E1 213 – Pattern Recognition and Neural Networks

Assignment # 1

Codes available at: https://github.com/vineeths96/Pattern-Recognition-1*

Vineeth S **M.Tech Artificial Intelligence** **SR No: 16543**

Problem 1

Problem 1 is a 2-class classification problem with two-dimensional feature vectors. The class conditional densities are normal. The problem has three subproblems.

Sub problem A

For subproblem a, the class conditional densities were assumed to be normal and the parameters for the densities were estimated from training data using Maximum Likelihood Estimation. For each class conditional density, the size of data used for estimation was varied as 5, 10, 25, 75 (random sampling from training data) and the estimated density was used to implement a Bayes classifier. The prior probabilities were taken to be equal for both classes since we have equal number of samples from both. Using these, posterior densities were calculated and Bayes classifier was implemented. For each case, a nearest neighbour classifier was also implemented using the same data used for estimation. The performance of both the classifier was observed by testing against the full test set (which is tabulated below).

N (training data size)	Bayes classifier accuracy	Nearest neighbour classifier accuracy
5	0.575	0.44
10	0.71	0.72
25	0.755	0.715
75	0.765	0.685
100	0.76	0.69

Th efficiency of both the classifiers increases (neglecting rare dips) with the number of samples used for estimating the density/class. This is quite obvious from the fact that a greater number of samples leads to better estimation of parameters. More specifically, estimated parameters converge in probability to true parameters when the number of samples is sufficiently large. We can also see that in general Bayes classifier outperforms nearest neighbour classifier, which conforms to the established fact that Bayes classifier has the best performance (minimum error) amongst all classifiers. This turns out to be particularly true when the number of samples is quite large, which contributes in finding better estimates in densities. The low accuracy of Bayes classifier (compared to other subproblems in this problem) maybe due to the fact that the means of two distributions are close – one being [0,0] and other being [1,1].

** GitHub repository access is private as of now. Access can be made available, if necessary.*

Sub problem B

For subproblem b, the class conditional densities were assumed to be normal and were considered as a Gaussian mixture. The labels of the training data were discarded and hence an EM algorithm (from scikit-learn) was used for estimating the mixture density. The size of data used for estimation was varied as 10, 25, 50, 100, 150, 200 (random sampling from training data) and the estimated densities were used to implement a Bayes classifier. The prior probabilities were taken as the corresponding weight returned by the EM algorithm. Using these, posterior densities were calculated and Bayes classifier was implemented. The performance of both the classifier was observed by testing against the full test set (which is tabulated below).

N (training data size)	**Bayes classifier accuracy**
10	0.5
25	0.83
50	0.965
100	0.955
150	0.965
200	0.97

The accuracy of the classifier increases with the number of samples owing to the reasons explained in subproblem a. The Bayes classifier performs exceptionally well in classification of test data for which one of the reasons could be relatively distant means at [0,0] and [3,3]. This type of mixture density estimation is used in unsupervised learning such as clustering. During training, it was observed that estimated means were close to true values and estimated covariance matrices were off from the true values by a margin. I later found out that this is due to the weight factor associated with the EM algorithm estimation.

Sub problem C

For subproblem c, two cases were considered – assuming both the class conditional densities to be normal and assuming one density as normal and other as exponential. In each case, the size of data used for estimation was varied as 5, 10, 25, 75 (random sampling from training data) and the estimated densities were used to implement a Bayes classifier. The prior probabilities were taken to be equal for both classes since we have equal number of samples from both. The two dimensions in exponential distribution was considered independent and then the parameters were estimated because the multivariate exponential distribution was complicated to deal with. Using these, posterior densities were calculated and Bayes classifier was implemented. The performance of the classifier in both cases was observed by testing against the full test set (which is tabulated below).

Class 1: Gaussian, Class 2: Gaussian

N (training data size)	Bayes classifier accuracy
5	0.98
10	0.98
25	0.99
75	0.99
100	0.99

Class 1: Gaussian, Class 2: Exponential

N (training data size)	Bayes classifier accuracy
5	0.83
10	0.98
25	0.98
75	0.98
100	0.98

The accuracy of both the classifier increases with the number of samples owing to the reasons explained in subproblem a. The classifier in which both densities are assumed to be Gaussian performs really good even for really small number of samples, again which I suspect is because of the relatively far means – at [0,0] and [3,6]. The classifier in which one of the densities was assumed to be exponential performs relatively better at higher number of samples. As explained earlier, multivariate exponential was hard to deal with and hence was assumed to be independent in each dimension, thus the joint density turned be the product of the one dimensional densities. By doing so, we lose the covariance information between the variables. Also, since exponential densities is defined for the positive values of random variable, the absolute value was taken for the training data set (for exponential). Even after these approximations, the classifier performed really well in classifying new data.

