Homework3-INF552-Rahul Ethiraj

1.) The LASSO and Boosting for Regression

a) Importing data

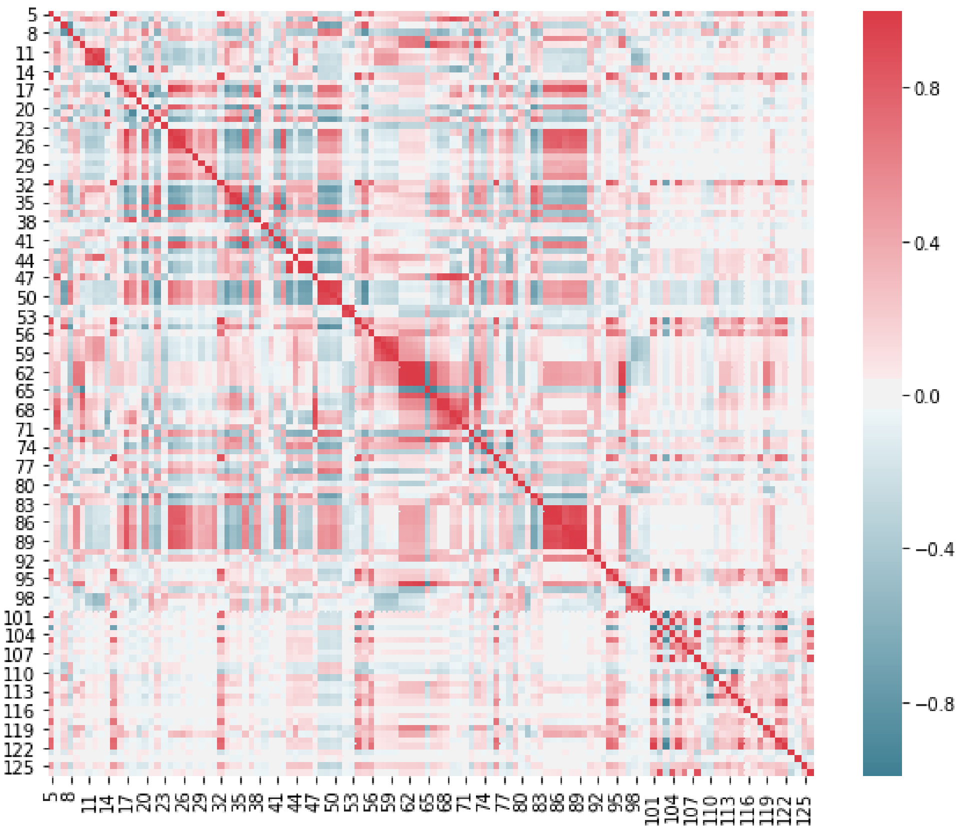
b) Train_test split data, imputation :

x_train_df
x_test_df

	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	120	121	122	123	124	125	126	127
0	0.19	0.33	0.02	0.90	0.12	0.17	0.34	0.47	0.29	0.32	0.20	1.0	0.37	0.72	0.34	0.60	0.20	0.06	0.04	0.90	0.5	0.32	0.14	0.20
1	0.00	0.16	0.12	0.74	0.45	0.07	0.26	0.59	0.35	0.27	0.02	1.0	0.31	0.72	0.11	0.45	0.45	0.08	0.03	0.75	0.5	0.00	0.15	0.67
2	0.00	0.42	0.49	0.56	0.17	0.04	0.39	0.47	0.28	0.32	0.00	0.0	0.30	0.58	0.19	0.39	0.02	0.08	0.03	0.75	0.5	0.00	0.15	0.43
3	0.04	0.77	1.00	0.08	0.12	0.10	0.51	0.50	0.34	0.21	0.06	1.0	0.58	0.89	0.21	0.43	0.28	0.08	0.03	0.75	0.5	0.00	0.15	0.12
4	0.01	0.55	0.02	0.95	0.09	0.05	0.38	0.38	0.23	0.36	0.02	0.9	0.50	0.72	0.16	0.68	0.02	0.08	0.03	0.75	0.5	0.00	0.15	0.03

c)

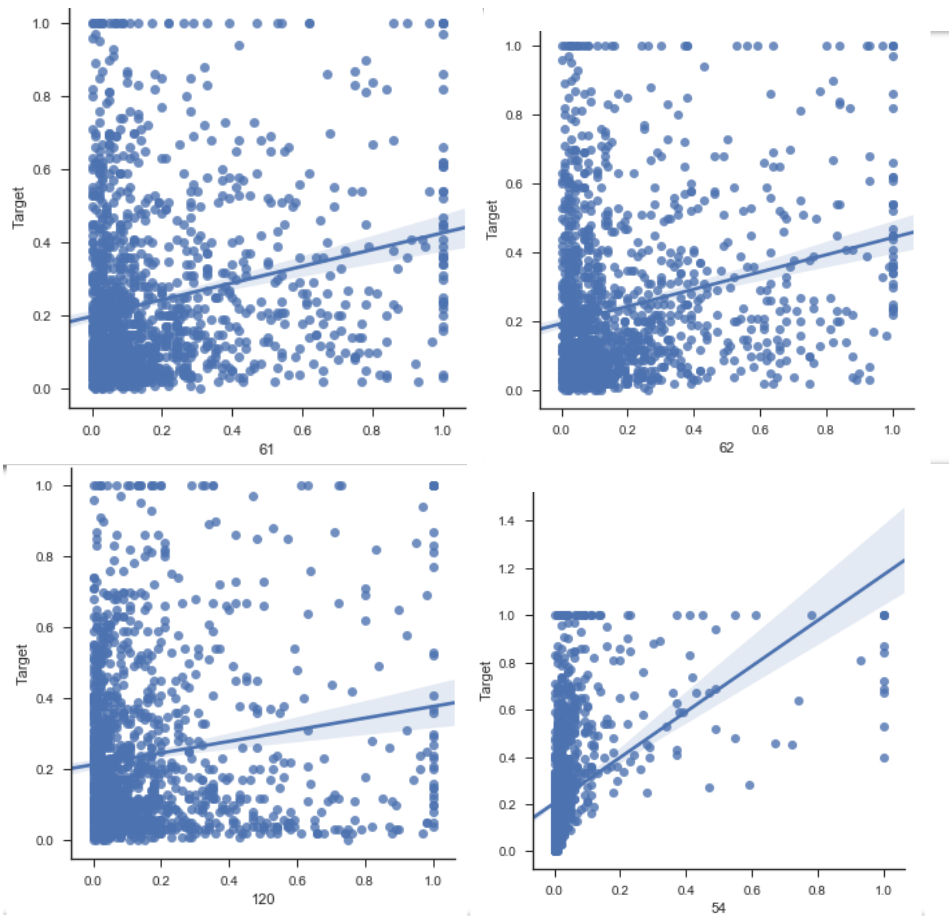
(i) Correlation matrix:

