Craig Calder March 2016

FOUNDATIONS OF DATA SCIENCE

Capstone Project – Modelling good loan candidates

SUMMARY

The objective of this project was to predict whether a person is a good or bad candidate for a medium term (36 or 60 month) loan based on available attributes. An example client would be a bank or credit union needing a model that would provide an agent the ability to provide an evaluation a potential customer's likelihood of being a good candidate for a loan based on a small number of variables.

Based on the modelling, the following two models performed well and were good predictors of a good versus bad loan candidate with the available attributes. As expected, LoanModel00 model that included 14 variables performed well, but notably, model LoanModel03b with just two variables: annual income and term performed acceptably.

Model	**AUC**
LoanModel00 = glm (bad_loan ~ loan_amnt + annual_inc + dti + delinq_2yrs + revol_util + total_acc + longest_credit_length + bankrpc_state_low + bankrpc_state_high + homeown_mort + homeown_rent + term + purpose + vstatus_verified, data=loan_train, family="binomial")	66.186%
LoanModel03b = glm (bad_loan ~ annual_inc + term, data=loan_train, family="binomial")	63.492%

AUC = Area Under Curve. See Section 6.

SECTION 1: DATA

The data source is available at: https://www.lendingclub.com/info/download-data.action

Loan.csv contains 163,987 observations with 15 variables. For the main dependent variables of interest, bad_loans, this data is unevenly split between good and bad loans: Good Loans (0): 133,971 observations (82%), Bad Loans (1): 30,016 observations (18%)

#	Variable Name	Description
1	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. Loan amount ranges from 500 to 35,000.
2	term	The number of months to pay the loan. Values are either 36 or 60 months (character type)
3	int_rate	Ranging from 5.42 the best candidate (e.g. lowest risk) to 26.06 for high risk customers. (dependent variable)
4	emp_length	Employment length in years.

5	home_ownership	The home ownership status provided by the borrower during registration. Our values are: RENT, OWN, MORTGAGE, OTHER (character type).
6	annual_inc	The self-reported annual income provided by the borrower during registration.
7	purpose	A category provided by the borrower for the loan request.
8	addr_state	The state provided by the borrower in the loan application
9	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.
10	delinq_2yrs	The number of 30+ days past-due incidences of delinquency in the borrower's credit file for the past 2 years
11	revol_util	Revolving line utilization rate, or the amount of credit the borrower is using relative to all available revolving credit.
12	total_acc	Total number of accounts
13	bad_loan	Numeric: 1 = Bad Loan, 0 = Good loan. Captures if the consumer was either a good or bad loan.
14	longest_credit_length	In months.
15	verification_status	Based on the state of the loan application

SECTION 2: IMPUTE MISSING DATA

We are able to ascertain that some observations are missing values. Specifically, emp_length (580 or 3.5%), annual_inc (4 or 0.002%), and delinq_2yrs (29), total_acc (2), revol_util (193), longest_credit_length (29 or 0.017%)

A "safe" maximum threshold of missing values is 5% of the total for large datasets, so using the “mice” function, we will impute the missing values rather than exclude those observations from the analysis:

```
mice(loan,m=1,maxit=25,meth='pmm',seed=500)
```

SECTION 3: WRANGLING

Factor values should be used for categorical data, so there are several variables that needed to be converted to factors and in some cases further broken down into dummy variables. Specifically, the following variables were "wrangled": bad_loan, term, add_state, purpose, verified, home_ownership

- bad_loan: converted to a factor (e.g. a categorical variable that can only take on two values, 1 or 0)
- add_state was initially organized into five regions of the country (e.g. West, South, Midwest, Northeast, East), however, it was later determined that four groups based on the prevalence of bankruptcies in the state produced more meaningful models (source http://www.valuepenguin.com/bankruptcy-filings-state).
- purpose, verified, and home_ownership were factored into a small number of factors and then simplified into dummy variables. https://en.wikipedia.org/wiki/Dummy_variable_(statistics). A dummy variable is one that takes the value 0 or 1 to indicate the absence or presence of some categorical effect that may be expected to shift the outcome.

Lastly, outliers were identified and removed. In particular, some of the observations had very high annual incomes. Therefore, the 27 observations exceeded $1,000,000 per year were excluded from the model. This seems reasonable since these high income customers would likely receive special treatment and may not reflect the model well.

