EIGENFACES FOR FACIAL RECOGNITION

NAVTEJPREET SINGH
2018CSB1107

INDIAN INSTITUTE OF TECHNOLOGY, ROPAR
PUNJAB - 140001

1. PRINCIPAL COMPONENT ANALYSIS AND EIGENFACES FOR FACIAL RECOGNITION

Dataset used for this task was “Labelled Faces in the Wild” from sci-kit learn datasets. This dataset had multiple pictures of the celebrities which were labelled. The dataset had about 12800 images each of which was a 62x47 pixels in size.

- Thus number number of predictor variables for each image (or datapoint) were 62 x 47 = 2914 and one target variable having name of the celebrity.
- Since, this is dataset is huge in terms of input dimensions training a model would require enormous amount of computing power and time.
- We selected the dataset for celebrities with minimum images 100 and our input dataset looked like following. It had a total of 1140 datapoints (1140 images) belonging to 5 different classes (5 people).

Class index	no. of faces	Class Name
0	236	Colin Powell
1	121	Donald Rumsfeld
2	530	George W Bush
3	109	Gerhard Schroeder
4	144	Tony Blair

	0	1	2	3	4	5	6	7	8	9	...	2906	2907	2908	2909	2910	2911	2912	2913	target	target_names
0	82.666664	87.666664	65.333336	53.000000	97.000000	128.666672	137.666672	145.666672	156.333328	164.333328	...	74.333336	91.000000	107.666664	109.666664	113.666664	106.000000	140.000000	199.333328	2	George W Bush
1	52.333332	49.333332	69.333336	92.333336	123.666664	152.666672	155.333328	158.333328	161.666672	165.000000	...	155.000000	207.000000	220.333328	225.666672	233.333328	235.666672	228.666672	222.000000	3	Gerhard Schroeder
2	37.333332	39.000000	41.666668	52.666668	82.666664	118.000000	127.666664	129.333328	138.333328	141.333328	...	137.666672	117.333336	103.666664	104.000000	106.333336	97.000000	71.666664	59.333332	1	Donald Rumsfeld
3	33.333332	30.666666	31.000000	48.333332	74.666664	92.000000	96.000000	98.000000	102.333336	107.333336	...	123.666664	120.666664	128.333328	146.000000	149.000000	140.666672	129.333328	132.333328	4	Tony Blair
4	169.666672	149.333328	130.333328	145.666672	175.333328	195.333328	203.666672	205.333328	205.666672	210.333328	...	76.666664	88.000000	95.333336	98.000000	101.333336	103.333336	107.000000	110.000000	1	Donald Rumsfeld
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1135	73.333336	73.000000	70.666664	82.333336	119.333336	149.333328	164.333328	174.333328	182.333328	187.333328	...	124.666664	134.333328	143.000000	158.000000	194.666672	230.000000	236.666672	231.333328	4	Tony Blair
1136	69.333336	64.333336	68.666664	92.333336	121.333336	141.666672	150.000000	154.666672	158.333328	159.000000	...	146.000000	136.666672	135.333328	141.000000	140.666672	132.000000	117.000000	115.333336	4	Tony Blair
1137	228.333328	224.333328	216.000000	200.000000	196.333328	186.333328	153.333328	111.666664	82.000000	97.000000	...	129.333328	133.000000	139.333328	143.666672	148.000000	149.666672	145.666672	164.666672	4	Tony Blair
1138	97.666664	118.333336	133.333328	145.000000	147.666672	148.000000	158.333328	165.333328	170.666672	177.333328	...	68.333336	101.666664	158.333328	194.666672	209.666672	219.000000	210.000000	142.000000	2	George W Bush
1139	39.333332	79.333336	109.333336	131.000000	98.000000	65.666664	71.666664	133.666672	195.333328	215.000000	...	7.000000	6.000000	5.000000	7.666667	14.333333	24.333334	34.333332	24.000000	4	Tony Blair

