CS464-MACHINE LEARNING HW2

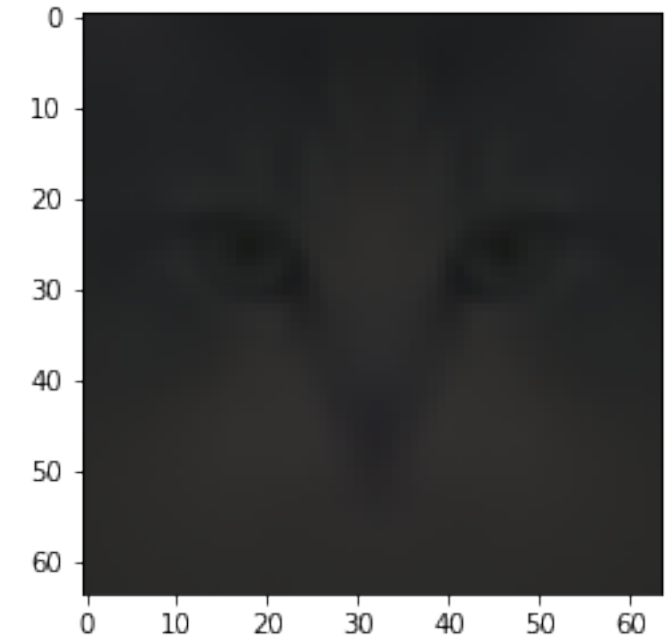
FIRAT YILDIZ
21502717
Section 02

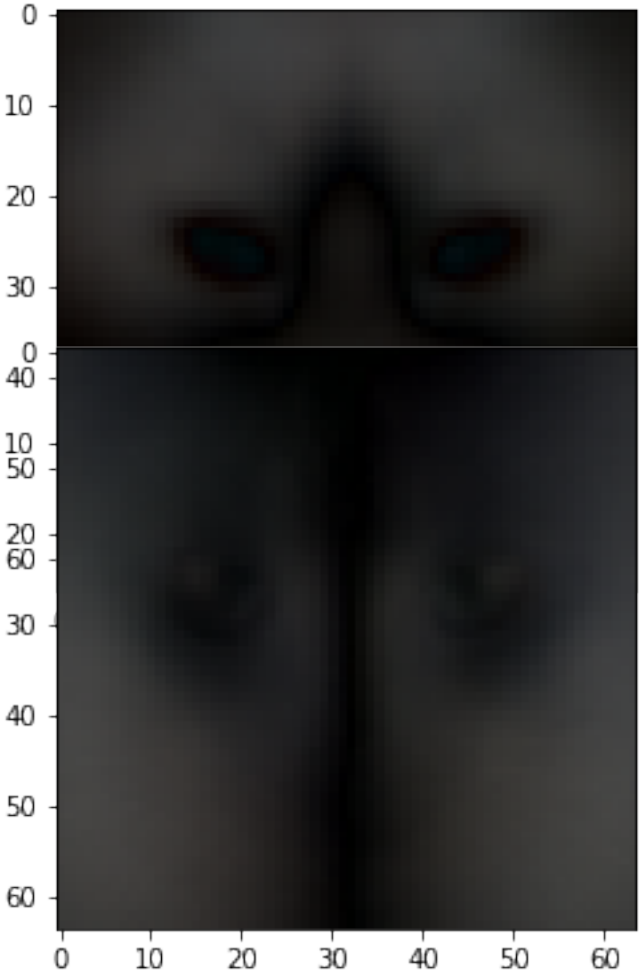
Q1)

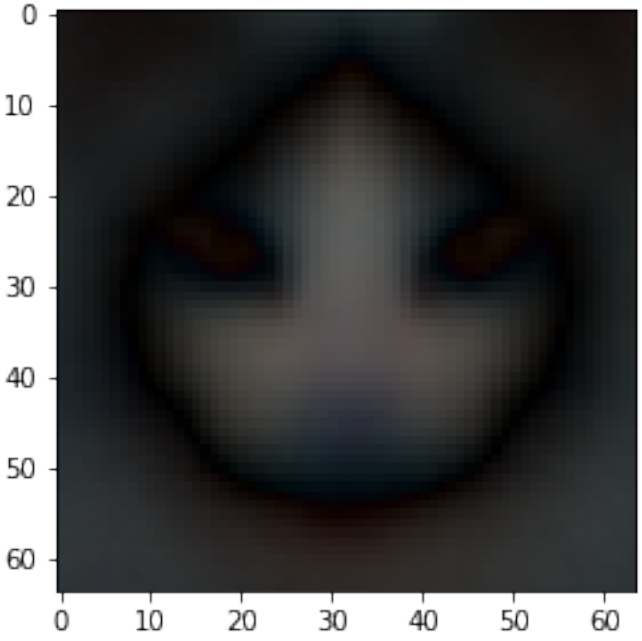
Q1.1) Both of the methods SVD (Singular Value Decomposition) and PCA (Principal Component Analysis) are used to reduce the dimension of the features without losing the important information. Also, calculation of the covariance matrix is required in the computation of PCA and SVD. In the PCA, when the covariance matrix is calculated, then eigenvalues and eigenvectors can be obtained. After that, eigenvalues and eigenvectors are ranked according their importance (i.e according to magnitude of eigenvalues). In the SVD case, SVD of covariance matrix is [U S V'] . To obtain important features, there is no need to rank according to eigenvalues as important features are ranked and can be obtained within the U matrix[2]. This is the main difference between SVD and PCA. We print the PVE values in order to understand at what degree the information lost by projection of the data. Because we express big data with small number of components, there is some loss and the output shows the loss for each color value for first 10 photos. That is the explanation of variance of Principal Component so less variance means less loss in the data. For example, we see more loss in the color of red at first photo compared to second one. As my expectation, by the increase of Principal Components PVE value drops. We expect this result because PVE gives the percentage of principle components so when it gets bigger, having data is expected to decrease.

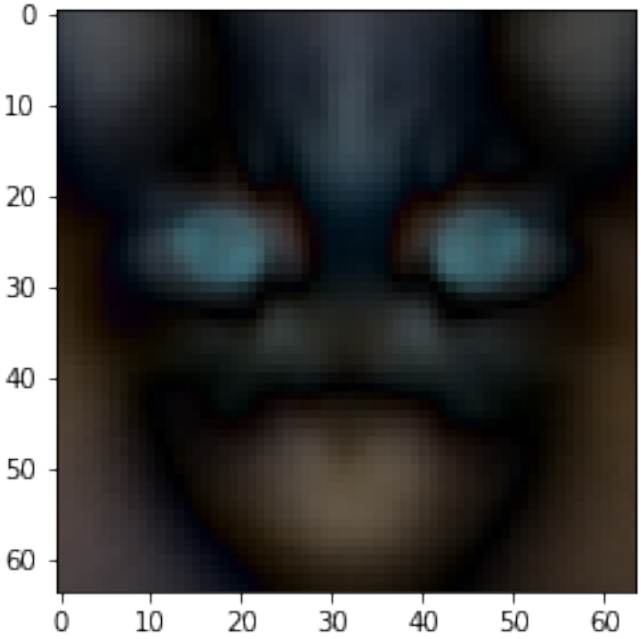
```
PVE RATIO FOR RED: [0.8173209  0.02851471 0.02443124 0.01644898 0.00877841 0.00612892
 0.00581728 0.00466474 0.00374913 0.00332108]
PVE RATIO FOR GREEN: [0.84269305 0.02530509 0.0213064  0.01386851 0.00723721 0.00531647
 0.00505388 0.0038718  0.0035987  0.00295344]
PVE RATIO FOR BLUE: [0.86574708 0.02321492 0.01840561 0.01223756 0.00564718 0.00451281
 0.0043853  0.00344601 0.00305898 0.00247767]
```

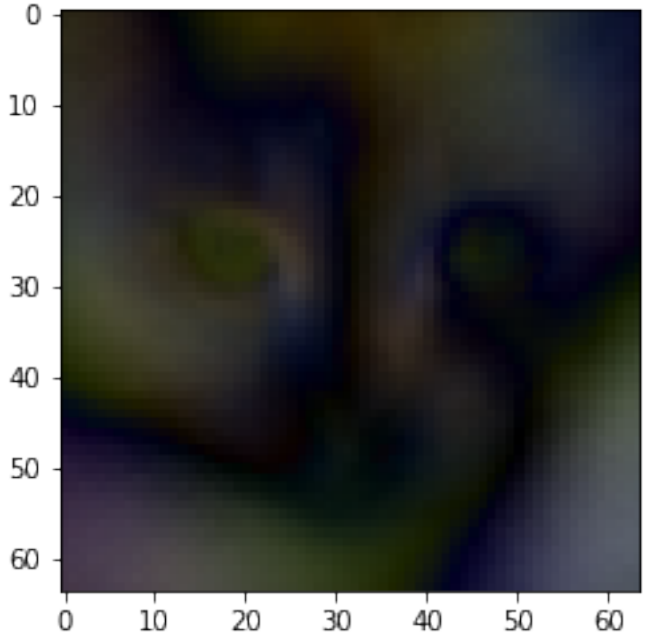
Q1.2)

