Calendar

-

[37 unread messages.37](#)
[Inbox](#)

- History
-

Help

Close

P5: Eigenfaces

- Due Mar 12, 2020 by 9:29am
- Points 100
- Submitting a file upload
- File Types zip, py, and png
- Available until Mar 12, 2020 at 9:29am

This assignment was locked Mar 12, 2020 at 9:29am.

Assignment Goals

- Explore Principal Components Analysis (PCA) and the related Python packages
- Make pretty pictures :)

Summary

In this project you'll be implementing a Python version of the facial analysis program we demonstrated in lecture, using Principal Components Analysis (PCA).

An image and an image projected onto our principal components

Professor Zhu's Matlab rendering of an image and its PCA projection

We'll walk you through the process step-by-step (at a high level).

Dataset

You'll be using the same dataset from lecture, which is available here: YaleB_32x32.matLinks to an external site.
Download YaleB_32x32.mat

Program Specification

Implement these **five (5)** Python functions to do PCA on our provided dataset, in a file called **pca.py**:

1. **load_and_center_dataset(filename)** — load the dataset from a provided .mat file, re-center it around the origin and **return** it as a NumPy array of floats
2. **get_covariance(dataset)** — calculate and **return** the covariance matrix of the dataset as a NumPy matrix (d x d array)
3. **get_eig(S, m)** — perform eigen decomposition on the covariance matrix S and **return** a diagonal matrix (NumPy array) with the largest m eigenvalues on the diagonal, *and* a matrix (NumPy array) with the corresponding eigenvectors as columns
4. **project_image(image, U)** — project each image into your m-dimensional space and **return** the new representation as a d x 1 NumPy array
5. **display_image(orig, proj)** — use matplotlib to display a visual representation of the original image and the projected image side-by-side

Load and Center the Dataset

First, download our sample dataset to the machine you're working on: YaleB_32x32.matLinks to an external site.
Download YaleB_32x32.mat

Once you have it, you'll want to use the SciPy.io function `loadmat()` to load the file into Python. (You may need to install SciPy (Links to an external site.). Get NumPy while you're at it.)

```
>>> from scipy.io import loadmat
>>> dataset = loadmat('YaleB_32x32.mat')
```

From this data, you'll want to grab the `'fea'` array. This is the array of face image representations. I'll also define a couple of related variables for you here (n, the number of images we're analyzing, and d, the dimension of those images).

```
>>> x = dataset['fea']
>>> n = len(x)
>>> d = len(x[0])
```

This should give you an *nxd* dataset, where each **row** is an image feature vector of 1024 pixels. (Note this configuration for future calculations.)

As mentioned in lecture, your next step is to re-center this dataset around the origin. You can take advantage of the fact that `x` (as defined above) is a NumPy array and as such, has this convenient behavior:

```
>>> x = np.array([[1,2,5],[3,4,7]])
>>> np.mean(x, axis=0)
=> array([2., 3., 6.])
>>> x - np.mean(x, axis=0)
=> array([[-1., -1., -1.],
          [ 1.,  1.,  1.]])
```

(You will probably need to change the data type of the feature array before you center it, since it reads in as a matrix of uint8 and the mean values are calculated as float64. You can use NumPy's `array()` constructor to change the type of an array.)

After you've implemented this function, it should work like this:

```
>>> x = load_and_center_dataset('YaleB_32x32.mat')
>>> len(x)
=> 2414
>>> len(x[0])
=> 1024
>>> np.average(x)
=> -8.315174931741023e-17
```

(Its center isn't *exactly* zero, but taking into account precision errors over 2414 arrays of 1024 floats, it's what we call Close Enough.)

Find Covariance Matrix

As given in the notes, the covariance matrix is defined as

$S = \frac{1}{n-1} \sum_i x_i x_i^\top$

To calculate this, you'll need a couple of tools from NumPy again:

```
>>> x = np.array([[1,2,5],[3,4,7]])
>>> np.transpose(x)
=> array([[1, 3],
          [2, 4],
          [5, 7]])
>>> np.dot(x, np.transpose(x))
=> array([[30, 46],
          [46, 74]])
>>> np.dot(np.transpose(x), x)
=> array([[10, 14, 26],
          [14, 20, 38],
          [26, 38, 74]])
```

The result of this function for our sample dataset should be a 1024x1024 matrix.

