EE 559 **Homework 1 (Week 3)** Posted: Monday, 1/24/2022
Jenkins Due: Tues., 2/1/2022, 11:59 PM PST

Submit by uploading two pdf files to the D2L Dropbox for this assignment (in "Week 3"):

(1) A single pdf file of your answers to the HW problems. Your solutions can be handwritten or typewritten. If handwritten, you can convert to pdf by scanning or using a smartphone camera; if you use a smartphone camera then use a pdf scanner app to convert the pictures to scan-like pdf (e.g., CamScanner).

(2) A single pdf file of your code. The code file must be computer readable (not a scan or a screenshot) – you can check for this by trying to select and copy text in the pdf file.

Always, when submitting files of your work by upload to D2L, after upload it is good practice to check the files that you uploaded to make sure they are the correct files and are complete.

This homework is written **to be solved using python 3 or MATLAB** (your choice), although python 3 is preferred and will be required starting with Homework 3. Resources for help with coding: help files, internet resources, discussions with other students (via piazza or direct communication), learning-python sessions, and TA office hours.

1. In this problem you will code a nearest-means classifier, using Euclidean distance (L_2 norm). You will then use the classifier on data we provide.

   A **nearest-means classifier** is a classifier that assigns any point in feature space to the class that has the nearest mean.

   Coding: Note that in this problem you are to write the code for the classifier and error-rate computation yourself; using a toolbox's or library's function or module that implements this (or similar) classifier is not sufficient. If you use python, this means that you can use built-in functions, numpy, and matplotlib. Use of other libraries or packages (e.g., pandas or scikit-learn) are not in the spirit of this homework problem and will result in points deducted.

   Your code should take as input a set of data points (training dataset) that are labeled according to class, and should calculate a representation of the classifier (class means and a way to use them to classify data points). Once the classifier is "trained" (class means are calculated), your code should be able to calculate the error rate obtained when classifying training data, and calculate the error rate obtained when classifying test data. Error rate is defined as the number of misclassified data points divided by the total number of data points classified, usually expressed as percentage. In addition, with the aid of a supplied plotting function, you will plot the training data points, resulting class means, decision boundaries, and decision regions.

   You will use your implemented nearest-means classifier on 3 different datasets: two synthetic datasets, and one real dataset ("wine" from UCI machine learning repository). The synthetic datasets each have 2 classes ($C = 2$); the wine dataset has 3 classes ($C = 3$).

   To help you, the following files are available for download on the course web site, in "Week 3", along with this homework assignment:

   (i) **Dataset files** (containing labeled data points).

   MATLAB: synthetic1.mat, synthetic2.mat, wine.mat.

For each dataset file, there are in total 4 variables: 'feature_train', 'label_train', 'feature_test' and 'label_test'. While feature_xxx stores the data points (inputs $\underline{x}_n$, $n = 1, 2, \cdots, N$) for training (or testing, respectively), label_xxx are the class labels for these data points (y_n, $n = 1, 2, \cdots, N$). Each row in feature_xxx corresponds to a data point, and each column corresponds to a feature. Class labels are stored as a column vector in label_xxx, where the i^{th} entry corresponds to the i^{th} data point in the feature_xxx file.

Python:

sythetic1_train.csv, synthetic1_test.csv, synthetic2_train.csv, synthetic2_test.csv, wine_train.csv, wine_test.csv.

Each csv file has one row for each data point, and one column for each feature; except the last column contains the class labels (each label is an integer from 1 to *C*, in which *C* is the number of classes).

(ii) **PlotDecBoundaries.m** or **PlotDecBoundaries.py** function helps you plot the decision boundary and regions.

To use PlotDecBoundaries.m (or PlotDecBoundaries.py), you only need to pass in 3 variables, which are training data points of all classes, class labels of all training data points, and sample means for all classes.

For training data points, you need to be consistent with the format of feature_train (or any of the csv files), where a row corresponds to a data point; it should contain no class labels. The class labels variable should also follow the format of label_train (or in python, one column of class labels, same order as rows of "training"). For sample mean, the k^{th} row needs to be set as the sample mean of k^{th} class.

Please answer the following parts. Note that the data is unnormalized; please keep it unnormalized throughout your work on this problem (that is, do not normalize the data yourself in this problem).

(a) For each of the two synthetic datasets, there are in total $C=2$ classes and $D=2$ features. For each synthetic dataset: (i) train the classifier (i.e., calculate the class means), plot the (training-set) data points, the resulting class means, decision boundaries, and decision regions (using PlotDecBoundaries()); also run the trained classifier to classify the data points from their inputs; give the classification error rate on the training set, and separately give the classification error rate on the test set. The test-set data points should never be used for training. Turn in the plots and error rates.

Hint: You might want to check your code first, by trying it out with just two or three training data points in each class, for which you can check the results.

(b) Is there much difference in error rate between the two synthetic datasets? Why or why not?

Parts (c)-(e) below use the "wine" dataset, which is for classifying the cultivar of the grape plant a wine was made from, given measured attributes of the wine. This dataset is briefly described at (in which "attribute" means "feature"):
http://archive.ics.uci.edu/ml/datasets/Wine

For the wine dataset, there are in total $C=3$ classes (grape cultivars) and $D=13$ features (measured attributes of the wine). In this problem (parts (c)-(e) below) you are to use only 2 features for each classification.

(c) Pick the first two features (x_1 = alcohol content, and x_2 = malic acid content), and repeat the procedure of part (a) for this dataset.

(d) Again for the “wine” dataset, find the 2 features among the 13 that achieve the minimum classification error on the training set. (We haven’t yet covered how to do feature selection in class, but will later in the semester. Try coming up with your own method - one that you think will give good results - and see how well it works.

Hint: 13 is not a lot of features, so computation time probably need not be a consideration.

Plot the data points and decision boundaries in 2D for your best performing pair of features, and give its classification error on the training set.

Then give its classification error on the test set. The purpose of the test set is only to estimate the error of the final classifier on unknown (previously unseen) data. Describe the method you used to choose the best pair of features.

Turn in a description of your method for choosing the best pair of features; a plot of the training data, decision boundaries, and decision regions using your chosen pair of features; state which features you chose; and give both error rates (training and testing).

(e) For the wine dataset, is there much difference in training-set error rate for different pairs of features? Justify your answer (e.g., by giving the error rate for a few different example pairs of features; or by giving the standard deviation of error rates over all possible pairs of features).

Also, answer the same question for the test-set error rate.