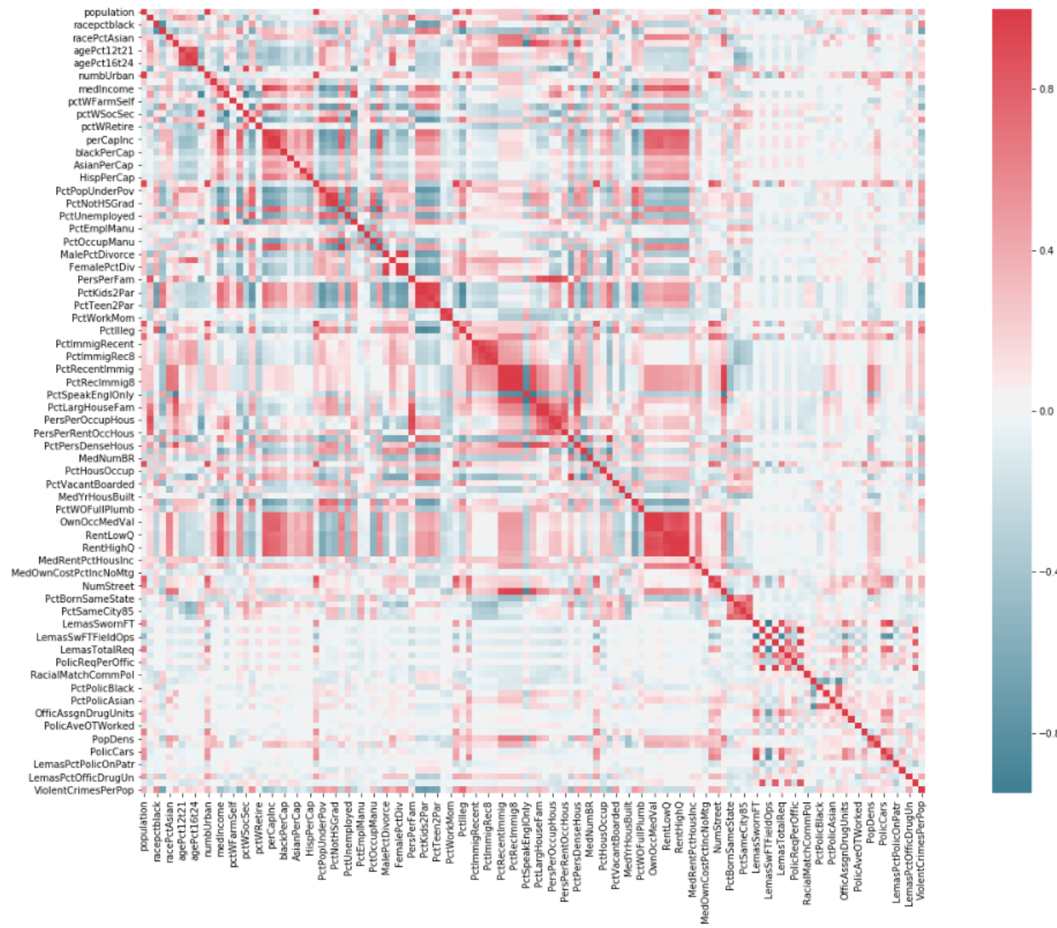
Name: Rohit Kulkarni

USC ID: 5402749044

1.c) Correlation matrix for training data

Correlation matric for test data



1.d) CV for training data

	population	householdsize	racepctblack	racePctWhite	racePctAsian	racePctHisp	agePct12t21	agePct12t29	agePct16t24	agePct65up	...	LandArea	PopDens	PctUsePubTrans	PolicCars	PolicOperBudg	LemasPctPolicOnPatr	LemasGangUnitDeploy	LemasPctOfficDrugUn	PolicBudgPerPop
0	2.241105	0.3558	1.428885	0.330213	1.3591	1.612091	0.369083	0.291315	0.50031	0.412776	...	1.645408	0.864499	1.39711	0.539823	0.774344	0.117988	0.371002	2.552946	0.320035

1 rows × 123 columns

CV for test data

	population	householdsize	racepctblack	racePctWhite	racePctAsian	racePctHisp	agePct12t21	agePct12t29	agePct16t24	agePct65up	...	LandArea	PopDens	PctUsePubTrans	PolicCars	PolicOperBudg	LemasPctPolicOnPatr	LemasGangUnitDeploy	LemasPctOfficDrugUn	PolicBudgPerPop
0	2.077145	0.346017	1.356525	0.304182	1.360714	1.622216	0.356343	0.288878	0.480111	0.454808	...	1.758711	0.89624	1.479509	0.482417	0.565815	0.134441	0.358745	2.563248	0.383252

1 rows × 123 columns

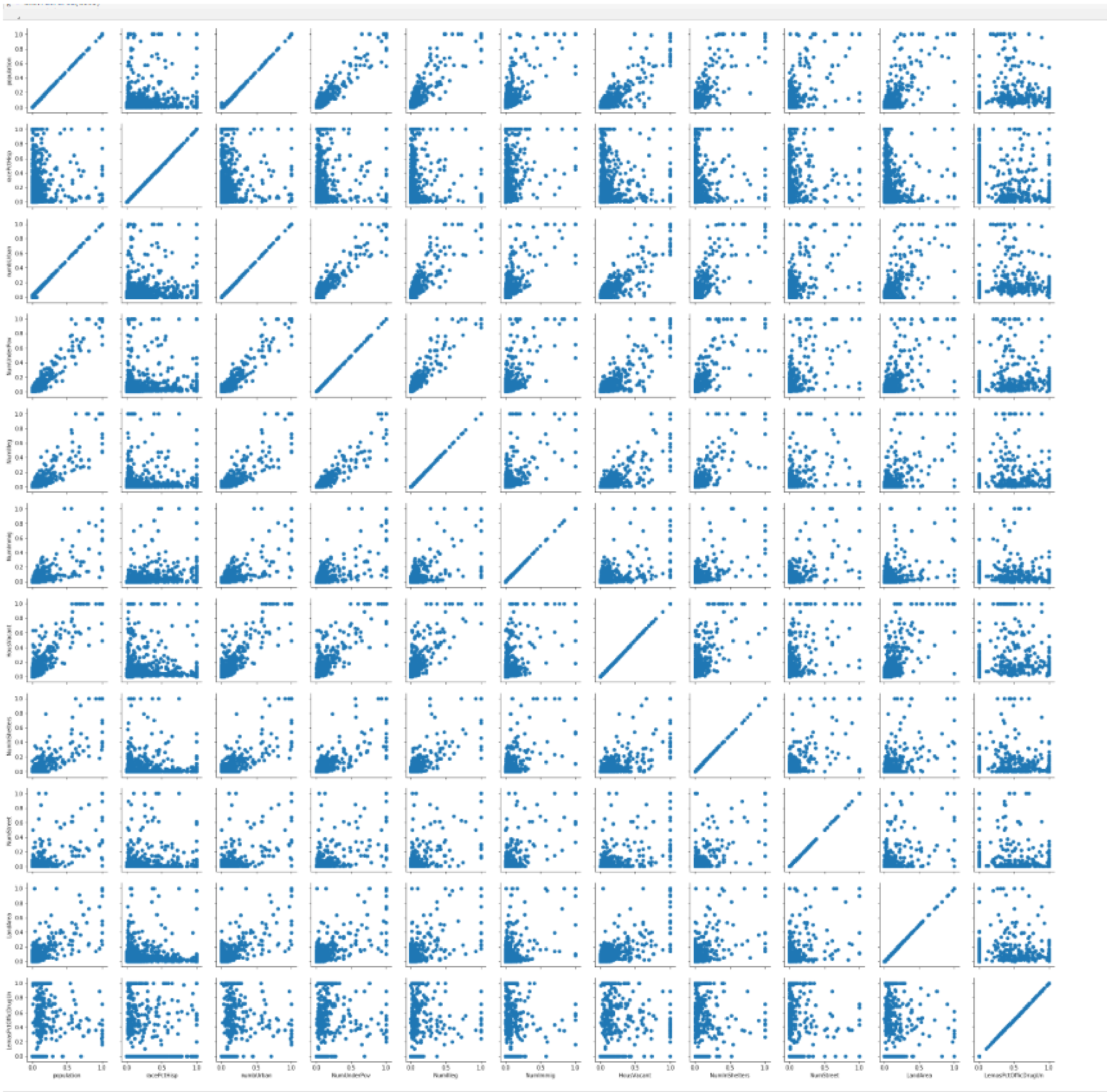
Top 11 features with highest CV for training set:

```
{'population': 2.2411046245803745,
 'racePctHisp': 1.612091005228411,
 'numbUrban': 2.0384614919156445,
 'NumUnderPov': 2.3424431162181505,
 'NumIlleg': 3.0589643472092356,
 'NumImmig': 2.9266352462888148,
 'HousVacant': 1.9684670491351257,
 'NumInShelters': 3.470952139705214,
 'NumStreet': 4.292922989491593,
 'LandArea': 1.6454078602149063,
 'LemasPctOfficDrugUn': 2.552945511727576}
```

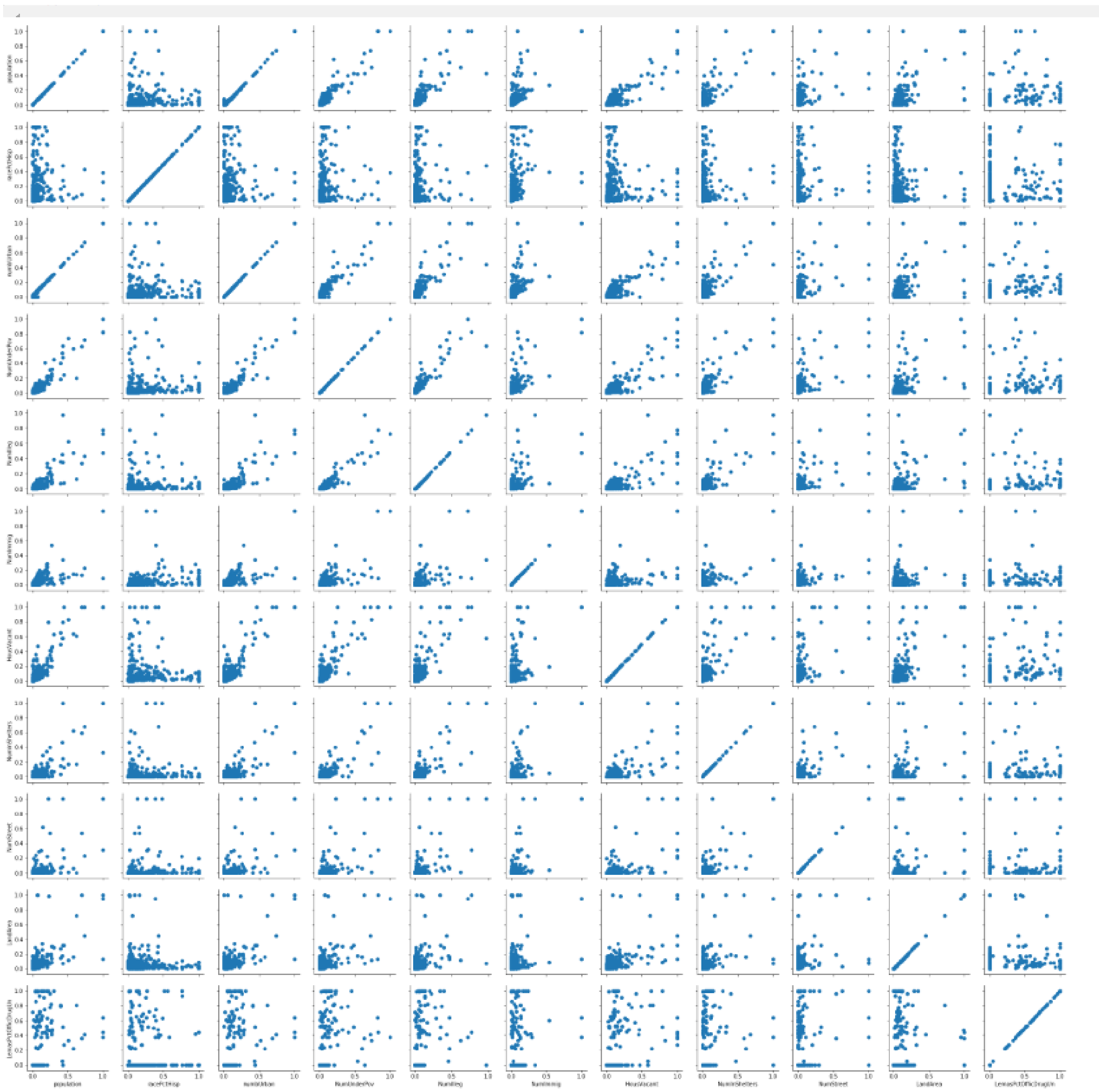
Top 11 features with highest CV for test set:

```
{'population': 2.0771450949119687,
 'racePctHisp': 1.6222157248397688,
 'numbUrban': 1.8792266468028023,
 'NumUnderPov': 2.1530367486954023,
 'NumIlleg': 2.729758056633798,
 'NumImmig': 2.803986484258142,
 'HousVacant': 1.9321407334264848,
 'NumInShelters': 3.5334392704842097,
 'NumStreet': 4.761101784255363,
 'LandArea': 1.7587114847016307,
 'LemasPctOfficDrugUn': 2.563247519271469}
```

