DATA SCIENCE MIDTERM PROJECT
TEAM 6

Abstract

Data Science deals with cleaning, preparation, analysis and visualization of large data. Fetching meaningful information from data

Guidance: Professor Srikanth

Rashmi Yadav
Nikita Khamkar
Prateek Mane

West Roxbury

Aim: Question 1

What data mining technique(s) would be appropriate to predict the value of a home in west roxbury? Justify your analysis and model.

Data set Description:

File – West Roxbury.xlsx

Features – 14

Records – 5802

Response Variable – Total (Total is the value of a home)

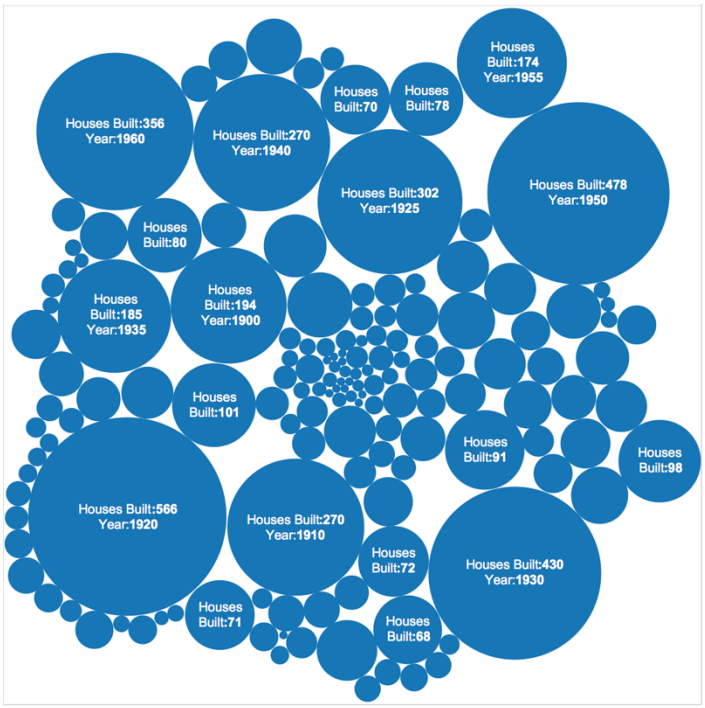
Column name and description –

TOTAL VALUE	Total assessed value for property, in thousands of USD
TAX	Tax bill amount based on total assessed value * tax rate
LOT SQFT	Total lot size of parcel in square feet
YR BUILT	Year property was built
GROSS AREA	Gross floor area
LIVING AREA	Total living area for residential properties (ft2)
FLOORS	Number of floors
ROOMS	Total number of rooms
BEDROOMS	Total number of bedrooms
FULL BATH	Total number of full baths
HALF BATH	Total number of half baths
KITCHEN	Total number of kitchens
FIREPLACE	Total number of fireplaces
REMODEL	When house was remodeled (Recent/Old/None)

Analysis on Dataset:

The West Roxbury dataset was analyzed in Tableau.

1) Homes constructed in each year



Each circle represents a year and the count of home built in that year. The size of circle depends on the count of homes built.

2) Year-wise impact on home cost based on area

Yr Built	Lot Sqft	Gross Area	Living Area	
1798	8861	5356	2953	438
1800	6119	2244	1154	294
1804	9030	3541	2060	474
1810	5000	2852	1834	383
		3116	1990	417
1820	8334	3112	1813	402
	9072	1742	1298	348
	16952	4659	2972	639
1835	3270	2483	1295	350
	14261	2539	1118	348
1840	8860	3133	1651	430
	12201	6196	3061	794
1845	2126	1719	958	277
	12679	4236	2332	435
1848	15790	5260	2890	807
1850	5322	2320	1188	280
	6004	3235	1802	318
	33384	2936	1870	477
1851	26825	8037	3527	705
1860	3906	3830	2280	398
	6189	2871	1643	348
	7785	2038	1056	307
	7796	3085	1808	360
	9960	3767	2418	439
	11918	5501	3125	678
	13637	4985	3026	582

This chart displays how many homes were built in each year, what was the cost of the House and the living area of the home and total area of the plot.

The value of the home is displayed in green label. The intensity of green color

becomes Dark with the value of home.

Its observed that the Living Area is directly proportional to the value of home.

3) Cost of home with respect to the rooms in the home

				Bedrooms								
Floors	Kitchen	Full Bath	Half Bath	1	2	3	4	5	6	7	8	9
1	1	1	0	270.9	297.7	308.0	309.9					
			1	307.3	326.3	341.8	350.6	350.4				
			2		437.6	432.6	494.5					
		2	0	280.3	353.9	353.0	363.3	344.7				
			1		405.3	417.7	472.0					
		3	0			501.3	458.4	383.8				
			1			511.7	547.2					
	2	1	0			331.1						
			1		328.2	320.0	339.9					
		2	0		398.2	365.4	330.1	317.6				
			1			465.5	480.6	464.7				935.1
			2						434.6			
		3	0			448.8	343.4	488.4				
		4	0						500.0			
1.5	1	1	0	284.6	304.2	313.7	293.2					
			1		332.2	349.8	373.9	528.9				
			2		388.9	337.5	411.4	503.1				
		2	0		345.0	345.2	396.1	350.8				
			1		363.1	413.7	418.8	445.1				
		3	0			392.5	341.5	509.4				
			1					634.2				
	2	1	1			336.6						
		2	0					396.5				
			1				409.5	534.0				
		3	0			330.1	471.6		505.6			
			1				498.7					

The value of the house is displayed in green label. The intensity of green color becomes Dark with the value of home.

For eg – 935.1K is the value of the home when the home has 1 floor, 2 kitchens, 2 full Baths, 1 half bath and 9 other rooms.

It is observed that the value of home is directly proportional to the number of rooms in the house. If two house have same number of rooms, then the house with more number of half baths has a higher cost.

4) Tax with respect to the area of home

Lot Sqft	Gross Area	Living Area	
997	1319	504	1,320
1017	1293	797	1,819
1037	2520	1680	2,225
1218	1965	971	2,575
1237	3039	1908	3,718
1358	2576	1598	3,072
1616	1439	856	2,298
1696	2496	1518	2,684
1800	3663	1904	3,789
1821	2452	1162	2,292
1830	3384	1908	4,625
1885	1825	1093	3,045
1980	1502	690	2,108
1999	1968	1296	3,455
2088	2192	1400	3,818
2100	1800	972	3,297
	2244	1400	3,868
	2414	1544	4,063
	2552	1326	3,609
2126	1719	958	3,479
2174	2952	1382	2,562
2275	2760	2138	3,906
	2821	1411	3,372

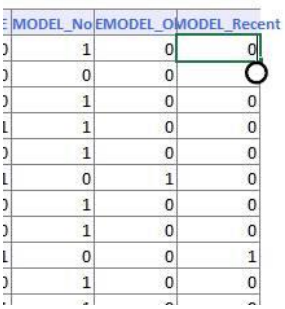
The tax of the house is displayed in green label. The intensity of green color becomes Dark with the area of home.

Data Transformation

The remodel column has three categorical values

Values – None, Recent, Old

This column is transformed into a binary column by creating categories.



MODEL_No	EMODEL_O	MODEL_Recent
1	0	0
0	0	
1	0	0
1	0	0
1	0	0
0	1	0
1	0	0
1	0	0
0	0	1
1	0	0

Correlation Matrix:

	TOTAL VALUE	TAX	LOT SQFT	YR BUILT	GROSS AREA	LIVING AREA	FLOORS	ROOMS	BEDROOMS	FULL BATH	HALF BATH	KITCHEN
TOTAL VALUE	1.000000	1.000000	0.546123	-0.100917	0.800519	0.837120	0.481523	0.638539	0.561871	0.432807	0.348167	0.018265

The correlation matrix infers, YR built is weakly correlated with the total value of Home and TAX is highly correlated with the total value. Hence TAX was dropped from The dataset.

Applied Models:

The data was partitioned in 80(training):20(test)

After the data partition, three models were generated: -

1) CART
2) Random Forest
3) Multi Linear Regression

Model Evaluation Criteria

1) RMS error
2) Average Error

1) CART

Calculate RMS Error

Training Data scoring - Summary Report

Total sum of squared errors	RMS Error	Average Error
1259862	19.02433	0.180033

Validation Data scoring - Summary Report

Total sum of squared errors	RMS Error	Average Error
1113961	21.90774	0.493669

From the table above,

Mean of Total Values column = 400

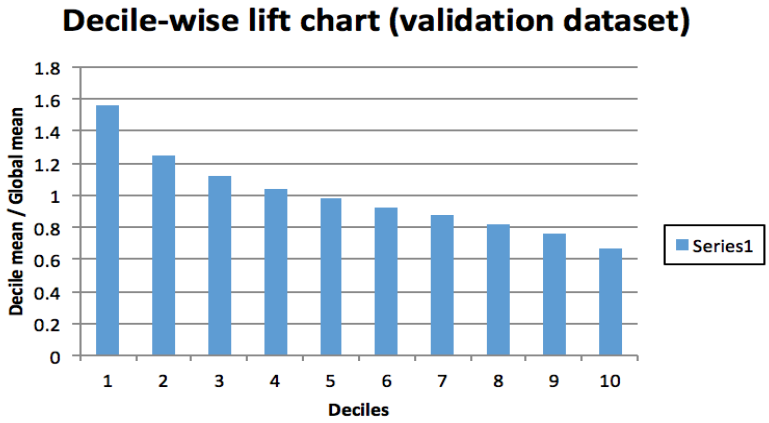
Standard deviation = 370, 430

RMS error = 21.91

Which lies within the standard deviation range i.e 400 + 21.91 = 421.91

Hence, model built by applying Multi Linear Regression Algorithm over the 80% training and 20% validation data is a valid model.

Decile Chart Analysis



The above decile chart forms a stepwise pattern and hence CART model satisfies the decile chart Evaluation criteria.

The decile-wise lift curve is drawn as the decile number versus the cumulative actual output variable value divided by the decile's mean output variable value. This bars in this chart indicate the factor by which the MLR model outperforms a random assignment, one decile at a time.

2) Random Forest

Calculate RMS error

Training Data scoring - Summary Report (Using Full-Grown Tree)

Total sum of squared errors	RMS Error	Average Error
48620.24	3.737288	1.75952E-14

Validation Data scoring - Summary Report (Using Full-Grown Tree)

Total sum of squared errors	RMS Error	Average Error
136136.8	7.658615	0.026513222

From the table above,

Mean of Total Values column = 400

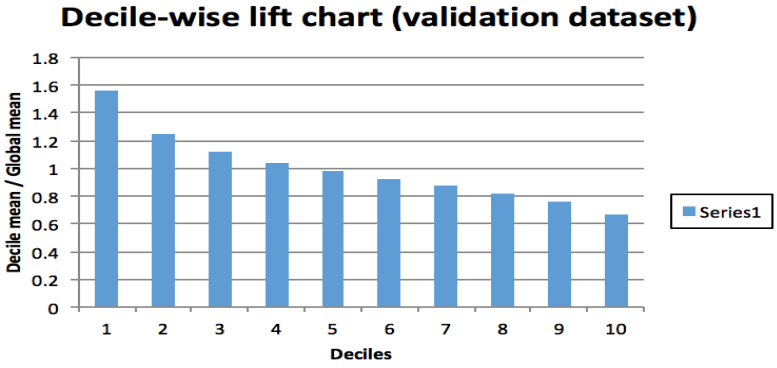
Standard deviation = 370, 430

RMS error = 7.66

Which lies within the standard deviation range i.e 400 + 7.66 = 407.66

Hence, model built by applying Random Forest Algorithm over the 80% training and 20% validation data is a valid model.

Decile Chart Analysis



The above decile chart forms a stepwise pattern and hence random forest model satisfies the decile chart evaluation criteria.