SECTION 4: BALANCING AND SPLITTING

Instead of a training set reflect the 82 / 18% split of good to "bad" loan observations, the training set was first balanced so that 75% of bad_loans = 1 observations (22,512) were matched up with 22,512 "good" loan observations (i.e. bad_loans = 0) observations.

Therefore, the training set is balanced equally between good and bad loans:

loan_train$bad_loan		loan_test$bad_loan	
0	1	0	1
22,512	22,512	111,459	7,504

Note: Additional analysis in Section 6 uses this balanced training data set to evaluate thresholds where the test data is balanced between "good" and "bad" loans.

loan_train$bad_loan		loan_test$bad_loan	
0	1	0	1
22,512	22,512	7,504	7,504

SECTION 5: MODEL 1 – LOGISTIC REGRESSION

Build Bivariate models to identify which of our variables are useful in predicting a particular outcome, and then check for multivariance.

STEP 1: CREATING BIVARIATE MODELS

Build Bivariate models to identify which of our variables are useful in predicting a particular outcome (e.g. models that predict the outcome using a single independent variable.) An independent variable shall be considered significant if there is at least one star at the end of the coefficients row for that variable (e.g. the probability column having a value smaller than 0.05)

Variable	Coefficient	P-Value	Significance
Loan_amount	1.84E-05	1.35E-55	***
emp_length	-0.00182616	0.485009021	
annual_inc	-5.39E-06	5.49E-104	***
dti	0.035582836	1.32E-177	***
delinq_2yrs	0.056976745	3.57E-05	***
revol_util	0.011514081	2.61E-195	***
total_acc	-0.00902606	4.00E-28	***
longest_credit_length	-0.00746845	3.89E-08	***
bankrpc_state_low (dummy)	-0.09342886	4.57E-05	***
bankrpc_state_med (dummy)	-0.04470727	0.032735737	*
bankrpc_state_medhigh (dummy)	0.059659539	0.003381705	**
bankrpc_state_high (dummy)	0.096607377	0.000106993	***
homeown_other (dummy)	0.060802301	0.072699274	.
homeown_mort (dummy)	-0.26221461	1.30E-43	***
homeown_rent (dummy)	0.244811429	5.09E-38	***
term (factor)		1.64E-297	***
purpose (factor)		3.41E-16	***
vstatus_verified (factor)		1.54E-67	***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Conclusion: from the above, the following variables are significant to predicting (P > +/- 0.8) bad_loan: loan_amnt, annual_inc, dti, delinq_2yrs, revol_util, longest_credit_length, total_acc, longest_credit_length, bankrpc_state_low, bankrpc_state_high, purchase_hardgood, purchase_financing, vstatus_verified, vstatus_notverified,

The following variables will not be used:
emp_length, bankrpc_state_med, bankrpc_state_medhigh, homeown_other

STEP 2: CHECK FOR MULTIVARIANCE

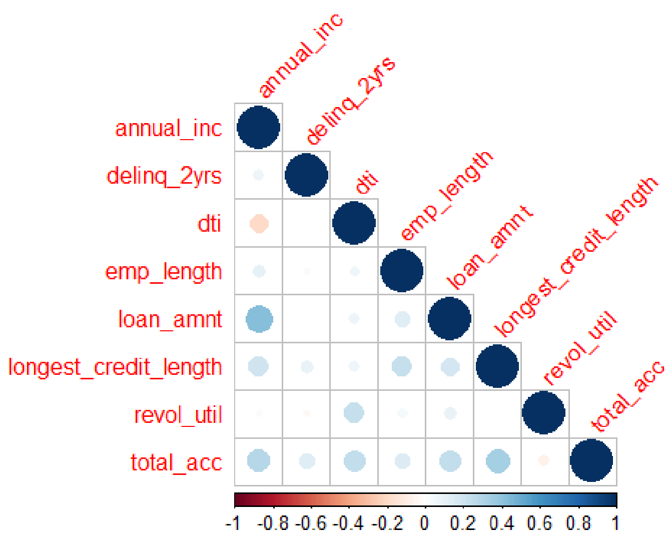
Often, variables that were significant in bivariate models are no longer significant in multivariate analysis due to correlation between the variables. Evaluating the variable pairs to determine if any have a high degree of correlation (a correlation greater than 0.8 or less than -0.8):

