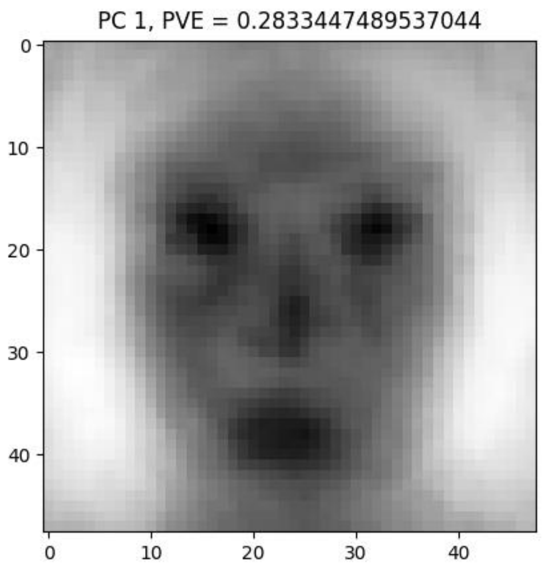
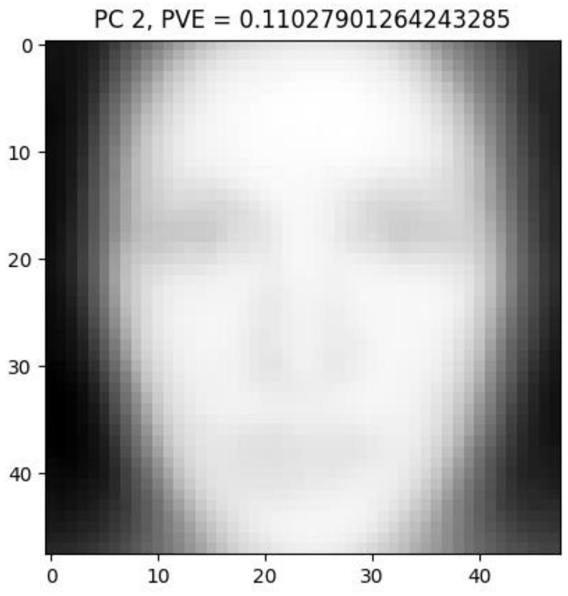
Batu Arda Düzgün

Question 1.1:

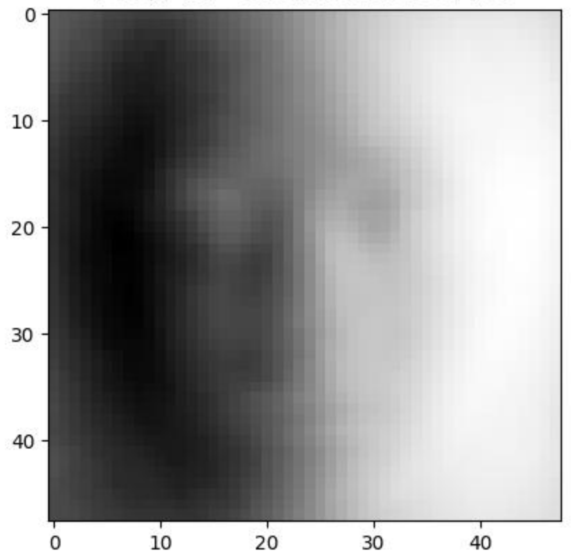
```
PVE for PC 1 is 0.2833447489537044
PVE for PC 2 is 0.11027901264243285
PVE for PC 3 is 0.09766803183987753
PVE for PC 4 is 0.06101507486957514
PVE for PC 5 is 0.032178286612646885
PVE for PC 6 is 0.028607248398294864
PVE for PC 7 is 0.020955561849916766
PVE for PC 8 is 0.02052135681601378
PVE for PC 9 is 0.018418297879458326
PVE for PC 10 is 0.014091219567233481
```



