1. (a) Develop boosted tree models (using either gbm or xgBoost) to predict loan_status. Experiment with different parameters using a grid of parameter values. Use cross-validation. Explain the rationale for your experimentation. How does performance vary with parameters, and which parameter setting you use for the 'best' model.

Model performance should be evaluated through use of same set of criteria as for the earlier models - confusion matrix based, ROC analyses and AUC, cost-based performance.

Provide a table with comparative evaluation of all the best models from each method; show their ROC curves in a combined plot. Also provide profit-curves and 'best' profit' and associated cutoff. At this cutoff, what are the accuracy values for the different models?

Ans: We have used XgBoost to predict loan_status.

We are varying max-depth and eta(learning-rate) in this model.

When we increase eta, the model converges quicker. It helps in making the computation faster because we need to input less rounds, but it does not help in reaching the best optimum whereas decreasing eta makes computation slower and makes easier in reaching the best optimum.

Increasing the makes-depth makes the model more complex and the model is likely to overfit.

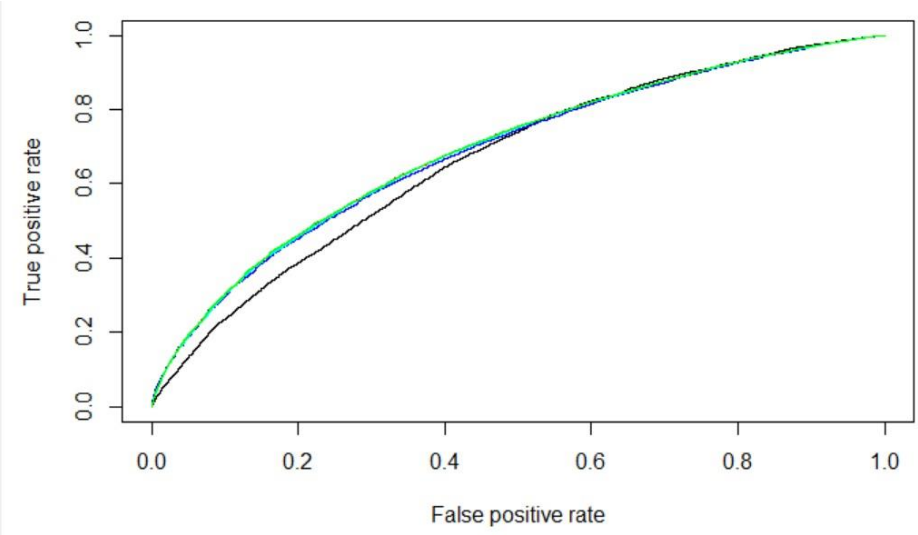
We have considered AUC as Performance Measure

max_depth	eta	Best Tree	Best Performance(AUC)
5	0.01	200	0.6806378
7	0.01	200	0.6810946
5	0.05	185	0.691004
7	0.05	135	0.6882082
5	0.1	77	0.689712
7	0.1	68	0.6868118

Variable Importance:

Feature	Gain	Cover	Frequency
int_rate	0.361239	0.17909	0.058538
dti	0.048539	0.053479	0.060851
acc_open_past_24mths	0.046623	0.065531	0.039796
loan_amnt	0.0316	0.048294	0.041647
annual_inc	0.031005	0.032503	0.045349
tot_hi_cred_lim	0.030729	0.036672	0.042342
mo_sin_old_rev_tl_op	0.023996	0.021009	0.033087
emp_length.n/a	0.021718	0.044999	0.024526
bc_open_to_buy	0.018707	0.021746	0.030541

total_rev_hi_lim	0.018402	0.016628	0.030773



ROC Curve

LEGENDS:

Black – Random Forest,

Red – glm Vanilla,

Blue – glm Weights,

Cyan – glm ridge,

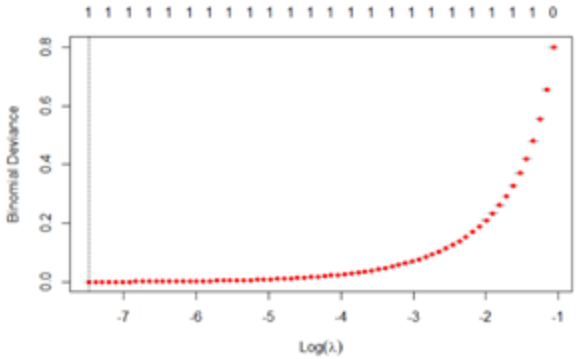
Green – glm lasso,

Yellow – XgBoost.