1.e) Scatter plot for 11 features selected from CV for training data

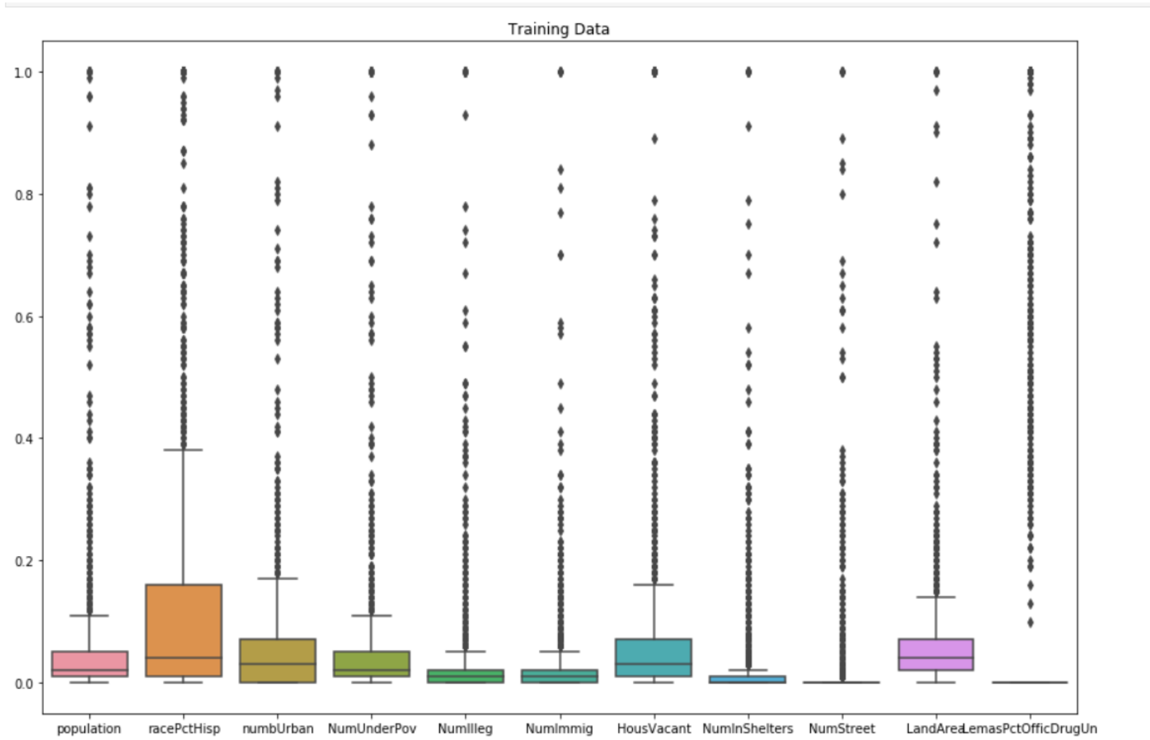


Scatter plot for 11 features selected from CV for test data

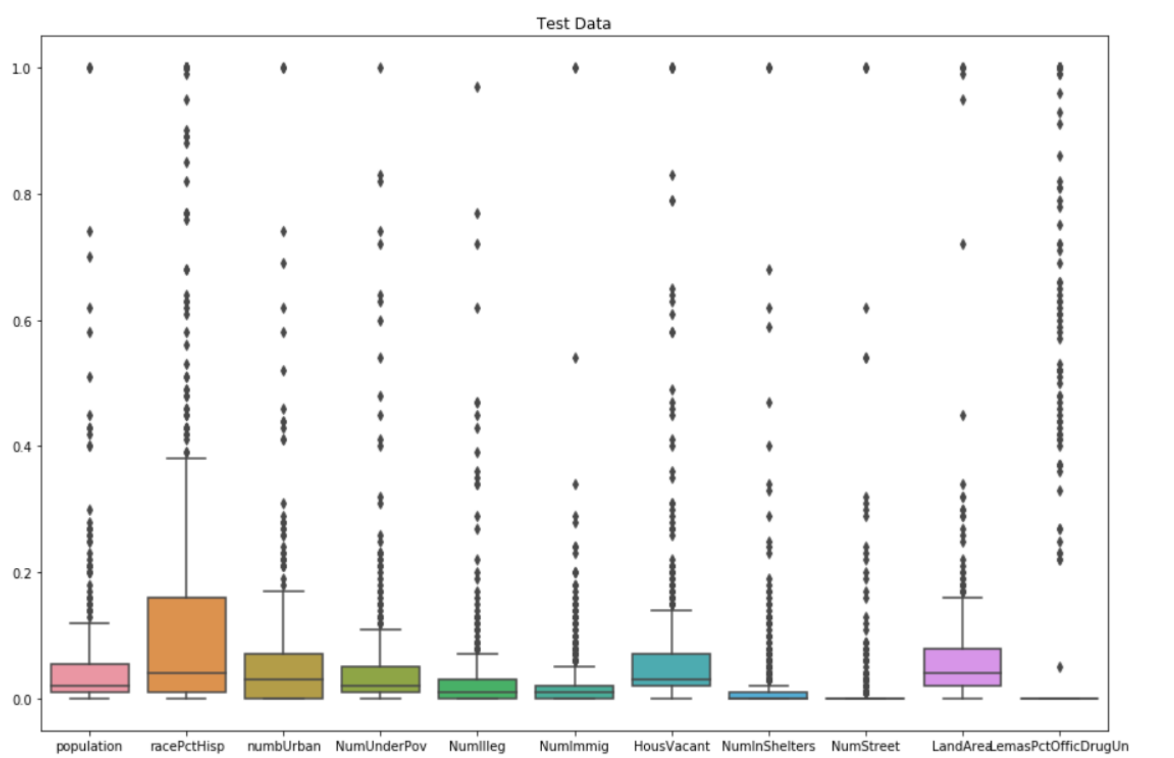


By looking at the scatter plots and box plots we can’t conclude much on the pattern of the plots

Box plot for 11 features selected from CV for training data



Box plot for 11 features selected from CV for test data



1.f) Linear model and test error

```
Coefficients:
 [[ 2.13143761e-02 -1.89271223e-02  2.78983299e-01  3.89685109e-03
  -2.93051530e-02  1.10497574e-02  1.75324626e-01 -3.45012797e-01
  -1.19751657e-01  5.61381476e-02 -1.82301243e-01  4.99625643e-02
  -1.71644327e-01 -1.37297267e-01  4.54779925e-02 -2.08226531e-01
   1.37417835e-01  6.15660474e-02 -1.23869637e-01  3.06028422e-01
  -4.45852418e-02 -1.96920314e-01 -3.96805409e-02 -4.33613714e-02
   2.98970048e-02  3.09714952e-02  3.24953633e-02  8.88745268e-02
  -2.10272096e-01 -5.15589445e-02  2.18042489e-02  9.98632456e-02
   2.31875943e-02  2.90526319e-01 -4.62717180e-02 -9.71560865e-03
   5.93142880e-02  5.88516592e-02  5.25623723e-01  2.44825078e-01
   1.84319457e-01 -6.80295252e-01 -1.11274746e-01 -4.04617684e-02
  -2.05636221e-01 -1.70293272e-02 -1.49037221e-02  5.42928899e-02
  -2.05649712e-01 -1.17116758e-01  6.36193074e-02 -1.63658775e-01
   5.12562745e-03  2.81435050e-02 -7.23858488e-04 -2.67532758e-02
  -7.62234260e-02 -2.69472809e-02  9.14541909e-02  3.67034226e-02
  -4.53484355e-02 -1.90428563e-01  1.15724843e-01 -3.00391950e-01
   4.86154165e-01  1.73770168e-02 -2.38724133e-01 -8.94362867e-01
   2.91981766e-01  1.21744429e-01  3.02824220e-02  1.64372416e-01
  -4.56016599e-02  7.70941922e-01  8.13244939e-02 -9.21805280e-02
  -3.12559670e-02  3.12498051e-02 -3.58505817e-02 -3.99161723e-01
   4.70839401e-01 -1.51133142e-01 -1.55101962e-01 -1.41830850e-01
  -4.70683328e-02  3.77447007e-01  8.34231616e-02 -3.29230420e-02
  -8.40862461e-02  9.70369366e-02  1.75632911e-01  1.13311278e-01
   3.85989529e-03 -2.30787143e-02  2.98857941e-02  1.62044204e-02
  -3.07537682e-01  1.93819475e+03 -5.05954898e-03  3.53469065e-01
  -3.31228806e-01 -1.07107541e-01  2.58511207e-01 -1.93818236e+03
  -4.36977106e-02 -5.79812179e-02 -1.14028100e-01 -1.81620946e-02
   3.21726590e-02  2.40608551e-02 -4.12361226e-02 -4.11426519e-02
  -3.20703644e-02  9.75054669e-02  2.54790963e-02 -6.71352640e-02
   1.51807076e-01  5.44358346e-01  1.27718720e-05  4.31458728e-02
  -6.45630097e-02 -1.95277847e-01]]
Mean squared error: 0.79
```