3) Multi linear Regression

Calculate RMS error

Training Data Scoring - Summary Report

Total sum of squared errors	RMS Error	Average Error
1.769605	0.022547	-4.53351E-13

Validation Data Scoring - Summary Report

Total sum of squared errors	RMS Error	Average Error
1.221201	0.022938	-0.000482807

From the table above,

Mean of Total Values column = 400

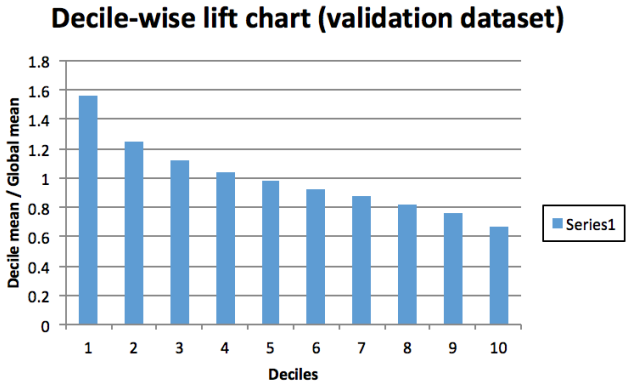
Standard deviation = 370, 430

RMS error = 0.023

Which lies within the standard deviation range i.e 400 + 0.023 = 400.023

Hence, model built by applying Random Forest Algorithm over the 80% training and 20% validation data is a valid model.

Decile Chart Analysis



The above decile chart forms a stepwise pattern and hence Multi Linear Regression model satisfies the decile chart evaluation criteria.

Best Model

Algorithm	RMS Error	Average Error
CART	7.66	0.027
Random Forest	21.91	0.49
Multi Linear Regression	0.023	-0.0005

From the table above, Multi Linear Regression has the lowest RMS Error and Average error. Hence, model built by applying Multi Linear Regression Algorithm over the 80% training and 30% validation data is the best model.

Model Recommendation & Reasons:

- Multi Linear Regression is the best model
- The variation in the model is less(based on RMS and average error) and is used to accurately predict the value of home
- Home value depends on all the features except Tax given in the dataset.

Mortgage Defaulter Prediction

Aim: Question 2

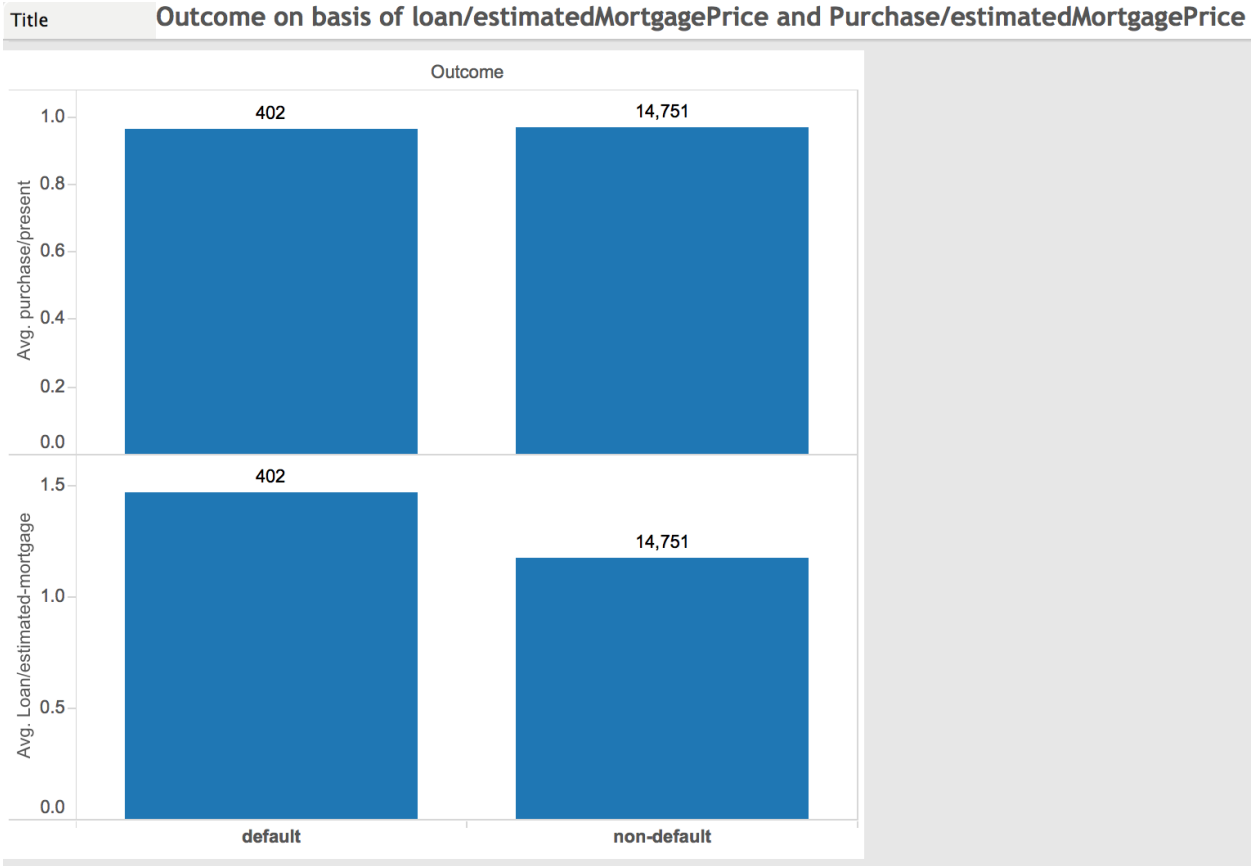
What data mining technique(s) would be appropriate in predicting the defaulter and non-defaulter approved mortgage? Conduct such an analysis.

Data set Description:

The Mortgage Defaulter data set contains data on mortgages that have been approved by bank underwriters. It basically contains columns related to the age of the loan borrower, Total loan amount, Ratio of loan to home purchase price, Borrower's credit score, whether a person is first time home buyer or not, borrowers total monthly debt expense, Borrower's total monthly income, Appraised value of home at origination, Purchase price for house, Tot_monthly_debt_exp/Tot_monthly_income, Current Loan Status that is whether its active, default or paid off, Outcome (either default or non-default), US state in which home is located, Median household income by state 2002-2004, if loan amount is greater than appraisal, Ratio of Loan amt to appraised value of home at origination. The goal is to predict the approved mortgages that may default in future.

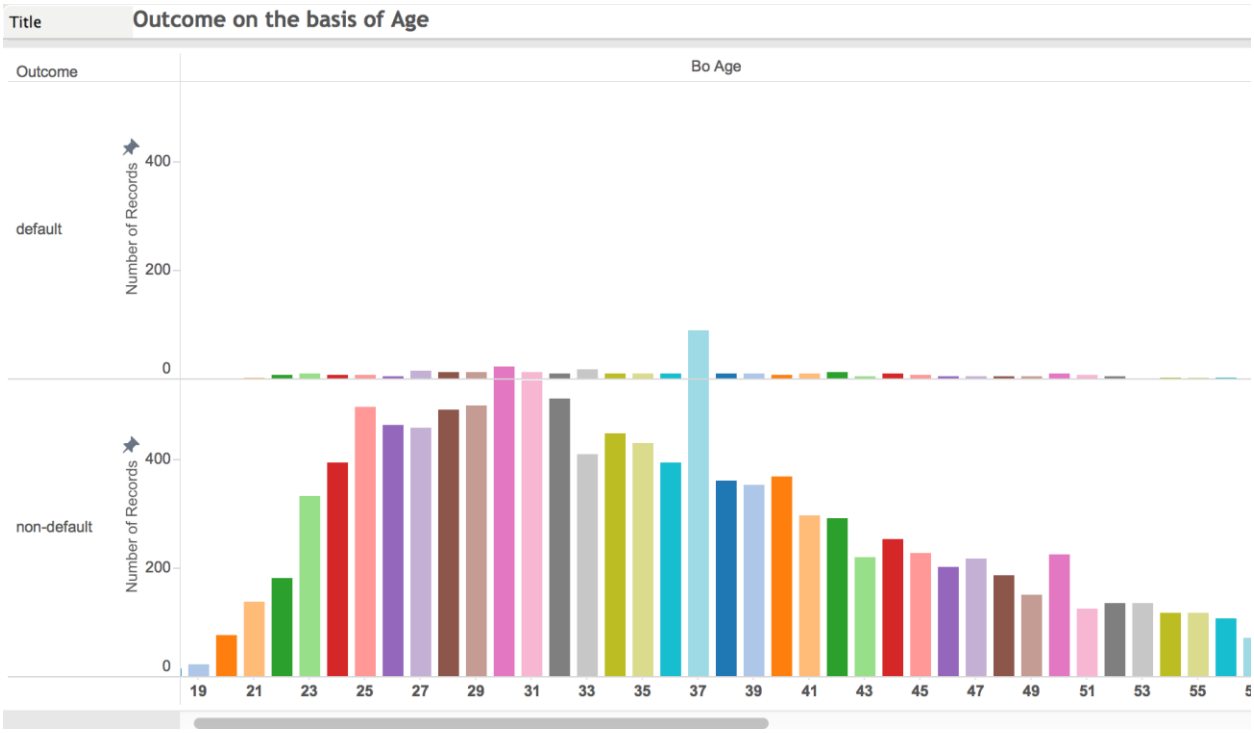
Analysis on Dataset:

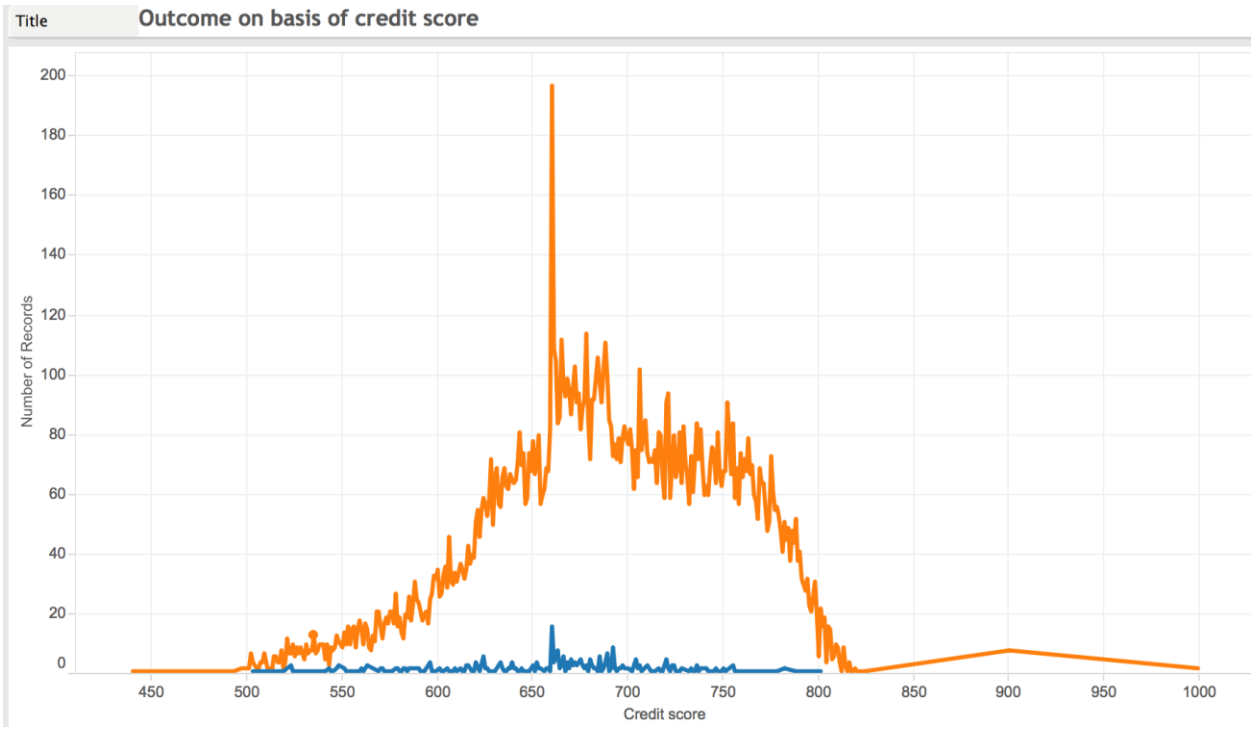
The analysis of the dataset is done in Tableau. So initially we tried finding what attributes are contributing in determining whether a particular approved Mortgage is default or non-default.



