

Face Recognition and Procesing Using a Machine Learning Technique: Capstone Project Submitted to the Data Incubator Program By Rubayyat Mahbub

Introduction ¶

Have you ever imagined how social media sites automatically recognize and tag your face when you upload a photo? Face recognition and processing is one of the most popular application area of computer vision where computer vision research is utilized in both military and commercial products. Face recognition is the process of identifying or verifying a person from an image and comparing the selected features from the image with a given database. Both face recognition and processing find a wide variety of application in security systems, device unlock, criminal identification, as well as in mobile phones and social media sites for face tagging. In a broader sense, image processing and recognition is also very popular in other fields such as the health sector.

My capstone project for the Data Incubator Program is face recognition and processing using a machine learning technique. I am using the PCA technique followed by KNN for this purpose.

The dataset: Yale Face Database

The dataset that I am using contains 165 face images of 15 indivifuals with different face expressions

Yale Face Database



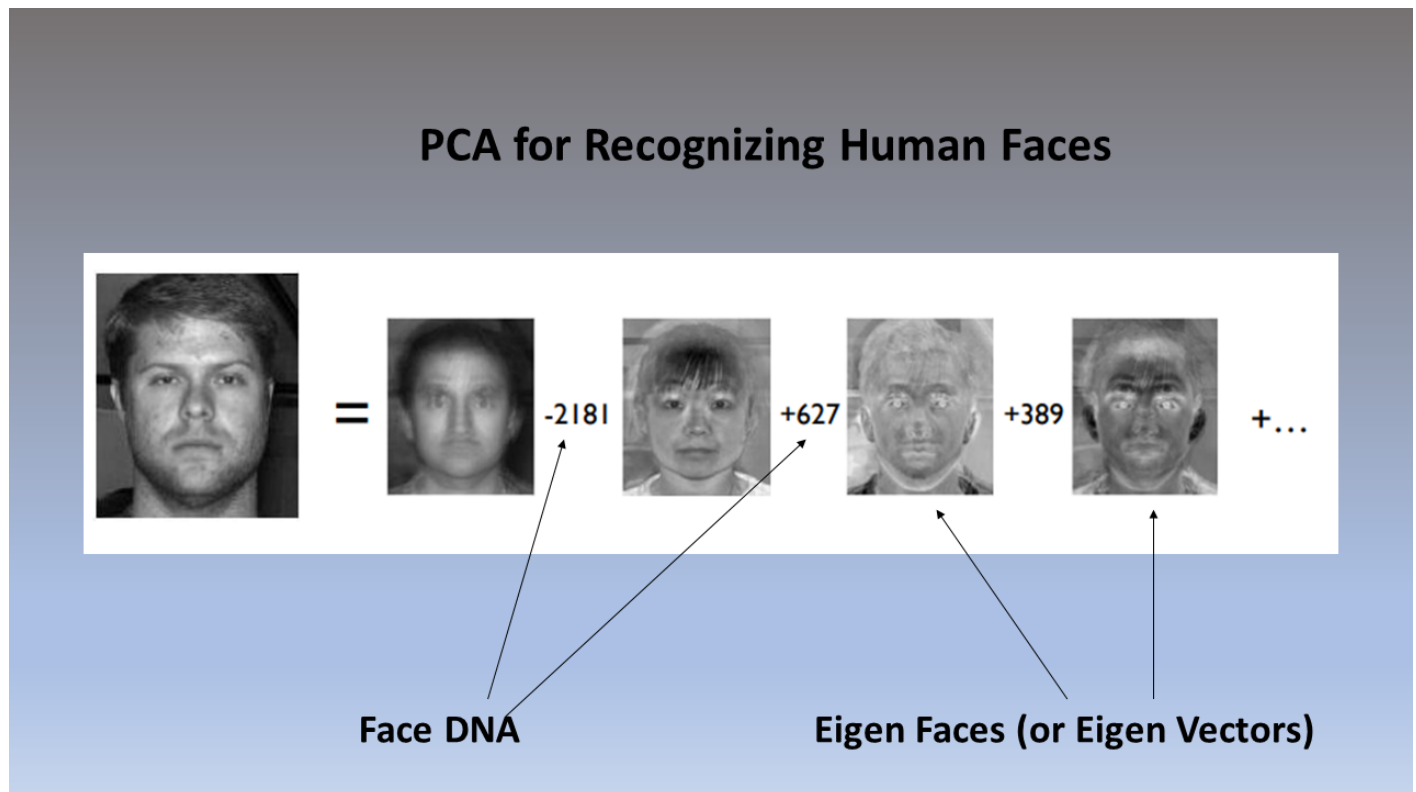
How PCA Works

PCA involves finding 'Eigen Faces' of the whole face database and 'Face DNA' corresponding to each face image. Eigenfaces is the name given to a set of eigenvectors of the dataset when they are used in the computer vision problem of human face recognition. Each eigenface though appear ghostly, carries some information of the total face database, the weight of which is represented by their corresponding eigenvalues. The linear combination of eigenfaces and face DNA allows any face in the database to be reconstructed.