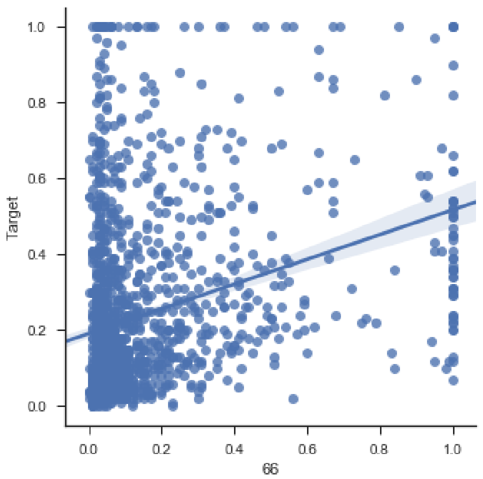


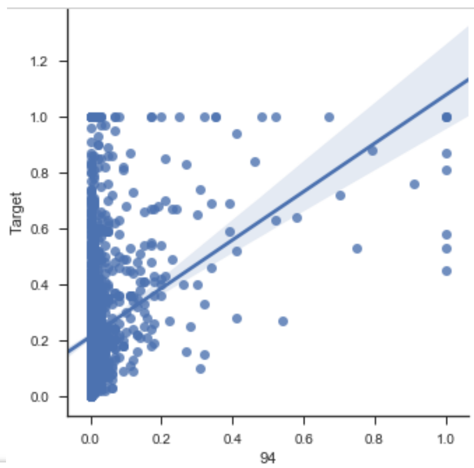
(d) Coefficient of variation CV

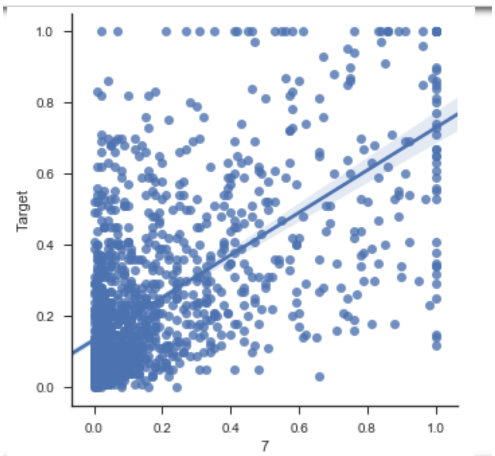
	Coefficient of variation
0	0.279637
1	0.057841
2	0.357587
3	0.079015
4	0.283898
5	0.375306
6	0.056777
7	0.041733
8	0.082446
9	0.075875
10	0.256736
11	0.284167
110	0.054138
111	0.011647
112	0.031785
113	0.183676
114	0.177134
115	0.324496
116	0.088847
117	0.091532
118	0.010325
119	0.054546
120	0.614103
121	0.029275

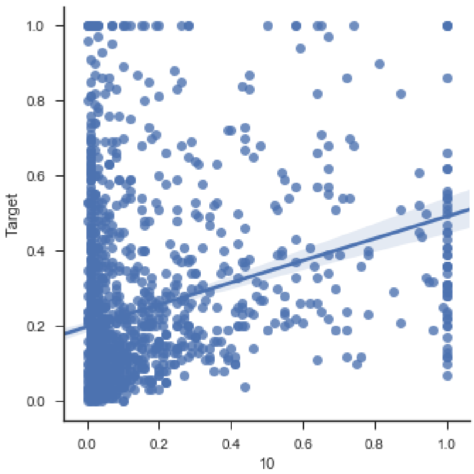
(e) Scatter plots

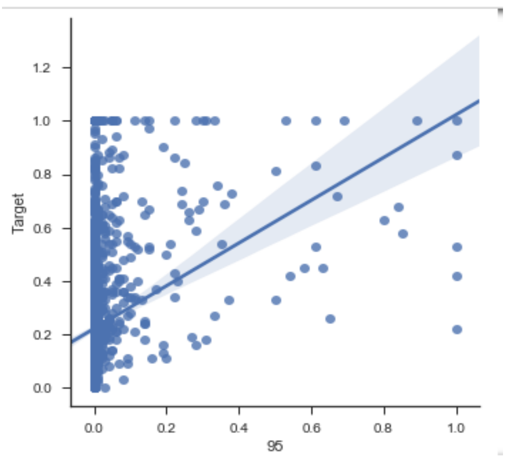


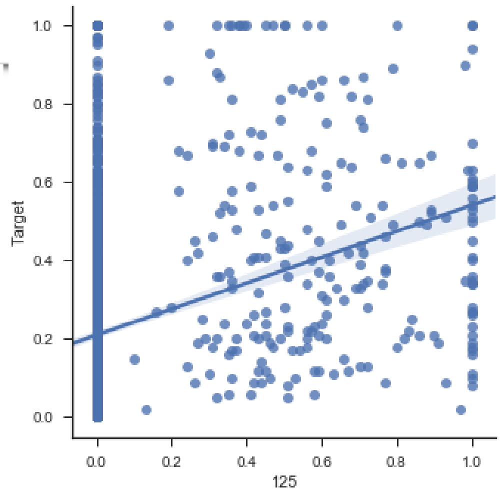

Target
1.0
0.8
0.6
0.4
0.2
0.0
0.0 0.2 0.4 0.6 0.8 1.0
66

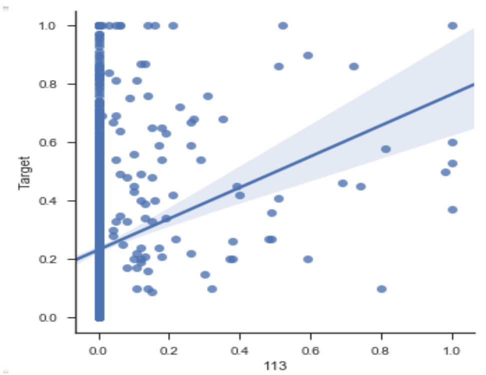

Target
1.2
1.0
0.8
0.6
0.4
0.2
0.0
0.0 0.2 0.4 0.6 0.8 1.0
94


Target
1.0
0.8
0.6
0.4
0.2
0.0
0.0 0.2 0.4 0.6 0.8 1.0
7


Target
1.0
0.8
0.6
0.4
0.2
0.0
0.0 0.2 0.4 0.6 0.8 1.0
10


Target
1.2
1.0
0.8
0.6
0.4
0.2
0.0
0.0 0.2 0.4 0.6 0.8 1.0
95


Target
1.0
0.8
0.6
0.4
0.2
0.0
0.0 0.2 0.4 0.6 0.8 1.0
125

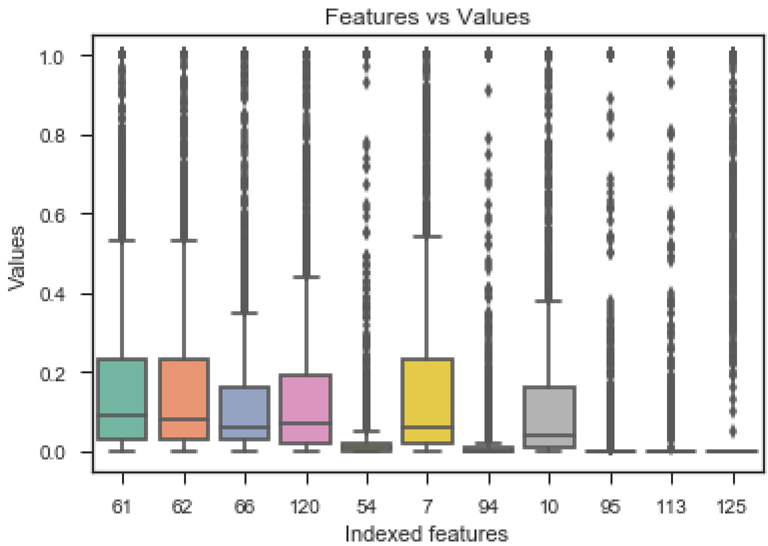

Target
1.0
0.8
0.6
0.4
0.2
0.0
0.0 0.2 0.4 0.6 0.8 1.0
113

Linear regression line has been fitted on all the scatter plots, which gives us the correlation.

Scatter plots are similar to line graphs in that they use horizontal and vertical axes to plot data points. Scatter plots show how much one variable is affected by another. The relationship between two variables is called their correlation.

Here, we see strong positive correlation between many of the features.