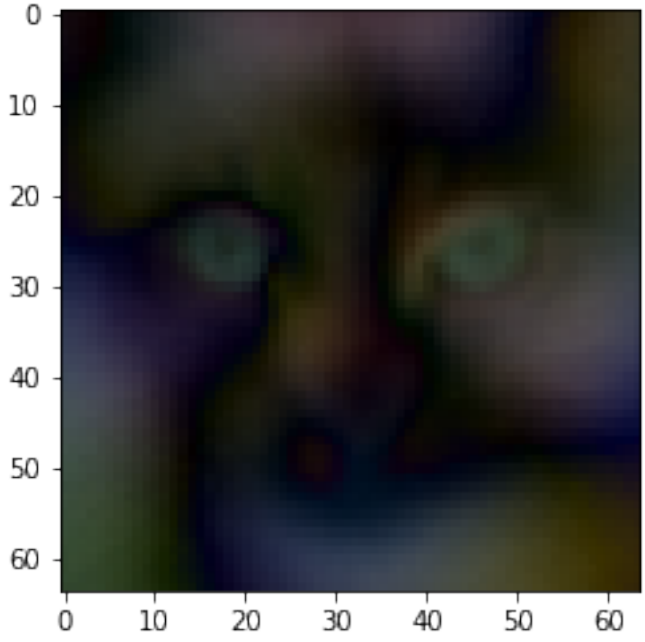

0
10
20
30
40
50
60
0
10
20
30
40
50
60

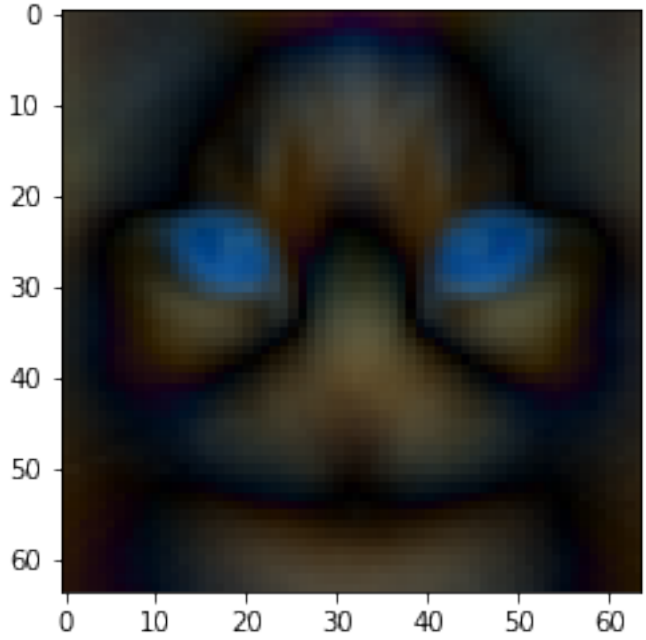

0
10
20
30
0
40
10
50
20
60
30
40
50
60
0
10
20
30
40
50
60

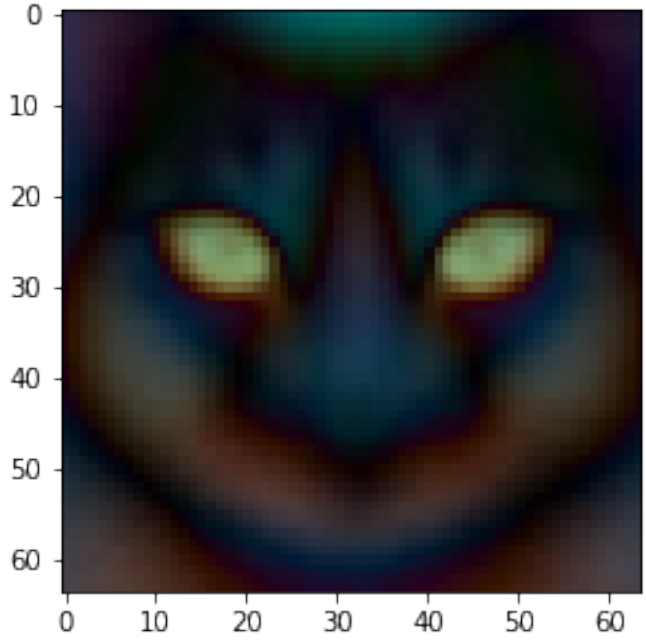

0
10
20
30
40
50
60
0
10
20
30
40
50
60

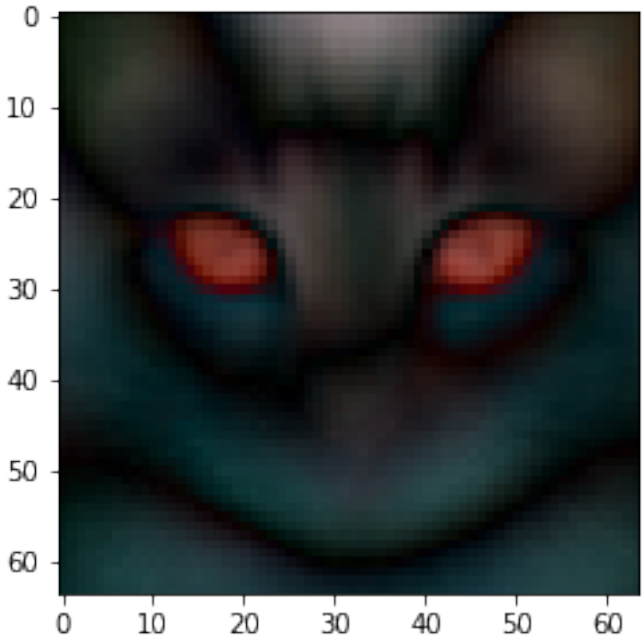

0
10
20
30
40
50
60
0
10
20
30
40
50
60


0
10
20
30
40
50
60
0
10
20
30
40
50
60


0
10
20
30
40
50
60
0
10
20
30
40
50
60


0
10
20
30
40
50
60
0
10
20
30
40
50
60

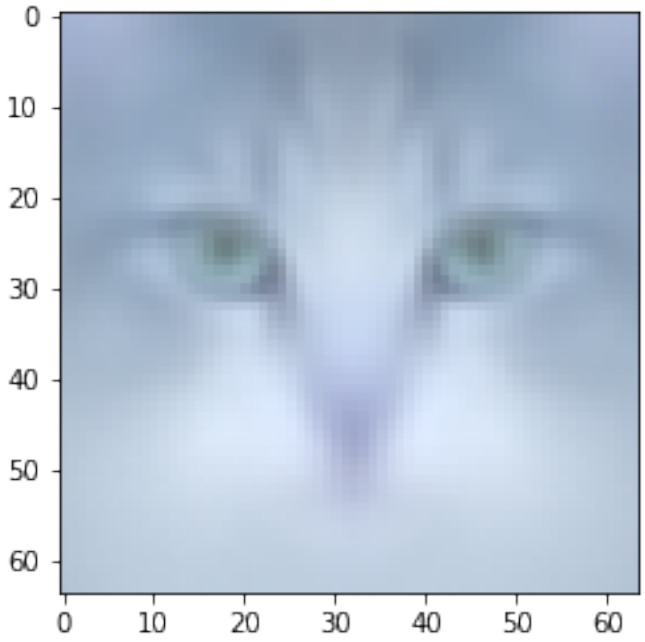

0
10
20
30
40
50
60
0
10
20
30
40
50
60



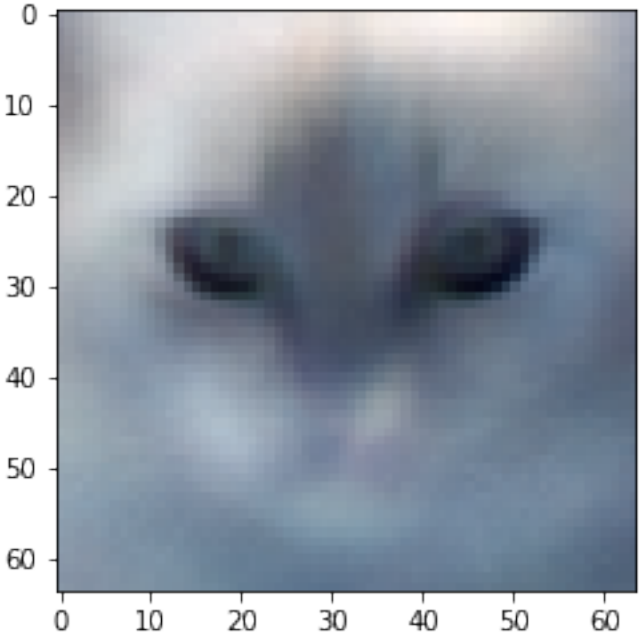
PCA applied first 10 images by order(first photo is 1 and second is 2 so on) were shown at the top. They were first reshaped from 64x64x3 to 4096x3 where 3 represents the color values. After PCA, we can express the photos with less number of components which makes our analysis easier about the photo. As seen photos are not as clear as in data set. This is the effect of PCA meaning analysis to express data with true few number of components.