In this bar chart we have tried analysis if the outcome is dependent on the average ratio of the loan amount to the estimated present price of the house and average ratio of the purchase price of the house to the estimated present price of the house. What we have observed here is firstly out of the total 15K records 14751 records are non-default and 402 records are default. This implies that the data is biased towards non-default. Herein default and non- default is somewhat dependent upon the ratio of the loan amount to the estimated present price of the house.

We further analyzed the outcome with other attributes like credit score and age of the borrower. Below is the analysis for this.

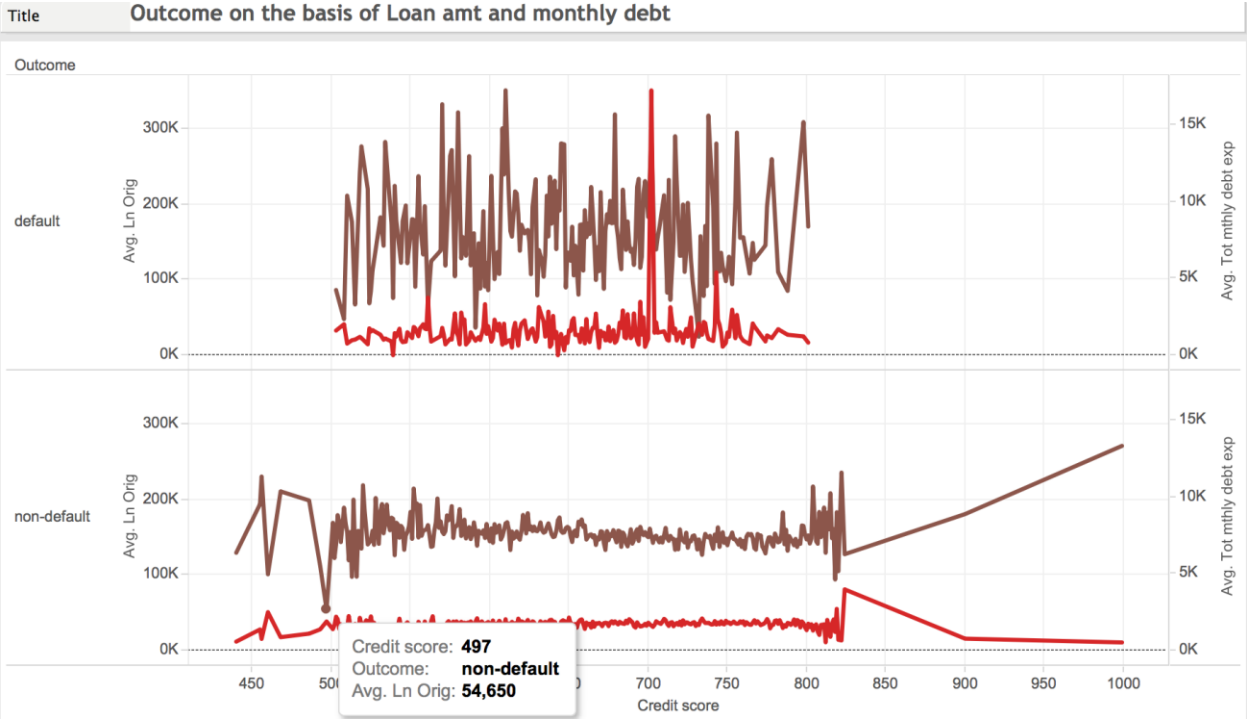




We can observe here that the outcome (default and non-default) is distributed over the range of age. For age 37 the default number is high as compared to other ages but at the same time the non-default number is also high for the same age. So we cant interpret anything useful from

this as age can't be a factor to determine the outcome of the approved mortgage. Going further we plot the graph between outcome and credit score and see that they are highly related and maximum default observed is for 660.

Next we also compared if the average of loan amount borrowed by borrower and their total monthly debt expense are related for determining the outcome. Here the brown line indicate the average loan amount and red lines indicate the debt expense. So we can see that borrower whose debt expense is high are quite probable to default and also borrower who have taken large loan amount and have credit score in the range of 640 to 680 are likely to default.



Now with these observation we will apply different models on these dataset. But before doing that we have calculated the correlation of outcome with all other attribute. This helped us in deciding which attributes are important.

Correlation Matrix:

Input Variable	Correlation
Bo_Age	-0.0002233
Ln_Orig	0.03313622
Orig_LTV_Ratio_Pct	0.0051409
Credit_score	-0.1113339
Tot_mthly_debt_exp	-0.0406081
Tot_mthly_incm	-0.0215746
orig_apprd_val_amt	-0.0186948

pur_prc_amt	-0.0245268
DTI Ratio	-0.0310822
Median_state_inc	-0.0088064
UPB>Appraisal	0.0326673
Loan/Estimated Price	0.03950256
Purchase/present	-0.0208022

Applied Models:

- **Logistic Regression**
- **Random Forest**
- **Cart**

As we could see in the dataset we had a column called as status (active, default, pay off). Here if the status was pay off it means the borrower has paid the loan amount and its non default. So we removed the number of row where the status was pay-off. So we were left with 8k data. The reason why we removed the pay off data is XLminer restricts the the partition upto 10k data and we had 15k data, and anyhow the pay off data were the borrower who had already paid the loan amount.

So now we ran our algorithm in the 8k resultant dataset

Model1 - Logistic Regression:

Regression Model

Input Variables	Coefficient	Std. Error	Chi2-Statistic	P-Value	Odds	CI Lower	CI Upper
Intercept	-4.30816	0.900838832	22.87122908	1.73225E-06	0.013458287	0.002302472	0.078666
Ln_Orig	-2.3E-07	1.43347E-06	0.025802183	0.872384369	0.99999977	0.99999696	1.000003
Credit_score	0.007749	0.000868836	79.5430266	4.71837E-19	1.007778977	1.006064304	1.009497
Tot_mthly_debt_exp	-7.52E-05	0.000138297	0.295878195	0.586478036	0.999924777	0.999653777	1.000196
Tot_mthly_incm	5.84E-05	4.60515E-05	1.608484941	0.204704904	1.000058407	0.999968147	1.000149
pur_prc_amt	-2.08E-06	1.34626E-06	2.388441998	0.122235398	0.999997919	0.999995281	1.000001
DTI Ratio	1.581912	0.70614789	5.018491588	0.025078006	4.864246399	1.218825109	19.41287
UPB>Appraisal	-0.206289	0.19014228	1.177052879	0.277956788	0.813597675	0.560480228	1.181025
Loanamt/presentamt	-0.065564	0.099403884	0.435037505	0.509527417	0.936538928	0.770749222	1.13799
purchaseamt/presentam	1.945993	0.650061963	8.961338813	0.002757527	7.000577783	1.957943428	25.03039

Residual DF	6616
Residual Dev.	2474.122619
# Iterations Used	4
Multiple R²	0.039429409

Variance-Covariance Matrix

	Intercept	Ln_Orig	Credit_score	ot_mthly_debt_ex	Tot_mthly_incm	pur_prc_amt	DTI Ratio	JPB>Appraisa	Loanamt/presentam	rchaseamt/presentamt
Intercept	0.811511	-1.02807E-07	-0.00053864	3.26217E-05	-1.36503E-05	1.28246E-07	-0.211545	0.008367	-6.01587E-05	-0.386659504
Ln_Orig	-1.03E-07	2.05485E-12	6.57693E-11	2.8025E-11	-6.81066E-12	-1.06573E-12	-1.37E-07	-1.3E-07	-9.30714E-08	1.27925E-07
Credit_score	-0.000539	6.57693E-11	7.54876E-07	-6.59506E-09	3.60355E-10	-9.96283E-11	6.93E-05	-3.17E-06	-4.13773E-06	3.89685E-05
Tot_mthly_debt_exp	3.26E-05	2.8025E-11	-6.5951E-09	1.9126E-08	-4.88138E-09	-2.71496E-11	-8.21E-05	8.48E-07	-7.72273E-07	-5.84634E-06
Tot_mthly_incm	-1.37E-05	-6.81066E-12	3.60355E-10	-4.88138E-09	2.12074E-09	-5.00814E-12	2.47E-05	3.97E-07	4.24693E-07	3.45455E-06
pur_prc_amt	1.28E-07	-1.06573E-12	-9.9628E-11	-2.71496E-11	-5.00814E-12	1.81242E-12	7.18E-08	1.15E-07	7.35905E-08	-2.96987E-07
DTI Ratio	-0.211545	-1.36658E-07	6.92539E-05	-8.21045E-05	2.47087E-05	7.18252E-08	0.498645	-0.000684	0.006341632	0.001972693
UPB>Appraisal	0.008367	-1.30031E-07	-3.175E-06	8.4809E-07	3.97112E-07	1.14964E-07	-0.000684	0.036154	0.000650765	-0.027104161
Loanamt/presentamt	-6.02E-05	-9.30714E-08	-4.1377E-06	-7.72273E-07	4.24693E-07	7.35905E-08	0.006342	0.000651	0.009881132	-0.010646861
purchaseamt/presentamt	-0.38666	1.27925E-07	3.89685E-05	-5.84634E-06	3.45455E-06	-2.96987E-07	0.001973	-0.027104	-0.010646861	0.422580556

Here in we can see the coefficient and P-value. here we can see that the P value is greater for credit score and loan amount and subsequently for ratio of LoanAmount/estimatedPriceOfMortgage. Also here we can see the covariance Matrix which speaks about the relation between each attribute with each other.

Training data confusion matrix:

Training Data Scoring - Summary Report

Cutoff probability value for success (UPDATABLE)	0.5

Confusion Matrix

	Predicted Class	
Actual Class	non-default	default
non-default	6304	0
default	322	0

Error Report

Class	# Cases	# Errors	% Error
non-default	6304	0	0
default	322	322	100
Overall	6626	322	4.859643827

Performance

Success Class	non-default
Precision	0.951403562
Recall (Sensitivity)	1
Specificity	0
F1-Score	0.975096674

Herein we can see none of the default approved mortgage is predicted as in the % error for this is 100% but the overall error is 4.85%, its less because one can run this model and predict the non-default approved mortgage correctly.

And as we run this model on test data we get the below matrix. Which gives 100% for the default approved Mortgage.

Validation Data Scoring - Summary Report

Cutoff probability value for success (UPDATABLE)	0.5

Confusion Matrix		
	Predicted Class	
Actual Class	non-default	default
non-default	1577	0
default	80	0

Error Report			
Class	# Cases	# Errors	% Error
non-default	1577	0	0
default	80	80	100
Overall	1657	80	4.828002414

Performance	
Success Class	non-default
Precision	0.951719976
Recall (Sensitivity)	1
Specificity	0
F1-Score	0.975262832

Now we try running random forest and cart model on the same dataset and see if there is any difference.

Model 2 - Random Forest:

Below is the importance of the attribute in creating the random tree for the dataset. As we can see the most important attribute is loan amount and credit score followed by other attributes.

Details of the random-trees ensemble

Variables	Ln_Orig	Credit_score	Loanamt/presentamt	Tot_mthly_incm	Tot_mthly_debt_exp	DTI Ratio	UPB>Appraisal
Importance	0.23404	0.202350659	0.111066556	0.108670426	0.106126563	0.07123	0.01474989

below is the confusion matrix for the training data. AS we can see as the data is biased this model is good to detect non-default approved mortgage but not for to detect the default approved mortgage. The precision is 0.95 and F1 score is 0.97.

Training Data scoring - Summary Report

Cutoff probability value for success (UPDATABLE)	0.5

Confusion Matrix		
	Predicted Class	
Actual Class	non-default	default
non-default	6304	0
default	321	1

Error Report			
Class	# Cases	# Errors	% Error
non-default	6304	0	0
default	322	321	99.68944099
Overall	6626	321	4.844551766

Performance	
Success Class	non-default
Precision	0.951547
Recall (Sensitivity)	1
Specificity	0.003106
F1-Score	0.975172

For validating data as well this model is able to predict all the non-default approved mortgage but unable to predict the default approved mortgage.

Validation Data scoring - Summary Report

Cutoff probability value for success (UPDATABLE)	0.5

Confusion Matrix		
	Predicted Class	
Actual Class	non-default	default
non-default	1577	0
default	80	0

Error Report			
Class	# Cases	# Errors	% Error
non-default	1577	0	0
default	80	80	100
Overall	1657	80	4.828002414

Performance	
Success Class	non-default
Precision	0.95172
Recall (Sensitivity)	1
Specificity	0
F1-Score	0.975263

We will try running the other algorithm to see the pattern that we get there.

