REPORT

Group Members and Contributions:

Naureen Firdous: Code for FastMap(50%) , report part 1
Nikitha Krishna Vemulapalli: Code for FastMap(50%) , Code for PCA using library, report part 2
Vijayantika Inkulla: Code for PCA , report part 3

PART1:

Data Structures and Libraries used in PCA

Numpy :We have used numpy arrays and the mathematical functions in numpy.

- linalg.eig : to calculate the Eigen values and vectors of the covariance matrix.
- mean() : to calculate the mean of the data.
- cov() : to calculate the covariance matrix of the data set.
- loadfromtxt(): to get a numpy array from the given text file.

Data Structures and Libraries used in FastMap:

Numpy :We have used numpy arrays and the mathematical functions in numpy.

- random.randint(): to get a random index for the initial centroids.
- genfromtxt(): to get a numpy array from the given csv file.
- loadfromtxt(): to get a numpy array from the given text file.
- argmax() : gives the index of the maximum valued element in the array.

Code level optimizations:

1. In FastMap, we have used a distance matrix to store the distances between the words, example: distance between word1 and word2 is same as distance between word2 and word1. This will ensure that the complexity doesn't become quadratic while recomputing the distances.
2. Since global variables decrease the performance, we have only used local variables.
3. We have used external libraries like numpy to calculate mean, covariance matrix, Eigen values and vectors.

Challenges :

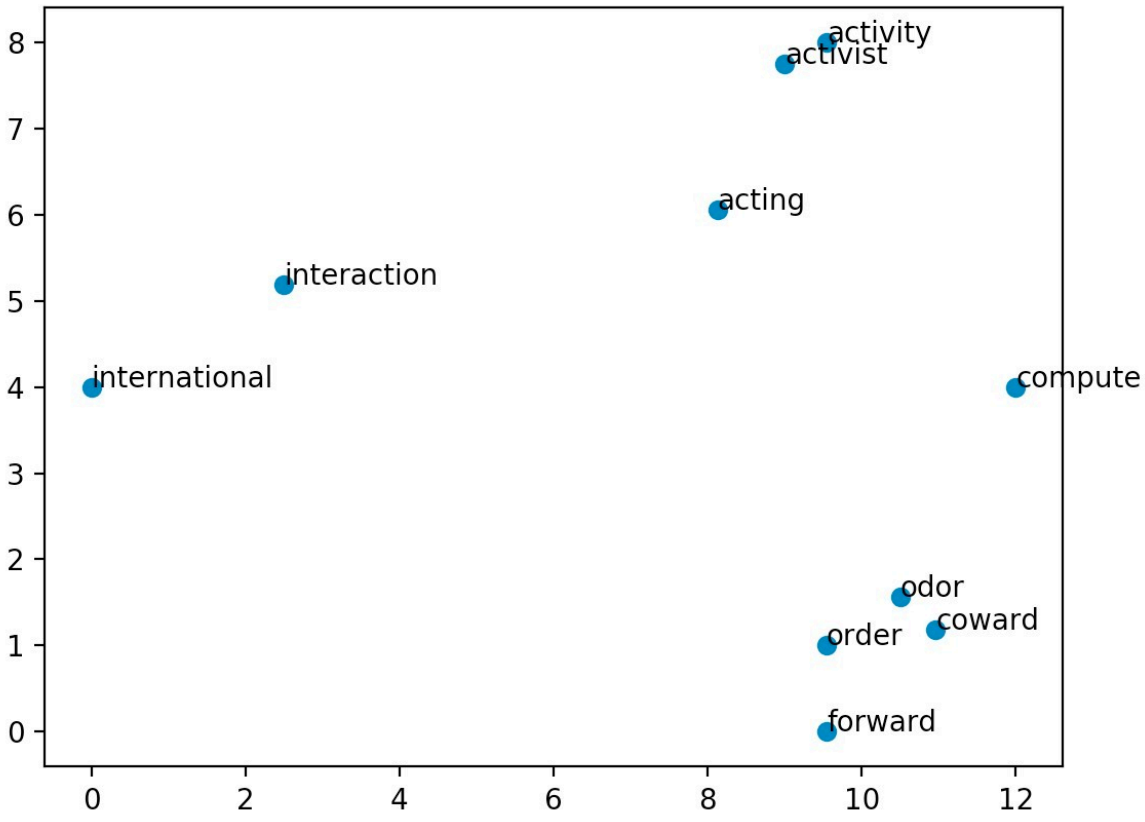
1. In FastMap, careful precautions were taken so that the entire list of words was not traversed to calculate every coordinate (different dimensions) of every object.
2. Eigen values and Eigen vectors were calculated using the function linalg.eig() in numpy. Calculating these values for the entire data set would have been very challenging.

Output:

PCA:

[[0.86667137 -0.23276482 0.44124968]
 [-0.4962773 -0.4924792   0.71496368]]

FastMap :



PART2:

PCA Library functions:

PCA can be implemented using the PCA class in decomposition collection of scikit-learn. It takes the number of dimensions to which data has to be reduced a the input.

The principal components of the data can be obtained using the 'components_' attribute.

FastMap Library functions:

FastMap doesn't have any library functions.

Improvements:

1. The run-time can be improved by optimizing the code.
2. Singular value decomposition can be added, similar to the PCA class of scikit learn.
3. In FastMap, the calculation of farthest points can be terminated when two successive iterations yield same pair of points.

PART3:

Some real life applications of PCA and FastMap are :

PCA :

1. **Image compression:** Image compression with PCA is one of the most frequently used applications of the dimension reduction technique. An image is nothing but a matrix of pixels represented by RGB color values. Thus, principal component analysis can be used to reduce the dimensions of the matrix (image) and project those new dimensions to reform the image that retains its qualities but is smaller in k-weight.

2. **Stock PortFolio Management:** Stocks can be managed by using PCA. The major risk factors can be identified as the principal components and data can be represented in the form of these components.

3. Genomic analysis :

DNA is usually represented using combinations of letters A,D,G,T. FastMap can be used to identify similarities between these DNA strings. Since they are objects and not points, FastMap can be used to embed these strings as points.