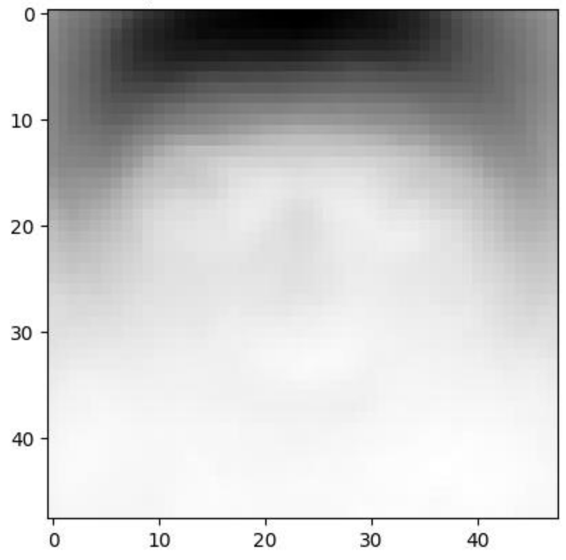


Batu Arda Düzgün

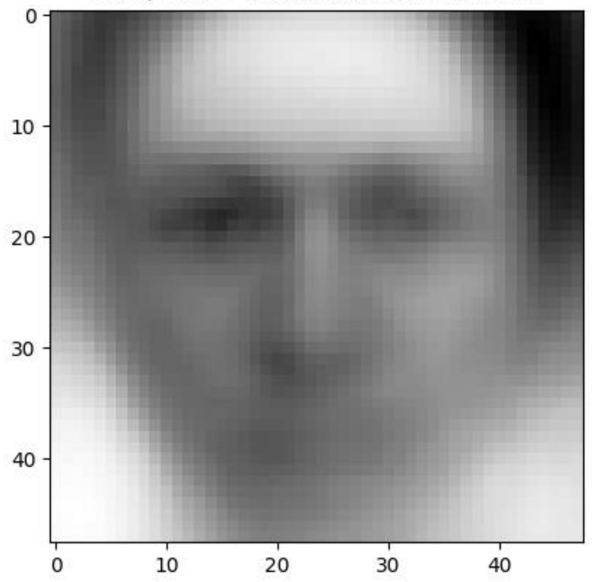
PC 3, PVE = 0.09766803183987753



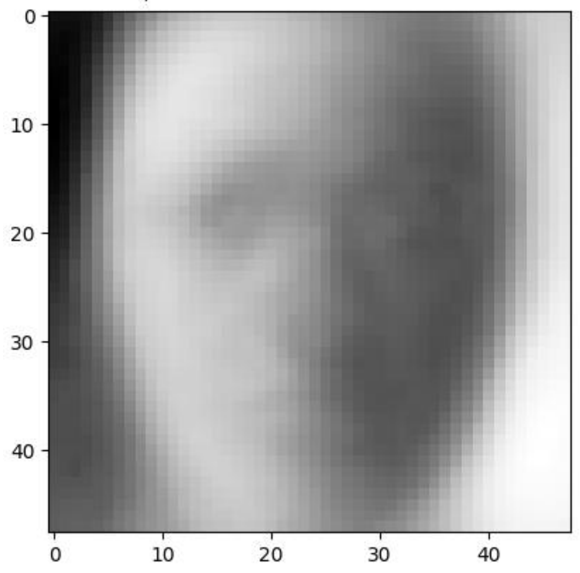
PC 4, PVE = 0.06101507486957514

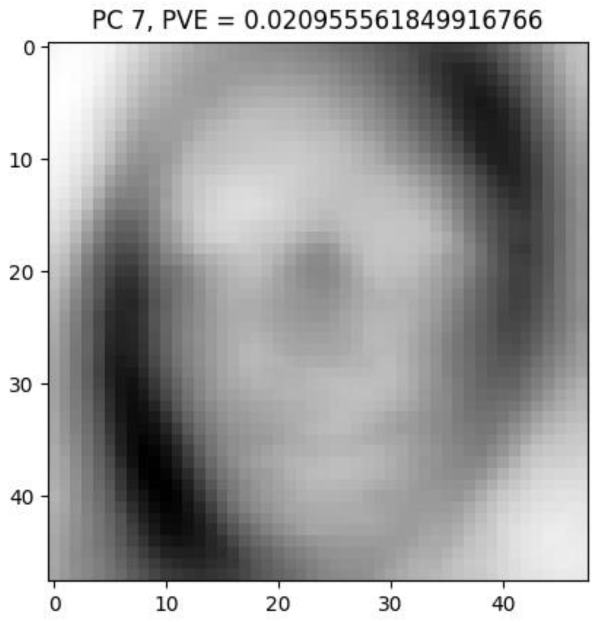


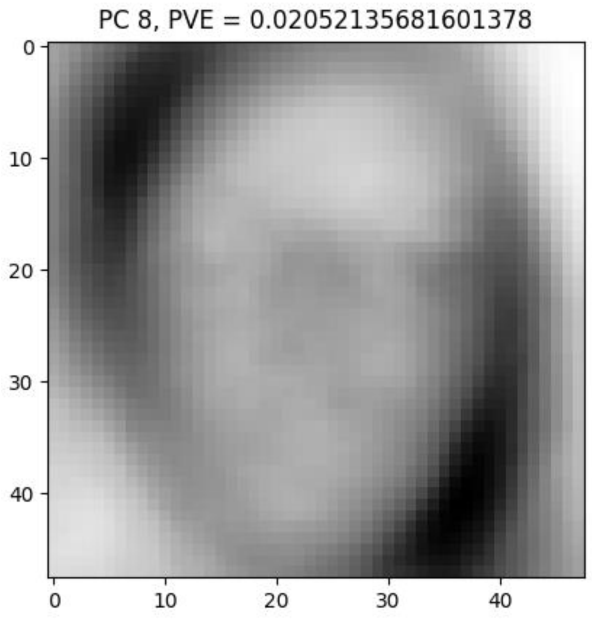
PC 5, PVE = 0.032178286612646885

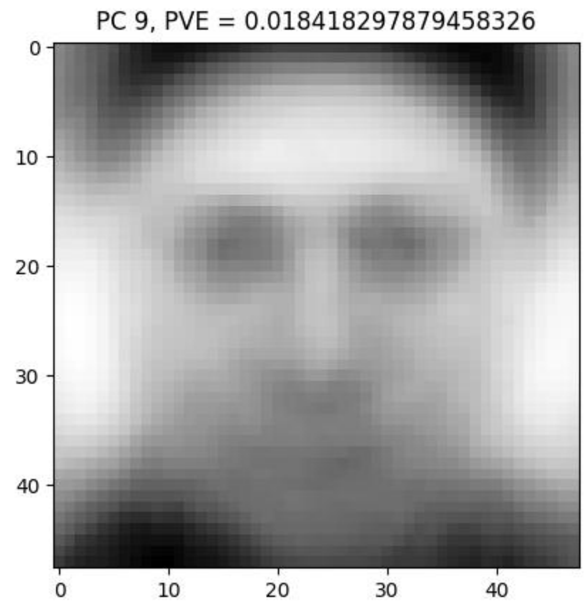


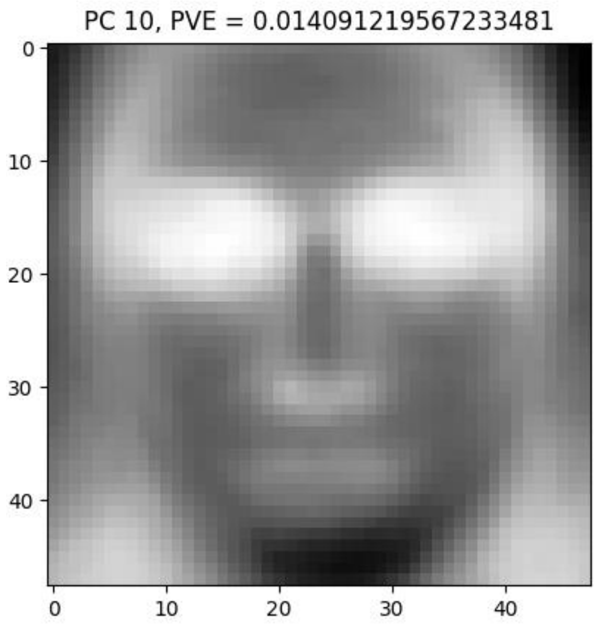
