INF552 Homework 3

Group members and contribution :
Krishna Akhil Maddali (Code development and report)
Yash Shahapurkar (Code development and research)
Myungjin Lee (Code development and research)

Part 1: Implementation

- Output after running the program

  - **PCA**

    The first two principal components corresponding to the two largest eigenvalues.

```
First component :       [ 0.86667137 -0.23276482  0.44124968]
Second component :  [-0.4962773  -0.4924792   0.71496368]
```

  - **Fastmap**

    The objects are embedded in the two-dimensional space with coordinates

X	Y
3.875	6.0625
3	7.75
0	4
1.0417	1.1875
2.4584	0
9.5	5.1875
2.4584	8
1.5	1.5625
2.4584	1
12	4

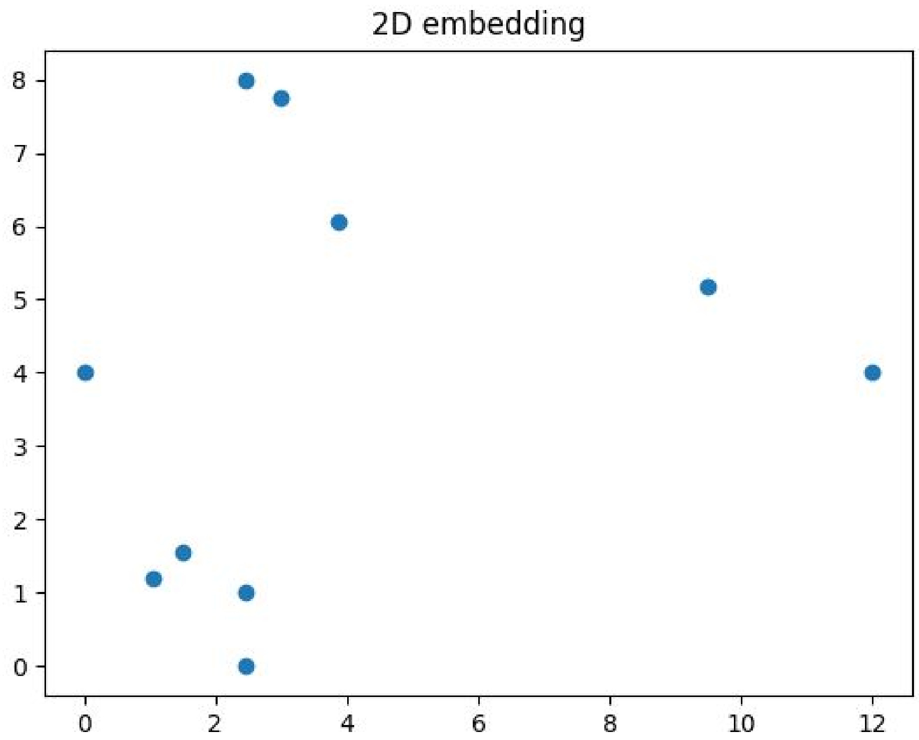
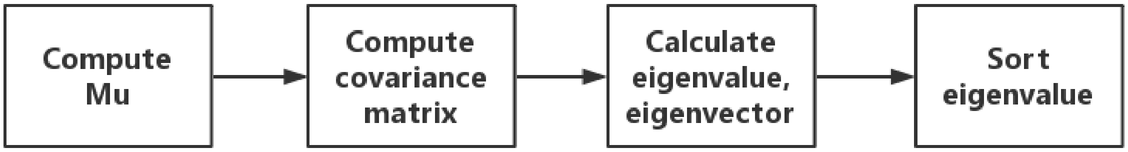


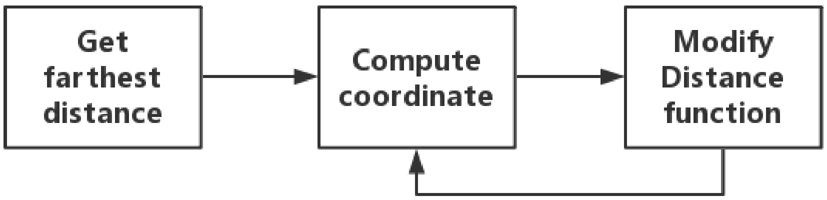
Figure - Fastmap results

- <u>Data structures</u>

  - PCA



  - FastMap



To store objects and the distances, we first store everything in lists and then convert it to an array for operations.
All objects in the columns and the corresponding distances are stored in arrays.
To find the coordinates, we first create an empty list and for each coordinate obtained, we append it to the list.
Next we calculate the new distance and store it an array.
This array is passed to the function to get the second coordinates of the point.
This coordinate is also stored in a list.

Code-level optimization

- PCA

  Since numpy package was slow to calculate covariance matrix as well as dealing with the data set, we have used dataframe from pandas package.

- FastMap

  We used a simple reflection trick, which avoided the need of adding duplicate entries for dist(j,i) when given dist(i,j) in the dataset.
  We kept all distances in float to avoid explicit type-casting issues.

Challenges

- PCA

- **Speed and construction of the data structure** : When we first read data with numpy array, the speed of the code was way too slow, so that we changed using numpy to dataframe by pandas.
- **Familiarization of using pandas** : The usage of dataframe by pandas is not very well familiarized which is different from numpy array, we spent some time for getting used to the functions such as subtracting the values, getting covariance matrix.

- FastMap

- The given dataset has ‘\t’ tabs and are stored as strings. Hence, converting them to the required data type was a little challenging.
- Since, we are given the symmetric distance eg, distance between 3 & 6 is given in the data. We know it is the same as distance between 6 & 3, but this entry is not given in the data. So we have to add such an inverted case to fetch distance, invariable of object order.

Part 2: Software Familiarization

- PCA

	Our code	Python PCA
Output	First component : [0.86667137 -0.23276482 0.44124968] Second component : [-0.4962773 -0.4924792 0.71496368]	First component : [-0.86667137 0.23276482 -0.44124968] Second component : [-0.4962773 -0.4924792 0.71496368]
Library	Our own code	Python Scikit (sklearn)
comparison	The output from our code matches exactly the same as the one from python sklearn library except the sign of the vector. However, the sign would not matter as long as the magnitudes of the each direction are the same because nature of component is vector.	

- Fastmap

Unfortunately, we could not find any specific library that implements Fastmap algorithm, and it is noteworthy to make one.

- How to improve our code

- PCA

  There is a possibility to improve the speed of the code since the calculation time of the sklearn library is shorter than that of our code. The implementation of the code can be improved by using pandas package for sorting the eigenvector.

- Fastmap

  The current code has the number of objects defined, An improved code could find the number of objects by itself. This can be done by using combination approach. Given 10 objects, we know there should be 45 entries as ${}^{10}C_2$ is 45. We could go the reverse way to find the number of objects.

  In this question, we have been given the distances. However, finding the farthest distances will take n^2 order of time. Using a heuristic, we can find the pair of farthest points in linear time.

Part 3: Applications

- **PCA**
- **Quantitative finance :** PCA can be applied to the risk management of interest rate derivatives portfolios. Trading multiple swap instruments which are usually a function of 30-500 other market quotable swap instruments is sought to be reduced to usually 3 or 4 principal components, representing the path of interest rates on a macro basis.
- **Neuroscience :** PCA can be also applied to identify a neuron from the shape of its action potential. In spike sorting, PCA is available to reduce the dimensionality of the space of action potential waveforms, and then performs clustering analysis to associate specific action potentials with individual neurons.

- **FastMap**
- **Finding similarity in images:** There are a lot of attributes which can define similarity of an image, other than just pixel values. The attributes can include color shades, presence of objects, texture, edges etc. However, clustering those high dimensional feature vectors cannot be visualized. The high dimensional features can be embedded into 2D or 3D features using fastmap which maintain

the distance in the original space. Thus, we can get a plot of the 2D representation of those images. Finally, if we run a clustering algorithm on them, we can find which images are similar to others based on the cluster assigned.

- **Association of words :** Words are another type of instance where the distance between words cannot be defined. Hence, metrics like Damerau– Levenshtein need to be used to define a distance between them. This necessarily cannot be represented in space. In order to define them in space, we can map these to a 2D space using fast map. Hence, once in a 2D space, we can visualize distance between the objects and find if they are similar or no. If the objects are close enough they can be similar. If not then they are different. This is the way how we can associate words with others. Eg A country can be associated to its capital, or a masculine word can be associated to a feminine word.

References

[1] - [Faloutsos and Lin, 1995] Christos Faloutsos and King-Ip Lin. Fastmap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In Proceedings of the ACM SIGMOD International Conference on Management of Data, 1995

[2] - The Fast Map Algorithm for Shortest Path Computations
Liron Cohen Tansel Uras Shiva Jahangiri Aliyah Arunasalam Sven Koenig T.K. Satish Kumar

[3] - Alpaydin, 2010] Ethem Alpaydin. Introduction to Machine Learning. The MIT Press, 2nd edition, 2010