

Three blue spheres of varying sizes are positioned in the upper right and lower right areas of the page. Thin blue lines extend from the top left and top right corners towards the spheres.

## Bank Loan Default Prediction Model

Capstone Project- Notes II

Build a model to predict default loan that will help the bank to take required actions.

**NIVEDITA DEY - PGP BABI May'19**  
**12/04/2020**

## Contents

1. Key Points from Notes 1 .....	2
3. Model Building .....	5
3.1 Logistic Regression .....	5
3.1.1 Performance Metrics .....	6
3.1.2 Interpretation.....	7
3.2 CART .....	7
3.2.1 Post Pruning .....	8
3.2.2 Performance Metrics .....	8
3.2.3 Interpretation.....	9
3.3 Naïve Bayes .....	10
3.3.1 Performance Metrics .....	10
3.3.2 Interpretation.....	11
3.4 Model Comparison.....	12
4. Model Tuning .....	12
4.1 Random Forest.....	12
4.1.1 tuneRF .....	13
4.1.2 Important Variable.....	13
4.1.3 Performance Metrics .....	13
4.1.4 Interpretation.....	14
4.2 Bagging.....	14
4.2.1 Performance Metrics .....	15
4.2.2 Interpretation.....	16
4.3 Boosting .....	16
4.3.1 Important Variable.....	16
4.3.2 Performance Metrics .....	17
4.3.3 Interpretation.....	17
5. Model Comparison.....	18
6. Conclusion.....	19

## 1. Key Points from Notes 1

- The data contains the details of Loans which have been issued between June 2007 and December 2015 period.

Maximum last payment date for the loans is January 2016. Hence we can consider data is collected post January 2016. Based on the loan issue date, it shows Monthly frequency of data collection.

- There are 226,786 rows and 41 columns.

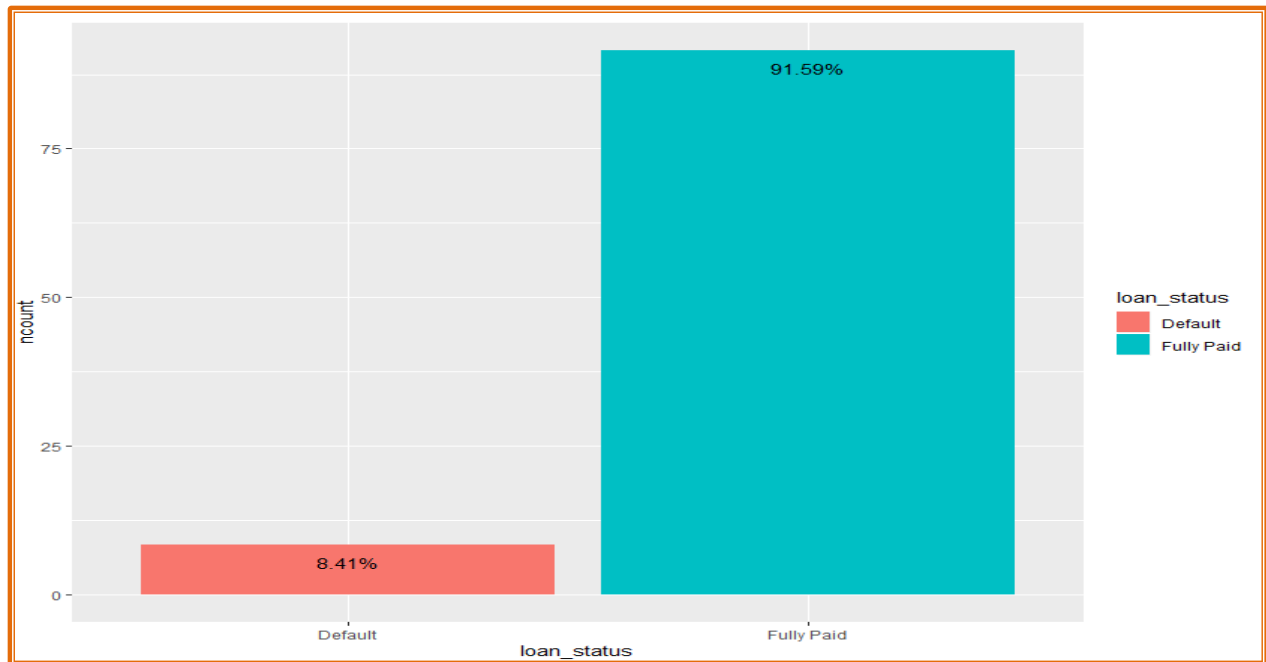
Out of which 25 are numeric columns, 11 character columns and 5 date columns.

The last variable 'loan\_status' is the dependent variable.

#	Fields	Description	Type
1	member_id	A unique Id for the borrower member.	Continuous
2	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.	Continuous
3	funded_amnt	The total amount committed to that loan at that point in time.	Continuous
4	funded_amnt_inv	The total amount committed by investors for that loan at that point in time.	Continuous
5	term	The number of payments on the loan. Values are in months and can be either 36 or 60.	Categorical
6	int_rate	Interest Rate on the loan	Continuous
7	installment	The monthly payment owed by the borrower if the loan originates.	Continuous
8	Grade	Assigned loan grade	Categorical
9	emp_length	Employment length in years. Possible values are between 0 and 10 where 0 means less than one year and 10 means ten or more years.	Categorical
10	home_ownership	The home ownership status provided by the borrower during registration. Our values are: RENT, OWN, MORTGAGE, OTHER.	Categorical
11	annual_inc	The self-reported annual income provided by the borrower during registration.	Continuous
12	verification_status	Status of the verification done	Categorical
13	issue_d	The month which the loan was funded	Date
14	pymnt_plan	Indicates if a payment plan has been put in place for the loan	Categorical
15	Desc	Loan description provided by the borrower	Categorical
16	Purpose	A category provided by the borrower for the loan request.	Categorical
17	addr_state	The state provided by the borrower in the loan application	Categorical
18	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	Continuous