Model 3 - Cart :

Below is the prior probability of the outcome and we can see the non-default probability is much higher than default. And there are 16 decision nodes.

Prior class probabilities

Empirical prior probabilities		
Class	**Probability**	
default	0.048596	
non-default	0.951404	<-- Success Class

Training Log (Growing the full tree using training data)

# Decision Nodes	% Error
0	4.859644
1	4.859644
2	4.859644
3	4.844552
4	4.844552
5	4.844552
6	4.82946
7	4.82946
8	4.82946
9	4.82946
10	4.82946
11	4.82946
12	4.82946
13	4.814368
14	4.814368
15	4.799276
16	4.799276

Now if we see the confusion matrix for the training data we get the below result.

Training Data scoring - Summary Report (Using Full-Grown Tree)

Cutoff probability value for success (UPDATABLE)	0.5

Confusion Matrix		
	Predicted Class	
Actual Class	non-default	default
non-default	6302	2
default	316	6

Error Report			
Class	# Cases	# Errors	% Error
non-default	6304	2	0.031726
default	322	316	98.13665
Overall	6626	318	4.799276

Performance	
Success Class	non-default
Precision	0.952251435
Recall (Sensitivity)	0.999682741
Specificity	0.01863354
F1-Score	0.975390806

Here out of 316 default approved mortgage only 6 are being detected as default and out of 6304 non-default approved mortgage 6302 are being detected as non-default. So here as well we can see this model is biased towards non-default approved mortgage.

Training Data scoring - Summary Report (Using Full-Grown Tree)

Cutoff probability value for success (UPDATABLE)	0.5

Confusion Matrix		
	Predicted Class	
Actual Class	non-default	default
non-default	6302	2
default	316	6

Error Report			
Class	# Cases	# Errors	% Error
non-default	6304	2	0.031726
default	322	316	98.13665
Overall	6626	318	4.799276

Performance	
Success Class	non-default
Precision	0.952251435
Recall (Sensitivity)	0.999682741
Specificity	0.01863354
F1-Score	0.975390806

Conclusion: So by running all the model in the dataset we couldn't come up with a correct model as the data is biased and all the model gives an appropriate result for non-default approved mortgage but not for default approved mortgage. So we decided to cut down the data by taking only 1000 non-default data and 402 default data randomly. This makes out dataset unbiased. Now I ran all the three model on the new dataset (1400 records).

Applied Model on new dataset:

Model 1 – Logistic Regression:

With the 1400 records in the new dataset we partitioned the dataset into 80:20 ratio with 80% training data and 20% validation data. We first ran the logistic regression algorithm. Here we see the P value for different attribute which helps us in doing the feature selection. As from here we concluded that we should remove attribute DTI Ratio, Total monthly Income, Total monthly debt expense.

Regression Model

Input Variables	Coefficient	Std. Error	Chi2-Statistic	P-Value	Odds	CI Lower	CI Upper
Intercept	-1.130115	1.558825905	0.525593692	0.468465	0.32299607	0.015217	6.856022812
Ln_Orig	-1.24E-05	2.15754E-06	33.20731297	8.28E-09	0.999987567	0.999983	0.999991796
Credit_score	0.010027	0.001635434	37.59147338	8.72E-10	1.010077594	1.006845	1.013320482
Tot_mthly_debt_exp	-0.000731	0.000336401	4.718107318	0.029847	0.999269563	0.998611	0.999928633
Tot_mthly_incm	0.000127	9.01492E-05	1.987098146	0.158645	1.000127086	0.99995	1.000303814
DTI Ratio	0.411208	1.370255468	0.090057594	0.764104	1.508639347	0.102854	22.12834834
Loanamt/presentamt	1.526852	0.142304247	115.1218913	7.4E-27	4.603659671	3.483162	6.084610817
purchaseamt/presentamt	-5.78358	0.930873399	38.60221702	5.2E-10	0.003077678	0.000496	0.019080248

Residual DF	833
Residual Dev.	661.9383
# Iterations Used	4
Multiple R^2	0.344235

Variance-Covariance Matrix

	Intercept	Ln_Orig	Credit_score	_mthly_debt_	Tot_mthly_incm	DTI Ratio	Loanamt/presentamt	urchaseamt/presentamt
Intercept	2.429938	-4.94006E-07	-0.0019373	0.000164	-4.98809E-05	-0.823948	0.01416987	-0.842579883
Ln_Orig	-4.94E-07	4.65497E-12	6.37602E-11	7.28E-11	-2.53346E-11	-1.96E-07	-2.07515E-07	1.5546E-07
Credit_score	-0.001937	6.37602E-11	2.67464E-06	-6.22E-08	1.25554E-08	0.000344	1.94016E-05	1.92792E-05
Tot_mthly_debt_exp	0.000164	7.28152E-11	-6.2248E-08	1.13E-07	-2.73952E-08	-0.000398	4.19578E-06	-3.61891E-05
Tot_mthly_incm	-4.99E-05	-2.53346E-11	1.25554E-08	-2.74E-08	8.12687E-09	0.000103	3.14673E-07	1.14715E-05
DTI Ratio	-0.823948	-1.95668E-07	0.000344433	-0.000398	0.000103328	1.8776	-0.02339465	0.110628912
Loanamt/presentamt	0.01417	-2.07515E-07	1.94016E-05	4.2E-06	3.14673E-07	-0.023395	0.020250499	-0.029902981
purchaseamt/presentamt	-0.84258	1.5546E-07	1.92792E-05	-3.62E-05	1.14715E-05	0.110629	-0.029902981	0.866525286

Training data confusion matrix:

Confusion Matrix		
	Predicted Class	
Actual Class	non-default	default
non-default	563	36
default	87	155

Error Report			
Class	# Cases	# Errors	% Error
non-default	599	36	6.010016694
default	242	87	35.95041322
Overall	841	123	14.6254459

Performance	
Success Class	non-default
Precision	0.866153846
Recall (Sensitivity)	0.939899833
Specificity	0.640495868
F1-Score	0.901521217

I the above mentioned confusion matrix as we reduced the data to make it unbiased we observe that now out of 242 default approved mortgage 155 are predicted correctly and 87 are predicted wrong and out of 599 non-default approved mortgage 563 non-default are predicted correct. Here the accuracy of the model is 0.90 and precision is 0.86

Validation data confusion matrix

Here with the validation data we are getting 60 error for default approved mortgage and 21 error for non-default approved mortgage. We are getting an accuracy of 0.90 and precision of 0.86.

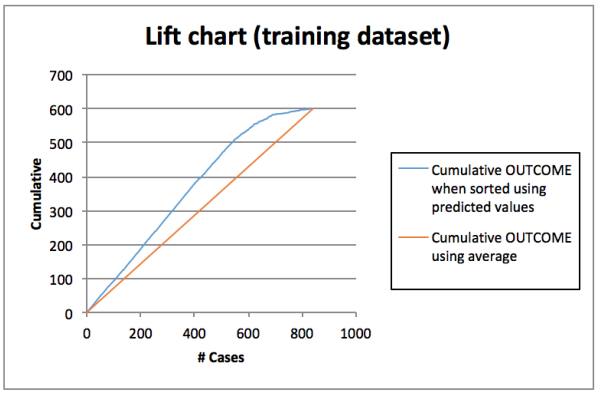
Confusion Matrix		
	Predicted Class	
Actual Class	non-default	default
non-default	379	21
default	60	100

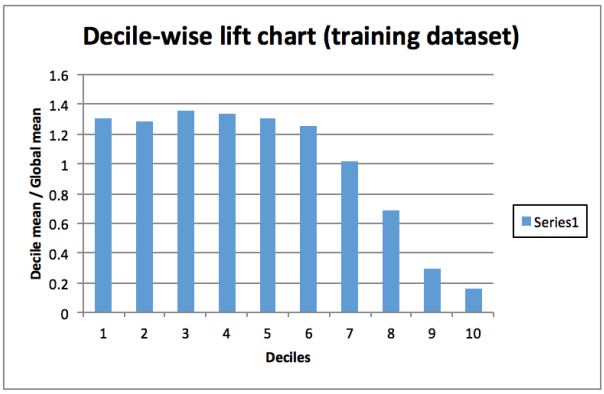
Error Report			
Class	# Cases	# Errors	% Error
non-default	400	21	5.25
default	160	60	37.5
Overall	560	81	14.46428571

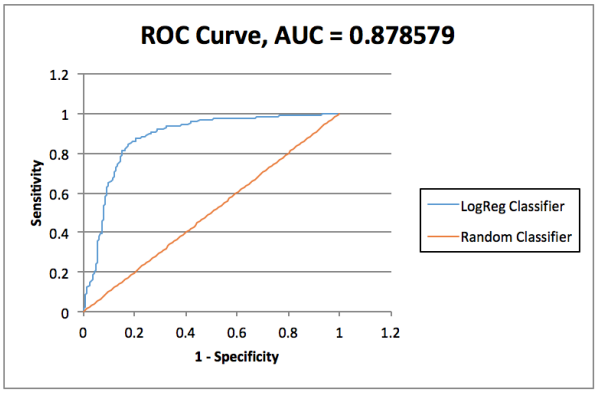
Performance	
Success Class	non-default
Precision	0.86332574
Recall (Sensitivity)	0.9475
Specificity	0.625
F1-Score	0.903456496

Training Lift Chart:

Lift Charts consist of a lift curve and a baseline. After the model is built using the Training Set, the model is used to score on the Training Set and the Validation Set. Then the data set are sorted using the predicted output variable value. After sorting, the actual outcome values of the output variable are cumulated, and the lift curve is drawn as the number of cases (x-axis) versus the cumulated value (y - axis). The baseline (red line connecting the origin to the end point of the blue line) is drawn as the number of cases versus the average of actual output variable values multiplied by the number of cases. The greater the area between the lift curve and the baseline, the better the model.



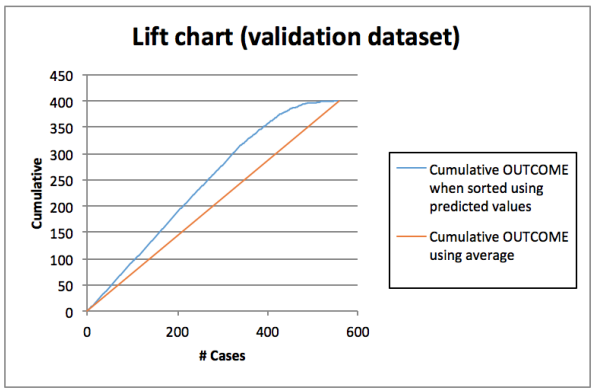


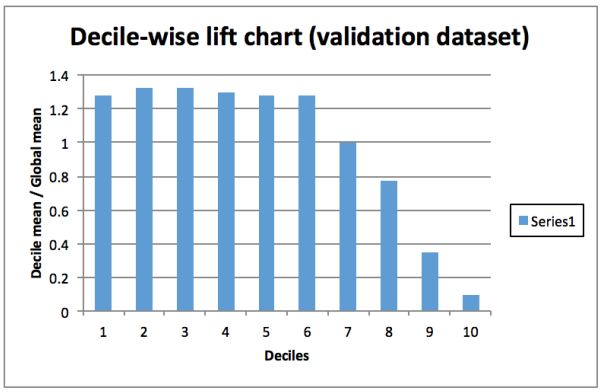


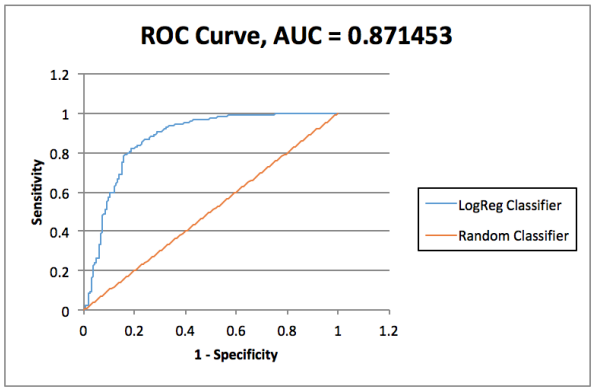
Decile	Mean	Std.Dev.	Min.	Max.
1	0.928571	0.259086	0	1
2	0.916667	0.278045	0	1
3	0.964286	0.186691	0	1
4	0.952381	0.214238	0	1
5	0.928571	0.259086	0	1
6	0.892857	0.311152	0	1
7	0.72619	0.448591	0	1
8	0.488095	0.50286	0	1
9	0.214286	0.41279	0	1
10	0.119048	0.32579	0	1

From the lift cart we can observe that in ROC curve we get the AUC=0.87. Also the curve is towards Y axis as expected.

