MIDTERM CASE STUDY

INFO7390 Advances in Data Sciences and Architecture

MARCH 18, 2016

GROUP 9

- Neha Gilson
- Aastha Grover
- Shuxian Wu

West Roxbury dataset

About the Data:

The dataset includes details about residential homes in Roxbury like the area of house, number of rooms, bedrooms, baths, kitchen and if it is remodeled. The only categorical variable was "Remodel".

Goal:

The goal of this case is to make a prediction for the total value of a home. The original dataset comes with following variables:

Input Variables:

TAX	Tax bill amount based on total assessed value multiplied by the 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)

Output Variable:

TOTAL VALUE	Total assessed value for property, in thousands of USD

Initial assumptions:

We took a look at the dataset and got a sense of how these variables will work interactively. We made assumptions, such as the tax variable may not be valid, considering the total value since the tax is calculated based on the value of home with a constant tax rate. Secondly, the remodel variable needs to be transformed into numerical value for prediction in regression since it is categorical.

Preprocessing:

We used Wrangler to clean the dataset by handling missing data, data format and data transformation.

For the remodel variable we created dummies in XLMiner:

1- None
2- Recent

3- Old

This is how the data set looked after cleansing and transforming.

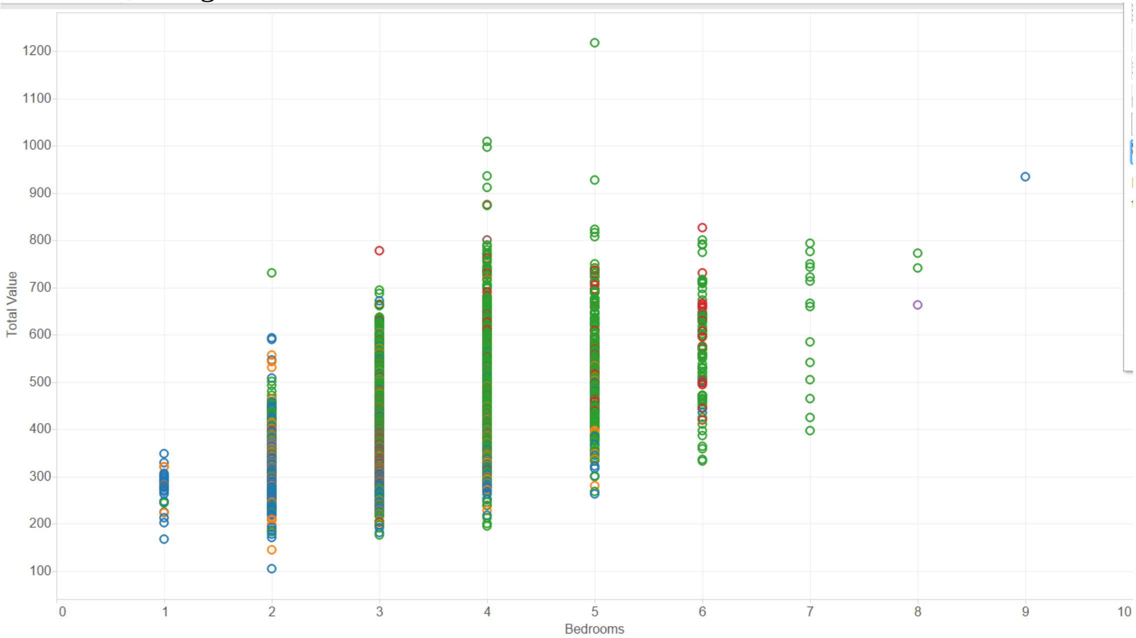
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	TOTAL_VA	TAX	LOT_SQFT	YR_BUILT	GROSS_AF	LIVING_AF	FLOORS	ROOMS	BEDROOM	FULL_BATI	HALF_BAT	KITCHEN	FIREPLACE	REMODEL	REMODEL	REMODEL	REMODEL_OLD
2	344.2	4330	9965	1880	2436	1352	2	6	3	1	1	1	0	1	1	0	0
3	412.6	5190	6590	1945	3108	1976	2	10	4	2	1	1	0	2	0	1	0
4	330.1	4152	7500	1890	2294	1371	2	8	4	1	1	1	0	1	1	0	0
5	498.6	6272	13773	1957	5032	2608	1	9	5	1	1	1	1	1	1	0	0
6	331.5	4170	5000	1910	2370	1438	2	7	3	2	0	1	0	1	1	0	0
7	337.4	4244	5142	1950	2124	1060	1	6	3	1	0	1	1	3	0	0	1
8	359.4	4521	5000	1954	3220	1916	2	7	3	1	1	1	0	1	1	0	0
9	320.4	4030	10000	1950	2208	1200	1	6	3	1	0	1	0	1	1	0	0
10	333.5	4195	6835	1958	2582	1092	1	5	3	1	0	1	1	2	0	1	0
11	409.4	5150	5093	1900	4818	2992	2	8	4	2	0	1	0	1	1	0	0
12	313	3937	5000	1960	2624	1485	1.5	6	3	2	0	1	1	1	1	0	0
13	344.5	4333	6768	1958	2844	1460	1.5	6	3	2	0	1	1	1	1	0	0
14	315.5	3968	5000	1889	2196	1290	2	6	3	1	0	1	0	1	1	0	0
15	575	7233	12288	2004	4616	2378	2	9	4	2	1	1	1	1	1	0	0
16	326.2	4103	5000	1954	2536	1272	1.5	6	3	1	1	1	1	1	1	0	0
17	298.2	3751	5000	1940	2129	864	1	7	3	2	0	1	0	1	1	0	0
18	313.1	3938	6949	1880	2612	1438	1.5	7	3	1	1	1	0	3	0	0	1
19	344.9	4338	10000	1950	2099	1445	1	7	3	1	1	1	1	1	1	0	0
20	330.7	4160	5000	1910	2408	1470	2	7	3	1	0	1	0	1	1	0	0
21	348	4377	9001	1875	2840	1632	2	7	3	1	0	1	0	1	1	0	0
22	317.5	3994	4450	1920	1400	1232	2	7	3	1	0	1	0	1	1	0	0
23	330.8	4161	5000	1889	2560	1302	1.5	6	2	1	0	1	0	2	0	1	0
24	357.8	4501	12255	1944	2631	1275	1.5	6	3	1	1	1	1	1	1	0	0
25	414.7	5216	12972	1892	3796	2054	1.5	6	3	3	0	1	0	1	1	0	0
26	318.8	4010	4717	1900	2512	1371	2	7	3	1	1	1	0	1	1	0	0
27	346.2	4355	5000	1910	2655	1541	2	7	3	1	0	1	0	1	1	0	0
28	245.1	3083	4142	1880	1892	927	1.5	5	2	1	0	1	0	1	1	0	0
29	317.4	3992	5000	1910	2596	1244	1.5	7	3	1	0	1	0	1	1	0	0
30	247.3	3111	4012	1896	1994	1048	1.5	6	3	1	0	1	0	1	1	0	0
31	320.8	4035	13275	1910	1516	808	1.5	6	2	1	1	1	0	1	1	0	0
32	328.8	4136	4717	1906	2824	1469	1.5	6	3	2	0	1	0	1	1	0	0
33	293.6	3693	4289	1910	2286	1436	2	7	3	1	0	1	0	1	1	0	0
34	280.1	3523	10688	1890	3014	1625	2	7	4	1	0	1	0	1	1	0	0
35	342.5	4308	4717	1925	2802	1620	2	7	3	1	1	1	0	1	1	0	0
36	337.2	4241	5000	1920	3112	1906	2	9	4	2	0	1	0	1	1	0	0
37	336.5	4233	6842	1910	2269	1393	2	6	3	2	0	1	0	1	1	0	0

roxbury_20 | Data_Partition | RT_Output | RT_PruneLog | RT_Stored | RTRandTrees_Output1 | RTRandTrees_Stored1 | RTRandTrees_Outp ...