#	Fields	Description	Type
		monthly income.	
19	delinq_2yrs	The number of 30+ days past-due incidences of delinquency in the borrower's credit file for the past 2 years	Continuous
20	earliest_cr_line	The month the borrower's earliest reported credit line was opened	Date
21	inq_last_6mths	The number of inquiries in past 6 months (excluding auto and mortgage inquiries)	Continuous
22	mths_since_last_delinq	The number of months since the borrower's last delinquency.	Continuous
23	open_acc	The number of open credit lines in the borrower's credit file.	Continuous
24	revol_bal	Total credit revolving balance	Continuous
25	revol_util	Revolving line utilization rate, or the amount of credit the borrower is using relative to all available revolving credit.	Continuous
26	total_acc	The total number of credit lines currently in the borrower's credit file	Continuous
27	out_prncp	Remaining outstanding principal for total amount funded	Continuous
28	out_prncp_inv	Remaining outstanding principal for portion of total amount funded by investors	Continuous
29	total_pymnt	Payments received to date for total amount funded	Continuous
30	total_pymnt_inv	Payments received to date for portion of total amount funded by investors	Continuous
31	total_rec_prncp	Principal received to date	Continuous
32	total_rec_int	Interest received to date	Continuous
33	total_rec_late_fee	Late fees received to date	Continuous
34	recoveries	post charge off gross recovery	Continuous
35	collection_recovery_fee	post charge off collection fee	Continuous
36	last_pymnt_d	Last month payment was received	Date
37	last_pymnt_amnt	Last total payment amount received	Continuous
38	next_pymnt_d	Next scheduled payment date	Date
39	last_credit_pull_d	The most recent month pulled credit for this loan	Date
40	application_type	Indicates whether the loan is an individual application or a joint application with two co-borrowers	Categorical
41	loan_status	Current status of the loan	Categorical

- We have renamed the column 'earliest\_cr\_line' to 'earliest\_cr\_line\_mnth' as it shows the month a borrower's earliest reported credit line was opened.
- The data is highly imbalanced. So while building the model we can either choose to undersample the minority class or oversample the majority class (SMOTE)



**Significant Variable :** Based on correlation Matrix, vif and annova we had successfully listed down the highly significant variables. They are:

1. member\_id,
2. terms,
3. installment,
4. grade,
5. emp\_length,
6. dti,
7. issue\_d,
8. revol\_bal, revol\_util,
9. total\_rec\_int,
10. total\_rec\_late\_fee,
11. last\_pymnt\_d,
12. last\_pymnt\_amnt,
13. last\_credit\_pull\_d

### 3. Model Building

We had created a dataset with the important 13 variables from Notes1 and the target variable. We then try various model building by splitting the data in 70:30 ratio (train:test) and check their performance.

We have even tried to SMOTE the data and build logistic regression model but there was no significant difference in performance metrics namely: accuracy, sensitivity and specificity. Hence, we have built various models on actual data only without any treatment for imbalance.

Please refer R-Code for Source Code of model with SMOTE data.

#### 3.1 Logistic Regression

Logistic regression is part of the supervised learning. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables..

There is significant change in log likelihood from the base model. Also based on the p-value we can reject the null hypothesis. Thus the model is valid.

```
Likelihood ratio test

Model 1: loan_status ~ member_id + installment + grade + emp_length +
  issue_d + dti + revol_bal + revol_util + total_rec_int +
  total_rec_late_fee + last_pymnt_d + last_pymnt_amnt + last_credit_pull_d
Model 2: loan_status ~ 1
#Df LogLik Df Chisq Pr(>Chisq)
1 29 -9324
2 1 -45807 -28 72965 < 2.2e-16 ***
---
signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

VIF is also below 5 for all variables.

