ECE 445 (ML for ENGG): Mini Jupyter Exercise #2

Waheed U. Bajwa (waheed.bajwa@rutgers.edu)

Objective: The goal of this exercise is to explore various aspects of *principal components analysis (PCA)* using both synthetic data and real data.

Synthetic Data

In this part of the exercise, we will work with three-dimensional data; this will enable us to visualize data using `matplotlib` and `mpl_toolkits.mplot3d` Python packages.

1. Create a matrix $\mathbf{A} \in \mathbb{R}^{3\times 2}$ whose individual entries are drawn from a Gaussian distribution with mean 0 and variance 1 in an independent and identically distributed (iid) fashion. Once generated, this matrix should not be changed for the rest of this exercise.
   - Matrices with iid Gaussian entries are always *full* rank. Verify this by printing the rank of $\mathbf{A}$; it should be 2.

Generation of Dataset #1

1. Each of our data sample $\mathbf{x} \in \mathbb{R}^3$ is going to be generated in the following fashion: $\mathbf{x} = \mathbf{A}\mathbf{v}$, where $\mathbf{v} \in \mathbb{R}^2$ is a random vector whose entries are iid Gaussian with mean 0 and variance 1. Note that we will have a different $\mathbf{v}$ for each new data sample (i.e., unlike $\mathbf{A}$, it is **not** fixed for each data sample).
   - Generate 500 data samples $\{\mathbf{x}_i\}_{i=1}^{500}$ using the aforementioned mathematical model.
   - Store the data samples into a *data matrix* $\mathbf{X}$, such that each data sample is a column in this data matrix. Print the dimensionality of $\mathbf{X}$ and confirm that it matches your expectations.
   - Since we can write $\mathbf{X} = \mathbf{A}\mathbf{V}$, where $\mathbf{V} \in \mathbb{R}^{2\times 500}$ is a matrix whose columns are the vectors $\mathbf{v}_i$'s corresponding to data samples $\mathbf{x}_i$'s, the rank of $\mathbf{X}$ is 2 *(Can you see why? Perhaps refer to Wikipedia?)*. Verify this by printing the rank of $\mathbf{X}$.

Singular Value and Eigenvalue Decomposition of Dataset #1

You will need to use the `numpy.linalg` package for the purposes of this part of the exercise.

1. Compute the singular value decomposition of $\mathbf{X}$ and the eigenvalue decomposition of $\mathbf{X}\mathbf{X}^T$ and verify (by printing) that:
   - The left singular vectors of $\mathbf{X}$ correspond to the eigenvectors of $\mathbf{X}\mathbf{X}^T$.
   - The eigenvalues of $\mathbf{X}\mathbf{X}^T$ are square of the singular values of $\mathbf{X}$.
   - The *energy* in $\mathbf{X}$, defined by $\|\mathbf{X}\|_F^2$, is equal to the sum of squares of the singular values of $\mathbf{X}$.
2. Since the rank of $\mathbf{X}$ is 2, it means that the entire dataset spans only a two-dimensional space in $\mathbb{R}^3$.
   - Since rank of $\mathbf{X}$ is 2, we should ideally only have two nonzero singular values of $\mathbf{X}$. However, unless you are really lucky, you will see that none of your singular values are exactly zero. Comment on why that might be happening (and if you are the lucky one then run your code again and you will hopefully become unlucky :).
   - What do you think is the relationship between the left singular vectors of $\mathbf{X}$ corresponding to the two largest singular values and the columns of $\mathbf{A}$? Try to be as precise and mathematically rigorous as you can.

PCA of Dataset #1

1. Since each data sample $\mathbf{x}_i$ lies in a three-dimensional space, we can have up to three principal components of this data. However, based on your knowledge of how the data was created (and subsequent discussion above), how many principal components should be enough to capture all variation in the data? Justify your answer as much as you can.

2. While *mean centering* is an important preprocessing step for PCA, we do not necessarily need to carry out mean centering in this problem since the mean vector will have small entries. Indeed, if we let x_1, x_2, and x_3 denote the first, second, and third component of the random vector $\mathbf{x}$ then it follows that $\mathbb{E}[x_k] = 0, k = 1, 2, 3$.

   - Formally show that $\mathbb{E}[x_k] = 0, k = 1, 2, 3$, for this particular problem.
   - Compute the mean vector $\mathbf{m}$ from the data matrix $\mathbf{X}$ and verify by printing that its entries are indeed small.

