< K – MEANS CLUSTERING >

Miner username: **Spydyyy** G#: **G01348097**

	IRIS Dataset	Image Dataset
Rank	163	161
V-Score	0.72	0.51

Objective:

Implement the K-means Clustering algorithm for the standard Iris Dataset and explore methods for dimensionality reduction to deal with Image data.

Pseudocode of Implemented K-means Algorithm:

- At First, picked k random data points as the initial centers, here k=3 for Iris Dataset and k=10 for Image dataset were picked up randomly by using random.sample () to get random indexes from the dataset and select the corresponding datapoints.
- I tried to pick the initial centers manually by choosing the minimum, maximum and the mean of the data as the 3 centroids, in a way to improve the V-score. But found that it didn't reflect any change in the clustering solution.

  Repeat K-means from here
- Defined a function nearest_centroids() for assigning the data points to the nearest centroid by computing the Euclidean distance using a function called cal_dist().
- Defined a function cal_NewCentroids() for calculating the mean of each cluster and made it as the new centroid for that cluster.
- Repeated the same loop for a maximum of 100 iterations for moving the centroids.
- Made a stopping condition such that the control exits the loop if the difference between the new centroids and the previous ones < 0.1. i.e., if the centroids move by a very little value or don't move at all, then it means K-means has converged.

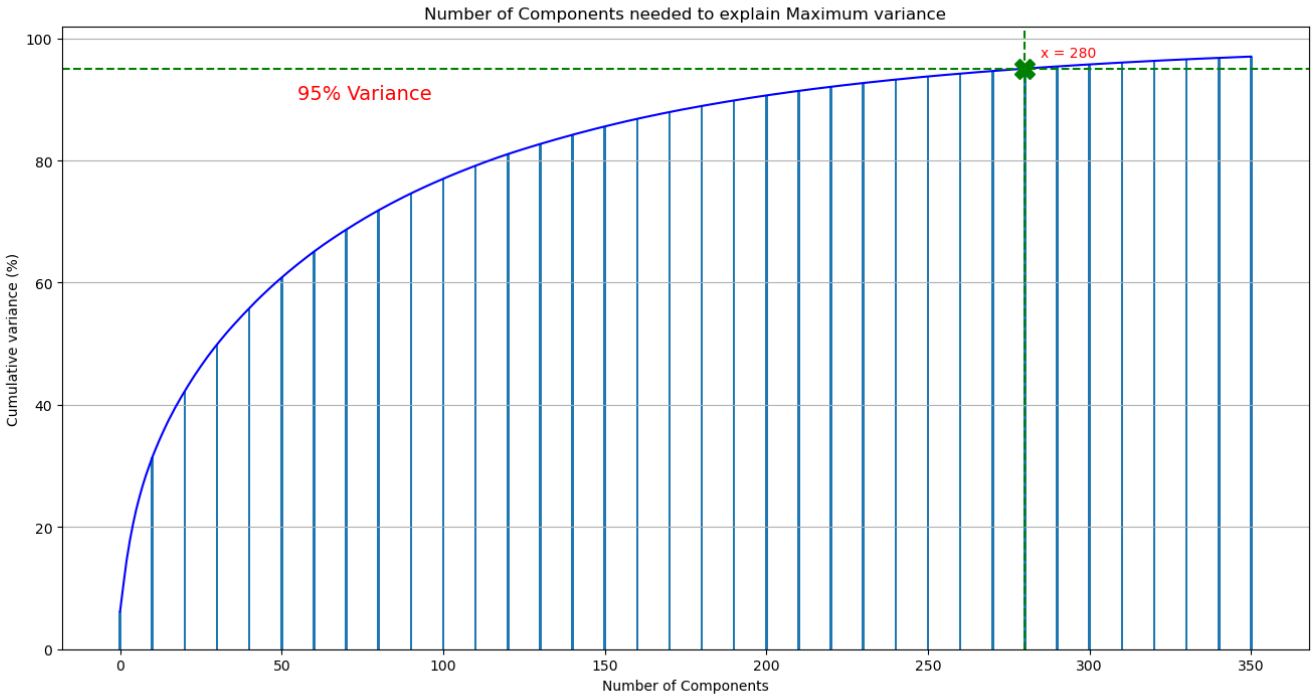
Technical Analysis:

After running the k-means algorithm repeatedly:

- The V-Score for the Iris Dataset turned out to be 72%.
- The V-Score for the Image Dataset is 51%, which doesn't seem like a good score generated by the algorithm even after applying PCA as the dimensionality reduction metric.
  - Since the Image Dataset consists of 10,000 rows X 280 principal components after applying PCA, it is still a huge dataset for computing distances and applying k-means for clustering into 10 or more clusters and repeating the whole process for many iterations.
  - So, obviously it's a time taking process as a whole for running the algorithm even for once, while K-means Clustering starts randomly and ends in different clustering solutions each time.
  - Here, the V-Score generated was only around 0.51 by the K-means clustering solution which was run for about 100 iterations for clustering the Image data with k=10 and that itself took a runtime of almost 30 Minutes.