```
> vif(LRModel_1)
      GVIF Df GVIF^(1/(2*Df))
member_id      8.713129 1      2.951801
installment    2.421037 1      1.555968
grade          1.385692 6      1.027556
emp_length     1.042981 11     1.001915
issue_d        9.762479 1      3.124497
dti            1.061773 1      1.030423
revol_bal      1.213830 1      1.101740
revol_util     1.180367 1      1.086447
total_rec_int  3.743820 1      1.934895
total_rec_late_fee 1.022400 1      1.011138
last_pymnt_d   1.021927 1      1.010904
last_pymnt_amnt 1.041254 1      1.020418
last_credit_pull_d 1.027991 1      1.013899
```

Based on response plot we choose cut off value as 0.4.

### 3.1.1 Performance Metrics

Train dataset:

cnt_non_resp	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks	cum_cnt	cum_resp_rate	lift
3172	80.02	12702	3172	0.95	0.02	0.93	15874	0.80	9.52
15334	3.40	13242	18506	0.99	0.13	0.86	31748	0.42	5.00
15828	0.29	13288	34334	1.00	0.24	0.76	47622	0.28	3.33
15839	0.22	13323	50173	1.00	0.35	0.65	63496	0.21	2.50
15867	0.04	13330	66040	1.00	0.45	0.55	79370	0.17	2.02
15869	0.03	13334	81909	1.00	0.56	0.44	95243	0.14	1.67
15873	0.01	13335	97782	1.00	0.67	0.33	111117	0.12	1.43
47614	0.02	13343	145396	1.00	1.00	0.00	158739	0.08	0.95

Confusion Matrix and Statistics		
Reference		
Prediction	0	1
0	143062	1025
1	2334	12318
Accuracy : 0.9788		
95% CI : (0.9781, 0.9795)		
No Information Rate : 0.9159		
P-Value [Acc > NIR] : < 2.2e-16		
Kappa : 0.8684		
McNemar's Test P-Value : < 2.2e-16		
Sensitivity : 0.92318		
Specificity : 0.98395		
Pos Pred Value : 0.84070		
Neg Pred Value : 0.99289		
Prevalence : 0.08406		
Detection Rate : 0.07760		
Detection Prevalence : 0.09230		
Balanced Accuracy : 0.95356		
'Positive' Class : 1		

Metrics	Value
Accuracy	0.978
Sensitivity	0.923
Specificity	0.983
KS	0.940
AUC	0.992
Gini Coefficient	0.902

Test dataset:

cnt_non_resp	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks	cum_cnt	cum_resp_rate	lift
1373	79.82	5431	1373	0.95	0.02	0.93	6804	0.80	9.52
6557	3.62	5677	7930	0.99	0.13	0.86	13607	0.42	5.00
6779	0.34	5700	14709	1.00	0.24	0.76	20409	0.28	3.33
6795	0.13	5709	21504	1.00	0.35	0.65	27213	0.21	2.50
6799	0.06	5713	28303	1.00	0.45	0.55	34016	0.17	2.02
6803	0.00	5713	35106	1.00	0.56	0.44	40819	0.14	1.67
6801	0.01	5714	41907	1.00	0.67	0.33	47621	0.12	1.43
20405	0.02	5719	62312	1.00	1.00	0.00	68031	0.08	0.95

Confusion Matrix and Statistics		
Reference		
Prediction	0	1
0	61354	482
1	958	5237
Accuracy : 0.9788		
95% CI : (0.9777, 0.9799)		
No Information Rate : 0.9159		
P-Value [Acc > NIR] : < 2.2e-16		
Kappa : 0.8676		
McNemar's Test P-Value : < 2.2e-16		
Sensitivity : 0.91572		
Specificity : 0.98463		
Pos Pred Value : 0.84536		
Neg Pred Value : 0.99221		
Prevalence : 0.08406		
Detection Rate : 0.07698		
Detection Prevalence : 0.09106		
Balanced Accuracy : 0.95017		
'Positive' Class : 1		

Metrics	Value
Accuracy	0.978
Sensitivity	0.915
Specificity	0.984
KS	0.939
AUC	0.992
Gini Coefficient	0.903

### 3.1.2 Interpretation

Based on the performance metrics of the model on testing data, we can say the model is good. Based on the test metrics we can interpret that:

1. The model will catch 92% of the customers who will default .
2. The model will catch 98% of the customers who will not default
3. Overall all accuracy is 98%
4. Out of the customers who are predicted as will default, 84% of them will actually default
5. Out of the customers who are predicted as will Not default, 99% of them will actually not default

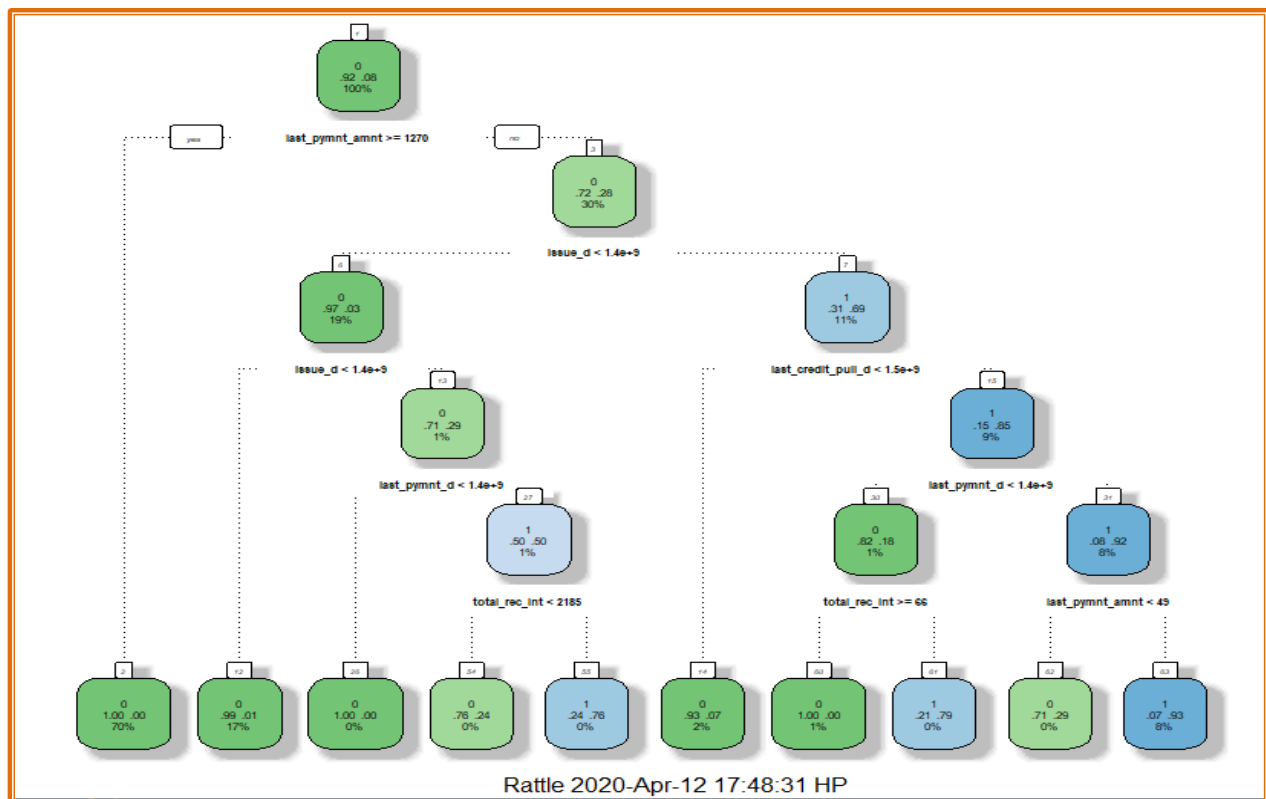
Please refer R-Code for Source Code of Logistic Regression Model for AUC curve, Rank etc.

### 3.2 CART

**Non Zero Variance:** We have verified that there is no variable with zero variance. Hence we will use all the variable for CART.

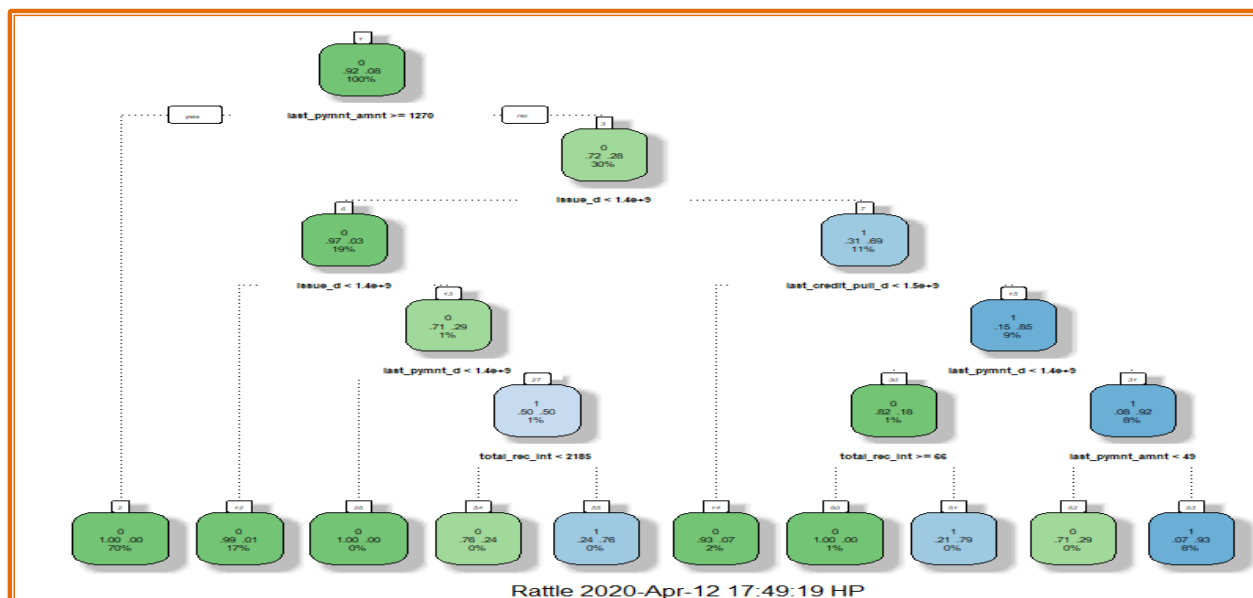
Minsplit: 900, minbucket: 300, xval: 10

The output of the CART Model is:



### 3.2.1 Post Pruning

To prune the tree, we find the best 'Complexity Parameter' of the tree. We prune the tree at  $cp=0$  to avoid overfitting.



### 3.2.2 Performance Metrics

Train Dataset:

cnt_non_resp	rate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks	cum_cnt	cum_resp_rate	lift
5072	71.53	12745	5072	0.96	0.03	0.93	17817	0.72	8.57
27394	1.37	13125	32466	0.98	0.22	0.76	45591	0.29	3.45
111230	0.20	13343	143696	1.00	0.99	0.01	157039	0.08	0.95
1700	0.00	13343	145396	1.00	1.00	0.00	158739	0.08	0.95

#### Confusion Matrix and Statistics

