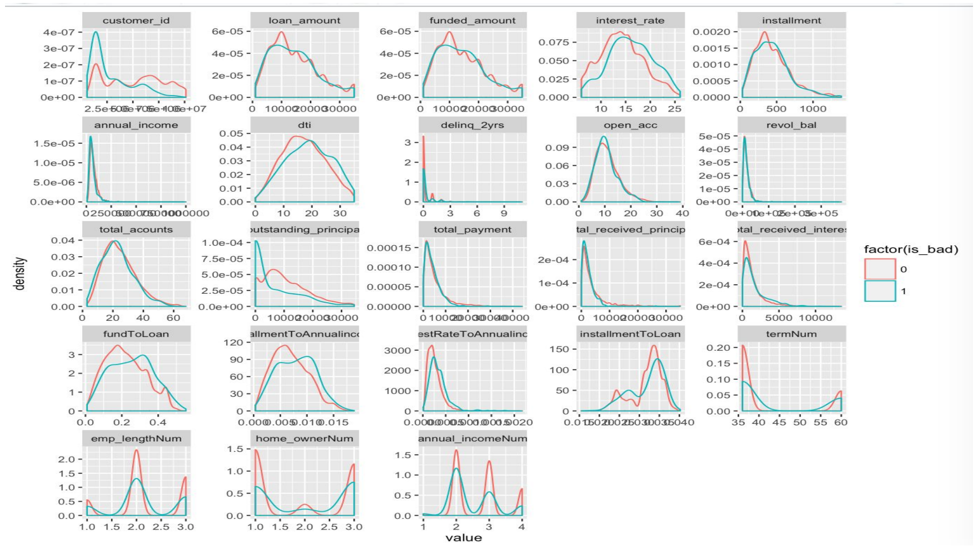
Methodology

1) Downloaded data from Redshift as a .csv file and read into R, storing into a dataframe.

2) Pre-process the data

- **Binary Classification of good and bad loans:** Create new category for dataframe, called is_bad, which acts like an indicator variable with 1 for bad loans and 0 for good loans. Statuses with "Late (16-30 days)", "Late (31-120 days)", "Default", "Charged Off" were classified as bad loans, statuses with “” were removed, and the rest were classified as good loans. I included “Charged Off” in bad loans as I did not want to work a ratio of good loans to bad loans of 9339 : 185. Putting “Charged Off” under bad loans improved the ratio to 9121 : 403, and it seemed okay because the definition of “Charged Off” was similar to “default”.
- **Removal of Nas:** Remove fields which have mostly NA values (> 70%), as those cannot be used as features -- these are “months since last deliq, X and X.1”
- **Numeric Typing:** Converted some factors into numerics, for example, interest percentages which are more suitable as a continuous distribution.
- **Modification of Factor Levels:** Simplified some factors, for example, grouped 13 employment length levels into 3 types (“transient” comprising <= 1 year emp_length, “stable” comprising 10+ year emp_length, and “middle” comprising (2 - 10 years). Also modified loan term (short and long term), home ownership(Mortagage, Own, Rent), annual income(< 20000, < 60000, < 100000, > 100000), delinq_2yrs (0, 1, 2, >2). Allows more meaningful compares.
- **Created New Ratios:** Based on intuition, created ratios that seemed meaningful, such as funded_amt : annual_inc (similar to dti), installment : annual_inc (might indicate how able to pay), interest : annual_inc, installment : loan.

3) Conduct Feature Selection - Here I plotted continuous distributions of numeric variables (I converted some factors to numerics) for both bad and good loans. I then picked out variables that seemed to have differences in the bad and good populations to be my features. I have attached the graphs.