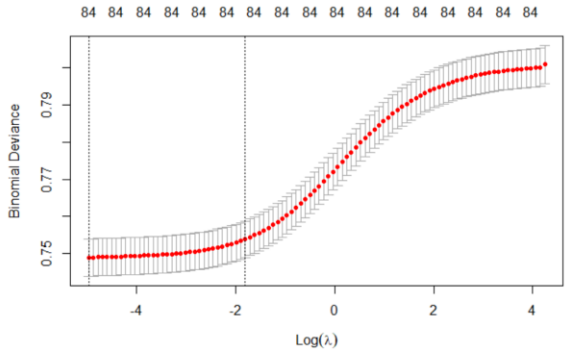
2. (a) Develop linear (glm) models to predict loan_status. Experiment with different parameter values, and identify which gives 'best' performance. Use cross-validation. Describe how you determine 'best' performance. How do you handle variable selection? Experiment with Ridge and Lasso, and show how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

Ans: The performance of the model was presented in the table below for all GLM, GLM ridge, GLM lasso, GLM elastic models. The best performance was determined based on the values of accuracy, AUC of ROC curves.

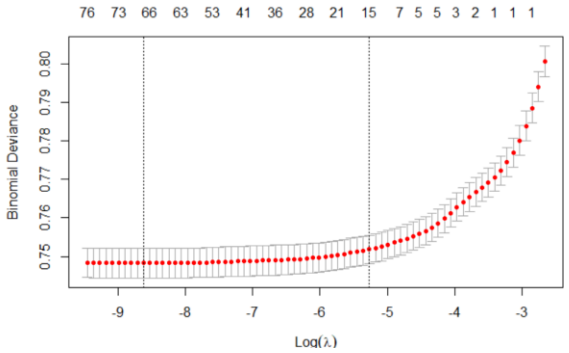
GLM model (vanilla) no alpha value:



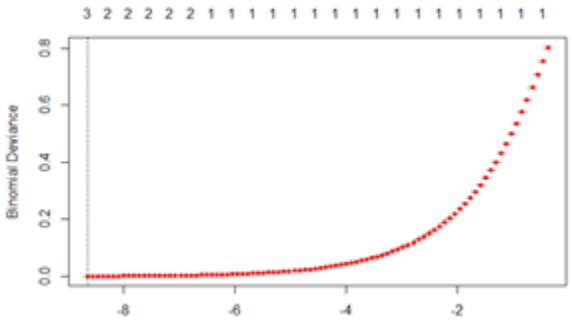
GLM Ridge regression model:(alpha=0)



GLM Lasso Regression model:(alpha=1)



GLM Elastic Regression model:(alpha=0.5)



alpha	lambdaMin	lambda1se	Lambda Min	AUC values
none	0.000218	0.006197	0.000586	0.681

0	0.00696118	0.1645964	0.0141981	0.69540
1	**0.00018064**	**0.0051448**	**0.0004043**	**0.68652**
0.5	0.00027330	0.0123939	0.000612	0.68642

From the above Performance Measures we can say that GLM Ridge regression fits the data better

Handling variable selection:

After cross-validation and regression techniques, we identify and remove variables with zero coefficients. Post that we build model using variables with non-zero coefficients.

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

Ans: The loss Function : $J(w) = \sum(x_i w_i - y_i)^2$

J(w) is the predicted outcome and x_i are the input data

And the link function is: ln(p/(1-p))

; p is the probability of the linear function

(c) Compare performance of models with that of random forests (from last assignment) and gradient boosted tree models.