READY

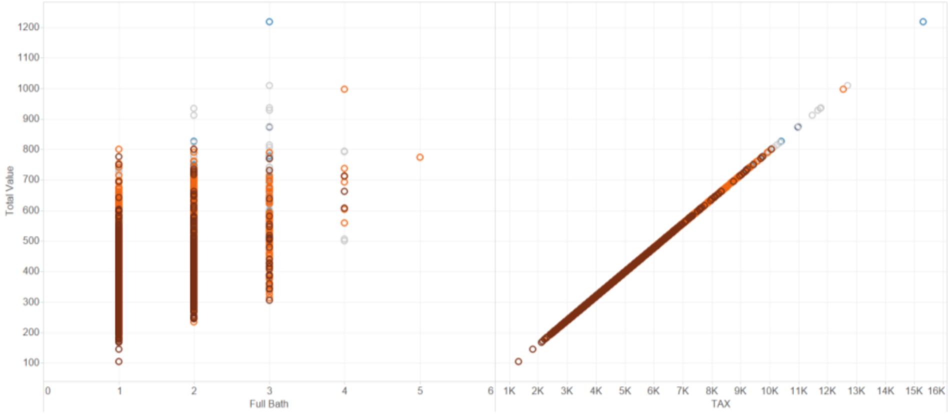
Data Exploration:

Through exploration in Tableau, we found that an increase in the number of Bedrooms does not have a very heavy impact on the value of the home. As seen below, the highest value homes are not necessarily the ones with more number of bedrooms, though that is the trend.



We also saw that most of the recently built houses that aren't remodeled have higher value.
An interesting observation was the area of living area had more impact on the value of the home that the total lot area.

We saw that (on the right) Tax and total value have a very close correlation and therefore it will have lesser impact on the prediction model.



We then used this analysis for variable selection in XLMiner.

Prediction Models

1. Regression:
   We used Multiple Linear Regression. The Regression Model and the R^2 values are as shown below for the best combination of selected features.

Regression Model

Input Variables	Coefficient	Std. Error	t-Statistic	P-Value	CI Lower	CI Upper	RSS Reduction
Intercept	81.73986	8.095967859	10.096367	1.21E-23	65.86652	97.6132	540722411.1
GROSS_ARI	0.039159	0.002148107	18.2295121	5.57E-71	0.034947	0.043371	22000133.51
LIVING_ARI	0.059046	0.003948602	14.9537353	4.8E-49	0.051305	0.066788	2293353.986
FLOORS	32.80695	2.175158162	15.0825561	7.74E-50	28.54223	37.07166	715820.0306
FULL_BATH	19.76298	1.807257727	10.9353422	2.17E-27	16.21959	23.30638	198496.3742
HALF_BATH	21.35429	1.684754698	12.6750127	5.05E-36	18.05108	24.6575	462409.4286
KITCHEN	-13.8607	6.630945631	-2.0903066	0.036663	-26.8617	-0.85976	11822.18145
FIREPLACE	21.02084	1.454702087	14.4502715	5.26E-46	18.16868	23.873	433093.5273
REMODEL_	22.81284	2.287246174	9.97393244	4.05E-23	18.32836	27.29732	216822.49

Residual DF	3472
R^2	0.776767
Adjusted R^2	0.776252
Std. Error Estimate	46.6859
RSS	7567479

The RMSE error is as shown below for training and validation data.

Training Data Scoring - Summary Report

Total sum of squared errors	RMS Error	Average Error
7567479	46.62551	1.02787E-13

Validation Data Scoring - Summary Report

Total sum of squared errors	RMS Error	Average Error
5442563	48.43483	-0.81780799

2. CART

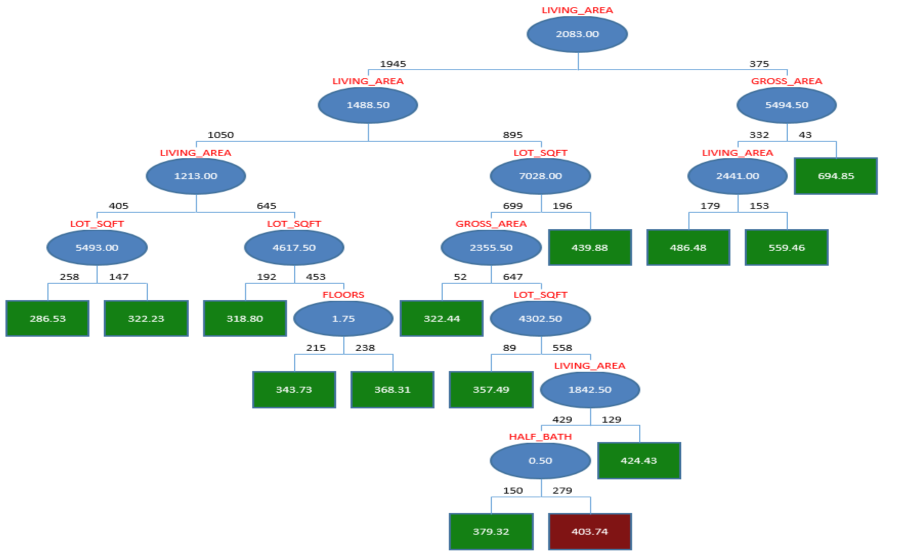
Training Data scoring - Summary Report (Using Best Pruned Tree)

Total sum of squared errors	RMS Error	Average Error
8830962	50.36769	-1.88444E-14

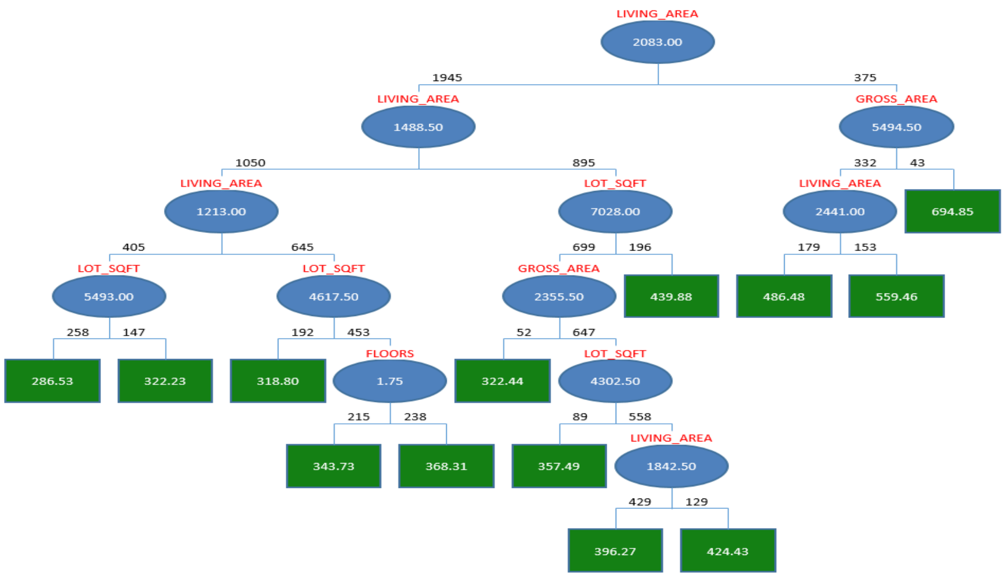
Validation Data scoring - Summary Report (Using Best Pruned Tree)

Total sum of squared errors	RMS Error	Average Error
7198673	55.70347	-0.141074006

Minmum Error Tree:



Best Pruned Tree:



3. Random Forest

Training Data scoring - Summary Report

Total sum of squared errors	RMS Error	Average Error
7499558	46.4158	0.767672

Validation Data scoring - Summary Report

Total sum of squared errors	RMS Error	Average Error
5803052	50.01315	-0.61458

Conclusions:

We compared the RMS errors and AUC in all the 3 above models and clearly, the best model is Multiple Linear Regression. RMSE has a significantly higher correlation to the distance from the ground truth on average than AUC.

