Project 1

Rohith Jayarajan
University of Maryland, College Park
(*e-mail:* rohith23@terpmail.umd.edu)

Contents

1 Overview

In this project, given a labeled data set of faces, the following machine learning techniques are used for the classification problem for recognizing faces.

1. Naive Bayes Classifier
2. K Nearest Neighbors Classifier
3. Naive Bayes Classifier with Principal Component Analysis (PCA)/ Linear Discriminant Analysis (LDA)
4. K Nearest Neighbors Classifier with Principal Component Analysis (PCA)/ Linear Discriminant Analysis (LDA)

The above mentioned techniques, their differences and shortcomings are studied and their performances are compared. The faces are classified based on the labels of subjects. Also the images from the first data set (data.m), are classified based on the states i.e. neutral, happy or illuminated.

2 Background on the Data Sets Used

Face recognition is performed on the below listed data sets

1. data.mat This data set consists images of size $24x21$ for 200 subjects with 3 faces per subject. The first image of the three faces is a neutral face, the second is a face with a facial expression and the third image has variations in illumination.
2. pose.mat The pose data set consists images of size $48x40$ for 68 subjects with 13 different poses (images) for each subject.
3. illumination.mat The illumination data set consists images of size $48x40$ for 68 subjects with 21 different images having varying illumination for each subject.

3 Machine Learning Technique Description

3.1 Naive Bayes Classifier

1. Calculate the class mean μ_i. $\forall i = 1,2,...C$
2. Calculate the class variance Σ_i. $\forall i = 1,2,...C$
3. Add an identity matrix I to Σ_i if Σ_i is singular as we need Σ_i^{-1} in the steps that follow.
4. Use the discriminant function in equation 1

$$g_i(x) = \frac{1}{2}(x-\mu_i)^T\Sigma_i^{-1}(x-\mu_i) - \frac{d}{2}\ln 2\pi - \frac{1}{2}\ln\det(\Sigma_i) + \ln P(w_i) \quad \forall_i = 1,.C \quad (1)$$

   We use this equation as we assume that the distribution of our data set is a multivariate Gaussian distribution. Also as all classes are assumed to have equal probability, $\ln P(w_i)$ in equation 1 is same for all classes.
5. Assign test data to class ω_m if $g_m(x)$ gives the maximum value.

3.2 K Nearest Neighbors Classifier

1. Calculate the Euclidean distance $d_E(x,x_i)$ between the test data and all other data elements i in the training set.

2. Assign test data to class ω_m decided by the highest vote from the k selected nearest neighbors based on lowest value of d_E.

3.3 Principal Component Analysis (PCA)

1. Firstly, mean normalize the features of the data set before splitting the data set into test and train sets.
2. Calculate the co-variance matrix $\Sigma = \frac{1}{m} X^T X$ where X is the training set.
3. Compute the Eigen-vectors of matrix Σ and select the first k Eigen-vectors $U_{reduced}$ (which are sorted in order of decreasing magnitude of the corresponding Eigen-values). Here k is the number of Principal Components needed.
4. Multiply the training data by the matrix $U_{reduced}$ which transforms the training data to a new training data matrix $X_{new} = U_{reduced} X$. Do the same for the test Data $Y_{new} = U_{reduced} Y$
5. Use this on the Naive Bayes Classifier where the training and test data are X_{new} and Y_{new}.

3.4 Linear Discriminant Analysis (LDA)

1. Compute the total mean μ, class mean μ_i and class variance matrix $\Sigma_i \; \forall_i = 1, 2..C$ where C is the number of classes.
2. Compute the within-class scatter matrix given by the following equation 2

$$S_w = \sum_{i=1}^{C} \sum_{\vec{X} \in D_i} (\vec{X} - \vec{\mu}_i)(\vec{X} - \vec{\mu}_i)^T \tag{2}$$

3. Compute the between-class scatter matrix given by the following equation 3

$$S_b = \sum_{i=1}^{C} n_i (\vec{\mu}_i - \vec{\mu})(\vec{\mu}_i - \vec{\mu}_i)^T \tag{3}$$

4. Solve the generalized Eigen-value problem $S_b \vec{W}_i = \lambda_i S_w \vec{W}_i$ and select the first k Eigen-vectors $U_{reduced}$ (which are sorted in order of decreasing magnitude of the corresponding Eigen-values). Here k is the number of Principal Components needed.
5. Multiply the training data by the matrix $U_{reduced}$ which transforms the training data to a new training data matrix $X_{new} = U_{reduced} X$. Do the same for the test Data $Y_{new} = U_{reduced} Y$
6. Use this on the Naive Bayes Classifier where the training and test data are X_{new} and Y_{new}.