Details regarding eigen faces can be found in the case study for Face Detection:

http://vision.jhu.edu/teaching/vision08/Handouts/case_study_pca1.pdf

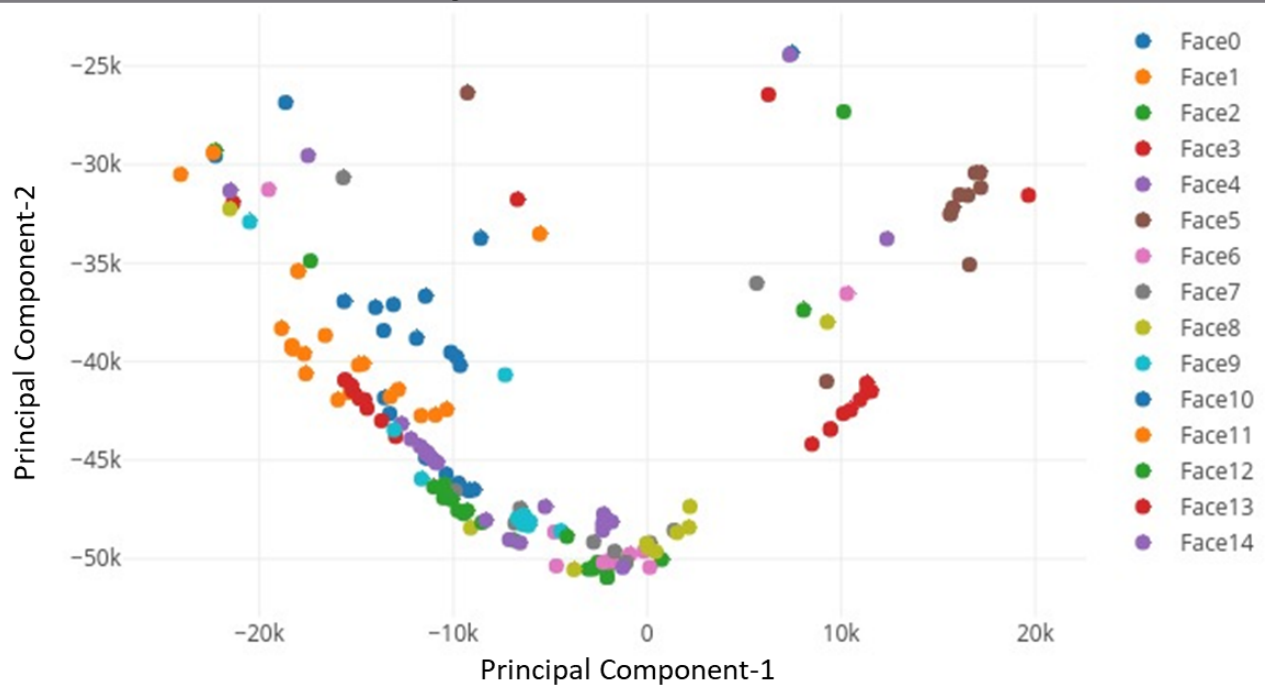
(http://vision.jhu.edu/teaching/vision08/Handouts/case_study_pca1.pdf)



All eigenfaces are not required to reconstruct and recognize a particular face. Some eigenfaces may correspond only to the noise in the data. The number of eigenfaces we consider sufficient enough to classify and recognize each face properly is directly related to the dimensionality reduction.