(e) Box plots



(f) Linear model using least squares

Mean squared test error : 0.01797697257164362

(g) Ridge regression model¶

Mean squared test error : 0.017630963071701356

(h) LASSO model

Unnormalized LassoCV :
====================
Mean squared test error : 0.017562937954615554
Total number of selected features : 45
Feature list by column numbers : [2, 7, 11, 13, 14, 15, 17, 18, 22, 23, 24, 25, 26, 28, 33, 38, 44, 45, 46, 48, 50, 59, 67, 68, 69, 71, 72, 74, 75, 76, 78, 82, 85, 86, 88, 90, 91, 94, 101, 102, 114, 115, 119, 120, 121]

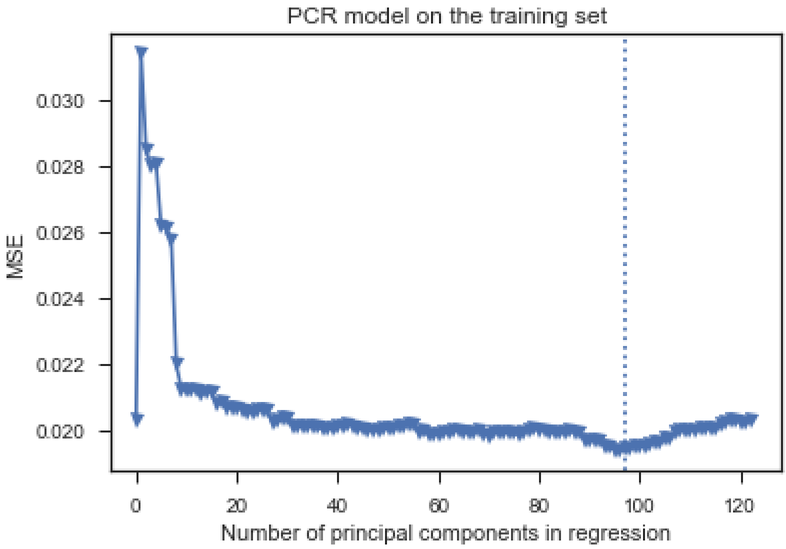
Normalized LassoCV :
====================
Mean squared test error : 0.017424186398454396
Total number of selected features : 32

Feature list by column numbers : [2, 3, 7, 11, 15, 17, 18, 23, 24, 25, 38, 44, 45, 48, 50, 68, 69, 71, 72, 74, 75, 86, 88, 90, 91, 94, 99, 102, 104, 108, 119, 121]

The test error for both unnormalized and normalized is almost same, but the number of features is greatl y reduced in the nomalized version of LassoCV.

(i) PCR model

```
Number of principal components in regression :  97
Mean squared test error                       :  0.017424186398454396
```



(j) L1 penalized gradient boosting tree

```
Fitting 5 folds for each of 20 candidates, totalling 100 fits
[Parallel(n_jobs=1)]: Done 100 out of 100 | elapsed:  1.1min finished
GridSearchCV.best_params_ :
=========================
{'reg_alpha': 0.0001}

GridSearchCV.grid_scores_ :
=========================
[mean: -0.02061, std: 0.00299, params: {'reg_alpha': 0.0001}, mean: -0.02061, std: 0.00299, params: {'reg_alpha': 0.52641052631
57894}, mean: -0.02061, std: 0.00299, params: {'reg_alpha': 1.052721052631579}, mean: -0.02061, std: 0.00299, params: {'reg_alp
ha': 1.5790315789473683}, mean: -0.02061, std: 0.00299, params: {'reg_alpha': 2.105342105263158}, mean: -0.02061, std: 0.00299,
params: {'reg_alpha': 2.6316526315789477}, mean: -0.02061, std: 0.00299, params: {'reg_alpha': 3.157963157894737}, mean: -0.020
61, std: 0.00299, params: {'reg_alpha': 3.684273684210526}, mean: -0.02061, std: 0.00299, params: {'reg_alpha': 4.2105842105263
15}, mean: -0.02061, std: 0.00299, params: {'reg_alpha': 4.736894736842105}, mean: -0.02061, std: 0.00299, params: {'reg_alph
a': 5.263205263157895}, mean: -0.02061, std: 0.00299, params: {'reg_alpha': 5.7895157894736835}, mean: -0.02061, std: 0.00299,
params: {'reg_alpha': 6.315826315789473}, mean: -0.02061, std: 0.00299, params: {'reg_alpha': 6.842136842105263}, mean: -0.0206
1, std: 0.00299, params: {'reg_alpha': 7.368447368421052}, mean: -0.02061, std: 0.00299, params: {'reg_alpha': 7.89475789473684
1}, mean: -0.02061, std: 0.00299, params: {'reg_alpha': 8.421068421052631}, mean: -0.02061, std: 0.00299, params: {'reg_alpha':
8.94737894736842}, mean: -0.02061, std: 0.00299, params: {'reg_alpha': 9.47368947368421}, mean: -0.02061, std: 0.00299, params:
{'reg_alpha': 10.0}]

GridSearchCV.best_score_ :
========================
-0.02060736583899276
```

2. Tree based methods

(b) Data preparation

i.) Train_test split data, imputation

The techniques used for dealing with missing values are known as imputation techniques. Examples inclu de mean, median, mode, ffill, bfill, pad etc.

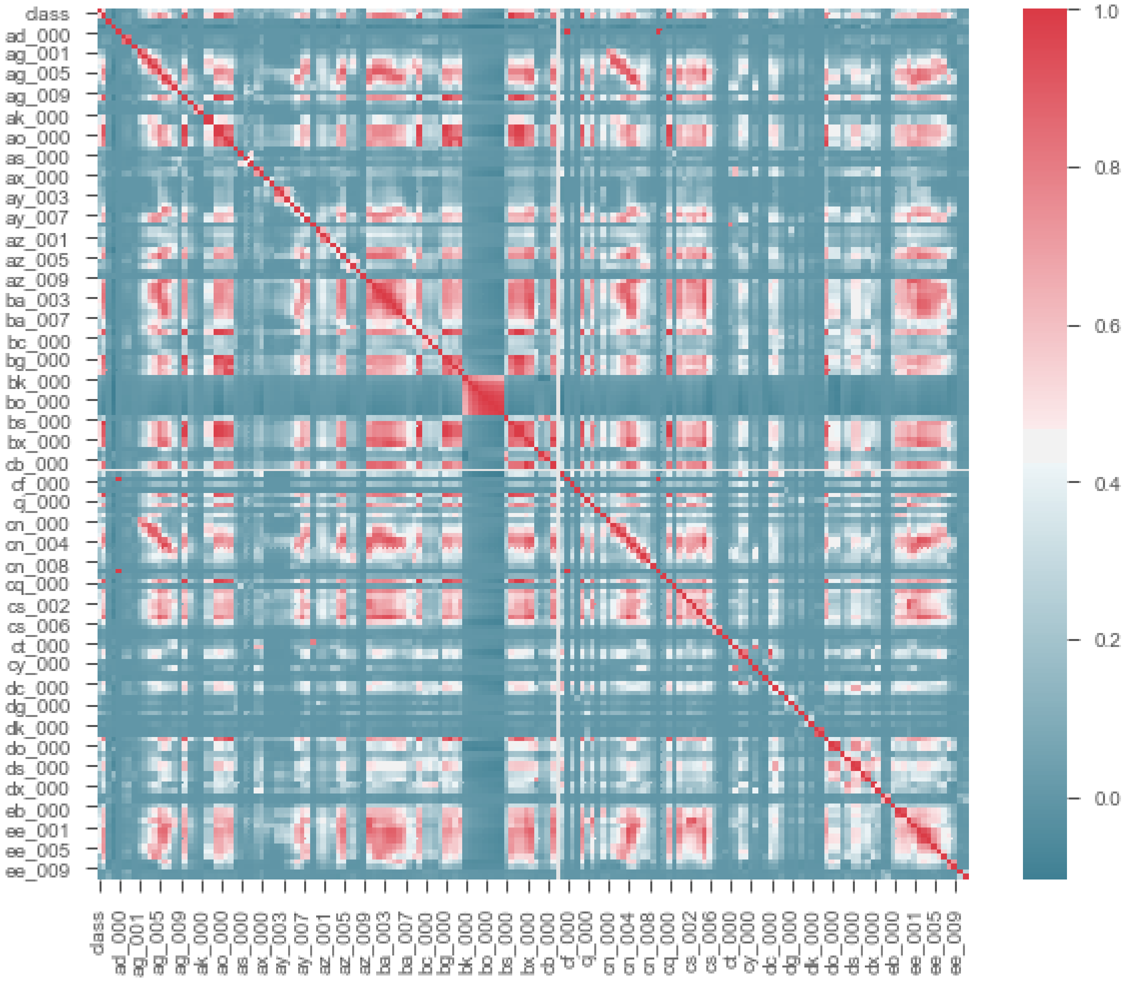
	class	aa_000	ab_000	ac_000	ad_000	ae_000	af_000	ag_000	ag_001	ag_002	ag_003	ag_004	ag_005	ag_006	ag_007	ag_008
0	0	76698	0.0	2.130706e+09	280.0	0.0	0.0	0.0	0.0	0.0	0.0	37250.0	1432864.0	3664156.0	1007684.0	25896.0
1	0	33058	0.0	0.000000e+00	126.0	0.0	0.0	0.0	0.0	0.0	0.0	18254.0	653294.0	1720800.0	516724.0	31642.0
2	0	41040	0.0	2.280000e+02	100.0	0.0	0.0	0.0	0.0	0.0	0.0	1648.0	370592.0	1883374.0	292936.0	12016.0
3	0	12	0.0	7.000000e+01	66.0	0.0	10.0	0.0	0.0	0.0	318.0	2212.0	3232.0	1872.0	0.0	0.0
4	0	60874	0.0	1.368000e+03	458.0	0.0	0.0	0.0	0.0	0.0	0.0	43752.0	1966618.0	1800340.0	131646.0	4588.0