Problem 2

Problem 2 is a 2-class classification problem with twenty-dimensional feature vectors. The class conditional densities are normal. The problem has three subproblems.

Sub problem A

For subproblem a, the class conditional densities were assumed to be normal and the parameters for the densities were estimated from training data using Maximum Likelihood Estimation. For each class conditional density, the size of data used for estimation was varied as 10, 20, 50, 200, 300, 500 (random sampling from training data) for each class and the estimated density was used to implement a Bayes classifier. The prior probabilities were taken as the corresponding weight returned by the EM algorithm. Using these, posterior densities were calculated and Bayes classifier was implemented. For each case, a nearest neighbour classifier was also implemented using the same data used for estimation. The performance of both the classifier was observed by testing against the full test set (which is tabulated below).

N (training data size)	**Bayes classifier accuracy**	**Nearest neighbour classifier accuracy**
10	0.5	0.922
20	0.5	0.943
50	0.929	0.934
200	0.984	0.964
300	0.983	0.963
500	0.985	0.959

The general observations are same as in problem 1, except that here we are dealing with twenty dimensional distribution. The two covariance matrices are same in this case and equal to the identity matrix. Thus, each dimension is independent for each class. But since variance is (relatively) low, the density would be a narrow peak near the mean. This helps in classification as the overlap of densities would be less.

Sub problem B

For subproblem b, the class conditional densities were assumed to be normal and the parameters for the densities were estimated from training data using Maximum Likelihood Estimation. For each class conditional density, the size of data used for estimation was varied as 10, 20, 50, 200, 300, 500 (random sampling from training data) for each class and the estimated density was used to implement a Bayes classifier. The prior probabilities were taken as the corresponding weight returned by the EM algorithm. Using these, posterior densities were calculated and Bayes classifier was implemented. For each case, a nearest neighbour classifier was also implemented using the same data used for estimation. The performance of both the classifier was observed by testing against the full test set (which is tabulated below).

N (training data size)	Bayes classifier accuracy	Nearest neighbour classifier accuracy
10	0.51	0.76
20	0.5	0.753
50	0.761	0.758
200	0.877	0.801
300	0.881	0.794
300	0.898	0.799

The general observations are same as in problem 1, except that here we are dealing with twenty dimensional distribution. The two covariance matrices are same in this case and equal to the three times identity matrix. Thus, each dimension is independent for each class. But since variance is (relatively) high, the density would be a wide spread around the mean. This reduces the classification accuracy as the overlap of densities would be more and hence the classification may be erroneous.

Sub problem C

For subproblem c, the class conditional densities were assumed to be normal and the parameters for the densities were estimated from training data using Maximum Likelihood Estimation. For each class conditional density, the size of data used for estimation was varied as 10, 20, 50, 200, 300, 500 (random sampling from training data) for each class and the estimated density was used to implement a Bayes classifier. The prior probabilities were taken as the corresponding weight returned by the EM algorithm. Using these, posterior densities were calculated and Bayes classifier was implemented. For each case, a nearest neighbour classifier was also implemented using the same data used for estimation. The performance of both the classifier was observed by testing against the full test set (which is tabulated below).

N (training data size)	Bayes classifier accuracy	Nearest neighbour classifier accuracy
10	0.5	0.931
20	0.5	0.948
50	0.993	0.954
200	0.996	0.977
300	0.996	0.972
300	0.997	0.976

The general observations are same as in problem 1, except that here we are dealing with twenty dimensional distribution. The two covariance matrices are not equal in this case and have non zero values as entries in off diagonal locations. However, it is to be noted that the dimensions are no longer independent and have different correlations for the classes. This in fact helps in classification since the covariance information is also available for use in estimating densities for classification.

Problem 3

Problem 3 is a 2-class classification problem with one-dimensional feature vectors. The class conditional densities are mixture of gaussian densities. The problem has two subproblems.

Sub problem A

For subproblem a, three different classifiers were implemented –

1. Assuming class conditional densities to be mixtures of two Gaussians, estimating them using EM algorithm and implementing Bayes classifier (with weights as priors)
2. Assuming class conditional densities are to be single Gaussian, estimating them using Maximum Likelihood method and implementing Bayes classifier (with equal priors)
3. Implement nearest neighbour classifier using the same training data as used for estimating densities

The accuracies are listed below.

