Programming Assignment 3: PCA and Fast Map

Srihari Akash Chinam
USC ID - 2953497706
Net ID - chinam

Sayan Nanda
USC ID - 2681592859
Net ID - snanda

Implementation

- Language used - Python version 2
- Libraries used - Numpy (arrays and matrices), matplotlib.pyplot, math, SciKit-Learn

Principal Component Analysis

The input file provided, “pca-data.txt”, consists of 6000 3D points with their co-ordinates provided. These points are loaded into a Numpy array and the eigenvalues and eigenvectors are calculated for each dimension using the inbuilt function of Numpy, linalg.eig. The 2 highest eigenvalues are then chosen, who’s vectors represent the new dimension directions, seen in Table 1.

Table 1 Direction vectors of the 2 chosen dimensions by Assignment Implementation

		Eigen Vector	
Direction 1	0.86667137	-0.23276482	0.44124968
Direction 2	-0.4962773	-0.4924792	0.71496368

Upon using the PCA implementation of SciKit-Learn on the same dataset, the Eigen vectors obtained are seen in Table 2. If observed carefully, direction 2 of both vectors is exactly the same, but in Direction 1, it is seen that the constants are exactly the same, but in the opposite direction. This implies that when mapped onto a 2D space, the points generated by these two algorithms will be mirror images.

Table 2 Direction vectors of the 2 chosen dimensions by SciKit-Learn Implementation

		Eigen Vector	
Direction 1	-0.86667137	0.23276482	-0.44124968
Direction 2	-0.4962773	-0.4924792	0.71496368

FastMap

In the input file “fastmap-data.txt”, the distance matrix is provided for 10 points. Using the FastMap algorithm, the 2D coordinates were generated. The pivot points obtained for this implementation are shown in Table 3. The final coordinates of the 10 points are shown in Table 4.

Table 3 Pivot Points obtained in the implementation of FastMap

	Pivot Points	
Direction 1	2	9
Direction 2	4	6

Table 4 Coordinates of the 10 points obtained with FastMap

	Coordinate 1	Coordinate 2
Point 0	3.875	6.0625
Point 1	3.0	7.75
Point 2	0.0	4.0
Point 3	1.04166667	1.1875
Point 4	2.45833333	0.0
Point 5	9.5	5.1875
Point 6	2.45833333	8.0
Point 7	1.5	1.5625
Point 8	2.45833333	1.0
Point 9	12.0	4.0

If observed in Table 4, we see that points 2 and 9 have the same second coordinate and points 4 and 6 have the same first coordinate as they represent a direction together. After coordinates were generated for the FastMap data, they were combined with the words from the file "fastmap-wordlist.txt" and mapped in the 2D space to obtain the output shown in Figure 1.

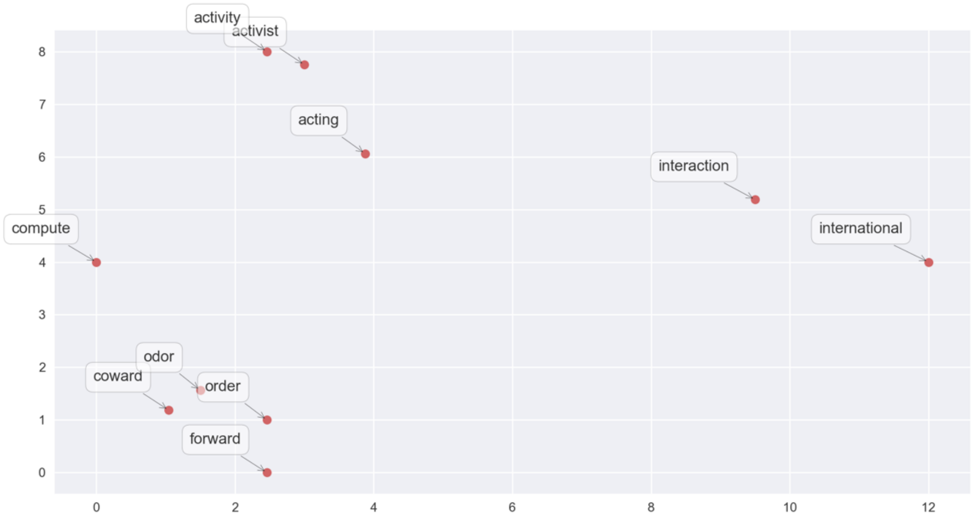

Figure 1 Mapping of words with their coordinates generated by FastMap

After this, using the 6000 points provided for PCA, a distance matrix of the dimensions 6000x6000 was generated, which was given as input to FastMap for reduction. The output obtained for this is compared with the outputs from the assignment implementation of PCA and the SciKit-Learn implementation in Figure 2. This was done to check the correctness of the implementation of the FastMap algorithm.

