Exercise: K-means Clustering and Principal Component Analysis

Overview:

In this exercise, you will implement the K-means clustering algorithm and apply it to compress an image. Then, you will use principal component analysis (PCA) to find a low-dimensional representation of face images.

List of Files for this assignment:

ex7_kmeans.py – main entry script for the first exercise on K-means
ex7_pca.py - main entry script for the second exercise on PCA
ex7data1.mat - Example Dataset for PCA
ex7data2.mat - Example Dataset for K-means
ex7faces.mat - Faces Dataset
bird_small.png - Example Image in PNG format
bird_small.mat - Example Image in MATLAB matrix format
ex7utils.py - contains the following functions related to algorithm progression and plotting:

- `displayData` - Displays 2D data stored in a matrix
- `drawLine` - Draws a line over an existing figure
- `plotDataPoints` - Initialization for K-means centroids
- `plotProgresskMeans` - Plots each step of K-means as it proceeds
- `runkMeans` - Runs the K-means algorithm

ex7modules_kmeans.py - the script that needs to be filled in by you for this assignment. It contains the following functions related to centroids for k-means:

- `findClosestCentroids`: Find closest centroids (used in K-means)
- `computeCentroids`: Compute centroid means (used in K-means)
- `initCentroids`: Initialization for K-means centroids

ex7modules_pca.py - the script that needs to be filled in by you for this assignment. It contains the following functions related to PCA:

- `pca`: Performs principal component analysis
- `projectData`: Projects a data set into a lower dimensional space
- `recoverData`: Recovers the original data from the projection

What you should do:

ex7modules_kmeans.py and ex7modules_pca.py contains functions that are not yet implemented. Your task is to implement those functions by filling in "YOUR CODE HERE" sections. The details can be explained as follows:

[Section 1: K-means Clustering]

In this this exercise, you will implement the K-means algorithm and use it for image compression. You will first start on an example 2D dataset that will help you gain an intuition of how the K-means algorithm works. After that, you wil use the K-means algorithm for image compression by reducing the number of colors that occur in an image to only those that are most common in that image. You will be using ex7_kmeans.py for this part of the exercise.
The K-means algorithm is a method to automatically cluster similar data examples together. Given a training set, you group the data into a few cohesive "clusters". runKmeans.py is the script file that executes this K-means algorithm in an iterative fashion. Functions initCentroids(), findClosestCentroids(), and computeCentroids() are invoked within the script file. initCentroids() guesses the initial centroids, findClosestCentroids() refines this guess by repeatedly assigning examples to their closest centroids, and computeCentroids() re-computes the centroids based on the assignments.
Your task is to complete the functions in ex7modules_kmeans.py. Implement the three functions by filling in "YOUR CODE HERE" sections. After you have finished filling in your code, activate Miniconda, change directory to where your ex7_kmeans.py is located, then type in following command and press Enter:

```
python ex7_kmeans.py
```

During the execution, you will see output text results in console and graphic results in a separate window. If your implementation is correct, the graphic results will be similar to what is shown on the "Sample Results" section of this instructions sheet.

[Section 2: Principal Component Analysis]

In this exercise, you will use principal component analysis (PCA) to perform dimensionality reduction. You will first experiment with an example 2D dataset to get intuition on how PCA works, and then use it on a bigger dataset of 5000 face image dataset. The provided script, ex7_pca.py, will help you step through this section of the exercise.
PCA consists of two computational steps: First, you compute the covariance matrix of the data. Then, you use singular value decomposition to compute the eigenvectors, which will correspond to the principal components of variation in the data.
Before using PCA, it is important to first normalize the data by subtracting the mean value of each feature from the dataset, and scaling each dimension so that they are in the same range. In the provided script ex7_pca.py, this normalization has been performed for you using the featureNormalize() function. After normalizing the data, the ex7_pca.py script will run PCA on the example dataset, and plot the corresponding principal components found. The script will also output the top principal component (eigenvector) found.
After computing the principal components, function `projectData()` reduces the feature dimension of your dataset by projecting each example onto a lower dimensional space, (e.g., projecting the data from 2D to 1D). Specifically, you are given a dataset X, the principal components U, and the desired number of dimensions to reduce to K. You should project each

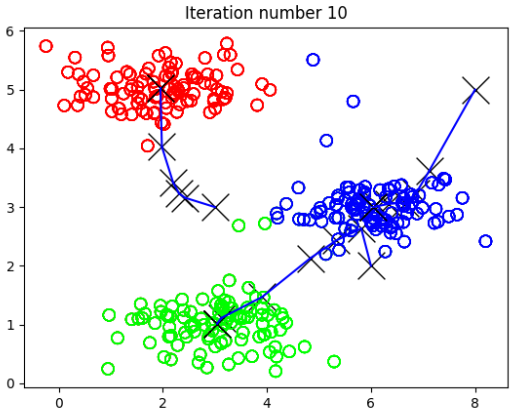
example in X onto the top K components in U. Note that the top K components in U are given by the first K columns of U. After projecting the data onto the lower dimensional space, the function `recoverData()` can approximately recover the data by projecting them back onto the original high dimensional space. It projects each example in Z back onto the original space and returns the recovered approximation in X_rec.
Your task is to complete the functions in ex7modules_pca.py. Implement the three functions by filling in "YOUR CODE HERE" sections. After you have finished filling in your code, activate Miniconda, change directory to where your ex7_pca.py is located, then type in following command and press Enter:

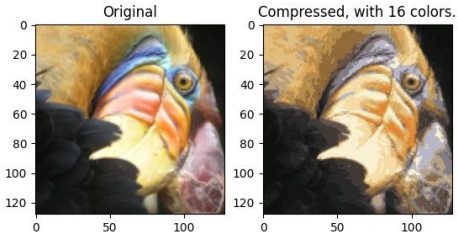
```
python ex7_pca.py
```

During the execution, you will see output text results in console and graphic results in a separate window. If your implementation is correct, the graphic results will be similar to what is shown on the "Sample Results" section of this instructions sheet.

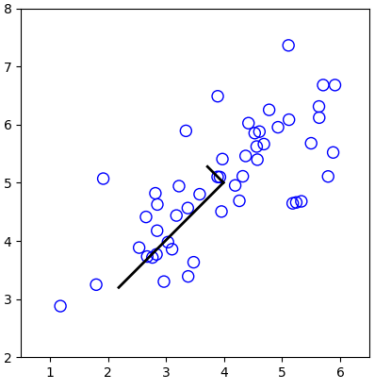
Sample Results:

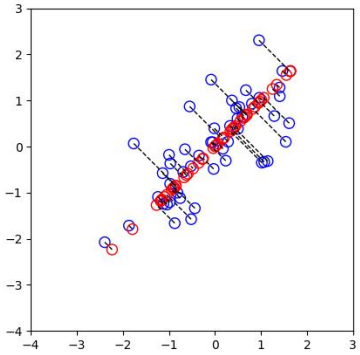
[Section 1: K-means Clustering]





[Section 2: Principal Component Analysis]





Original faces

Recovered faces