Microsoft: DAT210x Programming with Python for Data Science

Bookmarks

- Start Here
- 1. The Big Picture
- 2. Data And Features
- 3. Exploring Data
- 4. Transforming Data

Lecture: Transformations

Lecture: PCA
Quiz

Lab: PCA
Lab

Lecture: Isomap
Quiz

Lab: Isomap
Lab

Lecture: Data Cleansing
Quiz

4. Transforming Data > Lab: PCA > Assignment 1

Bookmark

Welcome to Module 4's PCA Labs!

In order to complete the PCA labs in this module, please make sure you download and unarchive this .zip file with all the datasets and files necessary.

Lab Assignment 1

In this assignment, you're going to experiment with a real life armadillo sculpture scanned using a Cyberware 3030 MS 3D scanner at Stanford University. The sculpture is available as part of their 3D Scanning Repository, and is a *very dense* 3D mesh consisting of 172974 vertices! The mesh is available for you, located at /Module4/Datasets/**stanford_armadillo.ply**. It is *not* a Python file, so don't attempt to load it with a text editor!

Dive Deeper

▸ 5. Data Modeling

Open up the Module4/**assignment1.py** starter code and read through it carefully. You will notice the use of a new library, Plyfile. This library loads up the 3D binary mesh for you. The mesh is further converted into a Pandas dataframe for your ease of manipulation. Complete the following tasks:

1. Before changing any of the code, go ahead and execute assignment1.py. You should see the 3D armadillo. Your goal is to reduce its dimensionality from three to two using PCA to cast a shadow of the data onto its two most important principal components. Then render the resulting 2D scatter plot.
2. Fill out the proper code in the do_PCA() and do_RandomizedPCA() methods. Be sure to **return** the

result of your transformation! You may even want to read the SciKit-Learn documentation on `.transform()`, just for future reference so you know what data type comes out of it.

3. Re-run the application! Then, answer the questions below:

Lab Questions

(3/3 points)
What direction was the armadillo's head facing?

● Left ✓

○ Up

○ Right

○ Down

Were you able to discern any **visual** differences between the transformed PCA results and the transformed RandomizedPCA results?

○ Yes, the RandomizedPCA version was no longer even recognizable as an armadillo

○ Yes, the RandomizedPCA version was a lot less true to the original than the regular PCA version

○ Yes, but it wasn't a lot... just minor differences

● No, they pretty much looked the same to me ✓

Which executed faster, RandomizedPCA or PCA?

○ PCA

● RandomizedPCA ✓

EXPLANATION

Start this lab by filling in the PCA code, just as you saw in the reading. The code to get RandomizedPCA working is very similar. Take a look at the SciKit-Learn docs for RandomizedPCA and you should see that only one or two alterations are necessary to get that running.

To see the execution times, look at the title of your plots. Or alternatively, print them out to the terminal. RandomizedPCA uses some approximation techniques to speed up the execution of PCA, but it's still very accurate. You shouldn't be able to notice any visible changes.

You have used 1 of 2 submissions