PC 6, PVE = 0.028607248398294864







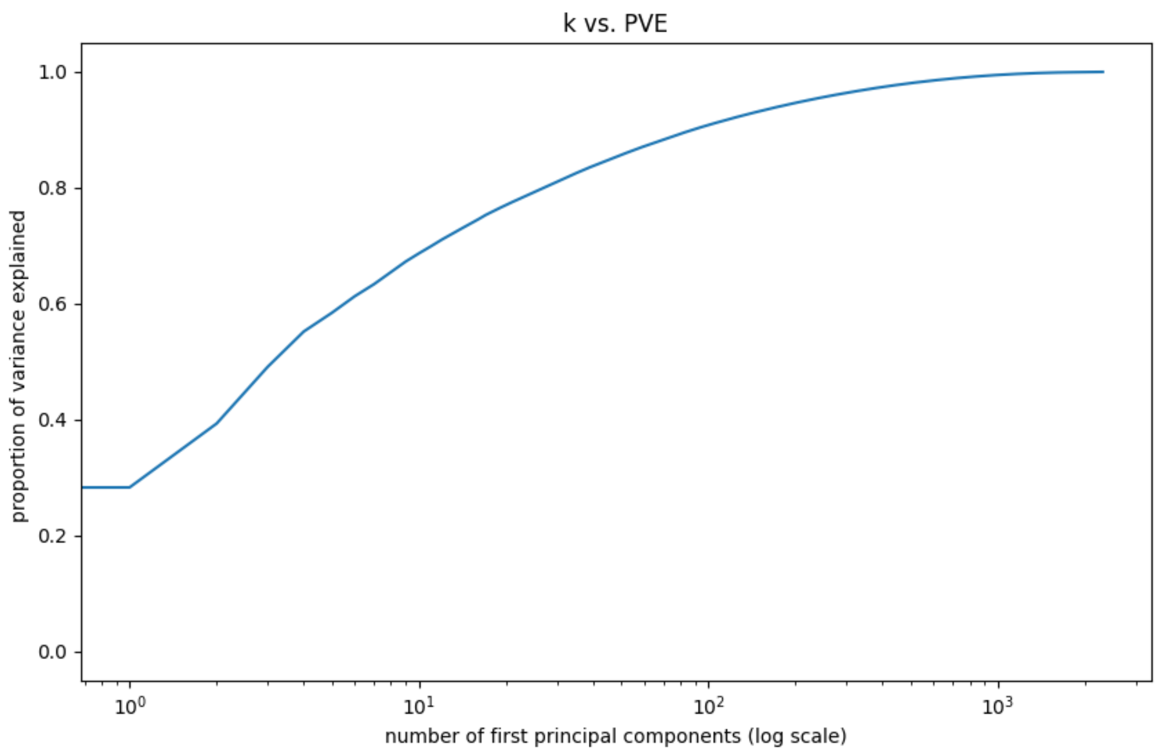




They do look like faces. However they have negative components in them so it is not distributed like a nouse, eyes or a mount. Still by looking at them it is easy to say that these are PCs of human faces. If we wanted them to be parts of a face with out any negative components we could have Non-negative matrix factorization.

