CPSC-51100: Statistical Programming

Programming Assignment 3 – Nearest Neighbor Classification

**Introduction**

*Machine learning* is an area of computer science whose aim is to create programs which improve their performance with experience. There are many applications for this, including: face recognition, recommendation systems, defect detection, robot navigation, and game playing. For this assignment, you will implement a simple machine learning algorithm called *Nearest Neighbor* which learns by remembering training examples. It then classifies test examples by choosing the class of the “closest” training example. The notion of “closeness” differs depending on applications. You will need to use the Nearest Neighbor algorithm to learn and classify types of Iris plants based on their sepal and petal length and width. There are three Iris types you will need to classify:

  

Iris Setosa Iris Versicolour Iris Virginica

The learning will be done by remembering training examples stored in a comma-separated file. The training examples include different measurements which collectively are called *features* or *attributes*,and a *class label* for different instances. These are:

1. sepal length in cm

2. sepal width in cm

3. petal length in cm

4. petal width in cm

5. class:

-- Iris Setosa

-- Iris Versicolour

-- Iris Virginica

To see how well the program “learned”, you will then load a file containing testing examples, which will include the same type of information, but for different instances. For each test instance, you will apply the Nearest Neighbor algorithm to classify the instance. This algorithm works by choosing a class label of the “closest” training example, where “closest” means shortest distance. The distance is computed using the following formula:

where are two instances (i.e. a training or a testing example), are their sepal lengths, are their sepal widths, are their petal lengths, and are their petal widths.

After you finish classifying each testing instance, you will then need to compare it to the “true” label that is specified for each example and compute the *accuracy*. Accuracy is measured as the number of correctly classified instances divided by the number of total testing instances.

**Requirements**

You are to create a program in Python 3 that performs the following:

1. Loads and parses the training and testing dataset files into separate NumPy ndarrays. Given what you know, the easiest way to do this is to create four separate arrays:

* 2D array of floats for storing training example attribute values
* 2D array of floats for storing testing example attribute values
* 1D array of strings for storing training example class labels
* 1D array of strings for storing testing example class labels

You can assume there are exactly 4 attribute values in the training and testing examples.

1. Classifies each testing example. You also need to output the true and predicted class label to the screen and save it into a new 1D array of strings. This is done by first computing the distance value for each pair of training and testing examples (their attribute values). Then, for each test example, find the training example with the closest distance. You can do all that easily with NumPy’s vectorized functions - **you shouldn’t use loops for this**.
2. Computes the accuracy. Go through the array of class labels for testing examples and compare the label stored in the array created in step (2). Count how many matches you get. Output the number of matches, divided by the number of testing examples as a percentage.

**Additional Requirements**

1. The name of your source code file should be NearestNeighbor.py. All your code should be within a single file.
2. You cannot import any package except for **NumPy**.
3. Your code should follow good coding practices, including good use of whitespace and use of both inline and block comments.
4. You need to use meaningful identifier names that conform to standard naming conventions.
5. At the top of each file, you need to put in a block comment with the following information: your name, date, course name, semester, and assignment name.
6. The output of your program should **exactly** match the sample program output given at the end.

**What to Turn In**

You will turn in the single NearestNeighbor.py file using BlackBoard.

**References**

The Iris data set is due to

R. A. Fisher (1936). ["The use of multiple measurements in taxonomic problems"](http://digital.library.adelaide.edu.au/coll/special/fisher/138.pdf). [*Annals of Eugenics*](http://en.wikipedia.org/wiki/Annals_of_Eugenics) **7** (2): 179–188. [doi](http://en.wikipedia.org/wiki/Digital_object_identifier):[10.1111/j.1469-1809.1936.tb02137.x](http://dx.doi.org/10.1111%2Fj.1469-1809.1936.tb02137.x).