Validating LiftChart:







Decile	Mean	Std.Dev.	Min.	Max.
1	0.910714	0.287736	0	1
2	0.946429	0.227208	0	1
3	0.946429	0.227208	0	1
4	0.928571	0.25987	0	1
5	0.910714	0.287736	0	1
6	0.910714	0.287736	0	1
7	0.714286	0.455842	0	1
8	0.553571	0.501621	0	1
9	0.25	0.436931	0	1
10	0.071429	0.25987	0	1

From the lift cart for test data we can observe that in ROC curve we get the AUC=0.87. Also the curve is towards Y axis as expected.

In the Lift Chart (Validating Set) pictured above, the red line originating from the origin and connecting to the point (600, 400) is a reference line that represents the expected number of CAT MEDV predictions if XLMiner simply selected random cases (i.e., no model was used). This reference line provides a yardstick against which the user can compare the model performance. From the Lift Chart below, we can infer that if we assigned 200 cases to class 1, about 150 1s would be included. If 200 cases were selected at random, we could expect about 180 1s.

The decile-wise lift curve is drawn as the decile number versus the cumulative actual output variable value divided by the decile's mean output variable value. This bars in this chart indicate the factor by which the MLR model outperforms a random assignment, one decile at a time.

ROC curves plot the performance of binary classifiers by graphing true positive rates (TPR) versus false positive rates (FPR) as the cutoff value grows from 0 to 1. The closer the curve is to the top-left corner of the graph, and the smaller the area above the curve, the better the performance of the model.

Model 2 - Cart:

Here we can see that for cart algorithm the probability for the two classes (default and non-default). In this model 20 decision nodes are taken as we can see for 20 decision node the %error is less.

Prior class probabilities

Empirical prior probabilities		
Class	Probability	
default	0.287753	
non-default	0.712247	<-- Success Class

Training Log (Growing the full tree using training data)

# Decision Nodes	% Error
0	28.77527
1	25.32699
2	16.76576
3	16.76576
4	16.52794
5	16.52794
6	16.52794
7	16.52794
8	16.52794
9	16.52794
10	14.38763
11	14.26873
12	14.26873
13	14.26873
14	11.89061
15	11.89061
16	11.89061
17	11.89061
18	11.89061
19	11.89061
20	11.89061

Full-Grown Tree Rules (Using Training Data)

#Decision Nodes	20

#Terminal Nodes	21

Training confusion matrix:

Now we can see that from the below confusion matrix we can see that out of 242 default approved mortgage 161 are correctly predicted and 81 are error while for non-default approved mortgage 580 are correctly predicted and 19 are wrong. The accuracy for this model is 0.92

Confusion Matrix		
	Predicted Class	
Actual Class	non-default	default
non-default	580	19
default	81	161

Error Report			
Class	# Cases	# Errors	% Error
non-default	599	19	3.171953
default	242	81	33.47107
Overall	841	100	11.89061

Performance	
Success Class	non-default
Precision	0.877458
Recall (Sensitivity)	0.96828
Specificity	0.665289
F1-Score	0.920635

Validation Confusion matrix:

Now we can see that from the below confusion matrix we can see that out of 242 default approved mortgage 161 are correctly predicted and 81 are error while for non-default approved mortgage 580 are correctly predicted and 19 are wrong. The accuracy for this model is 0.92

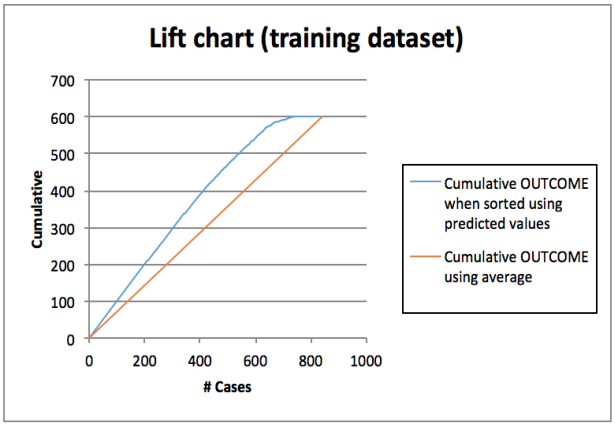
Confusion Matrix		
	Predicted Class	
Actual Class	non-default	default
non-default	386	14
default	76	84

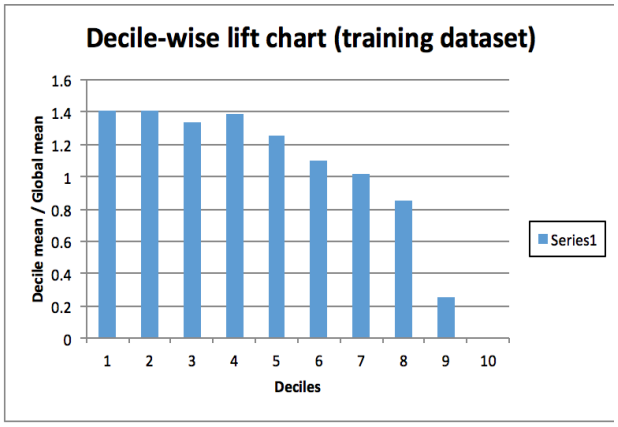
Error Report			
Class	# Cases	# Errors	% Error
non-default	400	14	3.5
default	160	76	47.5
Overall	560	90	16.07143

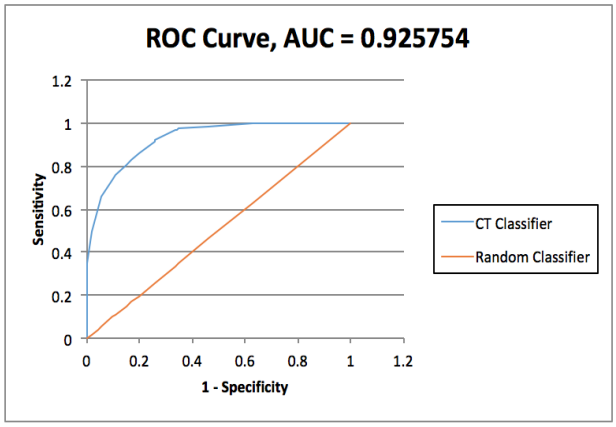
Performance	
Success Class	non-default
Precision	0.835498
Recall (Sensitivity)	0.965
Specificity	0.525
F1-Score	0.895592

Training liftchart:

Here in below chart we can see that in the ROC curve the AUC = 0.92 and also the curve is tilted towards Y axis as expected.



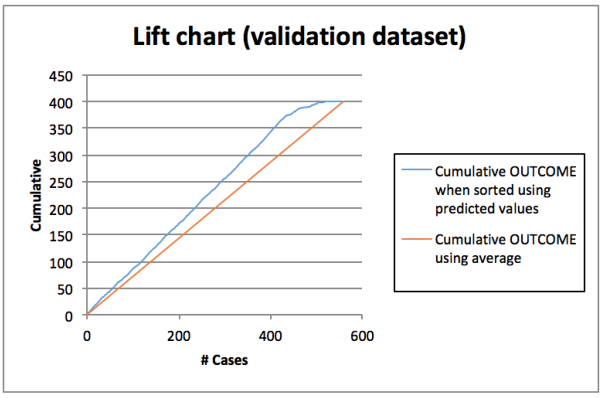


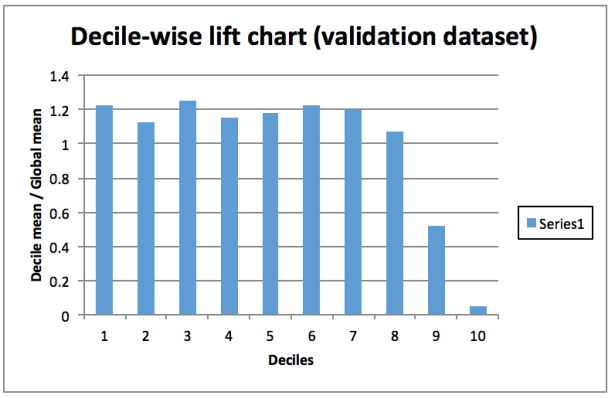


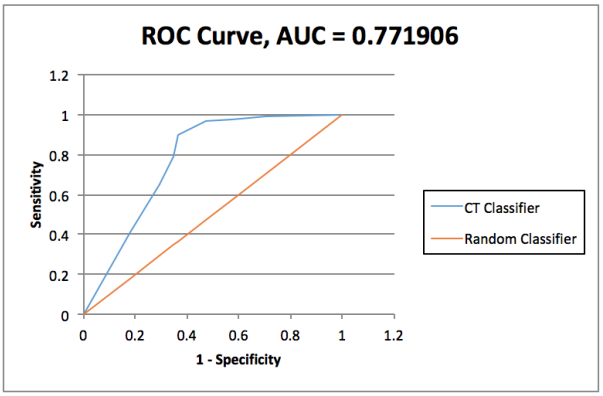
Decile	Mean	Std.Dev.	Min.	Max.
1	1	0	1	1
2	1	0	1	1
3	0.952381	0.214238	0	1
4	0.988095	0.109109	0	1
5	0.892857	0.311152	0	1
6	0.785714	0.41279	0	1
7	0.72619	0.448591	0	1
8	0.607143	0.491319	0	1
9	0.178571	0.385293	0	1
10	0	0	0	0

Validation Liftchart:

Here in below chart we can see that in the ROC curve the AUC = 0.77 and also the curve is tilted towards Y axis as expected.







Decile	Mean	Std.Dev.	Min.	Max.
1	0.875	0.333712	0	1
2	0.803571	0.400892	0	1
3	0.892857	0.312094	0	1
4	0.821429	0.386459	0	1
5	0.839286	0.370591	0	1
6	0.875	0.333712	0	1
7	0.857143	0.353094	0	1
8	0.767857	0.426021	0	1
9	0.375	0.488504	0	1
10	0.035714	0.187256	0	1

In the Lift Chart (Validating Set) pictured above, the red line originating from the origin and connecting to the point (600, 400) is a reference line that represents the expected number of CAT MEDV predictions if XLMiner simply selected random cases (i.e., no model was used). This reference line provides a yardstick against which the user can compare the model performance. From the Lift Chart below, we can infer that if we assigned 200 cases to class 1, about 150 1s would be included. If 200 cases were selected at random, we could expect about 180 1s.

The decile-wise lift curve is drawn as the decile number versus the cumulative actual output variable value divided by the decile's mean output variable value. This bars in this chart indicate the factor by which the MLR model outperforms a random assignment, one decile at a time.

ROC curves plot the performance of binary classifiers by graphing true positive rates (TPR) versus false positive rates (FPR) as the cutoff value grows from 0 to 1. The closer the curve is to the top-left corner of the graph, and the smaller the area above the curve, the better the performance of the model.

Model 3 - Random Forest:

Here we can observe the importance of each attribute, like credit score is the most import attribute. Now with the confusion matrix we get an overall %error as 11.11% and out of 284 default approved mortgage 210 were correctly predicted and 74 was error while for 697 non-default approved predicted mortgage 662 were correctly predicted.