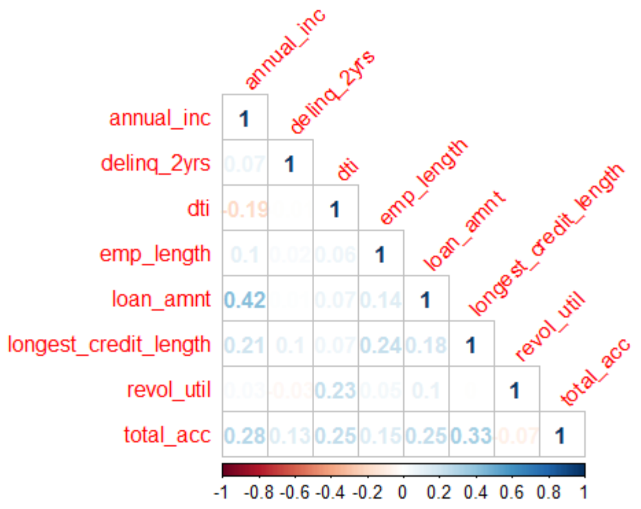
When correlation is close to 1: This means that there is a strong relationship between your two variables. This means that changes in one variable are strongly correlated with changes in the second variable. When correlation is very close to 1 we can conclude that there is a strong relationship between these two variables.

When correlation is close to 0: This means that there is a weak relationship between your two variables. This means that changes in one variable are not correlated with changes in the second variable. If correlation were 0.01, we could conclude that our variables were not strongly correlated.

When positive (+). This means that as one variable increases in value, the second variable also increase in value. Similarly, as one variable decreases in value, the second variable also decreases in value. This is called a positive correlation.

When negative (-). This means that as one variable increases in value, the second variable decreases in value. This is called a negative correlation.





Correlations displayed graphically:

Positive correlations are displayed in blue and negative correlations in red color.

Color intensity and the size of the circle are proportional to the correlation coefficients.

Conclusion: None of the numeric variables are excessively correlated and thus require additional management prior to building a candidate model:

```
LoanModeloo = glm (bad_loan ~ loan_amnt + annual_inc + dti + delinq_2yrs + revol_util + total_acc +
longest_credit_length + bankrpc_state_low +  bankrpc_state_high  + homeown_mort +
homeown_rent + term + purpose + vstatus_verified, data=loan_train, family="binomial")
```

SECTION 6: LOGISTIC REGRESSION MODEL VALIDATION

BINARY CLASSIFICATION CONFUSION MATRIX

Shows us the number of correct and incorrect predictions made by the classification model compared to the actual outcomes (e.g. test dataset)

- TP = True Positive: predicted positive, test positive
- FP = False Positive: predicted positive, test negative
- TN = True Negative: predicted negative, test positive
- FN = False Negative: predicted negative, test negative

Now we choose the threshold for the best trade off. Our choices are to be concerned with a high specificity (e.g. low false positive rate), pick the threshold that maximizes the true positive rate and minimizes the false positive rate (the number of times we predict bad loans, but they're actually good candidates). If you are more concerned with having a high sensitivity (e.g. high true positive rate) that minimizes false positive rate but has a very high true positive rate (the number of times we predict good candidates and they're actually good loan candidates.)

With a bank deciding on loan candidates, choosing someone who is a bad loan candidate can be compensated for by charging a higher interest rate, so we will try to minimize the number of times we turn away a good customer by aiming for a high sensitivity.

Threshold	TP	FP	TN	FN	Sensitivity	Specificity
t > .70	106,393	6,505	998	4,995	0.9552	0.1330
t > .65	101,629	5,867	1,636	9,759	0.9124	0.2180
t > .60	94,395	5,036	2,467	16,993	0.8474	0.3288
t > .55	83,695	4,056	3,447	27,693	0.7514	0.4594
t > .50	69,625	2,919	4,584	41,763	0.6251	0.6110
t > .45	53,333	1,911	5,592	58,055	0.4788	0.7453

Looking at these high level results, a cutoff of t = 0.55 is better choice, as it's balancing TP and TN where TP is the proportion of customers rightly identified as "bad" (75%) and TN is the proportion of customers rightly identified as "good" (46%).

