**Introduction**

Eigenfaces is the name given to a set of eigenvectors when they are used in the computer vision problem of human face recognition. The eigenvectors are derived from the covariance matrix of the probability distribution over the high-dimensional vector space of face images. The eigenfaces themselves form a basis set of all images used to construct the covariance matrix. This produces dimension reduction by allowing the smaller set of basis images to represent the original training images. Classification can be achieved by comparing how faces are represented by the basis set.

By a rule of thumb, we choose the number of eigen values based on how many it takes to cover 95% of variance. Those would correspond to the number of eigen vectors required.

Our analysis is spread across 3 datasets – IMFDB, IIIT-CFW and Yale.

Feature types we use for analysis - Eigen Face, Kernel Face, Fisher Face, Kernel Fisher Face, VGG Features and ResNet Features.

**Methodology**

The libraries used are -

scikit-learn, matplotlib, Pillow, numpy, pandas, scipy and matplotlib.

Except for VGG and ResNet features, in-built functions from sklearn are used for generating features.

We compress the data into 3D [Fig 1]and check for all combinations of datasets and feature types.

Number of eigen values are chosen based on the rule of thumb mentioned earlier, i.e., eigen values encompassing 95% variance. This is used to compress and reconstruct the images to see how well our classification performs.

An MLP has been used to classify the reconstructed images. The classification metrics have been plotted to compare the efficiency in reconstruction.

A tSNE dimenstionality reduction is also shown .