1.g) Ridge regression model with lambda chosen by cross validation

```
0.0466301673441609
```

```
reg = linear_model.Ridge(alpha = ridgecv.alpha_)
reg.fit(x_train,y_train)
```

```
Ridge(alpha=0.0466301673441609, copy_X=True, fit_intercept=True,
   max_iter=None, normalize=False, random_state=None, solver='auto',
   tol=0.001)
```

```
print(str(reg.coef_))
```

```
[[-0.04077452 -0.01668977  0.28501796  0.00911114 -0.02737643  0.01862504
   0.16483167 -0.3428528  -0.11222297  0.05206098 -0.08757362  0.04719112
  -0.1033941  -0.13665633  0.04533881 -0.19354963  0.13100954  0.06068982
  -0.12225709  0.24464891 -0.03901445 -0.20359898 -0.03869227 -0.04182225
   0.02976266  0.03025251  0.03163524  0.05579698 -0.19889279 -0.06550446
   0.03714833  0.09745029  0.0193682   0.28506243 -0.04739365 -0.01374812
   0.06129166  0.05594509  0.34352763  0.24697165 -0.03163303 -0.30385776
  -0.06907883 -0.03650568 -0.20041145 -0.01817829 -0.02101236  0.05066349
  -0.20345933 -0.10006477  0.0683127  -0.12826298  0.00365858  0.02337131
   0.00484926 -0.02781476 -0.0596613  -0.01004518  0.04821084  0.05439776
  -0.03557448 -0.18473322  0.05231708 -0.24000254  0.48294691 -0.08651599
  -0.17416497 -0.51949951  0.29322348  0.11657756  0.02657123  0.17591443
  -0.0470502   0.38907238  0.08426971 -0.08877788 -0.02870453  0.03403532
  -0.03695785 -0.29509885  0.3181245  -0.08639896 -0.1693399  -0.11676526
  -0.04744317  0.35523237  0.08537048 -0.0347202  -0.08438798  0.10559869
   0.1851947   0.10698761 -0.0011692  -0.01871145  0.02684307  0.01656342
  -0.00646607  0.0408589  -0.01358931  0.19824782 -0.28220705 -0.03790137
   0.19623359  0.04017444 -0.04514295 -0.04503291 -0.12135126 -0.03364643
   0.02868178  0.04577041 -0.25616051 -0.05428277 -0.03233173  0.10985767
   0.02784172 -0.06861444  0.09057942  0.35735599  0.02280333  0.04648865
   0.00766469 -0.14290669]]
```

Test error

```
ViolentCrimesPerPop    0.01803
dtype: float64
```

1.h) Co-efficient for the Lasso model with standardized features

```
[ 0.00000000e+00 -0.00000000e+00  2.01426606e-01 -1.65122575e-02
  0.00000000e+00  0.00000000e+00 -0.00000000e+00 -6.21841232e-02
 -0.00000000e+00  0.00000000e+00  0.00000000e+00  3.15010662e-02
  0.00000000e+00 -0.00000000e+00  0.00000000e+00 -1.36885581e-02
  0.00000000e+00  4.51117999e-03 -3.05682124e-02  0.00000000e+00
  0.00000000e+00  0.00000000e+00 -0.00000000e+00 -4.26998799e-06
  1.01466223e-02  0.00000000e+00  0.00000000e+00  0.00000000e+00
 -0.00000000e+00 -0.00000000e+00  0.00000000e+00 -0.00000000e+00
  0.00000000e+00  0.00000000e+00 -0.00000000e+00 -0.00000000e+00
  0.00000000e+00 -0.00000000e+00  1.18132334e-01  0.00000000e+00
  0.00000000e+00  0.00000000e+00  0.00000000e+00 -0.00000000e+00
 -2.49089093e-01 -0.00000000e+00 -0.00000000e+00 -0.00000000e+00
 -6.39776748e-02  0.00000000e+00  1.51575917e-01 -0.00000000e+00
 -0.00000000e+00 -0.00000000e+00 -0.00000000e+00 -0.00000000e+00
  0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
 -0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
  0.00000000e+00 -0.00000000e+00  0.00000000e+00 -0.00000000e+00
  1.47871465e-01  2.70632274e-02 -0.00000000e+00  9.49705053e-02
 -3.85053873e-02 -0.00000000e+00  3.87190610e-02 -7.57426922e-03
 -0.00000000e+00  0.00000000e+00 -0.00000000e+00  0.00000000e+00
  0.00000000e+00  0.00000000e+00 -0.00000000e+00  0.00000000e+00
  0.00000000e+00  0.00000000e+00  4.36337260e-02  0.00000000e+00
 -2.45378175e-02  0.00000000e+00  1.44807999e-01  2.60558995e-02
 -0.00000000e+00 -0.00000000e+00  3.59220259e-03  0.00000000e+00
 -0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
 -0.00000000e+00  1.40066957e-02  4.02887357e-02  0.00000000e+00
 -0.00000000e+00  0.00000000e+00 -0.00000000e+00 -0.00000000e+00
  0.00000000e+00 -0.00000000e+00 -0.00000000e+00 -0.00000000e+00
 -0.00000000e+00  0.00000000e+00  0.00000000e+00 -0.00000000e+00
  0.00000000e+00 -0.00000000e+00  1.82415348e-03  1.96770336e-02
  4.45632839e-03  1.04475884e-02]
```

Test Error

```
0.07979162370138596
```

Co-efficient for the Lasso model without standardized features

```
[-0.          0.          0.27249895 -0.         -0.01665789  0.
  0.07109539 -0.27784678  0.          0.0011751  -0.          0.04073145
  0.         -0.08891917  0.02771806 -0.10354325  0.05391932  0.04011279
 -0.10917987  0.         -0.         -0.06268167 -0.0217659  -0.03149086
  0.02913329  0.02311615  0.02483343 -0.         -0.14530918 -0.01971014
  0.          0.00743498 -0.          0.14574506 -0.01741799  0.
  0.01310179  0.          0.12481517  0.14171105 -0.08644452 -0.
  0.         -0.         -0.19722105 -0.03685726 -0.0107591   0.
 -0.13195846 -0.06160317  0.09706191 -0.0750557  -0.         -0.
 -0.         -0.00535548 -0.          0.          0.          0.03847502
  0.         -0.07598984 -0.         -0.07663781  0.05692306 -0.
 -0.         -0.06082072  0.22506971  0.06096321  0.00468589  0.12739235
 -0.05984836  0.          0.07417148 -0.06134435 -0.01607479  0.02542797
 -0.02501277 -0.01552814 -0.         -0.         -0.1619326  -0.
  0.          0.1670111   0.09050384 -0.01572268 -0.07872944  0.05915423
  0.16826345  0.05057059 -0.00066031 -0.          0.02370337 -0.
 -0.          0.          0.          0.07703976 -0.03093294  0.
  0.09245849  0.         -0.01682801 -0.         -0.02987966  0.00189325
  0.0065833   0.         -0.00505628 -0.03130874 -0.01443093  0.03645794
  0.01123726 -0.04554653  0.         -0.          0.02395635  0.04084537
  0.00131602  0.        ]
```

Test Error

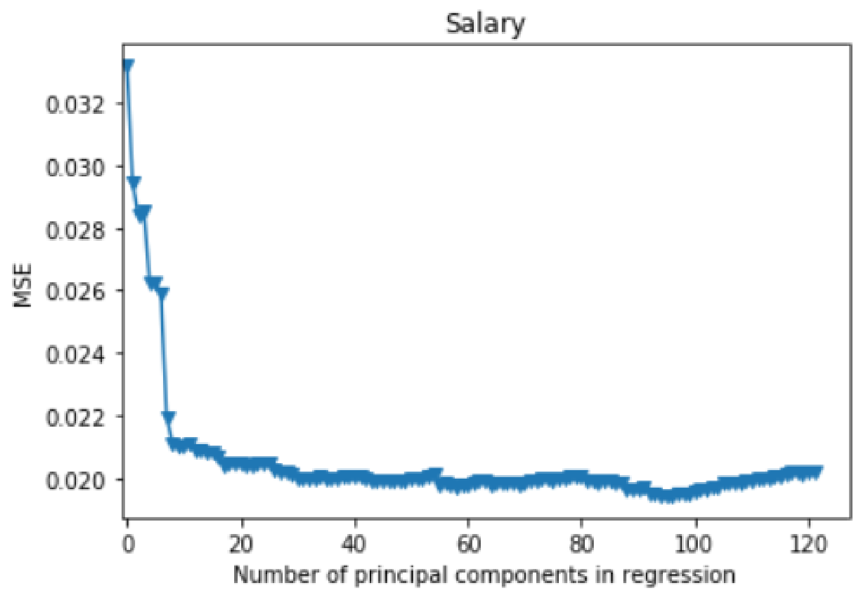
```
0.08361866691992705
```

Test error is more in the model without standardized features

1.i) PCR Model on the training set with M chosen by cross-validation

MSEs for training set

```
[0.033165927121858306, 0.029419347704705966, 0.02839679461776524, 0.028502439023133087, 0.026234815609429946, 0.02626585968541368, 0.0258728534
71978395, 0.021959466321352573, 0.02113467944013831, 0.02105015210753879, 0.021060967801450284, 0.021128486396771548, 0.020890532861087087, 0.0
20903123773029303, 0.020860058037153767, 0.020811564691183306, 0.020692684872407146, 0.020442885961710264, 0.020465347383767284, 0.020482909941
89208, 0.020500733294658454, 0.020414177780887326, 0.02044029612317662, 0.02046895156856308, 0.020477988642937078, 0.020483118624945874, 0.0202
79740059440127, 0.02021173707782086, 0.02022405907237875, 0.020141520747341417, 0.019990610487204317, 0.020017481991480065, 0.01999472347992598
7, 0.020010793683891994, 0.020055342610204564, 0.020016482133478678, 0.019994750582621852, 0.020017625588200996, 0.02003663272864928, 0.0200721
59025839358, 0.02008195907100758, 0.02003543262751797, 0.02002702091520193, 0.01992382615102269, 0.019936317566160537, 0.019919377104145743, 0.
019920195146917473, 0.019938356011035312, 0.019932702508595024, 0.019953711965773925, 0.01999640934116543, 0.020004453084061187, 0.020012222437
102916, 0.020037176742201436, 0.020104664568454486, 0.01980922567304589, 0.01984715791271577, 0.019762942838192306, 0.019743680459456566, 0.019
757271119794518, 0.019820463253473832, 0.01984091356351838, 0.019891389292592036, 0.019911789154550642, 0.019814791464133893, 0.019848882254275
033, 0.019858385635729982, 0.01987272342527037, 0.019858656066632153, 0.01981363483567427, 0.019863869045741216, 0.01989846782294003, 0.0199519
73261176776, 0.019988982449932483, 0.020013369209079524, 0.019894173161367087, 0.019963336889333806, 0.02002774194302136, 0.020060458376293017,
0.020036979282051698, 0.02005094207464954, 0.019898424967771625, 0.01989731278289108, 0.019888156787673007, 0.01989267197483464, 0.019935859802
766485, 0.019832230081626975, 0.01986472566956729, 0.01961895145073693, 0.01963970348060392, 0.019637110378485124, 0.019704282532971414, 0.0195
23753726074764, 0.01953000311219192, 0.01944852580968131, 0.0194459259850635, 0.019469297535756797, 0.019482191177151928, 0.019496796151259137,
0.019519726703834, 0.019547442000399873, 0.019665630480040133, 0.019680553461754638, 0.019700752343264977, 0.01972475314330777, 0.0198408569992
1296, 0.01983092431716628, 0.01987840065837645, 0.019871579431477885, 0.01989968684096443, 0.019947489451128095, 0.019994064139414867, 0.020003
211461822587, 0.020017761629816774, 0.020090966645165388, 0.020089672931258015, 0.020152438698470866, 0.02018729240551665, 0.02018512821844636
2, 0.02016043943573785, 0.020170135885417998, 0.020212069610632193]
```



Test Error for PCR Model

```
0.01881964401002089
```

1.j) XGBoost model Accuracy

```
Accuracy:63.72249184837258
```

Best Score

```
0.6482943720851919
```

Best Params

```
{'learning_rate': 0.05}
```

2. Tree-Based Methods

2.b.i) Data Imputation technique to deal with the missing values in the data set: Imputation is replacing missing values with substitute values.