Details of the random-trees ensemble

Variables	Credit_scor	pur_prc_an	Ln_Orig	Loanamt/p	DTI Ratio	Tot_mthly_	Tot_mthly_incm
Importance	0.604854	0.525722	0.498025	0.468716	0.345916	0.289018	0.267407155

Training Data scoring - Summary Report

Cutoff probability value for success (UPDATABLE)	0.5	Updating the value here will NOT update value in detailed report

Confusion Matrix		
	Predicted Class	
Actual Class	non-default	default
non-default	662	35
default	74	210

Error Report			
Class	# Cases	# Errors	% Error
non-default	697	35	5.021521
default	284	74	26.05634
Overall	981	109	11.11111

Performance	
Success Class	non-default
Precision	0.899457
Recall (Sensitivity)	0.949785
Specificity	0.739437
F1-Score	0.923936

Validation Data scoring - Summary Report

Cutoff probability value for success (UPDATABLE)	

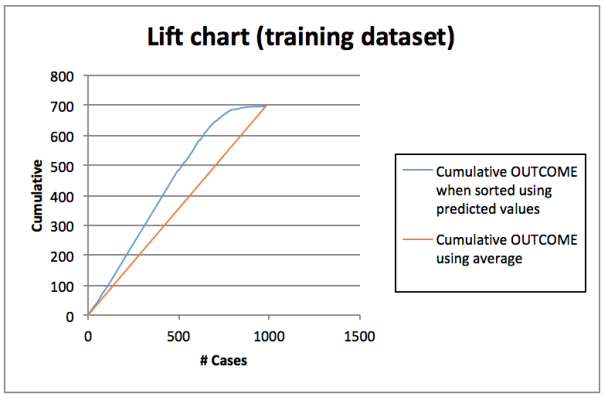
Confusion Matrix		
	Predicted Class	
Actual Class	non-default	default
non-default	292	10
default	43	75

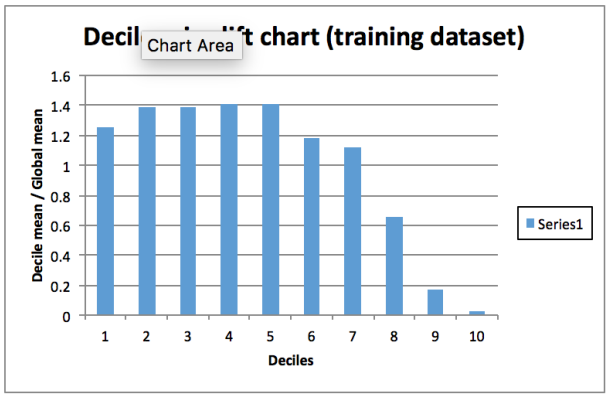
Error Report			
Class	# Cases	# Errors	% Error
non-default	302	10	3.311258
default	118	43	36.44068
Overall	420	53	12.61905

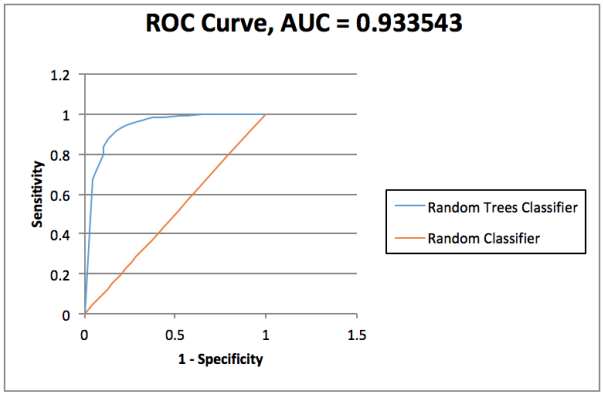
Performance	
Success Class	non-default
Precision	0.871642
Recall (Sensitivity)	0.966887
Specificity	0.635593
F1-Score	0.916797

With the confusion matrix we got the accuracy of the model as 0.91 and precision as 0.96

Here in below chart we can see that in the ROC curve the AUC = 0.93 and also the curve is tilted towards Y axis as expected.

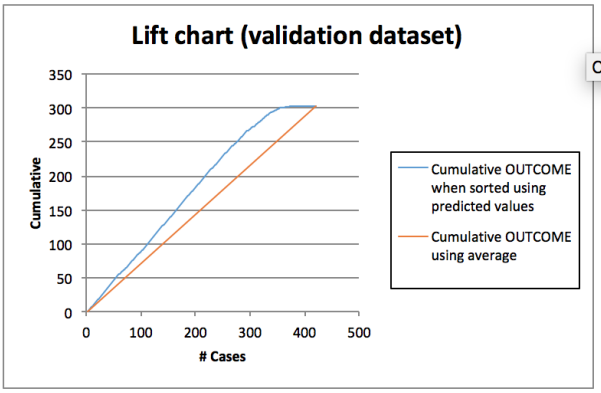


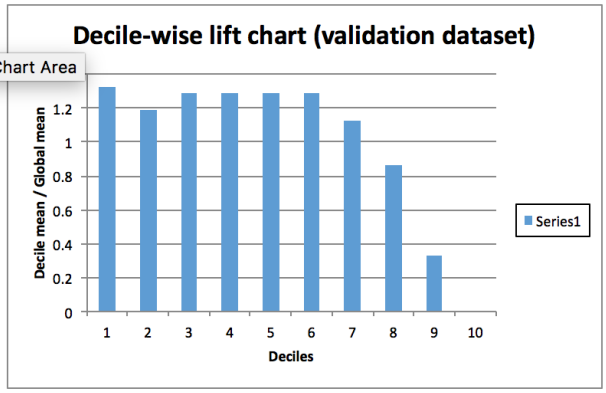


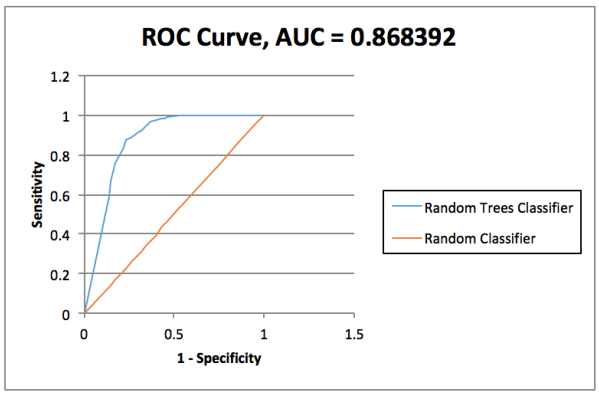


Decile	Mean	Std.Dev.	Min.	Max.
1	0.887755	0.31729	0	1
2	0.989796	0.101015	0	1
3	0.989796	0.101015	0	1
4	1	1.12E-16	1	1
5	1	1.12E-16	1	1
6	0.836735	0.371508	0	1
7	0.795918	0.405101	0	1
8	0.469388	0.501628	0	1
9	0.122449	0.329489	0	1
10	0.020408	0.142119	0	1

Here in below chart for validation data we can see that in the ROC curve the AUC = 0.86 and also the curve is tilted towards Y axis as expected.







Decile	Mean	Std.Dev.	Min.	Max.
1	0.952381	0.21554	0	1
2	0.857143	0.354169	0	1
3	0.928571	0.260661	0	1
4	0.928571	0.260661	0	1
5	0.928571	0.260661	0	1
6	0.928571	0.260661	0	1
7	0.809524	0.397437	0	1
8	0.619048	0.491507	0	1
9	0.238095	0.431081	0	1
10	0	0	0	0

In the Lift Chart (Validating Set) pictured above, the red line originating from the origin and connecting to the point (600, 400) is a reference line that represents the expected number of CAT MEDV predictions if XLMiner simply selected random cases (i.e., no model was used). This reference line provides a yardstick against which the user can compare the model performance. From the Lift Chart below, we can infer that if we assigned 200 cases to class 1, about 150 1s would be included. If 200 cases were selected at random, we could expect about 180 1s.

The decile-wise lift curve is drawn as the decile number versus the cumulative actual output variable value divided by the decile's mean output variable value. This bars in this chart indicate the factor by which the MLR model outperforms a random assignment, one decile at a time.

ROC curves plot the performance of binary classifiers by graphing true positive rates (TPR) versus false positive rates (FPR) as the cutoff value grows from 0 to 1. The closer the curve is to the top-left corner of the graph, and the smaller the area above the curve, the better the performance of the model.

Model Recommendation & Reasons:

- Random Forest model works very well compared to Cart and Logistic Regression.
- Also Random Forest Model most stable model amongst three
- Random forest algorithm has the accuracy better than the other two model
- Random forest has less overall % error, which tells that on predicted values random forest has less probability of error
- Not just accuracy of any model is important factor while deciding whether model is good or bad, the stability of model matter's the most.

Algorithm	Accuracy	Default error%	Non-default error%	precision	AUC Curve	Decile
Random Forest	91.6	36.4	3.3	87.1	0.86	Stepwise
CART	89.5	47.5	3.5	83.5	0.77	Stepwise
Linear Regression	97.5	37.5	5.25	86.3	0.87	Stepwise

Detecting Spam

Aim: Question 3

What data mining technique(s) would be appropriate to classify whether an email is a spam or not a spam? Justify your analysis and model.

Data set Description:

File – Spambase.xlsx

Features – 26

Records – 4601

Response Variable – Spam

Spam has two classes. 1 – The email is spam, 0 – The email is not spam

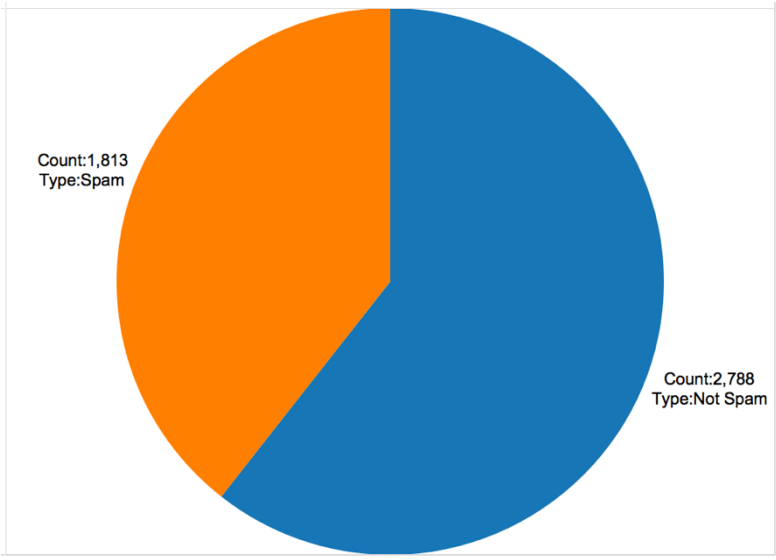
The features in the dataset are the average count of the words which appear in an email.

Analysis on Dataset:

Since the features consist of words, there is no scope to explore this dataset.

Analysis was done in Tableau

1) Spam vs Non Spam – Email Count



The pie chart displays the count of spam and non spam emails.

2) Spam vs Non Spam – Word List and Frequency

A	B	C	D	E
Non spam			**spam**	
word	**count**		**word**	**count**
you	3541.71		you	4105.61
george	3527.56		your	2502.61
hp	2496.58		will	997.1
will	1495.27		free	939.79
your	1223.1		our	931.8
hpl	1204.4		C!	931.361
re:	1159.14		all	732.08
edu	800.67		mail	635.47
address	681.57		email	578.76
meeting	604.46		business	521.25
all	559.22		remove	499.31
W_1999	551.31		W_000	447.91
W_650	540.33		font	431.56
our	504.74		money	385.95
W_85	472.44		internet	377.36
mail	466.07		credit	372.61
labs	462.4		over	317.05
lab	453.87		C$	316.329
C(	442.116		order	308.32
data	420.95		W_3d	298.55
technology	394.98		address	298.51
project	353.06		make	276.19
pm	339.24		people	260.25
C!	306.634		re:	226.79
telnet	295.62		receive	214.72
email	271.25		addresses	203.2
direct	231.73		C(	197.563
W_415	216.87		report	151.52
W_857	215.53		C#	143.004
free	205.16		W_1999	78.81
make	204.86		direct	66.57
cs	200.81		technology	53.51
original	196.78		C;	37.299

As displayed above, words and letters are categorized in two categories spam and non-spam. As seen in the real world scenario, spam emails contain words like free, business, etc. Count infers the total average occurrence of words in all the emails.