1140 rows × 2916 columns

NOTE: For all further references
Class 0 refers to **Colin Powell**
Class 1 refers to **Donald Rumsfeld**

Class 2 refers to **George W Bush**
Class 3 refers to **Gerhard Schroeder**
Class 4 refers to **Tony Blair**

Data was split in 70-30% ratio into train and test data using train_test_split from sklearn. A total of 798 images were set aside for training set and remaining 342 were allocated to testing set.
Dimensionality reduction was employed for reducing dimensions of input space from 2914 to 100 by applying PCA to train data with number of components equal to 100.
Now each of new 100 eigenvectors were used to represent the initial datapoint in terms of these new reduced number of dimensions. Here, these new eigenvectors were nothing but eigenfaces.

EIGENFACES:
When we apply PCA to the training data, it extracts orthonormal directions of maximum variance from the data.
Original data has n number of examples and m are number of dimensions.
Using PCA we get 100 eigenvectors $e_1, e_2, e_3, \dots, e_{100}$ with eigenvalues $\lambda_1, \lambda_2, \lambda_3, \dots, \lambda_{100}$ where $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_{100}$
Eigenvector e_1 is a $\boldsymbol{m}$ dimension vector in input space that represents the direction of maximum variance;
Eigenvector e_2 is a $\boldsymbol{m}$ dimension vector in input space that represents the direction of maximum variance in direction orthogonal to e_1;
And so on.
And as we know, now each datapoint can be represented in terms of these 100 new eigenvector directions instead of older 2914 pixel values.

Eigenfaces are nothing but eigenvectors when used in computer vision problem for facial recognition. These eigenvector or new vector space generally doesn't make a lot of sense which is a drawback of PCA. But in case of facial recognition, eigenfaces (eigenvectors) are "ghost faces" which represent not a particular face, rather capture the variance in the data so that each input image.

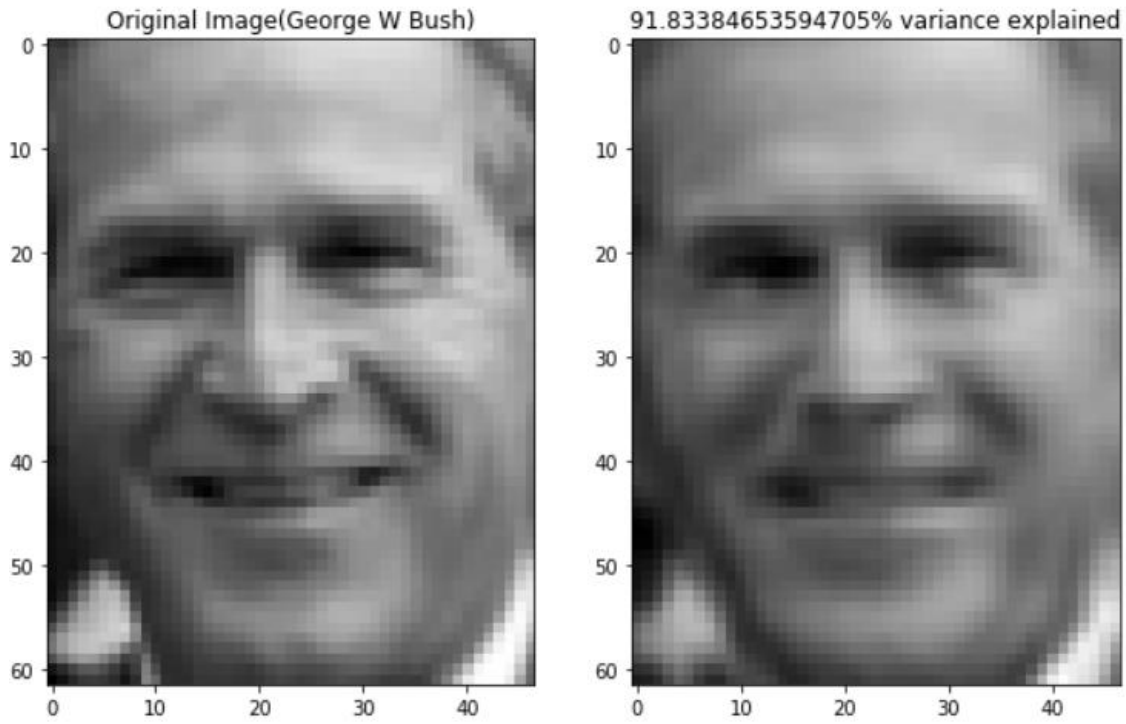
```
original shape of train_x = (798, 2914)
original shape of test_x = (342, 2914)

train_pc and test_pc are the sets when PCA represented each image in terms of new 100 eigenvectors(or eigenfaces):-
Shape of transformed train_X(or train_pc) = (798, 100)
Shape of transformed test_X(or test_pc) = (342, 100)

Variance explained on train set by new 100-D = 0.9183741594862842
Variance explained on test set by new 100-D  = 0.8796906049304873

approx_train and approx_test are the inverse transformed images from the train_pc and test_pc :-
approx_train shape = (798, 62, 47)
approx_test shape = (342, 62, 47)


eigenvectors shape = (100, 2914)
eigenfaces shape(after reshaping) = (100, 62, 47)
```