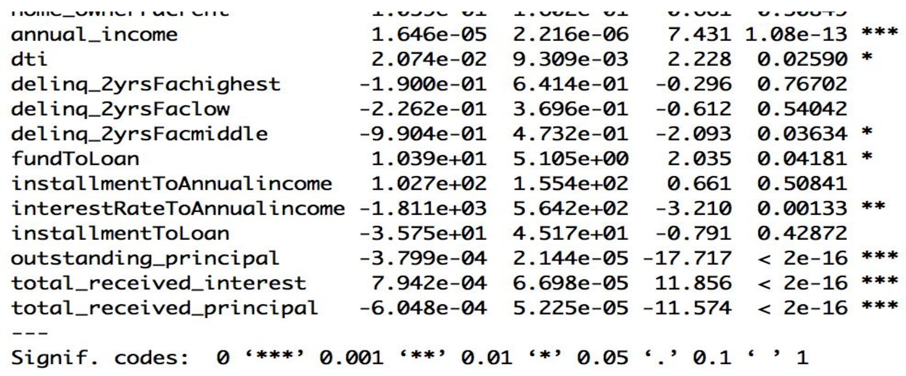
I used all of the above variables as my features, except for customer_id, revol_bal, total_payment, total_accounts, open_acc) , as these were insignificant (based on the plot above). I was also thinking of whether to remove total_received_principal, total_received_interest, outstanding_principal as these are values which we cannot have before a loan is even given out (for example outstanding principal is a value we would have only over the course of the loan repayment period). However, my results were better with them inside (more later).

4) Creation of training and test datasets: Out of the 9524 filtered records left, I split 25% into test data, 75% into training data.

Creation of Models

First I used a generalized linear model, I then used a decision tree with weights, and then extended that to a random forest. For each of these models, I evaluated their predictive power using ROC, AUC and KS. Why so? Firstly, the Receiver Operating Characteristic (ROC) is a plot of the true positive rate against the false positive rate, which works well for a model with an independent binary variable. I score the test data set by plotting the ROC curve with the test sample using the “prediction” function, so as to get all the Y-hats, then I plot a performance function based on these scores together with a 45’ line. An ideal predictive result would be to correctly classify bad and good loans to get a triangle with vertices at (0,0), (0,1), (1,1), so the more the area under the curve, better the model. AUC is derived from ROC. Secondly, the Kolmogorov-Smirnov statistic calculates the maximum difference between the accumulated true positive rate and false positive rate, so the higher and closer to 1 it is, the better.