Question 1.2:

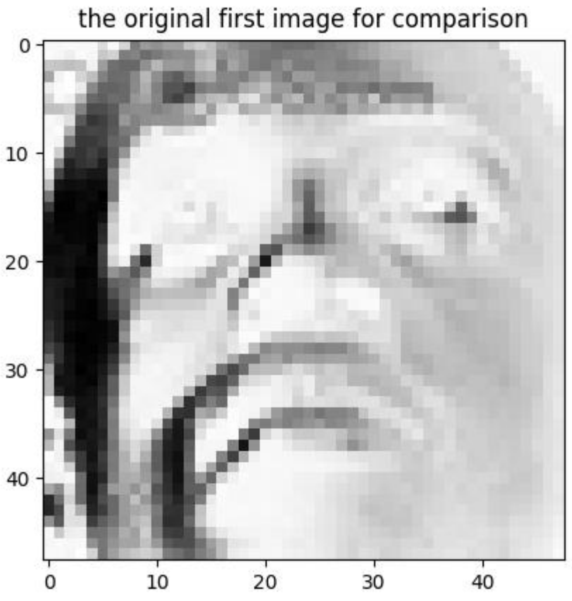
```
PVE for the first 1 principal components: 0.2833447489537044
PVE for the first 10 principal components: 0.6870788394291539
PVE for the first 50 principal components: 0.8569321884808174
PVE for the first 100 principal components: 0.9084447232542031
PVE for the first 500 principal components: 0.9806486239270216
```

