IDS 572 Assignment 2
Data Mining for Business

Viharika Bharti -655974244 <vbhart2@uic.edu>
Aditya Gajula – 660252623 <agajul2@uic.edu>
Chaitra Srirama – 674121942 <csrira2@uic.edu>

Models for investment in LendingClub loans

Question 1

(a1) Develop gradient boosted models to predict loan_status. Experiment with different parameter values and identify which gives 'best' performance. How do you determine 'best' performance?

We try to find the best GBM model for predicting Loan Status with the help of Parameter tuning.
The Parameter tuning is performed using the following grid search values:

```
treeDepth= (3,5)
minNodeSize= (10,30)
bagFraction= (0.5,0.8, 1)
shrinkage= (0.001,0.01,0.1)
n.trees= 500
cv.folds=5
```

Note the oversampled training data has been used for the parameter tuning. For each combination, we pick the best iteration to get the fitted scores with a probability of 0.5. Using this we get the Misclassification Error Rates and Accuracy of the models.

The following were Accuracy for each model developed from each of the 36 combination parameters developed from the Grid:

On Training Data:

```
> Accuracy_Trn
 [1] 0.58 0.58 0.58 0.58 0.58 0.58 0.58 0.58 0.58 0.58 0.58 0.58 0.60 0.60
0.60 0.60
0.60 0.60 0.60
[20] 0.60 0.59 0.60 0.59 0.60 0.64 0.66 0.64 0.66 0.64 0.66 0.64 0.66 0.64
0.66 0.64 0.66
```

On Test Data

```
> Accuracy_Tst
 [1] 0.88 0.88 0.88 0.88 0.88 0.88 0.88 0.88 0.88 0.88 0.88 0.88 0.87 0.87
0.87 0.87
 0.87 0.87 0.87
[20] 0.87 0.88 0.87 0.88 0.87 0.86 0.86 0.86 0.86 0.86 0.86 0.86 0.86 0.86
0.86 0.87 0.86
```

From the above results we find the accuracy of the models with different parameters are close to each oth er, without any high variation.

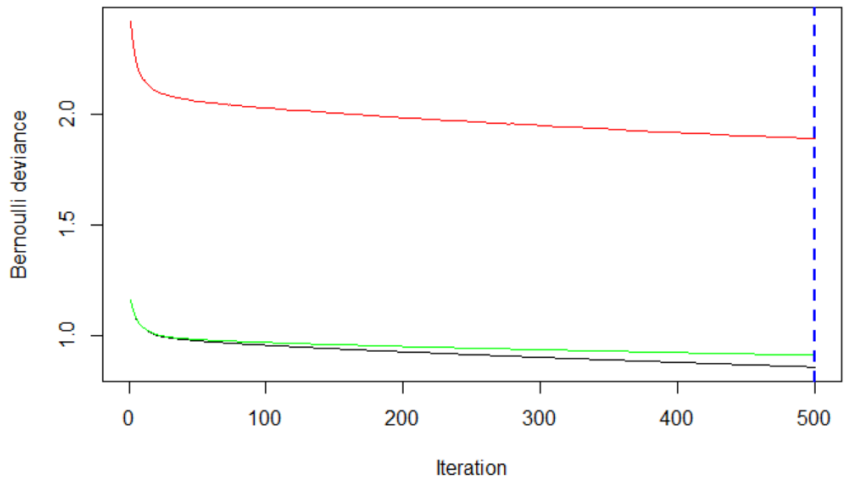
The highlighted Training and test Accuracy from the 32nd combination have been chosen to provide a goo d model and has the following parameters:

```
treeDepth= 5
minNodeSize= 30
bagFraction= 0.8
shrinkage= 0.1
n.trees= 500
cv.folds=5
```

We re-run the GBM model on oversampled training data with the above parameters to check other performance parameters as follows:

```
gbm_model<-gbm(loan_status~.,data=subset(os_lcdfTrn, select=-c(annRet,
actualTerm, actualReturn)), distribution = 'bernoulli', n.trees= 500,
n.minobsinnode=30, train.fraction= 0.7, bag.fraction=0.8, shrinkage=0.1,
interaction.depth= 5, cv.folds=5, n.cores=NULL)


A gradient boosted model with bernoulli loss function.
500 iterations were performed.
The best cross-validation iteration was 500.
The best test-set iteration was 500.
There were 58 predictors of which 56 had non-zero influence.
```

Best iteration performed: 500

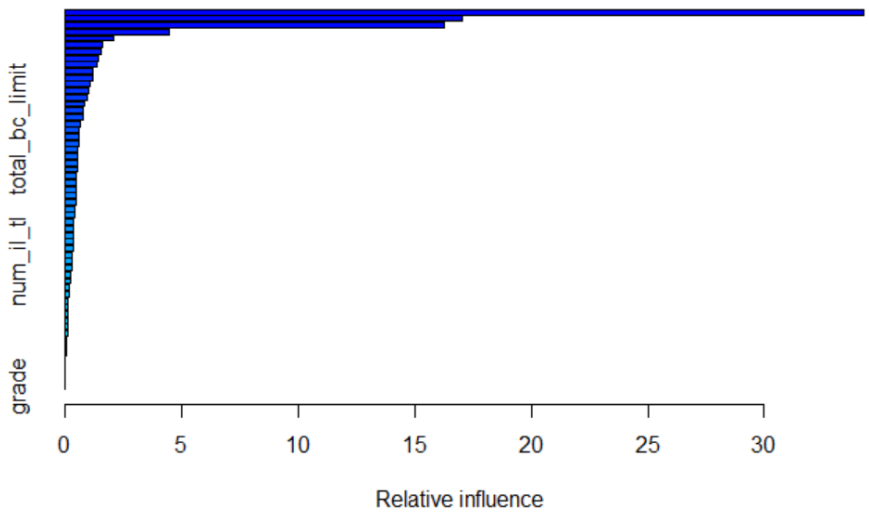


Confusion Matrix on Test Data.

```
          Reference
Prediction     0     1
         0 22954  3036
         1   762  1035

[1] "1 = Fully Paid, 0= Charged off"
```

Metric	Result
Misclassification Error	***0.14***
Accuracy	***0.86***

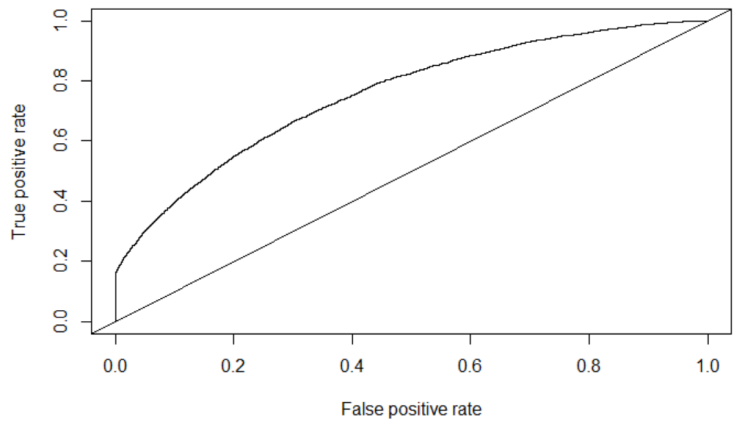
Variable Importance:



The top 15 contributing variables have been shown as follows:

	Variables	Importance
debt_settlement_flag	debt_settlement_flag	34.2740586
int_rate	int_rate	17.0249331
sub_grade	sub_grade	16.2507111
emp_length	emp_length	4.4483727
acc_open_past_24mths	acc_open_past_24mths	2.1001326
avg_cur_bal	avg_cur_bal	1.6154257
installment	installment	1.5540494
tot_hi_cred_lim	tot_hi_cred_lim	1.4163392
dti	dti	1.3533756
purpose	purpose	1.1945013
total_rev_hi_lim	total_rev_hi_lim	1.1705449
mo_sin_old_rev_tl_op	mo_sin_old_rev_tl_op	1.0474412
revol_bal	revol_bal	0.9864653
bc_util	bc_util	0.9606467
bc_open_to_buy	bc_open_to_buy	0.8224114

ROC Curve:



AUC value for the best model: 0.7555422

(a2) For the gbm model, what is the loss function, and gradient in the method you use? (Write the expression for these, and briefly describe).

Loss function is the distribution parameter which we use for the GBM model. We have used "Bernoulli" in our model.

The Bernoulli Loss function is given by:

$$L(y, F) = \ln(1 + \exp(-y\,F))$$

To minimize the loss we take the derivative of the loss function with respect to $F(x_i)$ and the final expression is partial derivative(J)/partial derivative of $F(x_i)$ which which is equal to $F(x_i) - y_i$

The Gradient method used in the model is Negative gradient which is:

$$-g(x_i) = y_i/(1 + \exp(y_i\,F(x_i))$$

The reason for choosing the above loss function is due to the nature of the response variable. The nature of the response variable is categorical i.e. typically takes on binary values $y \in \{0, 1\}$, thus, assuming that it comes from the Bernoulli distribution. In this case, the probability of class-wise response can be estimated by minimizing the negative log-likelihood associated with the class labels, hence this is also known as Logistic Loss.

