Unsupervised Learning: K-means & PCA

Yi Lin

03/07/2021

1 Introduction

Clustering is an unsupervised machine learning algorithm, which identifies patterns without labels and clusters data based on the features. In this project, I will examine whether a clustering algorithm (k-means) could determine the main clusters between different patients in the genetic fingerprints of all patients from the hospital without the labels. And I will identify which patient has a similar type of genetic composition as a specific patient (patient Z).

1.1 K-means

K-means clustering is the method of vector quantization, which aims to partition n observations into k clusters in which each observations belongs to the cluster with the closest mean [1]. In other words, It stands for the 'winner takes it all' principle by assigning several centroids based on the number of clusters given, and each data point is appointed to the cluster which centroid the nearest (most similar) to it. The purpose of K-means clustering is to minimize the squared Euclidean distance between the feature and the centroid of the cluster to which it belongs.

1.2 Principal Component Analysis (PCA)

PCA is the technique of reducing the dimension of feature space while maintaining most of its variance, which is called 'dimensionality reduction.' Although there are several methods to achieve dimensionality reduction, both feature elimination, and feature extraction are the main two classes. Based on Wikipedia explains, '*PCA is defined as an orthogonal linear transformation that transforms the data to a new coordinate system such that the greatest variance by some scalar projection of the data comes to lie on the first coordinate (called the first principal component), the second greatest variance on the second coordinate, and so on.*' [2]

In our project, genetic_fingerprint was renamed as 'patient_all' including 14398 rows $\times$ 386 columns, which contains all patients' genetic fingerprint; 'patient_zet' renamed as 'patient_Z' including 386 rows $\times$ 1 column, which contains the patient Z genetic fingerprint. I also transposed 'patinet_Z' to 1 row x 386 columns.

2 Identify the optimal number of cluster

I was first manually choosing the number of centroids when k = 2, k = 3, and k = 4, but it is not pretty sure to determine how segregated of clusters. Our purpose is to try to make the cluster as distinctive as possible since each cluster has its similar set of data points, which supports gathering more helpful information from our dataset.

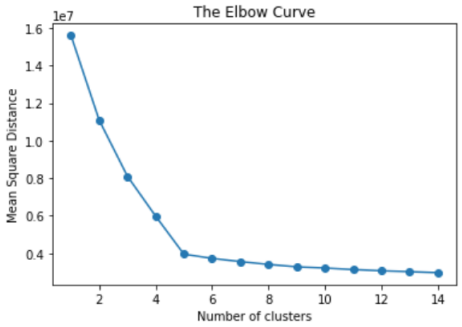
2.1 Elbow Method

The Elbow method is to find the optimal number of clusters in the K-means algorithm. The Elbow method is based on the value Within Cluster Sum of Squares (WCSS) for each solution to determine the number of clusters. WCSS is the same as the shared distance between each number of clusters and its centroid. In other words, the elbow curve basically draws a trade-off between the knowledge gain and the number of

centroids. Since there is no general rule on clustering solution, which depends on the dataset, I first test the elbow method with 15 clusters. And then plot the WCSS with respect to the number of components on a graph (Figure 1).

The elbow curve clearly shows that if we choose the number of centroids as 5, then we have the optimal opportunity of fine clustering. There is a steep decrease before the elbow while it becomes smoother after the elbow. This curve gives an insightful view in deciding between information gain and computational expense from 'patient_all.'

Figure 1: The Elbow Curve

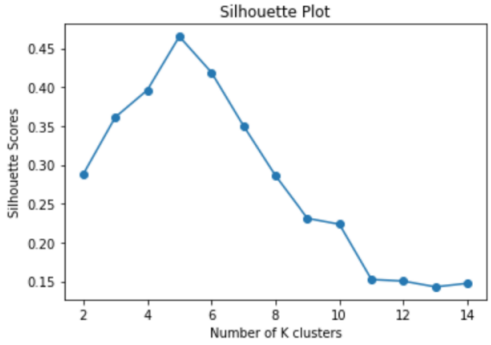


2.2 Silhouette Score

Silhouette score is another good measurement to determine the number of clusters to be formed from the dataset. It measures how similar an instance is to its own cluster (cohesion) compared to other clusters (separation) and ranges from -1 to $+1$. If most instances have a high value, then the clustering configuration is appropriate. If many points have negative values, then there are too many or too few clusters [3].