We would recommend that for future prediction of home values the MLR model gives a more accurate result.

Mortgage Defaults

About the data:

The dataset contains approved loans and the factors affecting whether these loans were defaulted or not.

Goal:

The goal of this case is to classify whether any future approved loans will be default or non default.

Input Variables:

Bo_Age	Borrower age
Ln_Orig	Value of loan, USD
Orig_LTV_Ratio_Pct	Ratio of loan to home purchase price
Credit_score	Borrower's credit score
First_home	First time home buyer? (Y/N)
Tot_mthly_debt_exp	Borrower's total monthly debt expense
Tot_mthly_incm	Borrower's total monthly income
orig_apprd_val_amt	Appraised value of home at origination
pur_prc_amt	Purchase price for house
DTI_ratio	Borrower debt to income ratio (Tot_mthly_debt_exp/Tot_mthly_incm)
Status	Current loan status
State	US state in which home is located
Median_state_inc	Median household income by state 2002-2004

UPB>Appraisal	Loan amount (Ln_Orig) greater than appraisal (orig_apprd_val_amt) 0-no, 1=yes

Output Variable:

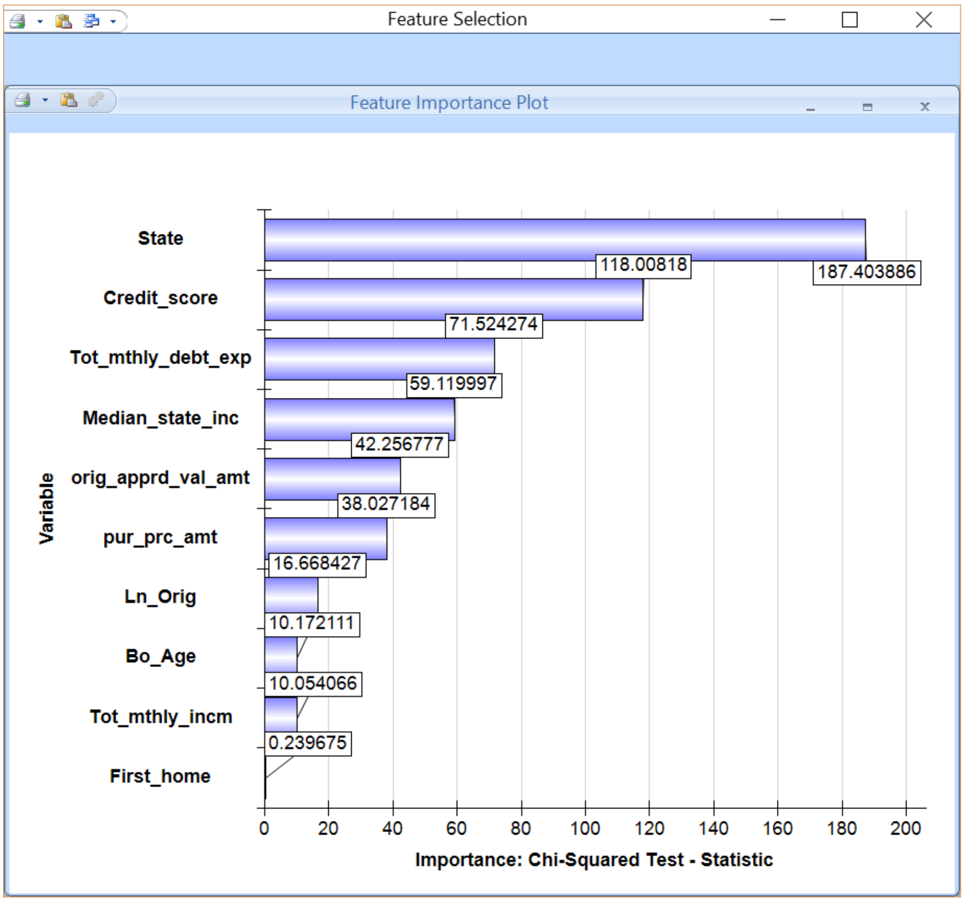
OUTCOME	Binary version of "Status" (either default or non-default)

Initial assumptions:

We took a look at the dataset and got a sense of how these variables will work interactively. We made assumptions that since there are features that are derived from other features they could be ignored in the classification model. For e.g. DTI ration is derived from Total monthly debt and total monthly income of the borrower.

Preprocessing:

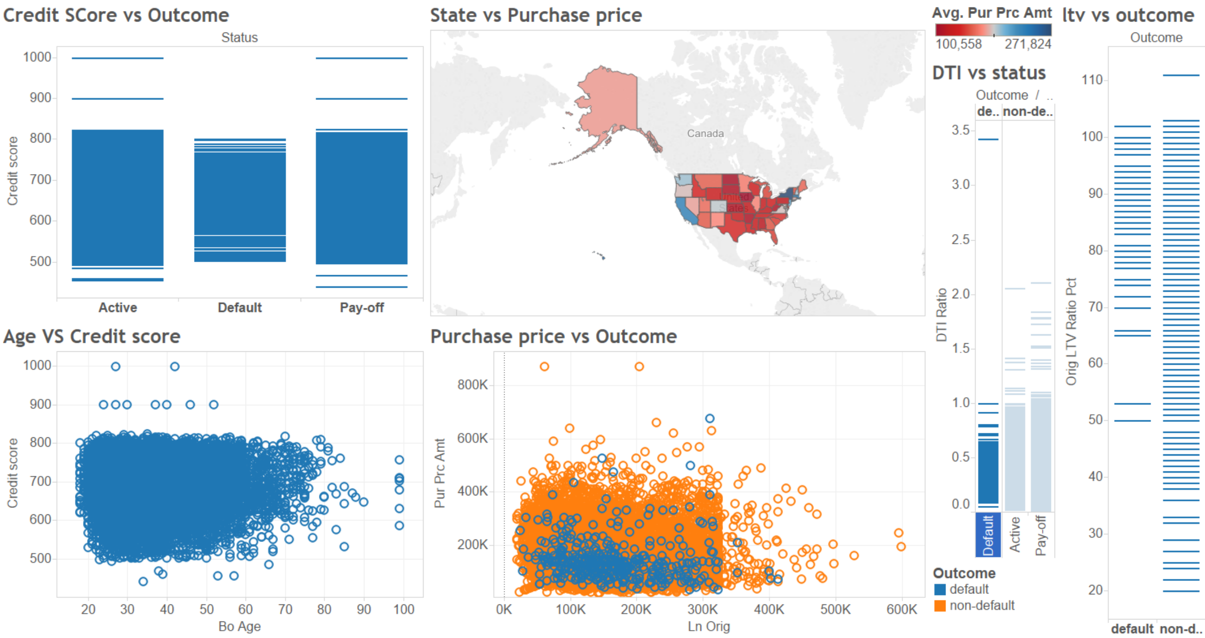
The last 2 columns of the data were irrelevant and they were removed. There was no missing data. There were some data that had Credit score above 850 which is an anomaly, so we removed them. There were a few categorical data for which we created dummy variables. Also, the data contained status of mortgage based on states. Since there were 50 such variables we used feature selection to decide which variables had more impact on the model.