(b1) Develop linear (glm) models to predict loan_status. Experiment with different parameter values and identify which gives 'best' performance. How do you determine 'best' performance? How do you handle variable selection? Experiment with Ridge and Lasso, and show how you vary these parameters, and what performance is observed.

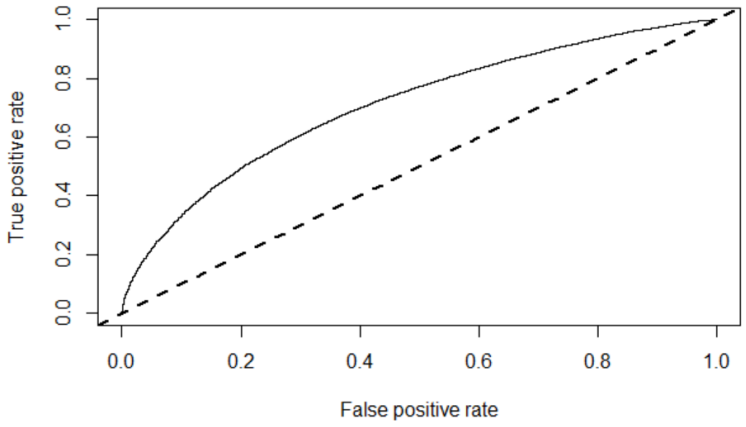
GLM models to predict Loan Status was developed as follows:

1) Vanilla case GLM:

Performance on Oversampled Training Data:

Metric	Accuracy
Accuracy	0.81

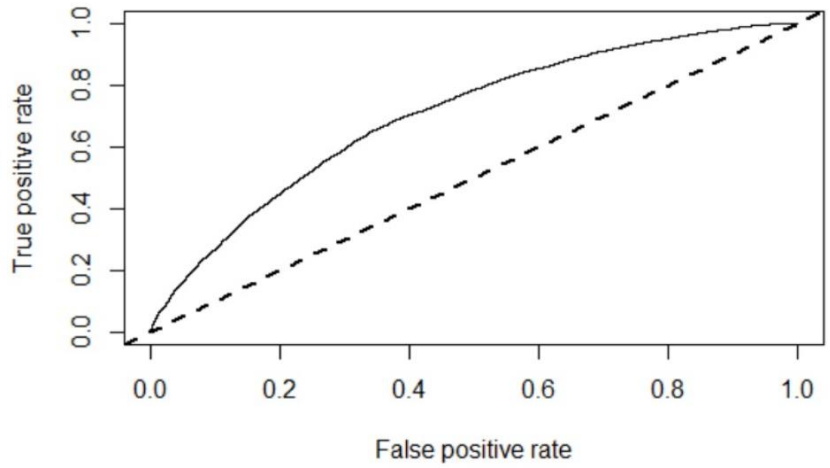
ROC Curve:



Performance on Test Data:

Metric	Accuracy
Accuracy	0.85

ROC Curve:



From the above we developed a logistic regression model to predict loan status from the lending club data. As a vanilla case model, we got an accuracy of 85%, with an Area under the ROC curve to be 0.7036909 on Test Data.

Variable selection can be handled by taking the best p-values for the variables. This is one way of selecting the variables according to their p-values (p-values < alpha value) called as the Forward Selection method.

```
Coefficients: (7 not defined because of singularities)
                                Estimate Std. Error z value Pr(>|z|)
(Intercept)                    2.205e+00  2.722e-01   8.103 5.38e-16 ***
loan_amnt                      7.970e-05  1.603e-05   4.971 6.66e-07 ***
funded_amnt                           NA         NA      NA       NA
int_rate                      -5.281e-02  4.406e-02  -1.199 0.230691
installment                   -2.801e-03  4.758e-04  -5.887 3.92e-09 ***
gradeB                        -7.637e-01  2.790e-01  -2.737 0.006198 **
gradeC                        -1.070e+00  4.147e-01  -2.580 0.009875 **
gradeD                        -9.871e-01  5.543e-01  -1.781 0.074932 .
gradeE                        -7.555e-01  6.991e-01  -1.081 0.279822
gradeF                        -1.350e+00  9.458e-01  -1.427 0.153549
gradeG                        -8.173e-01  1.364e+00  -0.599 0.548971
sub_gradeA2                   -1.234e-01  7.838e-02  -1.575 0.115326
sub_gradeA3                   -2.585e-01  9.499e-02  -2.721 0.006506 **
sub_gradeA4                   -2.727e-01  1.055e-01  -2.585 0.009737 **
sub_gradeA5                   -4.775e-01  1.267e-01  -3.770 0.000163 ***
sub_gradeB1                    1.283e-01  1.518e-01   0.845 0.398065
sub_gradeB2                    1.455e-01  1.100e-01   1.322 0.186154
sub_gradeB3                    3.983e-02  7.650e-02   0.521 0.602561
sub_gradeB4                    1.823e-02  4.282e-02   0.426 0.670214
sub_gradeB5                           NA         NA      NA       NA
sub_gradeC1                    1.457e-01  1.099e-01   1.325 0.185105
sub_gradeC2                    1.639e-01  9.393e-02   1.745 0.081008 .
sub_gradeC3                    1.502e-01  6.897e-02   2.178 0.029408 *
sub_gradeC4                    1.057e-01  4.798e-02   2.204 0.027556 *
sub_gradeC5                           NA         NA      NA       NA
sub_gradeD1                    7.178e-03  1.117e-01   0.064 0.948761
sub_gradeD2                   -7.168e-02  8.045e-02  -0.891 0.372964
sub_gradeD3                    1.870e-01  6.963e-02   2.685 0.007249 **
sub_gradeD4                   -6.761e-02  6.297e-02  -1.074 0.282936
sub_gradeD5                           NA         NA      NA       NA
sub_gradeE1                   -3.512e-01  1.657e-01  -2.119 0.034113 *
sub_gradeE2                   -3.468e-01  1.589e-01  -2.183 0.029036 *
sub_gradeE3                   -3.118e-01  1.455e-01  -2.143 0.032139 *
sub_gradeE4                   -2.387e-01  1.378e-01  -1.732 0.083306 .
sub_gradeE5                           NA         NA      NA       NA
sub_gradeF1                    2.803e-01  3.542e-01   0.791 0.428777
sub_gradeF2                    4.867e-02  3.555e-01   0.137 0.891089
sub_gradeF3                   -4.201e-01  3.737e-01  -1.124 0.261001
sub_gradeF4                    2.104e-01  3.785e-01   0.556 0.578297
sub_gradeF5                           NA         NA      NA       NA
sub_gradeG1                   -7.603e-02  9.461e-01  -0.080 0.935951
sub_gradeG2                   -8.556e-01  9.999e-01  -0.856 0.392142
sub_gradeG3                   -5.309e-01  1.168e+00  -0.454 0.649561
sub_gradeG4                    1.339e+00  1.043e+00   1.283 0.199342
sub_gradeG5                           NA         NA      NA       NA
emp_length1 year               3.888e-02  3.279e-02   1.186 0.235639
emp_length10+ years            2.235e-02  2.594e-02   0.862 0.388799
emp_length2 years              6.036e-02  3.074e-02   1.963 0.049615 *
emp_length3 years              1.040e-01  3.203e-02   3.248 0.001163 **
emp_length4 years             -2.219e-02  3.457e-02  -0.642 0.520979
emp_length5 years              4.900e-02  3.461e-02   1.416 0.156814
emp_length6 years              8.750e-02  4.007e-02   2.183 0.029013 *
```

```
emp_length7 years                    4.634e-03  3.855e-02   0.120 0.904334
emp_length8 years                   -6.945e-02  3.623e-02  -1.917 0.055217 .
emp_length9 years                    9.946e-02  4.061e-02   2.449 0.014326 *
emp_lengthn/a                       -4.819e-01  3.433e-02 -14.035  < 2e-16 ***
home_ownershipOWN                   -7.839e-02  2.296e-02  -3.414 0.000641 ***
home_ownershipRENT                  -1.962e-01  1.762e-02 -11.134  < 2e-16 ***
annual_inc                          -4.490e-07  1.323e-07  -3.395 0.000687 ***
verification_statusSource Verified  -5.674e-02  1.659e-02  -3.419 0.000628 ***
verification_statusVerified         -2.987e-02  1.875e-02  -1.593 0.111108
purposecredit_card                   5.992e-02  6.788e-02   0.883 0.377354
purposedebt_consolidation            7.051e-02  6.689e-02   1.054 0.291837
purposehome_improvement              1.019e-01  7.171e-02   1.421 0.155375
purposehouse                         3.142e-01  1.225e-01   2.565 0.010319 *
purposemajor_purchase               -6.241e-02  8.055e-02  -0.775 0.438407
purposemedical                       1.774e-02  8.918e-02   0.199 0.842329
purposemoving                       -5.175e-02  9.383e-02  -0.552 0.581246
purposeother                         9.445e-02  7.160e-02   1.319 0.187126
purposerenewable_energy             -1.399e-02  2.485e-01  -0.056 0.955109
purposesmall_business               -3.564e-02  9.457e-02  -0.377 0.706295
purposevacation                      2.137e-01  9.493e-02   2.251 0.024366 *
purposewedding                       7.979e+00  4.395e+01   0.182 0.855946
dti                                 -1.929e-02  9.562e-04 -20.170  < 2e-16 ***
delinq_2yrs                         -8.504e-02  1.042e-02  -8.163 3.27e-16 ***
inq_last_6mths                      -6.080e-02  8.138e-03  -7.472 7.90e-14 ***
mths_since_last_delinq               2.292e-04  3.686e-05   6.219 5.00e-10 ***
open_acc                             4.187e-02  2.208e-02   1.896 0.057998 .
pub_rec                              6.213e-02  2.648e-02   2.346 0.018964 *
revol_bal                            2.333e-06  8.948e-07   2.607 0.009121 **
total_acc                           -2.625e-02  9.930e-03  -2.643 0.008213 **
initial_list_statusw                -3.549e-02  1.347e-02  -2.634 0.008428 **
collections_12_mths_ex_med          -1.138e-01  3.760e-02  -3.027 0.002469 **
tot_coll_amt                        -4.658e-06  3.608e-06  -1.291 0.196681
tot_cur_bal                         -6.856e-07  4.324e-07  -1.586 0.112799
total_rev_hi_lim                    -2.868e-07  6.463e-07  -0.444 0.657157
acc_open_past_24mths                -5.019e-02  3.507e-03 -14.312  < 2e-16 ***
avg_cur_bal                          2.412e-07  1.076e-06   0.224 0.822641
bc_open_to_buy                      -7.330e-06  1.495e-06  -4.903 9.44e-07 ***
bc_util                              3.081e-04  4.896e-04   0.629 0.529208
chargeoff_within_12_mths            -1.440e-01  5.393e-02  -2.670 0.007582 **
delinq_amnt                         -2.481e-06  7.381e-06  -0.336 0.736778
mo_sin_old_il_acct                   5.785e-04  1.384e-04   4.181 2.90e-05 ***
mo_sin_old_rev_tl_op                 7.079e-04  8.331e-05   8.498  < 2e-16 ***
mo_sin_rcnt_rev_tl_op                3.243e-04  5.713e-04   0.568 0.570226
mo_sin_rcnt_tl                       5.156e-03  1.072e-03   4.809 1.52e-06 ***
mort_acc                             5.579e-02  1.089e-02   5.125 2.98e-07 ***
mths_since_recent_bc                 2.261e-04  6.871e-05   3.291 0.000998 ***
mths_since_recent_inq                1.792e-03  2.632e-04   6.807 9.95e-12 ***
num_accts_ever_120_pd               -1.002e-02  6.172e-03  -1.624 0.104432
num_actv_bc_tl                      -5.452e-03  7.984e-03  -0.683 0.494677
num_actv_rev_tl                     -3.319e-03  1.103e-02  -0.301 0.763446
num_bc_sats                         -9.628e-03  5.752e-03  -1.674 0.094130 .
num_bc_tl                           -2.540e-02  3.949e-03  -6.431 1.26e-10 ***
num_il_tl                            2.519e-02  9.941e-03   2.534 0.011279 *
num_op_rev_tl                       -5.139e-03  5.192e-03  -0.990 0.322249
num_rev_accts                        4.045e-02  1.015e-02   3.983 6.79e-05 ***
num_rev_tl_bal_gt_0                 -1.441e-02  1.151e-02  -1.252 0.210727
num_sats                            -3.363e-02  2.218e-02  -1.516 0.129544
num_tl_90g_dpd_24m                   6.438e-02  1.659e-02   3.881 0.000104 ***
num_tl_op_past_12m                   3.805e-03  5.639e-03   0.675 0.499821
pct_tl_nvr_dlq                      -2.771e-03  1.065e-03  -2.601 0.009297 **
percent_bc_gt_75                    -1.818e-03  3.305e-04  -5.499 3.83e-08 ***
pub_rec_bankruptcies                -1.393e-01  3.073e-02  -4.531 5.86e-06 ***
tax_liens                           -8.192e-02  3.104e-02  -2.639 0.008317 **
tot_hi_cred_lim                      1.079e-06  3.909e-07   2.761 0.005764 **
```

```
total_bal_ex_mort                -2.527e-06  6.260e-07  -4.036 5.43e-05 ***
total_bc_limit                    9.807e-06  1.040e-06   9.433  < 2e-16 ***
total_il_high_credit_limit        4.338e-06  6.319e-07   6.866 6.62e-12 ***
```