The Silhouette method clearly shows that k=5 should be chosen for the number of clusters since its silhouette score is the highest one. Similar to the Elbow method, the Silhouette plot has steeply increased before the optima while it has a dramatic decline after k=5. This plot also provides a meaningful and clear vision to determine the optimal number of clusters (Figure 2).

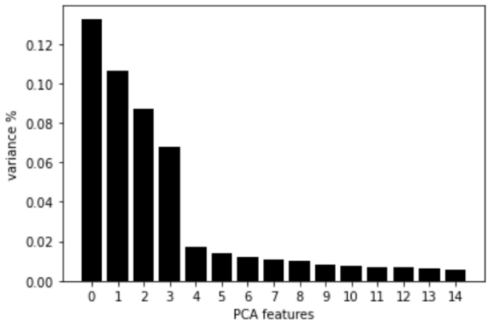
Figure 2: The Silhouette Plot



3 Data Visualization via PCA

To visualize the number of clusters we have and where they are, I employed PCA to reduce the number of features as dimensionality reduction in the dataset. I first fit the standardized data using PCA via StandScalar. Based on the cumulative variance plot, I can explore how many features I would like to keep (Figure 3).

Figure 3: Cumulative Variance Plot



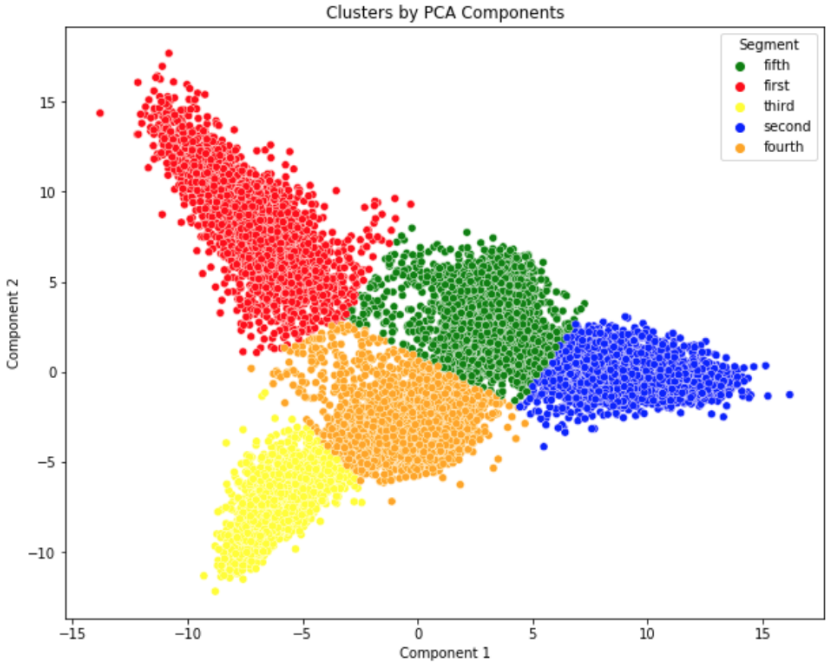
The Figure 3 graph shows the number of components we included (x-axis) with respect to the number of variance captured (y-axis). It clearly shows that the first 4 components explain the majority of the variance in the 'patients_all' dataset [4]. From this visualized case, I chose the first 4 components as the right choice. However, it might not be able to plot 4 principle components, so I finally decided 2 principal components visualize the clusters.

From the Figure 3, I performed PCA with 2 principal components. I calculated the outcome components score for the elements via *pca.transform*. And then, I incorporated the newly got PCA scores into the K-means algorithm. So then it can perform segmentation based on the 2 principal components rather than the original features.