Q1.3) In this part we are asked to use first 1, 50, 250 and 500 principal components to reconstruct the data. As we know all components offers and analysis for the data. If we increase the number of principal components used to express the data, we get better reconstruction as seen below.

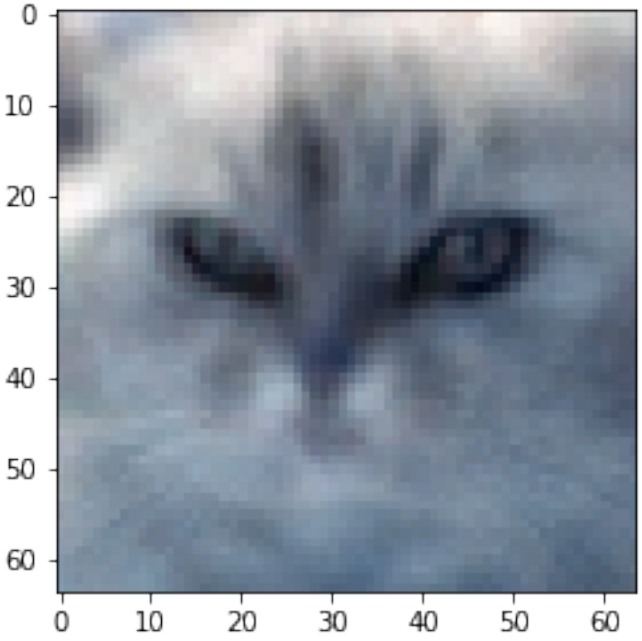
k = 1



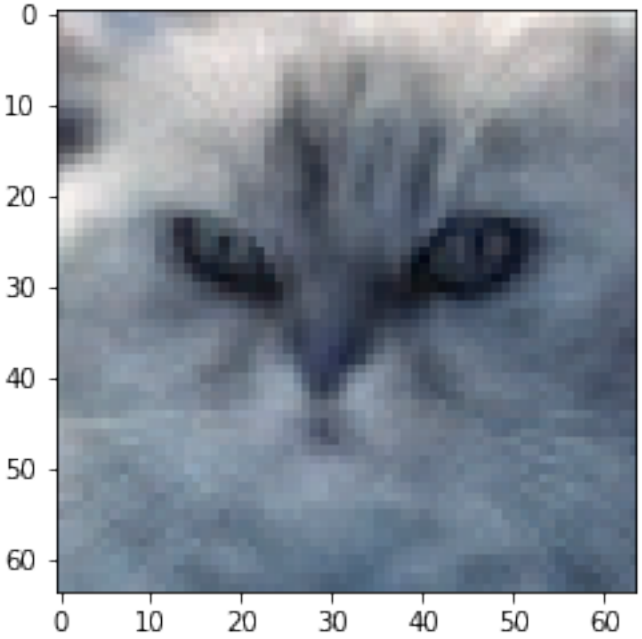
k = 50


0
10
20
30
40
50
60
0
10
20
30
40
50
60

k = 250


0
10
20
30
40
50
60
0
10
20
30
40
50
60

k = 500



Q 2)

Q 2.1)

$$Jn = \| y - X\beta \|^2 = (y - X\beta)^T (y - X\beta)$$

$$Jn = y^Ty - 2\beta^Tx^Ty + \beta^Tx^Tx\beta$$

$$\partial/\partial\beta\ (Jn) = \partial/\partial\beta\ (y^Ty) - 2\beta^Tx^Ty + \beta^Tx^Tx\beta = 0$$

$$\partial/\partial\beta\ (Jn) = -2\ x^Ty + 2\ x^Tx\beta$$

From here we get by dividing two sides by two and move first term other side:

$$x^Tx\beta = x^Ty$$

After that we multiply both sides to $(x^Tx)^{-1}$ and get β equalized to:

$$\beta = (x^Tx)^{-1}\ x^Ty$$

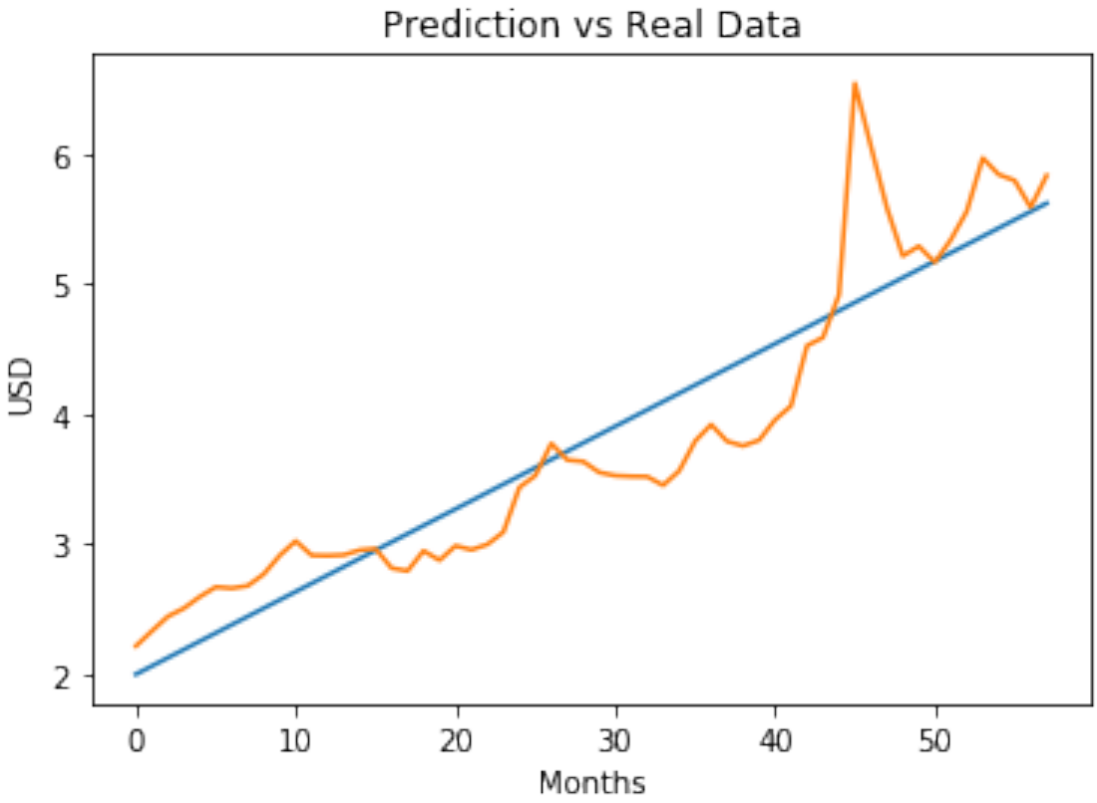
Q 2.2) For model A we have one feature deciding the prediction labels. So we have w0 and w1 defined below as

W0 and W1: [[2.00237154] [0.06342816]]

Those coefficients will be used in the calculation of regression line expressed in formula 2.1 in the assignment description. We are asked to define the MSE value which is mean squared error. That defines how close our regression line is close to data. We can interpret the data with less MSE value because of this. MSE value for model A is

mse: [[0.1878714]]