```

Reference
Prediction 0 1
0 144358 1083
1 1038 12260

Accuracy : 0.9866
95% CI : (0.9861, 0.9872)
No Information Rate : 0.9159
P-Value [Acc > NIR] : <2e-16

Kappa : 0.9131
McNemar's Test P-Value : 0.3394

Sensitivity : 0.91883
Specificity : 0.99286
Pos Pred Value : 0.92194
Neg Pred Value : 0.99255
Prevalence : 0.08406
Detection Rate : 0.07723
Detection Prevalence : 0.08377
Balanced Accuracy : 0.95585

'Positive' Class : 1

```

Metrics	Value
Accuracy	0.986
Sensitivity	0.918
Specificity	0.992
KS	0.923
AUC	0.982
Gini Coefficient	0.884

Test Dataset:

cnt_non_resp	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks	cum_cnt	cum_resp_rate	lift
2183	71.45	5462	2183	0.96	0.04	0.92	7645	0.71	8.45
11676	1.34	5621	13859	0.98	0.22	0.76	19480	0.29	3.45
47736	0.20	5719	61595	1.00	0.99	0.01	67314	0.08	0.95
717	0.00	5719	62312	1.00	1.00	0.00	68031	0.08	0.95

```
Confusion Matrix and Statistics
Reference
Prediction 0 1
0 61870 431
1 442 5288

Accuracy : 0.9872
95% CI : (0.9863, 0.988)
No Information Rate : 0.9159
P-Value [Acc > NIR] : <2e-16

Kappa : 0.9167

Mcnemar's Test P-Value : 0.735

Sensitivity : 0.92464
Specificity : 0.99291
Pos Pred Value : 0.92286
Neg Pred Value : 0.99308
Prevalence : 0.08406
Detection Rate : 0.07773
Detection Prevalence : 0.08423
Balanced Accuracy : 0.95877

'Positive' Class : 1
```

Metrics	Value
Accuracy	0.987
Sensitivity	0.924
Specificity	0.992
KS	0.928
AUC	0.982
Gini Coefficient	0.902

### 3.2.3 Interpretation

The main variable to split the node are last\_pmnt\_amt, issue\_d, last\_credit\_pull\_d.

The specificity is high which means there are few false positive.

The model is stable as evident from the output of confusion matrix for training and testing dataset.

Based on the test metrics we can interpret that.

1. The model will catch 92% of the customers who will default .
2. The model will catch 99% of the customers who will not default
3. Overall all accuracy is 98%
4. Out of the customers who are predicted as will default, 92% of them will actually default
5. Out of the customers who are predicted as will Not default, 99% of them will actually not default

Please refer R-Code for Source Code of CART for AUC curve etc

### 3.3 Naïve Bayes

Naïve Bayes is a Supervised Machine Learning algorithm that classifies a new data point into the target class using Baye's theorem and assuming all the predictors are independent to each other.