SKEWED TEST DATA SET INVESTIGATION

One potential problem with the remaining test dataset is that it is now heavily skewed in favor of bad_loans = 0 (i.e. "good" loans). This supplementary analysis explores the impact of the skewed nature of the data set affects the threshold determination.

table (loan_test$bad_loan)

0 (good)	1(bad)
111,459	7,504

After constructing a balanced test dataset that balances the number of good loans with 7504 bad loans:

table (loan_test$bad_loan)

0 (good)	1(bad)
7,504	7,504

Threshold Value	TP	FP	TN	FN	Sensitivity	Specificity
t > .70	7,169	6,505	998	334	0.9555	0.133
t > .65	6,848	5,867	1,636	655	0.9127	0.218
t > .60	6,336	5,036	2,467	1,167	0.8445	0.3288
t > .55	5,588	4,056	3,447	1,915	0.7448	0.4594
t > .50	4,629	2,919	4,584	2,874	0.617	0.611
t > .45	3,573	1,911	5,592	3,930	0.4762	0.7453

Although there is no change to the false positives (FP), true negatives (TN), which makes sense since the number of bad loan cases remained at 7,504 and those are either going to be correctly identified (TN) or incorrectly identified as positives (FP).

However, the proportion of FP and TN relative to the TP and FN increases dramatically because the dataset contains so many fewer "good" loan observations (7,504 vs. 111,459) that can now be categorized as either true positives or false negatives.

Given the above, it seems our model works best in a universe where the good loan candidates largely outweigh the number of bad candidates in similar proportions to the data set used to construct the model (e.g. Good Loans (0): 82%, Bad Loans (1): 18%). Otherwise one would need to reduce the threshold value to closer to t > 0.5 so as to better balance the true positives and true negatives with the false positives and false negatives.

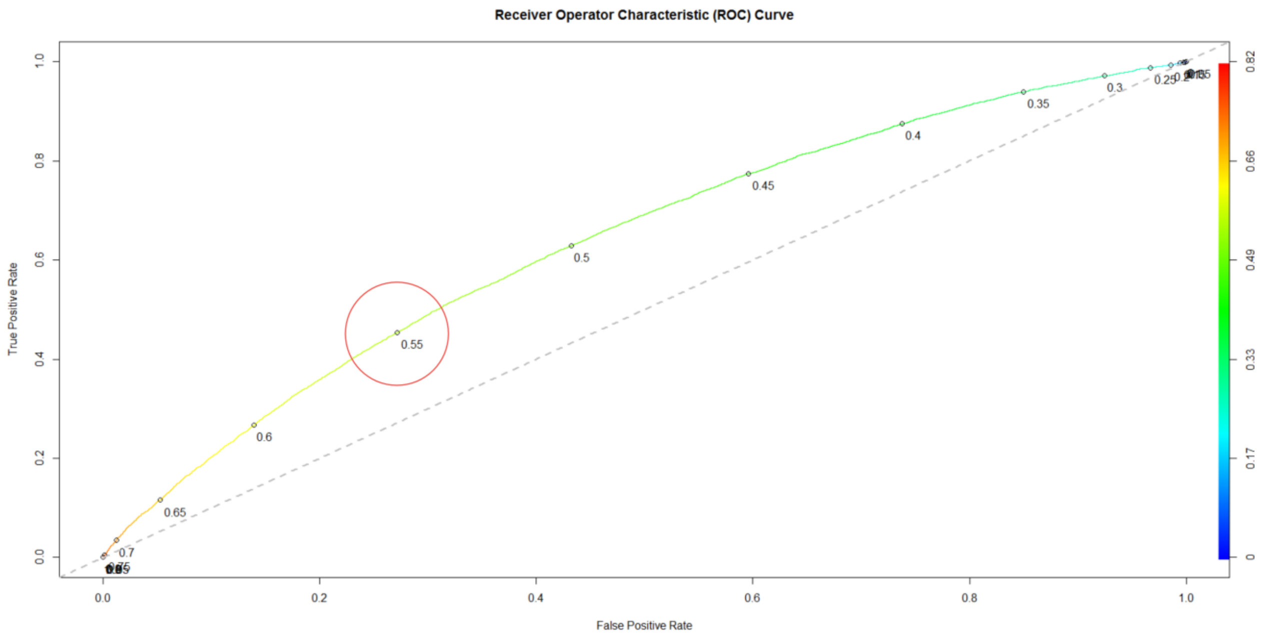
RECEIVER OPERATOR CHARACTERISTIC (ROC) CURVE