The following are common methods:

- Mean: the mean of the observed values for that variable.
- Substitution: the value from a new individual who was not selected to be in the sample.
- Hot deck: a randomly chosen value from an individual who has similar values on other variables.
- Cold deck: a systematically chosen value from an individual who has similar values on other variables.
- Regression: the predicted value obtained by regressing the missing variable on other variables.
- Stochastic regression: the predicted value from a regression plus a random residual value.
- Interpolation and extrapolation: an estimated value from other observations from the same individual.

2.b.ii) CV for training set

	aa_000	ab_000	ac_000	ad_000	ae_000	af_000	ag_000	ag_001	ag_002	ag_003	...	ee_002	ee_003	ee_004	ee_005	ee_006	ee_007
0	2.450938	2.297635	2.169622	193.924333	23.202587	18.670972	91.97698	34.763699	17.344552	8.544596	...	2.579055	2.558711	2.606081	2.829186	3.191612	4.962276

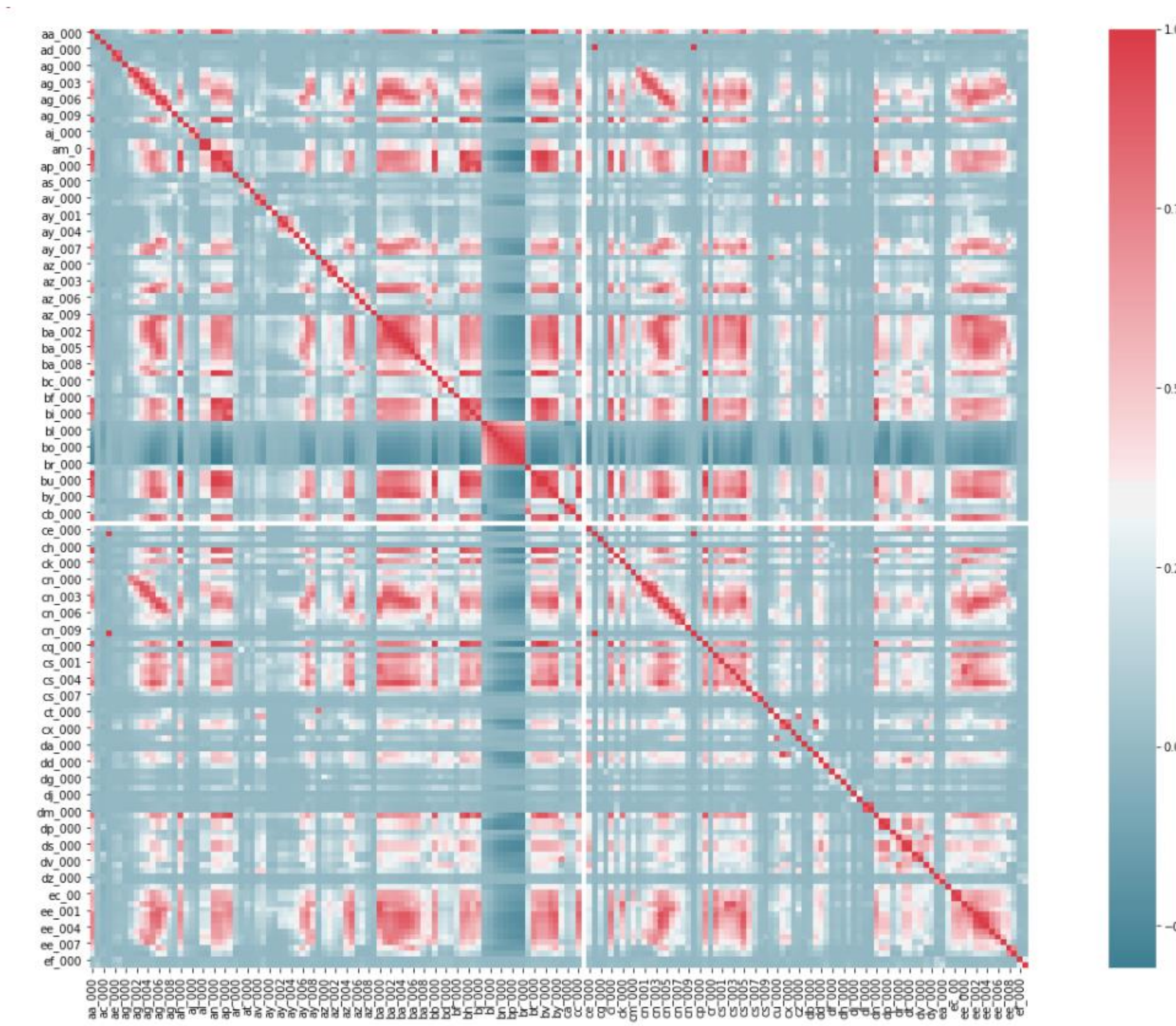
1 rows × 170 columns

CV for test set

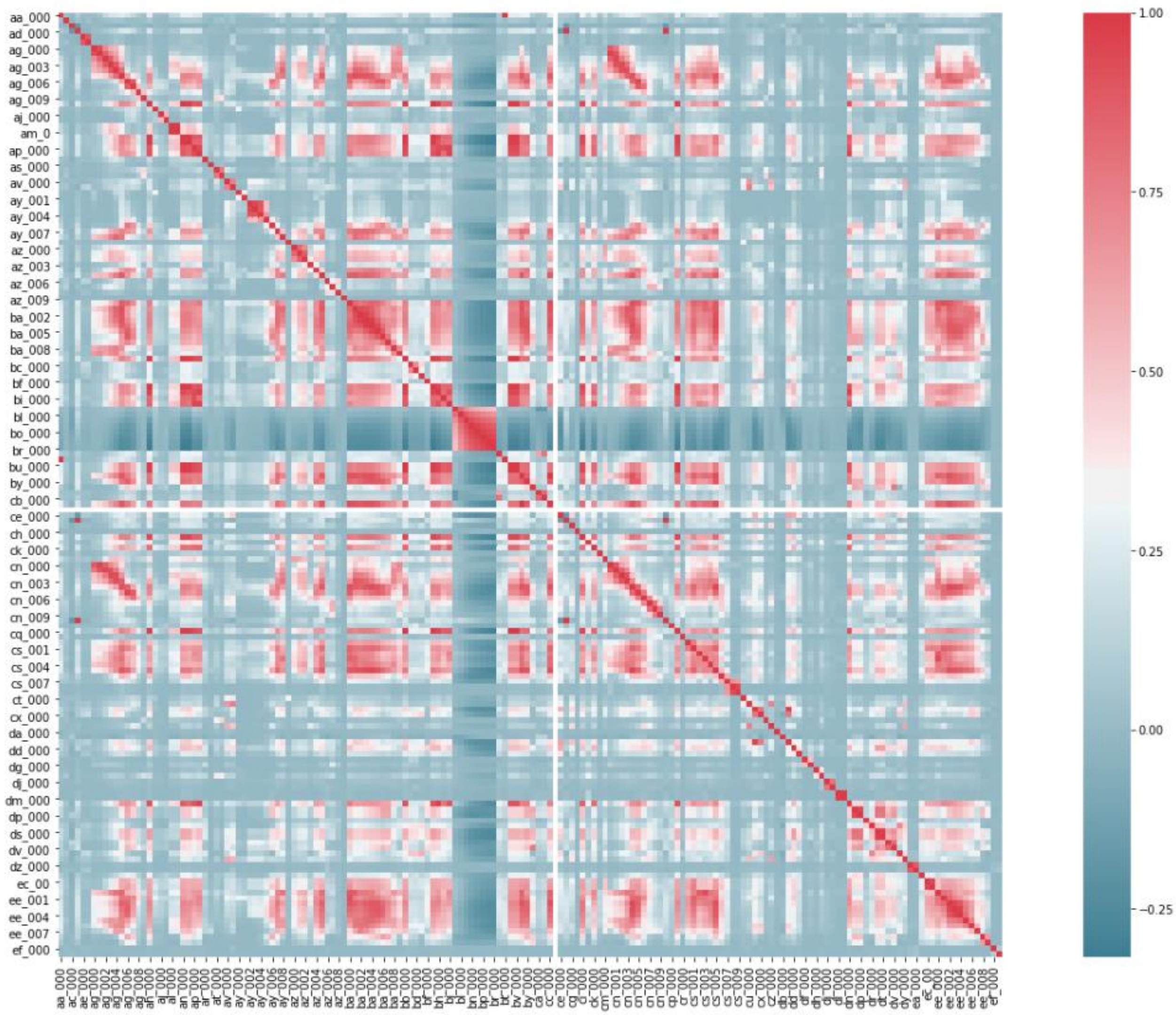
	aa_000	ab_000	ac_000	ad_000	ae_000	af_000	ag_000	ag_001	ag_002	ag_003	...	ee_002	ee_003	ee_004	ee_005	ee_006	ee_007
0	7.405254	1.677003	2.16037	1.717777	17.947223	16.003612	52.680695	43.197091	17.057915	9.072389	...	2.684667	2.620296	2.687163	2.943285	3.351986	4.594222