```
> ModelNB

Naive Bayes Classifier for Discrete Predictors

Call:
naiveBayes.default(x = x, y = y, laplace = laplace)

A-priori probabilities:
Y
      0      1
0.91594378 0.08405622
```

#### 3.3.1 Performance Metrics

Train Dataset:

cnt_non_resp	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks	cum_cnt	cum_resp_rate	lift
5541	65.09	10333	5541	0.77	0.04	0.73	15874	0.65	7.73
13352	15.89	12855	18893	0.96	0.13	0.83	31748	0.40	4.76
15770	0.66	12959	34663	0.97	0.24	0.73	47622	0.27	3.21
15827	0.30	13006	50490	0.97	0.35	0.62	63496	0.20	2.38
15597	1.74	13283	66087	1.00	0.45	0.55	79370	0.17	2.02
15854	0.12	13302	81941	1.00	0.56	0.44	95243	0.14	1.67
15847	0.17	13329	97788	1.00	0.67	0.33	111117	0.12	1.43
15869	0.03	13334	113657	1.00	0.78	0.22	126991	0.10	1.19
15867	0.04	13341	129524	1.00	0.89	0.11	142865	0.09	1.07
15872	0.01	13343	145396	1.00	1.00	0.00	158739	0.08	0.95

```
Confusion Matrix and Statistics

      Reference
Prediction  0      1
0  139013    2012
1   6383   11331

      Accuracy : 0.9471
      95% CI   : (0.946, 0.9482)
No Information Rate : 0.9159
P-Value [Acc > NIR] : < 2.2e-16

      Kappa : 0.701

McNemar's Test P-Value : < 2.2e-16

      Sensitivity : 0.84921
      Specificity : 0.95610
Pos Pred Value : 0.63966
Neg Pred Value : 0.98573
Prevalence : 0.08406
Detection Rate : 0.07138
Detection Prevalence : 0.11159
Balanced Accuracy : 0.90265

      'Positive' Class : 1
```

Metrics	Value
Accuracy	0.947
Sensitivity	0.849
Specificity	0.956
KS	0.877
AUC	0.960
Gini Coefficient	0.882

Test Dataset:

cnt_non_resp	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks	cum_cnt	cum_resp_rate	lift
2358	65.34	4446	2358	0.78	0.04	0.74	6804	0.65	7.73
5722	15.89	5527	8080	0.97	0.13	0.84	13607	0.41	4.88
6745	0.84	5584	14825	0.98	0.24	0.74	20409	0.27	3.21
6783	0.31	5605	21608	0.98	0.35	0.63	27213	0.21	2.50
6709	1.38	5699	28317	1.00	0.45	0.55	34016	0.17	2.02
6797	0.09	5705	35114	1.00	0.56	0.44	40819	0.14	1.67
6794	0.12	5713	41908	1.00	0.67	0.33	47621	0.12	1.43
6804	0.00	5713	48712	1.00	0.78	0.22	54425	0.10	1.19
6800	0.04	5716	55512	1.00	0.89	0.11	61228	0.09	1.07
6800	0.04	5719	62312	1.00	1.00	0.00	68031	0.08	0.95

Confusion Matrix and Statistics		
Prediction	Reference	
	0	1
0	59611	886
1	2701	4833
Accuracy : 0.9473		
95% CI : (0.9456, 0.9489)		
No Information Rate : 0.9159		
P-Value [Acc > NIR] : < 2.2e-16		
Kappa : 0.7007		
McNemar's Test P-Value : < 2.2e-16		
Sensitivity : 0.84508		
Specificity : 0.95665		
Pos Pred Value : 0.64149		
Neg Pred Value : 0.98535		
Prevalence : 0.08406		
Detection Rate : 0.07104		
Detection Prevalence : 0.11074		
Balanced Accuracy : 0.90087		
'Positive' Class : 1		

Metrics	Value
Accuracy	0.947
Sensitivity	0.845
Specificity	0.956
KS	0.880
AUC	0.961
Gini Coefficient	0.883

### 3.3.2 Interpretation

The model is stable as evident from the output of confusion matrix for training and testing dataset. Based on the test metrics we can interpret that.

1. The model will catch 84% of the customers who will default .
2. The model will catch 95% of the customers who will not default
3. Overall all accuracy is 94%
4. Out of the customers who are predicted as will default, 63% of them will actually default
5. Out of the customers who are predicted as will Not default, 98% of them will actually not default

Please refer R-Code for Source Code of Naïve Bayes Model for AUC curve, Rank etc

### 3.4 Model Comparison

Performance Measure	Logistic Regression	CART	Naïve Bayes
	Test Dataset	Test Dataset	Test Dataset
Confusion Matrix : Accuracy	0.978	0.987	0.947
Confusion Matrix : Sensitivity	0.915	0.924	0.845
Confusion Matrix : Specificity	0.984	0.992	0.956
KS	0.939	0.928	0.88
AUC	0.992	0.982	0.961
Gini Coefficient	0.903	0.902	0.883
Misclassification Rate	=1440/68031=0.021	=873/68031=0.013	=3587/68031=0.052

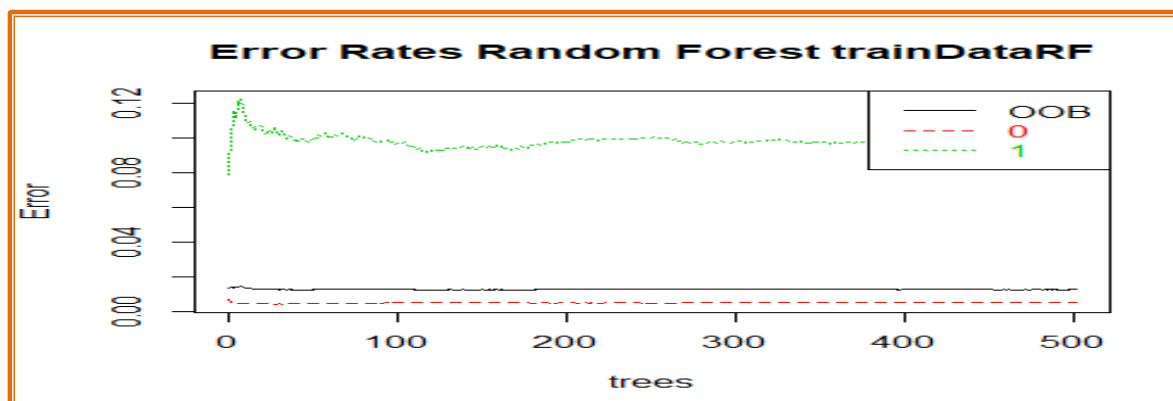
- For Naïve Bayes, the base assumption is that the predictor variables are independent and equally important. For our data, we have seen that the predictors are correlated. Hence, we can conclude that Naïve Bayes is not giving correct prediction.
- Out of Logistic regression, CART and Naïve Bayes, **CARTModel** has the highest Accuracy and sensitivity. Hence, we conclude that CART model is the best among the three.
- For CART, the misclassification rate is also low compared to the other two.

## 4. Model Tuning

### 4.1 Random Forest

We try to build the forest with 4 variables as candidate at each split and 1000 as the minimum size of terminal node.

We analyze the Out of Bag error to find ntree. In our case it is around 190.



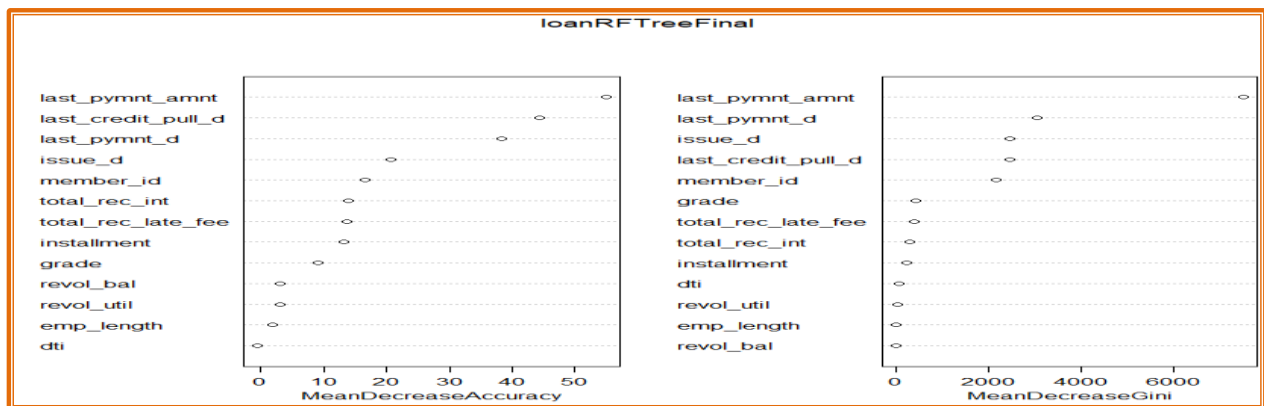
### 4.1.1 tuneRF

We use tuneRF function to get mtry value and build the tuned random forest. As per the below result mtry=3 has minimum out of bag error.