A) Generalized Linear Model using glm: I used the glm in R to plot the factor “is_bad” (the dependent variable) against the features I had earlier selected. Since is_bad is an indicator variable, I used assumed a binomial distribution as my “family” parameter, and since I wanted a linear regression, I chose to use the “logit” link to plot the log of odds ratio. Also used na.action to omit na records.



```
annual_income                 1.646e-05  2.216e-06   7.431 1.08e-13 ***
dti                           2.074e-02  9.309e-03   2.228  0.02590 *
delinq_2yrsFachighest        -1.900e-01  6.414e-01  -0.296  0.76702
delinq_2yrsFaclow            -2.262e-01  3.696e-01  -0.612  0.54042
delinq_2yrsFacmiddle         -9.904e-01  4.732e-01  -2.093  0.03634 *
fundToLoan                    1.039e+01  5.105e+00   2.035  0.04181 *
installmentToAnnualincome     1.027e+02  1.554e+02   0.661  0.50841
interestRateToAnnualincome   -1.811e+03  5.642e+02  -3.210  0.00133 **
installmentToLoan            -3.575e+01  4.517e+01  -0.791  0.42872
outstanding_principal        -3.799e-04  2.144e-05 -17.717  < 2e-16 ***
total_received_interest       7.942e-04  6.698e-05  11.856  < 2e-16 ***
total_received_principal     -6.048e-04  5.225e-05 -11.574  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Results:

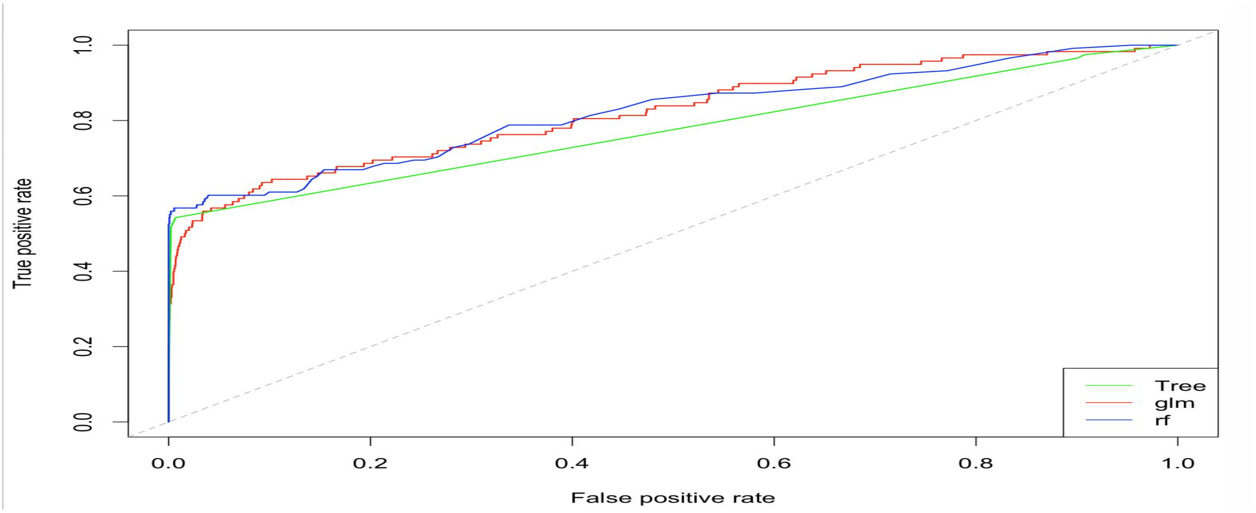
1) annual_income, outstanding_principal, total_received_interest, total_received_principal, dti, delinq_2yrs, fundToLoan, interestRateToAnnualincome are all significant at at least 5% level of significance, which validated my decision to modify / create them and then include them as features.

B) Weighted Tree Using rpart: Another alternative to linear regressions is to use partitioning (classification). Using a top-down approach, the algorithm chooses the variable that best splits the data into homogeneous subsets with the same equivalence classes, and then recursively continues until the subset at a node converges to the same value for the target variable. As the tree prediction gives a matrix for both y=0 and y=1, when plotting the ROC curve, we take the second column vector instead of the entire matrix. Furthermore, I used a prior of (0.5 : 0.5) to weight the tree as I realized that the number of bad loans to good loans was around 400 : 9000, which meant the decision tree would try to maximize the number of good loans identified correctly instead of bad loans, which makes the model accurate but not as useful for predicting bad loans.

C) Random Forest using randomForest: A generalization of random decision trees is the random forest algorithm, which constructs a lot of decision trees during training time such that the tendency of singular decision trees to overfit to the training set is corrected. I chose the classification instead of regression mode for this because I wanted an optimization of model B’s classification to compare with glm in A.

<u>**Evaluations:**</u>

Model	**AUC**	**KS Statistic**
GLM	0.798	0.497
Decision Tree	0.76	0.480
Random Forest	0.778	0.480



<u>**Conclusion:**</u> I think all three algorithms I used turned out to have very similar effects, which was surprising as I thought for a classification problem, a random forest would have the best predictive ability. What might have helped was identifying good features before plotting these models, resulting in a tight fit for the regression model. Furthermore, it may be that my choosing of which variables to use as factors and how to create factor levels were not ideal. With more time, I would vary my feature selection more creatively to build better regression and classification models, but as of now, to predict whether a loan is good or bad, I would just use call the predict function using the glm model titled “contModelBigger” in my rscript.