CSCI 611 - LAB#2 SPECIFICATIONS – version 2

Updated w/correction 2-8-21 (see #15)

OBJECTIVE:

- Familiarize with the Jupyter Notebook environment, Python commands and basic libraries.
- Explore the *MNIST* dataset
- Create a kNN Classifier for *MNIST*

PREPARATION & REFERENCES:

- Complete IMLP (Muller/Guido book) Ch.1, working through ch01-introduction code & notebook.
- Sklearn datasets Sklearn datasets

TASK:

WRITE Python code, using basic libraries (***sklearn, numpy, matplotlib***) to perform the tasks below.
Place the solution to each number below in a unique cell; use comment notation to include the number for reference, and a typed response to any explicit questions (in bold).

Load the Data

1. Using Sklearn.datasets *fetch_openml* loader, load the '*mnist_784*' , version 1 dataset.
   What type of object is returned? _____

Analyze the Data

Provide Python statement(s) to support your answers to the questions below:

2. Display the object's attributes (*keys).* **How many are there? _____**
3. Display a partial description of the dataset, just the first 200 characters.
4. **How many samples are in the dataset? _____**
5. **How many features does each data sample have? _____**
6. **What is the target shape? _____**
7. **What is the difference between the storage structure for the data and that for the target? _____**

Create a custom Function – just for practice…

8. Write a function that calculates an index value based on what alpha characters are passed to it. The function takes in two alpha characters and returns the summation of their numeric placement in the alphabet {'a'=1, 'b'=2,…'z'=26}. Example: *fnct('c','y')* would return 28.
9. Invoke the function with your initials, and display the **resulting index value. ? _____**

Plot your DIGIT from the Data – just for fun!

Each sample represents a single image as 28x28 pixels and each pixel holds a value 0-255 (the pixel's intensity from white to black). That's why there are 784 total features per sample – 1 per pixel.

10. You will first need to grab your digit's feature vector from the data, using your custom index value (#9 above).
11. Next, you will need to reshape it for plotting. Simply invoked the *reshape* method with your feature vector.
12. Now you are ready to plot it using *matplotlib pyplot's imshow* method. You need only the first two arguments: your reshaped array, and colormap 'binary'. **What is your digit? _____**

OK, back to the main task - Let's do some TRAINING

FIRST – separate your dataset into two sets, one for training and one for testing. We won't use the sklearn train_test_split this time, because the MNIST dataset has already been carefully separated into separate training and test sets. More importantly, processing it could take a long time, so we are just going to use a fraction of it.

13. Assign the first 6,000 samples as your **training set** (X_train, y_train), and the last 1,000 samples as your **test set** (X_test, y_test). Remember, X comes from your *data,* and y comes your *targets.*

NEXT – let's build a kNN classifier, with initial **k-value =3** [tip: enable GPUs (*Runtime->Change runtime type)*]

14. What is your test *score* when k is 3? _____
15. Does it improve when you set ~~algorithm~~ to '*distance*'? (yes/no) _____ set **weights** to 'distance'
16. Can you find a k that performs better? _____ If yes, what k? _____

DELIVERABLES

- ***xy***Lab2.py - Python source file, where ***xy*** are your initials; fully documented, including a general header & comments
- ***xy***Lab2.pdf – which SHOWS ALL … code, outputs, embedded comments - generated from a full run of matching .py file
  [***CLEAN:*** Before you print .pdf, be sure all cells and outputs are displayed, close help windows, remove unecessary cells]