1 rows × 170 columns

2.b.iii) Correlation matrix for training set



Correlation matrix for test set



2.b.iv) Top 13 feature with highest CV

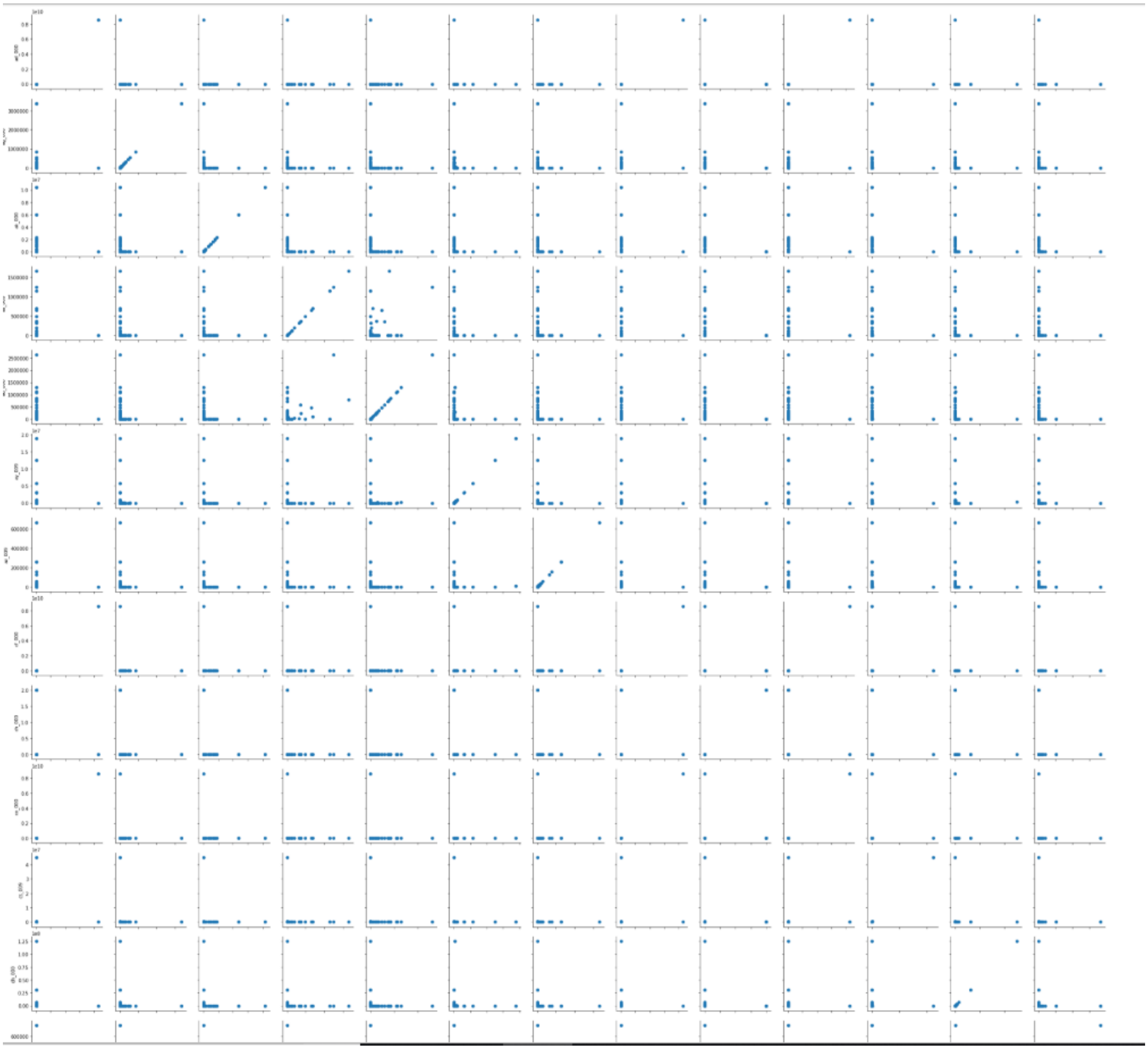
For Training set

```
{'ad_000': 193.92433336319496,
 'ag_000': 91.97698048338566,
 'ak_000': 74.5855911285946,
 'as_000': 85.56342057329114,
 'au_000': 67.94631220846355,
 'ay_009': 83.78960830045412,
 'az_009': 77.0594504458304,
 'cf_000': 194.35326136275233,
 'ch_000': 57.88627697279238,
 'co_000': 194.0391530490907,
 'cs_009': 234.45338812778706,
 'dh_000': 115.64846213471637,
 'dj_000': 111.14849878529981}
```

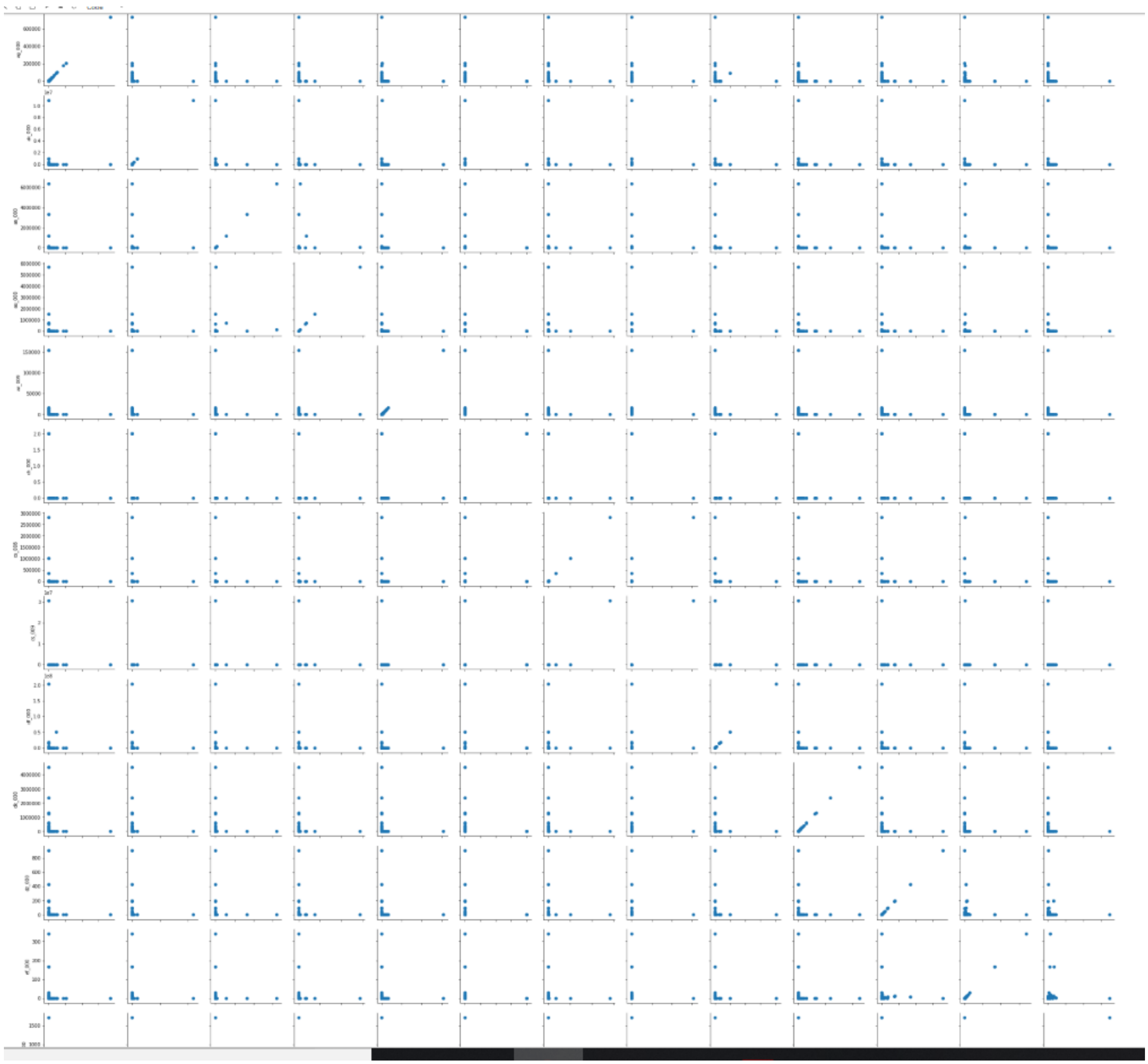
For test set

```
{'ag_000': 52.68069460174049,
 'ak_000': 93.41515521641949,
 'as_000': 82.90902291563049,
 'au_000': 84.93553042123239,
 'az_009': 59.1000217572831,
 'ch_000': 56.09322253971697,
 'cs_008': 62.78183247230092,
 'cs_009': 125.35907975754283,
 'df_000': 76.40236144718226,
 'dk_000': 45.99352997153798,
 'dz_000': 48.9167310161583,
 'ef_000': 49.94978581961838,
 'eg_000': 57.93011623794346}
```

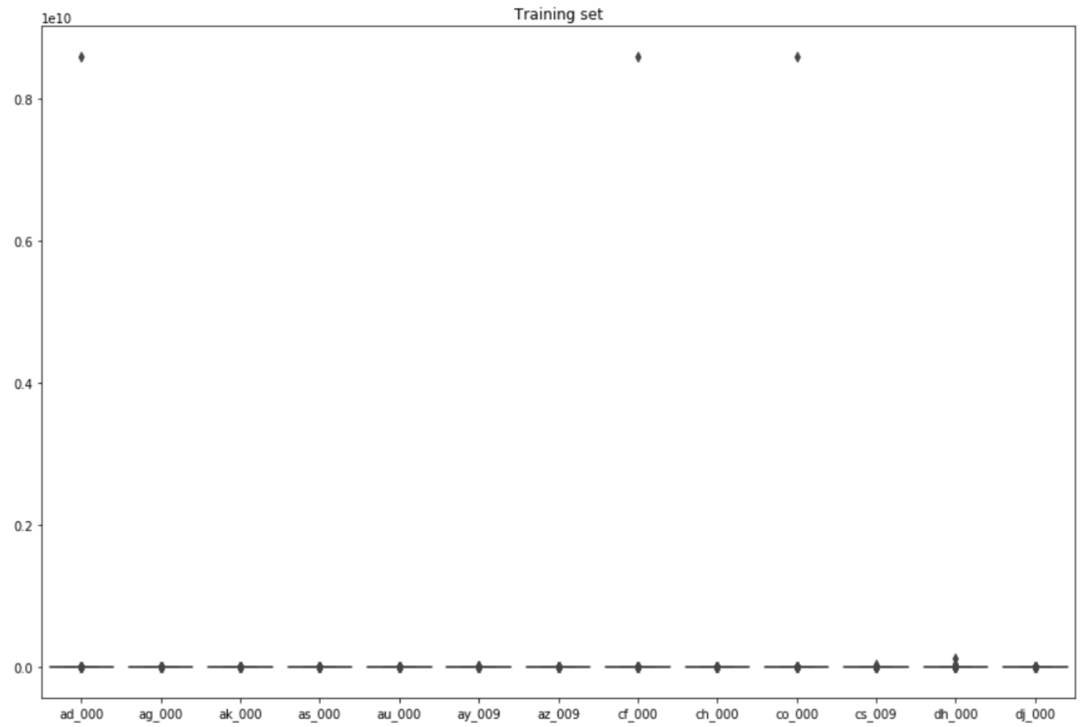
Scatter plot for 13 features selected from CV for training set



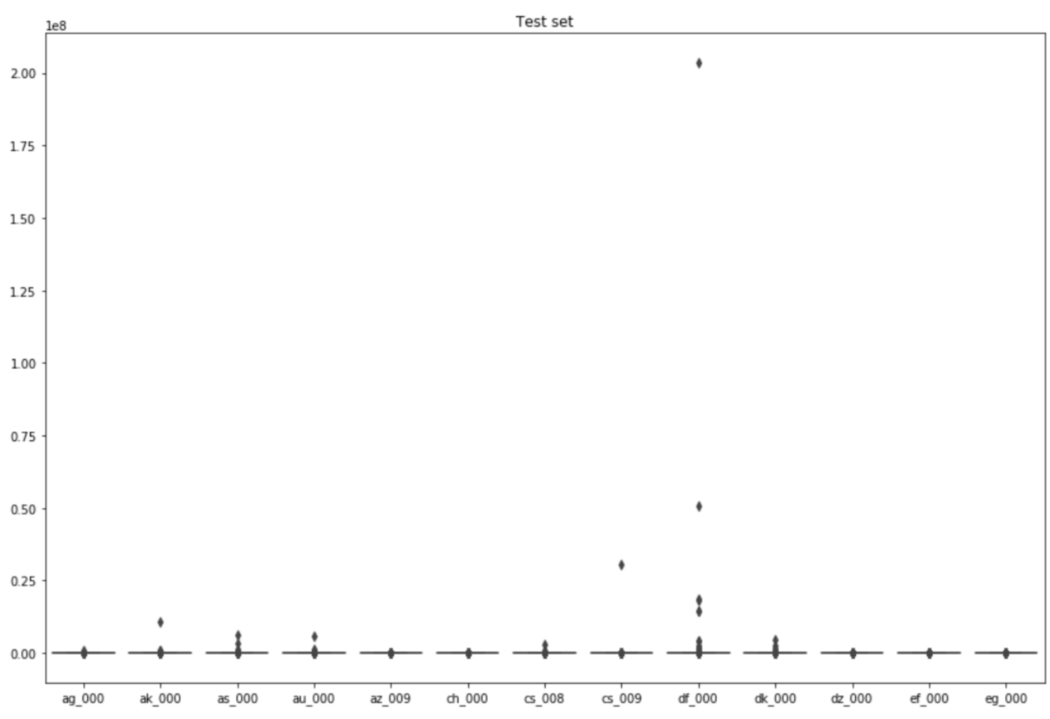
Scatter plot for 13 features selected from CV for test set



Box plot for training set



Box plot for test set



From scatter plots and box plots we can conclude that there seems to be negative skewed class for the data.

