Pattern Classification and Recognition
ECE 681

Spring 2016
Homework #1: ROC Curves and Nearest Neighbors Classification

Due: 4:30 PM, Thursday, February 11, 2016[1]

This homework assignment is worth **220 points**. (220/220 = 97%)
Up to **10 points** extra credit may be earned by going above and beyond the given problem statements (*e.g.*, performing additional analyses, or providing additional insightful interpration of the results).

Your homework is not considered submitted until all three components (hard-copy, Matlab `.m` code, and Blind Test Results Matlab `.mat` file) have been submitted.

Submit a **hard-copy** with your plots and commentary/interpretations to the homework box in Teer.
Submit your **Matlab `.m` code** as an Attachment to the Assignment in Sakai.
Submit your **Blind Test Results** in a Matlab `.mat` file as an Attachment to the Assignment in Sakai.

ROC Curves

Implement your own ROC curve generation function.

(5) 1. Generate an ROC curve for the following distributions of decision statistics:

H_0 decision statistics are 250 samples drawn from a normal (Gaussian) distribution with mean $\mu = 0$ and variance $\sigma^2 = 1$.
H_1 decision statistics are 250 samples drawn from a normal (Gaussian) distribution with mean $\mu = 4$ and variance $\sigma^2 = 4$.

(5) 2. Generate an ROC curve for the following distributions of decision statistics:

H_0 decision statistics are 300 samples drawn from a normal (Gaussian) distribution with mean $\mu = 0$ and variance $\sigma^2 = 1$.
H_1 decision statistics are 200 samples drawn from a normal (Gaussian) distribution with mean $\mu = 2$ and variance $\sigma^2 = 1$.

(10) 3. Comment on whether or not the ROCs you obtained in Questions 1 and 2 make sense, considering the distributions of the decision statistics in both cases.
HINT: Visualize the distributions of the decision statistics. The Matlab function `ksdensity` is a nice function for obtaining estimates of probability density functions given samples drawn from the pdfs.

[1] I (Dr. Tantum) will collect homework from the homework box after 4:30PM, allowing for an appropriate grace period.
DO NOT submit late work to the locked homework box in Teer.
Late work is to be submitted to me (Dr. Tantum) in person, to my mailbox in Hudson 130, or slid under my office door (Teer 310A).
Submitted by 4:30PM, Friday, February 12, 2016 = 1 day late.
Submitted by 4:30PM, Monday, February 15, 2016 = 2 days late.
Submitted by 4:30PM, Tuesday, February 16, 2016 = 3 days late.
Late submissions not accepted after 4:30PM, Tuesday, February 16, 2016.
Work submitted in person to me (Dr. Tantum), to my mailbox in Hudson 130, or slid under my office door (Teer 310A) after the submission deadline but prior to my collecting homework from the homework box will be treated as if it were submitted on time.
Work submitted to the homework box after I have collected homework from the box will receive zero credit.

(10) 4. Generate an ROC curve for the decision statistics and targets provided in the Matlab `.mat` file: `HW01rocBlindTestDecisionStatistics.mat`

(10) 5. Comment on whether or not the ROC for the blind decision statistics obtained in Question 4 makes sense.
HINT: You can "sanity check" this ROC curve (for which you do not know the underlying distributions for the decision statistics) by visualizing the decision statistics. In addition to (or instead of) looking at estimates of the pdfs, you can also look at a plot of the sorted decision statistics coded by class. Assume `ds` contains the sorted decision statistics, and `t` contains the associated target (truth or class) for each decision statistic and is either `0` or `1`. The following lines of code (which make use of *logical indexing*) will produce this plot:

```
figure
plot(find(t==0),ds(t==0),'b.','MarkerSize',8)
hold on
plot(find(t==1),ds(t==1),'ro','MarkerSize',5)
```

(30) 6. Submit the Matlab code that produced the above results for **ROC Curves**, including your implementation of a ROC curve generation function, as an Attachment to the Assignment in Sakai. (We should be able to run this code to replicate your results.)

KNN Classification

Implement your own KNN classifier, with the L_2-norm as the distance metric.

Train your KNN classifier, with the L_2-norm as the distance metric, using the following training data:

H_0 features are 50 samples drawn from a 2-dimensional normal (Gaussian) distribution with mean $\mu = [0\ 0]'$ and covariance matrix $C = \left[\begin{smallmatrix}1 & 0\\ 0 & 1\end{smallmatrix}\right]$ (the identity matrix).
H_1 features are 50 samples drawn from a 2-dimensional normal (Gaussian) distribution with mean $\mu = [2\ 3]'$ and covariance matrix $C = \left[\begin{smallmatrix}2 & 0\\ 0 & 2\end{smallmatrix}\right]$.

(10) 1. Generate ROC curves for the KNN classifier when applied to the above training data for each of the following choices of k: 1, 3, 5, 7, 11, 15, and 19. (You may plot all the ROCs on a single set of axes, provided that you use different line types and include a legend.)

(10) 2. Generate a separate set of testing data that follows the same distributions as the training data. Generate ROC curves for the KNN classifier trained using the original training data then applied to the new (previously unseen) testing data for each of the following choices of k: 1, 3, 5, 7, 11, 15, and 19. (You may plot all the ROCs on a single set of axes, provided that you use different line types and include a legend.)

(20) 3. Comment on whether or not the ROCs you obtained in Questions 1 and 2 make sense, considering both the values of k and whether the KNN is applied to the original training data, as in Question 1, or separate testing data, as in Question 2.

(20) 4. Evaluate performance as a function of the number of nearest neighbors, k, by plotting the area under the ROC curve (AUC) as a function of k for *both* the KNN classifier applied to the training data [ROCs generated in Question 1] *and* the KNN classifier applied to the testing data [ROCs generated in Question 2]. (You may plot AUC vs. k for both cases on a single set of axes, provided that you use different line types and include a legend.)

(20) 5. Comment on whether or not the performance trends you obtained in Question 4 make sense, considering both the values of *k* and whether the KNN is applied to the original training data, as in Question 1, or separate testing data, as in Question 2.

(30) 6. Submit the Matlab code that produced the above results for **KNN Classification**, including your implementation of the KNN classifier, as an Attachment to the Assignment in Sakai. (We should be able to run this code to replicate your results.)

Blind Test

Apply a KNN classifier developed (trained) as in **KNN Classification** to generate decision statistics for blind test data.

(20) 1. Select a value for the parameter *k* based on the performance evaluation completed for Question 4 in **KNN Classification**. (Feel free to consider additional values for *k* beyond those specified in Question 4 in **KNN Classification**, if you think it would be helpful or beneficial.)

Comment on the trade-offs you considered when choosing *k*, and justify your choice of *k*.

(20) 2. Generate decision statistics for the features provided in the Matlab `.mat` file:
`HW01knnBlindTestFeatures.mat`

Save the decision statistics to a Matlab `.mat` file, with the decision statistics stored in the vector `decStat` and saved in the same order as the blind test features. Submit the Matlab `.mat` file containing your decision statistics as an Attachment to the Assignment in Sakai.

(We know the corresponding targets for the blind test data, and will score your decision statistics to generate an ROC curve to evaluate your decision statistics.)