ENEE439M - Project 1

Anurag Bansal

Maryland, United States

Abstract

This project implements the following classifiers for face recognition.

- Maximum Likelihood Estimation with Gaussian assumption followed by Bayes rule
- K - Nearest neighbor (NN) rule
- Principal Component Analysis followed by Bayes classifier and K-NN rule
- Linear Discriminant Analysis followed by Bayes classifier and K-NN rule

Three datasets are used in this project:

1. DATA - Cropped images of 200 subjects, 3 images each, each image of size 24x21. The file data.mat has a variable face of size (24x21x600). The images corresponding to the person labeled n, n = 1, . . . , 200 can be indexed in Matlab as face(:,:,3*n-2), face(:,:,3*n-1) and face(:,:,3*n). The first image is a neutral face, the second image has a facial expression, and the third image has illumination variations.
2. ILLUMINATION - Cropped images of 68 subjects under 21 different illuminations. You can access the i th illumination of the j th subject as illum(:, i, j). Each image has been already reshaped to a vector.
3. POSE - Cropped images of 68 subjects under 13 different poses. You can access the i th pose of the j th subject as pose(:, :, i, j).

1. Implementation Theory

The implementation begins with getting the data from user defined dataset and converting it to suitable algorithm specific format which is, i th training image or test image of the j th subject in data(f, i, j) with f features for all samples of the training and the test class. The code also has the provision to get random training and test samples using another dataset structuring code, apart from the default set division. Once the data is ready in the data(:,i,j) format, it is then divided in the 66% to 34% training to test ratio first in default format/random format and passed to the various algorithms.

1.1. Maximum Likelihood Estimation with Gaussian assumption followed by Bayes rule

- We start with calculating the sample mean for each class that is given by μ_i, $\forall i = 1, ..., c$
- The next step is to calculate the covariance matrix for each class that is given by Σ_i, $\forall i = 1, ..., c$
- We find that Σ_i is singular, therefore we add a small constant diagonal matrix to make it non-singular, in our case we add a identity matrix, any matrix with constant diagonal elements can be chosen as the constant scalar value remains constant for all classes.
- Then calculate the discriminant function parameters A, B and C for the equation $g_i(x) = \bar{x}^t A_i \bar{x} + B_i^t \bar{x} + C_{i0}$ for each class
- Once the training is complete and the vectors of the discriminant parameter is obtained we start the test procedure by calculating the discriminant value for each test image and selecting the class with the highest discriminant value as the output.
- The outputs are stored in the vector and compared with the inputs to obtain the test accuracy for the algorithm. The results for the test are discussed in the section below.

1.2. K - Nearest neighbor (NN) rule

- For each test data $\bar{x}$, calculate its Euclidean distance $d(x_i, x)$, $\forall \bar{x}$.
- We then assign the class ω_k to $\arg\min_i g_i(x) = k$

1.3. Principal Component Analysis followed by Bayes classifier and K-NN rule

- First we have to mean center the data, i.e. substract the mean of the overall dataset from the training and test images. In the this step we therefore study the deviations from the mean only.

- Next we calculate the covariance matrix of data. The covariance matrix contains all necessary information to rotate the coordinate system.

- The rotation helps to create new variables which are uncorrelated, i.e. the covariance is zero for all pairs of the new variables. The decorrelation is achieved by diagonalizing the covariance matrix C. The eigenvectors V belonging to the diagonalized covariance matrix are a linear combination of the old base vectors, thus expressing the correlation between the old and the new time series. We find the eigenvalues of the covariance matrix C by solving the equation det(C D * E) = 0. The eigenvalues D of the covariance matrix, i.e. the diagonalized version of C, gives the variance within the new coordinate axes, i.e. the principal components. We now calculate the eigenvectors V and eigenvalues D of the covariance matrix C.

- Calculating the data set in the new coordinate system. We need to flip new data left/right since the second column is the one with the larges eigenvalue.

- The eigenvalues of the covariance matrix (the diagonals of the diagonalized covariance matrix) indicate the variance in this (new) coordinate direction. We can us this information to calculate the relative variance for each new variable by dividing the variances according to the eigenvectors by the sum of the variances.

1.4. Linear Discriminant Analysis followed by Bayes classifier and K-NN rule

- Calculate the sample mean for each class m_i,$\forall i = 1...c$

- Calculate the total sample mean m

- The next step is to calculate the covariance matrix for each class that is given by Σ_i, $\forall i = 1, ..., c$

- The within-class scatter matrix is defined as S_w :

$$S_w = \sum_{i-1}^{n} (x_i - \mu) * (x_i - \mu)^t$$

- The between-class scatter matrix is defined as S_b :

$$S_b = \sum_{i-1}^{n} n_k * (\mu_i - \mu) * (\mu_i - \mu)^t$$

, where n is the number of classes, $\boldsymbol{\mu}$ is the overall sample mean, and n_k is the number of samples in the k-th class.