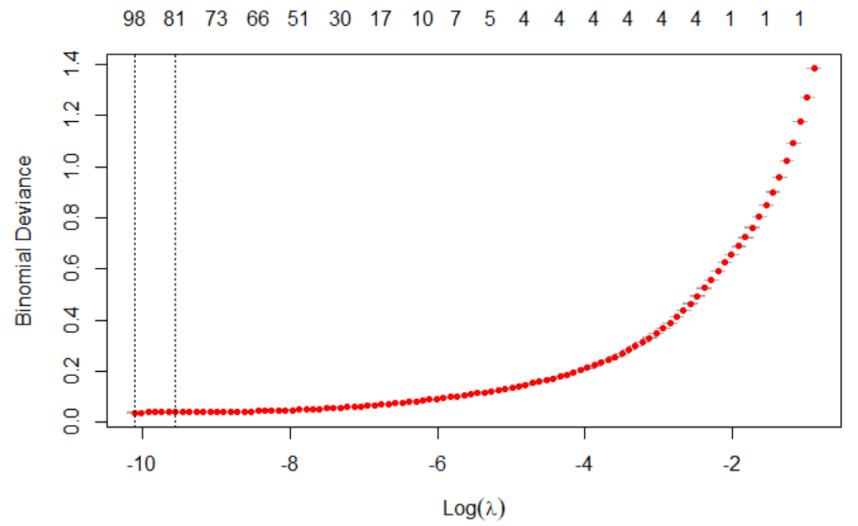
Experimenting with different parameter values along with lasso and ridge regression.

A) For lasso, alpha=1

Obtaining the best Lambda using cross validation

```
       Lambda Measure       SE Nonzero
min 4.072e-05 0.04033 0.001218      92
1se 7.809e-05 0.04138 0.001181      85
```

Plot of cross validation results:



A1) Fitting model with cross validation with Lamda.min

Model Accuracy on Training Data: 0.997995

Confusion matrix Training Data

```
          Reference
Prediction     0     1
         0 55228   102
         1    28  9479

Sensitivity: 0.997 for Fully Paid
```

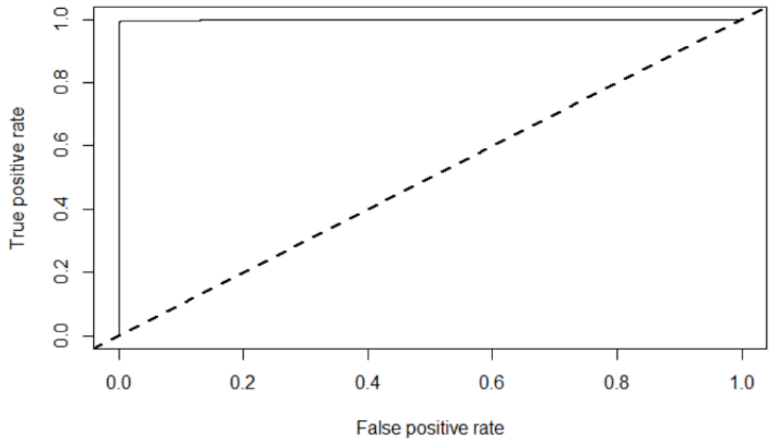
Model Accuracy on Test Data: 0.9975888

Confusion Matrix on Test Data

```
          Reference
Prediction     0     1
         0 23699    50
         1    17  4021
```

Sensitivity: 0.995 for Fully Paid

ROC curve:



AUC Value: 0.9989345

A2) To fit a model to using cv_lasso$lambda.1se i.e λ which gives the most regularized model (Lasso)

Model Accuracy on Training Data: 0.9978562

Confusion Matrix Training Data

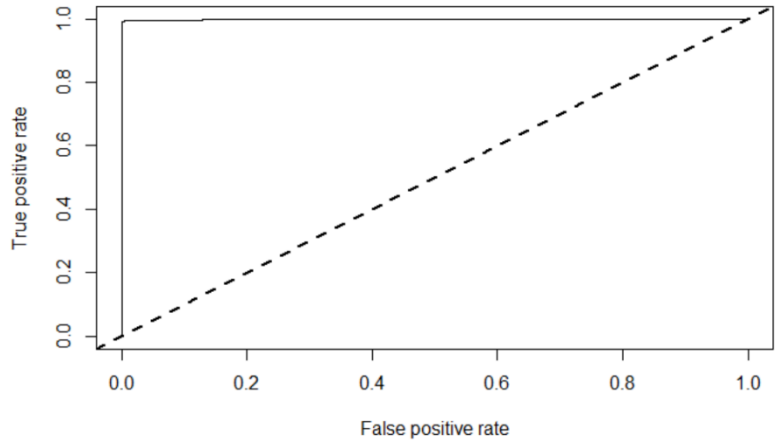
```
          Reference
Prediction     0     1
         0 55228   111
         1    28  9470
```