When we built the model based on the above variables, we noticed that including or dropping all the states in the model did not have a major impact. So we dropped the states with the least impact.

Data Exploration

The dashboard for data exploration in Tableau is as shown below:



- The first graph shows the relation between Credit score and Outcome of the mortgage. We see that most of the defaulters fall in the range of 570 – 766.
- The second graph shows that relation between Age and Credit score. We see that most of the older age group people have better credit scores and thus in turn are non-defaulters as per the first graph.
- The third graph shows the purchase price of the homes through different states. We can see that New York state and California have the highest purchase prices.
- The fourth graph shows purchase price versus outcome. It is interesting to see that most of the defaulters are for lower priced homes and lower loan amount mortgages indicated by blue. Whereas as the purchase price increases the loan amount increases and the mortgage ends up in the non-default category.
- We also see from the fifth and sixth graph that most of the defaulters have a lower DTI ratio and higher LTV ratio.

These explorations help us understand the effect of different features in the model. We understand that a good credit score would lean towards the non-defaulter category. The higher purchase price of the homes combined with lower LTV ratio and good DTI Ratio will probably be a non-defaulter.

Classification Model:

Random Forest:

Validation Data scoring - Summary Report

Cutoff probability value for success (UPDATABLE)	0.7	Updating the value here will NOT update value in detailed report

Confusion Matrix

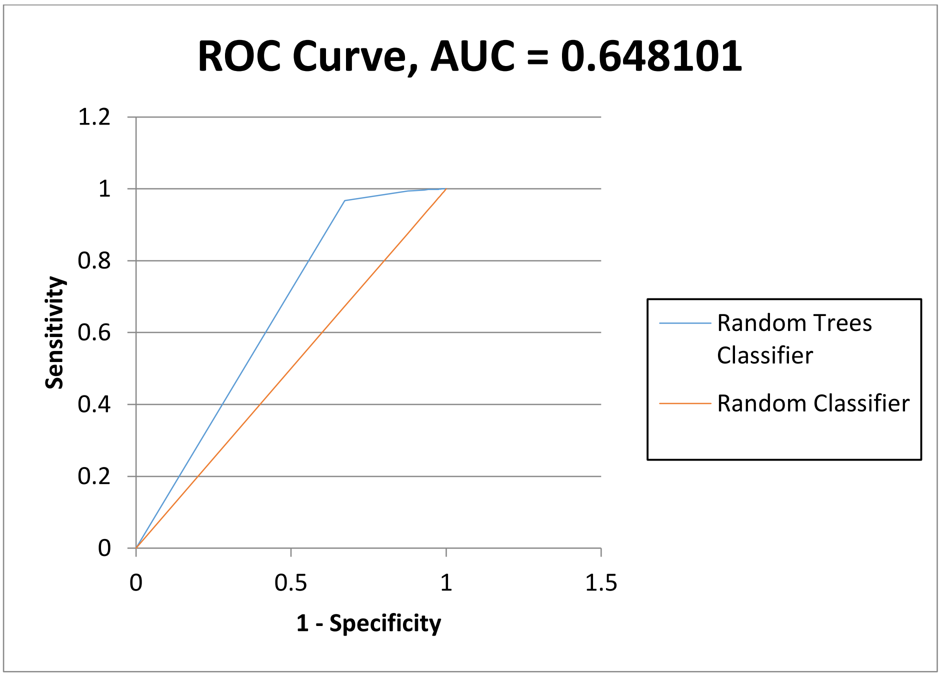
	Predicted Class	
Actual Clas	non-default	default
non-defaul	5896	5
default	160	0

Error Report

Class	# Cases	# Errors	% Error
non-defaul	5901	5	0.084731
default	160	160	100
Overall	6061	165	2.722323

Performance

Success Class	on-default
Precision	0.97358
Recall (Sensitivity)	0.999153
Specificity	0
F1-Score	0.986201



CART:

When considering status:

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

Cutoff probability value for success (UPDATABLE)	0.75	Updating the value here will NOT update value in detailed report

Confusion Matrix

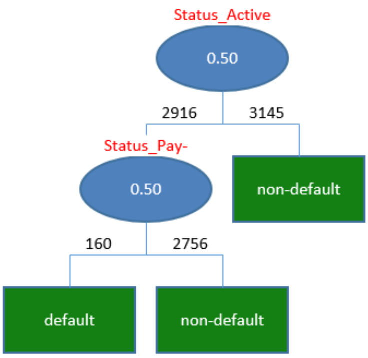
	Predicted Class	
Actual Clas	non-default	default
non-defaul	5901	0
default	0	160

Error Report

Class	# Cases	# Errors	% Error
non-defaul	5901	0	0
default	160	0	0
Overall	6061	0	0

Performance

Success Class	on-default
Precision	1
Recall (Sensitivity)	1
Specificity	1
F1-Score	1



When not considering status:

Cutoff probability value for success (UPDATABLE)	0.75	Updating the value here will NOT update value in detailed report

Confusion Matrix

	Predicted Class	
Actual Clas	non-default	default
non-defaul	5901	0
default	160	0

Error Report

Class	# Cases	# Errors	% Error
non-defaul	5901	0	0
default	160	160	100
Overall	6061	160	2.639828

Performance

Success Class	on-default
Precision	0.973602
Recall (Sensitivity)	1
Specificity	0
F1-Score	0.986624

Logistic Regression:

When not considering status:

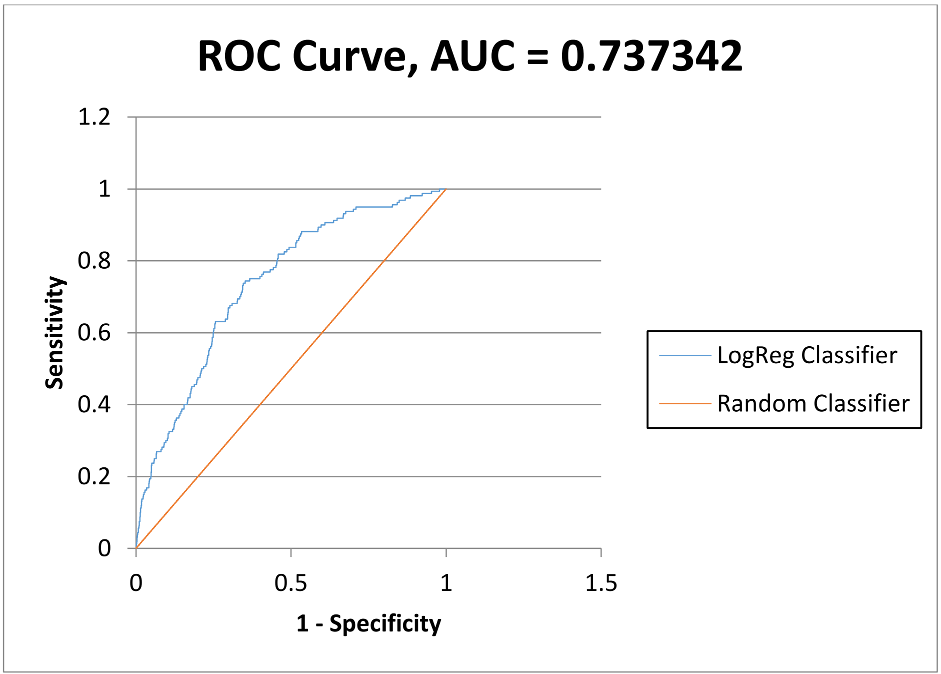
Cutoff probability value for success (UPDATABLE)	0.7	Updating the value here will NOT update value in detailed report