	class	aa_000	ab_000	ac_000	ad_000	ae_000	af_000	ag_000	ag_001	ag_002	ag_003	ag_004	ag_005	ag_006	ag_007	ag_008
0	0	60	0.0	20.0	12.0	0.0	0.0	0.0	0.0	0.0	2682.0	4736.0	3862.0	1846.0	0.0	0.0
1	0	82	0.0	68.0	40.0	0.0	0.0	0.0	0.0	0.0	0.0	748.0	12594.0	3636.0	0.0	0.0
2	0	66002	2.0	212.0	112.0	0.0	0.0	0.0	0.0	0.0	199486.0	1358536.0	1952422.0	452706.0	25130.0	520.0
3	0	59816	0.0	1010.0	936.0	0.0	0.0	0.0	0.0	0.0	0.0	123922.0	984314.0	1680050.0	1135268.0	92606.0
4	0	1814	0.0	156.0	140.0	0.0	0.0	0.0	0.0	0.0	0.0	72.0	17926.0	82834.0	3114.0	0.0

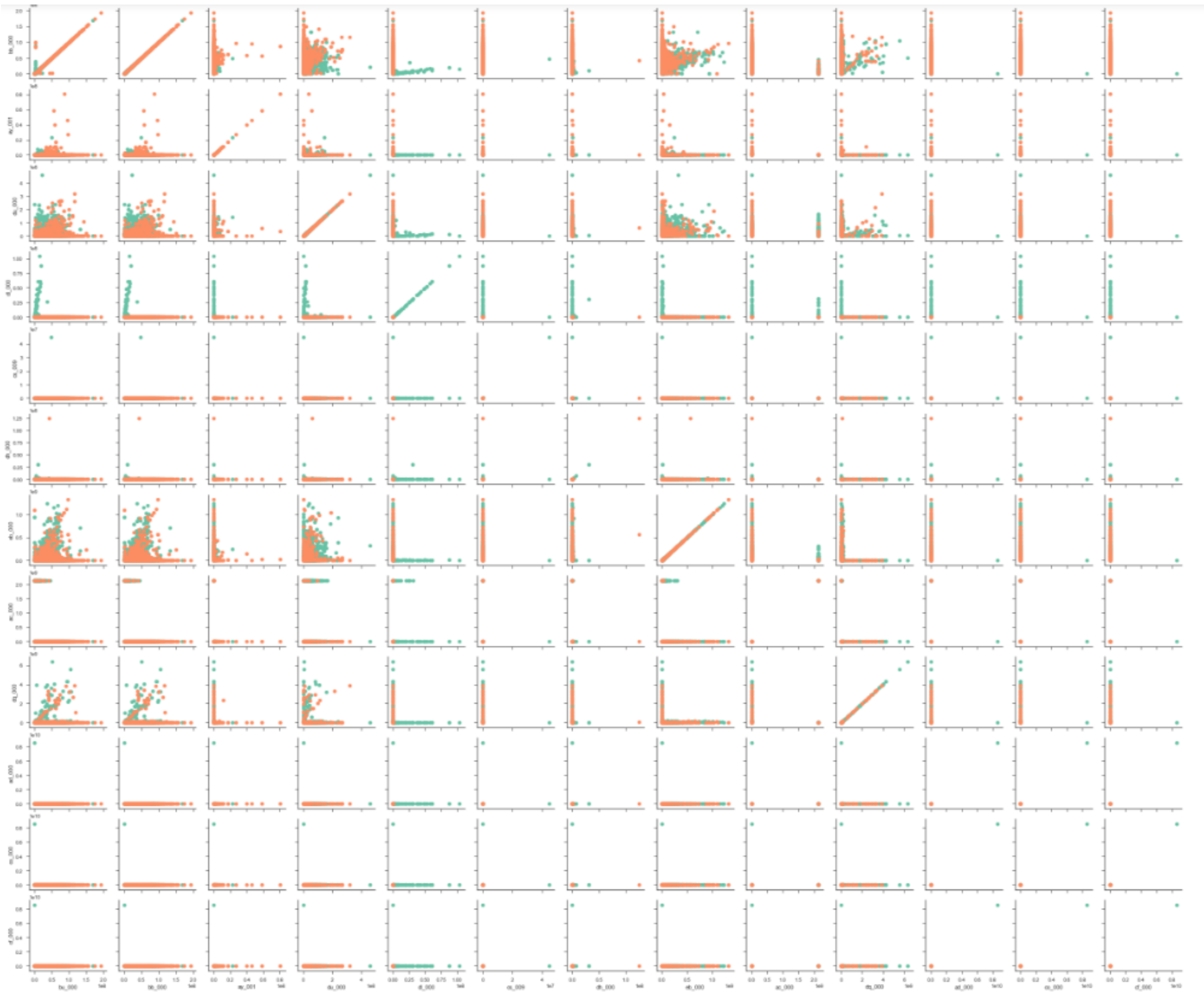
ii.) Coefficient of variation CV

	Coefficient of variation
0	3.564400e+05
1	1.752025e+01
2	1.794508e+09
3	8.562354e+09
4	3.827201e+03
5	3.999153e+03
6	1.892144e+06
7	1.198791e+06
8	2.625784e+06
9	6.550555e+06
10	1.290053e+07
11	9.595340e+06
158	7.931463e+06
159	8.387430e+06
160	2.980746e+06
161	1.390363e+06
162	3.046405e+06
163	3.174001e+06
164	3.423338e+06
165	8.615248e+06
166	1.457507e+06
167	2.687152e+05
168	2.107251e+02
169	3.665343e+02

(iii) Correlation matrix



(iv) Scatter plots

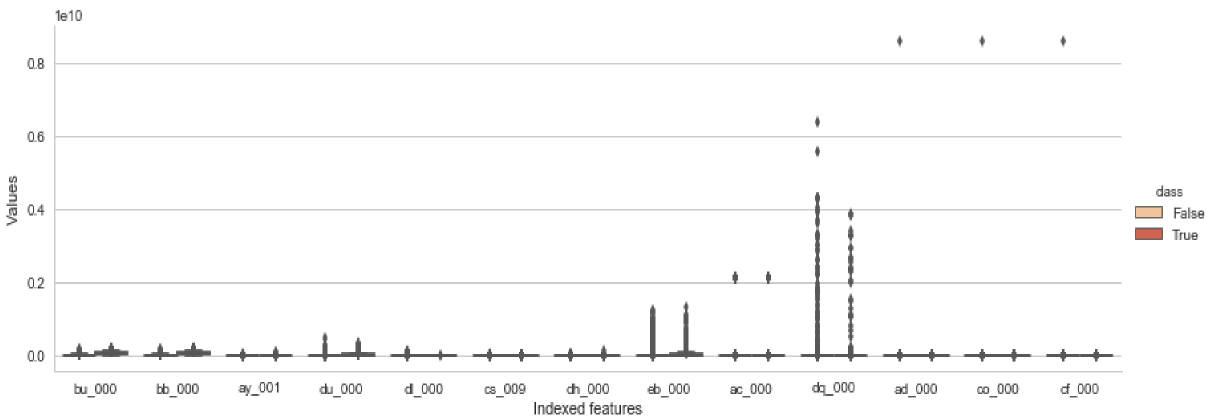


Linear regression line can be fitted on all the scatter plots, which gives us the correlation.