Model Accuracy on Test Data: 0.9974808

Confusion Matrix Test Data

```
          Reference
Prediction     0     1
         0 23700    54
         1    16  4017
```

ROC Curve:



AUC Value: 0.9989467

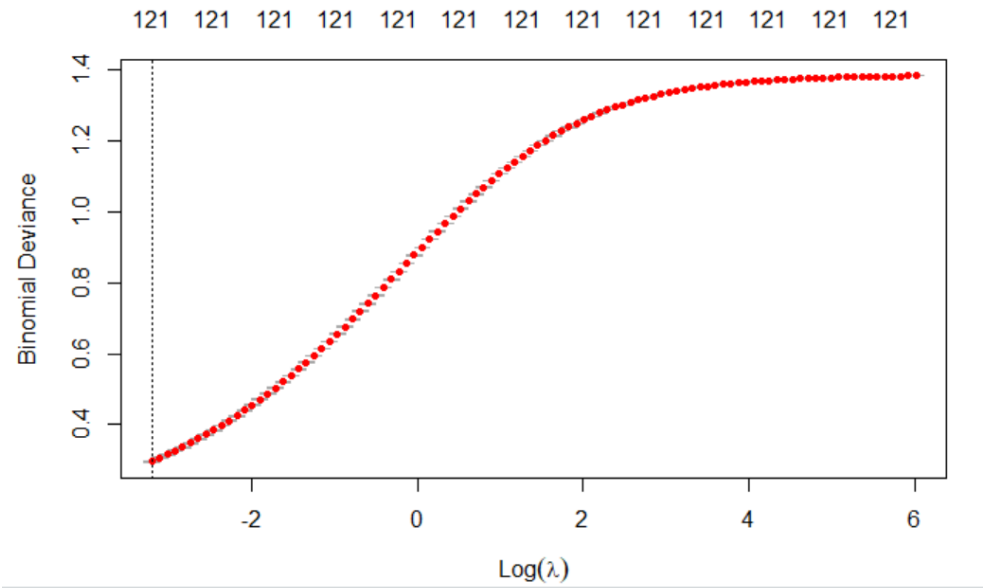
B) For ridge, alpha=0:

Obtaining the best Lambda using cross validation

```
Measure: Binomial Deviance

     Lambda Measure       SE Nonzero
min 0.04078   0.296 0.001772     121
1se 0.04078   0.296 0.001772     121
```

As we can observe the Lambda.min and Lamda.1se is the same i.e with a value of 0.04078

Plot of cross validation results:



Model Accuracy on Training Data: 0.9852245

Confusion Matrix on train data

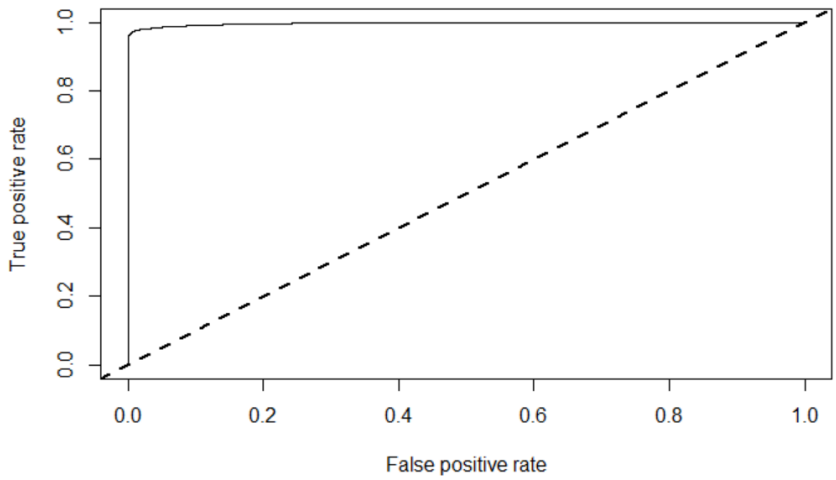
```
Prediction     0     1
         0 55256   958
         1     0  8623
```

Model Accuracy on Test Data: 0.9851009

Confusion Matrix on test data

```
          Reference
Prediction     0     1
         0 23715   413
         1     1  3658
```

ROC Curve:



AUC Value: 0.9974162

(b2) For the linear model, what is the loss function, and link function you use? (Write the expression for these, and briefly describe).

1)Loss function:

In the GLM models for loan status, the response variable is categorical and binomial in nature that is Fully Paid (1) and Charged Off (0). Hence the loss function would be that of a logistic loss function given as follows:

Logistic loss: L (y, F) = $\ln(1 + \exp(-y\,F))$

and in general, if we consider the linear function as y= w0 =w1x1 + w2x2…. Then the expression for regularized error is:

E(w,b) = 1 $/n$ ∑in = 1 $J(w)$ + α $R(w)$
Where J is the loss (cost) function, R is the regularization term R: penalizes for model complexity,
α is regularization (hyper) parameter.

Logistic Regression seeks f(x) to maximize likelihood of data which is equivalent to minimizing the logistic loss.

2) Link function:

The link function for logistic regression is given as follows:

$$\ln(\frac{p}{1-p}) = \beta_0 + \beta_1 x$$

$$\ln(odds) = \beta_0 + \beta_1 x$$

The above expression is also called the logit(p), which is the log of odds.
βo and β1 are the coefficients of the Logistic Regression.

(c) Compare performance of models with that of random forests (which you did in your last assignment).

In random forests, our best model was with number of trees = 200 and after comparing the performance measures of GBM, GLM and Random Forests, GBM and random Forests is our best model as there accuracy is close to each other on test Data, whereas GLM is leading to overfit on both Training and Test Datasets

Performance Metrics of the three models:

Model	Accuracy	AUC Values
Random Forest	0.85	0.69
GBM	0.86	0.75
GLM	0.99	0.99

(d) Examine which variables are found to be important by the best models from the different methods, and comment on similarities, difference. What do you conclude?

The below-mentioned variables are found to be important in predicting the output after comparing the mini-gini decrease (Random forests) and predictive performance (GBM and GLM) for the variables.
Sub_grade, dti, tot_hi_cred_limit are the most important variables observed in all the models (common to all the models). After observing the variable importance from various models, we concluded that variables which have very less importance can be ignored.

```
Sub_grade
Emp_length
Dti
Grade
Mo_sin_old_rev_tl_op
```

```
Bc_open_to_buy
Tot_hi_cred_limit
Int_rate
Acc_open_past_24mths
```

(e) In developing models above, do you find larger training samples to give better models? Do you find balancing the training data examples across classes to give better models?

Proportion of charged off and fully paid samples in dataset:

loan_status <fctr>	n <int>
Charged Off	9520
Fully Paid	55317

We need to resample data, because proportion is not balanced for charged off and fully paid. We can under sample, over sample or do both.

Undersampling Results:

loan_status <fctr>	n <int>
Fully Paid	9438
Charged Off	9520

Oversampling Results

loan_status <fctr>	n <int>
Fully Paid	55317
Charged Off	55220

Combination of over sampling and over sampling data:

loan_status <fctr>	n <int>
Fully Paid	32487
Charged Off	32350

We have considered oversampling on data set, due to the simple fact that under-sampling data results in loss of data. Data lost is not used in training the model hence opportunity to build good models is lost and also the test accuracy for the over-sampled data set is higher than the normal data set, which is supposed to be like that.

Test accuracy (over-sampled data) = 0.86

Test accuracy (normal data set) = 0.65

For all the models built in Question 1, we have utilized oversampled Training data.

Question 2:

Develop models to identify loans which provide the best returns. Explain how you define returns? Does it include Lending Club's service costs?

To identify loans which provide best returns we will develop models on actual returns. Here Actual return is our derived variable, it is calculated as follows:

Annual Return = (Total Payment – Funded Amount)/ Funded Amount

(total_pymnt -funded_amnt) / funded_amnt)

Then we calculate **Actual Term** (actualTerm) from our dataset.

So, actual return can be calculated by taking the ratio between annual return and actual term.

Actual Return = Annual Return/ Actual Term

((total_pymnt -funded_amnt) / funded_amnt) *(1/ actualTerm))

Lending club service costs: In our problem, we are not including the service cost because there is not relevant variable (which is supposed to be (service_fee_charge)) that can give data about the service costs.

But typically, investors pay LendingClub a service fee equal to approximately 1% of the amount of each borrower payment received within 15 days of the payment due date. There are a couple circumstances where the 1% fee would change: If borrowers miss a payment, investors do not pay a service fee.