Note: Understanding ROC curves: https://www.youtube.com/watch?v=5kdGhDqdwPk

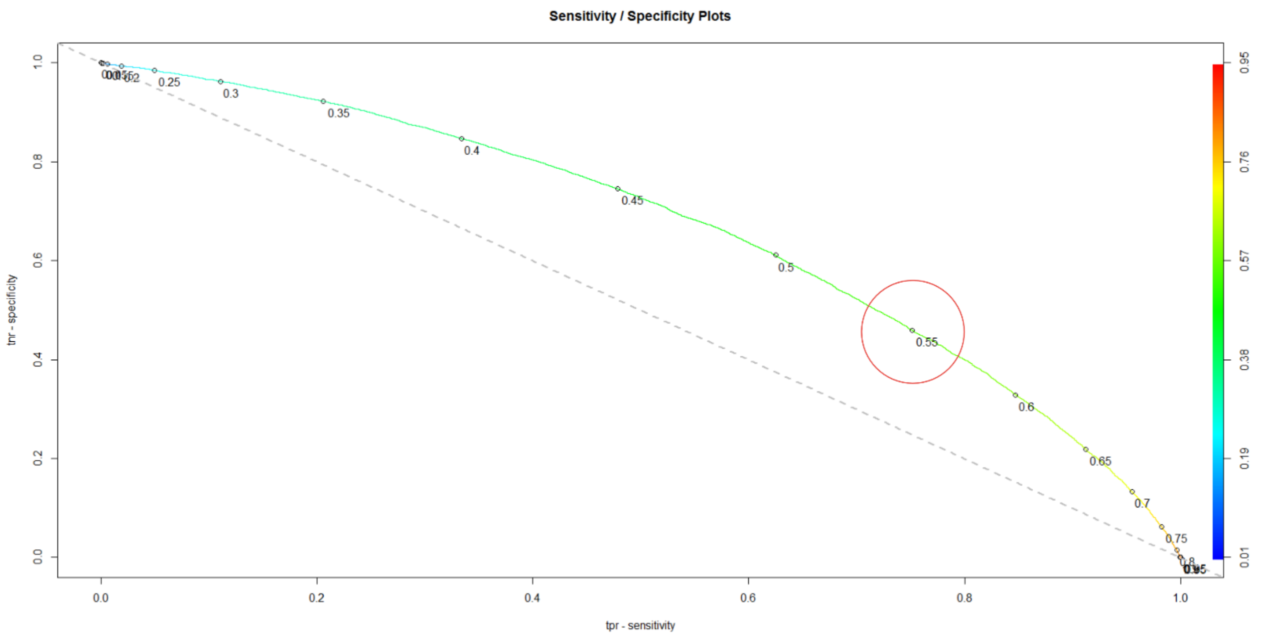
Using our new model, we calculate the performance of our model compared to the test data.

```
predictTest = predict(LoanModel00, type="response", newdata=loan_test)
ROCRpred = prediction(predictTest, loan_test$bad_loan)
ROCRperf = performance(ROCRpred, measure="tpr",x.measure="fpr")
SSperf =  performance(ROCRpred, measure="sens", x.measure="spec")
```

plot (ROCRperf, colorize = TRUE,print.cutoffs.at=seq(0,1,0.05), text.adj=c(-0.2,1.7), main="Receiver Operator Characteristic (ROC) Curve", xlab="True Positive Rate", ylab="False Positive Rate")
abline(a=0,b=1,lwd=2,lty=2,col="gray")



plot (SSperf, colorize = TRUE,print.cutoffs.at=seq(0,1,0.05), text.adj=c(-0.2,1.7), main="Sensitivity / Specificity Plots", xlab="tpr - sensitivity", ylab="tnr - specificity")
abline(a=1,b=-1,lwd=2,lty=2,col="gray")



AREA UNDER THE CURVE (AUC)

Compute the test set Area Under the Curve (AUC). The area under the ROC curve is often used as a measure of quality of the classification model - random model has an area under the curve equal to .5, a perfect classifier is equal to 1. Above .7 is highly desirable.

```
aucROCR = as.numeric(performance(ROCRpred, "auc")@y.values) = 0.6618582
```

From which we conclude that our first logistic regression model, LoanModel00, is acceptable, albeit somewhat short of our target .70.

SIMPLIFICATION

The questions that will now be explored are:

a) Can the model be simplified without significantly reducing its usefulness?
b) Can a different model do a better job (e.g. be more predictive)?