Confusion Matrix

	Predicted Class	
Actual Class	default	non-default
default	0	160
non-default	0	5901

Error Report

Class	# Cases	# Errors	% Error
default	160	160	100
non-default	5901	0	0
Overall	6061	160	2.639828411



When considering status:

Training Data Scoring - Summary Report

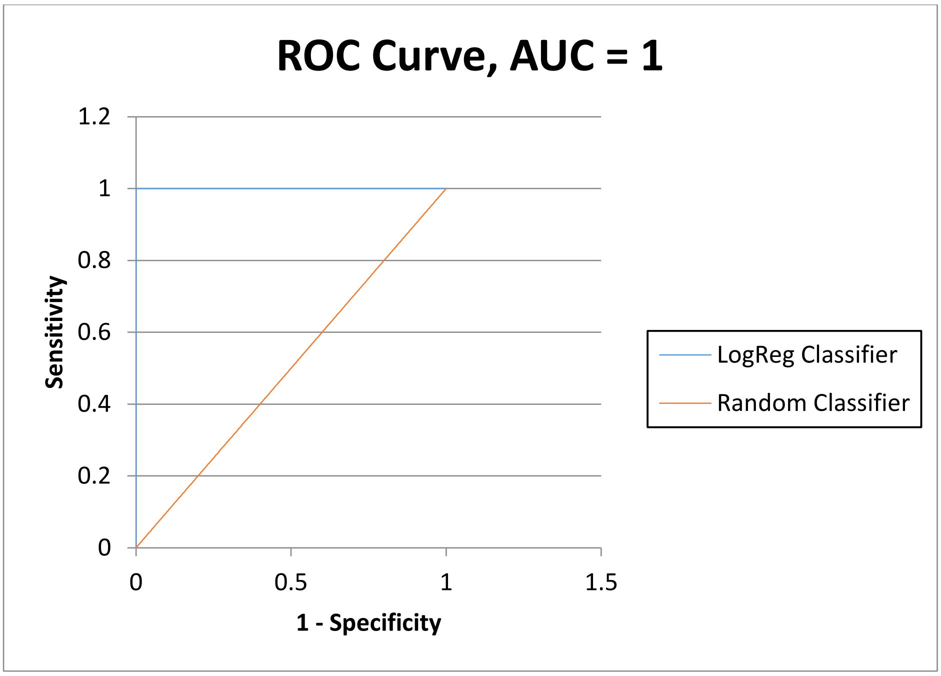
Cutoff probability value for success (UPDATABLE)	0.7	Updating the value here will NOT update value in detailed report

Confusion Matrix

	Predicted Class	
Actual Class	non-default	default
non-default	14751	0
default	0	402

Error Report

Class	# Cases	# Errors	% Error
non-default	14751	0	0
default	402	0	0
Overall	15153	0	0



Conclusion:

	Logistic	Cart	Random Forest
Overall % Error	2.63	0	2.7
AUC	0.73	1	0.64
Cutoff Value	0.7	0.75	0.7

The Random Forest Model gives a better classification of mortgage that will fall under non-default. But for classifying a default mortgage it is not a good model. But, if a mortgage falls under non-default and it is still in active status then we cannot say whether it will default in the future. So, Random Forest is not a very good classification model in this case.

When using status variable, CART and Logistic regression models give a perfect model to predict both the default and non-default mortgages. But since this variable is very closely related to the outcome in the sense that it is the binary variation of status, we chose to ignore this feature. When doing so, CART fails to develop a regression tree. Therefore, we conclude that Logistic regression gives a better model to predict the classification of mortgage into defaulters and non-defaulters.

Based on the confusion matrix and the AUC of ROC curve we come to a conclusion that Logistic regression is a better option.

Detecting Spam

About the data:

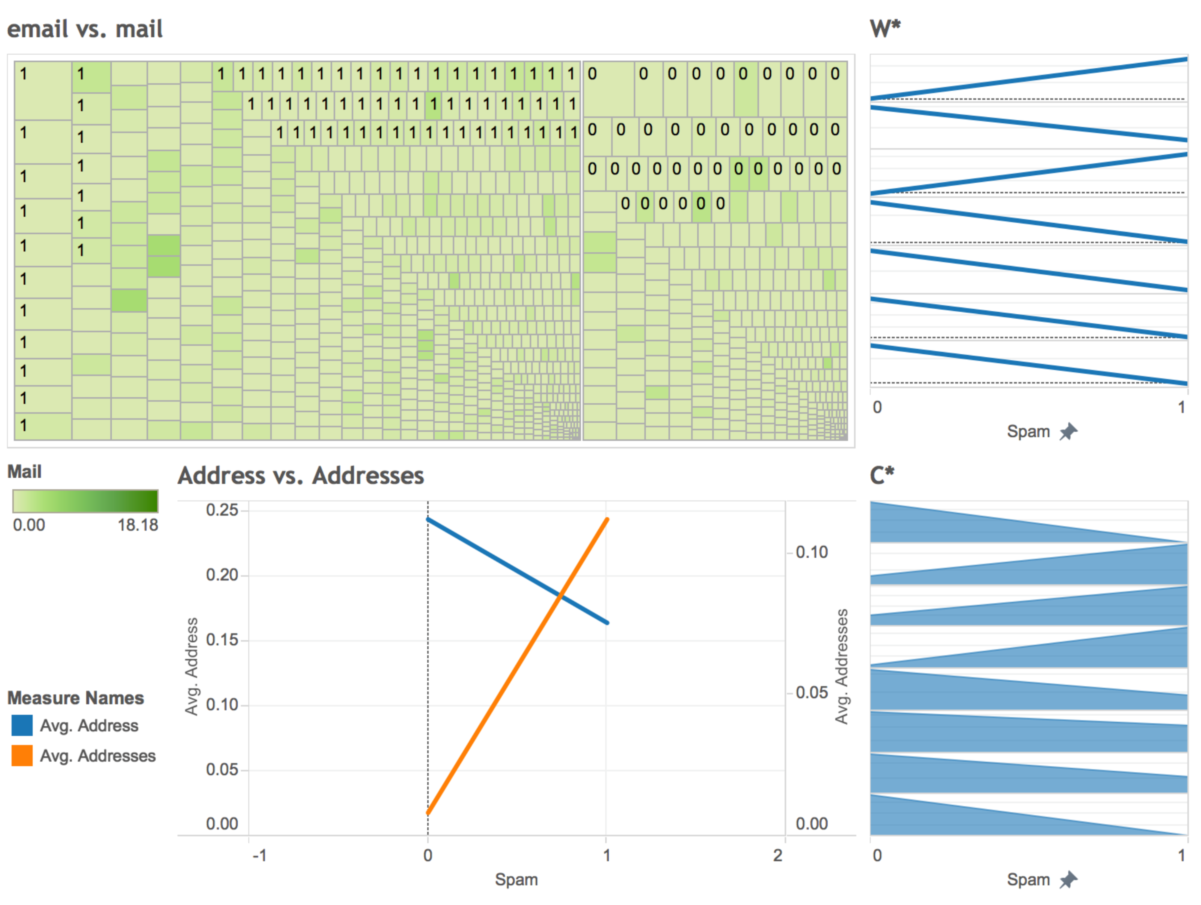
In the spam dataset, there are 57 input variables and the output variable is spam which has binary value for indicating whether an email is spam or not. Among the 57 input variables, we made assumptions that some of them can be group together to compare how they impact the value of spam.