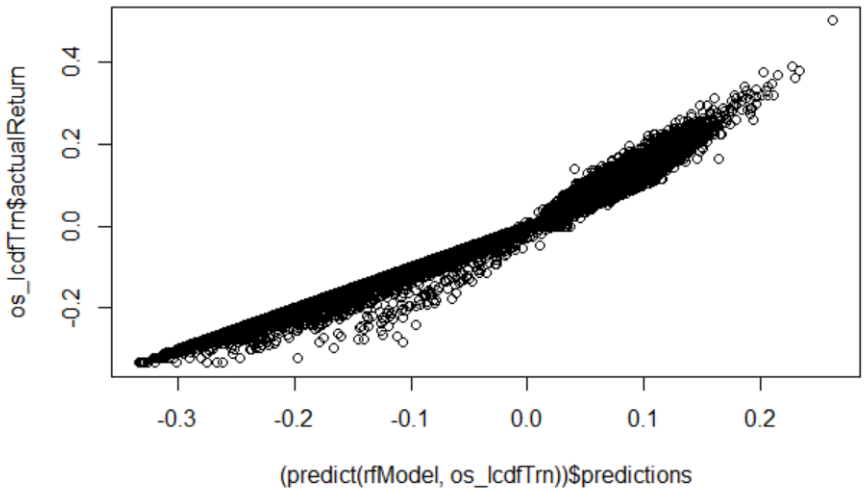
1) **Random Forest to predict Actual Return:**

Experimentation was performed by varying the number of trees as a parameter, in which we observed slight variation on the RMSE for training data, however this did not make a huge difference on Test Dataset.

Num.trees	RMSE for Training Data	RMSE for Test Data
50	0.01794232	0.08730879
100	0.01730409	0.08706549
200	0.1466807	0.08544102
500	0.01699767	0.08683095

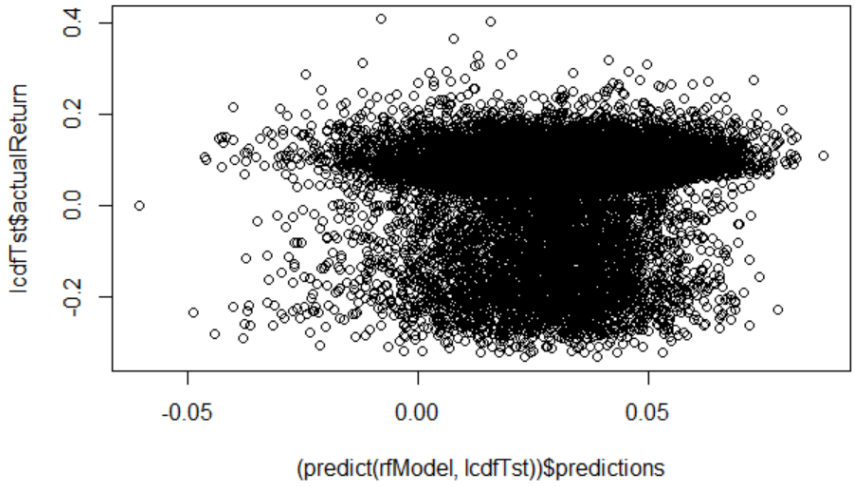
From the above we choose the model with 200 trees as it gives the lowest RMSE for both Training and Test Datasets

Training Data Set



RMSE: 0.01699767

Test Data Set



RMSE: 0.08683095

1) **GBM Models to predict Actual Return:**

We try to find the best GBM model for predicting Actual Returns with the help of Parameter tuning. The Parameter tuning is performed using the following grid search values:

```
TreeDepth= (2,5)
minNodeSize= (10,30)
bagFraction= (0.5,0.8, 1)
shrinkage= (0.001,0.01,0.1)
```

For each combination, we pick the Best Tree and the minimum RMSE to pick the best model.

For all 36 combinations of the parameters the best Tree and the minimum RMSE were as follows:

Best Tree for each iteration:

```
[1] 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000  994  996
987  999 1000
[18]  999 1000 1000 1000 1000 1000 1000  998 1000  995  998  992 1000  999
997 1000 1000
[35] 1000 1000
```

Minimum RMSE for each combination of parameters:

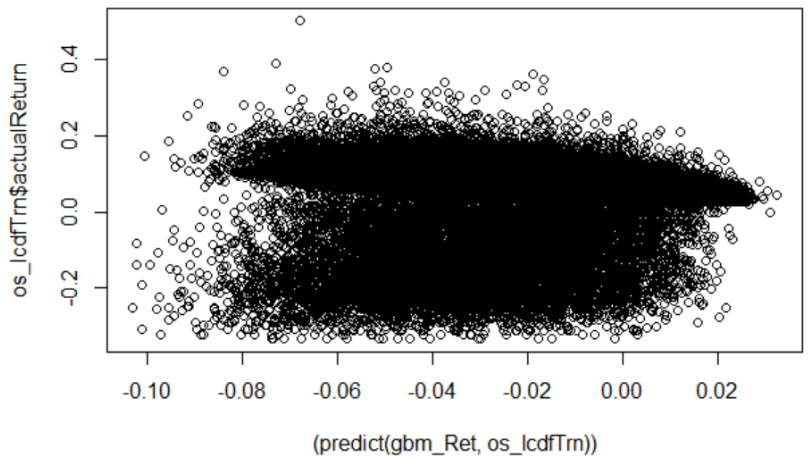
```
[1] 0.1676944 0.1666409 0.1676840 0.1666352 0.1677679 0.1666721 0.1677638
0.1666746
 [9] 0.1678248 0.1667131 0.1678248 0.1667131 0.1641126 0.1620584 0.1640827
0.1620601
[17] 0.1642782 0.1621291 0.1642892 0.1621056 0.1645202 0.1623974 0.1645206
0.1623887
[25] 0.1590383 0.1486320 0.1591534 0.1487008 0.1588247 0.1480171 0.1589644
0.1484259
[33] 0.1592951 0.1490495 0.1592576 0.1487846
```

The highlighted RMSE is the lowest RMSE found among all iterations. The iteration number for this model was 31 with the following parameters:

```
TreeDepth= 2
minNodeSize= 30
bagFraction= 0.8
shrinkage= 0.1
```

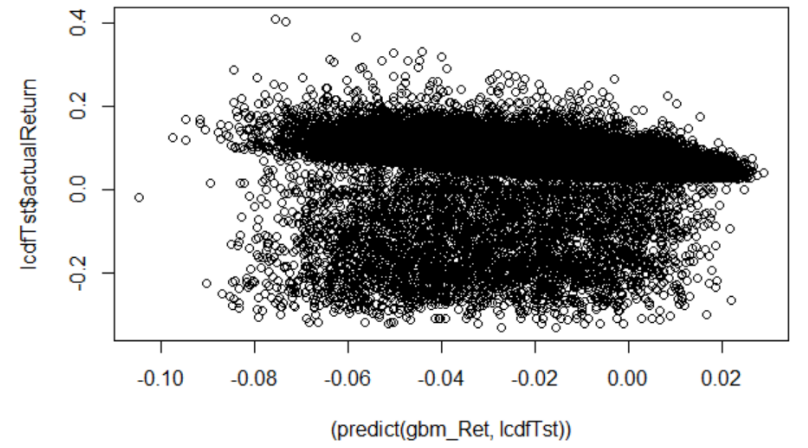
Running a GBM model using the above parameters, performance Metrics are as follows:

Training Data:



RMSE: 0.1101538

Test Data:



RMSE: 0.108638

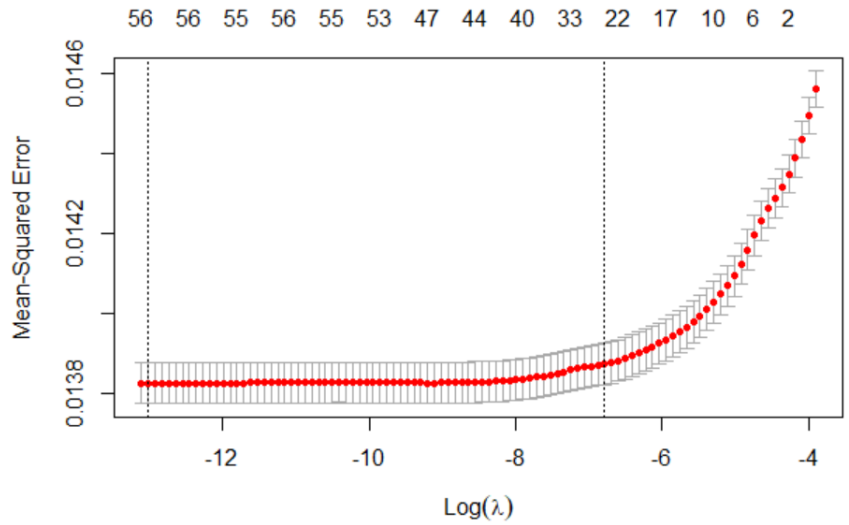
2) **GLM on Actual Return:**

1) Using Lasso:

```
Measure: Mean-Squared Error

       Lambda Measure        SE Nonzero
min 0.0000696 0.01383 6.447e-05      51
1se 0.0014986 0.01389 6.231e-05      22
```

We observe that the mean Squared Error of Lamda.min and Lambda.1se are approximately the same

```
RMSE: 0.1175078
```

