Lab Assignment 6

In this assignment, you'll flex your understanding of Isomap and KNeighbors, as well as practice splitting your data for testing and evaluation by taking your Module4/**assignment4.py** lab to the next level. If you haven't been able to complete module four's labs or haven't fully understood them, take a moment to re-do them all before proceeding.

This assignment was engineered to be truer to the life of a data scientist by being more challenging than previous ones, so do not be disheartened. If data explorers only needed to drop their observations into black-box algorithms without investing time to toggle parameters, and experiment and understand what those algorithms were truly doing to their data, they wouldn't be valued as much.

In module four's fourth lab assignment, you explored using isomap, an indispensable tool to have while working with non-linear datasets. Your goal this time is to train the KNeighborsClassifier to identify what direction a face is pointing towards: either up, down, left, or right.

This data takes the form of image samples that have been transformed either using PCA to reduce their linear dimensionality, or isomap to non-linearly do similar. Start by reviewing your lab work in the Module4/**assignment4.py** file before opening up the /Module5/**assignment6.py** starter code. You will need access to the **face_data.mat** file from Module four, as well as the new Module5/**face_labels.csv** file.

1. Add in the Module4/assignment4.py code responsible for: loading up the .mat file, properly rotating its images, and storing the whole thing into a Pandas dataframe object.
2. Load into a dataframe your classifications faces_labels.csv file. Make sure your dataframe and your .csv file align properly and start from the same values! This classification dataframe only has a single column in it, so create a series (a slice) that selects only that column and save it as **label**.
3. Do your train_test_split just as directed in the reading. Set random_state=7, and play around with test_size as documented. Your variables should be: `data_train, data_test, label_train,` and `label_test`.
4. Fill out the code for PCA, Isomap, and KNeighborsClassifier. Both PCA and Isomap should be reducing your training data's dimensionality down to 2D. You're free to experiment with different K values for KNeighborsClassifier.
5. Predict the accuracy of the test dataset / test label using `.score()` and print it out.
6. Answer the questions below:

Lab Questions

(1/1 point)
Only one of the following setups is ideal if you plan on using SciKit-Learn's KNeighbors classifier to predict the label of your samples after transforming them. Which is it?

- (●) Fit and transform your data using PCA or Isomap. Split your data. Then fit the KNeighbors model against the training data and labels. Then predict the class of your testing data. ✔

- ( ) Use preprocessing to scale your training and testing data. Split your data. Fit and transform your training data using PCA or Isomap, and fit the KNeighbors model against the training data and labels. Then predict the class of your testing data.

- ( ) Use preprocessing to scale your training and testing data. Split your data. Fit and transform your testing data using PCA or Isomap, and fit the KNeighbors model against the training data and labels. Then predict the class of your testing data.

- ( ) Fit and transform your data using PCA or Isomap. Then fit the KNeighbors model against your data and labels. Then split your data and predict the class of your testing data. ✔

EXPLANATION

Preprocessing is not a requirement, so that has been thrown into the mix and a red herring. The important thing to note here is that you only want to split your data before the steps involved with the actual prediction. Isomap and PCA are used simply to transform your data, e.g. as preprocessing transformations. The same transformation needs to be applied to your testing and training data, and even to future unlabeled samples you encounter in the future.

It doesn't make sense to preduct your testing data after you've trained your model against it, because it would have already seen the 'answers' or labels for it.

Also if you only transform your training data via PCA or Isomap but neglect to do so for your testing data, then you will have two different feature spaces and will not be able to use KNeighbors to predict a class between them.

You have used 1 of 2 submissions