For example, a 2D representation of 15 Yale faces using the first two eigenvectors is plotted below:

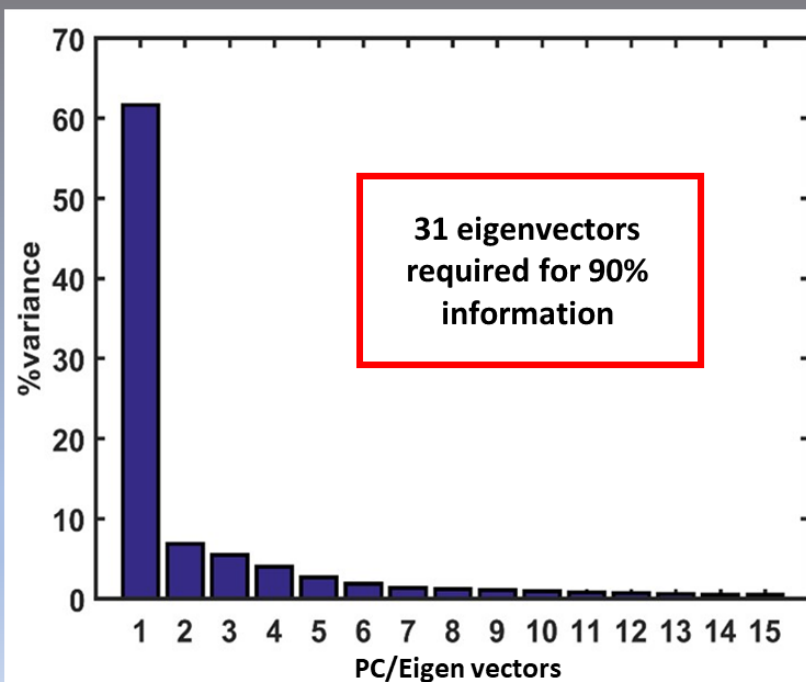
2D Representation of Yale Faces



The interactive version of this plot hosted by plotly can be found in this link: <https://plot.ly/~Rubayyat/0.embed> (<https://plot.ly/~Rubayyat/0.embed>)

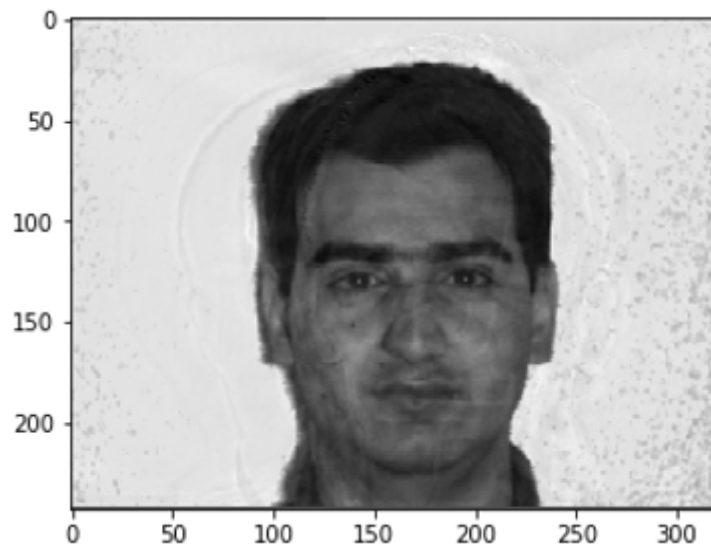
From the overlap of 2D data points of different face images, it is clear that the first two eigenvectors are not sufficient to separate the different face images of this database. The results are in alignment to the bar graph plotted below for Yale face database that shows the % variance carried out by first 15 eigenvectors. Through similar plot, I have computed that at least 31 are required to carry out 90 percent of the information of the yale face images.

Bar Graph Showing the % Information Carried Out by Eigen Vectors

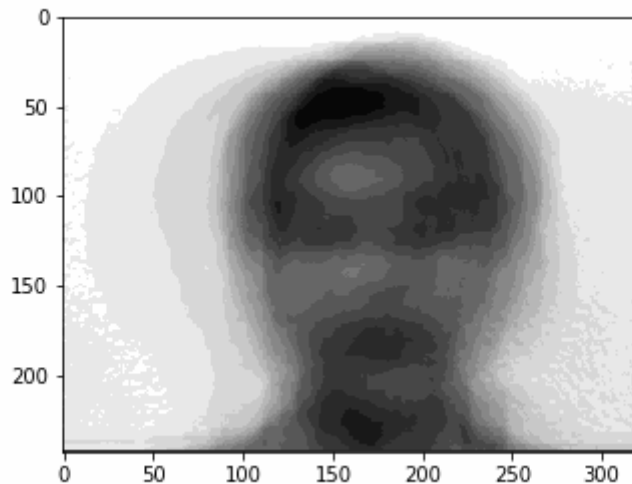


The fact is also supported by the animation shown below, where we see that the reconstruction of a face image improves with addition of eigenvectors/dimension from 0 to 100