USD/TRY vs Months Past Since November 2014 for model A is



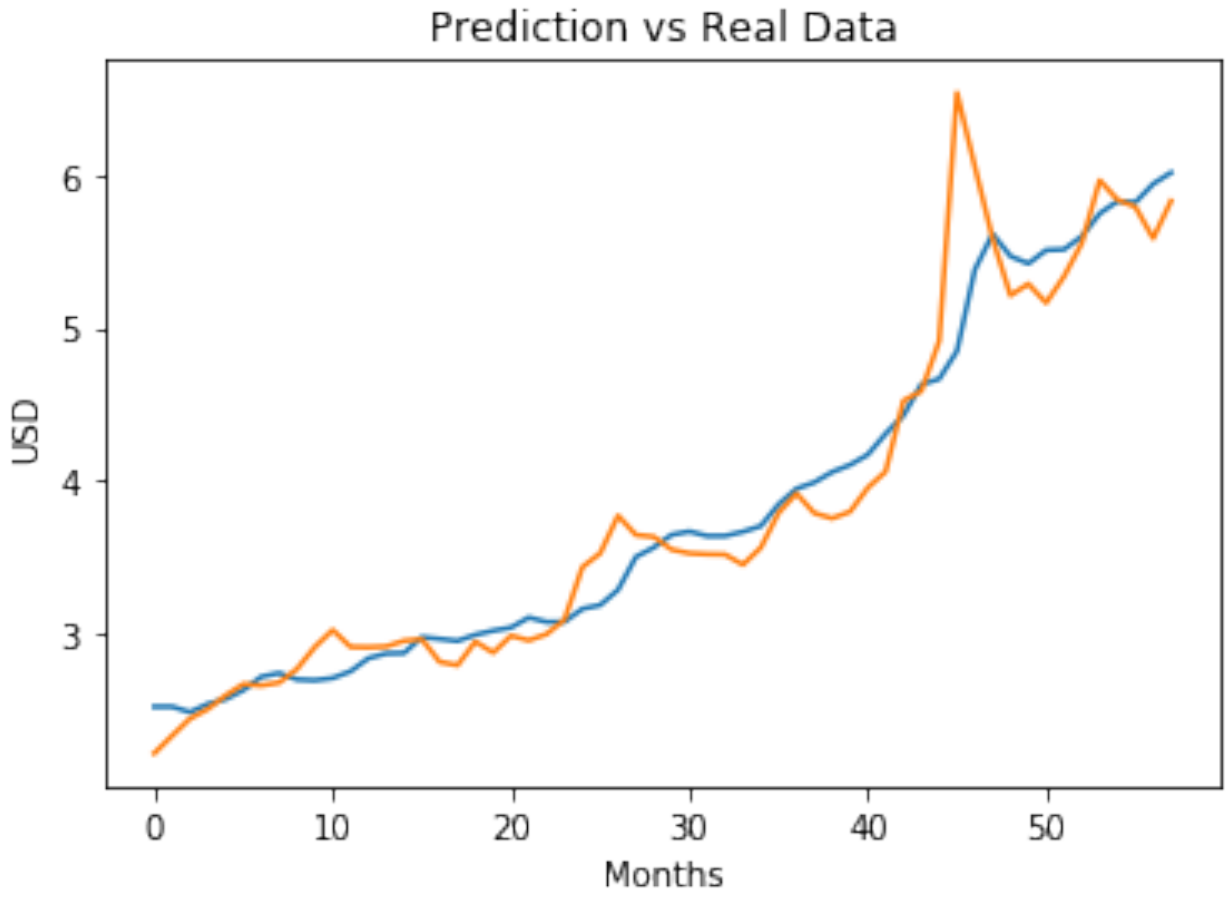
Q 2.3) For model B we have two features deciding the prediction labels. So we have w0, w1 and w2 by order defined below as:

W's:[[-3.28416548][-0.01032575[0.02338648]]

MSE value for model B is

mse: [[0.08914426]]

USD/TRY vs Months Past Since November 2014 for model B is:



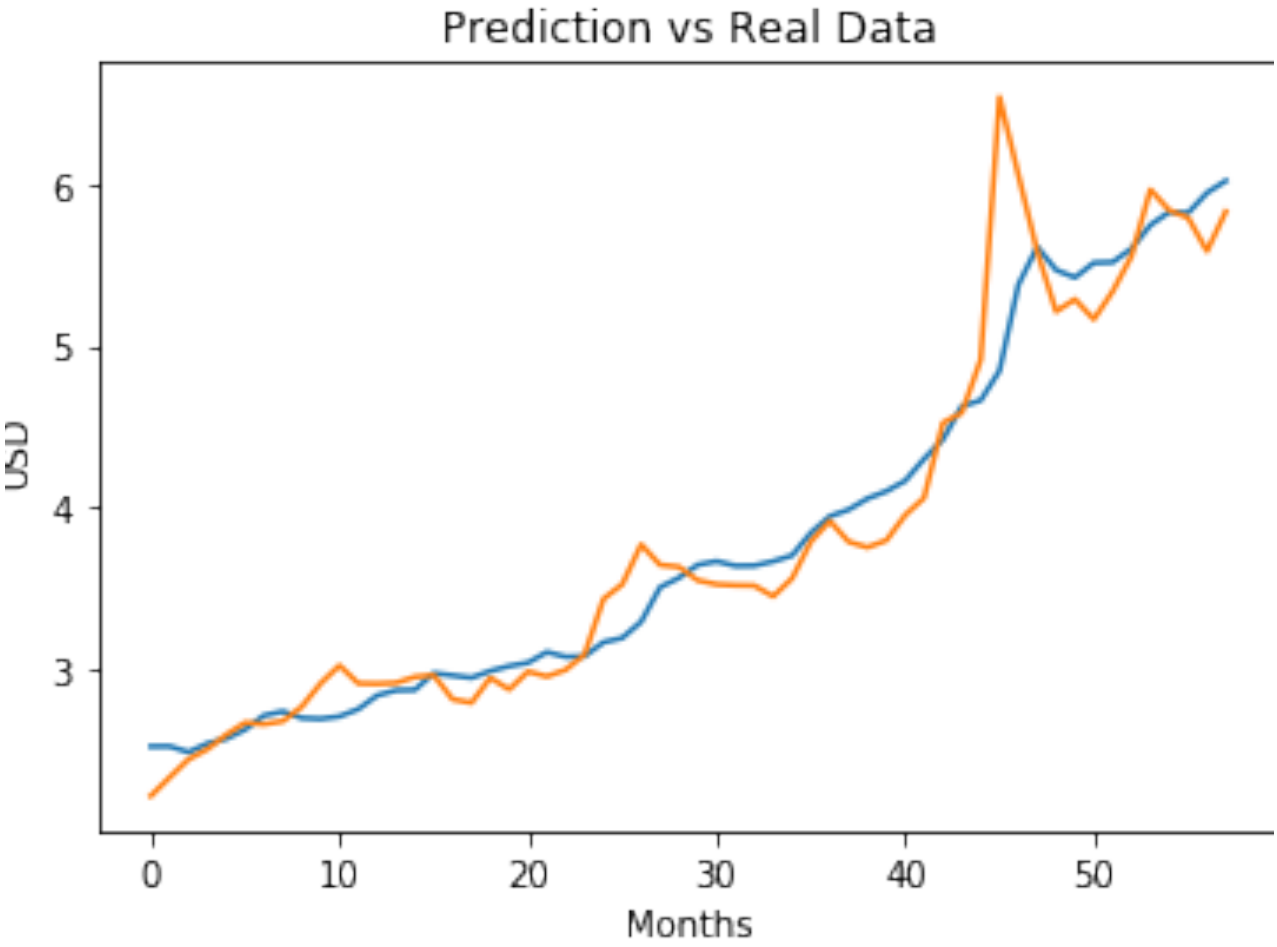
Q 2.4) For model C we have three features deciding the prediction labels. So we have w0, w1, w2 and w3 by order defined below as:

W's:[[-3.29005917e+00][-1.01887495e-02]
[2.32930116e-02[1.53083663e-03]]

MSE value for model C is

mse: [[0.08913535]]

USD/TRY vs Months Past Since November 2014 for model C is:



Q 2.5) The MSE values for model A, B and C are

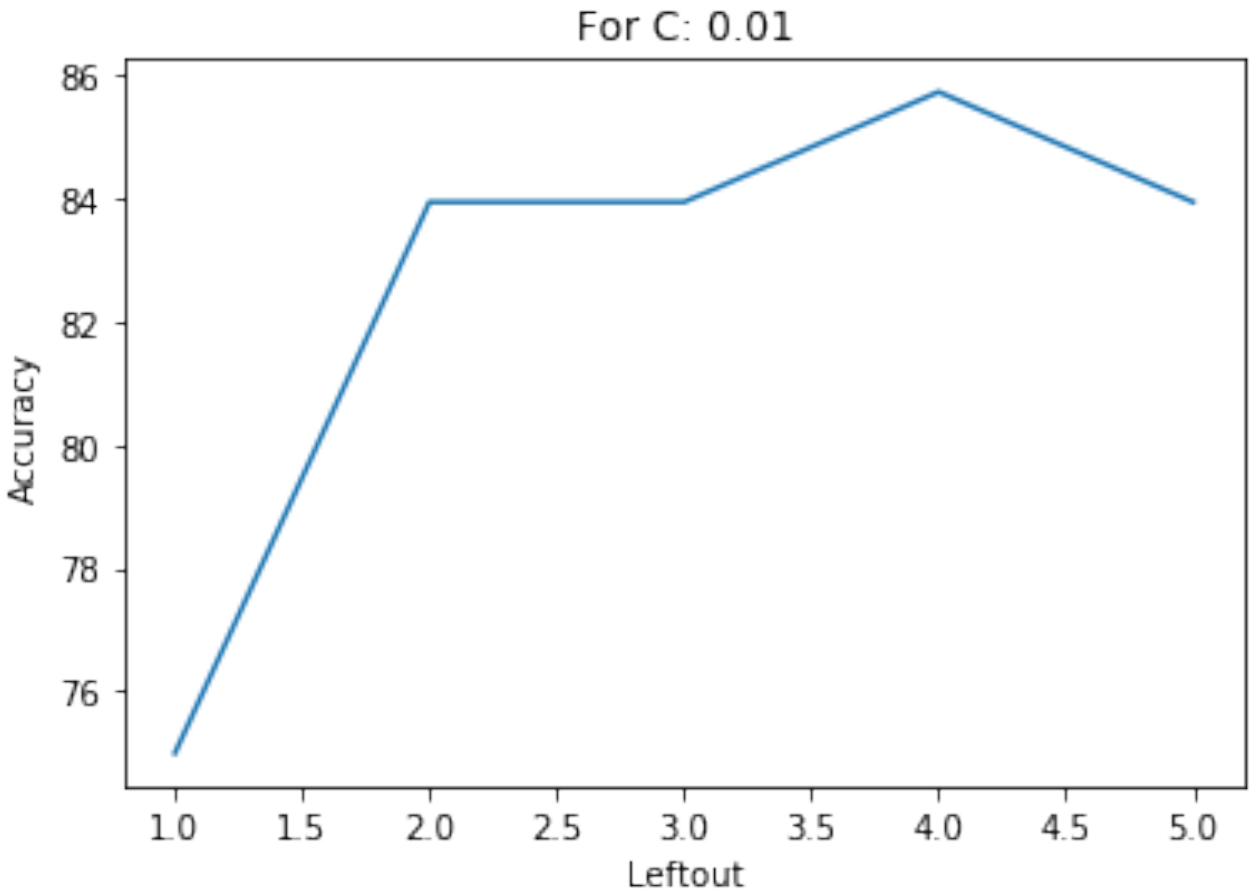
mse for model C: [[0.1055575]]
mse for model A: [[0.06305787]]
mse for model B: [[0.08556457]]

