# Homework 3: Principal Component Analysis and FastMap

**Yash Naik & Na Li**

Tasks: We both worked on the PCA and FastMap algorithm individually and then compared our results to reach an agreement.

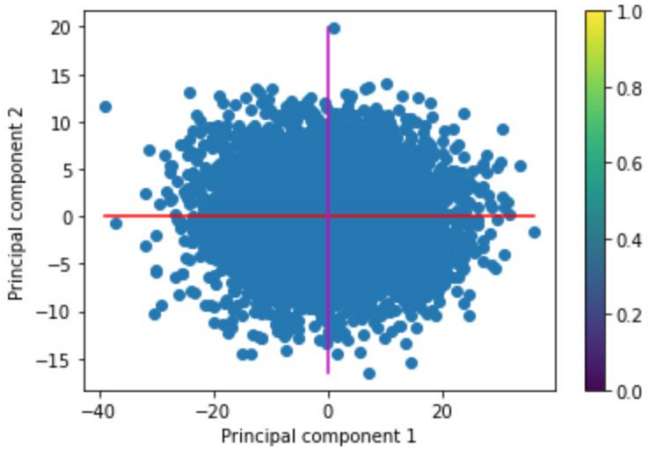
## Part 1: Implementation

**PCA**

| | 0 | 1 | 2 |
|---|---|---|---|
| **0** | 5.906263 | -7.729465 | 9.144945 |
| **1** | -8.640323 | 1.724260 | -10.696805 |
| **2** | 0.258541 | 0.230622 | 0.767439 |
| **3** | -5.234354 | 3.194685 | -1.894385 |
| **4** | 12.622863 | -3.507888 | 4.086258 |
| **5** | 0.785567 | 3.007478 | 0.001893 |
| **6** | -13.845237 | 6.070108 | -11.577550 |
| **7** | 6.917136 | -0.206895 | 4.910577 |
| **8** | 5.836017 | -3.178313 | 3.303392 |
| **9** | -10.478944 | 5.925388 | -13.357354 |

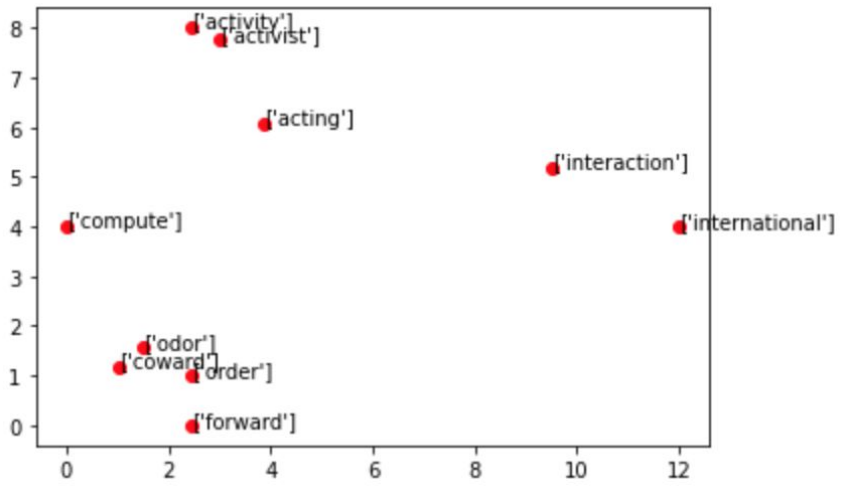
For this part of the assignment we had to read a file containing several 3D points in the Euclidean space. I used a pandas dataframe as the data structure to read the data as it makes data handling much easier, whereas Na Li used the built-in python file object to read the data from the file.

Implementing Principal Component Analysis or PCA to reduce the dimensions of our data from 3D to 2D was fairly easy and straightforward. The only little trouble we faced was matching our results since we both got different results in the first attempt. Also, finding the directions of the Principal components was a little ambiguous. The Red line identifies Principal component 1 and the Magenta symbolizes PC2.



**FastMap**

For this part of the assignment, our task was to implement fast searching from among 6000 points to map 10 words into points in 2-d space, using k feature-extraction functions. Implementing the fastmap algorithm was quite challenging as choosing the distance between the two farthest objects can occur in non-linear time i.e. O(N^2). The underlying purpose of this algorithm is also to reduce dimensionality of the dataset.
We used numpy arrays as our data structure to perform FastMap.




- 

## Part 2: Software Familiarization

Reducing dimensionality of a dataset has been made much handy and user friendly in python by implementing the PCA() function from the sci-kit learn library. Using this library function can get the job done in just one line of code.
However, for the FastMap algorithm, I could not find any library function in python that can solve the given problem at hand.

## Part 3: Applications

Principal component analysis is a crucial step in unsupervised learning and comes to aid when data is booned with the curse of dimensionality. PCA comes extremely handy when we deal with highly complex and high dimensional data. This technique reduces the dimensions of our data such that only the features with most correlations are dropped to avoid redundancy and thus, even with reduced dimensions most of the data integrity is preserved.
Some of the applications of PCA include, but are not limited to, Quantitative finance, image compression, facial recognition, medical data correlation etc.