Real Face:



Reconstructed Face:



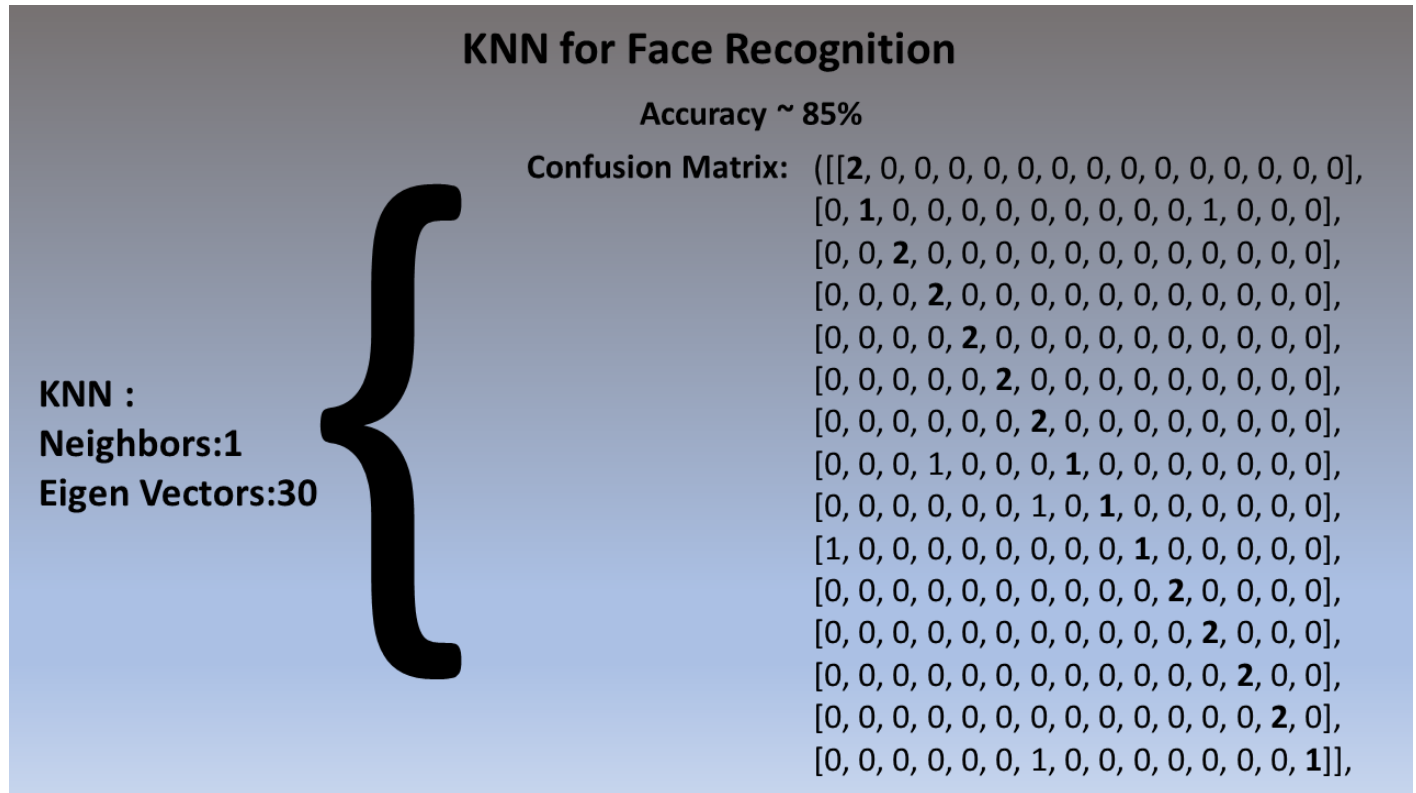
click on the link below to see the animation:

https://github.com/rmahbub/Data_Incubator_Challenge/blob/master/PCA_ANIMATION.gif

(https://github.com/rmahbub/Data_Incubator_Challenge/blob/master/PCA_ANIMATION.gif).