2. Experiments

2.1. Experiment 1 - General Run with Different Datasets and Random Selections

- Datasets Used - data.mat, illumination.mat and pose.mat
- Classifiers Used - Naive Bayes, KNN, PCA + Naive Bayes, PCA + KNN, LDA + Naive Bayes, LDA + KNN
- Training Data - Select training images in the 2:1 ratio images for each subject for training for data and pose dataset and 3:2 ratio for illumination dataset. For example, if we want to test the impact of facial expressions from dataset 1, then we select the neutral face and the face with illumination variations as training dataset. Here for this experiment we have set it to neutral and illumination for training. But the data is also experimented when the neutral, illumination and expression are ignored and the data is selected randomly by the algorithm from classes in the ratio of division
- Testing Data - We select the remaining images from the datasets as the training images.

Dataset	NB	KNN	PCA+NB	PCA+KNN	LDA+NB	LDA+KNN
Data	66.5	40	67	41.5	78	78.5
Illumination	95.73	36.02	93.08	95.58	98.12	97.22
Pose	75	47.79	80.51	47.98	74.26	74.26

Table 1: Dataset Default

This table gives the test accuracy for the data that is selected randomly from classes. The reduction factor for the dimensionality reduction algorithms PCA and LDA is selected to 50% for all dataset.

Dataset	NB	KNN	PCA+NB	PCA+KNN	LDA+NB	LDA+KNN
Data	38.5	24	40.5	27	86.5	85.00
Illumination	99.78	66.16	99.57	54.99	99.78	100
Pose	58.45	47.79	54.78	34.55	83.08	88.6

Table 2: Randomly Selected Training and Test Samples from class

2.1.1. Observations

- From the default dataset it can be observed that Bayes performs better that K-nearest neighbor.
- It can be observed that when the training samples are chosen at random in every class in dataset the test accuracy decreases significantly for both Naive Bayes and k-nn classifier.

2.2. Experiment 2 - Effect of variation(illumination, expression) in images

- Datasets Used - data.mat
- Classifiers Used - Naive Bayes, KNN, PCA + Naive Bayes, PCA + KNN, LDA + Naive Bayes, LDA + KNN
- Training Data - Consists of various combinations of 2 of the neutral, illumination and expression
- Test Data - The remaining one is fed in as the test image.

Training	Neutral Expression	Neutral Illumination	Expression Illumination	Average
Test	Illumination	Expression	Neutral	
Naive Bayes	64	66.5	72	67.5
K-NN	59.5	65	55.5	60
PCA+Bayes	63.5	66	71	66.8
PCA+KNN	58.5	65.5	55.5	59.8
LDA+Bayes	66	78	88.5	77.5
LDA+KNN	63	78	88.5	76.5

2.2.1. Observations

- The average test accuracy for Bayes and K-NN classifiers is 60% to 67% respectively which is comparatively low.
- LDA+Bayes and LDA+KNN classifiers performs the best and obtains 88.5% accuracy, which contradicts the intuition that the true error can't be decreased if we project the data to the lower dimension. It can be due to the zero frequency problem when you don't observe a count-valued attribute for a particular class. And to overcome this we applied a simple solution, add-one smoothing.
- From the results we observe that the training set containing the variation in expressions and illumination achieves the best accuracy.
- We can observe that the results with PCA are not that good, which shows that the PCA as an unsupervised dimensionality reduction technique would undermine the testing accuracy when its used for supervised learning.

2.3. Experiment 3 - Effect of Train to Test Ratio

- Datasets Used - illumination.mat and pose.mat
- Classifiers Used - Naive Bayes, KNN
- Training Data - The training data is selected in the percent from 10 to 90 from the datasets.
- Testing Data - We select the remaining images from the datasets as the training images.

Classifier	10%	20%	30%	40%	50%	60%	70%	80%	90%
Naive Bayes	26.47	44.55	51.96	88.69	95.74	99.82	100	100	100
KNN	22.98	29.76	39.41	54.52	42.50	92.28	100	100	100

Table 3: Dataset - Illumination

Classifier	10%	20%	30%	40%	50%	60%	70%	80%	90%
Naive Bayes	24.88	55.3	56.21	66.54	83.21	80.29	75.00	68.14	77.95
KNN	24.87	52.5	50.81	62.68	75.49	73.24	69.84	61.76	67.65

Table 4: Dataset - Pose

2.3.1. Observations

- The accuracy of Naive Bayes starts at a very low point at training set size of 24% then jumps to over 50% at training set size of 20%. For the Pose dataset as the training dataset changes from 10% to 50% the testing accuracy increases for both Bayes and k-nn classifiers and the maximum accuracy occurs with the training size of 50%, further as the training percentage is increased, the testing accuracy remains almost consistent and doesn't fluctuate much. This sows that when the training data is $< 50\%$ the model tries to under fit and therefore we cannot obtain high test accuracy. For the training percentage $>$ 50, the outliers in the data increase and therefore the trained model is undermined and when it is large the problem of over-fitting happens.