If we remove now less significant variables identified from summary of LoanModel01, namely remove the following variables: longest_credit_length, bankrpc_state_high, homeown_rent, purpose and create a new model:

LoanModel02 = glm (bad_loan ~ loan_amnt + annual_inc + dti + delinq_2yrs + revol_util + total_acc + bankrpc_state_low + homeown_mort + term + vstatus_verified, data=loan_train, family="binomial")

We obtain a test area under curve (AUC) = 66.1469% (a negligible decrease from 66.1858%) an encouraging sign that some simplification is possible.

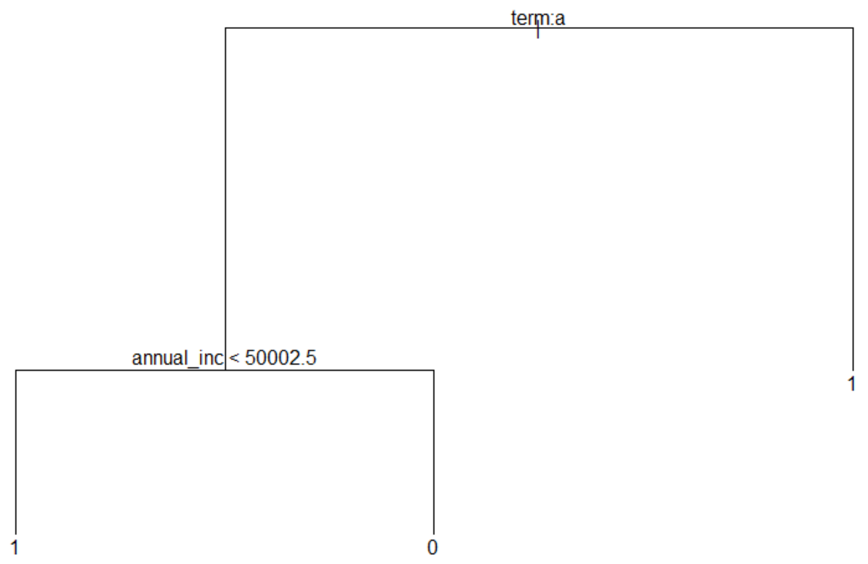
SECTION 7: CLASSIFICATION MODELS

One problem with the above model is that logistic regression models are not easily interpretable. The model coefficients in logistic regression indicate the importance and relative effect of variables, but do not give a simple explanation of how a decision is made.

Classification methods build what is called a tree by splitting on the values of the independent variables. To predict the outcome for a new observation or case, you can follow the splits in the tree and at the end, you predict the most frequent outcome in the training set that followed the same path. Some advantages of CART are that it does not assume a linear model, like logistic regression or linear regression, and it's a very interpretable model.

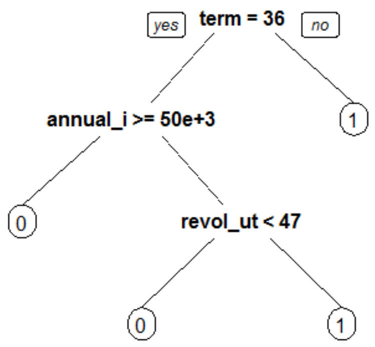
TREE PACKAGE

```
library(tree)

tr <- tree(bad_loan ~ ., data=loan_train)

plot(tr); text(tr)
```



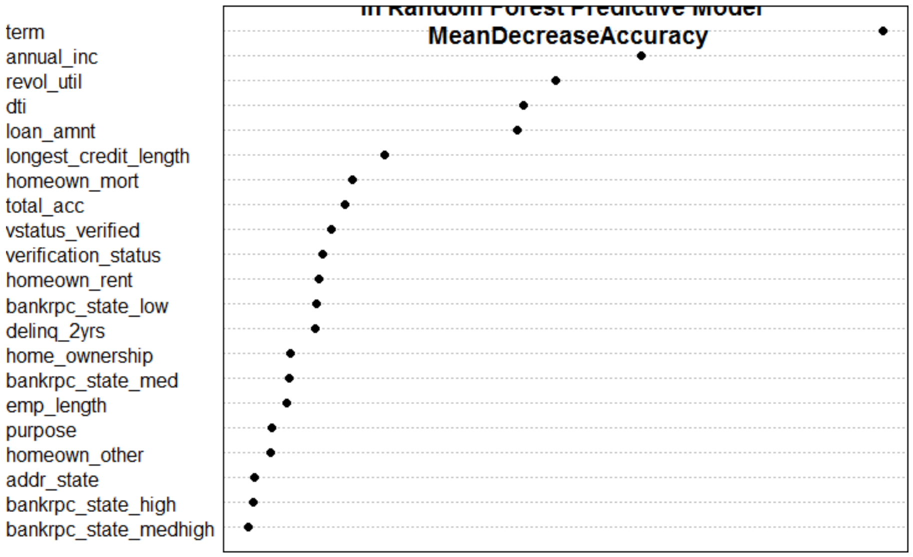
RPART PACKAGE

```
library (rpart.plot)
library(rpart)
LoanTree = rpart (bad_loan~., control=rpart.control(minsplit=20),data=loan_train)
prp(LoanTree)
```