Ans:

Model type	Accuracy(on test data)	AUC(unbalanced data)	AUC(with weights)
Random Forest	0.860	0.658	-
GLM Vanilla	0.859	0.68192	0.69778
GLM ridge	**0.8593**	**0.69540**	**0.70484**
GLM lasso	0.8591	0.68652	0.69668
GLM elastic	0.8592	0.68642	0.69745
xgboost		0.691004	-

From the performance table above, the GLM Ridge Regression is the best model among the models done.

(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?

Ans: Top 5 important variable are shown below by different models:

Decision Tree	Random Forest	GLM	xgboost
Employee Title	Sub-Grade	Home_ownership	Interest rate
Earliest credit	Interest rate	sub_grade	dti

zipcode	Annual Income	grade	acc_open_past_24mths
Employment length	Total high credit limit	acc_open_past_24mths	Loan amount
Title of loan	Total revolving credit limit	verification_status	Annual income

Different models have shown different variables of importance. It can be due to different loss functions and split gain functions.

(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 ?

Ans: We did run the models with different train and test data size's (0.6,0.7 and 0.8) and the results does not change. The error of the model was almost same for 60:40, 70:30, 80:20 partitions of training and testing data which are 0.1163, 0.1158, 0.1178, respectively. This suggests that larger training samples does not increase the model performance.

Balancing training data gave us the results which are consistent in nature. Balancing the data was done by adding the weights to each of the rows of the dataset based on the imbalances of the original data. The performance of the model after balancing the data is given in the above table under part C.

3. Develop models to identify loans which provide the best returns. Explain how you define returns? Does it include the Lending Club's service costs? Develop glm, rf, gbm/xgb models for this. Show how you systematically experiment with different parameters to find the best models. Compare model performance – explain what performance criteria do you use, and why.

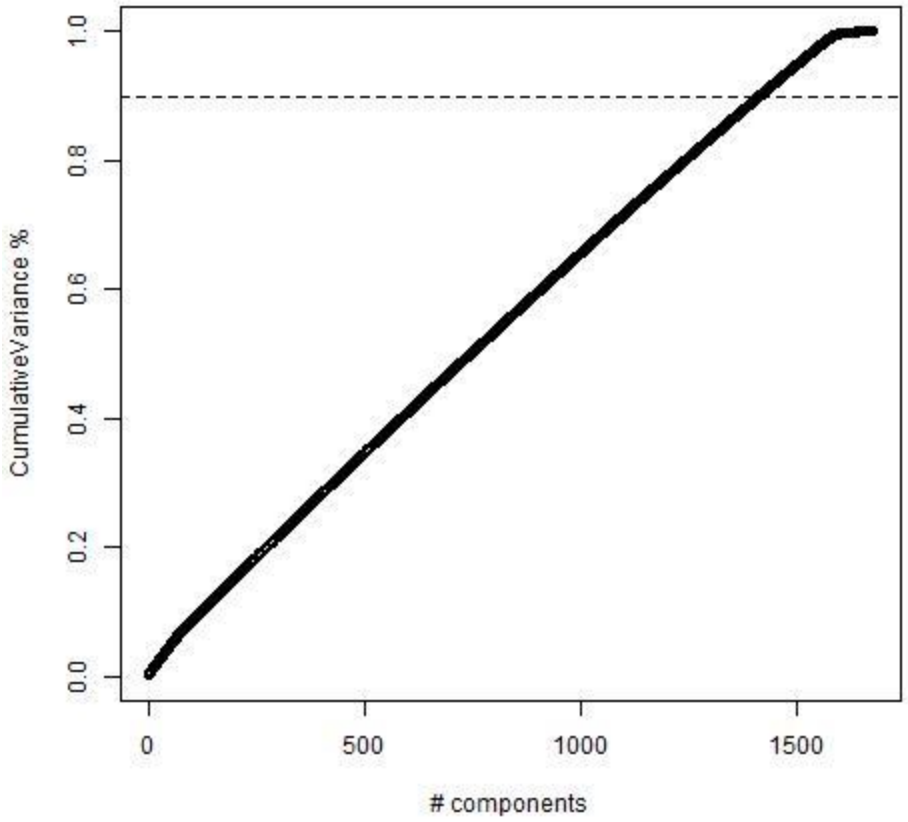
Ans: Returns are calculated using the following formula:

(total_pymnt - funded_amnt) / funded_amnt) * (1/actualTerm) * 100

Where,

actualTerm = +(issue_d - last_pymnt_d_new)

<u>Using PCA for dimentionality reduction,</u>



We can observe that increase in variance is almost linear. I.e., we need almost more than 90% of the PCA components to explain 90% of variance in data. So, using PCA will not be helpful in reducing dimensionality (I.e. reducing the size of data without losing information).

GLM:

alpha	lambdaMin	lambda1se	MSE (Test Data)
0	0.102293279	1.667125566	67.88946827
0.5	0.007702553	0.349301399	67.82812849
1	0.004638876	0.278093304	67.81724096

Decile Performance:

tile	count	avgpredRet	numDefaults	avgActRet	minRet	maxRet	avgTer	totA	totB	totC	totD	totE	totF

1	3000	8.02	488	8.21	-33.33	40.25	2.21	5	393	879	1088	439	173
2	3000	6.81	460	6.83	-32.13	31.97	2.19	32	897	1057	773	229	11
3	3000	6.19	446	6.21	-32.23	26.27	2.19	103	1144	1011	588	145	8
4	3000	5.73	455	5.56	-33.33	36.86	2.23	263	1109	1077	436	113	2
5	3000	5.33	406	5.42	-32.20	22.36	2.20	466	1168	962	345	59	0
6	3000	4.97	429	4.75	-31.22	24.75	2.26	754	1096	879	229	42	0
7	3000	4.62	396	4.35	-30.93	23.86	2.27	955	1080	798	136	30	1
8	3000	4.26	387	3.89	-32.53	19.23	2.28	1198	1116	594	81	11	0
9	3000	3.81	324	3.96	-31.28	26.80	2.33	1427	1093	423	49	8	0
10	3000	3.02	368	3.14	-32.29	16.73	2.38	1673	1069	239	15	4	0

RandomForest:

HyperParameter tuning

num_trees	mtry	depth	MSE (Test Data)
200	2	5	68.67951
500	2	5	68.69038
800	2	5	68.70328
200	5	5	68.2649
500	5	5	68.23703
800	5	5	68.24636
200	10	5	67.93686
500	10	5	67.93156
800	10	5	67.93141
200	15	5	67.75517
500	15	5	67.73904
800	15	5	67.74146
200	2	10	68.27134
500	2	10	68.26422
800	2	10	68.26514
200	5	10	67.72212
500	5	10	67.6971
800	5	10	67.69369
200	10	10	67.38021
500	10	10	67.36919
800	10	10	67.37427

200	15	10	67.26111
500	15	10	67.25369
800	15	10	67.24066
200	2	15	67.98846
500	2	15	67.97161
800	2	15	67.96154
200	5	15	67.45488
500	5	15	67.42432
800	5	15	67.41947
200	10	15	67.23391
500	10	15	67.22644
800	10	15	67.22879
200	15	15	67.23912
500	15	15	67.1906
800	15	15	67.16449
200	2	20	67.84799
500	2	20	67.80694
800	2	20	67.78673
200	5	20	67.39889
500	5	20	67.35403
800	5	20	67.33274
200	10	20	67.43309

500	10	20	67.34439
800	10	20	67.3265
200	15	20	67.47593
500	15	20	67.36494
800	15	20	67.323