Visualizing the data w.r.t. the eigenfaces.

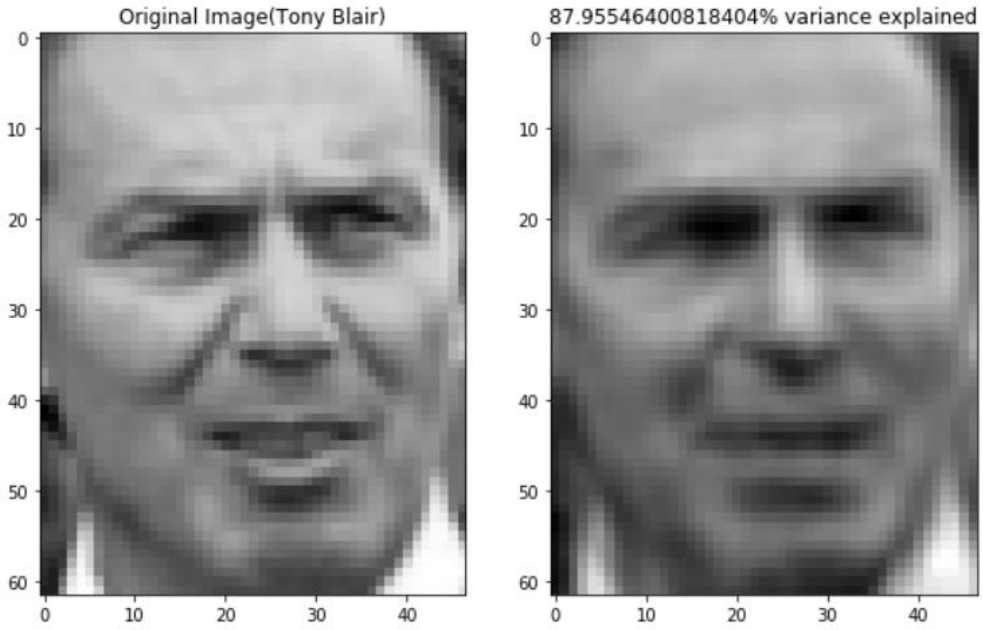
Using 100 eigenfaces, one of the image of George W Bush was plotted using inverse transform. i.e. using a linear of combination of 100 eigenfaces and then transforming it into original dimension 2914 for analysing how well this performed.

This is the values of George W Bush's image in eigenface space.