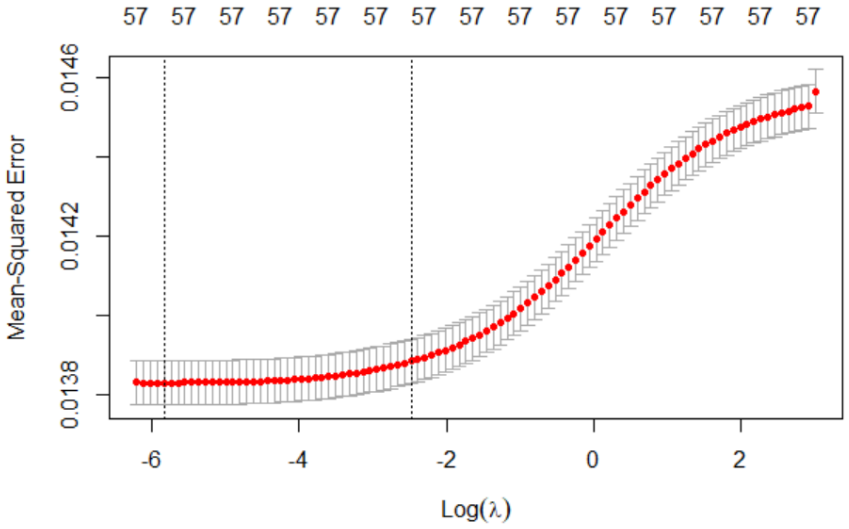


1) Using Ridge:

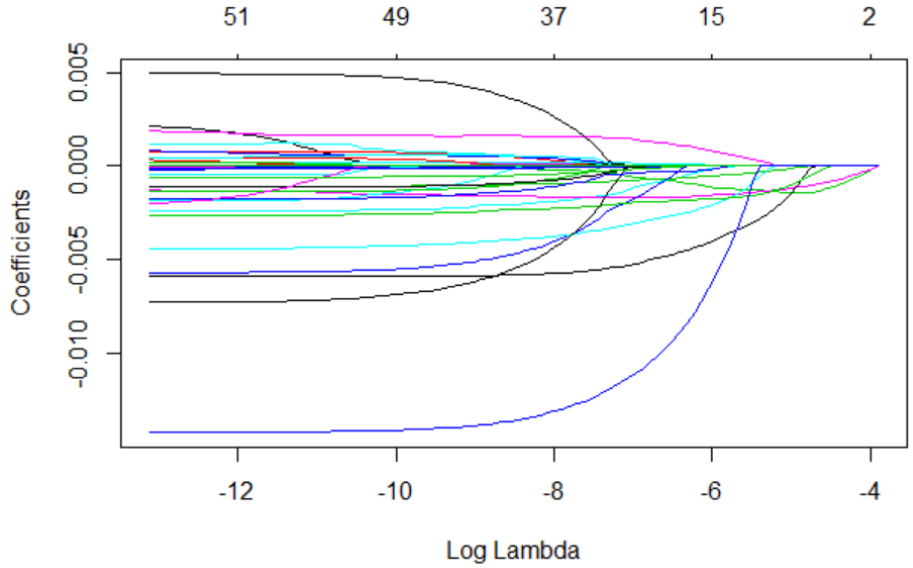
```
Measure: Mean-Squared Error

     Lambda Measure        SE Nonzero
min 0.00268 0.01383 4.528e-05      57
1se 0.06338 0.01387 4.521e-05      57
```

We observe that the mean Squared Error of Lamda.min and Lambda.1se are approximately the same

```
RMSE: 0.1175084
```



Lambda is a numeric value for amount of shrinkage. Below plot shows coefficients for variables as lambda varies:



Question 3: Considering results from Questions 2, how would you select loans for investment? Describe your approach and show performance.

In order to select loans for investment, we check the performance by deciles using each best model developed in Question 2.

1) ***Performance by deciles for Random Forest:***

On Training Data:

Tile	Count	Avg. Predicted Return	No. of Defaults	Avg Actual return	Minimum Return	Maximum Return	Average Term	Total A	Total B	Total C	Total D	Total E	Total F
1	11054	0.09129	432	0.12461	0.06885	0.50199	1.41224	21	1047	4563	3819	1327	218
2	11054	0.06765	442	0.08683	0.04819	0.15926	2.07543	156	3613	5210	1701	346	28
3	11054	0.05621	464	0.0701	0.04128	0.14121	2.28231	1237	5350	3800	563	96	8
4	11053	0.04665	646	0.05575	0.02142	0.12147	2.39418	3687	5701	1412	214	35	4
5	11054	0.03727	877	0.04226	0	0.11335	2.64262	7491	2754	521	201	79	8
6	11054	0.0058	8145	0.00519	-0.0756	0.08587	2.88847	3276	2389	3160	1656	526	42
7	11053	-0.0608	11053	-0.0638	-0.1953	-0.0294	3	1326	3180	3808	1920	697	111
8	11054	-0.1238	11054	-0.1275	-0.2836	-0.094	3	1043	2857	4188	2012	783	171
9	11054	-0.1784	11054	-0.1823	-0.3221	-0.1526	3	852	2766	3960	2411	838	206
10	11053	-0.2432	11053	-0.2466	-0.3333	-0.2054	3	675	2279	4073	2613	1158	198

From above decile table, we can see that the first decile has the maximum average actual return. To know more about people who can be targeted for maximum return within this decile group, we can investigate this decile more based on grades. Below table shows the top 50 results (out of 110,573) from first tile which has the maximum actual return but for investigation.

	Grade	Loan_Status	actualReturn	actualTerm	Int_rate	predRet	tile
1	G	1	0.5019910	0.08219178	26.77	0.2798895	1
2	E	1	0.3885929	0.08219178	18.25	0.2234717	1
3	E	1	0.3810701	0.08219178	19.99	0.2303576	1
4	D	1	0.3747726	0.08493151	16.99	0.2217274	1
5	F	1	0.3683521	0.08219178	21.99	0.2103212	1
6	E	1	0.3603845	0.08493151	20.99	0.2038452	1
7	C	1	0.3482239	0.08219178	14.65	0.2016963	1
8	D	1	0.3424083	0.08493151	17.86	0.1757963	1
9	E	1	0.3417710	0.08219178	18.25	0.2119356	1
10	D	1	0.3349551	0.08219178	16.99	0.2101195	1
11	G	1	0.3311023	0.16712329	28.49	0.1960152	1
12	E	1	0.3300907	0.08493151	19.99	0.1939551	1
13	E	1	0.3237347	0.08219178	20.99	0.1843632	1
14	F	1	0.3194927	0.08493151	23.99	0.1828363	1
15	D	1	0.3188478	0.08219178	16.99	0.1926076	1
16	D	1	0.3188059	0.08219178	15.61	0.1786616	1
17	D	1	0.3180958	0.08219178	16.99	0.1927095	1
18	E	1	0.3173502	0.08219178	18.25	0.2100631	1
19	D	1	0.3167170	0.08219178	17.86	0.1863695	1
20	F	1	0.3141694	0.08493151	21.99	0.1983592	1
21	F	1	0.3134638	0.08493151	22.99	0.1863539	1
22	D	1	0.3127579	0.08493151	17.57	0.1775130	1
23	E	1	0.3108941	0.08493151	19.99	0.1558714	1
24	D	1	0.3084512	0.08493151	16.99	0.1950842	1
25	D	1	0.3054969	0.08219178	17.57	0.1835774	1
26	E	1	0.3045426	0.08493151	19.19	0.1863054	1
27	E	1	0.2992594	0.08219178	18.25	0.1758397	1
28	D	1	0.2958849	0.08493151	17.57	0.1610419	1
29	D	1	0.2956222	0.08219178	16.99	0.1711061	1
30	F	1	0.2936598	0.16986301	24.99	0.1789931	1

31	D	1	0.2934079	0.08219178	16.55	0.1748808	1
32	D	1	0.2891969	0.08219178	17.57	0.1741944	1
33	D	1	0.2880580	0.08219178	17.57	0.1596442	1
34	D	1	0.2868024	0.08219178	15.61	0.1641705	1
35	D	1	0.2867162	0.08219178	16.99	0.1825354	1
36	F	1	0.2857390	0.08493151	25.78	0.1695170	1
37	F	1	0.2833309	0.33424658	23.99	0.1581378	1
38	E	1	0.2830385	0.16712329	19.99	0.1699042	1
39	E	1	0.2805613	0.16712329	20.99	0.1577247	1
40	F	1	0.2796837	0.33424658	24.99	0.1562642	1
41	D	1	0.2788522	0.08493151	15.61	0.1617529	1
42	D	1	0.2776644	0.08493151	17.86	0.1559790	1
43	E	1	0.2763918	0.16712329	19.99	0.1647577	1
44	F	1	0.2756360	0.24931507	21.99	0.1635852	1
45	D	1	0.2753722	0.08219178	17.86	0.1797844	1
46	D	1	0.2727943	0.08493151	15.61	0.1792197	1
47	F	1	0.2723462	0.25205479	21.99	0.1508406	1
48	D	1	0.2701723	0.08219178	16.55	0.1642142	1
49	D	1	0.2658265	0.08219178	16.55	0.1664469	1
50	F	1	0.2654165	0.58630137	25.78	0.1476242	1

We are considering loans with more than 20% return from first tile for our analysis here. There are total 338 loans which have more than 20% return.

Loan Grade	No of Loans
B	4
C	71
D	100
E	91
F	59
G	12

Maximum loan with highest return is for loan grade D and E. So, we should target these grade people for higher returns. Then, next we can target loan grade C and F people and at last grade G and B.
Similarly, we can do the same with each tile and make our investment plan to get higher returns.