Scatter plots are similar to line graphs in that they use horizontal and vertical axes to plot data points. Scatter plots show how much one variable is affected by another. The relationship between two variables is called their correlation .

Here, we see strong positive correlation between few of the features.

(iv) Box plots



(v) Imbalance

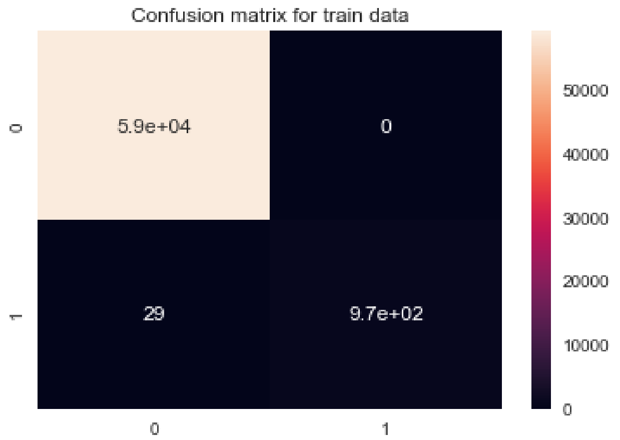
```
Training set :
=============
Number of 1s : 1000
Number of 0s : 59000

Test set :
=========
Number of 1s : 375
Number of 0s : 15625
```

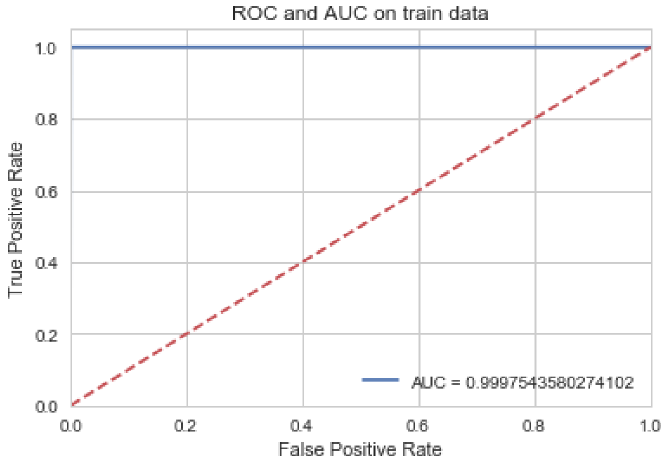
```
Yes, the data set is heavily imbalanced, as number of 0s are more than number of 1s.
```

(c) Random forest for training data with imbalance

Out-of-bag error estimate : 0.008633333333333382
Training error : 0.0004833333333333334



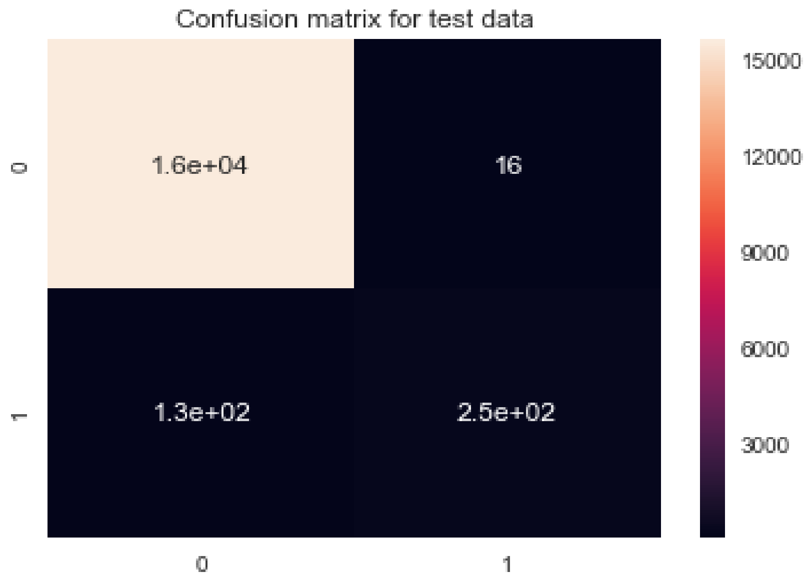

```
AUC :  0.9997543580274102
```



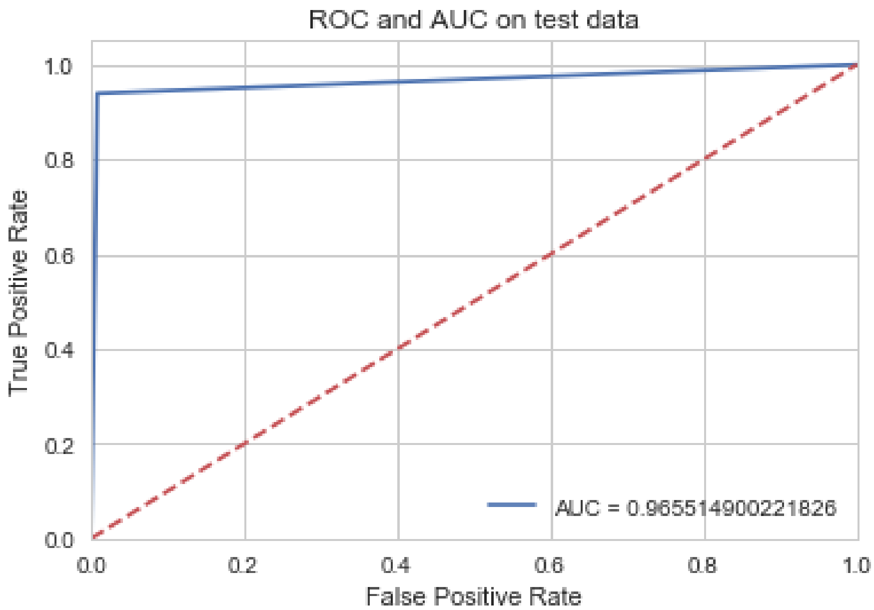

```
971
Misclassification rate :  0.0004833333333333334
```

Random forest for test data with imbalance:

Out-of-bag error estimate : 0.009383333333333299
Test error : 0.009



AUC : 0.965514900221826



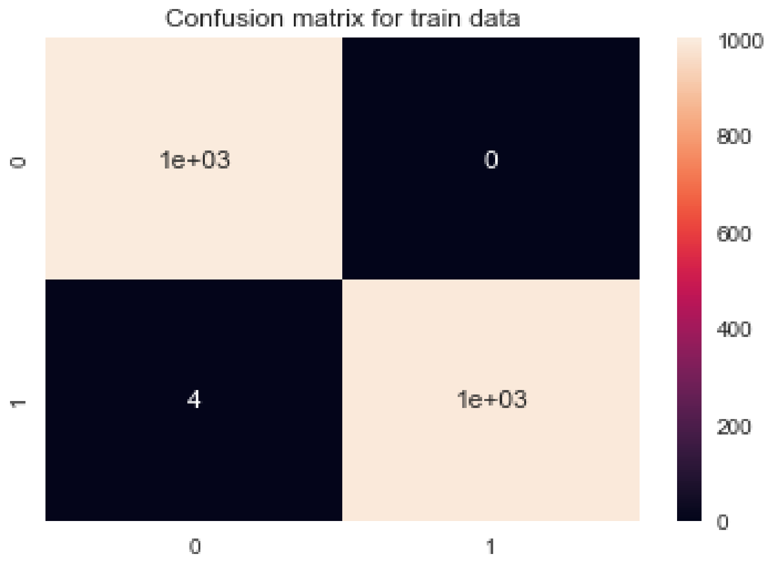
Misclassification rate : 0.009

(d) Random forest for train data with no imbalance

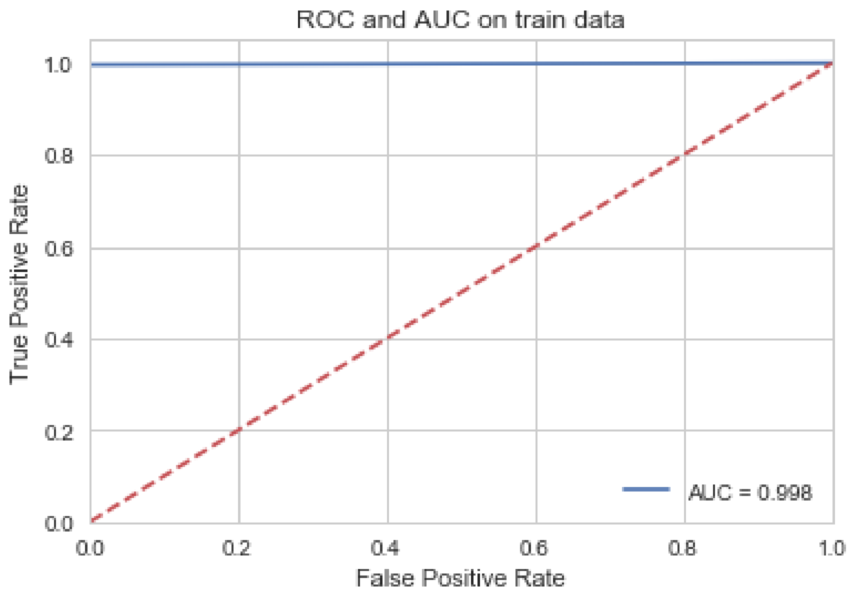
Class imbalance can be addressed in random forests by balancing the classes, resampling, downsampling and upsampling.

Out-of-bag error estimate : 0.06499999999999995

Training error : 0.002




```
AUC :  0.998
```

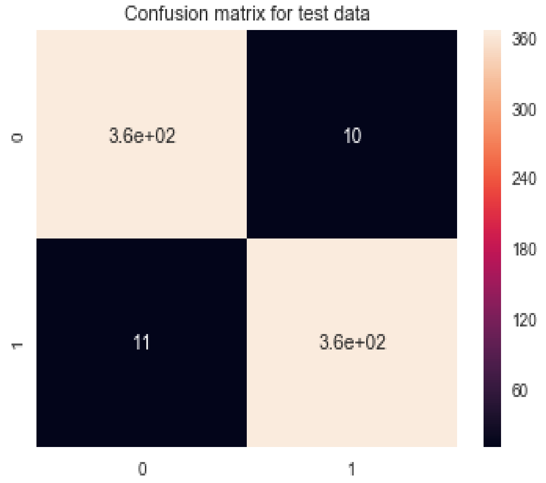



```
Misclassification rate :  6.666666666666667e-05
```

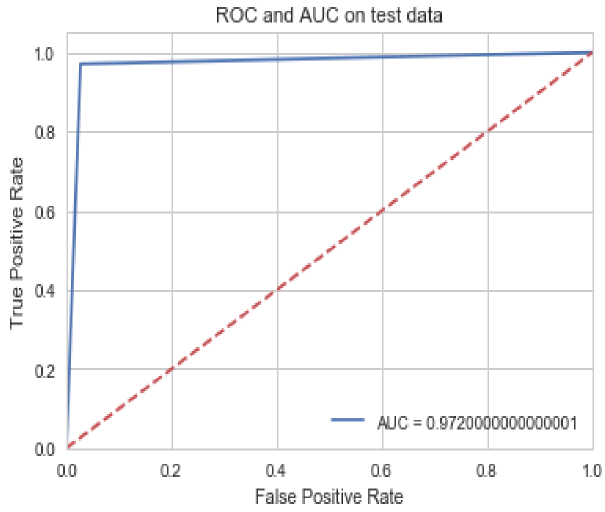
(d) Random forest for test data with no imbalance

Out-of-bag error estimate : 0.06699999999999995

Training error : 0.028




```
AUC :  0.9720000000000001
```




```
Misclassification rate :  0.0013125
```

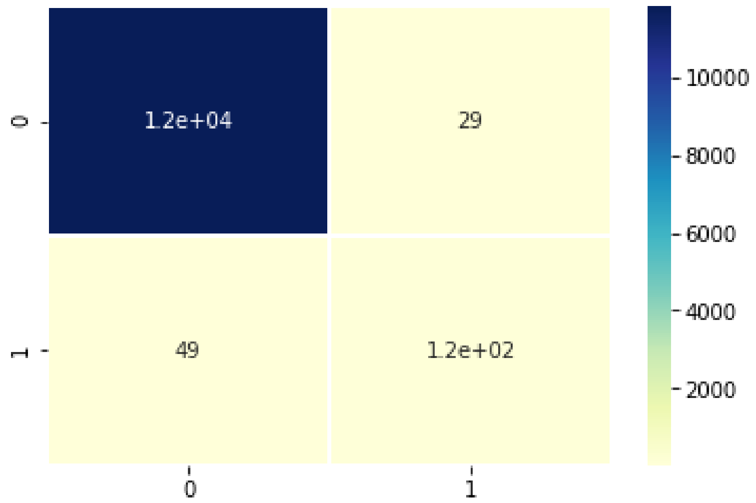
```
Resampling helps the model to reduce the imbalance and the error is reduced as compared to 2(c).
```

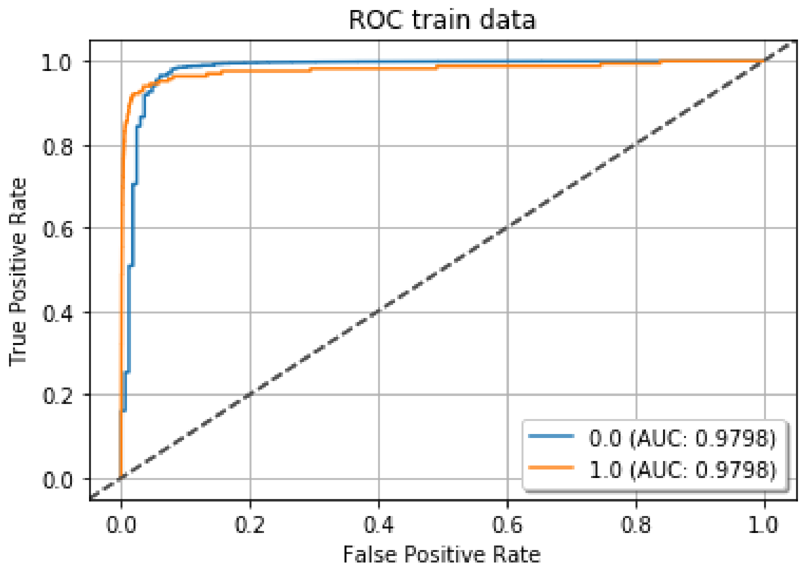
(e) Model Trees for training set

```
Summary  :
=======

Correctly Classified Instances       11922               99.35   %
Incorrectly Classified Instances        78                0.65   %
Kappa statistic                          0.7467
Mean absolute error                      0.0085
Root mean squared error                  0.0719
Relative absolute error                 31.0536 %
Root relative squared error             61.592  %
Total Number of Instances            12000


Misclassification rate for training :  0.0065
AUC                   :  0.9798301707760567
weightedAreaUnderROC :  0.9798301707760568
```



