CSC 461 Programming Languages – Fall 2017
Programming Assignment #1: Pattern Classification (Python)

Introduction

An important branch of machine learning involves pattern classification: recognizing regularities in data. Humans appear to be "hard-wired" for a wide variety of pattern recognition tasks. To classify patterns on the computer, we typically define a feature vector $[x_1,x_2,\ldots,x_n]$ comprised of n measurements. These n measurements define an n-dimensional feature space. If the features are chosen properly, class samples will map to non-overlapping clusters in feature space.

A *minimum distance classifier* is one of the simplest statistical pattern recognition approaches. In the training phase, we partition feature space by computing the centroid of the training samples from each class. To classify an unknown sample, we use a distance formula to find the closest class centroid. The Euclidean distance between two points $\boldsymbol{p}$ and $\boldsymbol{q}$ in n-dimensional space is given by

$$dist(\boldsymbol{p},\boldsymbol{q}) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2}$$

The class of an unknown sample is determined by the centroid of minimum distance from the sample.

Problem

Implement a minimum distance classifier classifier in Python. Input data will be read from a comma-separated value (CSV) file in a specific format, described below. Console output will include classifier accuracy. Individual sample classification results will be written to an output file.

Program usage

```
python mdc.py datafile.csv
```

Input data

Input data includes class and feature information. For example, the popular iris flower data set contains four measurements (features) from three different species (classes) of iris. Measurements include length and width of iris sepals and petals. Based on these four features, it is possible to classify a flower into one of three different iris species with high accuracy.

The input CSV file begins a descriptive header, consisting of the database name followed by class names. The second line contains labels (sample, class, petal length, petal width, etc.). The remainder of the file contains sample data, corresponding to the labels on line 2. Each sample record is comprised of a sample ID, class label, and feature measurements for that data sample. In general, the number of classes, features, and measurements will not be known in advance, and must be determined from reading the input file. For simplicity, you may assume that class names and sample IDs are integers, and measurements are floating point values. Class names will typically be sequential integer values starting at 0, followed by a more descriptive string (e.g., *0=Iris setosa, 1=Iris versicolor, 2=Iris virginica*). Samples are not necessarily listed in any meaningful order. As the name CSV implies, data fields are separated by commas.

The iris dataset CSV file (*iris.csv*) begins like this:

```
Iris database, 0=Iris-setosa, 1=Iris-versicolor, 2=Iris-virginica
sample, class, sepal length, sepal width, petal length, petal width
1, 0, 5.1, 3.5, 1.4, 0.2
2, 0, 4.9, 3, 1.4, 0.2
3, 0, 4.7, 3.2, 1.3, 0.2
. . .
```

Implementation notes

Classification accuracy will generally improve when the sample measurements are normalized. Use the following formula to normalize measurements to the range [0,1]:

$$(value - min) / (max - min)$$

where *min* and *max* are the minimum and maximum values in the data. You should do this individually for each set of measurements (petal length, petal width, etc.).

To determine classifier accuracy, different data sets must be used for training and testing. When the data set is relatively small, a rigorous method for testing classification accuracy is *leave-one-out cross-validation*: select one sample, train on the remaining *n-1* samples, and test only the selected sample for classification accuracy. Repeat until each sample has been tested. The percentage of correctly classified samples is a simple measure of classifier accuracy. Percent accuracy is defined as

$$100 \times (\#\ of\ correct\ classifications) / (\#\ of\ samples)$$

Accuracy should be computed for each class, and for the entire data set.

Program output

1) Print classification parameters, classification accuracy for each class, and overall classification accuracy to the standard output.
2) Print classification parameters, classification accuracy, and individual sample cross-validation results to a file named *datafile.cv* (for an input file named *datafile.csv*), marking incorrectly classified samples with an asterisk.

Sample run

```
% python mdc.py iris.csv

Console output
Iris database
MDC parameters: nclasses = 3, nfeatures = 4, nsamples = 150
class 0 (Iris-setosa): 50 samples, 100.0% accuracy
class 1 (Iris-versicolor): 50 samples, 88.0% accuracy
class 2 (Iris-virginica): 50 samples, 90.0% accuracy
overall:  150 samples,  92.7% accuracy

Contents of iris.cv
Iris database
MDC parameters: nclasses = 3, nfeatures = 4, nsamples = 150
class 0 (Iris-setosa): 50 samples, 100.0% accuracy
class 1 (Iris-versicolor): 50 samples, 88.0% accuracy
class 2 (Iris-virginica): 50 samples, 90.0% accuracy
overall:  150 samples,  92.7% accuracy

Sample,Class,Predicted
1,0,0
2,0,0
. . .
50,0,0
51,1,2 *
52,1,1
. . .
149,2,2
150,2,2
```

Program submission notes

- Complete your program by the due date (Friday September 22) in order to receive credit for this assignment. Late programs will not be accepted for partial credit unless prior arrangements have been made with the instructor.
- To receive full credit, your code must be readable, modular, nicely formatted, and adequately documented, as well as complete and correct. It must build and run successfully using the current Python 3.6 interpreter under Windows and Linux. If your program does not run correctly, indicate why. This will make it easier to give you partial credit.
- Work in teams of two students on this assignment. Your program will be pulled from your GitLab repository on the due date at midnight. Name the project *CSC461_F17_PA1*. Also add your instructor as a developer on the project. This will enable me to monitor your progress and pull your project for grading.

Partners for PA#1

Ian Beamer + Katherine MacMillan
Christopher Blumer + Ryan Hinrichs
Allison Bodvig + Soham Naik
Joey Brown + Lucas Carpenter
Kathleen Brown + Benjamin Garcia
Noah Brubaker + Nathan Ducasse
Aaron Campbell + Sierra Wahlin-Rhoades
Collin Chick + Aaron Gibbs
Jake Davidson + Ryley Sutton
Darla Drenckhahn + Isaac Rath
Jeremy Goens + Tanner Holthus
Naomi Green + Cameron Javaheri
Chad Heath + Andrew Housh
Lawrence Hoffman + Hannah Wegehaupt
Logan Lembke + Garret Odegaard
Kyle Lorenz + Kyle MacMillan
Ryan McCaskell + Mathias Wingert
Michael Pfeifer + Matthew Schallenkamp

When you finish the assignment, email me a brief description of team member contributions (both you and your partner) using the form *Teamwork Evaluation.docx*. Your comments will be kept private, but I may choose to grade team members individually.