On Test Data:

Tile	Count	Avg. Predicted Return	No. of Defaults	Avg Actual return	Minimum Return	Maximum Return	Average Term	Total A	Total B	Total C	Total D	Total E	Total F
1	2779	0.05622	331	0.06803	-0.3222	0.29221	2.13515	12	868	1191	576	124	7
2	2779	0.04566	342	0.05303	-0.2914	0.27526	2.19508	271	1336	830	278	61	3
3	2779	0.04096	320	0.0504	-0.3333	0.31905	2.22792	705	1119	682	219	45	8
4	2778	0.03738	317	0.04518	-0.3226	0.29161	2.24485	979	1000	530	213	48	7
5	2779	0.03431	310	0.04334	-0.3117	0.2494	2.25201	1129	891	527	186	42	4
6	2779	0.03119	332	0.04165	-0.3333	0.21858	2.27955	1133	834	567	186	51	8
7	2778	0.02762	366	0.04105	-0.3213	0.25717	2.31466	1031	804	616	246	76	5
8	2779	0.02323	470	0.04061	-0.3116	0.27104	2.29763	749	798	796	343	77	12
9	2779	0.01688	554	0.03797	-0.3228	0.40222	2.33949	522	723	947	443	123	18
10	2778	0.00164	790	0.03141	-0.3224	0.40787	2.37523	163	496	1004	696	340	68

From test data as well, we can see that the first decile has the maximum average actual return. To know more about people who can be targeted for maximum return within this decile group, we can investigate this decile more based on grades. There are 27,787 loans in first tile.
Considering 444 loans with return more than 12% for our analysis here.

Loan Grade	No of Loans
B	20
C	144
D	219
E	55
F	5
G	1

Loan grade D has the highest return. Second highest is grade C. Based on this, we can majorly target people from grade D and C from this group for better returns.

Performance by deciles for GBM:

On Training Data:

Tile	Count	Avg. Predicted Return	No. of Defaults	Avg Actual return	Minimum Return	Maximum Return	Average Term	Total A	Total B	Total C	Total D	Total E	Total F
1	11054	0.02305	2087	0.02028	-0.3333	0.21566	2.37118	7644	2799	524	73	14	0
2	11054	0.00522	3433	0.00608	-0.3333	0.22586	2.4345	4988	4633	1273	138	21	1
3	11054	-0.0046	4097	-0.0042	-0.3231	0.27279	2.47009	3277	5444	1990	314	28	1
4	11053	-0.0127	4766	-0.0073	-0.3333	0.30845	2.51068	1888	5239	3319	570	36	1
5	11054	-0.0199	5416	-0.0172	-0.3224	0.34822	2.54555	1074	4622	4266	1006	82	4

6	11054	-0.0269	5873	-0.0241	-0.3333	0.36038	2.58159	516	3859	5024	1498	144	13
7	11053	-0.0343	6604	-0.0372	-0.3333	0.31885	2.63898	244	2626	5678	2121	344	38
8	11054	-0.0421	7008	-0.0447	-0.3333	0.3181	2.66735	90	1778	5506	2991	659	30
9	11054	-0.0519	7503	-0.0553	-0.3333	0.38107	2.70978	32	758	4755	4144	1294	69
10	11053	-0.0714	8433	-0.0717	-0.3333	0.50199	2.7655	11	178	2360	4255	3263	837

On Test Data:

Tile	Count	Avg. Predicted Return	No. of Defaults	Avg Actual return	Minimum Return	Maximum Return	Average Term	Total A	Total B	Total C	Total D	Total E	Total F
1	2779	0.00396	95	0.04043	-0.3022	0.14501	2.22617	2316	463	0	0	0	0
2	2779	-0.0009	163	0.04071	-0.3131	0.20485	2.25334	1892	856	31	0	0	0
3	2779	-0.0046	221	0.04158	-0.3231	0.22433	2.26195	1279	1388	109	3	0	0
4	2778	-0.0088	293	0.04326	-0.3228	0.207	2.25579	681	1881	199	17	0	0
5	2779	-0.0137	323	0.04852	-0.3333	0.26533	2.25642	392	1675	645	66	1	0
6	2779	-0.019	386	0.05235	-0.3125	0.2227	2.24943	128	1267	1190	187	6	1
7	2778	-0.0253	489	0.05139	-0.3221	0.27675	2.25489	6	802	1512	421	34	2
8	2779	-0.0325	568	0.05063	-0.3333	0.27526	2.2898	0	429	1569	679	99	2
9	2779	-0.0404	682	0.0477	-0.3213	0.32899	2.27669	0	104	1518	894	227	29
10	2778	-0.0517	912	0.0361	-0.3224	0.40787	2.33705	0	4	917	1119	620	106

Performance by deciles for GLM:

On Training Data:

Tile	Count	Avg. Predicted Return	No. Of Defaults	Avg Actual return	Min Return	Max Return	Average Term	Total A	Total B	Total C	Total D	Total E	Total F
1	11054	0.02305	2087	0.02028	-0.3333	0.21566	2.37118	7644	2799	524	73	14	0
2	11054	0.00522	3433	0.00608	-0.3333	0.22586	2.4345	4988	4633	1273	138	21	1
3	11054	-0.0046	4097	-0.0042	-0.3231	0.27279	2.47009	3277	5444	1990	314	28	1
4	11053	-0.0127	4766	-0.0073	-0.3333	0.30845	2.51068	1888	5239	3319	570	36	1
5	11054	-0.0199	5416	-0.0172	-0.3224	0.34822	2.54555	1074	4622	4266	1006	82	4
6	11054	-0.0269	5873	-0.0241	-0.3333	0.36038	2.58159	516	3859	5024	1498	144	13
7	11053	-0.0343	6604	-0.0372	-0.3333	0.31885	2.63898	244	2626	5678	2121	344	38
8	11054	-0.0421	7008	-0.0447	-0.3333	0.3181	2.66735	90	1778	5506	2991	659	30
9	11054	-0.0519	7503	-0.0553	-0.3333	0.38107	2.70978	32	758	4755	4144	1294	69
10	11053	-0.0714	8433	-0.0717	-0.3333	0.50199	2.7655	11	178	2360	4255	3263	837

On Test Data:

Tile	Count	Avg. Predicted Return	No. of Defaults	Avg Actual return	Minimum Return	Maximum Return	Average Term	Total A	Total B	Total C	Total D	Total E	Total F
1	2779	0.00405	96	0.04065	-0.3022	0.20485	2.2208	2279	500	0	0	0	0
2	2779	-0.0009	164	0.04052	-0.3131	0.22433	2.25733	1849	902	28	0	0	0
3	2779	-0.0046	227	0.04156	-0.3231	0.19108	2.25421	1251	1405	120	3	0	0
4	2778	-0.0088	287	0.04325	-0.3228	0.207	2.2555	755	1790	215	18	0	0
5	2779	-0.0137	317	0.04893	-0.3333	0.26533	2.26439	432	1632	636	78	1	0
6	2779	-0.0191	394	0.05208	-0.3125	0.23694	2.24389	121	1273	1195	183	6	1
7	2778	-0.0253	483	0.05135	-0.3221	0.27675	2.25014	7	799	1514	415	40	2
8	2779	-0.0325	575	0.0502	-0.3333	0.27526	2.31191	0	463	1544	668	100	3
9	2779	-0.0403	671	0.04858	-0.3213	0.32899	2.27637	0	101	1514	889	247	23
10	2778	-0.0517	918	0.03554	-0.3224	0.40787	2.32698	0	4	924	1132	593	111

As seen from the above deciles, in all the models the higher-grade loans such as A and B have higher number of loans with lesser default rates but lower annual Returns. On the other Hand, the number of loans in the lower grades i.e. C and below have better annual rates but with higher rates. Hence, we see a trade-off between default rates (risk appetite) and higher returns.

The type of loan to be selected for investment depends upon on the risk appetite of an investor. This could have various underlying factors associated with the investor such as financial status, annual income, Assets, type of employment and other underlying background factors. An investor with a higher risk appetite and looking forward to a higher return can move ahead to invest in the lower grade loans, whereas an investor who does not wish to take risks and can settle for a lesser return can invest in higher grade loans.

Hence the tradeoff between default rate and annual returns can be applied variedly different range of investors and is subjective to investors mood of risk appetite.