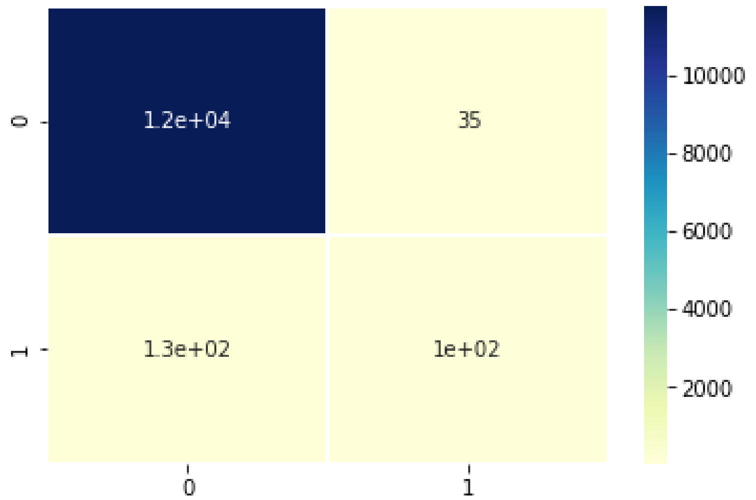


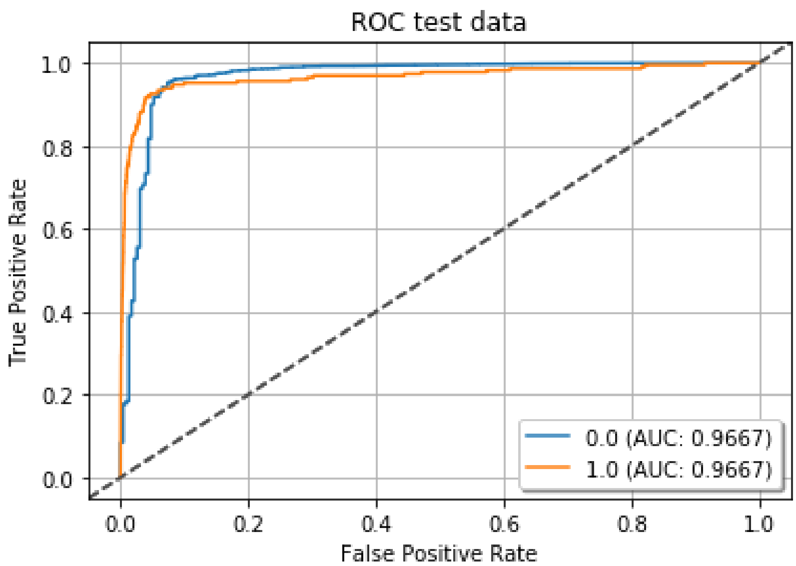
(e) Model Trees for test data

```
Summary  :
=======

Correctly Classified Instances       11836              98.6333 %
Incorrectly Classified Instances       164               1.3667 %
Kappa statistic                          0.5454
Mean absolute error                      0.0148
Root mean squared error                  0.1041
Relative absolute error                 45.3601 %
Root relative squared error             75.8662 %
Total Number of Instances            12000


Misclassification rate for testing :  0.013666666666666667
AUC                    :  0.9666645118392375
weightedAreaUnderROC :  0.9666645118392375
```



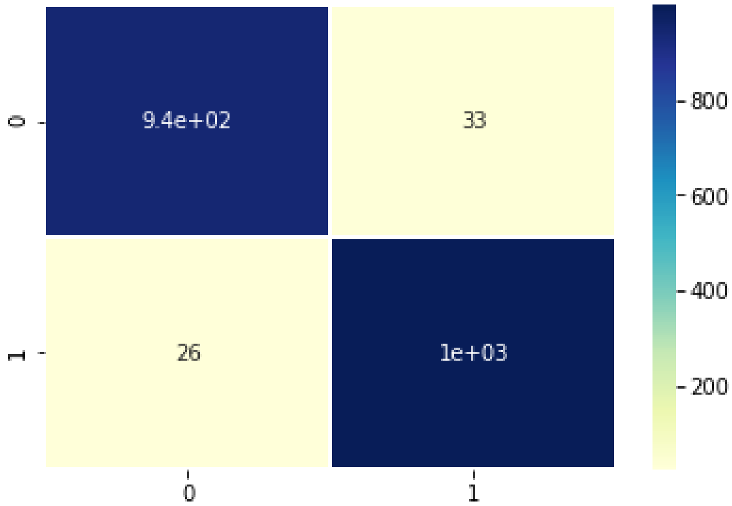


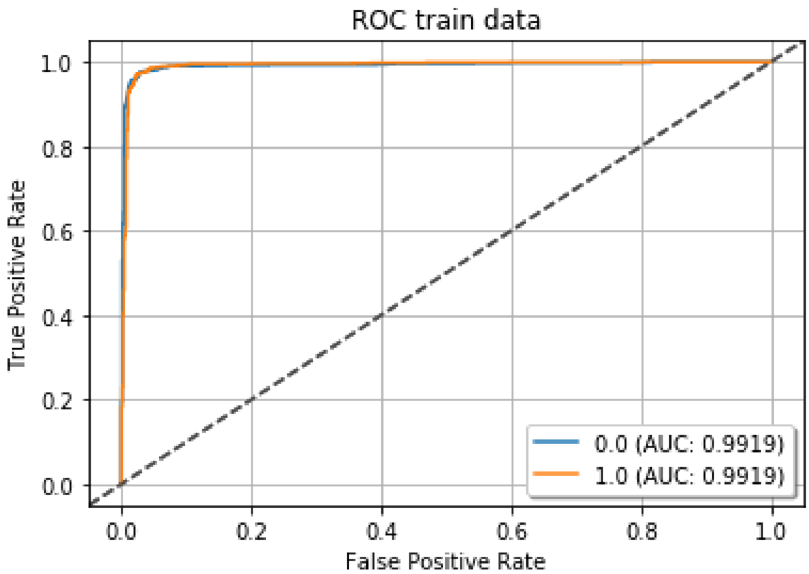
(f) SMOTE Model Trees for train data

```
Summary  :
=======

Correctly Classified Instances        1941               97.05    %
Incorrectly Classified Instances        59                2.95    %
Kappa statistic                          0.941
Mean absolute error                      0.043
Root mean squared error                  0.15
Relative absolute error                  8.6127 %
Root relative squared error             29.9988 %
Total Number of Instances             2000


Misclassification rate for training :  0.059
AUC                    :  0.9919218469838627
weightedAreaUnderROC :  0.9919218469838627
```



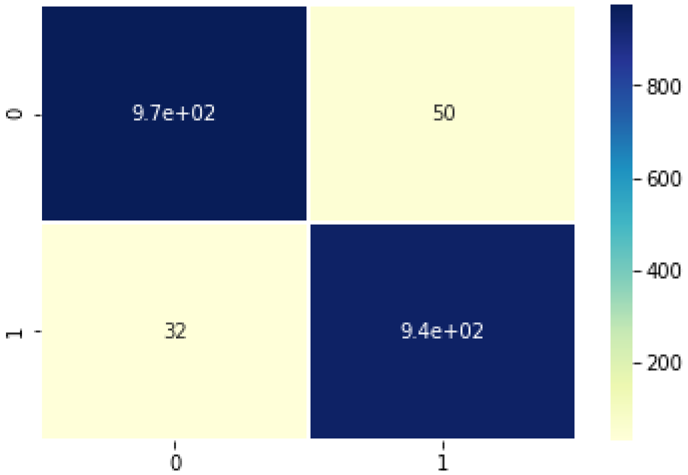


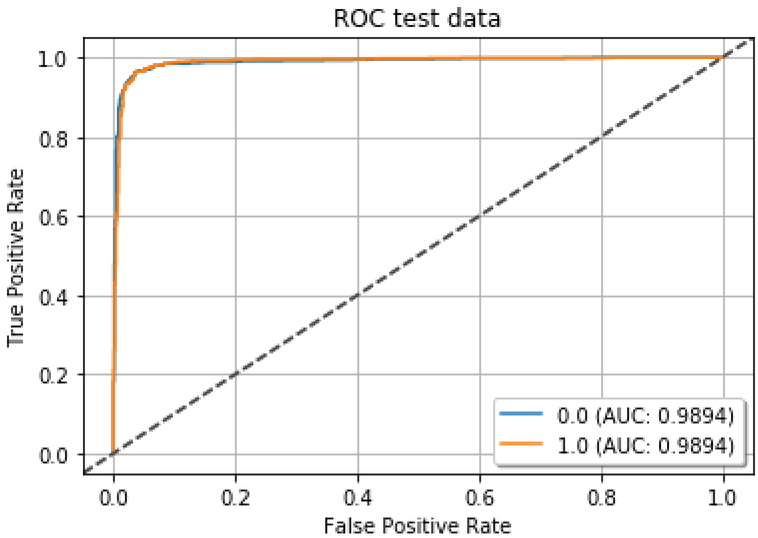
(f) SMOTE Model Trees for test data

```
Summary  :
=======

Correctly Classified Instances        1918               95.9    %
Incorrectly Classified Instances        82                4.1    %
Kappa statistic                          0.918
Mean absolute error                      0.0532
Root mean squared error                  0.1763
Relative absolute error                 10.6433 %
Root relative squared error             35.2264 %
Total Number of Instances             2000


Misclassification rate for testing :  0.082
AUC                              :  0.9894494187425148
weightedAreaUnderROC :  0.9894494187425148
```





Clearly, using SMOTE has proven beneficial as the results are surprising. AUC has increased when compared to uncompensated case.