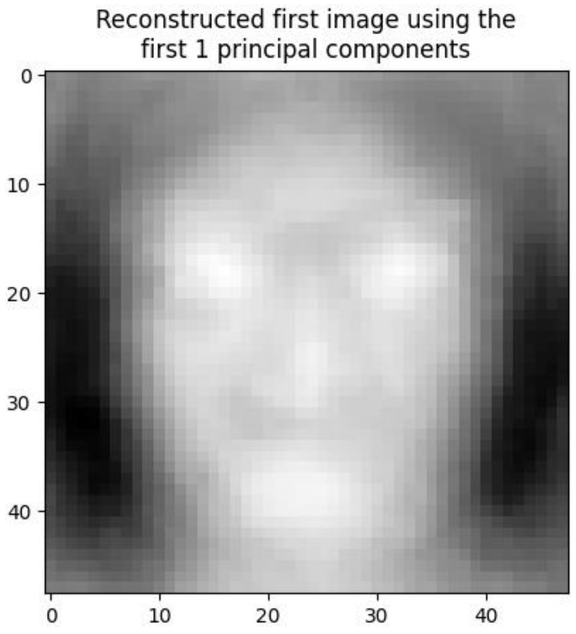


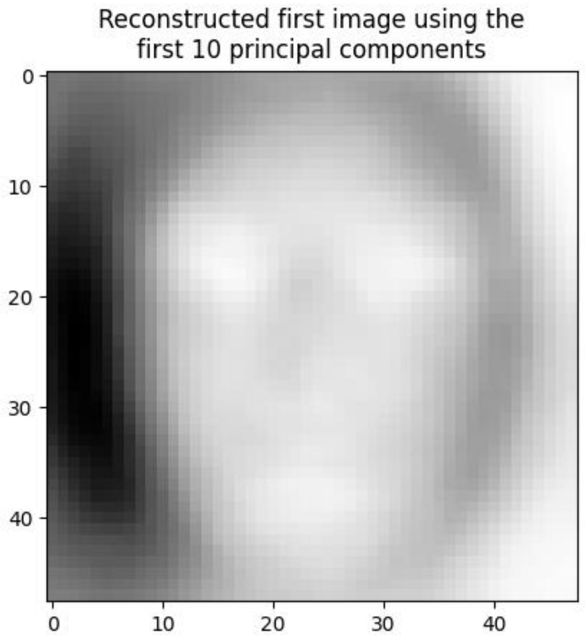
As the number of PCs used increase the effectiveness of using more PCs drastically decreases. So most of the information is in the first PCs and the information contained in PC after that point decrease drastically. The last 1500 PCs almost have no information in them and we will see this in the next question.

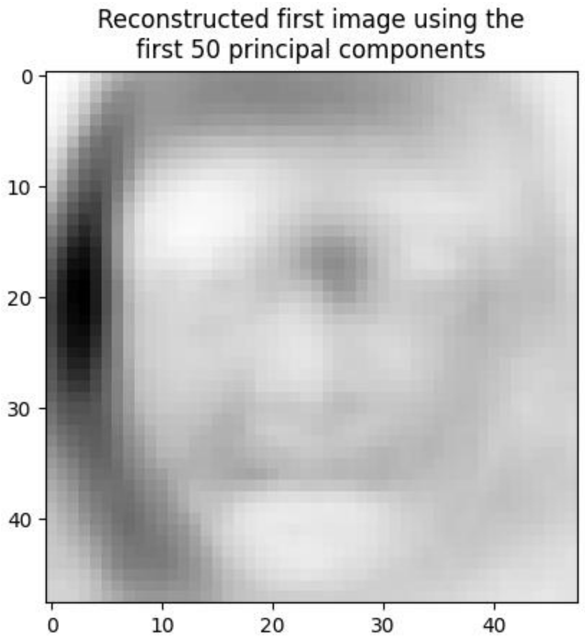
Question 1.3:

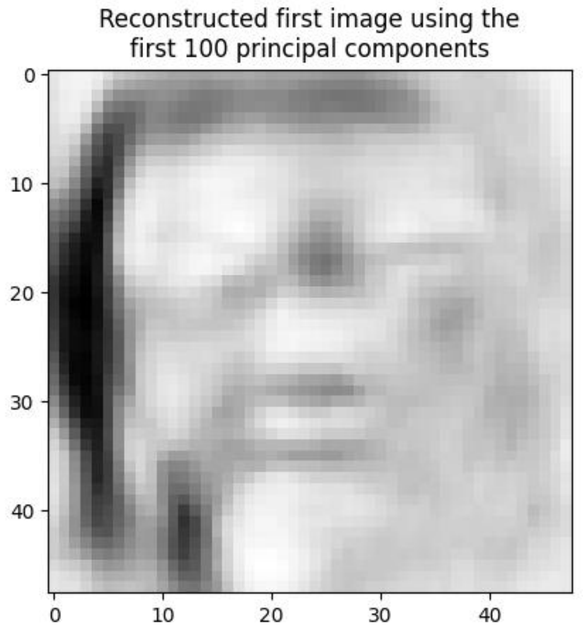
We can decrease the dimensions of the images by taking the images dot product with the PCs and then saving the results. Then we can take the product of these results with their respective PCs and add up the results to re construct the original images. We could also add the mean value we subtracted at the beginning but the “imshow” function already min-max normalizes the images so I omitted it.

