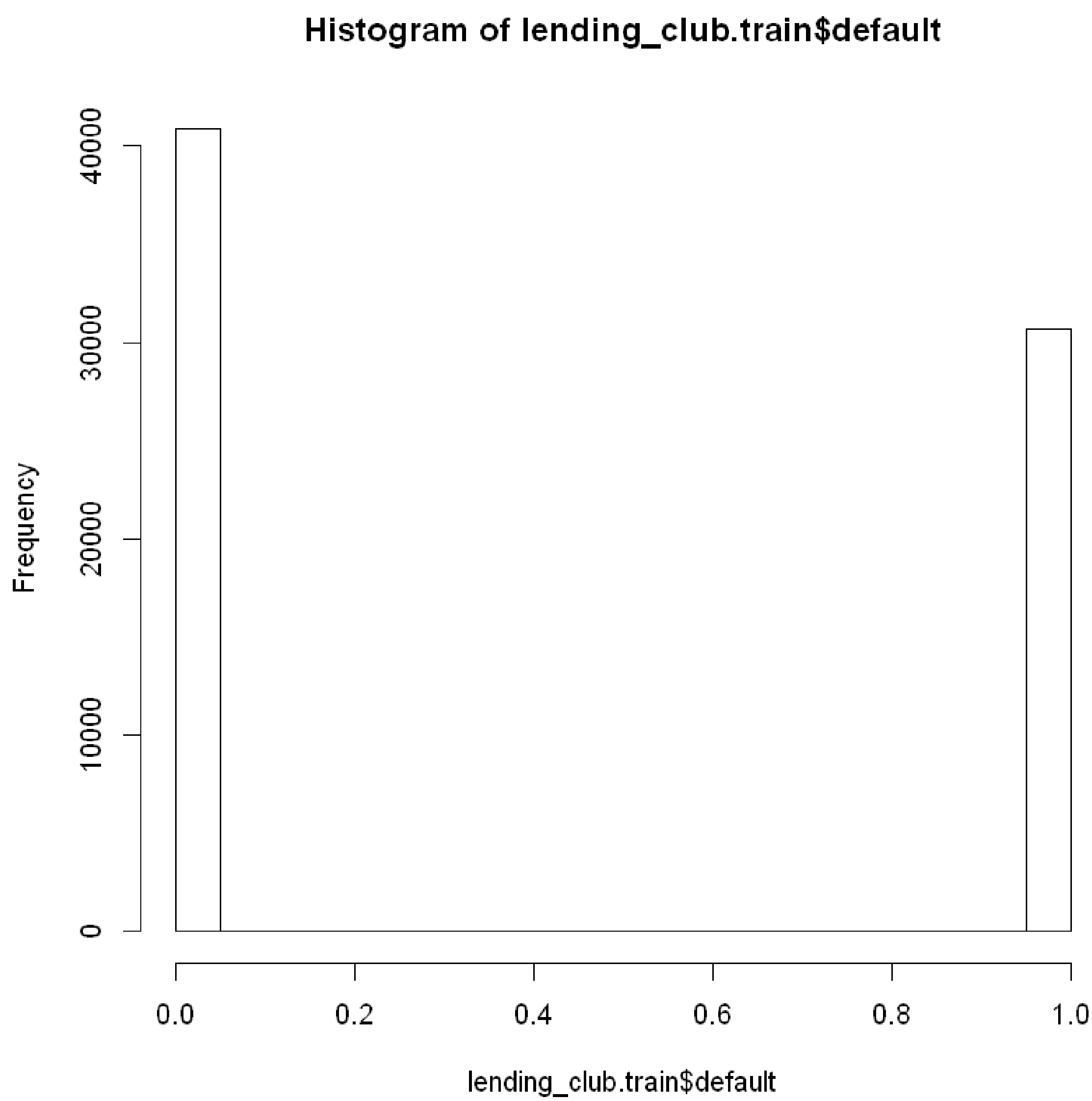
In [112]:
```
#install.packages("unbalanced", repos='http://cran.us.r-project.org',dependenc
ies = TRUE)
#library(unbalanced)

n <- ncol(lending_club.train)
y <- as.factor(lending_club.train$default)
x <- lending_club.train[ ,-n]
lending_club.train.smote <- ubSMOTE(X=x, Y=y)
lending_club.train.smote$default <- as.numeric(as.character(lending_club.trai
n.smote$Y))
```

In [113]:
```
lending_club.train <- cbind(lending_club.train.smote$X, default=lending_club.t
rain.smote$default)
```

After balancing the dataset

In [114]:
```
# How many records per class?
table(lending_club.train$default)

# Plot histogram of class distribution
hist(lending_club.train$default)

# Percentage of total dataset per class (0= non-default, 1= default)
prop.table(table(lending_club.train$default))

hist(lending_club.train$default)
```

```
    0     1
40896 30672

        0         1
0.5714286 0.4285714
```



Random Forest

Train model and make predictions - Pre-RFE

I am training the random forest algorithm against all of the variables/features. This model will be a baseline to compare to others.

In [115]:
```
library(caret)
#install.packages("e1071", repos='http://cran.us.r-project.org',dependencies =
 TRUE)
library(e1071)
library(randomForest)
set.seed(9)

ptm <- proc.time()
#########################
rf.fit <- randomForest(as.factor(default)~., data=lending_club.train, importan
ce=TRUE, ntree=400)

# Generate predictions based on model
lending_club.test$default.pred <- predict(rf.fit,lending_club.test)
########################
rf.pre.time <- proc.time() - ptm
```

Measure performance - Random Forest Pre-RFE

In [116]:
```
# Create Confusion Matrix
cm.pre_rfe <- confusionMatrix(lending_club.test$default.pred,lending_club.test
fault)
cm.pre_rfe

# area under a ROC curve
#auc(lending_club.test$default,as.numeric(lending_club.test$default.pred))
```

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 39421  2515
         1  1366  4349

               Accuracy : 0.9186
                 95% CI : (0.9161, 0.921)
    No Information Rate : 0.856
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.645
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9665
            Specificity : 0.6336
         Pos Pred Value : 0.9400
         Neg Pred Value : 0.7610
             Prevalence : 0.8560
         Detection Rate : 0.8273
   Detection Prevalence : 0.8801
      Balanced Accuracy : 0.8001