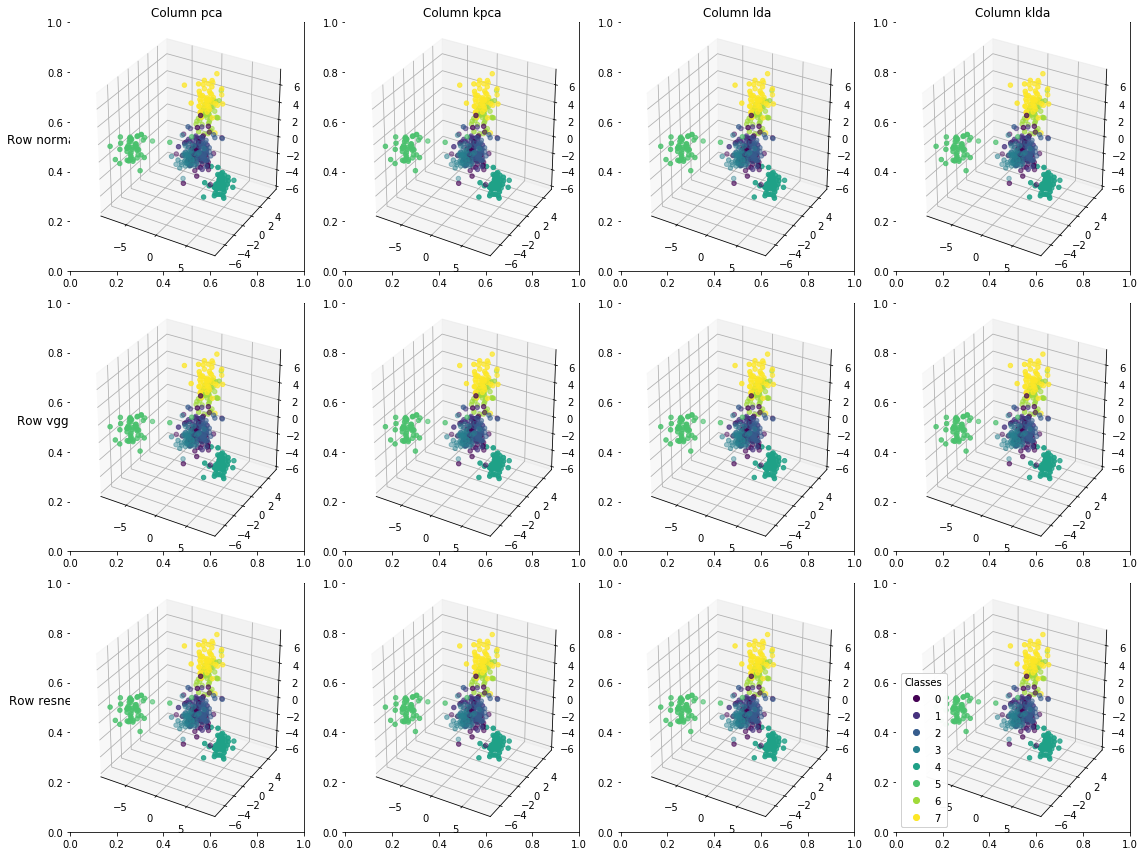


Fig 1(a). - IMFDB dataset compressed to 3D