The pictures of Iris types are due to the following:

"Kosaciec szczecinkowaty Iris setosa". Licensed under CC BY-SA 3.0 via Wikimedia Commons - http://commons.wikimedia.org/wiki/File:Kosaciec\_szczecinkowaty\_Iris\_setosa.jpg#/media/File:Kosaciec\_szczecinkowaty\_Iris\_setosa.jpg

"Iris versicolor 3". Licensed under CC BY-SA 3.0 via Wikimedia Commons - <http://commons.wikimedia.org/wiki/File:Iris_versicolor_3.jpg#/media/File:Iris_versicolor_3.jpg>

"Iris virginica" by Frank Mayfield - originally posted to Flickr as Iris virginica shrevei BLUE FLAG. Licensed under CC BY-SA 2.0 via Wikimedia Commons - http://commons.wikimedia.org/wiki/File:Iris\_virginica.jpg#/media/File:Iris\_virginica.jpg

Sample Program Output

CPSC-51100, [semester] [year]

NAME: [put your name here]

PROGRAMMING ASSIGNMENT #3

#, True, Predicted

1,Iris-setosa,Iris-setosa

2,Iris-setosa,Iris-setosa

3,Iris-setosa,Iris-setosa

4,Iris-setosa,Iris-setosa

5,Iris-setosa,Iris-setosa

6,Iris-setosa,Iris-setosa

7,Iris-setosa,Iris-setosa

8,Iris-setosa,Iris-setosa

9,Iris-setosa,Iris-setosa

10,Iris-setosa,Iris-setosa

11,Iris-setosa,Iris-setosa

12,Iris-setosa,Iris-setosa

13,Iris-setosa,Iris-setosa

14,Iris-setosa,Iris-setosa

15,Iris-setosa,Iris-setosa

16,Iris-setosa,Iris-setosa

17,Iris-setosa,Iris-setosa

18,Iris-setosa,Iris-setosa

19,Iris-setosa,Iris-setosa

20,Iris-setosa,Iris-setosa

21,Iris-setosa,Iris-setosa

22,Iris-setosa,Iris-setosa

23,Iris-setosa,Iris-setosa

24,Iris-setosa,Iris-setosa

25,Iris-setosa,Iris-setosa

26,Iris-versicolor,Iris-versicolor

27,Iris-versicolor,Iris-versicolor

28,Iris-versicolor,Iris-versicolor

29,Iris-versicolor,Iris-versicolor

30,Iris-versicolor,Iris-versicolor

31,Iris-versicolor,Iris-versicolor

32,Iris-versicolor,Iris-versicolor

33,Iris-versicolor,Iris-versicolor

34,Iris-versicolor,Iris-versicolor

35,Iris-versicolor,Iris-versicolor

36,Iris-versicolor,Iris-versicolor

37,Iris-versicolor,Iris-versicolor

38,Iris-versicolor,Iris-versicolor

39,Iris-versicolor,Iris-versicolor

40,Iris-versicolor,Iris-versicolor

41,Iris-versicolor,Iris-versicolor

42,Iris-versicolor,Iris-versicolor

43,Iris-versicolor,Iris-versicolor

44,Iris-versicolor,Iris-versicolor

45,Iris-versicolor,Iris-versicolor

46,Iris-versicolor,Iris-virginica

47,Iris-versicolor,Iris-versicolor

48,Iris-versicolor,Iris-virginica

49,Iris-versicolor,Iris-versicolor

50,Iris-versicolor,Iris-versicolor

51,Iris-virginica,Iris-virginica

52,Iris-virginica,Iris-virginica

53,Iris-virginica,Iris-virginica

54,Iris-virginica,Iris-virginica

55,Iris-virginica,Iris-virginica

56,Iris-virginica,Iris-virginica

57,Iris-virginica,Iris-versicolor

58,Iris-virginica,Iris-virginica

59,Iris-virginica,Iris-virginica

60,Iris-virginica,Iris-virginica

61,Iris-virginica,Iris-virginica

62,Iris-virginica,Iris-virginica

63,Iris-virginica,Iris-virginica

64,Iris-virginica,Iris-virginica

65,Iris-virginica,Iris-virginica

66,Iris-virginica,Iris-virginica

67,Iris-virginica,Iris-virginica

68,Iris-virginica,Iris-virginica

69,Iris-virginica,Iris-virginica

70,Iris-virginica,Iris-versicolor

71,Iris-virginica,Iris-virginica

72,Iris-virginica,Iris-virginica

73,Iris-virginica,Iris-virginica

74,Iris-virginica,Iris-virginica

75,Iris-virginica,Iris-virginica

Accuracy: 94.67%