- For illumination dataset, the test accuracy increases as the training percent increases and as the training percentage is > 70 we obtain 100% accuracy.

- As discussed in class, the test error for K-NN classifier cannot be worse than twice the Bayesian error rate, that is what can be observed, Bayes classifier performs better than the k-nn classifier.

2.4. Experiment 4 - Variation in feature Reduction factor

- Datasets Used - data.mat, illumination.mat and pose.mat

- Classifiers Used - PCA/LDA+Naive Bayes, PCA/LDA+KNN

- The the feature reduction factor is varied from 1 to C-1, where C = number of classes.

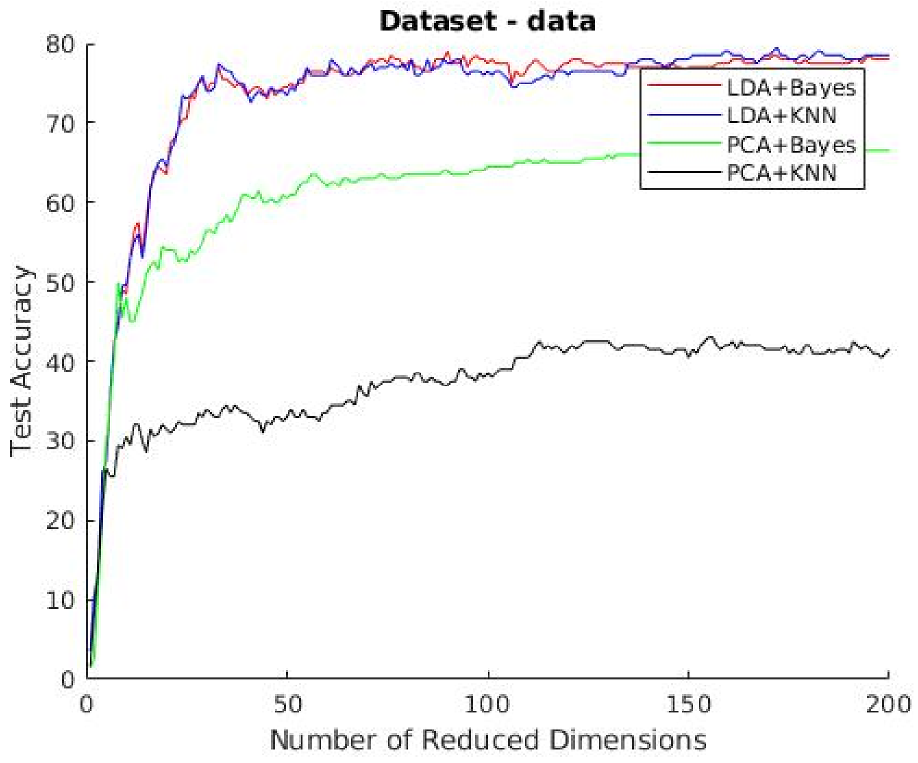

Figure 1: Effect of variation in feature reduction factor

2.4.1. Observations

- It can be observed that as the number of features increases the test accuracy also increases. The LDA can be observed to work only for (C-1) feature reductions at max. since Σ_b is of rank C 1.

3. Conclusions

In this project we implemented 2 classifiers viz. Naive Bayes Classifier and K-NN classifier and we applied dimensionality reduction to these algorithms. Various experiments were performed using the 3 different datasets. The first experiment showed that even though the training and test sample division happened randomly the trend in the test accuracy was maintained though the test accuracy dropped for certain cases. In the second experiment, we saw that the variance in the dataset can lead to good classification. The third, experiment shows the criteria of over-fitting and under-fitting and the how the increase in training sample affects the test accuracy. The last experiment shows the effect of variation of the feature reduction factor from 1 to C-1 features.

4. Extra-Credit

- Datasets Used - data.mat
- Classifiers Used - Naive Bayes, KNN, PCA + Naive Bayes, PCA + KNN, LDA + Naive Bayes, LDA + KNN
- An alternative viewpoint for the dataset DATA is to consider 3 classes, corresponding to the states of ”Happy”, ”Neutral” and ”Illuminated”. The goal here would be to train a classifier to detect the state given a test sample from any subject that is not used in training.

Classifiers	Test Accuracy
Naive Bayes	87.25
K-NN	79.41
PCA+Bayes	70.58
PCA+KNN	70.09
LDA+Bayes	88.23
LDA+KNN	87.25