3. Compute the top two principal components $\mathbf{U} = \begin{bmatrix}\mathbf{u}_1 & \mathbf{u}_2\end{bmatrix}$ of this dataset and print them.

4. Compute feature vectors $\widetilde{\mathbf{x}}_i$ from data samples $\mathbf{x}_i$ by projecting data onto the top two principal components of $\mathbf{X}$.

   - Reconstruct (approximate) the original data samples $\mathbf{x}_i$ from the PCA feature vectors $\widetilde{\mathbf{x}}_i$ by computing $\widehat{\mathbf{x}}_i = \mathbf{U}\widetilde{\mathbf{x}}_i$.
   - Ideally, since the data comes from a two-dimensional subspace, the *representation error*

   $$\sum_{i=1}^{500} \|\widehat{\mathbf{x}}_i - \mathbf{x}\|_2^2 = \|\widehat{\mathbf{X}} - \mathbf{X}\|_F^2$$

   should be zero. Verify (unless, again, you are super lucky) that this is, in fact, not the case. This error, however, is so small that it can be treated as zero for all practical purposes.

5. Now compute feature vectors $\widetilde{\mathbf{x}}_i$ from data samples $\mathbf{x}_i$ by projecting data onto the top principal component of $\mathbf{X}$.

   - Reconstruct (approximate) the original data samples $\mathbf{x}_i$ from the PCA feature vectors $\widetilde{\mathbf{x}}_i$ by computing $\widehat{\mathbf{x}}_i = \mathbf{u}_1\widetilde{\mathbf{x}}_i$.
   - Compute the representation error $\|\widehat{\mathbf{X}} - \mathbf{X}\|_F^2$ and show that this error is equal to the square of the second-largest singular value of $\mathbf{X}$.
   - Using `mpl_toolkits.mplot3d`, display two 3D scatterplots corresponding to the original data samples $\mathbf{x}_i$ and the reconstructed data samples $\widehat{\mathbf{x}}_i$.

Generation of Dataset #2

1. Create a vector $\mathbf{c} \in \mathbb{R}^3$ whose individual entries are drawn from a Gaussian distribution with mean 0 and variance 3 in an independent and identically distributed (iid) fashion. Once generated, this vector should not be changed for the rest of this exercise.

2. Each of our data sample $\mathbf{x} \in \mathbb{R}^3$ is going to be generated in the following fashion: $\mathbf{x} = \mathbf{A}\mathbf{v} + \mathbf{c}$, where $\mathbf{v} \in \mathbb{R}^2$ is a random vector whose entries are iid Gaussian with mean 0 and variance 1. Note that we will have a different $\mathbf{v}$ for each new data sample (i.e., unlike $\mathbf{A}$ and $\mathbf{c}$, it is **not** fixed for each data sample).

   - Generate 500 data samples $\{\mathbf{x}_i\}_{i=1}^{500}$ using the aforementioned mathematical model.
   - Except for the addition of $\mathbf{c}$ to each data sample, dataset #2 is identical to dataset #1 in terms of the generation mechanism. Addition of $\mathbf{c}$, however, shifts our data from the origin, which increases its rank. Verify this by printing the rank of $\mathbf{X}$ *(The curios may want to think about this question: what are the possible choices of* $\mathbf{c}$ *for which the rank of* $\mathbf{X}$ *will not increase?).*

PCA, Centering, and Dataset #2

1. Verify the importance of centering the data as an essential preprocessing step for PCA by carrying out the following steps:
   - Compute the top two principal components $\mathbf{U} = \begin{bmatrix}\mathbf{u}_1 & \mathbf{u}_2\end{bmatrix}$ of dataset #2 *without* centering the data.
   - Reconstruct (approximate) the original data samples $\mathbf{x}_i$ from the PCA vectors by computing $\widehat{\mathbf{x}}_i = \mathbf{U}\mathbf{U}^T\mathbf{x}_i$.
   - Compute the representation error $\|\widehat{\mathbf{X}} - \mathbf{X}\|_F^2$ and show that this error is nowhere close to being zero, despite the fact that $\mathbf{X}$ has only two major directions of variation.
   - Repeat the previous steps by first centering the data matrix $\mathbf{X}$ using the mean vector $\mathbf{m}$ and then showing that the approximated data samples $\widehat{\mathbf{x}}_i = \mathbf{U}\mathbf{U}^T(\mathbf{x}_i - \mathbf{m}) + \mathbf{m}$ once again lead to (almost) zero representation error *(Note: You are **not** allowed to use the original vector* $\mathbf{c}$ *in your calculations).*

Generation of Dataset #3

1. Each of our data sample $\mathbf{x} \in \mathbb{R}^3$ is *initially* going to be generated in the following fashion: $\mathbf{x} = \mathbf{A}\mathbf{v} + \mathbf{n}$, where $\mathbf{v} \in \mathbb{R}^2$ is a random vector whose entries are iid Gaussian with mean 0 and variance 1 and $\mathbf{n} \in \mathbb{R}^3$ is random noise whose entries are iid Gaussian with mean 0 and variance 0.01. Note that we will have a different $\mathbf{v}$ and $\mathbf{n}$ for each new data sample (i.e., unlike $\mathbf{A}$, these are **not** fixed for each data sample).
   - Generate 500 data samples $\{\mathbf{x}_i\}_{i=1}^{500}$ using the aforementioned mathematical model.
   - Transform these data samples into their normalized variants $\{\check{\mathbf{x}}_i\}_{i=1}^{500}$ for better visualization purposes as follows: $\check{\mathbf{x}}_i = \mathbf{x}_i/\|\mathbf{x}_i\|_2$; this effectively confines our data to a unit sphere in $\mathbb{R}^3$ *(Note that if we had not added noise to each data sample then this normalization would have confined our data to a unit circle in* $\mathbb{R}^3$*).*
   - Even though our data is centered (each element of our data has zero expected value; *can you convince yourself of this?*), our data no longer lies in a two-dimensional subspace within $\mathbb{R}^3$ due to the presence of noise. Verify this by printing both the rank of the normalized data matrix $\check{\mathbf{X}}$ and the singular values of $\check{\mathbf{X}}$.

PCA Denoising of Dataset #3

1. Using `mpl_toolkits.mplot3d`, display a 3D scatterplot corresponding to the normalized data samples $\check{\mathbf{x}}_i$.
2. *Denoise* the normalized data samples $\check{\mathbf{x}}_i$ by projecting and reconstructing them using the top two principal components associated with the normalized data matrix $\check{\mathbf{X}}$.
3. Using `mpl_toolkits.mplot3d`, display a 3D scatterplot corresponding to the *denoised* data samples $\check{\mathbf{x}}_i$.
4. Verify that the representation error between the normalized data matrix $\check{\mathbf{X}}$ and its denoised version $\widehat{\mathbf{X}}$ is given by the square of the third singular value of $\check{\mathbf{X}}$.

Real Data

In this part of the exercise, we will work with real-world data that corresponds to images of handwritten digit '0'. In order to load this data into your notebook, you should use the following code:

```
from sklearn.datasets import load_digits
images, labels = load_digits(1, return_X_y=True)
```

Once you run this code, you will have different 8×8 images of digit 0 stored in the `numpy` array `images` as flattened vectors in its rows. Note that you will have to transpose the `images` array to get it in the form where data samples are stored as columns in the array. In this exercise, we will ignore `labels` and treat `images` as our main dataset.

1. Display any one of the images in the dataset as an 8×8 grayscale image using `matplotlib.pyplot`.

2. Get the data ready for PCA by mean centering it; display the mean vector as an 8×8 grayscale image using `matplotlib.pyplot`.

3. Compute the singular value decomposition of the mean centered data $\bar{\mathbf{X}}$.

   - Plot the singular values of $\bar{\mathbf{X}}$ on a log scale (only the vertical axis).
   - Find the *smallest* integer k that satisfies $\frac{\sum_{i=1}^{k} \sigma_i^2}{\|\bar{\mathbf{X}}\|_F^2} \geq 0.90$.

4. Using k computed in the previous part, compute the top k principal components of $\bar{\mathbf{X}}$.

   - Display the first principal component as an 8×8 grayscale image using `matplotlib.pyplot`.

5. Project each mean centered image onto the top three principal components of $\bar{\mathbf{X}}$ to obtain a three-dimensional feature vector corresponding to each image.

   - Using `mpl_toolkits.mplot3d`, display a 3D scatterplot of these three-dimensional feature vectors.
   - Reconstruct the original images from the feature vectors (including accounting for the mean centering). Compute the representation error between the reconstructed images and the original images, and express the relationship between this representation error and the singular values of the mean centered $\bar{\mathbf{X}}$.
   - Display the reconstructed version of the image that you selected in 1.) as an 8×8 grayscale image using `matplotlib.pyplot`.