       'Positive' Class : 0
```

Recursive Feature Elimination (RFE)

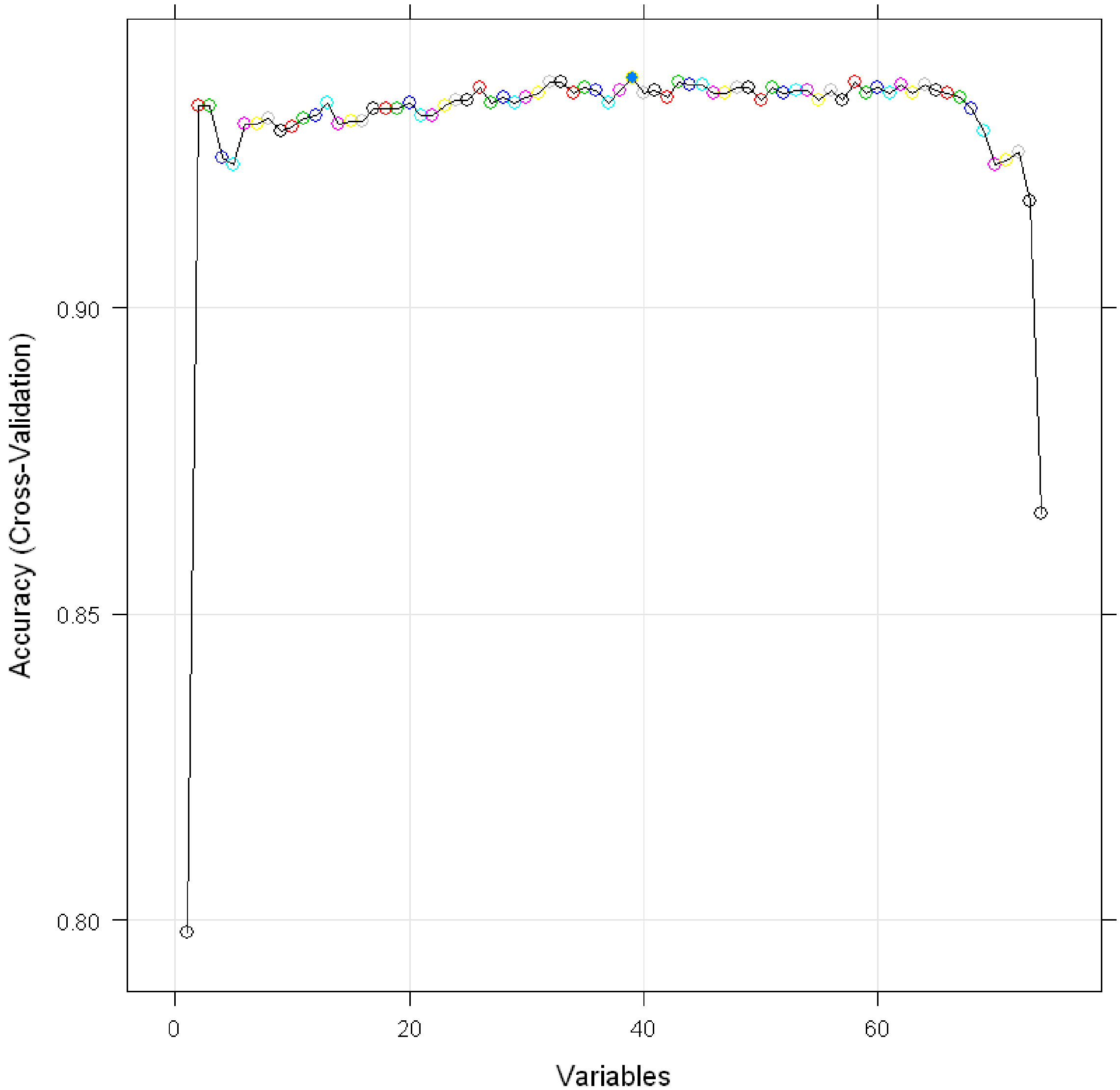
So far we have not considered significance when cleansing the data set. RFE analysis will identify a data set subset for optimal results.

In [119]:
```
# Let's use RFE to see if we can prune some variables/features and hopefully g
et a better result
set.seed(77)

# 10-fold cross-validation
control <- rfeControl(functions=rfFuncs, method="cv", number=10)
lending_club.train.rfe <- lending_club.train[sample(nrow(lending_club.train),2
000),]
rfe.train <- rfe(lending_club.train.rfe[,1:74], as.factor(lending_club.train.r
fe[,75]), sizes=1:74, rfeControl=control)

# how big is the optimal variable subset?
print(rfe.train$bestSubset)
```

```
[1] 39
```

Plot the number of variables vs. RMSE

In [120]:
```
plot(rfe.train, type=c("g", "o"), cex = 1.0, col = 1:74)
```



What specific variables/features comprise the optimal subset?

In [121]:
```
predictors(rfe.train)
```

"last_credit_pull_d" "last_fico_range_low" "last_fico_range_high" "inq_last_6mths" "revol_util" "num_tl_120dpd_2m" "num_tl_30dpd" "tot_coll_amt" "emp_length" "num_tl_90g_dpd_24m" "revol_bal" "int_rate" "fico_range_high" "fico_range_low" "annual_inc" "dti" "member_id" "term" "num_accts_ever_120_pd" "delinq_2yrs" "earliest_cr_line" "total_acc" "loan_amnt" "percent_bc_gt_75" "mort_acc" "installment" "purpose" "bc_util" "mths_since_last_delinq" "total_bal_ex_mort" "mths_since_recent_bc" "open_acc" "pct_tl_nvr_dlq" "num_sats" "acc_open_past_24mths" "grade" "bc_open_to_buy" "mths_since_recent_inq" "total_bc_limit"

In [122]:
```
predictors <- predictors(rfe.train)
formula <- paste("as.factor(default)",paste(predictors, collapse=" + "), sep="

formula <- as.formula(formula)
formula
```

```
as.factor(default) ~ last_credit_pull_d + last_fico_range_low +
    last_fico_range_high + inq_last_6mths + revol_util + num_tl_120dpd_2m +
    num_tl_30dpd + tot_coll_amt + emp_length + num_tl_90g_dpd_24m +
    revol_bal + int_rate + fico_range_high + fico_range_low +
    annual_inc + dti + member_id + term + num_accts_ever_120_pd +
    delinq_2yrs + earliest_cr_line + total_acc + loan_amnt +
    percent_bc_gt_75 + mort_acc + installment + purpose + bc_util +
    mths_since_last_delinq + total_bal_ex_mort + mths_since_recent_bc +
    open_acc + pct_tl_nvr_dlq + num_sats + acc_open_past_24mths +
    grade + bc_open_to_buy + mths_since_recent_inq + total_bc_limit
```

Train Random Forest model Post-RFE and generate predictions

In [123]:
```
# Redo the Random Forest model with the optimal subset according to RFE
set.seed(44)

ptm <- proc.time()
#########################
rf.fit.opt <- randomForest(formula, data=lending_club.train, ntree=400,
type='classification')

lending_club.test$default.pred.opt <- predict(rf.fit.opt,lending_club.test)
########################
rf.post.time <- proc.time() - ptm
```

Measure performance - Random Forest Post-RFE

In [124]:

```
# Create Confusion Matrix
cm.post_rfe <- confusionMatrix(lending_club.test$default.pred.opt,lending_clu
b.test$default)
cm.post_rfe
```

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 38180   660
         1  2607  6204

               Accuracy : 0.9314
                 95% CI : (0.9291, 0.9337)
    No Information Rate : 0.856
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7513
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9361
            Specificity : 0.9038
         Pos Pred Value : 0.9830
         Neg Pred Value : 0.7041
             Prevalence : 0.8560
         Detection Rate : 0.8012
   Detection Prevalence : 0.8151
      Balanced Accuracy : 0.9200

       'Positive' Class : 0
```

Bagged Decision Trees

Create model and make predictions

In [125]:
```
set.seed(3)
# BAGGED DECISION TREES
library(rpart)
#install.packages("adabag", repos='http://cran.us.r-project.org',dependencies
 = TRUE)
library(adabag)
lending_club.train$default.factor <- as.factor(lending_club.train$default)
lending_club.test$default.factor <- as.factor(lending_club.test$default)
formula_bdt <- paste("default.factor",paste(predictors,collapse=" +
"),sep="~")

ptm <- proc.time()
##########################
# mfinal indicates total number of trees grown
# and minsplit is the minimum number of observations that must exist in a node
 in order for a split to be attempted
bdt.bagging <- bagging(formula_bdt, mfinal=500, control=rpart.control(minsplit
 = 50), data=lending_club.train)

# make predictions
bdt.bagging.pred <- predict.bagging(bdt.bagging, newdata=lending_club.test)
#########################
bdt.time <- proc.time() - ptm
```

Bagged Decision Trees - Measure Performance

In [126]:
```
# Create Confusion Matrix
cm.bdt <- confusionMatrix(bdt.bagging.pred$class,lending_club.test$default)
cm.bdt

# find Area Under a ROC Curve (AUC)
#auc(lending_club.test$default,as.numeric(bdt.bagging.pred$class))
```

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 38025  1159
         1  2762  5705

               Accuracy : 0.9177
                 95% CI : (0.9152, 0.9202)
    No Information Rate : 0.856
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6959
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9323
            Specificity : 0.8311
         Pos Pred Value : 0.9704
         Neg Pred Value : 0.6738
             Prevalence : 0.8560
         Detection Rate : 0.7980
   Detection Prevalence : 0.8223
      Balanced Accuracy : 0.8817

       'Positive' Class : 0
```

Support Vector Machines

In [127]:
```
set.seed(10)

ptm <- proc.time()
#########################
# create SVM model
lc.svm.fit <- svm(formula, data=lending_club.train)

# make predictons
lending_club.test$default.pred.svm <- predict(lc.svm.fit,lending_club.test)
########################
svm.time <- proc.time() - ptm

# Create Confusion Matrix
cm.svm <- confusionMatrix(lending_club.test$default.pred.svm,lending_club.test
fault)
cm.svm
```

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 37322  1391
         1  3465  5473

               Accuracy : 0.8981
                 95% CI : (0.8953, 0.9008)
    No Information Rate : 0.856
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6329
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9150
            Specificity : 0.7973
         Pos Pred Value : 0.9641
         Neg Pred Value : 0.6123
             Prevalence : 0.8560
         Detection Rate : 0.7832
   Detection Prevalence : 0.8124
      Balanced Accuracy : 0.8562

       'Positive' Class : 0
```

Visualize the Prediction Model Using Conditional Inference Trees (CTree)

In [128]:
```
lending_club.init_sol <- lending_club_final

# train and test datasets
train_rows <- sample(nrow(lending_club.init_sol),(nrow(lending_club.init_sol)*

lending_club.init_sol.train <- lending_club.init_sol[train_rows,]
lending_club.init_sol.test <- lending_club.init_sol[-train_rows,]

lending_club.init_sol.train$default <- factor(lending_club.init_sol.train$defa
ult, levels=c(0,1), labels=c('PAID', 'DEFAULT'))
lending_club.init_sol.test$default <- factor(lending_club.init_sol.test$defaul
t, levels=c(0,1), labels=c('PAID', 'DEFAULT'))
```

In [129]:
```
# create model
default_pred.ctree <- ctree(formula, data=lending_club.init_sol.train,
                               controls=ctree_control(testtype="Bonferroni", m
insplit=4000, minbucket=4000))

# save model of tree to disk
png("ctree.png", res=80, height=900, width=2000)
   plot(default_pred.ctree)
dev.off()
```

png: 2

Measure Accuracy - CTree

In [130]:
```
lending_club.init_sol.test$default.pred <- predict(default_pred.ctree, newdata
nding_club.init_sol.test, type='response')
cm.ctree <- confusionMatrix(lending_club.init_sol.test$default.pred, lending_c
lub.init_sol.test$default, positive="PAID")
```

Compile Prediction Performance Results

In [131]:
```
# poplulate Accuracy (PCC)
acc <- cbind(as.numeric(cm.post_rfe$overall)[1],as.numeric(cm.bdt$overall)
[1],as.numeric(cm.svm$overall)[1])

# populate Kappa
kap <- cbind(as.numeric(cm.post_rfe$overall)[2],as.numeric(cm.bdt$overall)
[2],as.numeric(cm.svm$overall)[2])

# populate Sensitivity
sen <- cbind(as.numeric(cm.post_rfe$byClass)[1],as.numeric(cm.bdt$byClass)
[1],as.numeric(cm.svm$byClass)[1])

# populate Specificity
spec <- cbind(as.numeric(cm.post_rfe$byClass)[2],as.numeric(cm.bdt$byClass)[2]
numeric(cm.svm$byClass)[2])

# populate AUC
auc <- cbind(as.numeric(cm.post_rfe$byClass)[11],as.numeric(cm.bdt$byClass)[11
s.numeric(cm.svm$byClass)[11])

perf.mat <- rbind(acc,kap,sen,spec,auc)
rownames(perf.mat) <- c("PCC (1)","Kappa (2)","Sensitivity (3)","Specificity
 (4)","AUC (5)")
colnames(perf.mat) <- c("Random Forest - Post RFE","Bagged Decision Trees","SV
M")
```