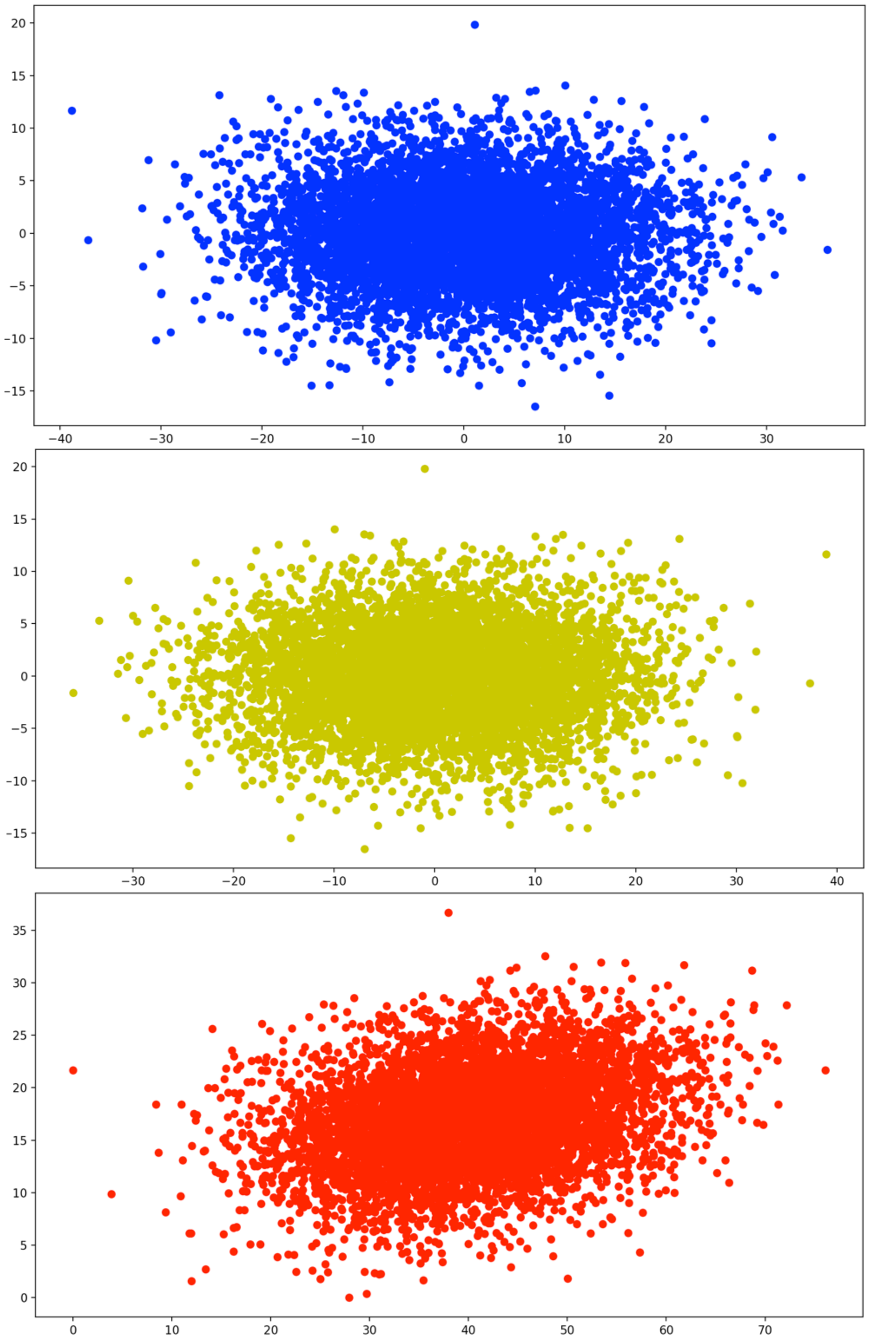

Figure 2 Assignment output in blue, SciKit-Learn output in yellow and FastMap output in red

When observed carefully, it is seen that SciKit-Learn's output (in yellow) is an exact mirror image of the assignment's output (in blue), while FastMap's output (in red) is rotated slightly in the anti-clockwise direction compared to the assignment's output.