Since I chose 5 clusters based on the Elbow and Silhouette method, I fitted our dataset with the K-means PCA model. Also, I created a new data frame with PCA scores and assigned clusters. Let's label them 'Component 1' and 'Component 2', and then I added the PCA K-means clustering labels as 'cluster.' I finally created a new column named 'Segment' and map the five clusters directly inside it [5].

To visualize the clusters on 2-dimension (2D) plane, I plotted data by PCA components (Figure 4). The X axis is 'Component 1', and Y axis is 'Component 2'.

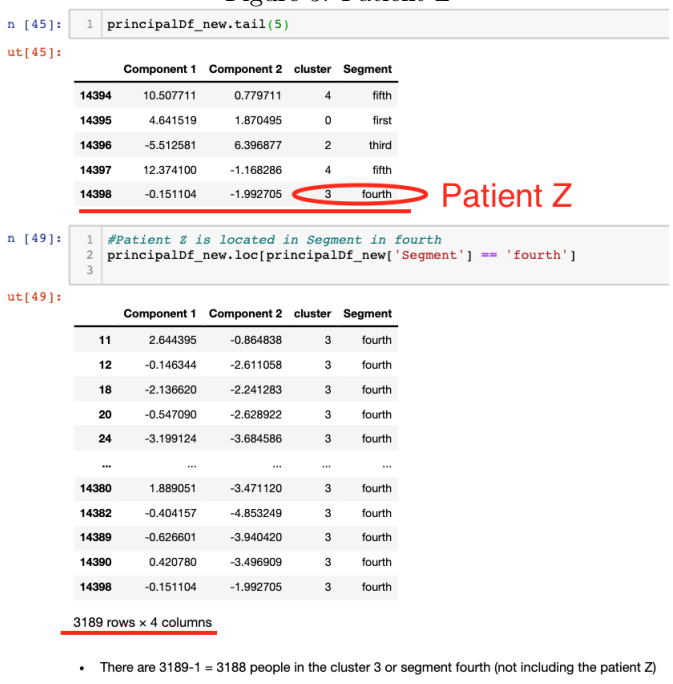
Figure 4: Clusters by PCA Components



4 Patient Z

To find the cluster to which patient Z belongs, I applied pd.concat to combine two datasets as a new data frame called 'new_df.' Notice that I also reset the index of 'new_df.' Like the previous section step, I first did data standardization, applied PCA into 2 principal components, fitted with K-means PCA models, then labeled PCA K-means clustering into 'Components 1', 'Component 2' and 'cluster.' I finally noticed that Patient Z belongs to cluster 3 and segment 'fourth'. And therefore, to find the number of patients in cluster 3 or segment fourth, I used *dataset.loc[dataset[Segment] == 'fourth']* to see that there are 3189 rows × 3 columns (Figure 5). In other words, there are 3189-1 = 3188 people in cluster 3 (not including the patient Z).

Figure 5: Patient Z



References

[1] K-means clustering. (2021, March 03). Retrieved March 06, 2021, from https://en.wikipedia.org/wiki/K-means_clustering

[2] Principal component analysis. (2021, February 22). Retrieved March 06, 2021, from https://en.wikipedia.org/wiki/Principal_component_analysis

[3] Silhouette (clustering). (2021, January 23). Retrieved March 06, 2021, from https://en.wikipedia.org/wiki/Silhouette_(clustering)

[4] Kavyazin, D. (2020, October 06). Principal component analysis and k-means clustering to visualize a high dimensional dataset. Retrieved March 06, 2021, from https://medium.com/@dmitriy.kavyazin/principal-component-analysis-and-k-means-clustering-to-visualize-a-high-dimensional-dataset-577b2a7a5fe2

[5] How to combine PCA and k-means clustering in Python?: 365 DataScience. (2021, February 22). Retrieved March 06, 2021, from https://365datascience.com/tutorials/python-tutorials/pca-k-means/