Eigenfaces for Facial Recognition

Navtejpreet Singh Indian Institute of Technology, Ropar

2018csb1107 Punjab - 140001

# **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

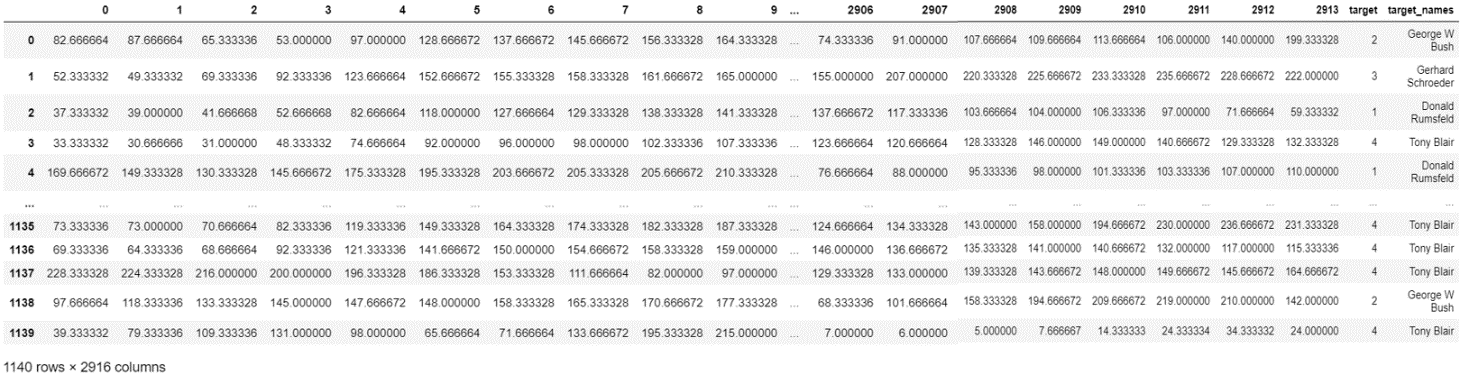
0 236 Colin Powell

1 121 Donald Rumsfeld

2 530 George W Bush

3 109 Gerhard Schroeder

4 144 Tony Blair



**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 number of examples and are number of dimensions.

Using PCA we get 100 eigenvectors with eigenvalues where

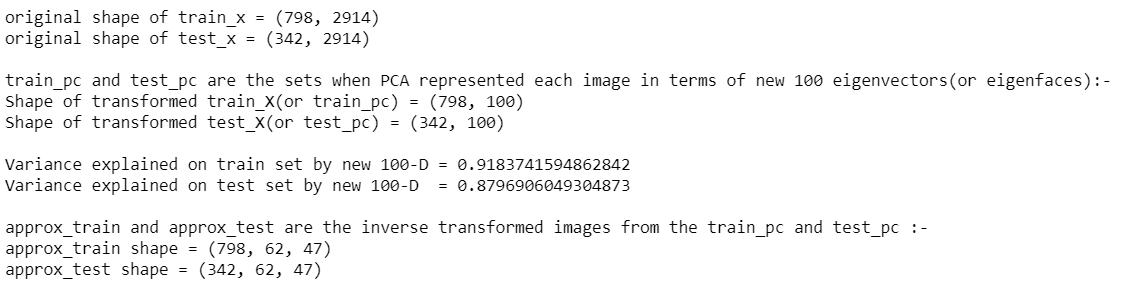
Eigenvector is a dimension vector in input space that represents the direction of maximum variance;

Eigenvector is a dimension vector in input space that represents the direction of maximum variance in direction orthogonal to ;

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.