Decile Performance:

tile	count	avgpredRet	numDefaults	avgActRet	minRet	maxRet	avgTer	totA	totB	totC	totD	totE	totF
1	3000	8.02	488	8.21	-33.33	40.25	2.21	5	393	879	1088	439	173
2	3000	6.81	460	6.83	-32.13	31.97	2.19	32	897	1057	773	229	11
3	3000	6.19	446	6.21	-32.23	26.27	2.19	103	1144	1011	588	145	8
4	3000	5.73	455	5.56	-33.33	36.86	2.23	263	1109	1077	436	113	2
5	3000	5.33	406	5.42	-32.20	22.36	2.20	466	1168	962	345	59	0
6	3000	4.97	429	4.75	-31.22	24.75	2.26	754	1096	879	229	42	0
7	3000	4.62	396	4.35	-30.93	23.86	2.27	955	1080	798	136	30	1

8	3000	4.26	387	3.89	-32.53	19.23	2.28	1198	1116	594	81	11	0
9	3000	3.81	324	3.96	-31.28	26.80	2.33	1427	1093	423	49	8	0
10	3000	3.02	368	3.14	-32.29	16.73	2.38	1673	1069	239	15	4	0

XGBoost Hyper-parameter tuning (Cross-validated with nFolds=10):

max_depth	**ETA**	**Lambda**	**BestTree**	**RMSE Value**	**MSE (Test Data)**
5	0.1	0.5	53	8.23914	67.88343
10	0.1	0.5	33	8.265954	68.326
5	0.5	0.5	5	8.277324	68.51409
10	0.5	0.5	3	8.365142	69.9756
5	0.1	0	59	8.243996	67.96347
10	0.1	0	53	8.23914	67.88343
5	0.5	0	53	8.23914	67.88343
10	0.5	0	53	8.23914	67.88343

Decile Performance:

tile	count	avgpredRet	numDefaults	avgActRet	minRet	maxRet	avgTer	totA	totB	totC	totD	totE	totF
1	3000	8.12	477	8.54	-32.13	40.25	2.14	0	111	944	1354	456	123
2	3000	6.69	477	6.90	-32.22	26.48	2.19	0	594	1502	734	151	15
3	3000	6.11	469	5.97	-33.33	31.97	2.21	0	1151	1336	419	83	9
4	3000	5.71	435	5.77	-32.23	34.20	2.23	4	1461	1151	306	68	8
5	3000	5.33	487	5.01	-33.33	23.34	2.28	66	1666	964	232	62	10
6	3000	4.95	421	4.96	-33.33	26.20	2.26	318	1731	719	173	53	5
7	3000	4.58	374	4.50	-32.29	36.86	2.25	875	1471	452	151	43	7
8	3000	4.23	318	3.96	-31.14	23.86	2.26	1649	933	274	108	33	3
9	3000	3.90	264	3.71	-31.27	24.77	2.29	2208	474	216	65	32	3

10	3000	3.18	437	3.02	-32.20	26.80	2.42	1756	573	361	198	99	12

We have considered MSE as the evaluation criterion. We are predicting a continuous variable and comparing different models on same set of training and test data. MSE provides a comparative figure to evaluate models. Although it is sensitive to outliers but our test dataset is same so the effect will be nullified.

We have observed that Random Forest shows the least MSE I.e., 67.16 followed by GLM (67.81) and then XGBoost (67.88) for cross-validation set.

For test data, XGBoost has lowest MSE (66.98856), followed by GLM (67.21) and then RF (67.323). It seems that XGBoost performs better on test data than the other algorithms. Although it is computationally intensive, it provides slightly better results on unseen data in terms of MSE.

From the decile performance across different models, we can observe that XGBoost seems to give provide less defaults in the top decile.

4. Considering results from Questions 1 and 2 above – that is, considering the best model for predicting loan-status and that for predicting loan returns -- how would you select loans for investment? There can be multiple approaches for combining information from the two models - describe your approach, and show performance. How does performance here compare with use of single models?

We have selected XGBoost algorithm to model loan_status and returns based on AUC and MSE values obtained on test data from above parts.

We have a constraint on the amount of money that can be invested. Based on this constraint and the requirement of higher returns, we will use the following approach of combining 2 models.

We first select top 1 decile of the data obtained using returns model. Then we calculate expected returns using scores given by loan_status model.