3.) ISLR 6.8.3, 4.) ISLR 6.8.5

Question 3: ISLR 6.8.3 :

a.) iv) Steadily decreases : As we increase 's' from 0, all β's increase from 0 to their least square estimate values. Training error for 0 βs is the maximum and it steadily decreases to the Ordinary Least square RSS.

b.) (ii) Decrease initially, and then eventually start increasing in a U-shape : when $s=0$, all βs are 0, the model is extremely simple and has a high test RSS. As we increase s, βs assume non-zero values and model starts fitting well on test data and so test RSS decreases. Eventually, as βs approach their blown OLS values, they start overfitting to the training data, increasing test RSS.

c.) iii) Steadily increases. As s increases model becomes more flexible thus results in steady increase in variance.

d.) iv) Steadily decreases. As we increase s with 0, the model becomes more flexible, this leads to decrease in bias.

e.) v) Remains constant. Irreducible error is model independent and hence irrespective of the choice of s, remains constant.

Question 4: ISLR 6.8.5:

a.) $\hat{\beta}_0 = 0$, $n = p = 2$

Minimize: $(y_i - \hat{\beta}_1 x_{11} - \hat{\beta}_2 x_{12})^2 + (y_2 - \hat{\beta}_1 x_{21} - \hat{\beta}_2 x_{22})^2 + \lambda (\hat{\beta}_1^2 + \hat{\beta}_2^2)$

b.) $x_{11} = x_{12} = x_1$ & $x_{21} = x_{22} = x_2$

Differentiating w.r.t $\hat{\beta}_1, \hat{\beta}_2$ and equating to 0, we get

$$\hat{\beta}_1 (x_1^2 + x_2^2 + \lambda) + \hat{\beta}_2 (x_1^2 + x_2^2) = y_1 x_1 + y_2 x_2 \quad - (1)$$

5.) ISLR 8.4.5

$$\hat{\beta}_1 (x_1^2 + x_2^2) + \hat{\beta}_2 (x_1^2 + x_2^2 + \lambda) = y_1 x_1 + y_2 x_2 \quad - (2)$$

(1) - (2)

we get,

$$\hat{\beta}_1 = \hat{\beta}_2$$

c) Like Ridge Regression,

Mimize : $(y_1 - \hat{\beta}_1 x_{11} - \hat{\beta}_2 x_{12})^2 + (y_2 - \hat{\beta}_1 x_{21} - \hat{\beta}_2 x_{22})^2 + \lambda (|\hat{\beta}_1| + |\hat{\beta}_2|)$

d) $(y_1 - \hat{\beta}_1 x_1 - \hat{\beta}_2 x_1)^2 + (y_2 - \hat{\beta}_1 x_2 - \hat{\beta}_2 x_2)^2$

given,

$$\sum_{i=1}^{n} |\hat{\beta}_i| \leq s$$

The lasso constraint takes the shape of a diamond with center at origin of $(\hat{\beta}_1, \hat{\beta}_2)$

Thus, if $x_{11} = x_{12} = x_1$,

$x_{21} = x_{22} = x_2$,

$x_1 + x_2 = 0, \quad y_1 + y_2 = 0$

Mimimizing,

$$2[y_1 - (\hat{\beta}_1 + \hat{\beta}_2 x_1)]^2 \geq 0$$

A unique solution $\hat{\beta}_1 + \hat{\beta}_2 = \frac{y_1}{x_1}$ exists. This is parallel to the edge of diamond of constraints of $[y_1 - (\hat{\beta}_1 + \hat{\beta}_2) x_1]^2$ intersects the diamond of constraints. So, the edge $\hat{\beta}_1 + \hat{\beta}_2 = s$ is also a solution. Thus, the optimization problem has many possible solutions

<u>Question 5: ISLR 8.4.5 :</u>

p= c (0.1, 0.15, 0.2, 0.2, 0.55, 0.6, 0.6, 0.65, 0.7, 0.75)

<u>Majority</u> approach :

sum (p>=0.5) > sum (p<0.5)

TRUE

6.) ISLR 9.7.3

The number of red predictions is greater than the number of green predictions based on a 50% threshold, thus RED.

Average approach:

$$\text{mean}(p) = 0.45$$

The average of the probabilities is less than the 50% threshold, thus GREEN.

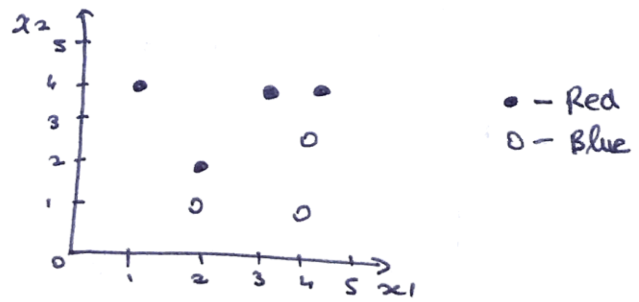
Question 6: ISLR 9.7.3:

(a) $x_1 = c(3,2,4,1,2,4,4)$

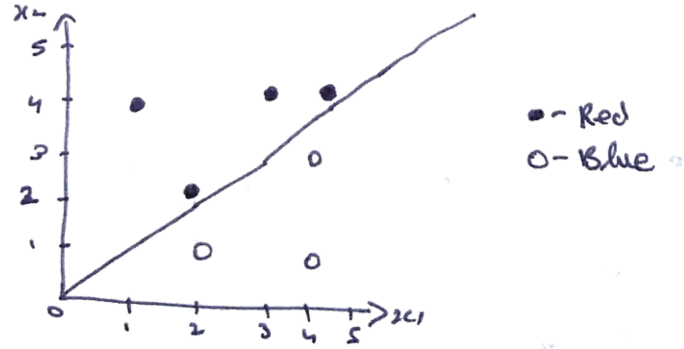
$x_2 = c(4,2,4,4,1,3,1)$

Colors = c("red", "red", "red", "red", "blue", "blue", "blue")

plot (x_1, x_2, col= colors, xlim= c(0,5), ylim= c(0,5)):



(b)



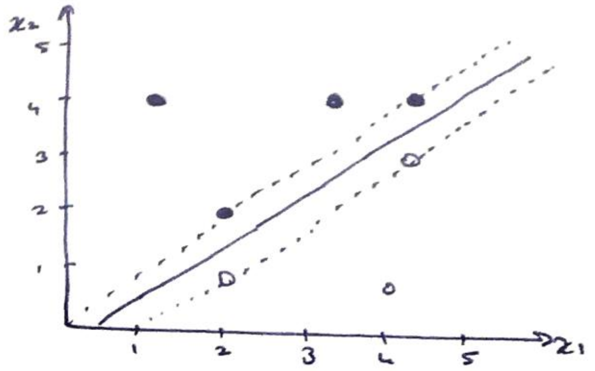
(c) If $x_1 - x_2 - 0.5 < 0$

Classify as red

else

Classify as blue

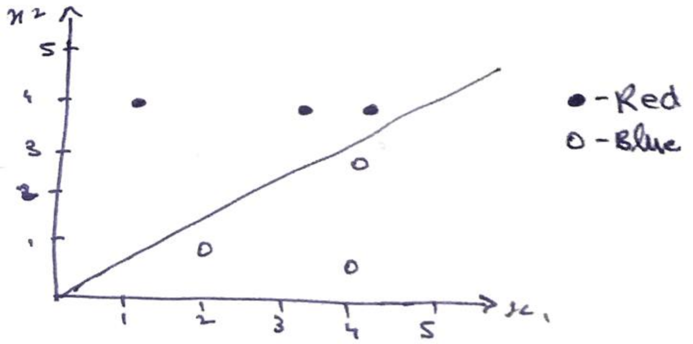
(d)



(e) Support vectors: (2,1), (2,2), (4,4), (4,3)

(f) If we moved (4,1), we would not change the maximal hyperplane and it is not a SV.

(g) $x_1 - x_2 - 0.3 = 0$ is not an optimal separating hyperplane.



(h)