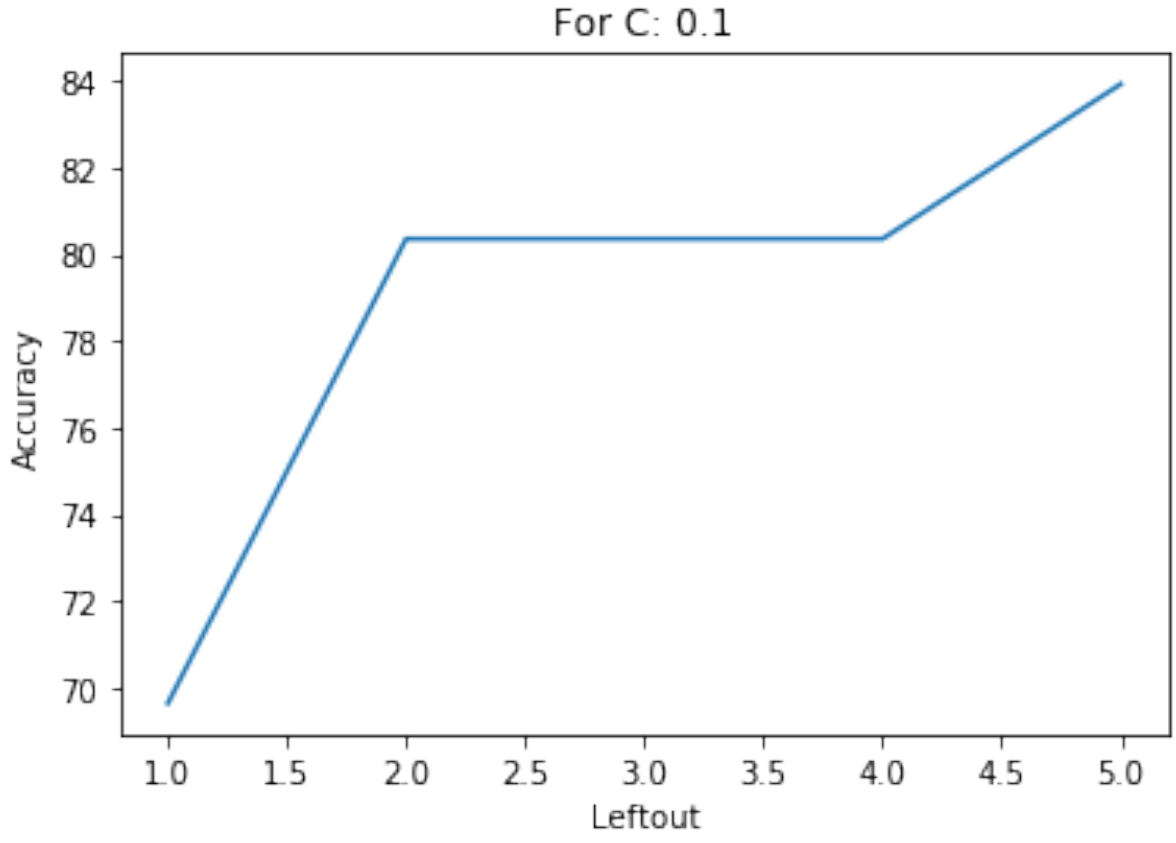
From this result we get the MSE values for different model. According to this data we prefer to use model A because its error of mean square is less than others. That means the regression line drawn by this is closer to data. These models are promising in prediction of exchange rates because MSE values are really low in USD/TRY rate prediction. Those models may be used.

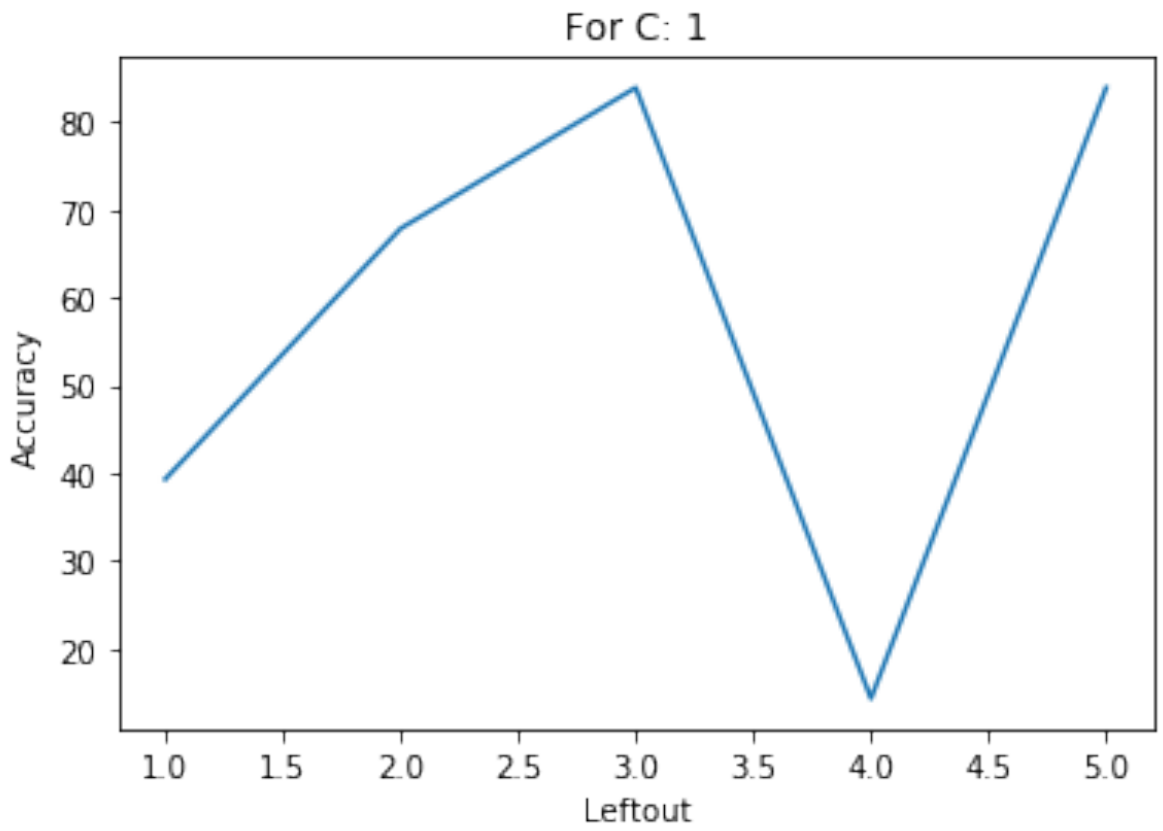
Q 2.6) In this case when we enter more data the regression becomes worse as not expected. That is because the features are not good to make prediction. Also the time affects our predictions so the predictions should be out of time.