Compile Time Performance Results

In [132]:
```
rf.mat <- as.matrix(rf.post.time)
svm.mat <- as.matrix(svm.time)
bdt.mat <- as.matrix(bdt.time)

et <- cbind(rf.mat[3], svm.mat[3], bdt.mat[3])
colnames(et) <- c("Random Forests","SVM","Bagged DT")
```

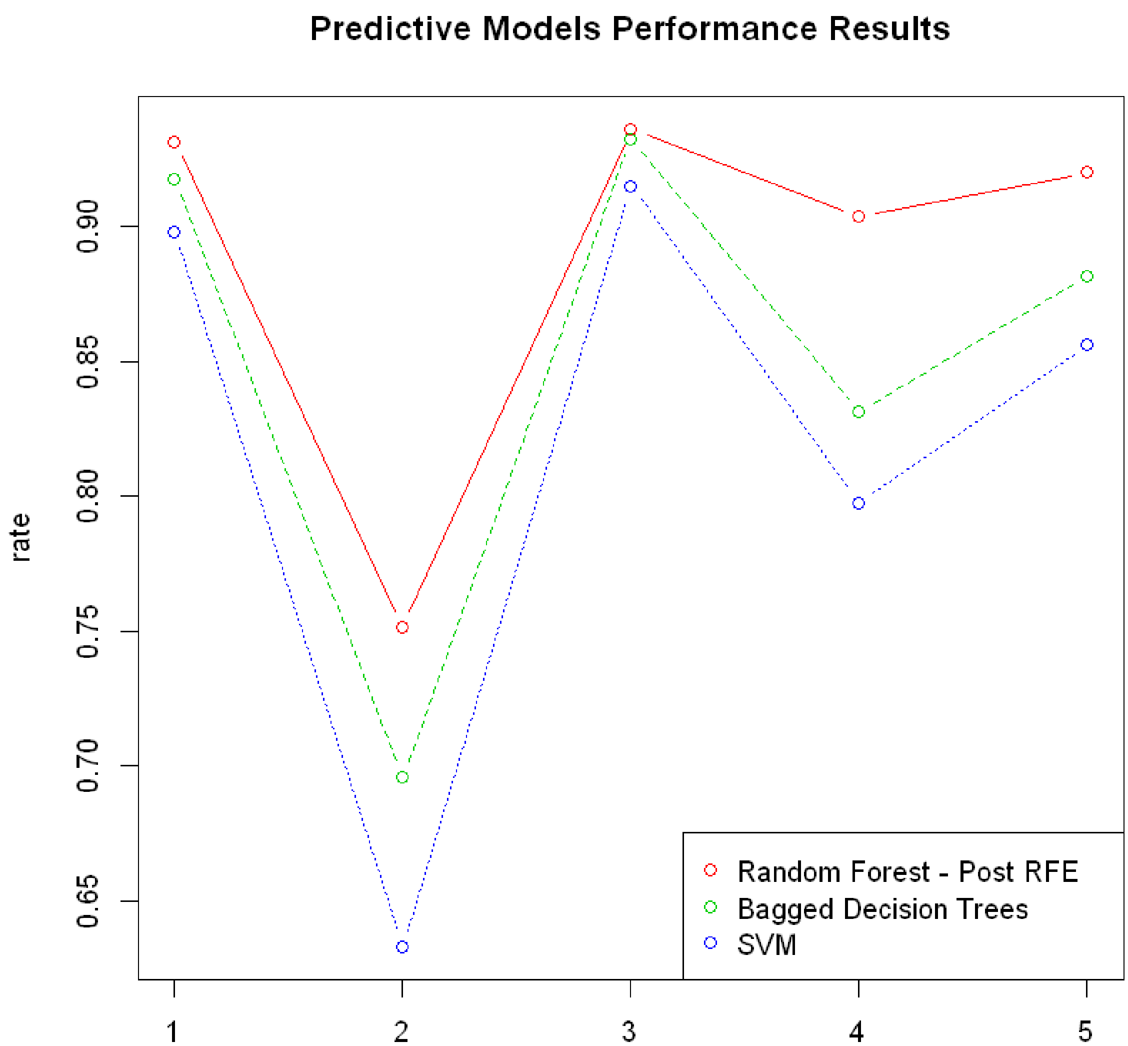
Results

Predictive Models Performance

This section will summarize the results of the prediction performance measures for all of the classification algorithms covered in this project. The measures of performance to be compared are Accuracy (Percent Correctly Classified), Area Under a ROC Curve (AUC), as well as Sensitivity and Specificity.

In [133]:
```
matplot(perf.mat, type = c("b"),pch=1,col = 2:4, ylab="rate", main="Predictive
 Models Performance Results")
legend("bottomright", legend = c("Random Forest - Post RFE","Bagged Decision T
rees","SVM"), col=2:4, pch=1)
perf.mat
```

	Random Forest - Post RFE	Bagged Decision Trees	SVM
PCC (1)	0.9314390	0.9177142	0.8980924
Kappa (2)	0.7513055	0.6958509	0.6328724
Sensitivity (3)	0.9360826	0.9322823	0.9150465
Specificity (4)	0.9038462	0.8311480	0.7973485
AUC (5)	0.9199644	0.8817152	0.8561975



Predictive Models Results Summary

Over all of the metrics used, the best performer is Random Forests, with Bagged Decision Trees in second and SVM last. According to Benchmarking state-of-the-art classification algorithms for credit scoring: A ten-year update[2], which compared Random Forests, Bagged Decision Trees and SVM among many others, Random Forests and Bagged Decision Trees outperformed SVM. The results here further reinforce that.