```
mtry = 3 OOB error = 1.08%
Searching left ...
mtry = 2 OOB error = 1.15%
-0.0622093 0.01
Searching right ...
mtry = 4 OOB error = 1.12%
-0.03546512 0.01
```

### 4.1.2 Important Variable

Based on the output of **Mean Decrease Gini** we can say the top 4 variables to predict if customer will default the loan or not are **last\_pymnt\_amt**, **issue\_d**, **last\_pymnt\_d** and **last\_credit\_pull\_d**



### 4.1.3 Performance Metrics

Train Dataset

	cnt_non_resp	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks	cum_cnt	cum_resp_rate	lift
63	2821	82.23	13053	2821	0.98	0.02	0.96	15874	0.82	9.76
63	15750	1.58	13306	18571	1.00	0.13	0.87	31877	0.42	5.00
87	126825	0.03	13343	145396	1.00	1.00	0.00	158739	0.08	0.95

```
Confusion Matrix and Statistics
      Reference
Prediction 0      1
0 144706 1030
1    690 12313

Accuracy : 0.9892
95% CI : (0.9886, 0.9897)
No Information Rate : 0.9159
P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.9288
McNemar's Test P-Value : 2.983e-16

Sensitivity : 0.92281
Specificity : 0.99525
Pos Pred Value : 0.94694
Neg Pred Value : 0.99293
Prevalence : 0.08406
Detection Rate : 0.07757
Detection Prevalence : 0.08191
Balanced Accuracy : 0.95903

'Positive' Class : 1
```

Metrics	Value
Accuracy	0.982
Sensitivity	0.922
Specificity	0.995
KS	0.959
AUC	0.996
Gini Coefficient	0.907

## Test Dataset:

cnt_non_resp	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks	cum_cnt	cum_resp_rate	lift
1257	81.71	5615	1257	0.98	0.02	0.96	6872	0.82	9.75
7027	1.38	5713	8284	1.00	0.13	0.87	13997	0.41	4.88
54028	0.01	5719	62312	1.00	1.00	0.00	68031	0.08	0.95

### Confusion Matrix and Statistics

```
Reference
Prediction 0 1
0 62030 421
1 282 5298

Accuracy : 0.9897
95% CI : (0.9889, 0.9904)
No Information Rate : 0.9159
P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.9321

McNemar's Test P-Value : 1.942e-07

Sensitivity : 0.92639
Specificity : 0.99547
Pos Pred Value : 0.94946
Neg Pred Value : 0.99326
Prevalence : 0.08406
Detection Rate : 0.07788
Detection Prevalence : 0.08202
Balanced Accuracy : 0.96093

'Positive' Class : 1
```

Metrics	Value
Accuracy	0.987
Sensitivity	0.926
Specificity	0.995
KS	0.961
AUC	0.997
Gini Coefficient	0.906

## 4.1.4 Interpretation

The model is stable as evident from the output of confusion matrix for training and testing dataset. Based on the test metrics we can interpret that –

1. The model will catch 92% of the customers who will default.
2. The model will catch 99% of the customers who will not default in loan payment
3. Overall all accuracy is 98%
4. Out of the customers who are predicted as will default, 95% of them will actually default
5. Out of the customers who are predicted as will Not default, 99% of them will actually not default

Please refer R-Code for Source Code Random Forest for AUC curve, Rank etc

## 4.2 Bagging

Bagging is also called as Bootstrap Aggregating. It is an ensemble machine learning algorithm designed to improve accuracy and stability of algorithm used in statistical classification and regression by reducing variance and avoiding overfitting.