Get m Largest Eigenvalues/vectors

You'll never have to do eigen decomposition by hand again (Links to an external site.)! Use `scipy.linalg.eigh()` to help, particularly the optional `eigvals` argument.

We want the *largest m eigenvalues* of S.

Return the eigenvalues as a diagonal matrix, in descending order, and the corresponding eigenvectors as columns in a matrix.

To return more than one thing from a function in Python:

```
def multi_return():
    return "a string", 5
mystring, myint = multi_return()
```

Make sure to return the diagonal matrix of eigenvalues FIRST, then the eigenvectors in corresponding columns. You may have to rearrange the output of eigh() to get the eigenvalues in decreasing order -- and *make sure to keep the eigenvectors in the corresponding columns* after that rearrangement.

```
>>> Lambda, U = get_eig(S, 2)
>>> print(Lambda)
[[1369142.41612494       0.        ]
 [      0.         1341168.50476773]]
>>> print(U)
[[-0.01304065 -0.0432441 ]
 [-0.01177219 -0.04342345]
 [-0.00905278 -0.04095089]
 ...
 [ 0.00148631  0.03622013]
 [ 0.00205216  0.0348093 ]
 [ 0.00305951  0.03330786]]
```

Project the Images

Given one of the images from your dataset and the results of your `get_eig()` function, create and return the *projection* of that image.

For any image $\mathbf{x}_i$, we project it into the M dimensional space as $\hat{\mathbf{x}}_i = \sum_{j=1}^{d} \alpha_{ij} \mathbf{u}_j$ where $\alpha_{ij} = \mathbf{u}_j^\top \mathbf{x}_i$. (The $\mathbf{u}_j$ are the eigenvector columns of U from the previous function.)

Find the alphas for your image, then use them together with the eigenvectors to create your projection.

```
>>> projection = project_image(x[0], U)
>>> print(projection)
[6.84122225 4.83901287 1.41736694 ... 8.75796534 7.45916035 5.4548656 ]
```

Visualize

We'll be using matplotlib's imshow (Links to an external site.). First, make sure you have the matplotlib library (Links to an external site.), for creating graphs.

Example output image showing correct formatting

Follow these steps to visualize your images:

1. Reshape the images to be 32x32 (you should have calculated them as 1d vectors of 1024 numbers).
2. Create a figure with one row of two subplots (Links to an external site.).
3. The first subplot (on the left) should be titled (Links to an external site.) "Original", and the second (on the right) should be titled "Projection".
4. Use `imshow()` with optional argument `aspect='equal'` to render the original image in the first subplot and the projection in the second subplot.
5. Use the return value of `imshow()` to create a colorbar (Links to an external site.) for each image.
6. Render (Links to an external site.) your plots!

Submission Notes

Please submit BOTH your Python code (**pca.py**) AND a sample image, including a projection of an image that is DIFFERENT from the sample image in this writeup.

Be sure to **remove all debugging output** before submission; your functions should run silently (except for the image rendering window).

Rubric

Title: PCA

Keep in mind that 184 students have already been assessed using this rubric. Changing it will affect their evaluations.

PCA

PCA

Criteria	**Ratings**		**Pts**
Edit criterion descriptionLinks to an external site. Delete criterion rowLinks to an external site. This criterion is linked to a Learning Outcome load_and_center_dataset() returns correctly formatted output Range ☐ threshold: pts	Edit ratingLinks to an external site. Delete ratingLinks to an external site. 10 pts Full Marks Links to an external site.	Edit ratingLinks to an external site. Delete ratingLinks to an external site. 0 pts No Marks	10 pts 10 pts -- Additional Comments
Edit criterion descriptionLinks to an external site. Delete criterion rowLinks to an external site. This criterion is linked to a Learning Outcome load_and_center_dataset() returns correctly centered output Range ☐ threshold: pts	Edit ratingLinks to an external site. Delete ratingLinks to an external site. 10 pts Full Marks Links to an external site.	Edit ratingLinks to an external site. Delete ratingLinks to an external site. 0 pts No Marks	10 pts 10 pts -- Additional Comments