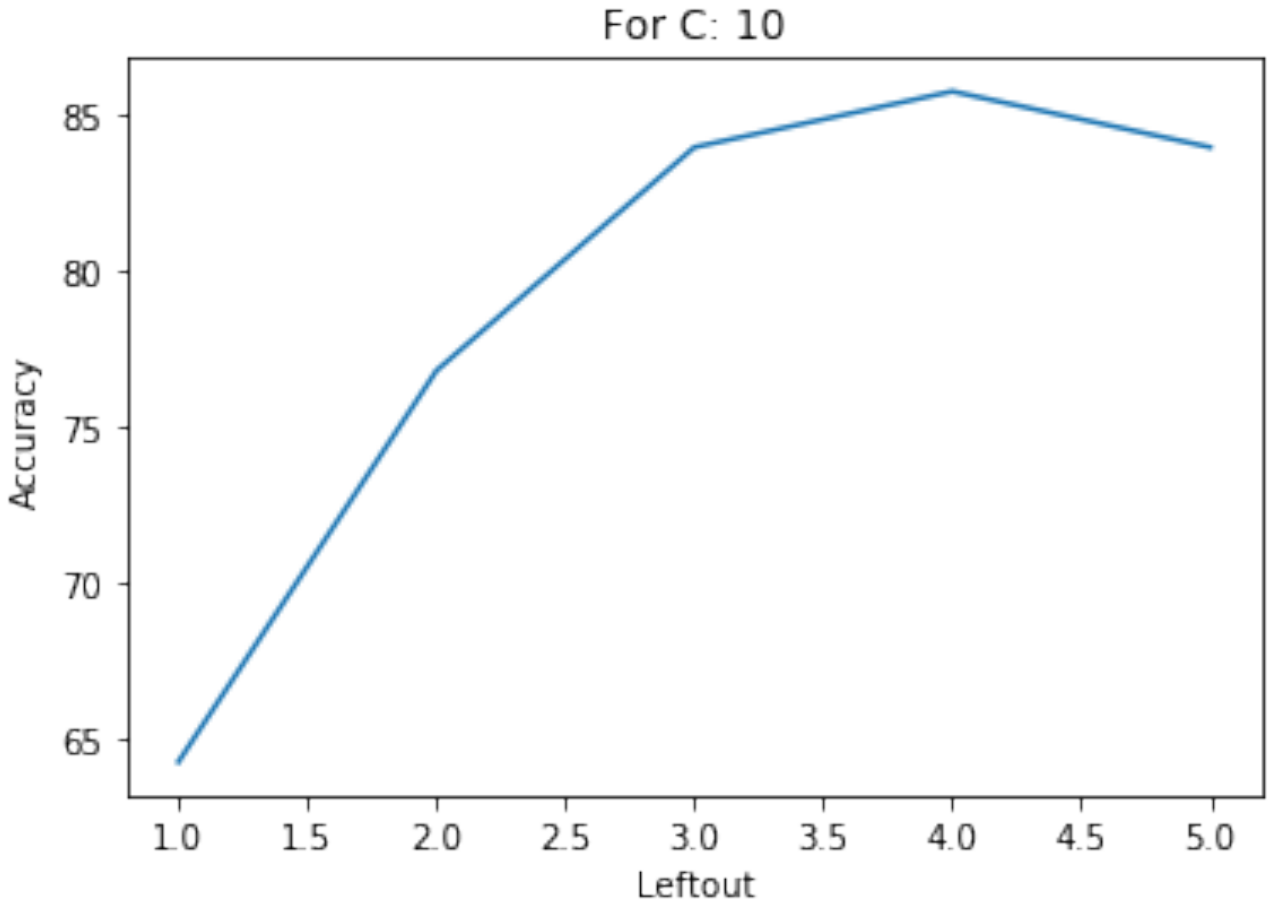
Q 3)

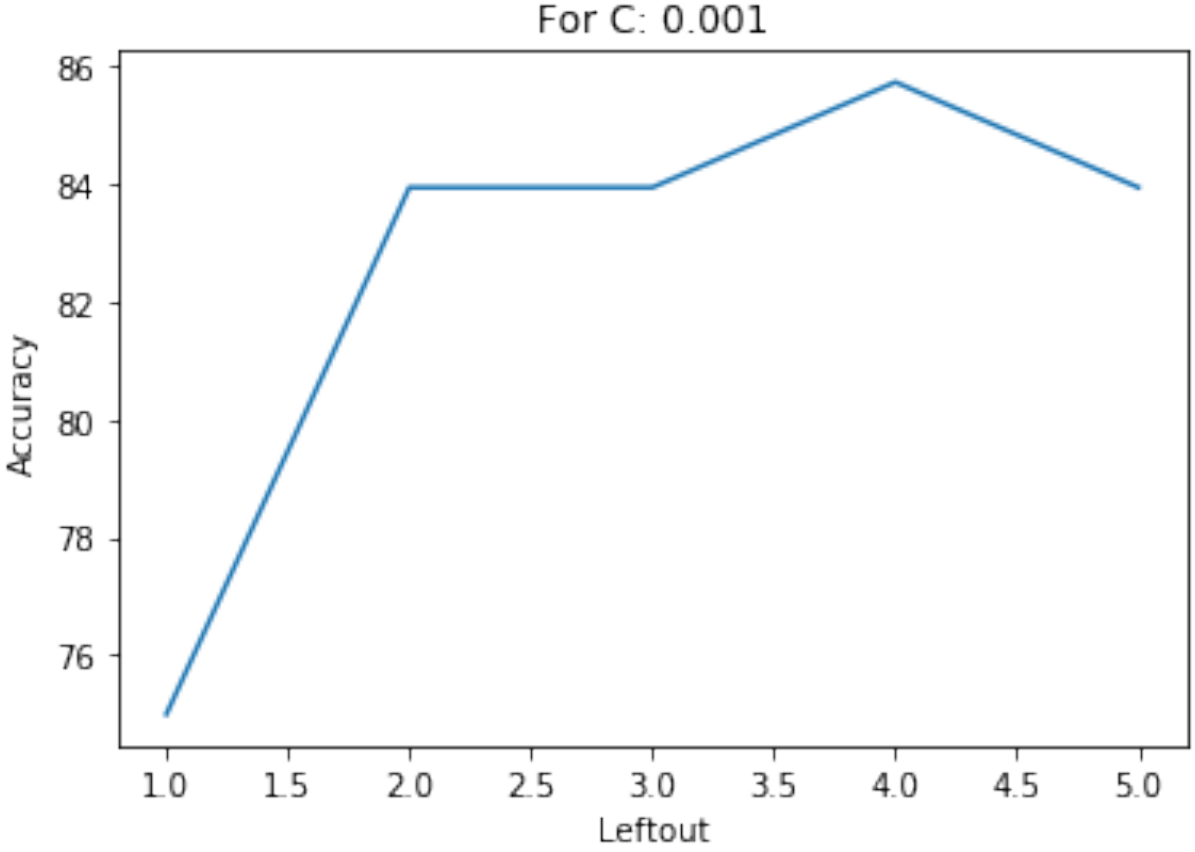
Q 3.1) For this part of question 3, the $[10^{-3}, 10^{-2}, 10^{-1}, 10, 10^{1}, 10^{2}]$ values were used as C values. For all C values Left-out folded vs Accuracy plots stemmed. Micro and macro averages of precision, re- call, negative predictive value (NPV), false positive rate (FPR), false discovery rate (FDR), F1 and F2 scores calculated. We use the optimal C as hyper parameter. We calculate its optimality with 5 folded cross validation.


For C: 0.01
Accuracy
86
84
82
80
78
76
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Leftout


For C: 0.1
Accuracy
84
82
80
78
76
74
72
70
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Leftout


For C: 1
Accuracy
80
70
60
50
40
30
20
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Leftout


For C: 10
Accuracy
85
80
75
70
65
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Leftout