Applied Models:

The data was partitioned in 70(training):30(test)

After the data partition, three models were generated: -

4) CART
5) Random Forest
6) Logistic Regression

Model Evaluation Criteria

1) Accuracy
2) Error
3) AUC Score
4) Lift
5) Decile

1. CART

Confusion Matrix

Validation Data scoring - Summary Report (Using Best Pruned Tree)

Cutoff probability value for success (UPDATABLE)	0.5	Updating the value here will NOT update value in detailed report

Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	467	78
0	47	788

Error Report			
Class	# Cases	# Errors	% Error
1	545	78	14.311927
0	835	47	5.6287425
Overall	1380	125	9.057971

Performance	
Success Class	1
Precision	0.9085603
Recall (Sensitivity)	0.8568807
Specificity	0.9437126
F1-Score	0.8819641

Accuracy – 88%

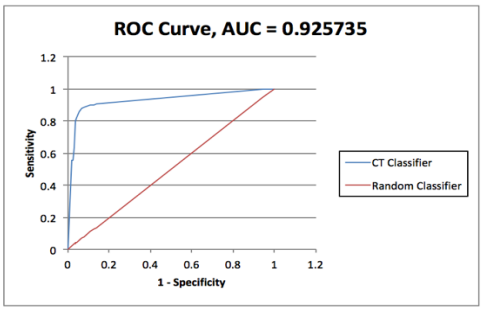
Precision – 90.86%

The value of precision should be higher as it implies that out of 835 non-spam emails 47 were categorized as spam.

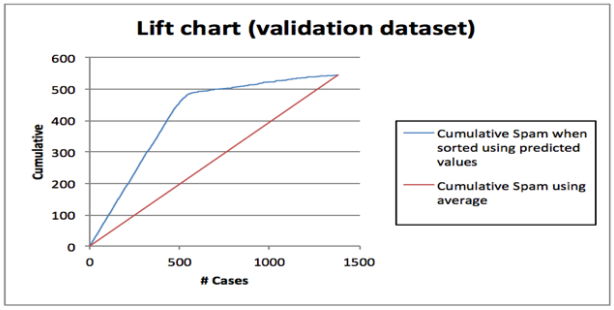
Categorizing a non-spam mail as spam is a blunder as the user will never get a chance to read this email.

Since 47 is a small number, CART model does a good job.

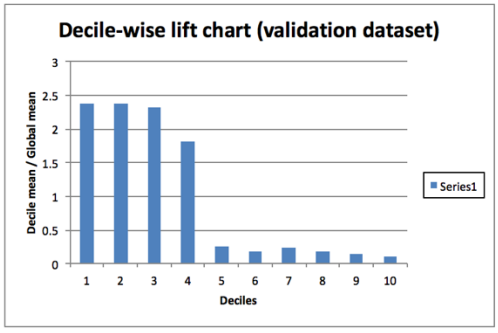
AUC Score



Lift Chart



Decile



In the Lift Chart (Validating Set) pictured above, the red line originating from the origin and connecting to the point (1400, 550) is a reference line that represents the expected number of CAT MEDV predictions if XLMiner simply selected random cases (i.e., no model was used). This reference line provides a yardstick against which the user can compare the model performance.

The model (represented in blue line) is lifted by 459 / 199 = 2.31 for 500 cases from the Random model (represented in red line). Hence the model satisfies the Lift chart

The decile-wise lift curve is drawn as the decile number versus the cumulative actual output variable value divided by the decile's mean output variable value. The above decile chart forms a stepwise pattern and hence CART model satisfies the decile chart.

ROC curves plot the performance of binary classifiers by graphing true positive rates (TPR) versus false positive rates (FPR) as the cutoff value grows from 0 to 1. The closer the curve is to the top-left corner of the graph, and the smaller the area above the curve, the better the performance of the model.

2. Random Forest

Confusion Matrix

Validation Data scoring - Summary Report

Cutoff probability value for success (UPDATABLE)	0.5	Updating the value here will NOT update value in detailed report

Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	492	53
0	44	791

Error Report			
Class	# Cases	# Errors	% Error
1	545	53	9.7247706
0	835	44	5.2694611
Overall	1380	97	7.0289855

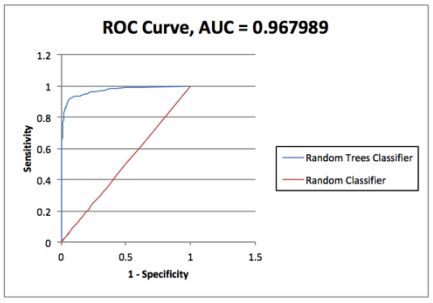
Performance	
Success Class	1
Precision	0.9179104
Recall (Sensitivity)	0.9027523
Specificity	0.9473054
F1-Score	0.9102683

Accuracy – 91%

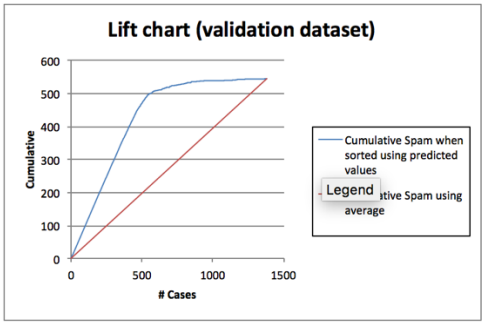
Precision – 91.79%

The value of precision should be higher as it implies that out of 835 non-spam emails 44 were categorized as spam.
Categorizing a non-spam mail as spam is a blunder as the user will never get a chance to read this email.
Since 44 is a small number, Random Forest model does a good job.
Better than CART.

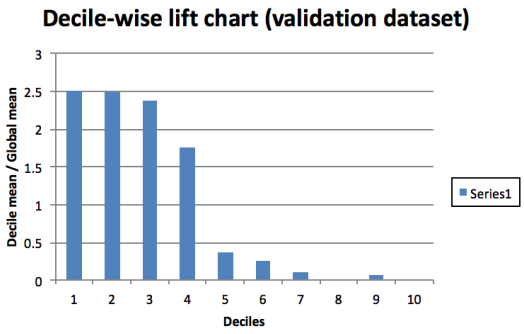
AUC Score



Lift Chart



Decile



In the Lift Chart (Validating Set) pictured above, the red line originating from the origin and connecting to the point (1400, 550) is a reference line that represents the expected number of CAT MEDV predictions if XLMiner simply selected random cases (i.e., no model was used). This reference line provides a yardstick against which the user can compare the model performance.
The model (represented in blue line) is lifted by 469 / 196 = 2.39 for 500 cases from the Random model (represented in red line). Hence the model satisfies the Lift chart

The decile-wise lift curve is drawn as the decile number versus the cumulative actual output variable value divided by the decile's mean output variable value. The above decile chart forms a stepwise pattern and hence CART model satisfies the decile chart.

ROC curves plot the performance of binary classifiers by graphing true positive rates (TPR) versus false positive rates (FPR) as the cutoff value grows from 0 to 1. The closer the curve is to the top-left corner of the graph, and the smaller the area above the curve, the better the performance of the model.

3. Linear Regression

Confusion Matrix

Validation Data scoring - Summary Report

Cutoff probability value for success (UPDATABLE)	0.5	Updating the value here will NOT update value in detailed report

Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	492	53
0	44	791

Error Report			
Class	# Cases	# Errors	% Error
1	545	53	9.7247706
0	835	44	5.2694611
Overall	1380	97	7.0289855

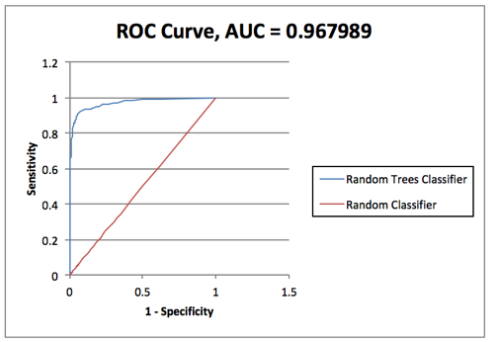
Performance	
Success Class	1
Precision	0.9179104
Recall (Sensitivity)	0.9027523
Specificity	0.9473054
F1-Score	0.9102683

Accuracy – 91%

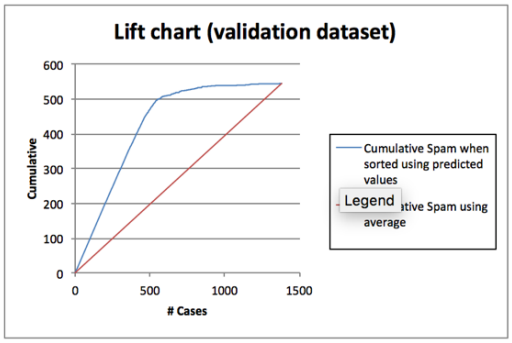
Presidion – 91.79%

The value of precision should be higher as it implies that out of 835 non-spam emails 47 were categorized as spam.
Categorizing a non-spam mail as spam is a blunder as the user will never get a chance to read this email.
Since 47 is a small number, CART model does a good job.

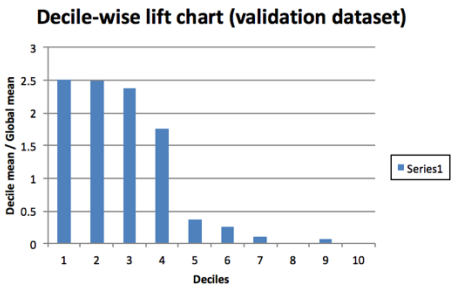
AUC Score



Lift Chart



Decile



In the Lift Chart (Validating Set) pictured above, the red line originating from the origin and connecting to the point (1400, 550) is a reference line that represents the expected number of CAT MEDV predictions if XLMiner simply selected random cases (i.e., no model was used). This reference line provides a yardstick against which the user can compare the model performance.
The model (represented in blue line) is lifted by 470/198 = 2.37 for 500 cases from the Random model (represented in red line). Hence the model satisfies the Lift chart

The decile-wise lift curve is drawn as the decile number versus the cumulative actual output variable value divided by the decile's mean output variable value. The above decile chart forms a stepwise pattern and hence CART model satisfies the decile chart.

ROC curves plot the performance of binary classifiers by graphing true positive rates (TPR) versus false positive rates (FPR) as the cutoff value grows from 0 to 1. The closer the curve is to the top-left corner of the graph, and the smaller the area above the curve, the better the performance of the model.

Best Model

Algorithm	**Accuracy**	**Error**	**Precision**	**AUC Score**	**Lift**	**Decile**
CART	88.20	9.9	90.86	0.93	2.31	Not Stepwise
Random Forest	91.03	7.5	91.79	0.97	2.39	Stepwise
Logistic Regression	89.94	8.6	91.30	0.97	2.37	Stepwise

From the table above, Random forest has the highest precision, accuracy, AUC score and a maximum lift. Hence, model built by applying Random forest classification Algorithm over the 70% training and 30% validation data is the best model.

If we are interested mainly in detecting spam messages, is this model useful? Use the confusion matrix, lift chart, and decile chart for the validation set for the evaluation.

Algorithm Used – Random Forest

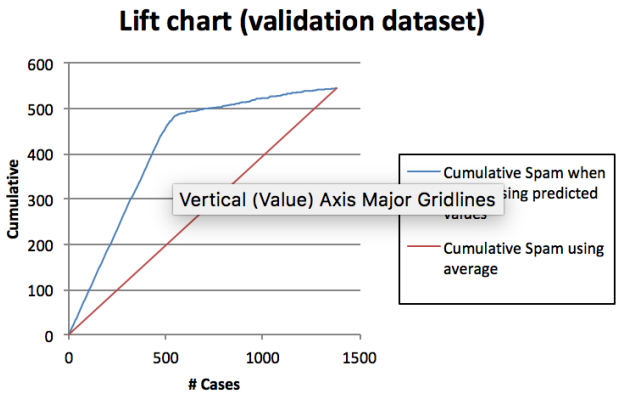
Confusion Matrix -

Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	492	53
0	44	791

Error Report			
Class	# Cases	# Errors	% Error
1	545	53	9.7247706
0	835	44	5.2694611
Overall	1380	97	7.0289855

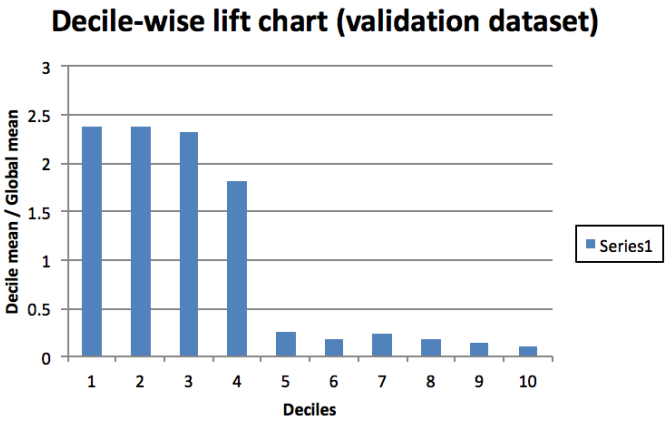
From the confusion matrix above, the error in classifying an email as spam is 9% and while Classifying an email as not spam is 5%. Since the overall error margin and the error margin between spam and not spam is narrow, the designed model is useful for spam detection.