Criteria	Ratings		Pts
Edit criterion descriptionLinks to an external site. Delete criterion rowLinks to an external site. This criterion is linked to a Learning Outcome get_covariance() returns correct result for input datasets Range ☐ threshold: pts	Edit ratingLinks to an external site. Delete ratingLinks to an external site. 10 pts Full Marks Links to an external site.	Edit ratingLinks to an external site. Delete ratingLinks to an external site. 0 pts No Marks	10 pts 10 pts -- Additional Comments
Edit criterion descriptionLinks to an external site. Delete criterion rowLinks to an external site. This criterion is linked to a Learning Outcome get_eig() returns both diagonal eigenvalue matrix and eigenvectors in columns Range ☐ threshold: pts	Edit ratingLinks to an external site. Delete ratingLinks to an external site. 5 pts Full Marks Links to an external site.	Edit ratingLinks to an external site. Delete ratingLinks to an external site. 0 pts No Marks	5 pts 5 pts -- Additional Comments
Edit criterion descriptionLinks to an external site. Delete criterion rowLinks to an external site. This criterion is linked to a Learning Outcome get_eig() returns correct eigenvalue matrix Range ☐ threshold: pts	Edit ratingLinks to an external site. Delete ratingLinks to an external site. 10 pts Full Marks Links to an external site.	Edit ratingLinks to an external site. Delete ratingLinks to an external site. 0 pts No Marks	10 pts 10 pts -- Additional Comments
Edit criterion descriptionLinks to an external site. Delete criterion rowLinks to an external site. This criterion is linked to a Learning Outcome get_eig() returns correct eigenvector matrix Range ☐ threshold: pts	Edit ratingLinks to an external site. Delete ratingLinks to an external site. 10 pts Full Marks Links to an external site.	Edit ratingLinks to an external site. Delete ratingLinks to an external site. 0 pts No Marks	10 pts 10 pts -- Additional Comments
Edit criterion descriptionLinks to an external site. Delete criterion rowLinks to an external site. This criterion is linked to a Learning Outcome project_image() correctly projects image vector onto eigenvectors Range ☐ threshold: pts	Edit ratingLinks to an external site. Delete ratingLinks to an external site. 10 pts Full Marks Links to an external site.	Edit ratingLinks to an external site. Delete ratingLinks to an external site. 0 pts No Marks	10 pts 10 pts -- Additional Comments
Edit criterion descriptionLinks to an external site. Delete criterion rowLinks to an external site. This criterion is linked to a Learning Outcome display_image() creates correctly formatted plot (titles/subplots/colorbars appear as specified) Range ☐ threshold: pts	Edit ratingLinks to an external site. Delete ratingLinks to an external site. 10 pts Full Marks Links to an external site.	Edit ratingLinks to an external site. Delete ratingLinks to an external site. 0 pts No Marks	10 pts 10 pts -- Additional Comments

Criteria	Ratings		Pts
Edit criterion descriptionLinks to an external site. Delete criterion rowLinks to an external site. This criterion is linked to a Learning Outcome display_image() displays images correctly for given input Range ☐ threshold: pts	Edit ratingLinks to an external site. Delete ratingLinks to an external site. 10 pts Full Marks Links to an external site.	Edit ratingLinks to an external site. Delete ratingLinks to an external site. 0 pts No Marks	10 pts 10 pts -- Additional Comments
Edit criterion descriptionLinks to an external site. Delete criterion rowLinks to an external site. This criterion is linked to a Learning Outcome Submitted sample image includes both original image and projected image as required Range ☐ threshold: pts	Edit ratingLinks to an external site. Delete ratingLinks to an external site. 10 pts Full Marks Links to an external site.	Edit ratingLinks to an external site. Delete ratingLinks to an external site. 0 pts No Marks	10 pts 10 pts -- Additional Comments
Edit criterion descriptionLinks to an external site. Delete criterion rowLinks to an external site. This criterion is linked to a Learning Outcome Submitted sample image differs from images provided in writeup Range ☐ threshold: pts	Edit ratingLinks to an external site. Delete ratingLinks to an external site. 5 pts Full Marks Links to an external site.	Edit ratingLinks to an external site. Delete ratingLinks to an external site. 0 pts No Marks	5 pts 5 pts -- Additional Comments

Total Points: 100 out of 100

Submission

Submitted!
Mar 11, 2020 at 8:44pm
Submission Details
Download p5_vpham.zip
Download Figure_1.png
Download Figure_2.png
Download pca.py
Grade: 100 (100 pts possible)
Graded Anonymously: no
View Rubric Evaluation

Comments:

No Comments