Description of Metrics Used

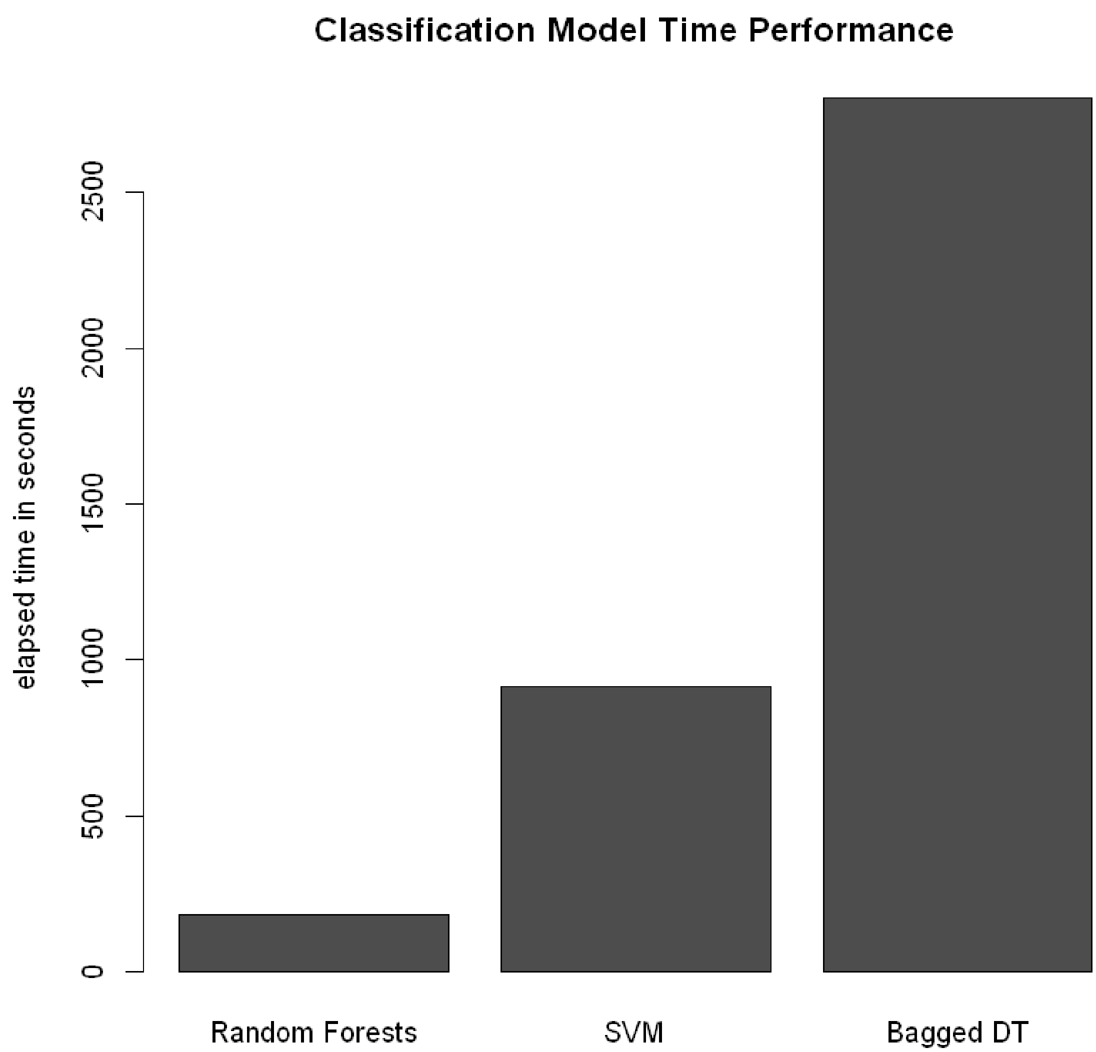
The first metric is Percentage Correctly Classified (PCC), which measures the classified loans as a percentage of all loans. The second is Kappa, which is a measure of agreement between two raters. This measure is considered more robust than PCC and other similar metrics[4]. The third is Sensitivity which is the proportion of borrowers who were predicted as not defaulting versus all the borrowers who actually did not default. In other words it measures the success rate of predicting who will not default. The fourth is Specificity which is the proportion of borrowers who were predicted as defaulting versus all the borrowers who did default. This metric is especially important since it captures the success rate of predicting who will default. The final metric is Area Under a ROC Curve which measures the relationship between the true positive rate (TPR) and the false positive rate (FPR). An AUC of 1 means that all borrowers predicted not default did not default in reality. An AUC of 0.5 would indicate that of the borrowers predicted not to default, half actually did while the other half did not.

Time Performance

This section will summarize the time performance for each predictive model.

In [134]:
```
et
barplot(et, main="Classification Model Time Performance", ylab="elapsed time i
n seconds")
```

Random Forests	SVM	Bagged DT
183.00	913.88	2803.99



Time Performance Summary

The time performance results show that Random Forests outperform Support Vector Machines and Bagged Decision Trees. Again, the best time performance comes from Random Forests meanwhile Bagged Decision Trees alogorithm unacceptably slow, especially for systems that require quick turnaround such as targeted display advertising.

Important Features for Prediction

This section lists the features in the dataset which are the most important for predicting loan default as determined by Recursive Feature Elimination. These features should be collected for each potential borrower to accurately determine his/her likelihood of default.

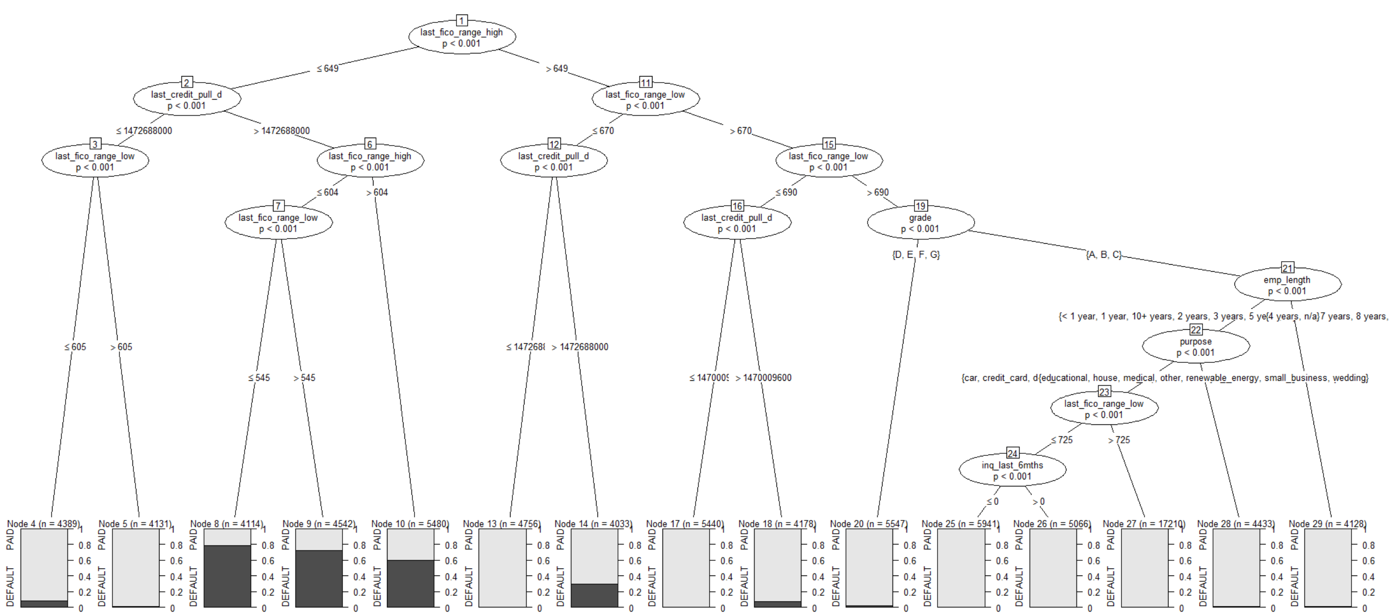
In [135]:
```
dataDict <- read.table("LCDataDictionary.csv", header = TRUE, sep=",")
dataDict[which(dataDict$Feature %in% predictors(rfe.train)),]
```