2.b.v) Number of positive data and negative data for training set respectively are:

```
1000
59000
```

Number of positive data and negative data for test set respectively are:

```
375
15625
```

This data set is imbalanced as you can see from the above screenshots.

2.c) Random forest classifier accuracy

```
Accuracy: 0.992375
```

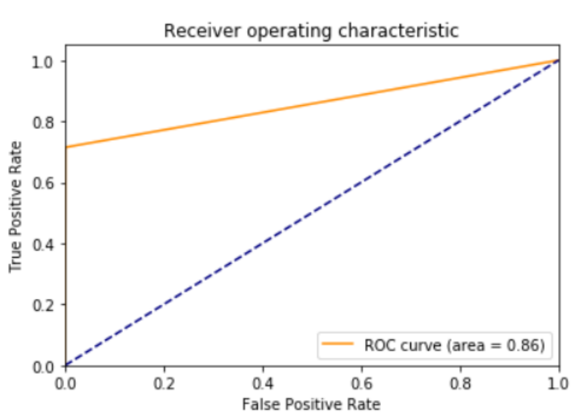
Confusion matrix

```
array([[15610,    15],
       [  107,   268]], dtype=int64)
```

AUC

```
0.8568533333333332
```

ROC



Miscalculation

```
Misclassfication: 0.008000000000000007
```

Out of Bag error

```
0.9939666666666667
```

Test error

```
Mean squared error: 0.01
```

OOB error is much greater compared to test error.

2.d)

There are usually two methods to deal with imbalanced data while using the random forest model. One approach is cost-sensitive learning and the other is sampling. For extremely imbalanced data, random forest generally tends to be biased towards the majority class.

The cost-sensitive approach would be to assign different weights to different classes. So if the minority class is assigned a higher weight and thus higher misclassification cost, then that can help reduce its biasness towards the majority class. You can use the class weight parameter of random forest in scikit-learn to assign weights to each class.

Secondly, there are different methods of sampling such as oversampling the minority class or under sampling the majority class etc... Although simple sampling methods improve the overall model performance, its preferable to go for a more specialized sampling method such as SMOTE and others to get a better model.

Most of the machine learning models suffer from the imbalanced data problem although there are some reasons to believe that generative models generally tend to perform better in case of imbalanced datasets

Random forest classifier accuracy for class balanced data

```
Accuracy: 0.9888125
```

Misclassification

```
Misclassfication: 0.011125000000000052
```

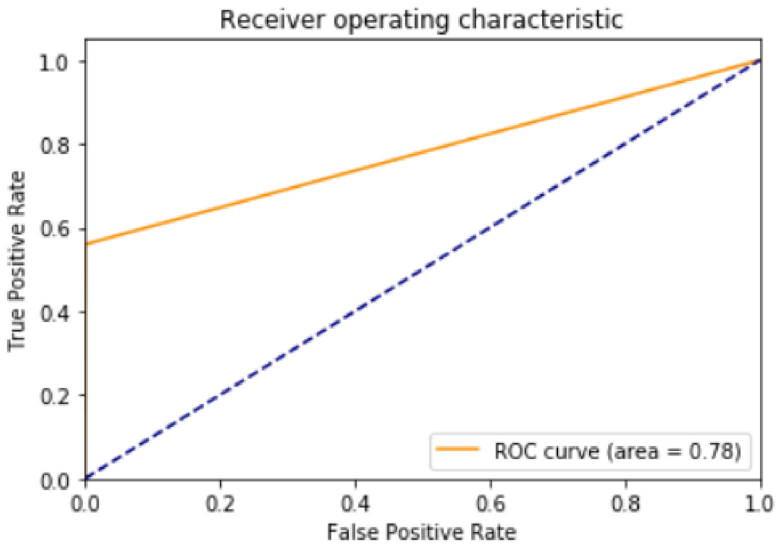
Confusion matrix

```
array([[15612,    13],
       [  165,   210]], dtype=int64)
```

AUC

0.7795839999999999

ROC Curve



2.e) Model trees using Weka

```
=== Summary ===

Correctly Classified Instances       59568               99.28   %
Incorrectly Classified Instances       432                0.72   %
Kappa statistic                          0.7542
Mean absolute error                      0.011
Root mean squared error                  0.0759
Relative absolute error                 33.648  %
Root relative squared error             59.2755 %
Total Number of Instances            60000

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.998    0.324    0.995      0.998    0.996      0.760    0.980     0.999     neg
                 0.676    0.002    0.862      0.676    0.758      0.760    0.980     0.831     pos
Weighted Avg.    0.993    0.319    0.992      0.993    0.992      0.760    0.980     0.997

=== Confusion Matrix ===

     a     b   <-- classified as
 58892   108 |     a = neg
   324   676 |     b = pos
```

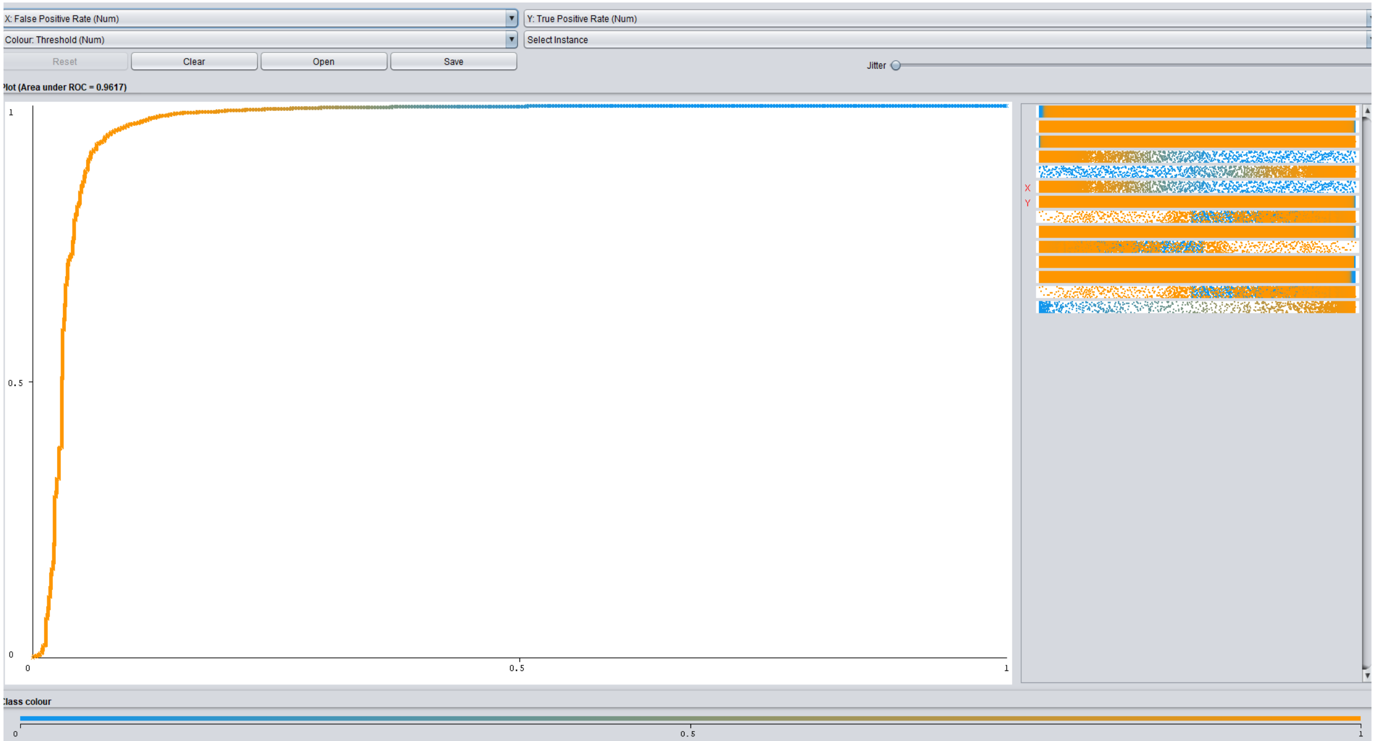
Predicting the test set

```
=== Re-evaluation on test set ===

User supplied test set
Relation:     Test_data
Instances:     unknown (yet). Reading incrementally
Attributes:   171

=== Summary ===

Correctly Classified Instances       15836              98.975  %
Incorrectly Classified Instances       164               1.025  %
Kappa statistic                          0.7578
Mean absolute error                      0.0139
Root mean squared error                  0.0906
Total Number of Instances            16000

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.997    0.296    0.993      0.997    0.995      0.761    0.977     0.998     neg
                 0.704    0.003    0.833      0.704    0.763      0.761    0.977     0.811     pos
Weighted Avg.    0.990    0.289    0.989      0.990    0.989      0.761    0.977     0.994

=== Confusion Matrix ===

     a     b   <-- classified as
 15572    53 |     a = neg
   111   264 |     b = pos
```

Using 5-fold cross validation method, confusion matrix

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances       59474               99.1233 %
Incorrectly Classified Instances       526                0.8767 %
Kappa statistic                          0.7069
Mean absolute error                      0.0122
Root mean squared error                  0.0847
Relative absolute error                 37.156  %
Root relative squared error             66.1754 %
Total Number of Instances            60000

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.997    0.352    0.994      0.997    0.996      0.710    0.962     0.998     neg
                 0.648    0.003    0.788      0.648    0.711      0.710    0.962     0.747     pos
Weighted Avg.    0.991    0.346    0.991      0.991    0.991      0.710    0.962     0.994