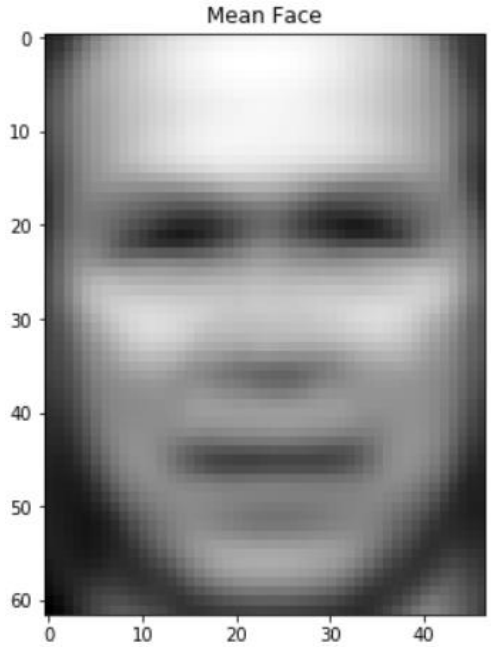
```
projecting image from train set on eigenfaces space
[ -175.36165    -1213.9962       663.50995      219.05484
  -330.2891       -29.44134      238.42355     -398.9524
  -184.02472     -103.257126     426.22607      219.84567
   227.60272     -383.9695        78.056175      -8.383299
  -177.83412     -326.1095      -198.33504      136.54405
    14.358753     153.9753      -189.87834     -141.88112
   194.54733     -112.0762        36.04404      226.22543
  -116.33371      -36.5725        63.65187      221.77205
   -36.20156      -31.847908      16.433556      63.354927
  -246.98378       47.36374     -44.24607     -244.21014
  -127.63591      -13.513142     153.89513        0.23078156
    -1.5487404   -106.918816     198.94832       61.358128
   118.07528      113.80192       39.09777       20.622395
   123.04775     -118.88479     -109.929214       0.89221764
    -0.6688132     78.47021       74.67274      -32.89592
    -5.182143     -24.772013      22.176537      27.047243
    40.17906      -56.38164      -36.73294       57.547
     0.8041878    -94.7445        61.58989      107.34185
    75.76773      -77.56638     -157.10106      -50.441936
   -64.50028      -61.372955     -67.90299      -54.26503
    -1.5380123    -63.087524      45.614         21.092758
   -36.719875      93.1417        18.027847      51.10135
   -42.7377        15.871099      42.561447       0.29511738
     3.6087742   -150.0195        76.55196     -155.68405
    24.222317      22.390308      71.415855      12.101528  ]
```



As we see 100 eigenfaces are representing the images quite accurately considering the massive decrease in dimensions. Also, note this image is taken form the training set, on which PCA was trained. What about its performace on images from testing set.

