Name: Brian Thibodeau

|  |  |  |
| --- | --- | --- |
| Problem | Points | Score |
| 1(a) | 20 |  |
| 1(b) | 10 |  |
| 1(c) | 10 |  |
| 1(d) | 10 |  |
| 2(a) | 20 |  |
| 2(b) | 10 |  |
| 2(c) | 10 |  |
| 2(d) | 10 |  |
| Total | 100 |  |

Notes:

1. The exam is closed books and notes except for one double-sided sheet of notes.
2. Please indicate clearly your answer to the problem.
3. If I can’t read or follow your solution, it is wrong and no partial credit will be awarded.

***Problem No. 1: Consider 5 data points: (0,1), (-1,0), which belong to class 1, and (1,0), (0, -1), and (-1/2, 1/2), which belong to class 2. In this problem we are going to walk through the K-MEANS clustering process.***

**(a) Assume your initial guesses for two cluster centers are (0,0) and (1/2,1/2). Execute an iteration of K-MEANS by computing the new cluster centers and assigning the data points to the correct cluster. Use averaging to compute the new cluster center.**

 *Solution*-The pseudo code for the K-MEANS clustering algorithm is given below. The algorithm begins with an initial guess of the number of clusters and the means of those clusters.

 **Begin initialize** *n*, *c*, $μ\_{1}$, $μ\_{2}$, … , $μ\_{c}$

 **Do** classify *n* samples according to nearest $μ\_{i}$

 Recompute $μ\_{i}$

 **Until** no changes in $μ\_{i}$

 **Return** $μ\_{1}$, $μ\_{2}$, … , $μ\_{c}$

 **end**

 Each data point is assigned a cluster based of the shortest squared Euclidean distance to each cluster mean. After all data points are assigned a cluster, a new cluster mean is computed using the data points that were assigned to that cluster. This process repeats until the cluster means converge to a constant mean. The algorithm is implemented in Matlab using the built in function *kmeans* and in the following:



**Figure 1. (Left) Plot of data and cluster centers before clustering. (Right) Plot of data and cluster centers after K-MEANS clustering.**

After a single iteration, the K-MEANS clustering algorithm moves the cluster centers to a point such that the squared Euclidean distances from each point to the cluster center are minimized. The new cluster centers are computed to be (-1/2, -1/6) and (1/2, 1/2).

**(b) Assign an identity to each cluster based on a majority-voting scheme and draw the maximum likelihood decision surface.**

*Solution* - After the data has been clustered, each cluster is assigned a class identity using a majority voting scheme. Referring to Figure 2, the cluster with center (-1/2, 1-/6) is identified (or labeled) as class 2 since the number of data points originally pooled from class 2 out numbers the data points originally from class 1 by a 2:1 vote. As result, the cluster with center (1/2, 1/2) is identified as class 1 by default even though the vote is split 1:1. The ML decision surface is determined by first finding the midpoint on a line between the cluster centers and then plotting the perpendicular bisector through that midpoint. The decision surface can then be used for ML classification.



**Figure 2. Class identification using a majority voting scheme after clustering the data**

 **(c) Consider two test data points: (-3/4, 3/4), which belongs to class 1, and (1/2, 1/2), which belongs to class 2. Compute the probability of error based on your K-MEANS clustering.**

*Solution* - Having found the decision surface, the probability of error for the test points (-3/4, 3/4) and (1/2, 1/2) is determined by inspection of Figure 2. The test point (-3/4, 3/4) falls in Class 1’s region and was known to belong to Class 1. Thus the test point (-3/4, 3/4) is classified correctly. Applying the same analysis to the second test point, one finds the test point (1/2, 1/2) is misclassified. Thus one of the two test points are misclassified and the probability of error is 1 / 2 = 0.5

**(d) Compute the probability of error based on a k-nearest neighbor rule. How different should this result be from (c) for large k?**

*Solution* – Using the test points from part (c), the k-nearest neighbor rule classifies a test point by first finding the *k* nearest data points and computing the Euclidean distance to each of the data points. Then After the nearest “neighbors” are identified, the test points are classified according to a majority vote. For the simple case when *k* = 1, the nearest data point to the test point (-3/4, 3/4) is (-1/2, 1/2). Then since the data point (-1/2, 1/2) is known to belong to class 2 and it is the only data point, the test point (-3/4, 3/4) is assigned to class 2. The result is a misclassification of the test point (-3/4, 3/4) since it was known to belong to class 1. Similarly, the test point (-1/2, 1/2) is also misclassified when *k* = 1. The result is a probability of error equal to 0.5.

Tables 1 and 2 on the following page summarize the nearest neighbors and their distances for each of the test points for *k* = 1, 3, 5.