Expected Value = (probability of Full Paid) * (Predicted Actual Returns)

Now based on these scores, we divide the loans into top 20 tiles and evaluate our options of investments.

<u>Comparing with a single model:</u>

Returns model,

The loans in top 1 decile of this model gives average actual returns of 8.12%

tile	**count**	**avgPredRet**	**numDefaults**	**avgActRet**	**minRet**	**maxRet**	**avgTer**	**totA**	**totB**	**totC**	**totD**	**totE**	**totF**

1	3000	8.12	477	8.54	-32.13	40.25	8.54	0	111	944	1354	456	123
2	3000	6.69	477	6.90	-32.22	26.48	6.90	0	594	1502	734	151	15
3	3000	6.11	469	5.97	-33.33	31.97	5.97	0	1151	1336	419	83	9
4	3000	5.71	435	5.77	-32.23	34.20	5.77	4	1461	1151	306	68	8
5	3000	5.33	487	5.01	-33.33	23.34	5.01	66	1666	964	232	62	10
6	3000	4.95	421	4.96	-33.33	26.20	4.96	318	1731	719	173	53	5
7	3000	4.58	374	4.50	-32.29	36.86	4.50	875	1471	452	151	43	7
8	3000	4.23	318	3.96	-31.14	23.86	3.96	1649	933	274	108	33	3
9	3000	3.90	264	3.71	-31.27	24.77	3.71	2208	474	216	65	32	3
10	3000	3.18	437	3.02	-32.20	26.80	3.02	1756	573	361	198	99	12

Loan_Status Model,

The loans in top 1 decile of this model gives average actual returns of 4.01%

tile	**count**	**avgSc**	**numDefaults**	**avgActRet**	**minRet**	**maxRet**	**avgTer**	**totA**	**totB**	**totC**	**totD**	**totE**	**totF**

1	3000	0.97	82	4.02	-30.27	13.18	2.18	2886	113	1	0	0	0
2	3000	0.94	167	4.51	-31.22	25.41	2.26	2050	933	17	0	0	0
3	3000	0.93	244	4.77	-30.12	19.34	2.27	1128	1763	101	8	0	0
4	3000	0.91	291	5.23	-31.28	25.80	2.25	533	2090	347	26	4	0
5	3000	0.89	330	5.73	-32.23	22.14	2.22	184	1926	796	86	6	2
6	3000	0.87	373	6.14	-32.53	22.18	2.21	70	1558	1161	185	24	2
7	3000	0.84	484	6.04	-32.29	27.18	2.23	20	973	1555	394	53	5
8	3000	0.81	571	5.95	-33.33	27.52	2.23	4	526	1668	649	131	19
9	3000	0.78	704	5.37	-33.33	34.20	2.30	1	211	1464	1056	235	33
10	3000	0.70	913	4.57	-33.33	40.25	2.39	0	72	809	1336	627	134

Combined Model,

tile2	**count**	**avgPredRet**	**numDefaults**	**avgActRet**	**minRet**	**maxRet**	**avgTer**	**totA**	**totB**	**totC**	**totD**	**totE**	**totF**

1	150	11.02	23	10.68	-30.88	23.34	10.68	0	0	7	73	53	16
2	150	9.66	23	9.08	-29.71	23.68	9.08	0	1	19	75	37	17
3	150	8.96	12	11.40	-19.37	27.18	11.40	0	2	45	71	21	9
4	150	8.73	27	7.87	-30.68	25.80	7.87	0	2	47	75	19	4
5	150	8.42	22	8.97	-27.11	20.79	8.97	0	4	56	71	17	1
6	150	8.29	16	9.17	-28.55	40.25	9.17	0	11	54	62	20	3
7	150	8.14	27	7.85	-24.62	20.57	7.85	0	10	60	61	15	4
8	150	8.08	18	9.37	-30.71	22.36	9.37	0	11	60	55	20	3
9	150	7.88	19	8.85	-24.33	22.84	8.85	0	16	66	50	15	3
10	150	7.77	16	8.29	-25.63	20.33	8.29	0	21	56	65	7	1
11	150	7.71	24	7.55	-26.38	24.14	7.55	0	12	77	45	15	1
12	150	7.69	20	8.71	-26.30	23.04	8.71	0	6	74	56	11	3
13	150	7.64	20	8.71	-32.13	27.52	8.71	0	7	74	51	13	5
14	150	7.58	29	7.80	-30.93	22.01	7.80	0	4	60	69	14	3