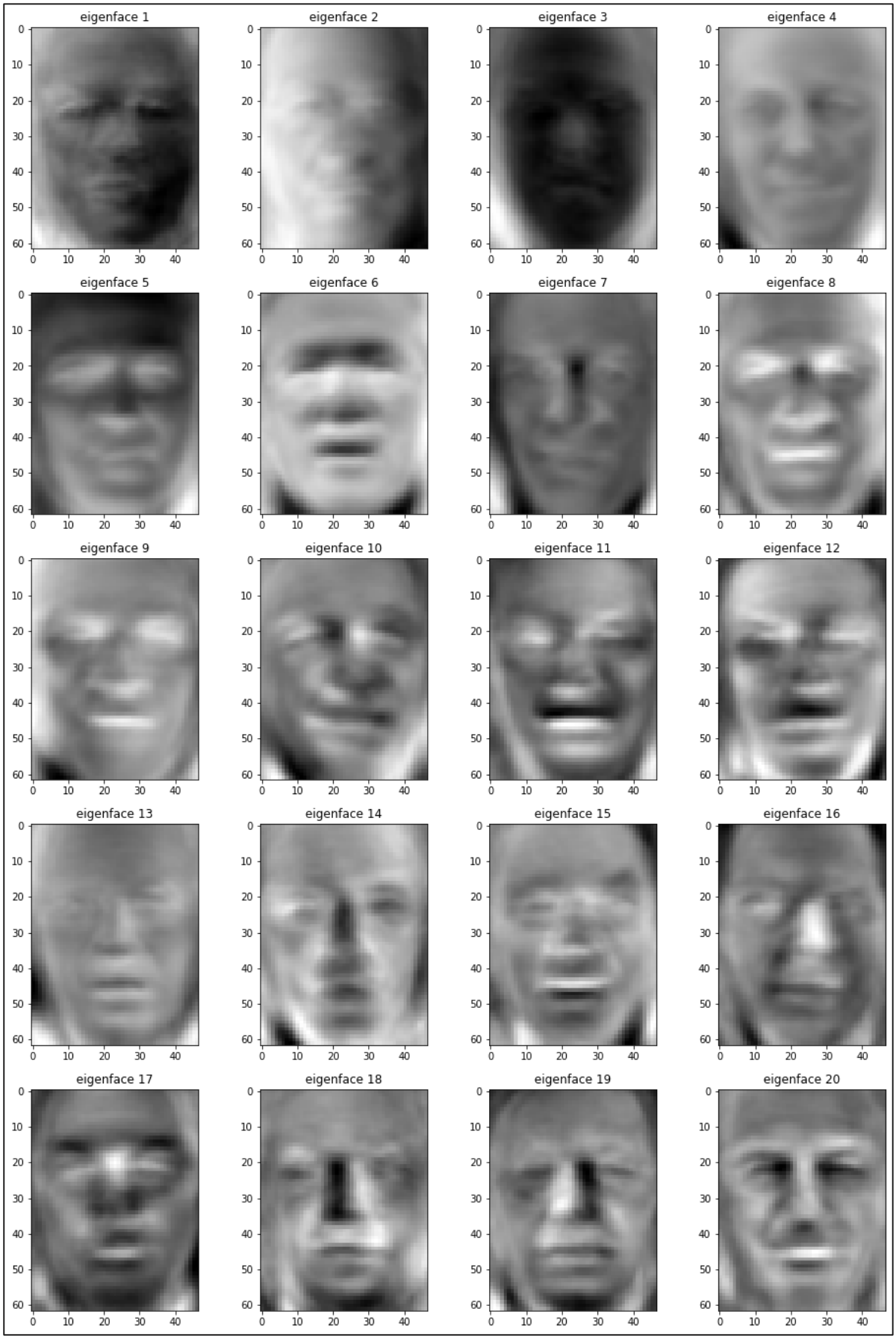


PCA can represent new images also fairly well. Although, faces with very uncommon features(outliers having huge variance from the mean face) might be represented poorly. The mean face is shown below that is mean of value all the faces in the train set.



Let us now also see how our eigenfaces look like. Remember each of the eigenfaces is actually an eigenvector representing particular direction in original input space of 2914 features. We are going to view only the first 20 eigenfaces.
Following points are to be kept in mind:

- 1st eigenface captures the maximum variance of all the faces from the mean face, 2nd eigenface captures 2nd highest variance of all faces from the mean face, and so on.
- These eigenfaces individually are nothing but “ghost faces”, i.e. no such actual face is found the training images. But any image can be represented as a linear combination of these eigenfaces. They look quite scary too.



First 20 eigenfaces

Nearest Neighbour Classifier on LFW

Once we reduced data to 100D dimensions per image. Now let's try to train a k-nearest neighbour classifier in these new 100 reduced dimensions. Just to remind we will train on train_pc set of shape $(798,100)$ and then check the results on train_pc and test_pc $(342,100)$.

