Name: Netra Inamdar

CU Username: ninamda

ECE 8560- Pattern Recognition

Take-home #2: Results Report

Part 1: Assessing Classification Results from Take-home 1

Case 1 Analysis and Probability of Error:

Case 1 correct test set pattern: 2-3-1-3-1-2

Correctly classified in case 1	13383 out of 15000 samples
Accuracy for case 1	**89.220000 %**
Probability of error for case 1	**10.780000 %**

Confusion matrix for case 1 is:

4768	135	97
373	4444	183
396	433	4171

P(Error) from matrix: [(232+556+829)/15000] * 100 = **10.78 %**

Case 2 Analysis and Probability of Error:

Case 2 correct test set pattern: 2-3-1-1-1-2

Correctly classified in case 2	13774 out of 15000 samples
Accuracy for case 2	**91.826667 %**
Probability of error for case 2	**8.173333 %**

Confusion matrix for case 2 is:

7318	132	50
442	4464	94
250	258	1992

P(Error) from matrix: [(182+536+508)/15000] * 100 = **08.17 %**
(Note: Calculated in the same way for all further P(error) calculations)

Part 2: Separating Hyperplane using Ho-Kashyap Classifier

Ho-Kashyap classifier: The ho-kashyap algorithm is used to get the separating hyperplane for 2 classes, w1 and w2, in training and testing dataset.

Training set (H) accuracy, Hyperplane, and probability of error:

Hyperplane from subsets of H (training set) separating classes w1 & w2 is:

w=[1 x1 x2 x3 x4] * transpose(w12) where w12 is : **[-0.1309 0.0032 -0.0033 0.0004 -0.0047]**

Confusion matrix for training set:

3927	1073
1538	3462

Error Analysis:

Correctly classified in training set	7389 out of 10000 samples
Accuracy for training set	**73.890000 %**
Probability of error for training set	**26.110000 %**

Testing set (S_T) accuracy, Hyperplane, and probability of error:

Confusion matrix for testing set:

3947	1053
1554	3446

Error Analysis:

Correctly classified in testing set	7393 out of 10000 samples
Accuracy for testing set	**73.930000 %**
Probability of error for testing set	**26.070000 %**

Part 3: k-NNR classifier results:

K=1 Confusion matrix and Probability of Error:

In k-NNR,the strategy is to first calculate distance from given vector to all vectors in training set. The vector which is closest is selected and the same class is assigned to the given vector.

Confusion matrix for testing set with k =1:

4640	360
477	4523

Error Analysis for K=1::

Correctly classified in 1-NNR	9163 out of 10000 samples
Accuracy for 1-NNR	**91.630000 %**
Probability of error for 1-NNR	**8.370000 %**

K=3 Confusion matrix and Probability of Error:

For K=3, 3 closest vectors are selected and voting decides the class of given vector.

Confusion matrix for testing set with k =3:

4761	239
497	4503

Error Analysis for K=3::

Correctly classified in 3-NNR	9264 out of 10000 samples
Accuracy for 3-NNR	**92.640000 %**
Probability of error for 3-NNR	**7.360000 %**

K=5 Confusion matrix and Probability of Error:

For K=5, 5 closest vectors are selected and voting decides the class of given vector.

Confusion matrix for testing set with k =5:

4814	186
506	4494

Error Analysis for K=5::

Correctly classified in 3-NNR	9308 out of 10000 samples
Accuracy for 3-NNR	**93.080000 %**
Probability of error for 3-NNR	**6.920000 %**

Part 4: SVM results:

Software Tool Used: libSVM, Reasons and Justification are as below:

libSVM is faster in training and gives better accuracy compared to other tools. It is a very easy to use package with MATLAB and Python interface that makes it preferable over others. The tools folder in libSVM package was helpful in adjusting parameters according to the desired accuracy. Along with these benefits, it is an open source machine learning tool with the code that can be modified. This provides the flexibility to the user.

Implementation of SVM classifier using libSVM is user-friendly and provides all necessary APIs as desired. It lets you experiment with the kernel selection or many such other options and through libsvmread and libsvmwrite, we can convert the data into appropriate format as required. Considering the ease and flexibility along with accuracy and efficiency, I have preferred libSVM with MATLAB interface in this assignment.