15	150	7.55	29	6.98	-27.54	23.77	6.98	0	2	66	65	14	3
16	150	7.52	24	8.56	-27.39	26.27	8.56	0	0	45	87	12	6
17	150	7.48	37	6.42	-29.85	22.86	6.42	0	1	38	80	29	2
18	150	7.50	30	7.42	-30.96	23.80	7.42	0	1	23	94	29	3
19	150	7.44	28	8.65	-25.09	21.86	8.65	0	0	11	95	38	5
20	150	7.40	33	8.43	-25.02	25.16	8.43	0	0	6	54	57	31

It combines loans with higher returns and higher probability of being paid off. The loans in the top 3 tile of this model gives an average actual return higher than 10% (10.7%, 9.1% and 11.4%). This approach will be useful for investors with high-risk and returns appetite.

5. 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 – for this, you can use one of the methods (glm, rf, or gbm/xgb) which you find to give superior performance from earlier questions.

Can this provide a useful approach for investment? Compare performance with that in Question 4.

For modelling on lower grade loans,

XGBoost gave the highest AUC value of 0.6216346 for the test data. (As compared to RF (0.599) and GLM (0.612))

Performance in across deciles is shown in following table:

tile	count	avgSc	numDefaults	avgAnnRet	minRet	maxRet	avgTer	totA	totB	totC	totD	totE	totF
1	1296	0.89	127	7.44	-32.21	25.41	2.21	0	0	1195	91	9	1
2	1296	0.86	180	6.85	-32.22	25.80	2.20	0	0	1132	149	13	2
3	1296	0.84	197	6.93	-31.11	27.18	2.17	0	0	1046	219	29	1

4	1296	0.83	218	6.67	-32.19	23.68	2.19	0	0	970	286	36	3
5	1296	0.81	240	6.49	-31.14	25.73	2.20	0	0	843	390	55	8
6	1296	0.80	264	6.15	-31.05	27.52	2.28	0	0	774	418	93	10
7	1296	0.78	294	5.77	-33.33	24.39	2.28	0	0	700	473	107	15
8	1296	0.76	323	5.26	-32.16	40.25	2.32	0	0	599	522	152	22
9	1296	0.73	385	4.68	-33.33	27.67	2.37	0	0	468	562	220	42
10	1295	0.66	434	4.10	-32.20	36.86	2.43	0	0	192	630	366	91

Comparing with the model in 4th part, the combined model gives loans with higher average returns than the model of lower grades. Although the default rate is lower in this model, the default rate in the top tile of the combined model is similar to the proportion of overall default rate but loans have higher average returns. This method can be useful for investors with medium risk and returns appetite.

6. Considering all your results, which approach(s) would you recommend for investing in LC loans? Explain your rationale.

Thinking from an investor's perspective, we can gauge that they are under a constraint of money and time. Based on their risk averse nature, we should use different models that rightly serve their risk appetite.

Based on the above analysis,

We have observed that XGboost seems to perform better but requires high computational resources.

Considering 3 types of investors based on their risk-returns appetite as Low, Medium, and High

For Low,

We will recommend results based on XGBoost loan_status model. As it gives us loans with highest probability to be paid-off. Although the loans will have lower returns, the investors in this category are not concerned about higher returns.

For Medium,

We will recommend combined model that we created in question 5. We observed that default rate slightly increases but the returns increase significantly. This combination will serve best to this category of investors.

For High,

We will recommend model created on lower grade loans. We have observed that loans in top tile of this approach gives highest returns as compared to approaches above. Investors in this category prioritize high returns so this model serves them best.