	Feature	Description
2	acc_open_past_24mths	Number of trades opened in past 24 months.
5	annual_inc	The self-reported annual income provided by the borrower during registration.
9	bc_open_to_buy	Total open to buy on revolving bankcards.
10	bc_util	Ratio of total current balance to high credit/credit limit for all bankcard accounts.
14	delinq_2yrs	The number of 30+ days past-due incidences of delinquency in the borrowers credit file for the past 2 years delinq_amnt,The past-due amount owed for the accounts on which the borrower is now delinquent. desc,Loan description provided by the borrower dti,A ratio calculated using the borrower?s total monthly debt payments on the total debt obligations excluding mortgage and the requested LC loan divided by the borrower?s self-reported monthly income. dti_joint,A ratio calculated using the co-borrowers total monthly payments on the total debt obligations excluding mortgages and the requested LC loan divided by the co-borrowers combined self-reported monthly income earliest_cr_line,The month the borrowers earliest reported credit line was opened
15	emp_length	Employment length in years. Possible values are between 0 and 10 where 0 means less than one year and 10 means ten or more years.
17	fico_range_high	The upper boundary range the borrower?s FICO at loan origination belongs to.
18	fico_range_low	The lower boundary range the borrower?s FICO at loan origination belongs to.
21	grade	LC assigned loan grade
28	inq_last_6mths	The number of inquiries in past 6 months (excluding auto and mortgage inquiries)
29	installment	The monthly payment owed by the borrower if the loan originates.
30	int_rate	Interest Rate on the loan
32	last_credit_pull_d	The most recent month LC pulled credit for this loan
33	last_fico_range_high	The upper boundary range the borrower?s last FICO pulled belongs to.
34	last_fico_range_low	The lower boundary range the borrower?s last FICO pulled belongs to.

	Feature	Description
37	loan_amnt	The listed amount of the loan applied for by the borrower. If at some point in time the credit department reduces the loan amount then it will be reflected in this value.
40	member_id	A unique LC assigned Id for the borrower member.
45	mort_acc	Number of mortgage accounts.
46	mths_since_last_delinq	The number of months since the borrowers last delinquency. mths_since_last_major_derog,Months since most recent 90-day or worse rating mths_since_last_record,The number of months since the last public record. mths_since_rcnt_il,Months since most recent installment accounts opened mths_since_recent_bc,Months since most recent bankcard account opened. mths_since_recent_bc_dlq,Months since most recent bankcard delinquency mths_since_recent_inq,Months since most recent inquiry. mths_since_recent_revol_delinq,Months since most recent revolving delinquency. next_pymnt_d,Next scheduled payment date num_accts_ever_120_pd,Number of accounts ever 120 or more days past due num_actv_bc_tl,Number of currently active bankcard accounts num_actv_rev_tl,Number of currently active revolving trades num_bc_sats,Number of satisfactory bankcard accounts num_bc_tl,Number of bankcard accounts num_il_tl,Number of installment accounts num_op_rev_tl,Number of open revolving accounts num_rev_accts,Number of revolving accounts num_rev_tl_bal_gt_0,Number of revolving trades with balance >0 num_sats,Number of satisfactory accounts num_tl_120dpd_2m,Number of accounts currently 120 days past due (updated in past 2 months) num_tl_30dpd,Number of accounts currently 30 days past due (updated in past 2 months) num_tl_90g_dpd_24m,Number of accounts 90 or more days past due in last 24 months num_tl_op_past_12m,Number of accounts opened in past 12 months open_acc,The number of open credit lines in the borrowers credit file.
55	pct_tl_nvr_dlq	Percent of trades never delinquent
56	percent_bc_gt_75	Percentage of all bankcard accounts > 75% of limit.
60	purpose	A category provided by the borrower for the loan request.
63	revol_bal	Total credit revolving balance
64	revol_util	Revolving line utilization rate or the amount of credit the borrower is using relative to all available revolving credit.
67	term	The number of payments on the loan. Values are in months and can be either 36 or 60.
69	tot_coll_amt	Total collection amounts ever owed

	Feature	Description
72	total_acc	The total number of credit lines currently in the borrowers credit file total_bal_ex_mort,Total credit balance excluding mortgage total_bal_il,Total current balance of all installment accounts total_bc_limit,Total bankcard high credit/credit limit total_cu_tl,Number of finance trades total_il_high_credit_limit,Total installment high credit/credit limit total_pymnt,Payments received to date for total amount funded total_pymnt_inv,Payments received to date for portion of total amount funded by investors total_rec_int,Interest received to date total_rec_late_fee,Late fees received to date total_rec_prncp,Principal received to date total_rev_hi_lim ÿ,Total revolving high credit/credit limit url,URL for the LC page with listing data. verification_status,Indicates if income was verified by LC not verified or if the income source was verified verified_status_joint,Indicates if the co-borrowers joint income was verified by LC not verified or if the income source was verified

Loan Approval Decision Process



This decision tree above can be used to predict whether a potential borrower will or will not default on their loan. By starting at the top node of the tree and by following the decisions you eventually arrive at a probability of default or non-default (paid). By studying this tree we can glean some important insights:

The last credit pull date (last_credit_pull_d) is an indicative feature when sizing up a potential borrower. If a potential borrower has had a credit pull within 4 years and 7 months ago or earlier this is of significant concern.