=== Confusion Matrix ===

     a     b   <-- classified as
 58826   174 |     a = neg
   352   648 |     b = pos
```

AUC and ROC Curve

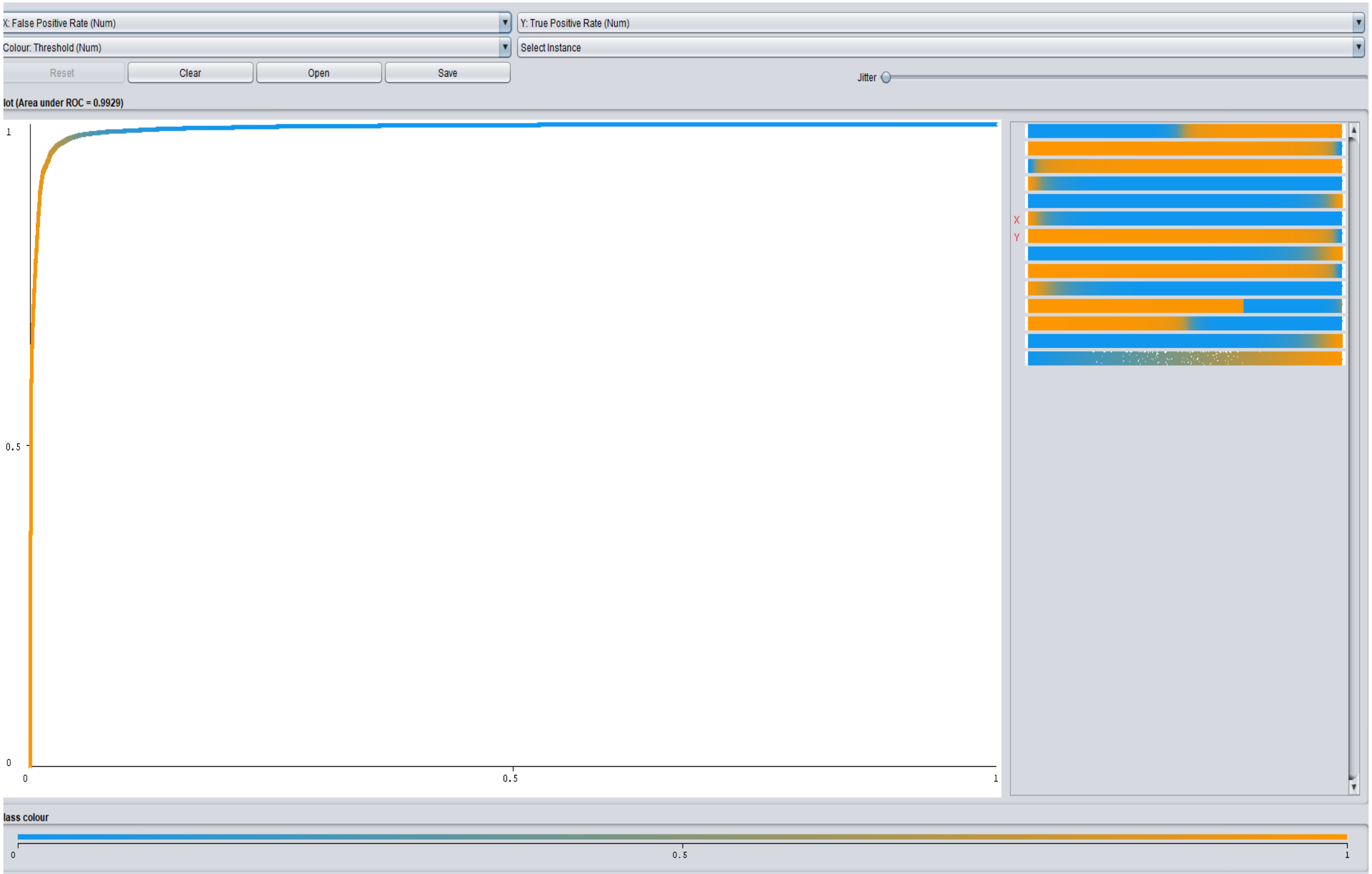


2.f) Model Tree using SMOTE

```
eg_000        1.0006


Time taken to build model: 1065.97 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances      114237               96.811  %
Incorrectly Classified Instances      3763                3.189  %
Kappa statistic                          0.9362
Mean absolute error                      0.0536
Root mean squared error                  0.1596
Relative absolute error                 10.7176 %
Root relative squared error             31.9172 %
Total Number of Instances           118000

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.977    0.041    0.960      0.977    0.968      0.936    0.993     0.993     neg
                 0.959    0.023    0.977      0.959    0.968      0.936    0.993     0.992     pos
Weighted Avg.    0.968    0.032    0.968      0.968    0.968      0.936    0.993     0.992