Lift Chart –



The model (represented in blue line) is lifted by 459 / 199 = 2.31 for 500 cases from the Random model (represented in red line). Hence the model satisfies the Lift chart Evaluation criteria.

Decile –



The above decile chart forms a stepwise pattern and hence model satisfies the decile chart

Model Recommendation & Reasons:

- Random Forest is better than CART and Logistic regression
- Random forest is the best performing model with respect to accuracy, precision, lift and decile analysis
- The model is fit to classify both the spam and non spam emails.
- The concern in this problem is to look out for emails which are not spam but marked as spam. The count of such emails should be as small as possible.
  Hence Precision as criteria for evaluation is the most important parameter.
  The precision is maximum for random forest. Hence there is narrow chance that the model will predict a non-spam email as spam.

Feedback Prediction for Blog

Aim:

What data mining technique(s) would be appropriate in predicting the total number of comments on blog in next 24 hours? Conduct such an analysis.

Data set Description:

The blog post data set provided is crawled from internet from Hungarian blog post website. The data set contains one train csv file and several other test csv files. The train csv files has around 281 columns, out of which five columns(56-60) describes the occurrence of comments before or after base time, meaning the column has the number of comments on each particular blog post before or after certain base time. The columns (1-55) are the average, min, max, standard deviation, and mean of columns 56-60. The data set contains 200 columns of high frequency random words and their count in particular blog post. We have around seven columns for weekdays from Mon to Friday indicating binary digits on which week day the post was released and another seven columns for weekdays from Mon to Friday describing binary digits for base time. We also have one column describing trackbacks in blog post.

Analysis on Datset:

The columns 56 to 60 are number of comments and according to base time and other factors, and we had average, min, etc. of these numbers, but our question was how is average, min, etc. is calculated because after observing the first fifty columns we found that the numbers are constant for a range and again varying for some range and so on. The assumption made for this was the columns 56-60 must be having numbers of comments for same blog post or comments for same topic of blog post and so the average is calculated for those numbers which lies in same group.

The data in train file is given for two years 2010 and 2011 and we have different test files for February and March month of 2012. We have merged the test files in four different groups one test file consisting of all data for February month only, and three files for march because we notice the pattern in naming convention of march test files that for end of march test files the files were saved with different format of 01 and rest of march files were saved with 00, so one march file consisting of entire march data set, one containing just files with 00 names and lastly one containing just 01 names files, also the reason behind doing the same, was to check the stability of model since the base time were selected randomly so there might be some overlapping in base time. If we apply model on many test files and we see stability in model for different files then model is advisable to use.

Below is the code used for merging the test files in R Studio.

```
#reading all the csv file from local

a <- read.csv ("/Users/rashmiyadav/Desktop/DataScience/BlogFeedback/blogData_test-2012.02.01.00_00.csv")
b <- read.csv("/Users/rashmiyadav/Desktop/DataScience/BlogFeedback/blogData_test-2012.02.02.00_00.csv")
c <- read.csv("/Users/rashmiyadav/Desktop/DataScience/BlogFeedback/blogData_test-2012.02.03.00_00.csv")
d <- read.csv("/Users/rashmiyadav/Desktop/DataScience/BlogFeedback/blogData_test-2012.02.04.00_00.csv")
e <- read.csv("/Users/rashmiyadav/Desktop/DataScience/BlogFeedback/blogData_test-2012.02.05.00_00.csv")
f <- read.csv("/Users/rashmiyadav/Desktop/DataScience/BlogFeedback/blogData_test-2012.02.06.00_00.csv")
```

```
# assigning header to all the file
names(a)
names(b) <- names(a)
names(eem) <- names(a)

# binding all the february file in one and march file in one

feb <- rbind(a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z,af,bf,cf)
mar00 <- rbind(am,bm,cm,dm,em,fm,gm,hm,im,jm,km,lm,mm,nm,om,pm,qm,rm,sm,tm,um,vm,wm,xm,ym)
mar01 <- rbind(zm,aam,bbm,ccm,ddm,eem)

march <- rbind(am,bm,cm,dm,em,fm,gm,hm,im,jm,km,lm,mm,nm,om,pm,qm,rm,sm,tm,um,vm,wm,xm,ym,zm,aam,bbm,ccm,ddm,eem)

#writing the binded csv file in the local

write.csv(feb, "/Users/rashmiyadav/Desktop/DataScience/BlogFeedback/febdata.csv", row.names=FALSE)
write.csv(mar00, "/Users/rashmiyadav/Desktop/DataScience/BlogFeedback/mar00data.csv", row.names=FALSE)
write.csv(mar01, "/Users/rashmiyadav/Desktop/DataScience/BlogFeedback/mar01data.csv", row.names=FALSE)
write.csv(march, "/Users/rashmiyadav/Desktop/DataScience/BlogFeedback/marchdata.csv", row.names=FALSE)
```

The correlation matrix describes the how input attributes are closely related to output attributes and based on this matrix we eliminated the attributes from data set in order to avoid the over fitting of model. Avoiding over fitted model is important mainly because then model gives the accuracy based on noisy data which is not good sign.

Co-relation matrix: The highly co related variables are shown below.

	Target No of comments in next 24hrs
Average	0.485464073
Std Dev	0.424615858
Min	0.053220686
Max	0.356604233
Median	0.491707225
Average	0.497631274
Std Dev.1	0.433577859
Min.1	0.034915539
Max.1	0.322105946
Median.1	0.506540261
Average.1	0.490111451
Std Dev.2	0.439151622
Min.2	
Max.2	0.322774902
Median.2	0.489673584
Average.2	0.47199879
Std Dev.3	0.384654317
Min.3	0.053220686
Max.3	0.299688146
Median.3	0.486315568
Average.3	0.503374635

A	B
	Target No of comments in next 24hrs
Length of	0.048209172
63	0.004429267
64	0.033751668
65	-0.001567582
66	-0.000267548
67	0.064112303
68	0.021816863
69	0.061238154
70	0.001491315
71	-0.005750275
72	0.031312132
73	-0.001567582
74	0.000457146
75	0.005094144
76	-0.001567582
77	0.033373688
78	-0.003552652
79	0.044714564
80	-0.001567582
81	-0.001567582
82	0.002536885

Due to large number of attributes in dataset the correlation matrix file generated is huge. To view entire matrix click here corr.csv

Applied Models:

- **Model 1 – Random Forest**
- **Model 2 - Cart(Decision Tree)**
- **Model 3 - Linear Regression**

Model 1: Performing Prediction algorithm using Random Forest algorithm for predicting the total numbers of comments in next 24 hours on train and test files of blog feedback dataset.
The parameters selected were on basis of co relation matrix. Also this problem was solved in Python. The three models were implemented in Python.

Model score, accuracy are shown below for just February test files:

```
metrics.mean_absolute_error(test_out,RandomForest_predict)
```

```
5.8959787655204767
```

```
metrics.r2_score(test_out, RandomForest_predict)
```

```
0.4679801300697628
```

```
metrics.mean_squared_error(test_out, RandomForest_predict) ** 0.5
```

```
24.00404233710999
```

Model score, accuracy are shown below for just March(All) test files:

```
metrics.mean_absolute_error(test_out,RandomForest_predict)
```

```
6.3947825950468768
```

```
metrics.r2_score(test_out, RandomForest_predict)
```

```
0.16098946643355749
```

```
metrics.mean_squared_error(test_out, RandomForest_predict) ** 0.5
```

```
26.117543092269621
```

Model score, accuracy are shown below for just March(00) test files:

```
metrics.mean_absolute_error(test_out,RandomForest_predict)
```

```
5.8682576649107068
```

```
metrics.r2_score(test_out, RandomForest_predict)
```

```
0.026258769071276111
```

```
metrics.mean_squared_error(test_out, RandomForest_predict) ** 0.5
```

```
24.321312991807684
```

Model score, accuracy are shown below for just March(01) test files:

```
metrics.mean_absolute_error(test_out,RandomForest_predict)
```

```
8.2039268235407903
```

```
metrics.r2_score(test_out, RandomForest_predict)
```

```
0.34448559758173081
```

```
metrics.mean_squared_error(test_out, RandomForest_predict) ** 0.5
```

```
31.518688535418235
```

Model 2 CART: Performing Prediction algorithm using CART algorithm for predicting the total numbers of comments in next 24 hours on train and test files of blog feedback dataset.

The parameters selected were on basis of co relation matrix.

Model score, accuracy are shown below for just February test files:

```
metrics.mean_absolute_error(test_out, Cart_predict)
```

```
7.1654941965925492
```

```
metrics.r2_score(test_out, Cart_predict)
```

```
0.15190513018814333
```

```
metrics.mean_squared_error(test_out, Cart_predict) ** 0.5
```

```
30.306980937806365
```

Model score, accuracy are shown below for just March(All) test files:

```
metrics.mean_absolute_error(test_out, Cart_predict)
```

```
7.6365503729247113
```

```
metrics.r2_score(test_out, Cart_predict)
```

```
-0.3715869332270223
```

```
metrics.mean_squared_error(test_out, Cart_predict) ** 0.5
```

```
33.393372560418783
```

Model score, accuracy are shown below for just March(00) test files:

```
metrics.mean_absolute_error(test_out, Cart_predict)
```

```
7.5055484049645464
```

```
metrics.r2_score(test_out, Cart_predict)
```

```
-0.91484571524503244
```

```
metrics.mean_squared_error(test_out, Cart_predict) ** 0.5
```

```
34.106105456555852
```

Model score, accuracy are shown below for just March(01) test files:

```
metrics.mean_absolute_error(test_out, Cart_predict)
```

```
8.7720605573571522
```

```
metrics.r2_score(test_out, Cart_predict)
```

```
0.12791910183203958
```

```
metrics.mean_squared_error(test_out, Cart_predict) ** 0.5
```

```
36.354268170411594
```

Model 3 Linear Regression: Performing Prediction algorithm using Linear Regression algorithm for predicting the total numbers of comments in next 24 hours on train and test files of blog feedback dataset.

The parameters selected were on basis of co relation matrix.

Model score, accuracy is shown below for just February test files:

```
metrics.mean_absolute_error(test_out, LR_Predict)
```

```
6.9580073309722899
```

```
metrics.r2_score(test_out, LR_Predict)
```

```
0.42407579033798881
```

```
metrics.mean_squared_error(test_out, LR_Predict) ** 0.5
```

```
24.97486361364815
```

Model score, accuracy is shown below for just March(All) test files:

```
metrics.mean_absolute_error(test_out, LR_Predict)
```

```
7.0681584101389392
```

```
metrics.r2_score(test_out, LR_Predict)
```

```
0.16498943011140044
```

```
metrics.mean_squared_error(test_out, LR_Predict) ** 0.5
```

```
26.055211315195464
```

Model score, accuracy are shown below for just March(00) test files:

```
metrics.mean_absolute_error(test_out, LR_Predict)
```

```
6.4784656888507888
```

```
metrics.r2_score(test_out, LR_Predict)
```

```
0.064441725274235417
```

```
metrics.mean_squared_error(test_out, LR_Predict) ** 0.5
```

```
23.839693036702378
```

Model score, accuracy are shown below for just March (01)test files:

```
metrics.mean_absolute_error(test_out, LR_Predict)
```

```
9.094347717124684
```

```
metrics.r2_score(test_out, LR_Predict)
```

```
0.30141495944779872
```

```
metrics.mean_squared_error(test_out, LR_Predict) ** 0.5
```

```
32.537685739399173
```

Model Recommendation & Reasons:

- Random Forest model works very well compared to Cart and Linear Regression.
- Also Random Forest Model most stable model amongst three
- Random Forest model has less root mean square error than cart and Linear Regression, which tells that on predicted values random forest has less probability of error
- Not just accuracy of any model is important factor while deciding whether model is good or bad, the stability of model matter's the most.