Goal:

Building a classification model to classify an email into spam or normal email.

Data Exploration:

Since there are so many combinations of the input variables, we decided only some parts of them as examples in our report.



- email vs. mail: The graph indicates the relationship among variable email, mail and spam. We found that mail is less significant impact on spam and email tends to be more influential.

- W*: Grouping variables starting with "W" following by numbers and characters, we compared the average of each variable so that we are able to assume the first and the third variables in the chart has higher significant level in classification model.
- Address vs. Addresses: Comparing address (blue line) and addresses (orange line), addresses has more records on indicating a spam email.
- C*: Similar to W* chart, we grouped the variables stating with "C" following by special symbols. We might use the 2nd, 3rd, 4th variables in our model.

Classification Model:

To test our assumption, we started with Feature Selection:

Detailed Feature Selection Report

	Chi-Squared Test			Information	
Feature Identifier	Chi2-Statistic	Chi2: P-Value	Cramer's V	Mutual Information	Gain Ratio
make	137.5187	7.78249E-26	0.1729	0.0217	0.0411
address	28.6703	7.02269E-05	0.0789	0.0067	0.0379
all	340.1574	7.69784E-68	0.2719	0.0527	0.048
W_3d	20.0478	0.002715637	0.066	0.0038	0.1093
our	230.793	1.12004E-44	0.224	0.0358	0.0582
over	182.1153	6.96364E-36	0.199	0.029	0.0774
remove	350.2192	8.13028E-71	0.2759	0.0619	0.1647
internet	83.2277	7.68489E-16	0.1345	0.0133	0.0694
order	214.0686	1.91308E-43	0.2157	0.0334	0.0742
mail	6.2012	0.287131843	0.0367	0.0011	0.0059
receive	362.183	2.2684E-73	0.2806	0.0582	0.1038
will	68.3415	1.05027E-11	0.1219	0.0136	0.0134
people	76.4904	1.89293E-14	0.1289	0.0117	0.0295
report	31.2848	2.23646E-05	0.0825	0.0049	0.0256
addresses	173.962	3.66557E-34	0.1944	0.0293	0.1023
free	139.45	1.30921E-27	0.1741	0.0233	0.1104
business	218.4334	8.25082E-43	0.2179	0.0349	0.0797
email	160.2524	1.41635E-30	0.1866	0.0251	0.0535
you	496.0813	4.8753E-102	0.3284	0.0785	0.0544
credit	96.1872	6.37157E-20	0.1446	0.0172	0.1353
your	885.1952	9.5748E-185	0.4386	0.1404	0.1067
font	45.9188	2.46298E-07	0.0999	0.0076	0.0496
W_000	487.9573	2.1765E-99	0.3257	0.0875	0.1855
money	66.1567	8.80188E-12	0.1199	0.0111	0.0885
hp	266.0901	3.97289E-52	0.2405	0.0605	0.1083
hpl	154.7122	2.03684E-29	0.1834	0.0358	0.0985
george	134.5469	7.04265E-26	0.171	0.032	0.0912
W_650	126.6065	3.2133E-24	0.1659	0.0276	0.0781
lab	57.6654	1.3352E-09	0.112	0.0138	0.0805
labs	156.4096	4.13418E-29	0.1844	0.0344	0.0799
telnet	38.9824	7.02497E-08	0.092	0.0089	0.0696
W_857	91.8235	7.00666E-16	0.1413	0.0219	0.0862
data	42.531	4.5989E-08	0.0961	0.0094	0.0665
W_415	90.6935	1.18163E-15	0.1404	0.0213	0.0823
W_85	41.5368	5.03096E-09	0.095	0.01	0.0848
technology	62.8084	4.14031E-11	0.1168	0.0128	0.0408
W_1999	127.766	8.21352E-24	0.1666	0.0246	0.0505
parts	8.7875	0.185885124	0.0437	0.0019	0.0435

Selected Predictors

Feature Rank	Feature Name	Chi-Squared Test
1	your	885.195213
2	you	496.081275
3	W_000	487.9572861
4	receive	362.1829967
5	remove	350.2192079
6	all	340.1573859
7	hp	266.0901139
8	our	230.7929561
9	business	218.4333606
10	order	214.0685625
11	over	182.115294
12	addresses	173.9619595
13	email	160.2523771
14	labs	156.4095527
15	hpl	154.712187
16	free	139.4500479
17	make	137.5187064
18	george	134.5469238
19	C$	129.6825235
20	W_1999	127.765953
21	W_650	126.6065054
22	original	103.3914124
23	credit	96.18716851
24	W_857	91.82345911
25	W_415	90.69350939
26	meeting	84.93075976
27	internet	83.22773692
28	re:	80.92512644
29	people	76.49044907
30	CAP_tot	73.28576977
31	will	68.3414871
32	money	66.15673511
33	edu	66.10136636
34	technology	62.80842936

For our model, we decided to consider only top 22 out of 57 which has Chi-Squared Test value larger than 100.

Logistic Regression:

Regression Model

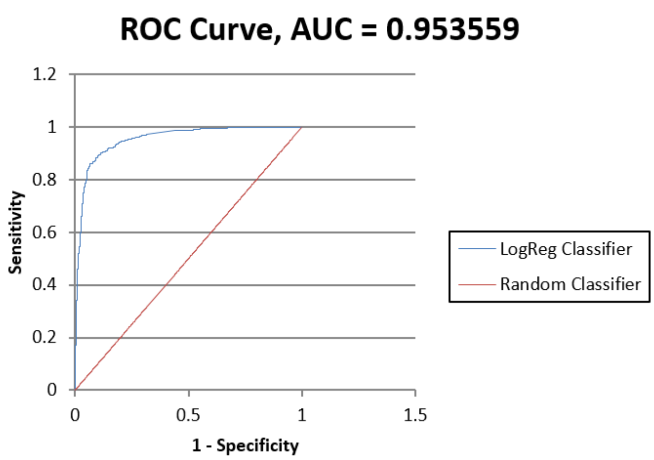
Input Variables	Coefficient	Std. Error	Chi2-Statistic	P-Value	Odds	CI Lower	CI Upper
Intercept	-1.674156	0.113905	216.0278907	6.65E-49	0.187466	0.149957	0.234358
make	-0.612527	0.298894	4.19967561	0.040432	0.541979	0.301692	0.973648
our	0.364309	0.088511	16.94131139	3.86E-05	1.439519	1.210255	1.712213
over	1.204124	0.303057	15.78676227	7.09E-05	3.333837	1.840694	6.038195
remove	3.271507	0.466437	49.19367236	2.32E-12	26.35101	10.56246	65.73994
order	1.553831	0.361006	18.52589633	1.68E-05	4.729553	2.330942	9.596407
receive	0.156934	0.374669	0.1754451	0.675317	1.169919	0.561354	2.438228
addresses	2.737868	1.331878	4.225675773	0.039817	15.45399	1.135911	210.2505
free	1.300777	0.170827	57.98212795	2.65E-14	3.67215	2.627316	5.132493
business	1.28063	0.262981	23.71375372	1.12E-06	3.598907	2.14942	6.025874
email	0.423368	0.181095	5.465396376	0.019397	1.527096	1.070823	2.177785
you	0.037153	0.0341	1.187090162	0.275917	1.037852	0.970755	1.109587
your	0.272186	0.057685	22.26404941	2.38E-06	1.312831	1.172484	1.469976
W_000	3.013064	0.622724	23.41131846	1.31E-06	20.34966	6.004733	68.96371
hp	-2.253787	0.423924	28.26506674	1.06E-07	0.105001	0.045745	0.241011
hpl	-1.145955	0.537424	4.546743735	0.032981	0.31792	0.110882	0.911537
george	-8.615513	3.064265	7.905143394	0.004929	0.000181	4.47E-07	0.073559
W_650	0.046164	0.259905	0.031549071	0.859021	1.047247	0.629242	1.742932
labs	-0.388739	0.326134	1.420774867	0.233276	0.677911	0.35774	1.284631
W_1999	-0.30017	0.234431	1.639473523	0.200398	0.740692	0.467832	1.172697
original	-1.01239	0.75894	1.779432796	0.182219	0.363349	0.082094	1.608181
C$	9.172841	1.072114	73.20245556	1.17E-17	9631.949	1177.956	78758.86