Overall the most indicative feature for default is the high range of the last FICO score (last_fico_range_high). If last_fico_range_high is greater than 649 the probability of default is minimal. The only exception to that would be when the borrower has had a credit pull within the last 4 years and 7 months. Meanwhile, if last_fico_range_high is less than 649 and the last credit pull date is 4 years and 7 months ago or earlier, then the probability of default is about 60% or higher.

The most dangerous potential borrower is someone with a last_fico_range_high of less than 545. Potential borrowers with such a low FICO score are at least 70% likely to default.

NOTES: **This tree is based on the Conditional Inference Trees model generated above (see heading "Visualize the Prediction Model Using Conditional Inference Trees (CTree)"). It's performance measures are displayed below.**

This tree has been compressed and does not show all of the possible decision points. This has been done for display purposes. A larger more detailed tree can be generated for a more in-depth analysis.

*Please note that 147000960 translates to Jan. 27, 2012 and since the data is current to Sept. 1, 2016, it roughly translates to 4 years and 7 months ago.

In [136]:
```
cm.ctree
```

```
Confusion Matrix and Statistics

          Reference
Prediction  PAID DEFAULT
   PAID    28758     936
   DEFAULT  1919    4126

               Accuracy : 0.9201
                 95% CI : (0.9173, 0.9229)
    No Information Rate : 0.8584
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6961
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9374
            Specificity : 0.8151
         Pos Pred Value : 0.9685
         Neg Pred Value : 0.6825
             Prevalence : 0.8584
         Detection Rate : 0.8047
   Detection Prevalence : 0.8309
      Balanced Accuracy : 0.8763

       'Positive' Class : PAID
```

Conclusion

The purpose of this project is to determine the characteristics or features of a borrower which are important in determining their creditworthiness and create a framework by which a lender could examine those features and come to a conclusion as to whether a potential borrower is or is not likely to default. The characteristics/features provided above help a lender determine what kind of data he or she needs to compile in order to screen potential borrowers. These features are listed under the **Important Features for Prediction** section above. Once the lender has gathered all of the data he or she will need a framework to turn the raw data into a decision. The framework provided in this project is the decision tree provided in the **Loan Approval Decision Process** section above. By following the decision tree and asking each question at each node in regards to a potential borrower, the lender will eventually arrive at general probability of default. With this information, he or she can decide on whether or not to issue the loan. If the borrower decides to lend, the potential rate of default can help determine the level of risk and the appropriate rate of interest to compensate. In addition to creating a decision process for each potential new borrower, the decision tree provides general insights which can be used for making quick decisions.

Overall the best performing classification model was Random Forests. With an accuracy rate of 93%, it can be generally relied upon for consistently accurate predictions.

Real Life Example

In this section I am going to take the important characteristics/features from a few individual borrowers in the Lending Club dataset and make a predictions using the Random Forest-Post RFE model.

In [140]:
```
lc_tiny_test <- lending_club.test[sample(nrow(lending_club.test),15),1:75]

lc_tiny_test$default.pred <- predict(rf.fit.opt,lc_tiny_test)

lc_tiny_test <- cbind(lc_tiny_test[,which(colnames(lc_tiny_test) %in% predicto
rs(rfe.train))],
                      default=lc_tiny_test[,75],
default.pred=lc_tiny_test[,76])
lc_tiny_test
```

	annual_inc	delinq_2yrs	inq_last_6mths	mths_since_last_delinq	open_acc	tota
141468	32000	0	0	36.1423584259484	12	18
159379	35000	0	1	36.1423584259484	9	9
181214	365000	0	4	36.1423584259484	8	33
173670	24000	0	0	36.1423584259484	10	25
7239	48000	0	1	36.1423584259484	11	17
182874	61548	1	0	2	8	16
163293	63000	0	1	36.1423584259484	10	32
7462	66000	0	0	36.1423584259484	9	31
151414	141696	0	1	36.1423584259484	13	27
12778	106921	0	0	36.1423584259484	7	21
154552	31820	0	0	36.1423584259484	10	17
140533	1e+05	0	1	36.1423584259484	23	34
2968	20400	0	1	27	7	8
150520	63945	0	0	36.1423584259484	11	23
17345	193000	0	1	32	12	48

Appendix A - Data Dictionary

In [141]:
```
dataDict
```

	Feature	Description
1	acc_now_delinq	The number of accounts on which the borrower is now delinquent.
2	acc_open_past_24mths	Number of trades opened in past 24 months.
3	addr_state	The state provided by the borrower in the loan application
4	all_util	Balance to credit limit on all trades
5	annual_inc	The self-reported annual income provided by the borrower during registration.
6	annual_inc_joint	The combined self-reported annual income provided by the co-borrowers during registration
7	application_type	Indicates whether the loan is an individual application or a joint application with two co-borrowers
8	avg_cur_bal	Average current balance of all accounts
9	bc_open_to_buy	Total open to buy on revolving bankcards.
10	bc_util	Ratio of total current balance to high credit/credit limit for all bankcard accounts.
11	chargeoff_within_12_mths	Number of charge-offs within 12 months
12	collection_recovery_fee	post charge off collection fee
13	collections_12_mths_ex_med	Number of collections in 12 months excluding medical collections
14	delinq_2yrs	The number of 30+ days past-due incidences of delinquency in the borrowers credit file for the past 2 years delinq_amnt,The past-due amount owed for the accounts on which the borrower is now delinquent. desc,Loan description provided by the borrower dti,A ratio calculated using the borrower?s total monthly debt payments on the total debt obligations excluding mortgage and the requested LC loan divided by the borrower?s self-reported monthly income. dti_joint,A ratio calculated using the co-borrowers total monthly payments on the total debt obligations excluding mortgages and the requested LC loan divided by the co-borrowers combined self-reported monthly income earliest_cr_line,The month the borrowers earliest reported credit line was opened
15	emp_length	Employment length in years. Possible values are between 0 and 10 where 0 means less than one year and 10 means ten or more years.
16	emp_title	The job title supplied by the Borrower when applying for the loan.*

	Feature	Description
17	fico_range_high	The upper boundary range the borrower?s FICO at loan origination belongs to.
18	fico_range_low	The lower boundary range the borrower?s FICO at loan origination belongs to.
19	funded_amnt	The total amount committed to that loan at that point in time.
20	funded_amnt_inv	The total amount committed by investors for that loan at that point in time.
21	grade	LC assigned loan grade
22	home_ownership	The home ownership status provided by the borrower during registration. Our values are: RENT OWN MORTGAGE OTHER.
23	id	A unique LC assigned ID for the loan listing.
24	il_util	Ratio of total current balance to high credit/credit limit on all install acct
25	initial_list_status	The initial listing status of the loan. Possible values are ? W F
26	inq_fi	Number of personal finance inquiries
27	inq_last_12m	Number of credit inquiries in past 12 months
28	inq_last_6mths	The number of inquiries in past 6 months (excluding auto and mortgage inquiries)
29	installment	The monthly payment owed by the borrower if the loan originates.
30	int_rate	Interest Rate on the loan
...	...	...
44	mo_sin_rcnt_tl	Months since most recent account opened
45	mort_acc	Number of mortgage accounts.