4 Experiments

4.1 Changing the Train-Test Ratio on Pose and Illumination Dataset

Data Set	Machine Learning Technique	Train-Test Split(Train/Test in %) NOTE: Ceil of %val*data is taken								
		10	20	30	40	50	60	70	80	90
pose	Naive Bayes	24.88	55	56.20	75.42	83.08	80.29	68.13	52.94	77.94
	K-Nearest Neighbors (K = 1)	44.11	52.5	50.81	69.74	75.49	73.23	61.76	45.58	67.64
illumination	Naive Bayes	18.87	54.22	61.34	91.42	96.61	99.81	100	100	100
	K-Nearest Neighbors (K = 1)	19.03	43.93	45.37	50.73	42.79	93.19	100	100	100

Fig. 1. Effect of Train-Test Ratio

Analysis:

- For the data set 'pose.mat' we see an increase in test accuracy from 24.88% to 83.088% as the Test-Train split is increased from 10% to 50% for the Naive-Bayes classification and an increase from 44.11% to 75.49% for K-Nearest Neighbors. But as the Train-Test ratio is further increased, the test accuracy drops. This is due to the fact that when the Train-Test ratio is small (<40%) the classifier is under-fitting the data and when it is large, the problem of over-fitting arises.
- But this is not the case for the data set 'illumination.m' as the test accuracy continues to increase and remains 100% when the Train-Test ratio is increased.
- Theoretically, the test error for K-Nearest Neighbors is not worse than twice the Bayesian error rate and this is also observed in this experiment as Naive-Bayes classifier performs better than K-Nearest Neighbors classifier.

4.2 Comparison of the Techniques on Data Set 'data.m'

Train Data	Test Data	Accuracy (in %) [K = 1; Reduced Dimensions of Features: 199]						
		Naive-Bayes	KNN	Naive-Bayes with PCA	KNN with PCA	Naive-Bayes with LDA	KNN with LDA	Average
Neutral Expression	Illumination	64	59.5	63.5	58.5	51	50	57.75
Expression Illumination	Neutral	72	55.5	71	55.5	88	86.5	71.41
Neutral Illumination	Expression	66.5	65	66	65	77.5	78.5	69.75
Average		67.5	60	66.83	59.66	72.16	71.66	

Fig. 2. Comparison of the Different Techniques Used

Analysis:

- From the value of average accuracy, we can observe that using LDA with Naive-Bayes gives us the best classifier among the different methods we have used. Also, if we compare the average accuracy values of methods that use Naive-Bayes v/s KNN, Bayes clearly outperforms KNN in all scenarios and this is consistent with the theoretical results. The fact from theory that KNN has an error rate less than or equal to twice the Bayesian error rate is also reinforced by the results.
- It is evident by the results in Figure 2 that using LDA on any of the two methods gives us a better classifier than the others. This tells that LDA does a better job than PCA as it attempts to find a lower dimensional feature space that maximizes the separability of class.
- If we look at the properties of the data sets used for training and testing, we can deduce that using Expression and Illumination as train data and Neutral as test data gives us the best classifier for face recognition. This is because the classifier also learns information about the illumination in this setup, whereas in the setup where illumination is used for testing gives us the poorest average accuracy as the classifier had no chance to learn about the variations in illumination.

4.3 Changing the Subspace of Feature Dimension in LDA

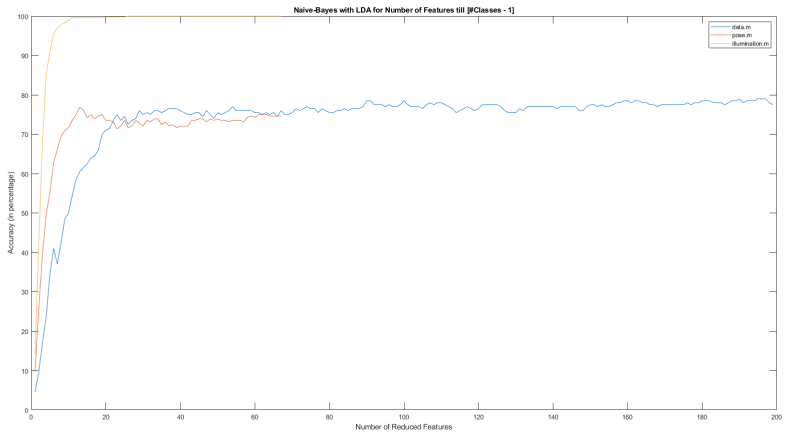

Fig. 3. Effect on Choosing Different Subspace of Features

Analysis:

- When the dimension of feature vector was reduced to a very low value (say, 0 to 10 features), the Naive-Bayes classifier displayed a poor classification accuracy. This could be due to the fact that a good decision can't be made when the data is under-represented.
- We see a trend that as the number of reduced feature dimension increases, the accuracy of the Naive-Bayes classifier increases quickly. As number of useful features increases, the classifier is able to make a more informed decision regarding the class.

- The accuracy peaks at a reduced dimension of $C-1$ where C is the number of classes and this is in agreement with the theory. This tells us that dimension reduction techniques remove the correlation in the data and we could use this useful data, void of correlations, to make good predictions.

5 Conclusion

The two Machine Learning techniques- Naive Bayes Classifier and K-Nearest Neighbors were implemented on the face recognition data sets. Along with these techniques, dimension reduction techniques, PCA and LDA, were also built and deployed for the above mentioned classifiers. The variation in obtained results were observed and studied by changing the hyper-parameters; which in our case were -1. Train-Test Ratio 2. value of k in K- Nearest Neighbors 3. The number of reduced features dimension of training and test data in PCA and LDA.

Increasing the value of train-test ratio often helped in giving a higher accuracy. And in K-Nearest Neighbors, increasing the value of K was associated with a drop in the accuracy of the classifier.

6 Extra Credit

As an extension for the project, a Naive Bayes classifier was built for the data set 'data.m' to classify the faces as having a state of 'Neutral', 'Happy', or 'Illumination'. An accuracy of 87.25% was obtained with a Train-Test Split of 66% and 90% accuracy was obtained with a Train-Test Split of 50%.