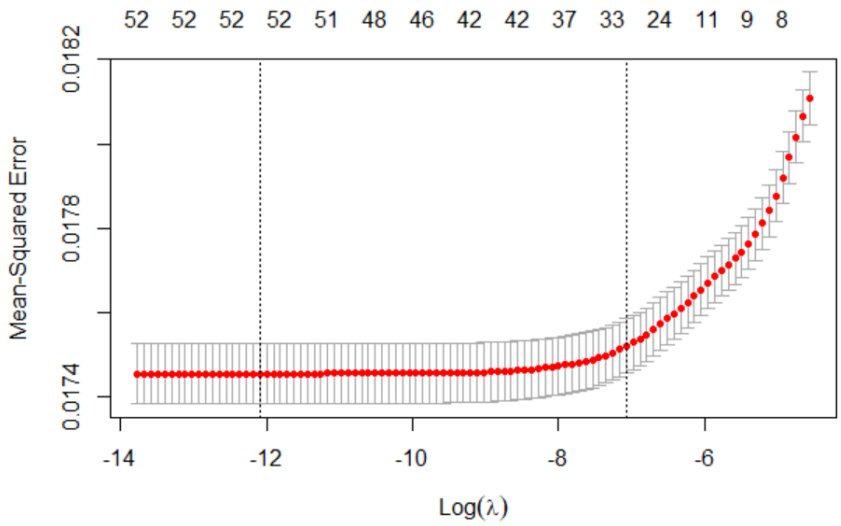
4. As seen in data summaries and your work in the first assignment, higher grade loans are less likely to default, but also carry lower interest rates; many lower grad loans are fully paid, and these can yield higher returns. One approach may be to focus on lower grade loans (C and below), and try to identify those which are likely to be paid off. Develop models from the data on lower grade loans, and check if this can provide an effective investment approach. Compare performance of models from different methods (glm, gbm, rf).
Can this provide a useful approach for investment? Compare performance with that in Question 3.

Since, there are many fully paid lower grade loans which can yield higher returns, we will drop the higher-grade loans (Grade A and Grade B) from the data set and develop model. We developed model using GLM as this is our best model for set of attributes considered.

Mean-squared error is calculated for both lambda.min and lambda.1se.

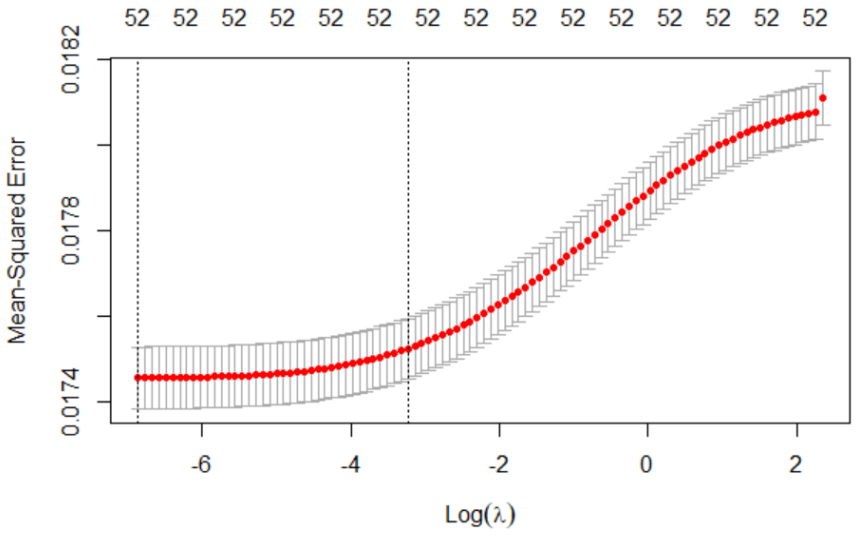
Lasso:

```
Measure: Mean-Squared Error

      Lambda   Measure         SE Nonzero
min 0.000537 0.0001950 5.604e-06       4
1se 0.001494 0.0002004 5.930e-06       3
```



Ridge:

```
Measure: Mean-Squared Error

    Lambda   Measure         SE Nonzero
min  0.013 0.0003896 7.092e-06     110
1se  0.013 0.0003896 7.092e-06     110
```

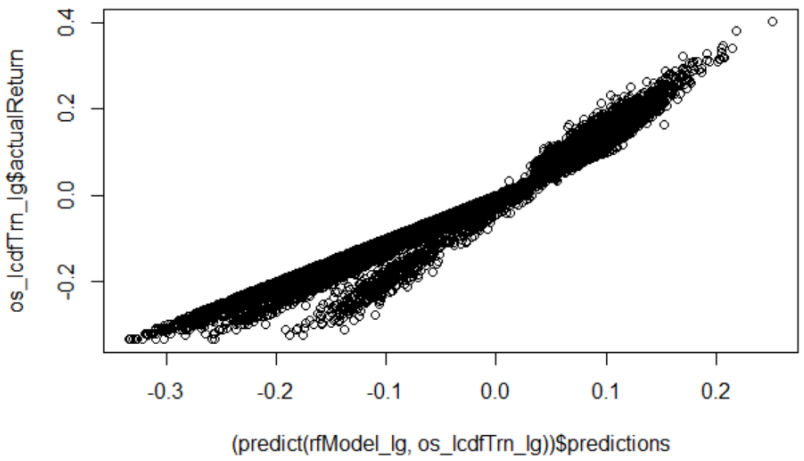


We have also developed the model on lower grades loan data using random forest to evaluate performance.

Random forest for lower grades:

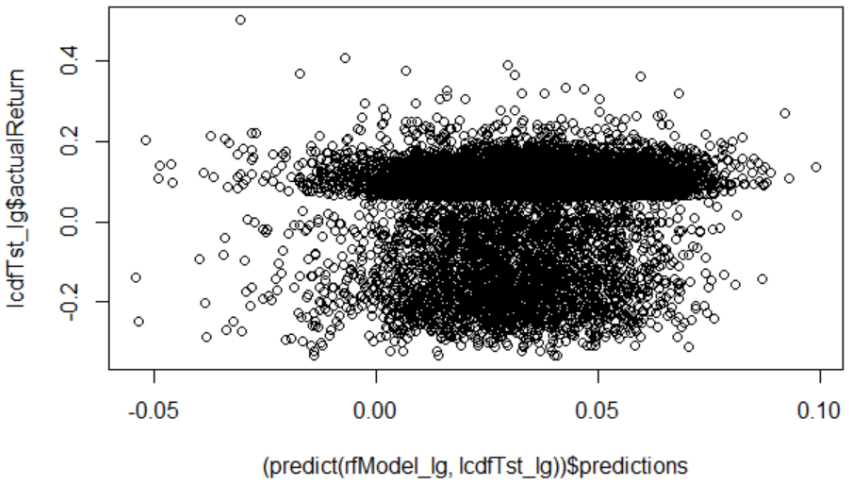
```
Growing trees.. Progress: 96%. Estimated remaining time: 1 seconds.
Computing permutation importance.. Progress: 39%. Estimated remaining time:
47 seconds.
Computing permutation importance.. Progress: 78%. Estimated remaining time:
17 seconds.
[1] 0.02619315
[1] 0.1110879
```

Random forest model on oversampled Training data:



Tile	Count	Avg. Predicted Return	No. of Defaults	Avg Actual return	Minimum Return	Maximum Return	Average Term	Total C	Total D	Total E	Total F
1	4417	0.10595	84	0.14978	0.09073	0.40222	1.01802	1426	2023	785	156
2	4416	0.08317	106	0.11147	0.07669	0.18387	1.70549	2576	1375	426	33
3	4416	0.07265	134	0.09546	0.05975	0.15966	2.24954	2819	1289	295	12
4	4416	0.06429	171	0.08225	0.05178	0.16263	2.64715	3433	851	123	9
5	4416	0.05536	338	0.07288	0.03311	0.11851	2.85748	3899	449	67	1
6	4416	0.01346	3613	0.01293	-0.096	0.11399	2.86492	2772	1171	399	65
7	4416	-0.0589	4416	-0.0693	-0.2128	-0.0257	3	2539	1328	419	119
8	4416	-0.1232	4416	-0.1378	-0.3106	-0.0944	3	2544	1320	456	87
9	4416	-0.1775	4416	-0.1899	-0.3224	-0.1528	3	2387	1366	570	78
10	4416	-0.2402	4416	-0.2501	-0.3333	-0.2041	3	2176	1433	650	116

Random forest model on oversampled Test data:



Tile	Count	Avg. Predicted Return	No. of Defaults	Avg Actual return	Minimum Return	Maximum Return	Average Term	Total C	Total D	Total E	Total F
1	1216	0.06375	199	0.07213	-0.3106	0.36038	2.08735	681	430	97	7
2	1215	0.05259	220	0.05983	-0.2914	0.30454	2.21027	839	288	83	5
3	1215	0.04634	214	0.06176	-0.31	0.33009	2.17565	850	295	66	4
4	1216	0.04121	227	0.06025	-0.3333	0.33496	2.23704	823	312	74	6
5	1215	0.03669	254	0.05439	-0.322	0.31905	2.24405	834	290	80	10
6	1215	0.03215	255	0.05103	-0.3221	0.36605	2.27602	821	316	72	5
7	1216	0.02732	332	0.03712	-0.3103	0.38859	2.34794	811	317	77	10
8	1215	0.02195	331	0.041	-0.3099	0.30515	2.39087	742	346	109	18
9	1215	0.01513	354	0.03757	-0.3333	0.32649	2.37603	714	367	112	18
10	1215	0.00049	425	0.02657	-0.3333	0.50199	2.3864	511	405	228	60

On the lower grade loans in the test data, as number of defaults increases, average actual return decreases. However, in the Grades E and F, we see the number of the datapoint are comparatively less than in Loan Grade C and D. Hence investors having a considerable risk appetite can invest in Loan Grades C and D as it would give a considerable returns given a certain risk rather than in loan Grades E and F where although returns are high these are more to default and also the datapoints with returns in these loans are considerably low.