Linear Model : Support vector set, Hyperplane parameters & Classification performance:

There are total **4671 support vectors** in this case. Support vectors are in svm_model.SVs file.

Final result after all iterations are completed with linear model:
optimization finished, #iter = 10000000
nu = 0.451265
obj = -4514.357797, rho = -0.879570
nSV = 4671, nBSV = 4448
Total nSV = 4671
Accuracy = 72.26% (7226/10000) (classification)
Model =
Parameters: [5×1 double]
nr_class: 2
totalSV: 4671
rho: -0.8796
Label: [2×1 double]
sv_indices: [4671×1 double]
ProbA: -0.0753
ProbB: 0.0448
nSV: [2×1 double]
sv_coef: [4671×1 double]
SVs: [4671×4 double]

w = [-0.0026 -0.0246 -0.0008 -0.0027]
bias is = 0.8796

RBF Model : Support vector set, Hyperplane parameters & Classification performance:

There are total **8107 support vectors** in this case. Support vectors are in svm_model.SVs file.

optimization finished, #iter = 22895
nu = 0.030168
obj = -4712.620180, rho = 0.345947
nSV = 8107, nBSV = 15
Total nSV = 8107
Accuracy = 89.05% (8905/10000) (classification)
Model=
Parameters: [5×1 double]
nr_class: 2
totalSV: 8107
rho: 0.3459
Label: [2×1 double]
sv_indices: [8107×1 double]
ProbA: -3.3512
ProbB: -0.0724
nSV: [2×1 double]
sv_coef: [8107×1 double]
SVs: [8107×4 double]

w = 1.0e+05 * [0.5488 -0.1999 0.0231 -2.0125]
bias is = -0.3459

Classification Performance with this SVM:

The linear kernel gives accuracy of **72.26 %** and the RBF kernel gives accuracy of **89.05 %**. The hyperplane parameters are given in the previous section.

Comparison of SVM with other models:

Classifier Model	*Accuracy for S_T (Training Set of case 1)*
Bayesian Classifier	89.22
Ho-Kashyap Model	73.93
K-NNR (with K=1)	91.63
SVM Classifier (with RBF model)	89.05

Part 5: Unsupervised (c-means) results

The two distance measures used are: Manhattan distance and Euclidean distance.

Reason for selection and Influence of these distance measures on the solution:

The reason for choosing these two measures is to understand whether any clusters naturally develop, whether we get similar number of vectors in clusters and whether the mean vectors in the two results are close, despite being different when it comes to implementation of these distance measures. The manhattan distance is based on absolute value distance, as opposed to squared error (Euclidean) distance. In practice, we get similar results most of the time. Absolute value distance should gives robust results, whereas Euclidean would be influenced by unusual values. If two points are close on most variables, but more discrepant on one of them, Euclidean distance will exaggerate that discrepancy, whereas Manhattan distance will shrug it off, being more influenced by the closeness of the other variables.

1. Manhattan Distance:

C=2 Analysis:

No. of elements in clusters: c1: 7978, c2: 7024

Mean Vectors:

-10.8616	3.0738	-13.6780	27.6710
100.4396	0.1927	-3.1722	-65.6649

C=3 Analysis:

No. of elements in clusters: c1: 4155, c2: 3362, c3: 7486

Mean Vectors:

144.5115	0.1675	4.4630	-47.8267
25.5444	4.6863	-24.7040	122.8883
-9.0061	1.2595	-8.9375	-60.7600

C=4 Analysis:

No. of elements in clusters: c2: 2780, c4: 3338, c3: 1767, c1: 7119

Mean Vectors:

17.2314	0.1956	-30.3711	-53.9027
24.7891	4.9144	-23.6712	142.2306
-59.4619	4.8398	66.4545	-52.2208
159.4883	0.6823	9.9543	-47.8844

C=5 Analysis:

No. of elements in clusters: c1: 1879, c2: 4154, c3: 4529, c4: 1874, c5: 2569

Mean Vectors:

-96.2881	3.6205	15.7905	-51.1905
23.8902	6.1992	-25.3499	178.5340
33.7962	0.0770	-18.4830	10.6637
35.9459	0.9045	-13.8503	-97.4193
176.2131	1.3075	10.7670	-47.6613

2. Euclidean Distance:

C=2 Analysis:

No. of elements in clusters: c1:11933 c2: 3069

Mean Vectors:

46.6269	0.9482	-4.8241	-54.3118
20.3367	4.7455	-24.0634	132.8603

C=3 Analysis:

No. of elements in clusters: c1: 4200, c2: 2924, c3: 7879

Mean Vectors:

144.0438	0.1393	4.7714	-48.8236
24.9806	4.5195	-24.2922	137.4039
-7.5036	1.5331	-10.2077	-55.4778

C=4 Analysis:

No. of elements in clusters: c1: 7024, c2: 2786, c3: 2156, c4: 3038

Mean Vectors:

33.5901	0.2707	-22.3436	-54.4284
24.6348	4.7744	-23.9502	142.1163
-87.8752	3.1240	22.1276	-52.1861
165.8232	1.2984	14.6683	-46.5970

C=5 Analysis:

No. of elements in clusters: c1: 4238, c2: 1778, c3: 4233, c4: 2773, c5: 1983

Mean Vectors:

35.9573	0.4910	-17.9956	-93.6359
24.5132	6.0611	-24.9577	183.1467
31.0341	0.5463	-21.8174	15.3627
171.7607	1.2439	14.3117	-46.9351
-93.1501	3.6637	21.1247	-52.5025

Compare the results of 1. and 2. above. Do any clusters naturally develop?

As we can see from the results, the mean value for cluster 2 is almost same for all c=2,3,4,5 with both distance measures. (Except for manhattan c=2 part). Thus, we can say that cluster 2 naturally develops. Also, if we compare the mean vectors and number of clusters from euclidean and manhattan distance measures, we can see that the results are quite similar for number of clusters in each class and corresponding mean vectors. For eg, for the 1st distance measure, if the distribution is 2000,3000,4000,7000 then the corresponding mean vectors are close to the mean vectors for 2nd distance measure, even if the order of number of clusters in each class is different in 2nd cluster.

Assess whether the clusters found above are related to the known (estimated) class means:

The known (estimated) class means obtained for training set H from take home-1 are compared to the means obtained in the above 2 methods for c=3 and they are almost similar. Also, the number of vectors in each cluster(for c=3) and their distribution is comparable to the known number of vectors in each class for 3 classes training set. (Close to 5000 in each)

Part 6: Comparison and analysis of Parts 1-5:

1. Bayesian Classifier Analysis: The accuracy of bayesian classifier for both case 1 and case 2 is very good compared to other classifiers. (89.22% and 91.82%) But this requires estimation of parameters and representing data in a closed form is not always possible.

2. Ho- Kashyap algorithm: The probability of error for this algorithm is high (26%) compared to all other algorithms but it is a more practical method as this doesn't require the estimation of parameters or closed form expression as in case of bayesian classifier.

3. In non-parametric case, as seen above, k=5 gives better results (93% accuracy) as compared to K=3 (92% accuracy) and K=1 (91% accuracy) in K-nearest neighbor algorithm analysis. But since it is a brute force method, it is computationally inefficient. In this, as K increases, the probability of error decreases and accuracy increases.

4. SVM Classifier: libSVM gives a robust analysis and all required parameters for classification with a good accuracy. Rbf kernel model has a better performance (89% accuracy) compared to Linear kernel model. The benefit of SVM iis that we can capture much more complex relationships between our data vectors without having to perform difficult transformations on our own. The downside is that the training time is much longer as it's much more computationally intensive.

5. Unsupervised data analysis using c-means: C means clustering is beneficial in unsupervised cases where the class distribution is not known already. In c=3 case, the mean vectors are comparable to the known estimates of c=3 training set (H) and some clusters naturally develop as in cluster 2 for case 1 as shown in above example. It is the most convenient method used for unsupervised data classification that gives a good accuracy and closeness to the groundtruth.