For C: 0.001
86
84
82
80
78
76
Accuracy
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Leftout

```
Accuracy with different C values: [82.5        82.5
78.92857143 57.85714286 78.92857143 76.78571429]

True positive for C(0.01): 0
True negative for C(0.01): 95
False negative for C(0.01): 0
False negative for C(0.01): 20
Recall: 0.0
Precision: 0
FDR: 0
NPV: 0.8260869565217391
FPR: 0.0
f1: 0
f2: 0
REAL->     Pos.        Neg.
     |--------------------------|
Pos. |    0        0            |
Neg. |    20        95           |
     |--------------------------|
Total Accuracy with best C(0.01)= 82.6086956521739

True positive for C(0.1): 2
True negative for C(0.1): 94
False negative for C(0.1): 1
```

```
False negative for C(0.1): 18
Recall: 0.1
Precision: 0.6666666666666666
FDR: 0.3333333333333333
NPV: 0.8392857142857143
FPR: 0.010526315789473684
f1: 1.7391304347826086
f2: 1.2048192771084336
REAL->     Pos.        Neg.
     |--------------------------|
Pos. |    2         1           |
Neg. |    18         94          |
     |--------------------------|
Total Accuracy with best C(0.1)= 83.47826086956522

True positive for C(1): 3
True negative for C(1): 92
False negative for C(1): 3
False negative for C(1): 17
Recall: 0.15
Precision: 0.5
FDR: 0.5
NPV: 0.8440366972477065
FPR: 0.031578947368421054
f1: 1.5384615384615383
f2: 1.1627906976744187
REAL->     Pos.        Neg.
     |--------------------------|
Pos. |    3         3           |
Neg. |    17         92          |
     |--------------------------|
Total Accuracy with best C(1)= 82.6086956521739

True positive for C(10): 6
True negative for C(10): 66
False negative for C(10): 29
False negative for C(10): 14
Recall: 0.3
```

```
Precision: 0.17142857142857143
FDR: 0.8285714285714286
NPV: 0.825
FPR: 0.30526315789473685
f1: 0.7272727272727273
f2: 0.8695652173913043
REAL->     Pos.        Neg.
     |--------------------------|
Pos. |    6        29           |
Neg. |    14        66           |
     |--------------------------|
Total Accuracy with best C(10)= 62.60869565217392

True positive for C(100): 12
True negative for C(100): 31
False negative for C(100): 64
False negative for C(100): 8
Recall: 0.6
Precision: 0.15789473684210525
FDR: 0.8421052631578947
NPV: 0.7948717948717948
FPR: 0.6736842105263158
f1: 0.4166666666666667
f2: 0.6410256410256411
REAL->     Pos.        Neg.
     |--------------------------|
Pos. |    12        64           |
Neg. |    8        31           |
     |--------------------------|
Total Accuracy with best C(100)= 37.391304347826086

Micro averages precision for gamma values:
0.19166666666666668
Macro averages precision for gamma values: 0.0
True positive for C(0.001): 0
True negative for C(0.001): 95
False negative for C(0.001): 0
False negative for C(0.001): 20
```

```
Recall: 0.0
Precision: 0
FDR: 0
NPV: 0.8260869565217391
FPR: 0.0
f1: 0
f2: 0
REAL->     Pos.         Neg.
     |--------------------------|
Pos. |    0         0           |
Neg. |    20         95           |
     |--------------------------|
Total Accuracy with best C(0.001)= 82.6086956521739
```

Q 3.2) For this part of question 3, the $[2^{-4}, 2^{-3}, 2^{-2}, 2^{-1}, 2^{0}, 2^{1}]$ values were used as Gamma values. Micro and macro averages of precision, recall, negative predictive value (NPV), false positive rate (FPR), false discovery rate (FDR), F1 and F2 scores calculated. We use the optimal Gamma as hyper parameter. We calculate its optimality with 5 folded cross validation.

```
True positive for Gamma(0.125): 0
True negative for Gamma(0.125): 95
False negative for Gamma(0.125): 0
False negative for Gamma(0.125): 20
Recall: 0.0
Precision: 0
FDR: 0
NPV: 0.8260869565217391
```

```
FPR: 0.0
f1: 0
f2: 0
REAL->     Pos.        Neg.
     |--------------------------|
Pos. |    0        0            |
Neg. |    20        95           |
     |--------------------------|
Total Accuracy with Gamma(0.125)= 82.6086956521739

True positive for Gamma(0.25): 0
True negative for Gamma(0.25): 95
False negative for Gamma(0.25): 0
False negative for Gamma(0.25): 20
Recall: 0.0
Precision: 0
FDR: 0
NPV: 0.8260869565217391
FPR: 0.0
f1: 0
f2: 0
REAL->     Pos.        Neg.
     |--------------------------|
Pos. |    0        0            |
Neg. |    20        95           |
     |--------------------------|
Total Accuracy with Gamma(0.25)= 82.6086956521739

True positive for Gamma(0.5): 0
True negative for Gamma(0.5): 95
False negative for Gamma(0.5): 0
False negative for Gamma(0.5): 20
Recall: 0.0
Precision: 0
FDR: 0
NPV: 0.8260869565217391
FPR: 0.0
f1: 0
```

```
f2: 0
REAL->     Pos.        Neg.
     |--------------------------|
Pos. |    0        0            |
Neg. |    20        95           |
     |--------------------------|
Total Accuracy with Gamma(0.5)= 82.6086956521739

True positive for Gamma(1): 0
True negative for Gamma(1): 95
False negative for Gamma(1): 0
False negative for Gamma(1): 20
Recall: 0.0
Precision: 0
FDR: 0
NPV: 0.8260869565217391
FPR: 0.0
f1: 0
f2: 0
REAL->     Pos.        Neg.
     |--------------------------|
Pos. |    0        0            |
Neg. |    20        95           |
     |--------------------------|
Total Accuracy with Gamma(1)= 82.6086956521739

True positive for Gamma(2): 0
True negative for Gamma(2): 95
False negative for Gamma(2): 0
False negative for Gamma(2): 20
Recall: 0.0
Precision: 0
FDR: 0
NPV: 0.8260869565217391
FPR: 0.0
f1: 0
f2: 0
REAL->     Pos.        Neg.
```

```
     |--------------------------|
Pos. |    0         0           |
Neg. |    20         95          |
     |--------------------------|
Total Accuracy with Gamma(2)= 82.6086956521739

Micro averages precision for gamma values: 0
Macro averages precision for gamma values: 0.0
True positive for Gamma(0.0625): 0
True negative for Gamma(0.0625): 94
False negative for Gamma(0.0625): 1
False negative for Gamma(0.0625): 20
Recall: 0.0
Precision: 0.0
FDR: 1.0
NPV: 0.8245614035087719
FPR: 0.010526315789473684
f1: 0
f2: 0
REAL->     Pos.        Neg.
     |--------------------------|
Pos. |    0         1           |
Neg. |    20         94          |
     |--------------------------|
Total Accuracy with best Gamma(0.0625)=
81.73913043478261
```

Q 3.3) We use the optimal Gamma as our hyper parameter. Because three classes to classify the SVC library uses one vs all approach in this part our output for this is

```
True positive for Gamma(0.0625): 4
True negative for Gamma(0.0625): 78
False negative for Gamma(0.0625): 12
False negative for Gamma(0.0625): 16
Recall: 0.2
Precision: 0.25
FDR: 0.75
NPV: 0.8297872340425532
FPR: 0.13333333333333333
f1: 1.1111111111111112
f2: 1.0416666666666667
REAL->     Pos.        Neg.
     |-------------------------|
Pos. |    4         12         |
Neg. |    16         78         |
     |-------------------------|
Micro averages precision for multiple label values: 0.25
Macro averages precision for multiple label values: 0.375
Total Accuracy with multiple label input: 46.08695652173913
```

Our accuracy in this case is lower than the binary classification because we have more class to classify. In one to all approach, we take one class as our 0(or 1) and others as 1(or 0). In our case all predictions come 0 so accuracy is less than others. Because of this problem all the values calculated which are FDR, NVP, FPR F1, F2 and precision recall values are not that true. Probably there is a problem with data which causes all the predictions to be zero.

Q 4)

Q 4.1) In question 4 part 1 we are expected to calculate FDR, NVP, FPR F1, F2, precision, recall and accuracy for full batch sized in logistic regression. The interpretation of the weights in logistic regression differs from the interpretation of the weights in linear regression, since the outcome in logistic regression is a probability between 0 and 1. The weights do not influence the probability linearly any longer. The weighted sum is transformed by the logistic function to a probability[1]. During the prediction we multiply PCA values of features so if any feature has higher value(with absolute) affecting more the prediction. So V4, V5, V9, V10, V11, V17, V22, V25, V26, V28 becomes most important 10 features according to dataset. When you check for dataset, you can see that they have biggest absolute values. For this part, some learning rate values. Here is the output

```
126
True Positive: 47
True Negative: 63
False Positive: 0
False Positive: 16
Precision: 1.0
Recall: 0.746031746031746
FDR: 0.0
NPV: 0.7974683544303798
FPR: 0.0
f1: 1.1454545454545455
f2: 1.0535117056856187
REAL->     Pos.        Neg.
     |-------------------------|
Pos. |    47        0          |
```

```
Neg. |    16        63           |
     |--------------------------|
Test Accuracy for learn rates of 0.00001 in 1000
iterations:
87.3015873015873
126
True Positive: 48
True Negative: 63
False Positive: 0
False Positive: 15
Precision: 1.0
Recall: 0.7619047619047619
FDR: 0.0
NPV: 0.8076923076923077
FPR: 0.0
f1: 1.135135135135135
f2: 1.05
REAL->     Pos.        Neg.
     |--------------------------|
Pos. |    48        0           |
Neg. |    15        63           |
     |--------------------------|
Test Accuracy for learn rates of 0.0001 in 1000 iterations:
88.09523809523809
126
True Positive: 49
True Negative: 62
False Positive: 1
False Positive: 14
Precision: 0.98
Recall: 0.7777777777777778
FDR: 0.02
NPV: 0.8157894736842105
FPR: 0.015873015873015872
f1: 1.1150442477876106
f2: 1.0430463576158941
REAL->     Pos.        Neg.
     |--------------------------|
Pos. |    49        1           |
Neg. |    14        62           |
```

```
     |--------------------------|
Test Accuracy for learn rates of 0.001 in 1000 iterations:
88.09523809523809
126
True Positive: 52
True Negative: 58
False Positive: 5
False Positive: 11
Precision: 0.9122807017543859
Recall: 0.8253968253968254
FDR: 0.08771929824561403
NPV: 0.8405797101449275
FPR: 0.07936507936507936
f1: 1.05
f2: 1.0194174757281553
REAL->     Pos.        Neg.
     |--------------------------|
Pos. |    52        5           |
Neg. |    11        58           |
     |--------------------------|
Test Accuracy for learn rates of 0.01 in 1000 iterations:
87.3015873015873
126
True Positive: 51
True Negative: 51
False Positive: 12
False Positive: 12
Precision: 0.8095238095238095
Recall: 0.8095238095238095
FDR: 0.19047619047619047
NPV: 0.8095238095238095
FPR: 0.19047619047619047
f1: 1.0
f2: 1.0
REAL->     Pos.        Neg.
     |--------------------------|
Pos. |    51        12           |
Neg. |    12        51           |
     |--------------------------|
Test Accuracy for learn rates of 0.1 in 1000 iterations:
```

```
80.95238095238095
Best learn rate is: 0.001
```

In here we see some increase and decrease in accuracy values. The reason for that is step size. If we choose big step size in gradient descent, we can pass over the total minimum value. Otherwise, we can never reach the maximum size so best learning rate becomes 0.001 which is somewhere in the middle of values tried.

As well as this one I tried with different iteration counts which are from 100 to 1000. Here is the output

```
126
True Positive: 49
True Negative: 62
False Positive: 1
False Positive: 14
Precision: 0.98
Recall: 0.7777777777777778
FDR: 0.02
NPV: 0.8157894736842105
FPR: 0.015873015873015872
f1: 1.1150442477876106
f2: 1.0430463576158941
REAL->     Pos.        Neg.
     |-------------------------|
Pos. |    49        1          |
Neg. |    14        62          |
     |-------------------------|
Test Accuracy for learn rates of 0.1 in 100 iterations:
88.09523809523809
126
True Positive: 49
True Negative: 62
False Positive: 1
False Positive: 14
```

```
Precision: 0.98
Recall: 0.7777777777777778
FDR: 0.02
NPV: 0.8157894736842105
FPR: 0.015873015873015872
f1: 1.1150442477876106
f2: 1.0430463576158941
REAL->     Pos.        Neg.
     |--------------------------|
Pos. |    49        1           |
Neg. |    14        62           |
     |--------------------------|
Test Accuracy for learn rates of 0.1 in 200 iterations:
88.09523809523809
126
True Positive: 50
True Negative: 62
False Positive: 1
False Positive: 13
Precision: 0.9803921568627451
Recall: 0.7936507936507936
FDR: 0.0196078431372549
NPV: 0.8266666666666667
FPR: 0.015873015873015872
f1: 1.105263157894737
f2: 1.0396039603960394
REAL->     Pos.        Neg.
     |--------------------------|
Pos. |    50        1           |
Neg. |    13        62           |
     |--------------------------|
Test Accuracy for learn rates of 0.1 in 300 iterations:
88.88888888888889
126
True Positive: 50
True Negative: 62
False Positive: 1
False Positive: 13
Precision: 0.9803921568627451
Recall: 0.7936507936507936
```

```
FDR: 0.0196078431372549
NPV: 0.8266666666666667
FPR: 0.015873015873015872
f1: 1.105263157894737
f2: 1.0396039603960394
REAL->     Pos.        Neg.
     |--------------------------|
Pos. |    50        1           |
Neg. |    13        62           |
     |--------------------------|
Test Accuracy for learn rates of 0.1 in 400 iterations:
88.88888888888889
126
True Positive: 51
True Negative: 61
False Positive: 2
False Positive: 12
Precision: 0.9622641509433962
Recall: 0.8095238095238095
FDR: 0.03773584905660377
NPV: 0.8356164383561644
FPR: 0.031746031746031744
f1: 1.0862068965517242
f2: 1.0327868852459017
REAL->     Pos.        Neg.
     |--------------------------|
Pos. |    51        2           |
Neg. |    12        61           |
     |--------------------------|
Test Accuracy for learn rates of 0.1 in 500 iterations:
88.88888888888889
126
True Positive: 51
True Negative: 59
False Positive: 4
False Positive: 12
Precision: 0.9272727272727272
Recall: 0.8095238095238095
FDR: 0.07272727272727272
NPV: 0.8309859154929577
```

```
FPR: 0.06349206349206349
f1: 1.0677966101694916
f2: 1.0260586319218243
REAL->     Pos.        Neg.
     |--------------------------|
Pos. |    51        4           |
Neg. |    12        59           |
     |--------------------------|
Test Accuracy for learn rates of 0.1 in 600 iterations:
87.3015873015873
126
True Positive: 51
True Negative: 59
False Positive: 4
False Positive: 12
Precision: 0.9272727272727272
Recall: 0.8095238095238095
FDR: 0.07272727272727272
NPV: 0.8309859154929577
FPR: 0.06349206349206349
f1: 1.0677966101694916
f2: 1.0260586319218243
REAL->     Pos.        Neg.
     |--------------------------|
Pos. |    51        4           |
Neg. |    12        59           |
     |--------------------------|
Test Accuracy for learn rates of 0.1 in 700 iterations:
87.3015873015873
126
True Positive: 51
True Negative: 58
False Positive: 5
False Positive: 12
Precision: 0.9107142857142857
Recall: 0.8095238095238095
FDR: 0.08928571428571429
NPV: 0.8285714285714286
FPR: 0.07936507936507936
f1: 1.0588235294117647
```

```
f2: 1.0227272727272727
REAL->     Pos.        Neg.
     |--------------------------|
Pos. |    51        5           |
Neg. |    12        58           |
     |--------------------------|
Test Accuracy for learn rates of 0.1 in 800 iterations:
86.5079365079365
126
True Positive: 52
True Negative: 58
False Positive: 5
False Positive: 11
Precision: 0.9122807017543859
Recall: 0.8253968253968254
FDR: 0.08771929824561403
NPV: 0.8405797101449275
FPR: 0.07936507936507936
f1: 1.05
f2: 1.0194174757281553
REAL->     Pos.        Neg.
     |--------------------------|
Pos. |    52        5           |
Neg. |    11        58           |
     |--------------------------|
Test Accuracy for learn rates of 0.1 in 900 iterations:
87.3015873015873
126
True Positive: 52
True Negative: 58
False Positive: 5
False Positive: 11
Precision: 0.9122807017543859
Recall: 0.8253968253968254
FDR: 0.08771929824561403
NPV: 0.8405797101449275
FPR: 0.07936507936507936
f1: 1.05
f2: 1.0194174757281553
REAL->     Pos.        Neg.
```

```
     |--------------------------|
Pos. |    52        5           |
Neg. |    11        58           |
     |--------------------------|
Test Accuracy for learn rates of 0.1 in 1000 iterations:
87.3015873015873
```

We see some increase and decrease in the accuracy rates. But in total there is an increase when iteration count increases because with more iteration we have bigger chance to create better weights to serve prediction. We try more to reach best weights. Those trials have been made with full batch size which is 800 for this data. That means the weight value is updated once in 800 times. In the data given, I normalized the data to get higher accuracy. The position of the data cumulated in the plot was not near to the origin so we should normalize the data for better prediction.

Q 4.2) For this part I tried the batch size as 32 which shows better chance to update the weights. We add learn rate times gradient value at each time to weights and less batch size give us the chance of updating more. The situation is even better for stochastic logistic regression whose batch size is one. The output for mini batch(32)

```
126
True Positive: 49
True Negative: 62
False Positive: 1
False Positive: 14
Recall: 0.7777777777777778
FDR: 0.02
NPV: 0.8157894736842105
```

```
FPR: 0.015873015873015872
f1: 0
f2: 0
REAL->     Pos.        Neg.
     |-------------------------|
Pos. |    49        1          |
Neg. |    13        63          |
     |-------------------------|
Test Accuracy for learn rates of 0.1 in 1000 iterations:
89.09523809523809
```

The output for stochastic one

```
126
True Positive: 49
True Negative: 62
False Positive: 1
False Positive: 14
Precision: 0.98
Recall: 0.7777777777777778
FDR: 0.02
NPV: 0.8157894736842105
FPR: 0.015873015873015872
f1: 1.1150442477876106
f2: 1.0430463576158941
REAL->     Pos.        Neg.
     |-------------------------|
Pos. |    49        0          |
Neg. |    13        64          |
     |-------------------------|
Test Accuracy for learn rates of 0.1 in 1000 iterations:
90.09523809523809
```

Q 4.3) NPV, FPR, FDR, F1 and F2 score might be a better measure to use if we need to seek a balance between precision and recall and there is an uneven class distribution (large number of Actual Negatives). Let us think a

prediction case having huge number of true negative values. It would have huge accuracy but for example positive here represents fraud case. So in this model it is better idea to user NPV, FPR, FDR, F1 and F2 score to get better predictions instead of precision or recall.