Classification report of training data:

	precision	recall	f1-score	support
0	0.84	0.64	0.72	228
1	0.40	0.71	0.51	55
2	0.92	0.72	0.81	465
3	0.16	0.85	0.28	13
4	0.32	0.84	0.46	37
accuracy			0.70	798
macro avg	0.53	0.75	0.56	798
weighted avg	0.82	0.70	0.74	798

Confusion Matrix of training data:

```
[[145  22  24  16  21]
 [  2  39   3   6   5]
 [ 25  34 335  33  38]
 [  0   0   0  11   2]
 [  0   3   2   1  31]]
```

Classification report of testing data:

	precision	recall	f1-score	support
0	0.78	0.55	0.65	91
1	0.43	0.62	0.51	16
2	0.88	0.69	0.77	212
3	0.10	1.00	0.17	4
4	0.21	0.53	0.30	19
accuracy			0.64	342
macro avg	0.48	0.68	0.48	342
weighted avg	0.79	0.64	0.69	342

Confusion Matrix of test data

```
[[ 50   4  16  10  11]
 [  0  10   3   2   1]
 [ 14   8 146  19  25]
 [  0   0   0   4   0]
 [  0   1   1   7  10]]
```

As can be seen from classification report KNN classifier performed okay with accuracy of 70% on train data while 65% on test data. Number of neighbours were set to 11 for getting optimal results here.

PCA explained Variance and Performance

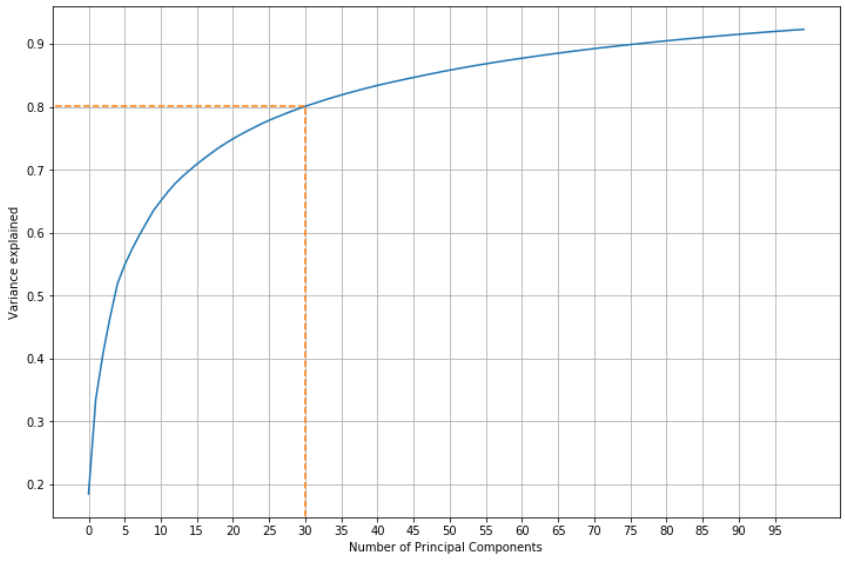
When employing PCA on our lfw data we set the number of components to 100, which explained about 91.83% variance in input data with 2914 components. We would now like to see how many components are sufficient to explain 80% total variance of input data.
First let us ratio of explained variance by first 100 eigenvectors.

```
this is the ratio of total variance of input explained by each PC (upto 100):
 [0.18504766 0.15055579 0.07243592 0.05893923 0.05196254 0.02997191
 0.02445717 0.02219532 0.0201753  0.01981962 0.01514933 0.01462891
 0.01303677 0.01107222 0.01001909 0.00963    0.00900656 0.0085527
 0.00830037 0.00732865 0.00697003 0.00655209 0.00607294 0.00593163
 0.00566211 0.00516548 0.00488804 0.00463696 0.00453454 0.00412421
 0.00405912 0.00380719 0.00377324 0.00365788 0.0035136  0.00333739
 0.00321314 0.00313268 0.00305839 0.00294016 0.00285341 0.00263027
 0.00255827 0.00253387 0.00246649 0.00244547 0.00240048 0.0023703
 0.00231304 0.00228523 0.0022162  0.00213958 0.00209969 0.00200936
 0.00196517 0.00189084 0.00187191 0.00181587 0.00178092 0.00173232
 0.00168196 0.00166358 0.00164337 0.00160119 0.00157574 0.00153717
 0.00149335 0.00147427 0.00143818 0.00143043 0.00140321 0.00134237
 0.001312   0.00129068 0.00127518 0.00123258 0.00122571 0.00120901
 0.00118367 0.00117479 0.00112919 0.00112227 0.00110312 0.00108941
 0.00106167 0.00103924 0.00101843 0.00100923 0.00099465 0.000992
 0.00096773 0.00093622 0.00093407 0.00090156 0.00088606 0.00084692
 0.0008409  0.00083231 0.00081833 0.00080958]
```