Method	**Accuracy**
EM Algorithm	0.925
MLE Gaussian	0.925
Nearest Neighbour	0.895

As we can see, the classifier with EM algorithm and MLE estimation for densities work well compared to nearest neighbour classifier. In this problem, each class conditional density is composed of two gaussians. Specifically, for this subproblem we can see that each density within a class have equal variance and close means – Class I having means 0 and 4, Class II having means 8 and 12. Hence, the class densities do not overlap significantly in this case, which in turn helps in classifying. The EM algorithm is able properly learn the parameters of distribution and even the MLE, where we assume that the mixture density is a single Gaussian performs well in estimation. The two densities within a class are relatively close enough that even after replacing them with a single Gaussian there is no significant decay in classification.

Sub problem B

For subproblem b, three different classifiers were implemented –

1. Assuming class conditional densities to be mixtures of two Gaussians, estimating them using EM algorithm and implementing Bayes classifier
2. Assuming class conditional densities are to be single Gaussian, estimating them using Maximum Likelihood method and implementing Bayes classifier
3. Implement nearest neighbour classifier using the same training data as used for estimating densities

The accuracies are listed below.

Method	**Accuracy**
EM Algorithm	0.72
MLE Gaussian	0.57
Nearest Neighbour	0.68

As we can see, the classifier with EM algorithm outperforms the other two classifiers in terms of accuracy of classification. The MLE performs the worst amongst all three classifiers, even behind nearest neighbour classifier. In this problem, each class conditional density is composed of two gaussians. Specifically, for this subproblem we can see that each density within a class have different variance and distant means – Class I having means 0 and 8 and variances 4 and 5, Class II having means 4 and 12 and variances 4 and 5. The class densities overlap significantly in this case since the means (and distribution) lie between other densities. This introduces ambiguity in classification and it pulls down the accuracy. In this case, the EM algorithm is able learn the parameters of distribution to a moderate extent. This is mainly because EM algorithm works by forming clusters of values that could be from the same distribution. But here there is significant overlap in the densities which introduces error in clustering into a wrong density from a wrong class. The MLE, where we assume that the mixture density is a single Gaussian, performs the worst for similar reasons. The densities across the classes overlap and hence replacing the two class conditional densities with a single Gaussian, essentially loses too much of information. This drags the accuracy below the other classifiers.

Problem 4

Problem 4 is a 2-class document classification problem. The data set consists of 2000 movie reviews with two classes positive and negative. The problem has two subproblems.

Sub problem A

For subproblem a, 'bag-of-word' representation (where each feature is binary) was used. The input reviews were read from the csv file and was stored in two vectors. The positive reviews were encoded to numeric 1 and negative reviews were encoded to numeric 0. The reviews were cleaned – converting all reviews to lowercase, removing numbers, stopwords (which occur frequently) etc. These words was then represented in bag-of-word scheme using the CountVectorizer() in scikit-learn library. The dataset was split into two parts – 80% training data and 20% testing data. Multinomial Naïve Bays() was used to train the model.

The accuracy for the classifier obtained was: 0.855

The bag of words representation just creates a set of vectors containing the vocabulary words and their occurrences in the document. This model can be used for simpler tasks since it is easy to understand and interpretable. Additionally, the vectors would also contain many 0s, thereby resulting in a sparse matrix (which hard to deal with in terms of storage as well as computation).

Sub problem B

For subproblem b, TF-IDF based feature vector representation was used. The input reviews were read from the csv file and was stored in two vectors. The positive reviews were encoded to numeric 1 and negative reviews were encoded to numeric 0. The reviews were cleaned – converting all reviews to lowercase, removing numbers, stopwords (which occur frequently) etc. These words was then represented in TFID scheme using the TFIDVectorizer() in scikit-learn library. The dataset was split into two parts – 80% training data and 20% testing data. Multinomial Naïve Bays() was used to train the mode

The accuracy for the classifier obtained was: 0.83

Unlike bag of words, TF-IDF model contains information on the more important words and the less important ones. TF-IDF gives larger values for less frequent words and is high when both IDF and TF values are high (the word is rare in all the documents combined but frequent in a single document). This is because IDF reduces the weight given to common words, and highlights the uncommon words in a document. TF-IDF do not necessarily improve the final classifier's accuracy above plain bag-of-words. The difference between the two is that TF-IDF has the ability to "stretch" the word count as well as "compress" it depending on the importance of the words. Therefore, TF-IDF could altogether eliminate uninformative words.