	Feature	Description
46	mths_since_last_delinq	The number of months since the borrowers last delinquency. mths_since_last_major_derog,Months since most recent 90-day or worse rating mths_since_last_record,The number of months since the last public record. mths_since_rcnt_il,Months since most recent installment accounts opened mths_since_recent_bc,Months since most recent bankcard account opened. mths_since_recent_bc_dlq,Months since most recent bankcard delinquency mths_since_recent_inq,Months since most recent inquiry. mths_since_recent_revol_delinq,Months since most recent revolving delinquency. next_pymnt_d,Next scheduled payment date num_accts_ever_120_pd,Number of accounts ever 120 or more days past due num_actv_bc_tl,Number of currently active bankcard accounts num_actv_rev_tl,Number of currently active revolving trades num_bc_sats,Number of satisfactory bankcard accounts num_bc_tl,Number of bankcard accounts num_il_tl,Number of installment accounts num_op_rev_tl,Number of open revolving accounts num_rev_accts,Number of revolving accounts num_rev_tl_bal_gt_0,Number of revolving trades with balance >0 num_sats,Number of satisfactory accounts num_tl_120dpd_2m,Number of accounts currently 120 days past due (updated in past 2 months) num_tl_30dpd,Number of accounts currently 30 days past due (updated in past 2 months) num_tl_90g_dpd_24m,Number of accounts 90 or more days past due in last 24 months num_tl_op_past_12m,Number of accounts opened in past 12 months open_acc,The number of open credit lines in the borrowers credit file.
47	open_acc_6m	Number of open trades in last 6 months
48	open_il_12m	Number of installment accounts opened in past 12 months
49	open_il_24m	Number of installment accounts opened in past 24 months
50	open_il_6m	Number of currently active installment trades
51	open_rv_12m	Number of revolving trades opened in past 12 months
52	open_rv_24m	Number of revolving trades opened in past 24 months
53	out_prncp	Remaining outstanding principal for total amount funded
54	out_prncp_inv	Remaining outstanding principal for portion of total amount funded by investors
55	pct_tl_nvr_dlq	Percent of trades never delinquent

	Feature	Description
56	percent_bc_gt_75	Percentage of all bankcard accounts > 75% of limit.
57	policy_code	publicly available policy_code=1 new products not publicly available policy_code=2
58	pub_rec	Number of derogatory public records
59	pub_rec_bankruptcies	Number of public record bankruptcies
60	purpose	A category provided by the borrower for the loan request.
61	pymnt_plan	Indicates if a payment plan has been put in place for the loan
62	recoveries	post charge off gross recovery
63	revol_bal	Total credit revolving balance
64	revol_util	Revolving line utilization rate or the amount of credit the borrower is using relative to all available revolving credit.
65	sub_grade	LC assigned loan subgrade
66	tax_liens	Number of tax liens
67	term	The number of payments on the loan. Values are in months and can be either 36 or 60.
68	title	The loan title provided by the borrower
69	tot_coll_amt	Total collection amounts ever owed
70	tot_cur_bal	Total current balance of all accounts
71	tot_hi_cred_lim	Total high credit/credit limit
72	total_acc	The total number of credit lines currently in the borrowers credit file total_bal_ex_mort,Total credit balance excluding mortgage total_bal_il,Total current balance of all installment accounts total_bc_limit,Total bankcard high credit/credit limit total_cu_tl,Number of finance trades total_il_high_credit_limit,Total installment high credit/credit limit total_pymnt,Payments received to date for total amount funded total_pymnt_inv,Payments received to date for portion of total amount funded by investors total_rec_int,Interest received to date total_rec_late_fee,Late fees received to date total_rec_prncp,Principal received to date total_rev_hi_lim ÿ,Total revolving high credit/credit limit url,URL for the LC page with listing data. verification_status,Indicates if income was verified by LC not verified or if the income source was verified verified_status_joint,Indicates if the co-borrowers joint income was verified by LC not verified or if the income source was verified

	Feature	Description
73	zip_code	The first 3 numbers of the zip code provided by the borrower in the loan application.

References

1. Baesens, B., Van Gestel, T., Viaene, S., Stepanova, M., Suykens, J., & Vanthienen, J. (2003). Benchmarking state-of-the-art classification algorithms for credit scoring. Journal of the Operational Research Society, 54(6), 627-635.
2. Lessmann, S., Seow, H., Baesens, B. and Thomas, L. (2013). Benchmarking state-of-the-art classification algorithms for credit scoring: A ten-year update. 1st ed. [ebook] pp.1-32. Available at: http://www.business-school.ed.ac.uk/waf/crc_archive/2013/42.pdf (http://www.business-school.ed.ac.uk/waf/crc_archive/2013/42.pdf) [Accessed 15 Sep. 2016].
3. Caruana, R. and Niculescu-Mizil, A. (2006). An Empirical Comparison of Supervised Learning Algorithms. 1st ed. [ebook] Pittsburgh, Pennsylvania, USA: Cornell University, pp.1-7. Available at: https://www.cs.cornell.edu/~caruana/ctp/ct.papers/caruana.icml06.pdf (https://www.cs.cornell.edu/~caruana/ctp/ct.papers/caruana.icml06.pdf) [Accessed 16 Sep. 2016].
4. En.wikipedia.org. (2016). Cohen's kappa. [online] Available at: https://en.wikipedia.org/wiki/Cohen's_kappa (https://en.wikipedia.org/wiki/Cohen's_kappa) [Accessed 24 Nov. 2016].

In [ ]: