CS464 Introduction to Machine Learning

Fall 2021

HOMEWORK #2

REPORT

Section 1

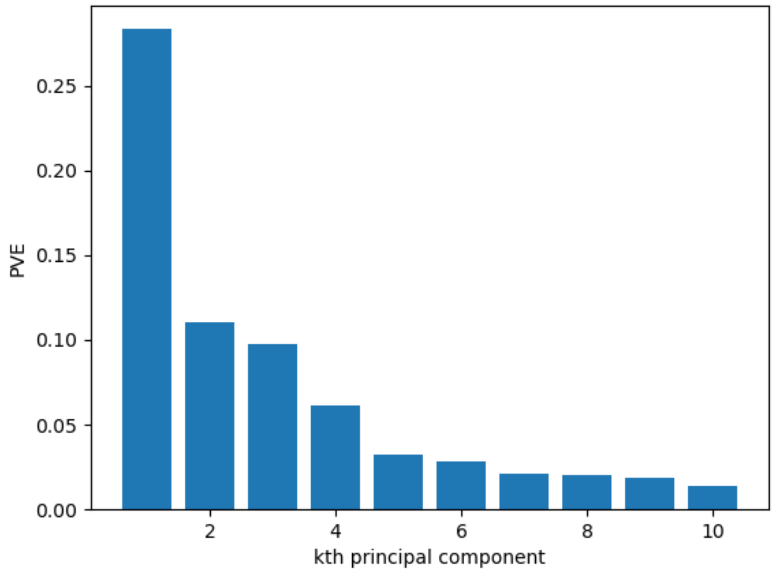
Lara Fenercioğlu

21802536

Question 1

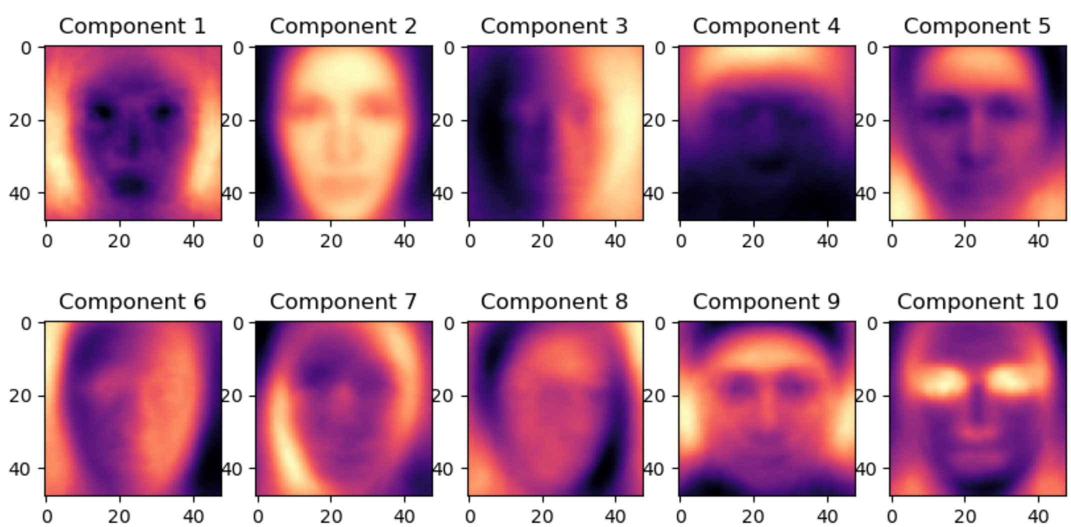
1.

The below table shows the first 10th principal components of the PVE.

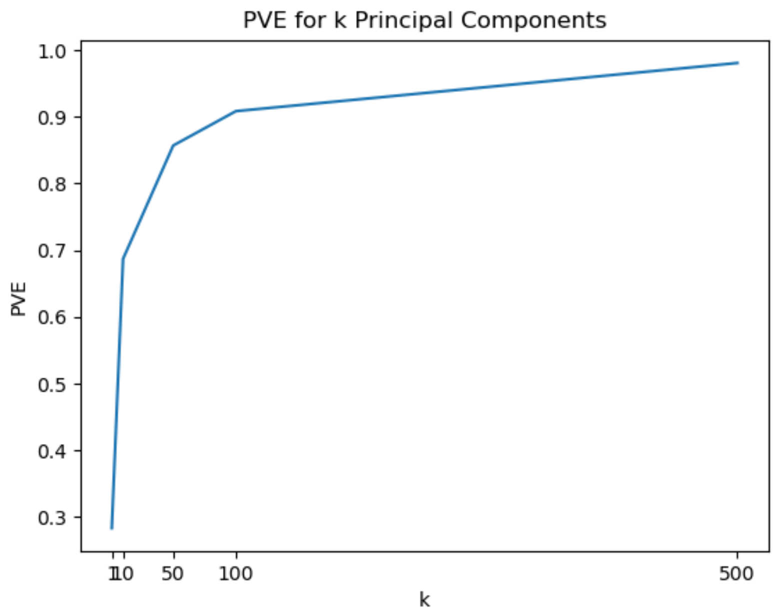


PVE for 10 components: [0.28334475 0.11027901 0.09766803 0.06101507 0.03217829 0.02860725 0.02095556 0.02052136 0.0184183 0.01409122]

Here are the representations of the first 10 principal components as images.



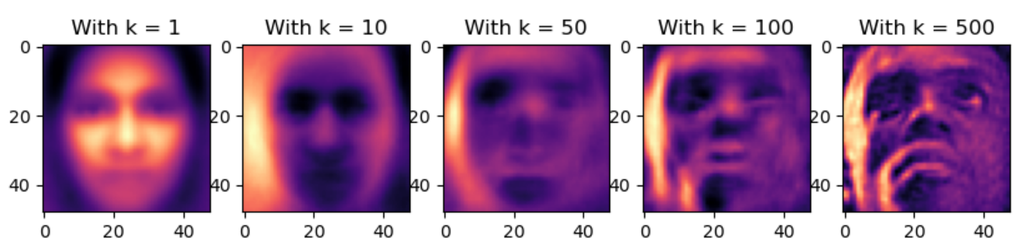
2.



There is a drastic increase in between k=1 and k =10. This means that if we increase the number of principal components from 1 to 10, then the PVE gets so much bigger but afterwards even though we increase k, the PVE stays the same, it converges.

3.

Reconstruction formula is the following $X \bullet PC \bullet PC^T + \mu$
In order to reconstruct an image we first perform a dot product with centered feature matrix and principal components. This calculation's result is used again as a dot product operand with the transpose of the principal components. Then we add the mean vector to the result. Finally, we reshape the image to 48x48.



By analyzing the above images with different principal components, we can conclude that the more principal components we have, the more precise image we will get from the reconstruction operation. For instance, the first image with k is 1 has a lower complexity compared to when k is 500.

Question 2

1.

X denotes features where Y denotes the labels and ß is the weights.

$$J_n = ||y - X\beta||^2 \rightarrow \frac{d(||y-X\beta||^2)}{d\beta} = \frac{d\,(y-X\beta)^T(y-X\beta)}{d\beta} = -2X^TY + 2X^TX\beta = 0$$

$$\beta = (X^TX)^{-1}X^TY$$

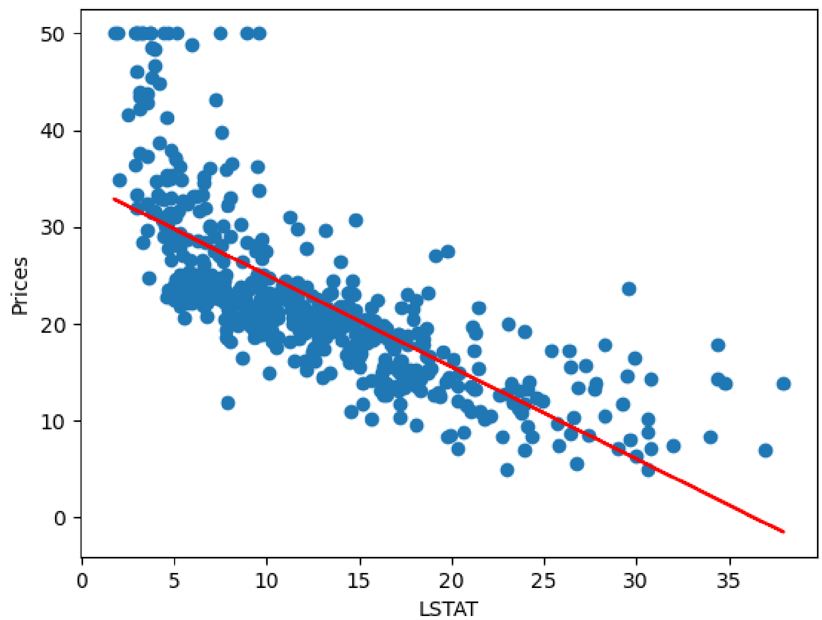
Then we can calculate the weights by the above equation.

2.

X^TX is calculated as 13 by using the numpy.linalg.matrix_rank() function. So we can say that the feature set X has 13 independent features. X^TX is not invertible because rank of the X equals the rank of X^TX which is 13.

3.

Coefficients of the linear regression model is: [[34.55384088] [-0.95004935]]
Mean Squared Error is calculated as: 38.482967229894165
The lower the MSE, the better the model forecasts. So, since our MSE value is quite high, the model isn't working so well. If we had considered more features, the MSE value might have been lower.



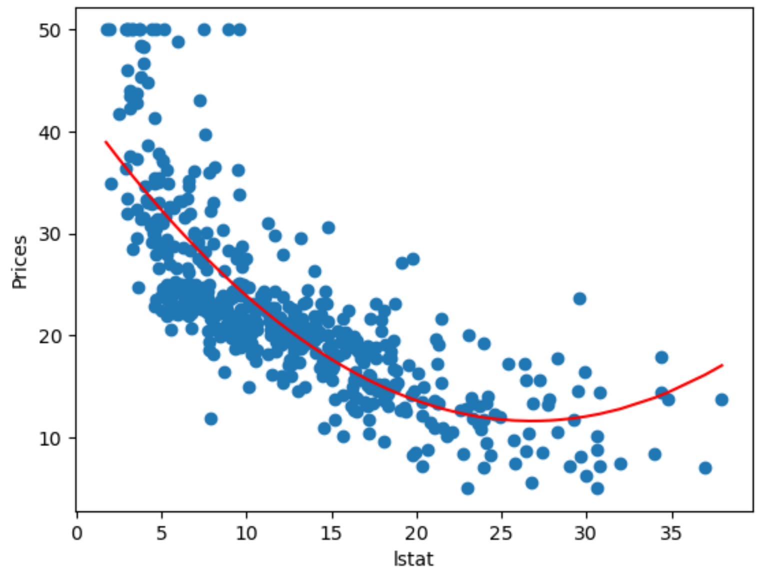
The red line indicates the line that fits the best and blue dots are the actual labels. The LSTAT values that are in between 10 to 25 are predicted almost correctly with some exceptions. However there are many outliers as well since we try to make predictions based on only one feature.

4.

Coefficients of the polynomial regression model is: [[42.86200733] [-2.3328211] [0.04354689]]

Mean Squared Error is calculated as: 30.330520075853748

Comparing this MSE to the one we get from the Linear Regression model, this model performs better since the MSE value has decreased.



The red polynomial line indicates the line that fits the best and blue dots are the actual labels. Compared to the linear regression model, polynomial regression is better in predicting especially between 0-10 and 25-35 intervals where linear regression results worse. It more fits the data however there are still outliers since the model is based on only one feature.

Question 3

1.

-----For learning rate 1e-05 -----
Full Batch Gradient Ascent accuracy= 69.27374301675978
Precision= 0.4492753623188406
Recall= 0.6458333333333334
NPV= 0.8454545454545455
FPR= 0.2900763358778626
FDR= 0.5507246376811594
F1= 0.5299145299145299
F2= 0.5938697318007664
Confusion Matrix:
[[31 17]
 [38 93]]
-----For learning rate 0.0001 -----
Full Batch Gradient Ascent accuracy= 70.39106145251397
Precision= 0.463768115942029
Recall= 0.6666666666666666

NPV= 0.8545454545454545
FPR= 0.2824427480916031
FDR= 0.5362318840579711
F1= 0.5470085470085471
F2= 0.6130268199233716
Confusion Matrix:
[[32 16]
 [37 94]]
-----For learning rate 0.001 -----
Full Batch Gradient Ascent accuracy= 70.39106145251397
Precision= 0.463768115942029
Recall= 0.6666666666666666
NPV= 0.8545454545454545
FPR= 0.2824427480916031
FDR= 0.5362318840579711
F1= 0.5470085470085471
F2= 0.6130268199233716
Confusion Matrix:
[[32 16]
 [37 94]]
-----For learning rate 0.01 -----
Full Batch Gradient Ascent accuracy= 67.59776536312849
Precision= 0.42028985507246375
Recall= 0.6170212765957447
NPV= 0.8363636363636363
FPR= 0.30303030303030304
FDR= 0.5797101449275363
F1= 0.5
F2= 0.5642023346303502
Confusion Matrix:
[[29 18]
 [40 92]]
-----For learning rate 0.1 -----
Full Batch Gradient Ascent accuracy= 69.83240223463687
Precision= 0.42028985507246375
Recall= 0.6744186046511628
NPV= 0.8727272727272727
FPR= 0.29411764705882354
FDR= 0.5797101449275363
F1= 0.5178571428571429

F2= 0.6016597510373445
Confusion Matrix:
[[29 14]
 [40 96]]

I have chosen 0.0001 as the learning rate since it produced the best accuracy and used it as the learning rate for the further calculations.

2.

Mini Batch Gradient Ascent accuracy= 70.39106145251397
Precision= 0.36231884057971014
Recall= 0.7352941176470589
NPV= 0.9181818181818182
FPR= 0.30344827586206896
FDR= 0.6376811594202898
F1= 0.4854368932038835
F2= 0.6097560975609756
Confusion Matrix:
[[25   9]
 [44 101]]
Stochastic Gradient Ascent accuracy= 70.39106145251397
Precision= 0.37681159420289856
Recall= 0.7222222222222222
NPV= 0.9090909090909091
FPR= 0.3006993006993007
FDR= 0.6231884057971014
F1= 0.49523809523809526
F2= 0.6103286384976525
Confusion Matrix:
[[26 10]
 [43 100]]

3.

F scores carry more accurate and informative information when the classes are imbalanced and also when we need a balance between recall and precision.

FDR, FPR, NPV carry more information than accuracy when false positives are detrimental.