```
> ModelBagging
Bagging classification trees with 25 bootstrap replications
Call: bagging.data.frame(formula = loan_status ~ ., data = trainDataBag,
  control = rpart.control(maxdepth = 5, minsplit = 3))
```

## 4.2.1 Performance Metrics

Train Dataset:

cnt_non_resp	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks	cum_cnt	cum_resp_rate	lift
145396	8.41	13343	145396	1	1	0	158739	0.08	0.95

```
Confusion Matrix and Statistics
      Reference
Prediction 0      1
0 144487 1265
1   909 12078

      Accuracy : 0.9863
      95% CI : (0.9857, 0.9869)
    No Information Rate : 0.9159
    P-Value [Acc > NIR] : < 2.2e-16

      Kappa : 0.91

  Mcnemar's Test P-Value : 2.663e-14

      Sensitivity : 0.90519
      Specificity : 0.99375
    Pos Pred Value : 0.93001
    Neg Pred Value : 0.99132
      Prevalence : 0.08406
    Detection Rate : 0.07609
    Detection Prevalence : 0.08181
    Balanced Accuracy : 0.94947

    'Positive' Class : 1
```

Metrics	Value
Accuracy	0.986
Sensitivity	0.905
Specificity	0.993
KS	0.928
AUC	0.965
Gini Coefficient	0.917

Test Dataset:

cnt_non_resp	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks	cum_cnt	cum_resp_rate	lift
62312	8.41	5719	62312	1	1	0	68031	0.08	0.95

```
Confusion Matrix and Statistics
      Reference
Prediction 0      1
0 61920 572
1   392 5147

      Accuracy : 0.9858
      95% CI : (0.9849, 0.9867)
    No Information Rate : 0.9159
    P-Value [Acc > NIR] : < 2.2e-16

      Kappa : 0.9067

  Mcnemar's Test P-Value : 8.156e-09

      Sensitivity : 0.89998
      Specificity : 0.99371
    Pos Pred Value : 0.92923
    Neg Pred Value : 0.99085
      Prevalence : 0.08406
    Detection Rate : 0.07566
    Detection Prevalence : 0.08142
    Balanced Accuracy : 0.94685

    'Positive' Class : 1
```

Metrics	Value
Accuracy	0.985
Sensitivity	0.899
Specificity	0.993
KS	0.928
AUC	0.965
Gini Coefficient	0.918

## 4.2.2 Interpretation

The model is stable as evident from the output of confusion matrix for training and testing dataset. Based on the test metrics we can interpret that

1. The model will catch 90% of the customers who will default.
2. The model will catch 99% of the customers who will not default in loan payment
3. Overall all accuracy is 98%
4. Out of the customers who are predicted as will default, 92% of them will actually default
5. Out of the customers who are predicted as will Not default, 99% of them will actually not default

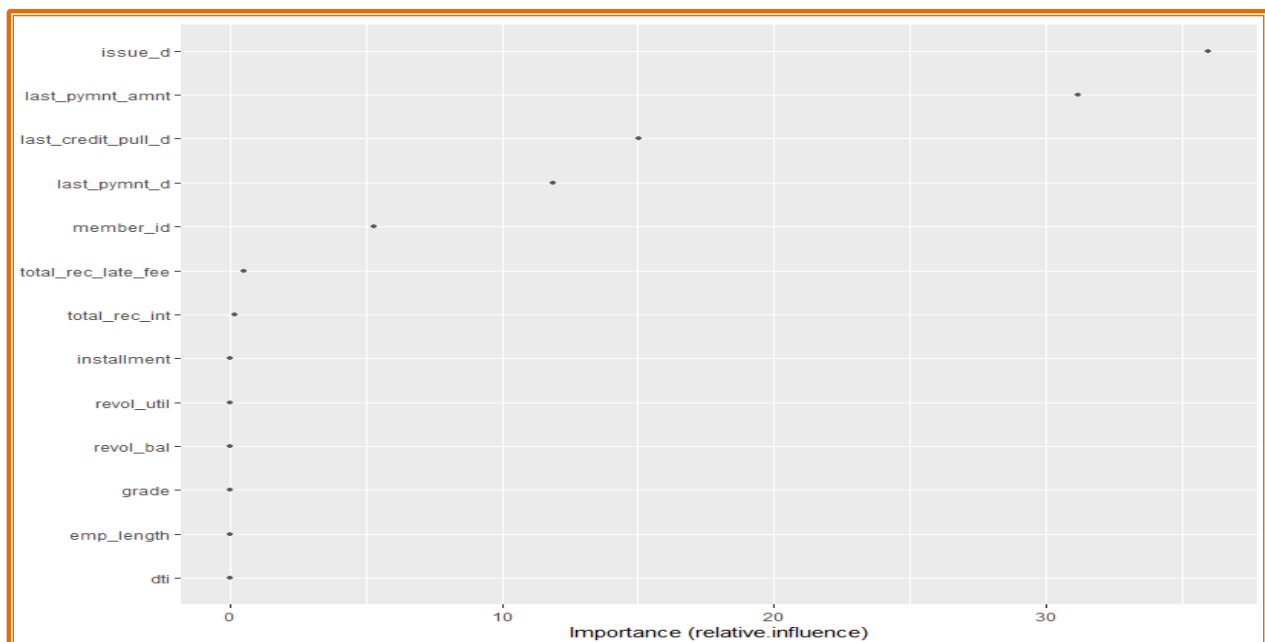
Please refer R-Code for Source Code of Bagging for AUC curve, Rank etc

## 4.3 Boosting

Boosting is another ensemble algorithm which is used to reduce bias and also variance, in supervised learning. In ensemble algorithm, set of weak learners are combined to form strong learner.