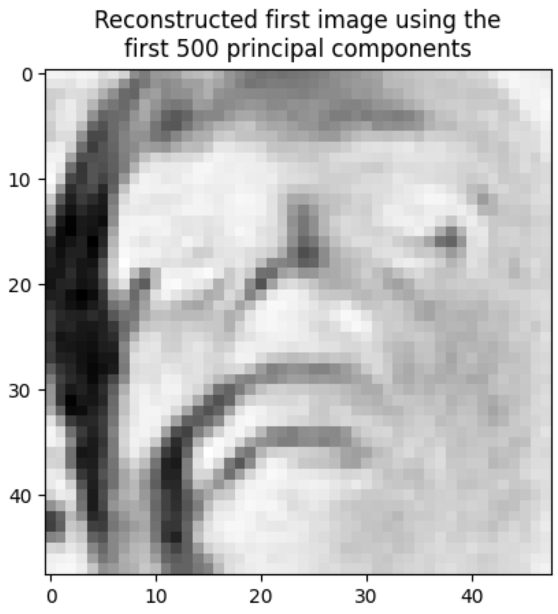

the original first image for comparison
0
10
20
30
40


Reconstructed first image using the first 1 principal components
0
10
20
30
40


Reconstructed first image using the first 10 principal components
0
10
20
30
40


Reconstructed first image using the first 50 principal components
0
10
20
30
40


Reconstructed first image using the first 100 principal components
0
10
20
30
40


Reconstructed first image using the first 500 principal components
0
10
20
30
40

The reconstruction using only the first PC is simply the first PC which is not meaning full other then telling us that it is a face. The reconstruction from the first 10 PCs again can be called a face yet it only slightly resembles the original. This is interesting as the first 10 PCs contain 68% of the information. The reconstruction from the first 50 PCs is only slightly better than the 10 PC one even though it has 85% of the information. This reminds us that the small details on faces is very important and most of the first PCs are just the general shape of the face and potential shadows on the face. The reconstruction from the first 100 PCs resembles the original. The reconstruction from the first 500 PCs looks almost perfect. We can say by using the first 500 PCs we can decrease the size of these images to 500 features without loss of significant information.

Question 2.1:

$$\beta: the\ weight\ vector$$

$$X: are\ the\ values\ of\ the\ features$$

$$y: the\ labels\ vector$$

We want to minimize the error J_n:

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

$$J_n = y^T y - 2\beta^T X^T y + \beta^T X^T X\beta$$

Where the derivative is 0 will give us the result.

$$\frac{\delta J_n}{\delta \beta} = -2X^T y + 2X^T X\beta = 0$$

$$2X^T y = 2X^T X\beta$$

$$\beta = (X^T X)^{-1} X^T y$$

Question 2.2:

```
***** Question 2.2: *****

X^T X is rank 13
```

$X^T X$ is rank 13, and there are 13 features so it is full rank which means it is invertible.

