CAIRO UNIVERSITY

FACULTY OF ENGINEERING

HEMN454 – Data Mining and Machine Learning in Healthcare

Task 2 – Clustering and Dimensionality Reduction

Amira Mahmoud Farid 1170498

Clustering & Dimensionality Reduction

Code Description

1. Data reading, cleaning, and adjustments
   1.1. Read Data
   1.2. Drop description and references rows from the data
   1.3. Rename all data
   1.4. Split the columns of data
   1.5. Renaming the columns
   1.6. Convert data into numeric values
2. PCA Function
   2.1. **Subtract the mean of each variable**
   Subtract the mean of each variable from the dataset so that the dataset should be centered on the origin. Doing this proves to be very helpful when calculating the covariance matrix.
   2.2. **Calculate the Covariance Matrix**
   Calculate the Covariance Matrix of the mean-centered data. You can know more about the covariance matrix in this really informative Wikipedia article here.
   The covariance matrix is a square matrix denoting the covariance of the elements with each other. The covariance of an element with itself is nothing but just its Variance.
   That's why the diagonal elements of a covariance matrix are just the variance of the elements.
   2.3. **Compute the Eigenvalues and Eigenvectors**
   Compute the Eigenvalues and Eigenvectors for the calculated Covariance matrix. The Eigenvectors of the Covariance matrix we get are Orthogonal to each other and each vector represents a principal axis
   A Higher Eigenvalue corresponds to a higher variability. Hence the principal axis with the higher Eigenvalue will be an axis capturing higher variability in the data.
   Orthogonal means the vectors are mutually perpendicular to each other. Eigenvalues and vectors seem to be very scary until we get the idea and concepts behind it.
   2.4. **Sort Eigenvalues in descending order**
   Sort the Eigenvalues in the descending order along with their corresponding Eigenvector. Remember each column in the Eigen vector-matrix corresponds to a principal component, so arranging them in descending order of their Eigenvalue will automatically arrange the principal component in descending order of their variability.
   Hence the first column in our rearranged Eigen vector-matrix will be a principal component that captures the highest variability.
   2.5. **Select a subset from the rearranged Eigenvalue matrix**
   Select a subset from the rearranged Eigenvalue matrix as per our need i.e. number_comp = 2. This means we selected the first two principal components.
   2.6. **Transform the data**
   Finally, transform the data by having a dot product between the Transpose of the Eigenvector subset "eigenvector_subset transpose" and the Transpose of the mean-centered data "x_mean transpose". By transposing the outcome of the dot product, the result we get is the data reduced to lower dimensions from higher dimensions.

2.7. **Call PCA function and print results**

Principal component analysis or PCA in short is famously known as a dimensionality reduction technique.

3. K – medoids

3.1. **Algorithm**

3.1.1. Initialize: select k random points out of the n data points as the medoids.

3.1.2. Associate each data point to the closest medoid by using any common distance metric methods.

3.1.3. While the cost decreases:

For each medoid m, for each data o point which is not a medoid:

3.1.3.1. Swap m and o, associate each data point to the closest medoid, recompute the cost.

3.1.3.2. If the total cost is more than that in the previous step, undo the swap.

3.2. **Steps**

3.2.1. Let the randomly selected 2 medoids

3.2.2. Calculating cost

The dissimilarity of each non-medoid point with the medoids is calculated and tabulated

Each point is assigned to the cluster of that medoid whose dissimilarity is less

3.2.3. Randomly select one non-medoid point and recalculate the cost

The dissimilarity of each non-medoid point with the medoids C1 and C2 is calculated and tabulated.

Each point is assigned to that cluster whose dissimilarity is less

Swap Cost = New Cost – Previous Cost

As the swap cost is not less than zero, we undo the swap

3.3. **The time complexity is O(k * (n-k)^2).**

References

https://www.askpython.com/python/examples/principal-component-analysis#:~:text=Principal%20Component%20Analysis%20from%20Scratch%20in%20Python%20Principal,and%20statistics.%20PCA%20is%20an%20unsupervised%20statistical%20method.

https://github.com/letiantian/kmedoids

https://towardsdatascience.com/k-medoids-clustering-on-iris-data-set-1931bf781e05

https://www.geeksforgeeks.org/ml-k-medoids-clustering-with-example/