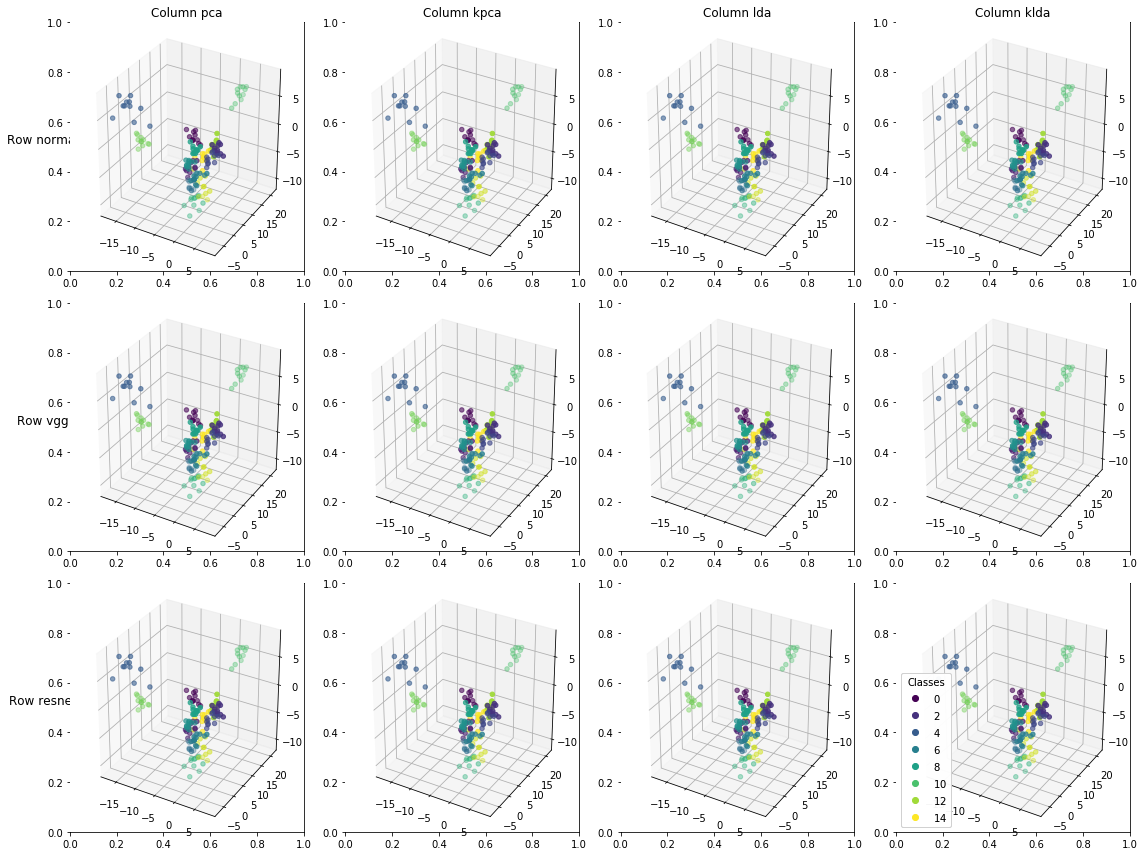
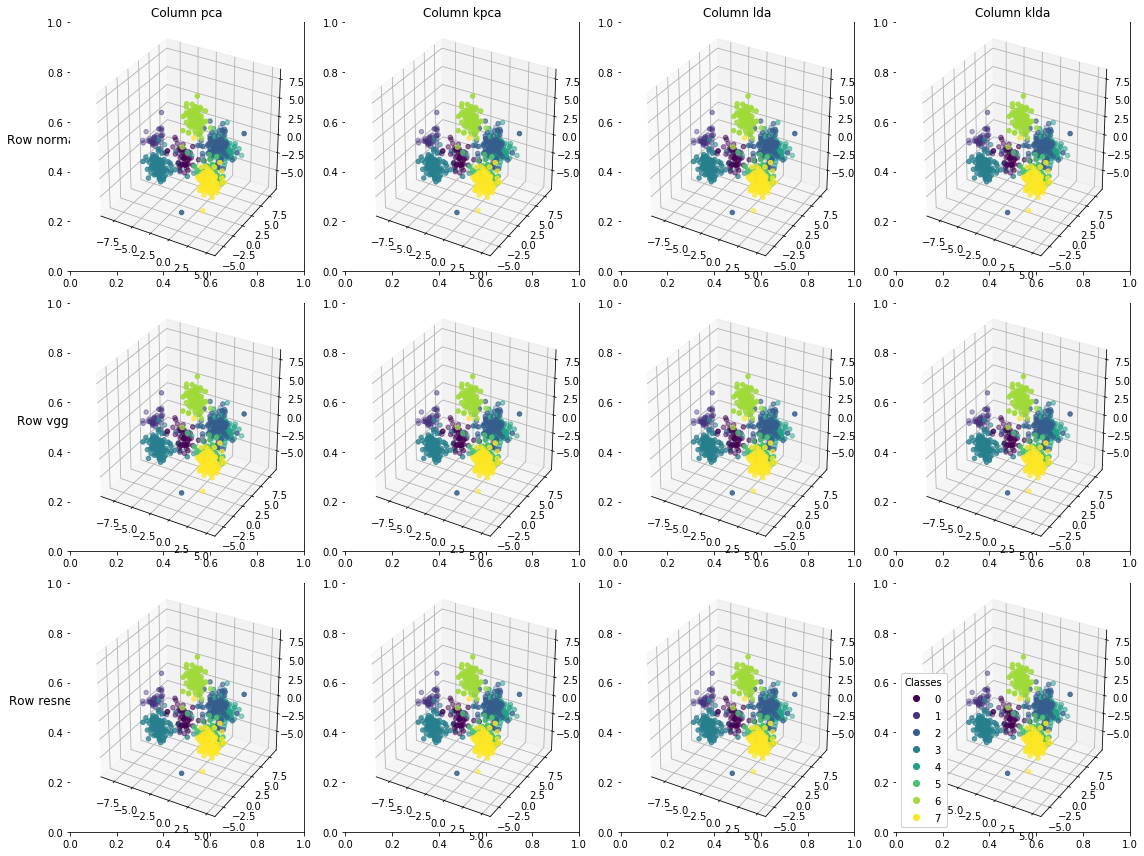


