Assignment 3: Dimensionality Reduction & SVMs

Submission Deadline: May 22, 2020 (11:59 PM)

Deliverables: **Project Report** in BMVC format (template can be downloaded from https://www.overleaf.com/project/5e301ad33c2f38000171a776, Ensure you insert your name and Entry Number at the top of your solution) and **submit your code in Python (Solutions in other programming languages will invoke a penalty). Make a zip file containing the code and report, and upload the zip file with your name as title on Google classroom.**

This assignment will involve considerably more work than the previous two, so start early! Also, significantly delayed submissions will not be considered, that is, solutions submitted more than 12 hours after the deadline will receive a ZERO score. This assignment carries 20% weightage over your cumulative FDS score (all scripts will be graded out of 100 and then scaled by a factor of 0.2), so please give this assignment your best shot!

Ensure that you include all your findings in the report, any results that are part of a Python notebook etc., will NOT BE EVALUATED.

You may use any available resource to help you with the assignment, but Plagiarism is an offense and will be strictly dealt with as per university policy.

Q1: Principal Component Analysis and Eigenfaces for Face Recognition

Eigenfaces (https://en.wikipedia.org/wiki/Eigenface) was proposed by Turk and Pentland in the 1980's for face recognition, and represent one of the early breakthroughs in the field of Computer Vision. The 'trick' in this approach involves significantly reducing the dimensionality of a high-dimensional vector (a 320 x 240 pixel image can be viewed as a 76800-dimensional vector) to far fewer dimensions employing Principal Component Analysis, and being able to reconstruct each training image as a linear combination of 'eigen-faces', which resemble ghost faces.

Download data from the "*Labeled Faces in the Wild*" dataset via the following command, so that you obtain 100 faces/individual.

```
lfw_dataset = fetch_lfw_people(min_faces_per_person=100)
```

Download the dataset into train and test sets employing the `train_test_split` command, with `test_size` specified as 0.3.

(1) Perform PCA on the dataset setting the number of PCA components to 100.

(2) Project each face in the training and test datasets onto the 100-dimensional eigenface space. This should transform each face image to a 100-D vector. Choose any three personalities/identities from the train set, and plot the points corresponding to the train+test faces for these identities (appropriately color

coded), after projecting them to 2/3 dimensions via tSNE. Describe your observations regarding the clusters observed.

(3) Use a nearest neighbor classifier, i.e., assign each test face to the identity whose training example is the closest. E.g., if for the current text example, the identity of the closest training example is *George Bush*, the test example will be labeled as *George Bush*. Based on the nearest neighbor classifier, **generate the classification report** for all identities in the training set.

(4) Plot the first 20 eigenfaces,

(5) Instead of hardcoding the number of PCA components, one could also retain as many eigenvectors so that a certain amount of the original data variance is preserved. Determine how many eigenfaces (let us say this number is *x*) are required to retain **80% variance** of the original training set. Repeat the nearest neighbor classification procedure with *x* eigenfaces, and compare the results with respect to (3). Report your observations.

Q2: Dimensionality Reduction and Visualization with PCA, LDA and tSNE

We will attempt to employ the various dimensionality reduction techniques to visualize the the **Fisher Iris** dataset in this section. Fisher Iris comprises 150 4-dimensional data samples arising from three Iris flower species. The four features measured for each data sample are: *sepal length and width, petal length and width*.

(1) Employ PCA to reduce the dimensionality of the Fisher Iris Dataset to 2. Upon projecting each of the 150 data samples onto this 2-dimensional feature space, attempt to explain data variation along the first and 2nd eigen-vectors. This can be done by dividing the range of values along the two eigen-directions into two/three categories (e.g., low/high values), and plotting the data appropriately to make observations such as "*High values along eigen-direction 1 correspond to high values of sepal length and width*". For a hint, see Example 2 in https://towardsdatascience.com/pca-vs-tsne-el-cl%C3%A1sico-9948181a5f87.

(2) LDA adopts a supervised approach to dimensionality-reduction by projecting labelled *C* class data onto a *C*-1 dimensional space. Consider Fisher iris data arising from a pair of classes (you will have 3 Comb 2 = 3 combinations, when considering two classes at a time). For a pair of classes (*C* = 2), LDA will project the data onto a line (1-D feature space).

Plot the projected points and the lines of separation corresponding to the three pairs of classes. Finally, plot the projected 3-class Fisher Iris data on the 2D feature space that maximizes inter-class scatter while minimizing intra-class scatter.

(3) Project the 4-D Fisher Iris data onto (a) 2D and (b) 3D via tSNE. Use at least two different measures for the 'metric' parameter and comment on the observed plots.

Q3: Data Classification with Linear and Non-linear SVMs

Convert the 4D Iris data to 2D by considering either of the following feature pairs: *petal length* and *petal width* or *sepal length* and *sepal width*.

(a) Assuming that the pair of classes can be linearly separated, plot the max-margin hyperplane (i.e, the hyperplane center along with the margin edges) for the three pairs of classes, along with a plot of the two class-data (you may split the dataset in 70:30 for the train and test sets, and employ the training set for SVM model synthesis). Highlight the support vectors in each case. Print the classification report in each case.

(b) The standard SVM classifier function assumes a *C* value of 1. Tabulate the classification report for C values of 0.001 and 1000 (no need for plots for this question). Based on the classification reports obtained for different *C* values, report your observations.

(c) Repeat (a) employing an RBF instead of a linear kernel- plotting the center of the separating hyperplane is sufficient in this case. Tabulate the classification reports for *C* values of 0.001, 1 and 1000 for the RBF SVM and compare the results observed in the three conditions.