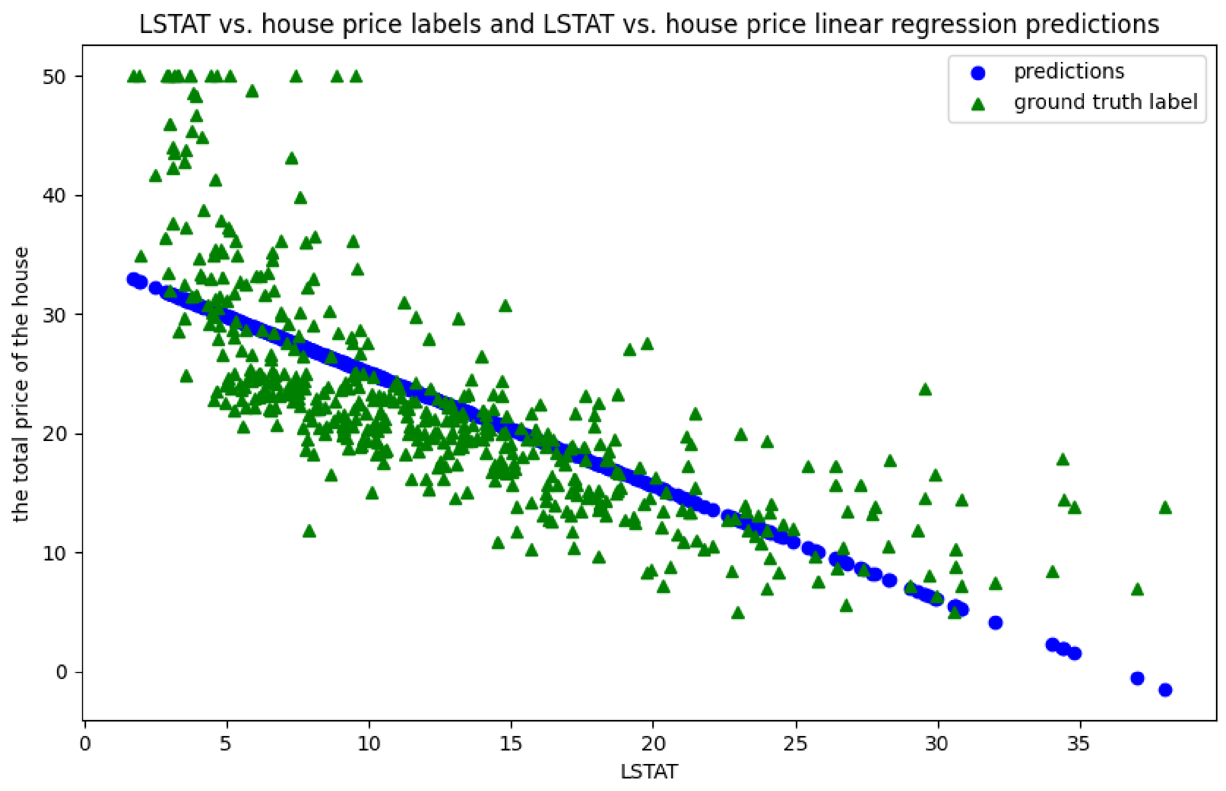
Question 2.3:

```
***** Question 2.3: *****

w0: [34.55384088]
w1: [-0.95004935]

the MSE for linear regression with only LSTAT: 38.48296722989415
```

w0 is the coefficient of the constant. We can see the average price of the houses is 34.55, w1 is the coefficient of the LTSAS and it is negative so we can say the house price is negatively proportional compared to it.



Question 2.4:

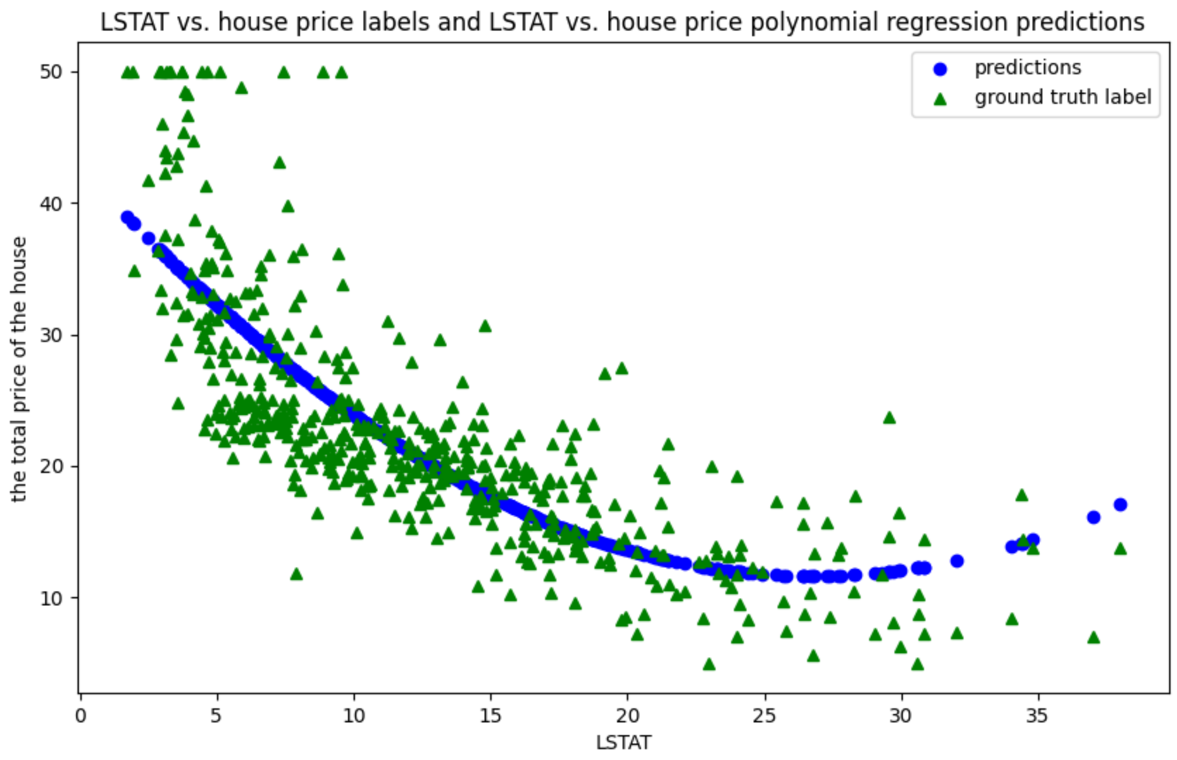
```
***** Question 2.4: *****

w0:  [42.86200733]
w1:  [-2.3328211]
w2:  [0.04354689]

the MSE for polynomial regression with only LSTAT: 30.33052007585372
```