=== Confusion Matrix ===

     a     b   <-- classified as
 57670  1330 |     a = neg
  2433 56567 |     b = pos
```

Accuracy before SMOTE was 99.123% and after SMOTE is 96.811%

AUC and ROC Curve



ISLR 6.8.3) Suppose we estimate the regression coefficients in a linear regression model by minimizing

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 \quad \text{subject to} \quad \sum_{j=1}^{p}|\beta_j| \le s$$

for a particular value of s. For parts (a) through (e), indicate which of i. through v. is correct. Justify your answer.

a) As we increase *s* from 0, the training RSS will:

(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 β s is the maximum and it steadily decreases to the Ordinary Least Square RSS.

b) Repeat (a) for test RSS:

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

c) Repeat (a) for variance:

(iii) Steadily increase: When s=0, the model effectively predicts a constant and has almost no variance. As we increase s, the models includes more β s and their values start increasing. At this point, the values of β s become highly dependent on training data, thus increasing the variance.

d) Repeat (a) for (squared) bias:

(iv) Steadily decrease: When s=0, the model effectively predicts a constant and hence the prediction is far from actual value. Thus bias is high. As s increases, more β s become non-zero and thus the model continues to fit training data better. And thus, bias decreases.

e) Repeat (a) for the irreducible error:

(v) Remains constant: By definition, irreducible error is model independent and hence irrespective of the choice of s, remains constant.

ISLR 6.8.5) It is well-known that ridge regression tends to give similar coefficient values to correlated variables, whereas the lasso may give quite different coefficient values to correlated variables. We will now explore this property in a very simple setting.

Suppose that n = 2, p = 2, x11 = x12, x21 = x22. Furthermore, suppose that y1+y2 = 0 and x11+x21 = 0 and x12+x22 = 0, so that the estimate for the intercept in a least squares, ridge regression, or lasso model is zero: ˆ β0 = 0.

a) Write out the ridge regression optimization problem in this setting.

a) A general form of Ridge regression optimization looks like:

$$\text{Minimize}: \sum_{i=1}^{n}\left(y_i - \hat{\beta}_0 - \sum_{j=1}^{p}\hat{\beta}_j x_j\right)^2 + \lambda \sum_{i=1}^{p}\hat{\beta}_i^2$$

In this case $\hat{\beta}_0 = 0$ and $n = p = 2$. So, the optimization looks like:

~~Mi~~

$$\text{Minimize}: \left(y_1 - \hat{\beta}_1 x_{11} - \hat{\beta}_2 x_{12}\right)^2 + \left(y_2 - \hat{\beta}_1 x_{21} - \hat{\beta}_2 x_{22}\right)^2 + \lambda\left(\hat{\beta}_1^2 + \hat{\beta}_2^2\right)$$

b) Argue that in this setting, the ridge coefficient estimates satisfy ˆ β1 = ˆ β2.

b) Now we are given that $x_{11} = x_{12} = x_1$ and $x_{21} = x_{22} = x_2$. We take derivative of above expressions with respect to both $\hat{\beta}_1$ and $\hat{\beta}_2$ and setting them equal to zero find that

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

Symmetry in these expressions suggests that $\hat{\beta}_1^* = \hat{\beta}_2^*$

c) Write out the lasso optimization problem in this setting.

c) Like Ridge regression,

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

$\lambda(|\hat{\beta}_1| + |\hat{\beta}_2|)$

d) Argue that in this setting, the lasso coefficients ˆ β_1 and ˆ β_2 are not unique—in other words, there are many possible solutions to the optimization problem in (c). Describe these solutions.

d) Here is a geometric interpretation of the solns for the equation in c above. We use the alternate form of Lasso constraints $|\hat{\beta}_1| + |\hat{\beta}_2| < s$. The Lasso constraint take the form $|\hat{\beta}_1| + |\hat{\beta}_2| < s$, which when plotted take the familar shape of a diamond centered at origin $(0,0)$. Next consider the squared optimization constraint $(y_1 - \hat{\beta}_1 x_{11} - \hat{\beta}_2 x_{12})^2 + (y_2 - \hat{\beta}_1 x_{21} - \hat{\beta}_2 x_{22})^2$. We use the facts $x_{11} = x_{12}$, $x_{21} = x_{22}$, $x_{11} + x_{21} = 0$, $x_{12} + x_{22} = 0$ & $y_1 + y_2 = 0$ to simplify it to

Minimize : $2 \cdot (y_1 - (\hat{\beta}_1 + \hat{\beta}_2) x_{11})^2$

This optimization problem has a simple soln: $\hat{\beta}_1 + \hat{\beta}_2 = \frac{y_1}{x_{11}}$. This is a line parallel to the edge of Lasso-diamond $\hat{\beta}_1 + \hat{\beta}_2 = s$. Now soln to the original of Lasso optimization problem are contours of the function $(y_1 - (\hat{\beta}_1 + \hat{\beta}_2) x_{11})^2$ that touch the Lasso-diamond $\hat{\beta}_1 + \hat{\beta}_2 = s$. Finally as $\hat{\beta}_1$ and $\hat{\beta}_2$ very along the line $\hat{\beta}_1 + \hat{\beta}_2 = \frac{y_1}{x_{11}}$

③

Camlin Page

Date / /

these contours touch the Lasso-diamond edge $\hat{\beta}_1 + \hat{\beta}_2 = s$ at different points. As a result, the entire edge $\hat{\beta}_1 + \hat{\beta}_2 = s$ is a potential soln to the Lasso optimization problem!

Similar argument can be made for the opposite Lasso-diamond edge: $\hat{\beta}_1 + \hat{\beta}_2 = -s$

Thus, the Lasso problem does not have a unique soln. The general form of soln is given by 2 line segments:

$\hat{\beta}_1 + \hat{\beta}_2 = s$; $\hat{\beta}_1 \geq 0$; $\hat{\beta}_2 \geq 0$ and $\hat{\beta}_1 + \hat{\beta}_2 = -s$, $\hat{\beta}_1 \leq 0$, $\hat{\beta}_2 \leq 0$

ISLR 8.4.5) Suppose we produce ten bootstrapped samples from a data set containing red and green classes. We then apply a classification tree to each bootstrapped sample and, for a specific value of X, produce 10 estimates of P(Class is Red|X):0.1, 0.15, 0.2, 0.2, 0.55, 0.6, 0.6, 0.65, 0.7, and 0.75.

There are two common ways to combine these results together into a single class prediction. One is the majority vote approach discussed in this chapter. The second approach is to classify based on the average probability. In this example, what is the final classification under each of these two approaches?

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

A) Majority approach

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

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

B) Average approach

mean(p)

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

ISLR 9.7.3) Here we explore the maximal margin classifier on a toy data set.

Obs.	X_1	X_2	Y
1	3	4	Red
2	2	2	Red
3	4	4	Red
4	1	4	Red
5	2	1	Blue
6	4	3	Blue
7	4	1	Blue

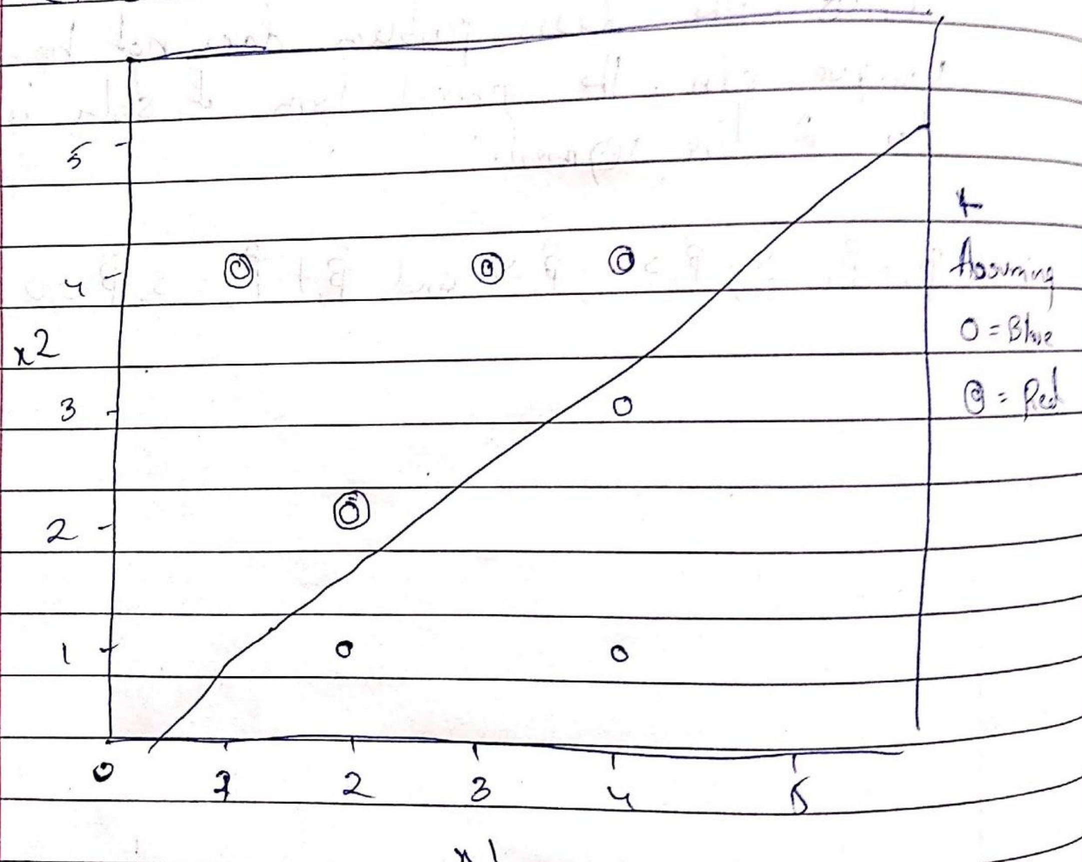
Sketch the observations.

a) We are given n = 7 observations in p = 2 dimensions. For each observation, there is an associated class label.

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")

b) Sketch the optimal separating hyperplane, and provide the equation for this hyperplane

b) (a) & (b)



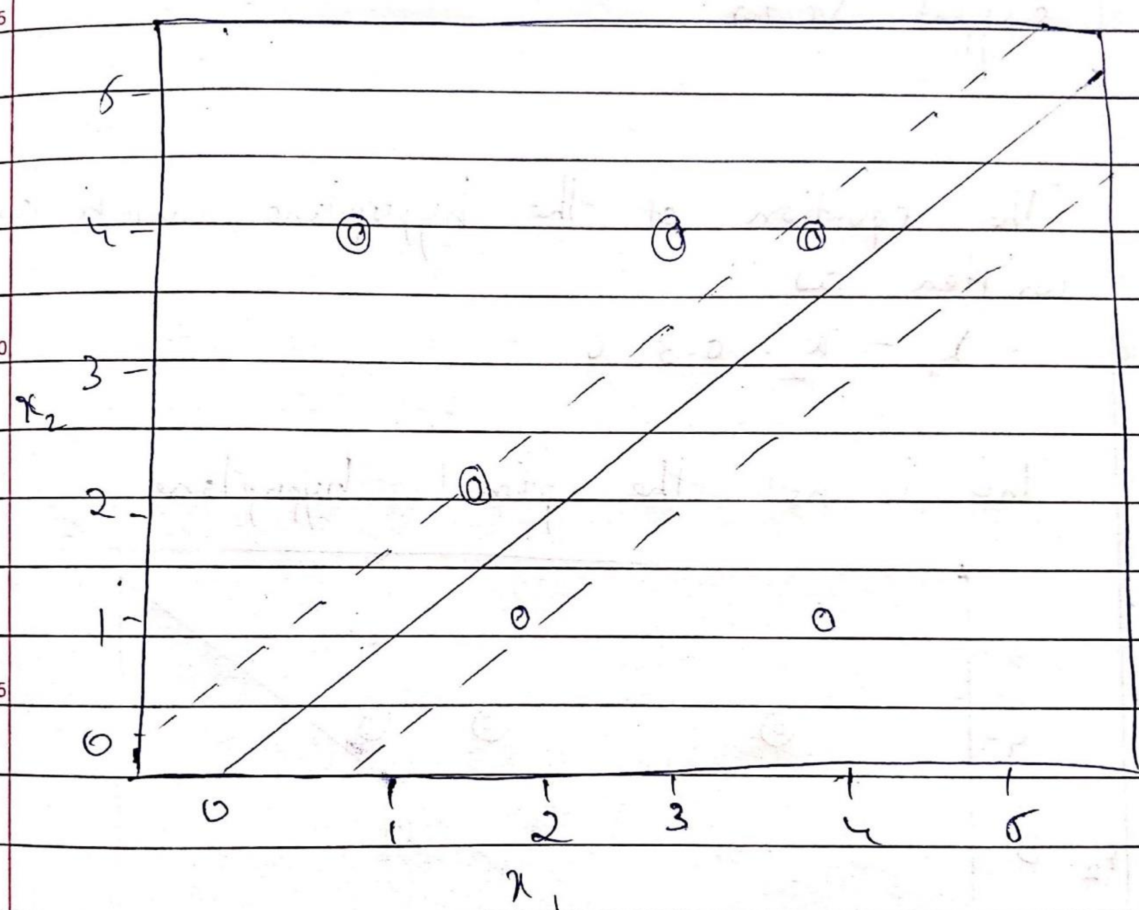
As shown above, the optical separating hyperplane has to be shown betwen (2,1) and (2,2) & between (4,3) and (4,4)

c) Describe the classification rule for the maximal margin classifier. It should be something along the lines of “Classify to Red if β0 +β1X1 +β2X2 > 0, and classify to Blue otherwise.” Provide the values for β0, β1, and β2.

c) Classification Rule is:
→ Classified to class "RED" if $X_1 - X_2 - 0.5 < 0$
and to class "Blue" otherwise.

d) On your sketch, indicate the margin for the maximal margin hyperplane.

d)



* Assuming o = blue, ⊚ = Red
The margin here is equal to 1/4

e) Indicate the support vectors for the maximal margin classifier.

e) The support vectors are the point at (2,1), (2,2), (4,3) and (4,4)

f) Argue that a slight movement of the seventh observation would not affect the maximal margin hyperplane.

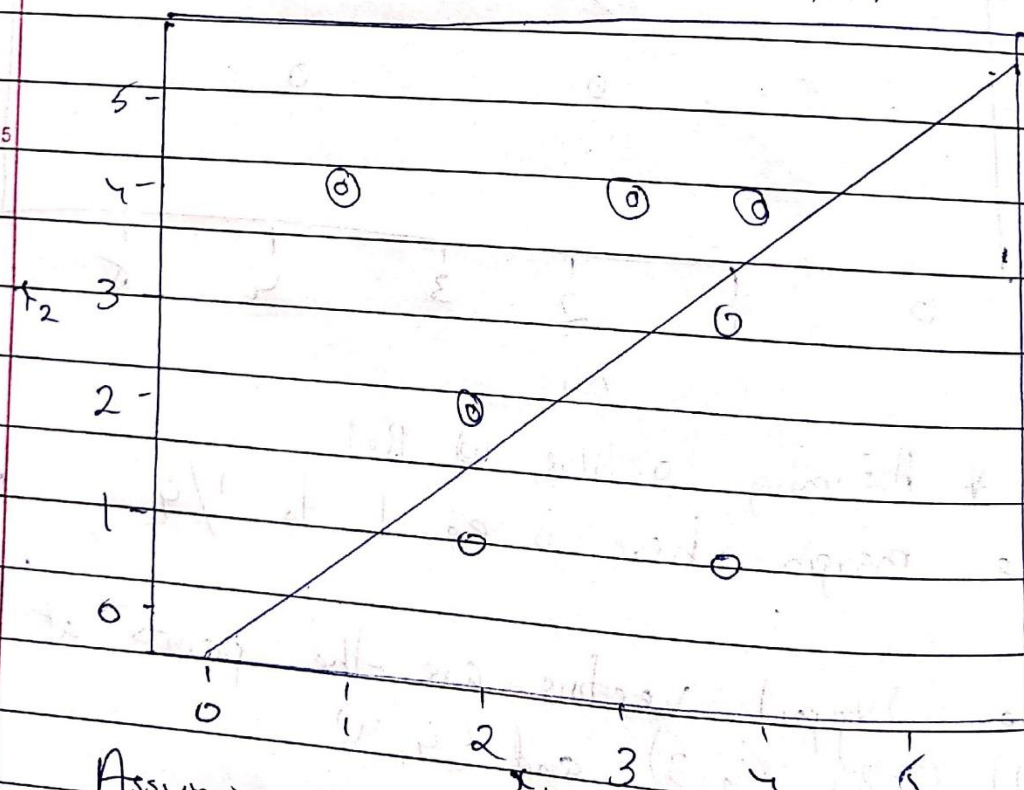
f) By looking at the plot, we can come to understanding that, if we moved point (4,1), it would end up not making any change to the maximal margin hyperplane lines if is not a support vector.

g) Sketch a hyperplane that is not the optimal separating hyperplane, and provide the equation for this hyperplane.

g) The equation of the hyperplane, which can be written as:

$$x_1 - x_2 - 0.3 = 0$$

This is not the optimal hyperplane

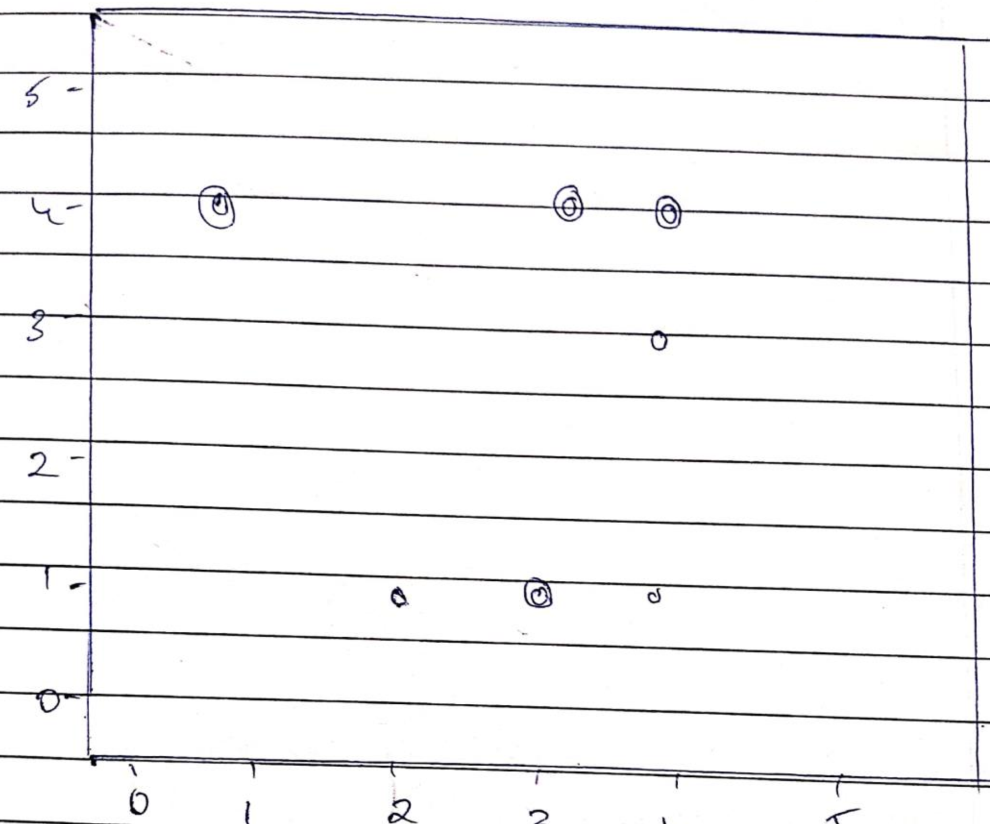


Assuming O = Blue, ⊙ = Red

h) Draw an additional observation on the plot so that the two classes are no longer separable by a hyperplane.

Date / /

h) when the red point (3,1) is added to the plot, the two c-lanes are obviously not separable



Assuming O = Blue, ⊚ = Red