Now summing we can count the number of Principal components for whom the sum of explained variance ratio is equal to 0.80 or more.

```
PCs = 0  &  0.18504765629768372
PCs = 1  &  0.3356034457683563
PCs = 2  &  0.4080393686890602
PCs = 3  &  0.46697859838604927
PCs = 4  &  0.5189411416649818
PCs = 5  &  0.5489130467176437
PCs = 6  &  0.5733702145516872
PCs = 7  &  0.5955655388534069
PCs = 8  &  0.6157408412545919
PCs = 9  &  0.6355604622513056
PCs = 10  &  0.6507097948342562
PCs = 11  &  0.6653387015685439
PCs = 12  &  0.6783754667267203
PCs = 13  &  0.6894476916640997
PCs = 14  &  0.699466778896749
PCs = 15  &  0.7090967828407884
```

```
PCs = 16  &  0.7181033473461866
PCs = 17  &  0.7266560476273298
PCs = 18  &  0.7349564181640744
PCs = 19  &  0.742285072337836
PCs = 20  &  0.7492551039904356
PCs = 21  &  0.7558071981184185
PCs = 22  &  0.7618801384232938
PCs = 23  &  0.7678117719478905
PCs = 24  &  0.7734738835133612
PCs = 25  &  0.7786393649876118
PCs = 26  &  0.7835274050012231
PCs = 27  &  0.7881643627770245
PCs = 28  &  0.7926988988183439
PCs = 29  &  0.7968231132254004
PCs = 30  &  0.8008822286501527
```



So, first 31 components can explain 80.088% of the total variance explained. Now, we employed the transformed train and test data into KNN. Transformed train data and test data have shapes (798, 31) and (342, 31) respectively. The classification report and confusion matrix for train data is:

```
              precision    recall  f1-score   support

           0       0.86      0.62      0.72       240
           1       0.45      0.68      0.54        65
           2       0.88      0.73      0.80       441
           3       0.19      0.81      0.31        16
           4       0.29      0.78      0.42        36

    accuracy                           0.69       798
   macro avg       0.53      0.72      0.56       798
weighted avg       0.80      0.69      0.73       798

[[148  22  32  15  23]
 [  1  44   8   6   6]
 [ 22  30 321  30  38]
 [  1   0   0  13   2]
 [  0   2   3   3  28]]
```

The classification report and confusion matrix of test data:

```
              precision    recall  f1-score   support

           0       0.75      0.52      0.61        93
           1       0.39      0.53      0.45        17
           2       0.87      0.70      0.77       207
           3       0.07      0.75      0.13         4
           4       0.28      0.62      0.38        21

    accuracy                           0.63       342
   macro avg       0.47      0.62      0.47       342
weighted avg       0.77      0.63      0.68       342

[[ 48   5  17  11  12]
 [  0   9   3   4   1]
 [ 16   8 144  19  20]
 [  0   0   0   3   1]
 [  0   1   2   5  13]]
```

Performance on 31 components has accuracy of 69% and 63% on train and test data. Performance of pca with 31 components are similar to the pca with 100 components.

2. REFERENCES

- Wikipedia.org
- StackOverflow.com
- Towardsdatascience.com
- Scikit-Learn.org