w0 is the coefficient of the constant. We can see the average price of the houses is 34.55, w1 is the coefficient of the LSTAT and it is negative so we can say the house price is negatively proportional compared to it. w2 is the coaficent of square of LSTAT and it is positive which means the negative effect of having high LSTAT is decreasing as LSTAT increases. Another way to say this is; having low LSTAT is more valuable for lower LSTAT values.

MSE of the polynomial regression is lower then linear regresion because the data is non linearly distributed and we can capture the more of the non linear behavior with this.



Question 3.1:

```
***** Question 3.1: *****

MSE for learn rate = 1e-05 : 0.3854748603351955
MSE for learn rate = 0.0001 : 0.3407821229050279
MSE for learn rate = 0.001 : 0.30726256983240224
MSE for learn rate = 0.01 : 0.3240223463687151
MSE for learn rate = 0.1 : 0.45251396648044695
```

We can see learn rate 0.001 works the best so we will be using that for the rest of the question.

```
precision for learn rate 0.001: 0.45652173913043476
recall for learn rate 0.001: 0.9130434782608695
negative predictive value for learn rate 0.001: 0.8536585365853658
false positive rate for learn rate 0.001: 0.6818181818181818
false discovery rate for learn rate 0.001: 0.5434782608695652

F1 score for learn rate 0.001: 0.608695652173913
F2 score for learn rate 0.001: 0.7608695652173912
```

Question 3.2:

```
accuracy for learn rate 0.001: 0.6703910614525139
true positive count for learn rate 0.001: 29
true negative count for learn rate 0.001: 91
false positive count for learn rate 0.001: 19
false negative count for learn rate 0.001: 40

micro precision for learn rate 0.001: 0.6703910614525139
macro precision for learn rate 0.001: 0.6494115776081425
micro recall for learn rate 0.001: 0.659217877094972
macro recall for learn rate 0.001: 0.6092885375494071

micro negative predictive value for learn rate 0.001: 0.659217877094972
macro negative predictive value for learn rate 0.001: 0.63558352402746

micro false positive rate for learn rate 0.001: 0.3407821229050279
macro false positive rate for learn rate 0.001: 0.3907114624505929

micro false discovery rate for learn rate 0.001: 0.3407821229050279
macro false discovery rate for learn rate 0.001: 0.36441647597254


micro F1 score for learn rate 0.001: 0.659217877094972
macro F1 score for learn rate 0.001: 0.6221583209999252

micro F2 score for learn rate 0.001: 0.659217877094972
macro F2 score for learn rate 0.001: 0.6143720249203214
```

There is some drop in accuracy of the algorithm but it now runs faster due to not using all the points to train at each round.

Question 3.3:

If the original data has a very low number of negative labels then the algorithm could predict true for everything a still achieve a very high accuracy, precision and recall. This is not a good prediction yet because focus on the wrong aspects of the model and make it seem good. To prevent this from happening performance metrics such as NPV, FPR, FDR, F1 and F2 are useful.