Fig 1(b). - Yale dataset compressed to 3D

Fig 1(c). - CFW dataset compressed to 3D

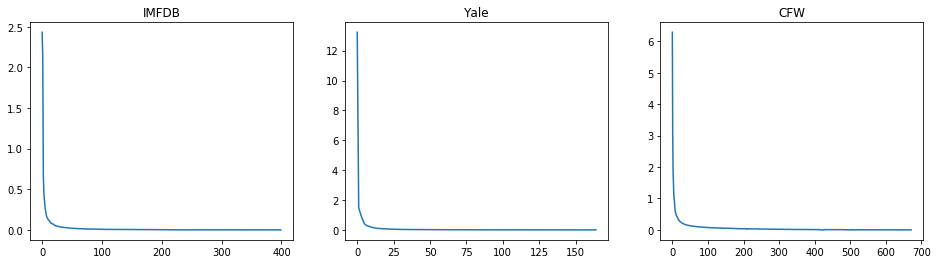


Fig 2. The Eigen value spectra for each dataset

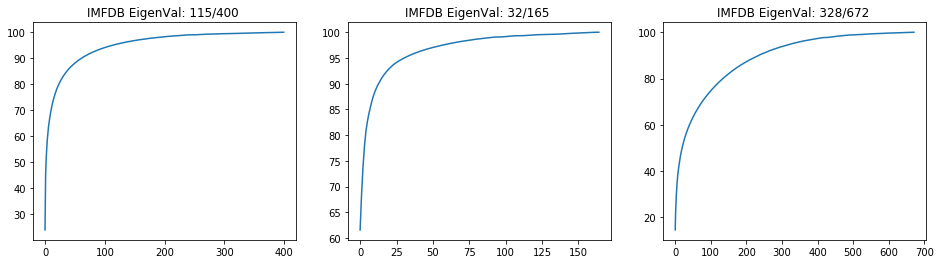
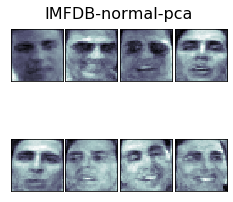


Fig 3. The increase in cumulative variance with eigen values for each dataset.



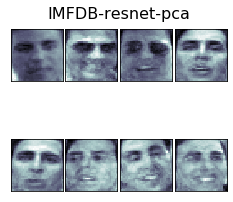


Fig 4. A sample comparison of performance between normal, vgg and resnet PCA features.

The classification results are shown below -

Feature Dimension Space Accuracy Precision f1-score

0 normal-pca 115 0.608333 0.668726 0.659682

1 normal-kpca 115 0.600000 0.617253 0.609804

2 normal-lda 3 0.741667 0.779443 0.776636

3 normal-klda 3 0.741667 0.779443 0.776636

4 vgg-pca 115 0.600000 0.674603 0.663074

5 vgg-kpca 115 0.600000 0.617253 0.609804

6 vgg-lda 3 0.741667 0.779443 0.776636

7 vgg-klda 3 0.741667 0.779443 0.776636

8 resnet-pca 115 0.608333 0.653771 0.636308

9 resnet-kpca 115 0.600000 0.617253 0.609804

10 resnet-lda 3 0.741667 0.779443 0.776636

11 resnet-klda 3 0.741667 0.779443 0.776636

Fig 5(a) – IMFDB

Feature Dimension Space Accuracy Precision f1-score

0 normal-pca 32 0.877551 0.740000 0.728889

1 normal-kpca 32 0.816327 0.648519 0.674965

2 normal-lda 3 0.816327 0.787734 0.744815

3 normal-klda 3 0.816327 0.787734 0.744815

4 vgg-pca 32 0.877551 0.708889 0.710635

5 vgg-kpca 32 0.816327 0.648519 0.674965

6 vgg-lda 3 0.816327 0.787734 0.744815

7 vgg-klda 3 0.816327 0.787734 0.744815

8 resnet-pca 32 0.857143 0.714127 0.714141

9 resnet-kpca 32 0.816327 0.648519 0.674965

10 resnet-lda 3 0.816327 0.787734 0.744815

11 resnet-klda 3 0.816327 0.787734 0.744815

Fig 5(b) – Yale

Feature Dimension Space Accuracy Precision f1-score

0 normal-pca 328 0.331683 0.398494 0.364525

1 normal-kpca 328 0.351485 0.316790 0.314541

2 normal-lda 3 0.900990 0.881190 0.873517

3 normal-klda 3 0.900990 0.881190 0.873517

4 vgg-pca 328 0.371287 0.373847 0.332299

5 vgg-kpca 328 0.351485 0.316790 0.314541

6 vgg-lda 3 0.900990 0.881190 0.873517

7 vgg-klda 3 0.900990 0.881190 0.873517

8 resnet-pca 328 0.371287 0.390587 0.362107

9 resnet-kpca 328 0.351485 0.316790 0.314541

10 resnet-lda 3 0.900990 0.881190 0.873517

11 resnet-klda 3 0.900990 0.881190 0.873517

Fig 5(c) – IIIT-CFW