Software Familiarization

The python library, SciKit-Learn offers an implementation for PCA. In this implementation there is an option to 'whiten'. This multiplies the square root component vectors to the square root of n-samples which is subsequently divided by the singular values. This can improve the accuracy of the prediction but at the same time it sacrifices some information. There are different versions of the Singular Vector Decomposition solver. Our implementation is similar to the ‘full’. For other implementation, there may be parameters introduced to increase the flexibility. These include tolerance, iterated power and a random state. There is also an option to guess the number of components. This is determined by the amount of variance.

Some of the improvements from the SciKit implementation are giving the option to whiten the data. Improvements such as experimenting with different SVD solvers are not viable since we are moving from a 3D data set to a 2D data set. The randomized SVD solver is useful when the number of components is moving from a much higher number to very low. For example, 4096 to 16 (in some face recognition data sets).

Applications

Principal Component Analysis is used in risk assessment [1]. Features up to 500 in number is reduced to 3-4 principal component. These are used to represent paths of interest rates. In neuroscience, PCA is used in a technique called spike-triggered covariance analysis [2]. It is also used to identify of a neuron from shape of its action potential. In another application called spike sorting [3], clustering analysis is performed after PCA to associate specific action potentials with individual neurons.

Other common progressions of PCA are in the exploratory phase of data analysis, preprocessing of data. Its used in computer graphics, meteorology and oceanography. An interesting application of PCA is in the monitoring and disturbance detection of a hydrotreating process [4]. It was found to be useful in monitoring a set of 38 variables and diagnosed significant disturbances and their causes. PCA is also used in chemometrics [5]. Chemometrics is the use of mathematical and statistical methods to understand chemical information. PCA is considered a basic building block in this.

Fastmap finds its application in many academic areas of study. Such as searching in high dimensional spaces, high-dimensional indexing, multi-dimensional scaling, searching in metric spaces. This gives it applications in fields such as visual analysis [6], video segment retrieval [7].

Individual Contributions

Sayan Nanda

- Implementation of models
    - FastMap
    - Visualization of word list
- Documentation
    - Software Familiarization
    - Applications of PCA and FastMap

Srihari Akash Chinam

- Implementation of models
  - PCA
  - SciKit-Learn PCA
  - Visualization of PCA and FastMap

- Documentation
  - Implementation of PCA and FastMap
  - Comparison of performances of assignment, SciKit-Learn and FastMap implementations
  - Applications of PCA

References

[1] Sun, Yuebing, et al. "Spatial, sources and risk assessment of heavy metal contamination of urban soils in typical regions of Shenyang, China." Journal of hazardous materials 174.1 (2010): 455-462.

[2] Pillow, Jonathan W., and Eero P. Simoncelli. "Dimensionality reduction in neural models: an information-theoretic generalization of spike-triggered average and covariance analysis." Journal of vision 6.4 (2006): 9-9.

[3] Adamos, Dimitrios A., Efstratios K. Kosmidis, and George Theophilidis. "Performance evaluation of PCA-based spike sorting algorithms." Computer methods and programs in biomedicine 91.3 (2008): 232-244.

[4] http://pubs.acs.org/doi/abs/10.1021/ie0714605

[5] https://www.intechopen.com/books/analytical-chemistry/pca-the-basic-building-block-of-chemometrics

[6] Huang, Zhexue, David W. Cheung, and Michael K. Ng. "An empirical study on the visual cluster validation method with fastmap." Database Systems for Advanced Applications, 2001. Proceedings. Seventh International Conference on. IEEE, 2001.

[7] Malheiros, Viviane, et al. "A visual text mining approach for systematic reviews." Empirical Software Engineering and Measurement, 2007. ESEM 2007. First International Symposium on. IEEE, 2007.