 Table I Table II.

|  |  |  |
| --- | --- | --- |
| TEST POINT: (-0.75, 0.75) |  | TEST POINT: (-0.5, 0.5) |
|
| Nearest Neighbors (k=1) | Distance | Neighbors Class | Nearest Neighbors (k=1) | Distance | Neighbors Class |
| (-0.5, 0.5) | 0.3536 | 2 | (0, 1) | 0.7071 | 1 |
|  |  |
|
| Nearest Neighbors (k=3) | Distance | Neighbors Class | Nearest Neighbors (k=3) | Distance | Neighbors Class |
| (-0.5, 0.5) | 0.3536 | 2 | (0, 1) | 0.7071 | 1 |
| (0, 1) | 0.7906 | 1 | (1, 0) | 0.7071 | 2 |
| (-1, 0) | 0.7906 | 1 | (-0.5, 0.5) | 1 | 2 |
|  |  |  |  |  |  |
| Nearest Neighbors (k=5) | Distance | Neighbors Class | Nearest Neighbors (k=5) | Distance | Neighbors Class |
| (-0.5, 0.5) | 0.3536 | 2 | (0, 1) | 0.7071 | 1 |
| (0, 1) | 0.7906 | 1 | (1, 0) | 0.7071 | 2 |
| (-1, 0) | 0.7906 | 1 | (-0.5, 0.5) | 1 | 2 |
| (1, 0) | 1.9039 | 2 | (-1, 0) | 1.5811 | 1 |
| (0,-1) | 1.9039 | 2 | (0,-1) | 1.5811 | 2 |

For *k* = 3, the test point (-3/4, 3/4) is assigned to class 1 while the test point (1/2, 1/2) is assigned to class 2. The class assignment is determined by a majority vote. Since 2 out of the 3 nearest neighbors to (-3/4, 3/4) belonged to class 1, the test point also gets assigned to class 1. Thus for *k* = 3, each test point is correctly classified and the probability of error is 0/2 = 0.0. Finally, for *k* = 5, the majority vote misclassifies test point (-3/4, 3/4), but correctly classifies test point (1/2, 1/2) resulting in a probability of error equal to 1/2 = 0.5.

Compared to part (c), the error for a k-nearest neighbor rule will approach the Bayes error rate at infinity. For small *k*, the k-nearest neighbor rule is suboptimal and is illustrated by the case when *k* = 1. Figure 3 graphically illustrates the k-nearest neighbor rule. The test points are shown in green and the nearest neighbors to the test points are circled in blue. Note that for *k* = 5, all test points are enclosed since only 5 training points are considered.





**Figure 3. Graphical representation of the k-nearest neighbor rule.**

**Problem No. 2**: Consider the same 5 data points above.

1. **Construct a dendogram for the data.**

A dendrogram for the 5 data points is constructed using Agglomerative hierarchical clustering. The Euclidean distance from each data point to every other point is computed. The results are summarized in a matrix, *D* whose columns represent the data points and whose rows represent the distances between the data points.

$$D=\left[\begin{array}{c}0.000 1.414 1.414 2.000 0.707\\1.414 0.000 2.000 1.414 1.414\\1.414 0.000 2.000 1.414 1.581\\2.000 1.414 1.414 0.000 1.581\\0.707 0.707 1.581 1.581 0.000\end{array}\right]$$

A spanning tree is then formed by stepping through each data point and finding the shortest unused distance to a data point that isn’t in the same cluster. This process repeats until the desired number of clusters is achieved. Figure (4) illustrates the dendrogram for the 5 data points in problem 1 using the distance matrix, *D*.

|  |  |
| --- | --- |
| Index | Data Point |
| 1 | (0,1) |
| 2 | (-1,0) |
| 3 | (1,0) |
| 4 | (0,-1) |
| 5 | (-1/2,1/2)  |



**Figure 4. Dendrogram for the 5 training points in problem 1. Data points are referenced by their indices.**

1. **Construct a top-down clustering (e.g., LBG) clustering (you can also think of this as a crude decision tree).**

*Solution –* Unlike part (a) where the 5 data points were clustered using Agglomerative Hierarchical Clustering, the data can also be clustered using a top-down approach. Top down clustering can be conceptualized by considering it as a continuation of *k*-means clustering. In problem 1(a), it was shown that the 5 data points could be partitioned into two clusters using *k-*means. What’s to say that we don’t continue partitioning each of the new clusters? For instance, referring to figure 1 (right), each new cluster could be further divided into two additional clusters resulting in a total of four clusters. This is like performing *k*-means on each of the clusters resulting from *k*-means.



**Figure 5. Cluster centers after performing a second iteration of *k-*means clustering**

Figure 5 illustrates the results after performing *k-*means clustering on the results from problem 1(a). When a second iteration of *k-*means clustering is performed, each of the orginal two clusters is further divided into two more clusters. The result is 3 of the 5 data points being in there own classs. If a third iteration of *k*-means clustering were performed, the remaining two data points would each become their own cluster.

1. **If you were to use your dendogram to do unsupervised clustering of the data, what clusters would you create (specify them by the mean and the elements associated with the cluster).**

*Solution* – Unlike problem 1 where the data was labeled, unsupervised clustering implies that we are unaware of their labels. However we can use the dendrogram in problem 2(a) along with a similarity measure to two choose which clusters most appropriately model the data.

**(d) Suppose (0, 1) and (1, 0) occur 5 times more often than the rest of the data points. How would you adjust your strategy for clustering the data? How would that impact your decision regions?**