Residual Dl	2739
Residual D	1455.628
# Iteration	9
Multiple R²	0.606502

R^2=0.61

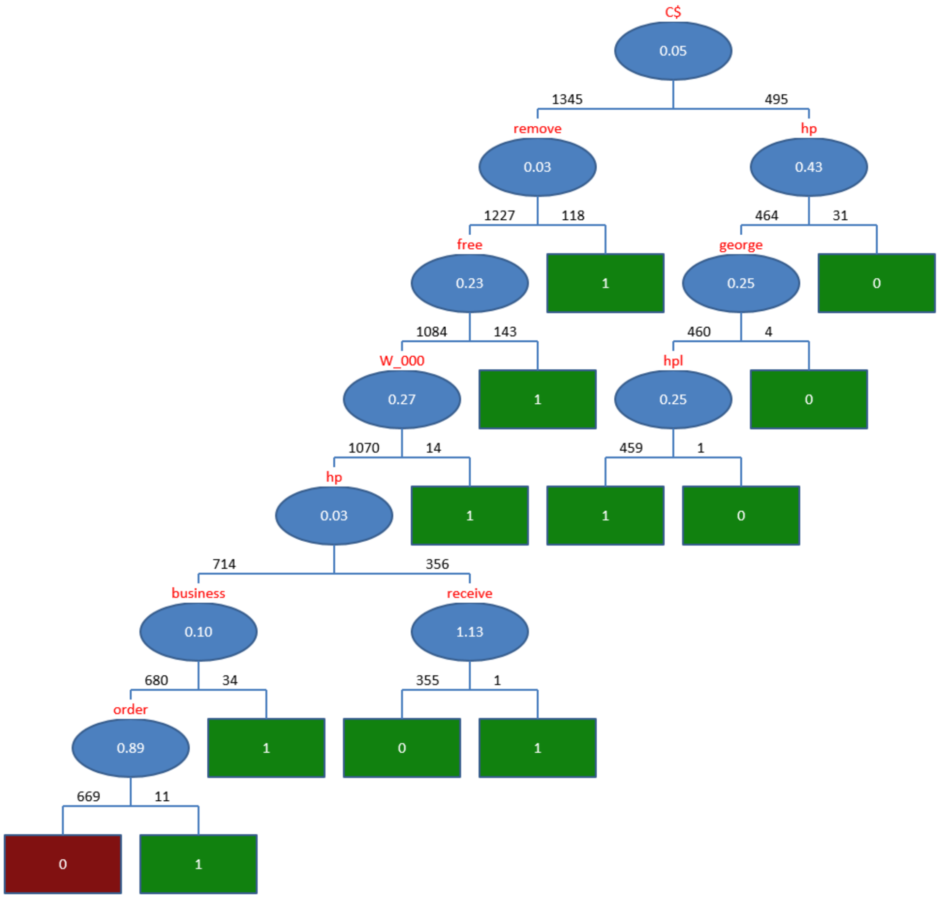
Performance Evaluation:

Under cutoff value of 0.45, we get lowest overall % Error and highest AUC which is close to 1.

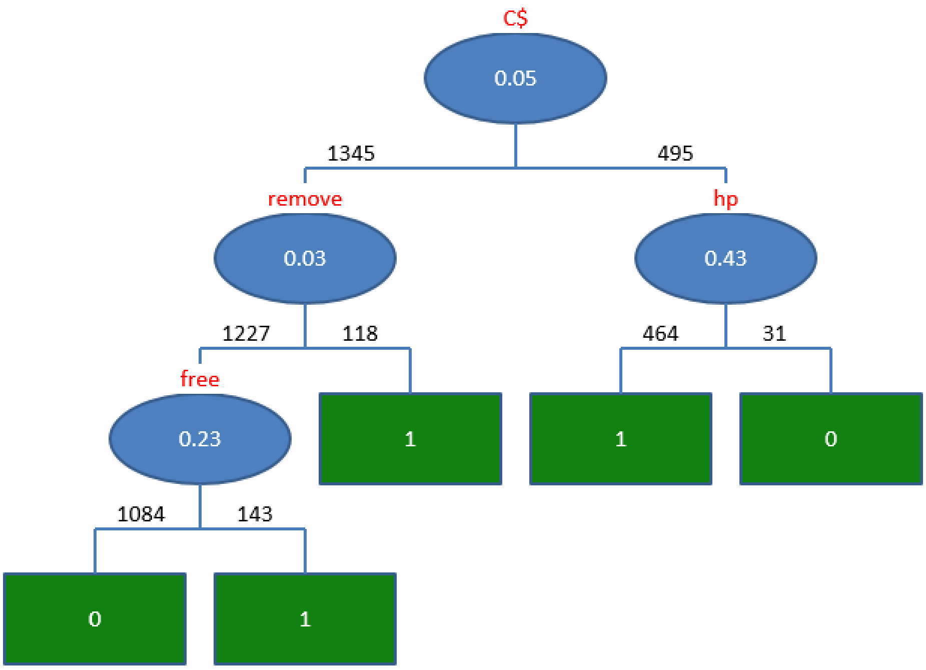


Cart:

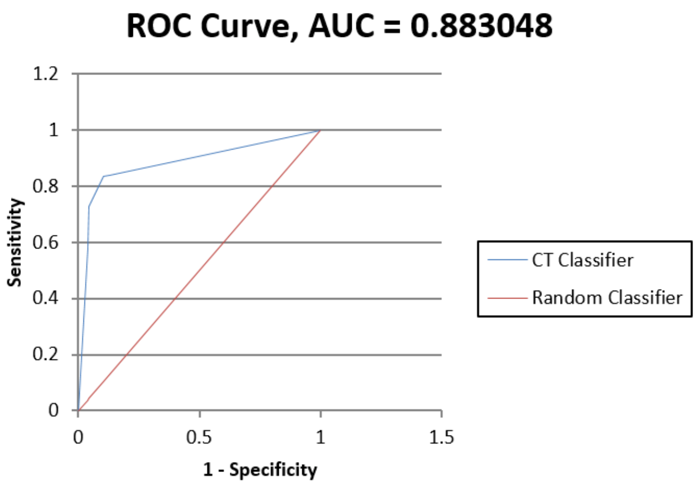
Minimum Error Tree:



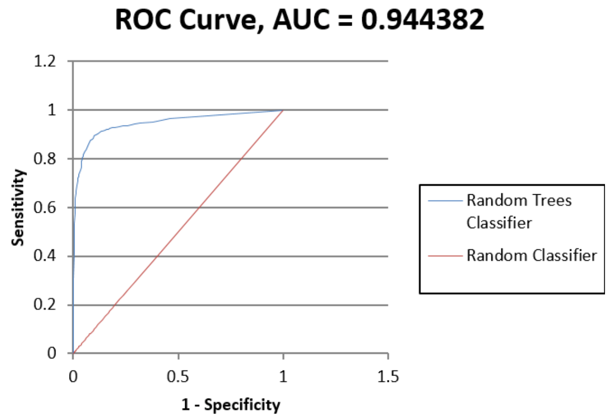
Best Pruned Tree:



Performance Evaluation:



Random Forest:



Performance Evaluation Comparison:

	Logistic	Cart	Random Forest
Overall % Error	9.89	12.35	10
AUC	0.9536	0.8830	0.9443
Cutoff Value	0.45	0.75	0.5

Conclusion:

Comparing the three value of each model listed in the table above, Logistic has the lowest Overall % Error and Highest AUC value, so that we think logistic regression is probably the best model to classify if an email is spam or not under certain selection of input variables.

Blog feedback problem

About the data

Every dataset contains 281 columns. The rows represents different blog posts.
As mentioned in the question, we have to consider only the blog posts which were published 72 hours before the basetime.

Columns 263-269 are the binary indicator for the weekday of the basetime and columns 272-276 represent the binary indicator for the weekday of the date of publication.
So we make combinations of basetime and date of publication such that each combination has different of 3 days that is 72 hours.
The combinations of columns are as follows:
Both date of publication and basetime columns given in the data are weekdays.

Date of Publication of Blog Post	Base Time
Column 275 [Saturday]	Column 263 [Monday]
Column 276 [Sunday]	Column 264 [Tuesday]
Column 270 [Monday]	Column 265 [Wednesday]
Column 271 [Tuesday]	Column 266 [Thursday]
Column 272 [Wednesday]	Column 267 [Friday]
Column 273 [Thursday]	Column 268 [Saturday]
Column 274 [Friday]	Column 269 [Sunday]

- We make a separate model for each combination of *Date of Publication* & *Base Time*.
- There are Test datasets for each day of February 2012 and March 2012.
- Fit the model and predict the values for the corresponding Test dataset.
- Training dataset will be trained for each of the above combinations.
- For Each Combination we take only those blogs which have Date of publication-Basetime=72 hours

For example: Model trained for combination 270-265 will be applied to predict values on Test dataset with date 2012/03/28 which has a basetime of Wednesday. (All values of column 265 are 1) .

Prediction Model

Refer to the Example below:
 # for Wednesday dataset 2012/3/28-Wednesday
Command 1: testdatawednesday1<-subset(test.dataset.wednesday1, var265==1 & var270==1)
Command 2: testdatawednesday1
Command 3: data4<-subset(train.dataset, var270==1 & var265==1)

We built multiple linear regression model and tested it for various days using various input parameters. The combinations of input parameters are as follows:

- **Basic Features**
- **Basic+Weekday**

- **Basic+Parent**
- **Basic+Textual**
- **Bagging**

The best one we found was Regression with Bagging

```
# regression with bagging
length_divisor<-2
iterations<-100
predictions<-foreach(m=1:iterations,.combine=cbind) %do% {
 training_positions <- sample(nrow(data2), size=floor((nrow(data2)/length_divisor)))
  train_pos<-1:nrow(data2) %in% training_positions
  lm_fit<-lm(var281 ~
var52+var57+var53+var58+var54+var59+var55+var60+var277+var278+var279+var280,data
=data2[train_pos,])
  predict(lm_fit,newdata=testdatafriday)
}
predictions<-rowMeans(predictions)
error<-sqrt((sum((testdatafriday$var281-predictions)^2))/nrow(testdatafriday))
RMSE.rtree =sqrt(mean((predictions - testdatafriday$var281)^2))

summary(lm_fit)
```

Console C:/Users/aasth/AppData/Local/Temp/Temp1_Group_6_Random_Forest.zip/Group_6_Random_Forest/

```
> plot(predictions)
> summary(lm_fit)

Call:
lm(formula = var281 ~ var52 + var57 + var53 + var58 + var54 +
    var59 + var55 + var60 + var277 + var278 + var279 + var280,
    data = data2[train_pos, ])

Residuals:
     Min       1Q   Median       3Q      Max
-116.835   -0.243    0.333    0.381  192.087

Coefficients: (3 not defined because of singularities)
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.33340    0.31854  -1.047 0.295437
var52        0.30519    0.01710  17.852  < 2e-16 ***
var57       -3.40116    0.99327  -3.424 0.000633 ***
var53        0.06908    0.01804   3.830 0.000134 ***
var58       -2.08997    1.06006  -1.972 0.048847 *
var54       -0.07853    0.01897  -4.139 3.69e-05 ***
var59        3.08455    1.04501   2.952 0.003210 **
var55             NA         NA      NA       NA
var60             NA         NA      NA       NA
var277      -0.02201    0.29858  -0.074 0.941248
var278            NA         NA      NA       NA
var279      -0.01888    0.24085  -0.078 0.937540
var280       0.06278    0.70757   0.089 0.929308
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 11.34 on 1475 degrees of freedom
Multiple R-squared:  0.6148,    Adjusted R-squared:  0.6125
F-statistic: 261.6 on 9 and 1475 DF,  p-value: < 2.2e-16
```

RMSE Error came out to be 8.12

Experiment 2: CART Regression

We built CART Regression model and tested on the same dataset.

Refer to the example below:
trainset1<-subset(train.dataset, var267==1 & var272==1)
frmla =var281 ~ var52+var57+var53+var58+var54+var59+var55+var60
fit = rpart(frmla, method="anova", data=trainset1)
printcp(fit) # display the results
plotcp(fit) # visualize cross-validation results
summary(fit)
pred=predict(fit,testdatafriday)
summary(pred)
RMSE.rtree =sqrt(mean((pred - testdatafriday$var281)^2))
RMSE.rtree

MAE.rtree <- mean(abs(pred-testdatafriday$var281))
MAE.rtree
<u>RMSE Error came out to be 8.10</u>

Experiment 3: Random Forest
#randomforest
fitted <- randomForest(var281 ~ var52+var57+var53+var58+var54+var59+var55+var60, trainset1)
predicted= predict(fit,testdatafriday)

RMSE Error came out to be 17.32

Evaluationg Model's Performance:

RMSE Errors Table:

	Bagging(Regression)	Cart Decision Tree	Random Forest
Saturday 2012/03/31	28.70281	22.179	22.179
Wednesday 2012/03/28	8.10	8.12	17.32
Wednesday 2012/02/29	13.711	13.73	15.62
Friday 2012/3/30	11.82	12.32	9.32

We can clearly see that Regression used with Bagging gives the minimum RMSE Errors. Also the R^2 and adjusted R^2 values for Regression Bagging are high which means the predicted values are close to the real ones.
The T values for Bagging shows the used features as the most significant ones in this model

Recommendation:

After trying these 3 models on all the datasets for different days of the week, we found that the model with best performance was Regression with bagging.
Reason: Regression with bagging performs best because it takes 100 randomly selected subsets of the basic features and predicted values for all of these 100 subsets of features.

The bagging model takes the average of these 100 prediction values. Hence bagging model accounts for the maximum variance of information included for prediction.
We found the RMSE Error values to be lowest for the regression model used with Bagging. *Hence we highly recommend this model.*