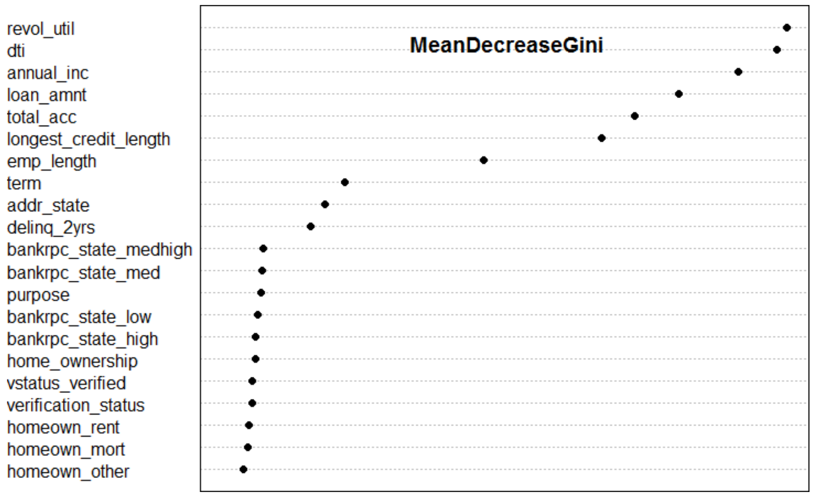
RANDOM FOREST PACKAGE

```
library(randomForest)
set.seed(415)
fit <- randomForest(bad_loan ~ ., data=loan_train, importance=TRUE, ntree=100, do.trace=TRUE)
plot(fit);  print(fit)
varImpPlot(fit, sort = TRUE, type = 1, pch = 19, col = 1, cex = 1, main = "Relative Importance of Loan Variables \nin Random Forest Predictive Model \n MeanDecreaseAccuracy")
```



MeanDecreaseGini is a measure of variable importance based on the Gini impurity index used for the calculation of splits during training. A low Gini (i.e. higher decrease in Gini) means that a particular predictor variable plays a greater role in partitioning the data into the defined classes.

```
varImpPlot(fit,sort = TRUE, type = 2, pch = 19, col = 1, cex = 1, main = "Relative Importance of Loan Variables \nin Random Forest Predictive Model \n MeanDecreaseGini")
```



SECTION 8: AREA UNDER CURVE (AUC) FOR MODEL VARIATIONS

Bringing all of the insights back from the classification models, let's try to re-run the glm models but with just the variables identified by the classification models. Consolidating all of the outputs into a single table, we see the following:

- Model00: Original
- Model 02: remove variables (e.g. simplify the model)
- Model 03a: Based on RPART: term, annual_inc, revol_util
- Model 03b: Based on Tree: term, annual_inc
- Model 04a, 04b: Based on randomForest: revol_util, dti, annual_inc, loan_amnt, total_acc, longest_credit_length, emp_length

Model # and Type	AUC	# of variables	Description of variables…
Model 01 : Log Reg	66.186%	14	loan_amnt + annual_inc + dti + delinq_2yrs …..
Model 02 : Log Reg	66.147%	11	loan_amnt + annual_inc + dti + delinq_2yrs …..
Model 03a: rpart	64.859%	3	annual_inc + revol_util + term
Model 03b: tree	63.492%	2	annual_inc + term
Model 04a: Rand Forest	63.387%	4	revol_util + dti + annual_inc + loan_amnt
Model 04b: Rand Forest	63.575%	8	revol_util + dti + annual_inc + loan_amnt ….