```
> gbm.fit
gbm(formula = loan_status ~ ., distribution = "multinomial",
     data = trainDataBoost, n.trees = 200, interaction.depth = 3,
     shrinkage = 0.01, cv.folds = 10, verbose = FALSE, n.cores = NULL)
A gradient boosted model with multinomial loss function.
200 iterations were performed.
The best cross-validation iteration was 200.
There were 13 predictors of which 8 had non-zero influence.
```

### 4.3.1 Important Variable



### 4.3.2 Performance Metrics

Train dataset:

Confusion Matrix and Statistics		
Prediction	Reference	
	0	1
0	144305	1010
1	1091	12333
Accuracy : 0.9868		
95% CI : (0.9862, 0.9873)		
No Information Rate : 0.9159		
P-Value [Acc > NIR] : < 2e-16		
Kappa : 0.9143		
McNemar's Test P-Value : 0.08093		
Sensitivity : 0.92430		
Specificity : 0.99250		
Pos Pred Value : 0.91873		
Neg Pred Value : 0.99305		
Prevalence : 0.08406		
Detection Rate : 0.07769		
Detection Prevalence : 0.08457		
Balanced Accuracy : 0.95840		
'Positive' Class : 1		

Metrics	Value
Accuracy	0.986
Sensitivity	0.924
Specificity	0.992

Test Dataset:

Confusion Matrix and Statistics		
Prediction	Reference	
	0	1
0	61852	433
1	460	5286
Accuracy : 0.9869		
95% CI : (0.986, 0.9877)		
No Information Rate : 0.9159		
P-Value [Acc > NIR] : <2e-16		
Kappa : 0.9149		
McNemar's Test P-Value : 0.3843		
Sensitivity : 0.92429		
Specificity : 0.99262		
Pos Pred Value : 0.91994		
Neg Pred Value : 0.99305		
Prevalence : 0.08406		
Detection Rate : 0.07770		
Detection Prevalence : 0.08446		
Balanced Accuracy : 0.95845		
'Positive' Class : 1		

Metrics	Value
Accuracy	0.986
Sensitivity	0.924
Specificity	0.992

### 4.3.3 Interpretation

The model is stable as evident from the output of confusion matrix for training and testing dataset. Based on the test metrics we can interpret that -

1. The model will catch 92% of the customers who will default.
2. The model will catch 99% of the customers who will not default in loan payment
3. Overall all accuracy is 98%
4. Out of the customers who are predicted as will default, 92% of them will actually default
5. Out of the customers who are predicted as will Not default, 99% of them will actually not default

Please refer R-Code for Source Code of Boosting for AUC curve, Rank etc

## 5. Model Comparison

Performance Measure	Logistic Regression			CART			Naïve Bayes			Random Forest			Bagging			Boosting		
	Train Dataset	Test Dataset	Dev	Train Dataset	Test Dataset	Dev	Train Dataset	Test Dataset	Dev	Train Dataset	Test Dataset	Dev	Train Dataset	Test Dataset	Dev	Train Dataset	Test Dataset	Dev
Confusion Matrix : Accuracy	0.978	0.978	0	0.986	0.987	-0.001	0.947	0.947	0	0.982	0.987	-0.005	0.986	0.985	0.001	0.986	0.986	0
Confusion Matrix : Sensitivity	0.923	0.915	0.008	0.918	0.924	-0.006	0.849	0.845	0.004	0.922	0.926	-0.004	0.905	0.899	0.006	0.924	0.924	0
Confusion Matrix : Specificity	0.983	0.984	-0.001	0.992	0.992	0	0.956	0.956	0	0.995	0.995	0	0.993	0.993	0	0.992	0.992	0
KS	0.94	0.939	0.001	0.923	0.928	-0.005	0.877	0.88	-0.003	0.959	0.961	-0.002	0.928	0.928	0			0
AUC	0.992	0.992	0	0.982	0.982	0	0.96	0.961	-0.001	0.996	0.997	-0.001	0.965	0.965	0			0
Gini Coefficient	0.902	0.903	-0.001	0.884	0.902	-0.018	0.882	0.883	-0.001	0.907	0.906	0.001	0.917	0.918	-0.001			0
Misclassification Rate	0.021	0.021	0	0.013	0.013	0	0.053	0.052	0.001	0.011	0.01	0.001	0.014	0.014	0	0.013	0.013	0
Total			0.007			-0.03			0			-0.01			0.006			0

- KS value for most of the models are more than 90% , hence they will Perform well to separate the default and fully paid cases
- AUC is more than 95% for the all models, hence we can consider them as good classifier
- Random Forest have the highest accuracy , sensitivity and specificity.
- Misclassification Rate for Random Forest is lowest.
- Out of Logistic regression, CART and Naïve Bayes, **CART Model** has the highest Accuracy and sensitivity. Hence, we conclude that CART model is the best among the three.
- Comparing CART with Bagging and Boosting, we can conclude that the model developed using **CART is the best**. The performance of boosting is exactly same on test and train dataset

## 6. Conclusion

We have built various models to understand the factors which influence loan being defaulted.

As per model comparison, CART is the best as accuracy is about 98% and sensitivity is also 92%

The model built using ensemble technique (Random Forest, Bagging and Boosting) is also good model as accuracy is about 98% and there is balance between sensitivity and specificity.

Boosting is the most stable model

Issue\_d, last\_pymnt\_amt, last\_credit\_pull\_d play important role to predict if customer with default or not