NumPy Assignment – Building a KNN Classifier using NumPy

This assignment is a good exercise to practice many of the concepts we covered in the "Numerical Python" section.

1- Define a class named 'KNNClassifier' that accepts "K" (number of neighbors) at initialization time.

Example usage: ***knn = KNNClassifier(5)***

2- Create a method named "**fit**" to accept all input data (X) and labels (y). "X" is a matrix that each row is a sample, and the columns are the features (shape = [num of samples, num of features]). "y" is a vector with the corresponding class numbers (shape = [num of samples,])

Example usage: ***knn.fit(X, y)***

3- Create another method called "predict" that accepts input samples in a matrix (X) and returns the predicted classes in a vector. "X" shape is similar to the "X" in "fit" method.

Example usage: ***predicted_classes = knn.predict(X_new)***

4- Now, write a program to do the following:

a. Load the MNIST data given in the class (mnist.csv)
b. Shuffle the samples
c. Split it to 80% for training and 20% for validation
d. Separate labels from input data for both training and validation data
e. For each K from 1 to 25 (create a loop), instantiate an object from "KNNClassifier" with that number of neighbors.
f. Feed training data using "fit" method
g. Predict the labels for validation data using 'predict' method
h. Compare the predicted labels with the true labels and calculate accuracy:

Accuracy = (Num of correct predictions) / (total number of samples)

i. Plot a line graph that shows the accuracy for each "K"
j. Identify which "K" gives the best result
k. Calculate accuracy of the training data for the same "K"