Face Recognition

I cross-validated the number of eigenvectors required for proper classification and recognition of the Yale face images. I found that using 30 eigenvectors with 1 nearest neighbor gave me a classification accuracy close to 85% with KNN. The confusion matrix is shown below:



Flask Based Face Recognition Web App

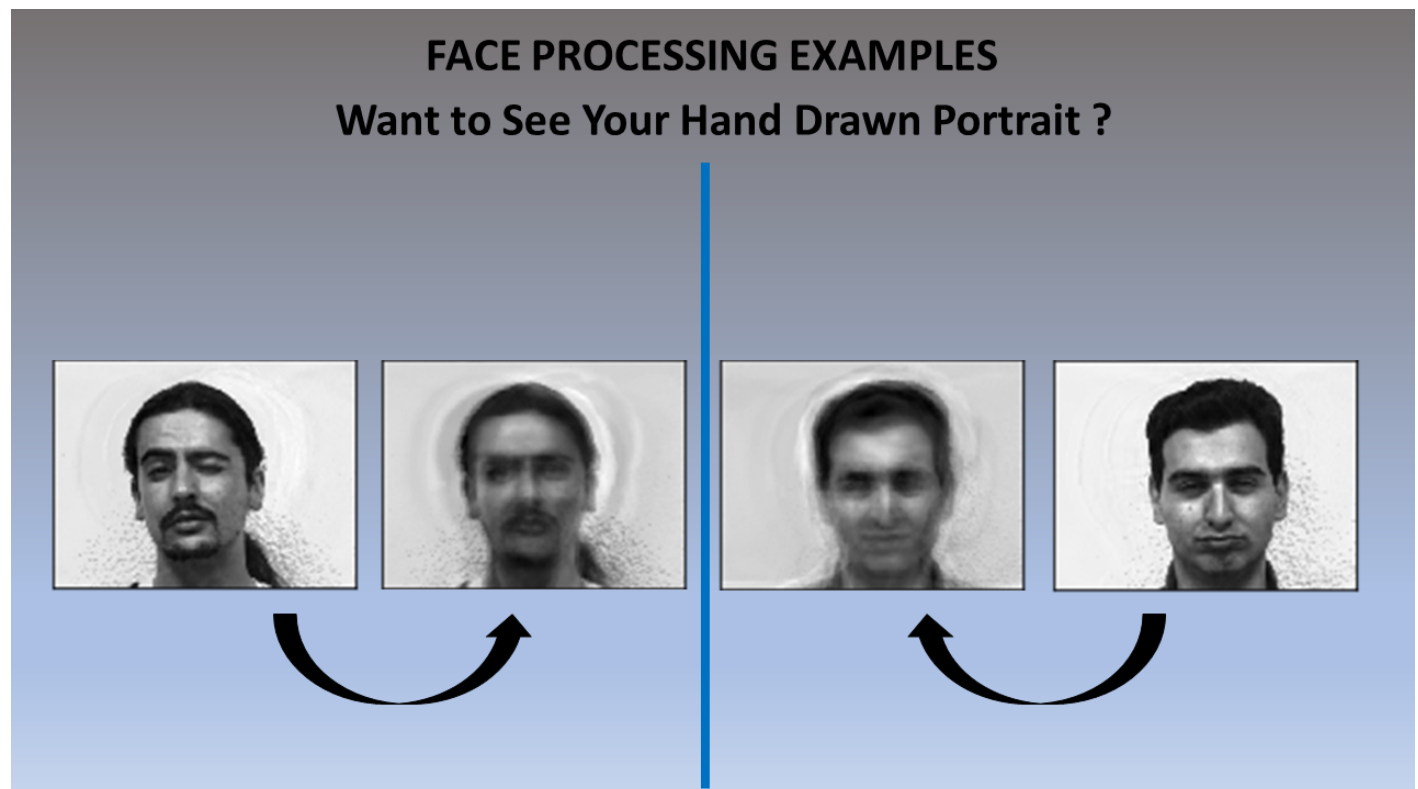
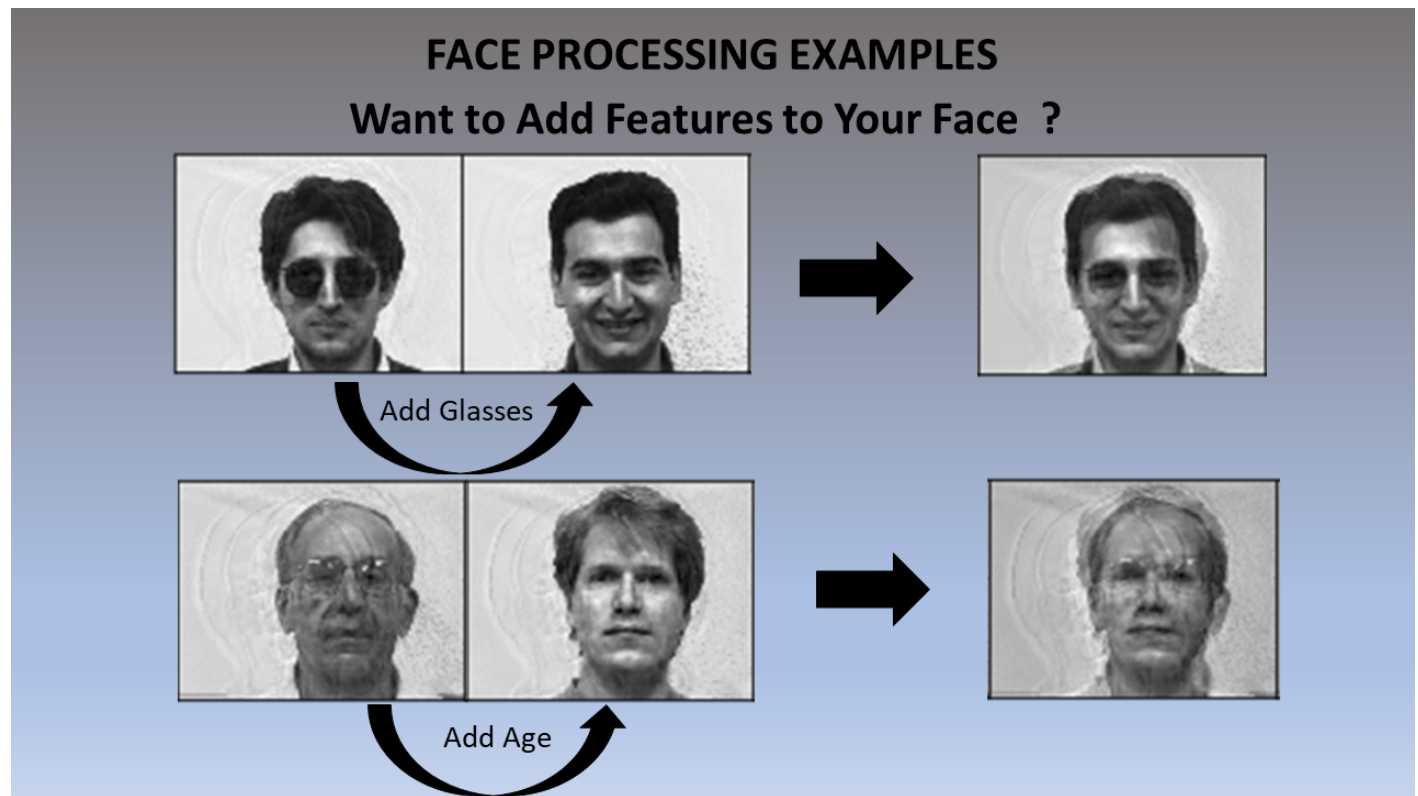
The trained KNN and saved eigen faces were loaded into a flask app using Heroku which given a face image can detect whether the face is present in the database or not and can also identify the person ID associated with the face image.

Here is the link to the web app: <https://rmahbub-face-recog-webapp.herokuapp.com/> (<https://rmahbub-face-recog-webapp.herokuapp.com/>).

Detail codes for the web app along with some sample images to try the app can be found in this github repository: https://github.com/rmahbub/Data_Incubator_Capstone_Project.git (https://github.com/rmahbub/Data_Incubator_Capstone_Project.git).

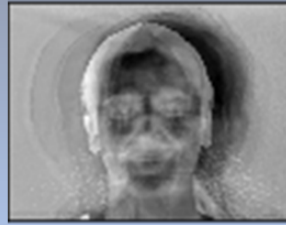
Face Processing

PCA also allows for Face Processing. Face Processing finds a wide variety of applications in fun apps in social media sites, mobile phones, etc. Some examples performed on the Yale Face Database are shown below:



FACE PROCESSING EXAMPLES

Want to See Your Negative Face ?



Detail codes regarding the analysis discussed here can be found at:

https://github.com/rmahbub/Data_Incubator_Capstone_Project.git

(https://github.com/rmahbub/Data_Incubator_Capstone_Project.git).