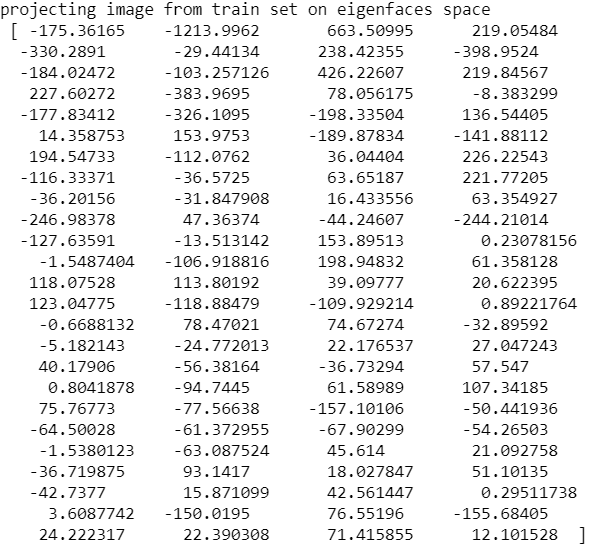


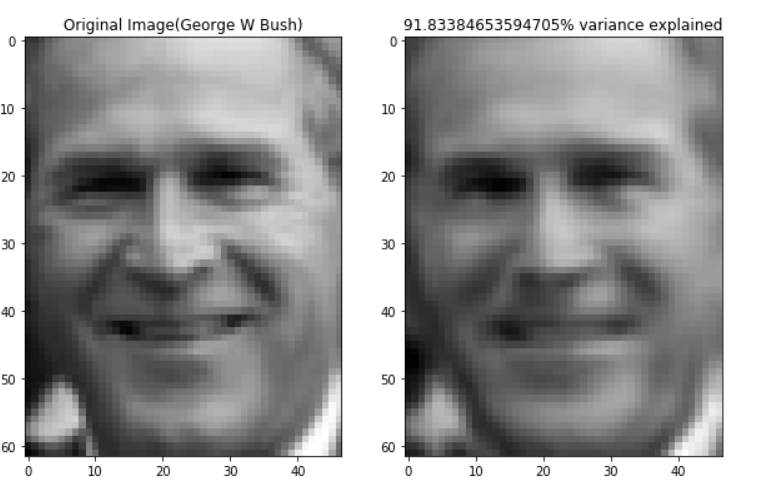


**Visualizing the data w.r.t. the eigenfaces.**

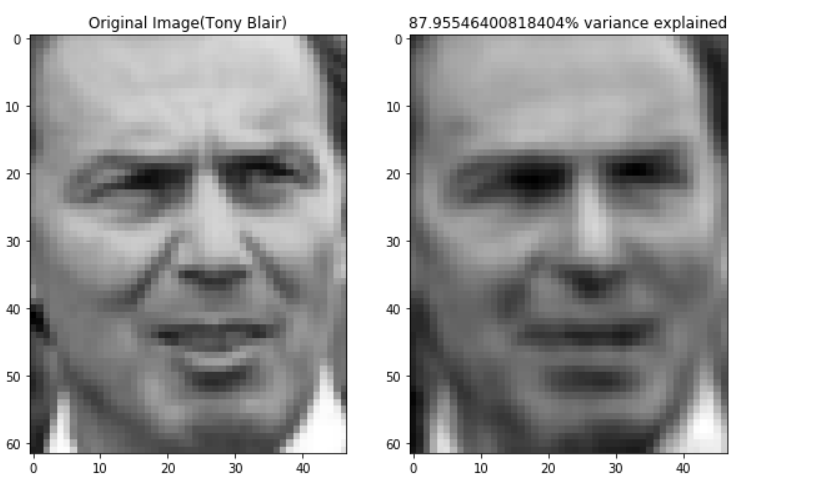
Using 100 eigenfaces, one of the image of George W Bush was plotted using inverse transform. i.e. using a linear of combination of 100 eigenfaces and then transforming it into original dimension 2914 for analysing how well this performed.

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

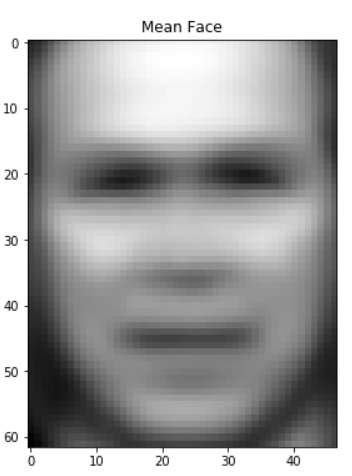




As we see 100 eigenfaces are representing the images quite accurately considering the massive decrease in dimensions. Also, note this image is taken form the training set, on which PCA was trained. What about its performace on images from testing set.



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.

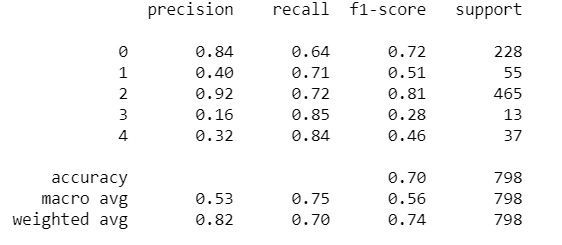


First 20 eigenfaces

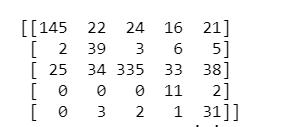
**Nearest Neighbour Classifier on LFW**

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 and then check the results on train\_pc and test\_pc.

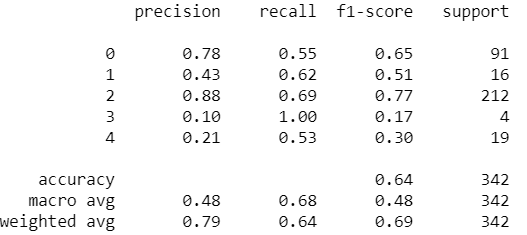
Classification report of training data:



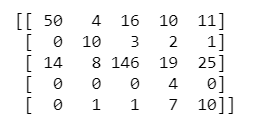
Confusion Matrix of training data:



Classification report of testing data:



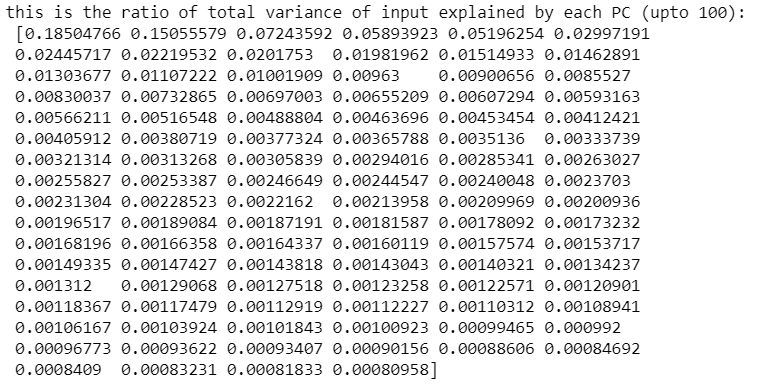
Confusion Matrix of test data



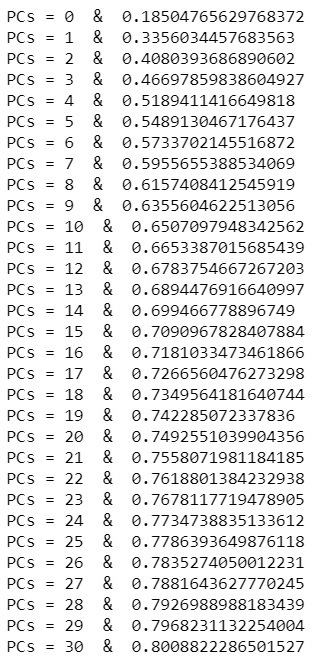
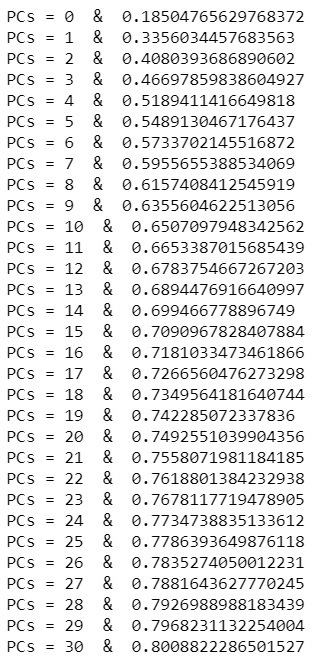
As can be seen from classification report KNN classifier performed okay with accuracy of on train data while on test data. Number of neighbours were set to 11 for getting optimal results here.

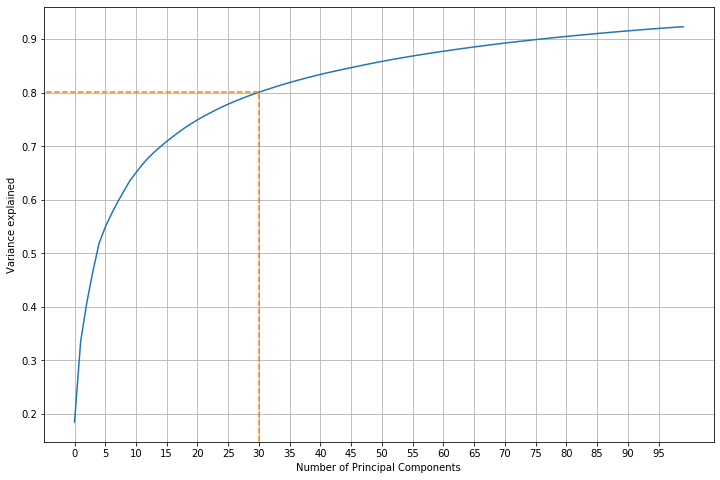
**PCA explained Variance and Performance**

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.

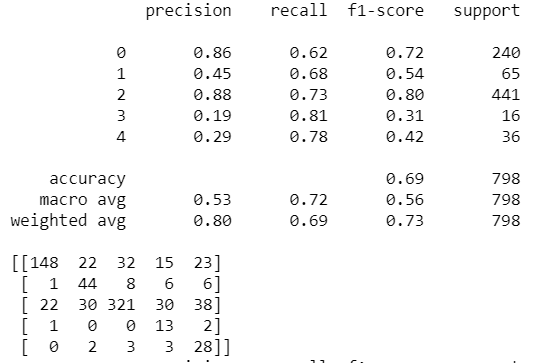
First let us ratio of explained variance by first 100 eigenvectors. 

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.

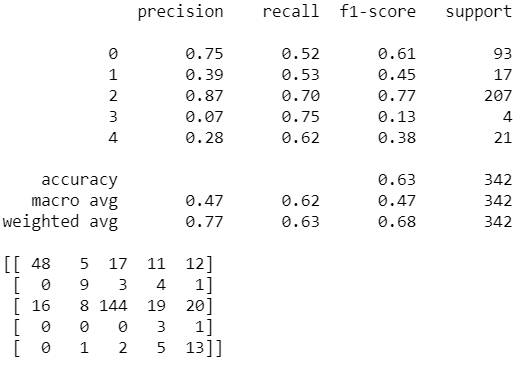
 



So, first 31 components can explain 80.088% of the total variance explained. Now, we employed the transformed train and test data into KNN. Transformed train data and test data have shapes (798, 31) and (342, 31) respectively. The classification report and confusion matrix for train data is:



The classification report and confusion matrix of test data:



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.

# **REFERENCES**

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