Dimensionality Reduction using PCA (Principal Component Analysis):

- The intuition behind PCA is to find a handful of Eigen vectors with high Eigen values in the Input data that is enough to explain almost the overall variance (95 – 99%) in the dataset and using them to perform a change of basis on the data.
- With just a first few Eigen vectors known as the principal components, we can still compute and represent the exact dataset in much lower dimensions [2-100], which was previously available only in a higher dimension [500-n].
- The application of Linear Algebra helps to carry out this impossible task with ease and is even highly efficient.
- PCA cuts down a lot of computational time by reducing the dimensions from some 500-1000 features or even worse, to just a few features, and still represents almost the same amount of information as represented by the original dataset.

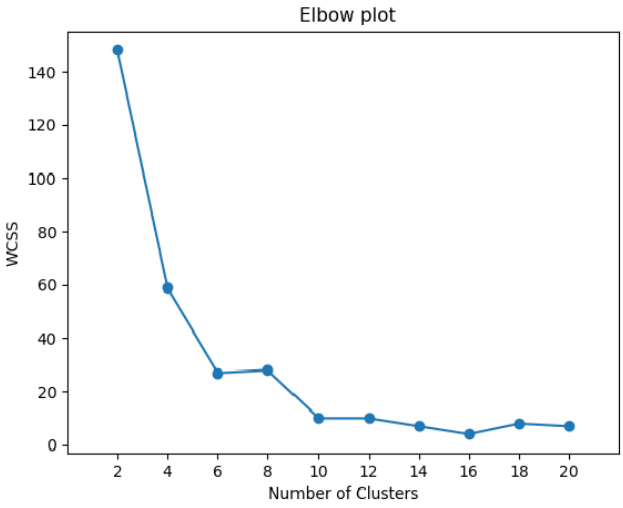


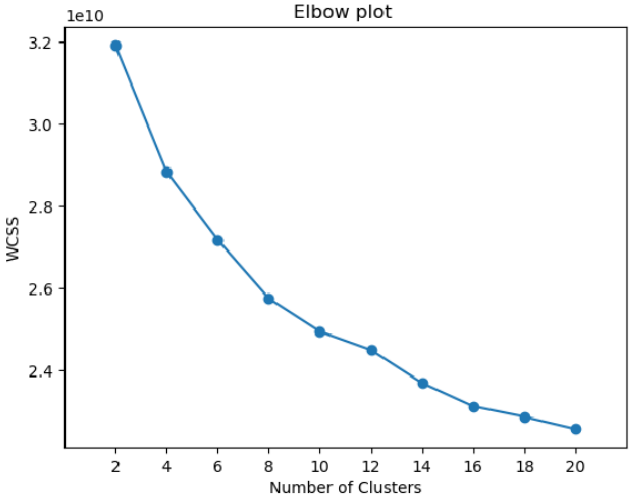
*< Number of Principal Components **Vs** (%) of Cumulative Variance in the Dataset >*

In order to find the total number of principal components (features) we want to retain from the dataset that contributes to the maximum variance, we plot the above graph with Number of Principal Components in the x-axis and the Percentage of Cumulative Variance in the y-axis.

We can clearly notice that when the number of components is between 250-350, the dataset retains the maximum variance. In order to pick a number (keeping in mind the computations that are to be performed), I chose to go with the first 280 components, since it explains almost 95% of the overall variance and also take some reasonable computing time.

So, the Image Dataset was significantly reduced from 784 dimensions to 280 dimensions by using the PCA class of the Sci-kit Learn library.





Iris Dataset **Image Dataset**

*< Number of Clusters **Vs** Within Cluster Sum of Squared error >*

As you can see, the above plot represents the number of clusters ranging from 2 to 20 in steps of 2 against the Within Cluster Sum of Squared error. This plot is constructed to measure how well the implemented k-means algorithm performs by measuring the distances between centroids and datapoints with respect to the number of clusters.

- In Iris dataset, we can clearly see that the WCSS starts decreasing rapidly when we start increasing the number of clusters and when it reaches 6 clusters, the intra-cluster distances seem to be very less, i.e., clusters are more tightly packed and eventually leads close to 0 for 10 or more clusters. So, the optimal k value is 6 clusters since that's where the elbow point is seen, and we did implement it for 3 clusters for which the V-score was around 70%.
- The plot for Image dataset almost took about 66 minutes for running because it had to compute the entire dataset with 280 features for 2 to 20 clusters and iterate 100 times for each k value.
    - So here, it is observed that WCSS did decrease as the number of clusters increases, but it's still high for 10 clusters, which is our given primary k value to cluster the image data.
    - This indicates that the k value should be far more higher than 10 for the WCSS to be reasonably low. But I'm not sure why K should be more than 10 for clustering 10 digits.
    - When I was trying to find the answer for the same, I found out from an online citation *[see below for reference]* that since there can be multiple ways of orientation to write a particular number, the algorithm views them as drastically different images. So, there is a need for more than 1 cluster to represent a particular number and hence this could be the reason why the V-score of my clustering solution was always around 50%, which conflicts my earlier intuition about increasing the number of iterations for k-means to converge. So, I guess if we increase the number of clusters in this same model, then we could achieve some better results of the V-scores. Reference to the article: *https://medium.com/@joel_34096/k-means-clustering-for-image-classification-a648f28bdc47*

Conclusion:

Therefore, the k-means algorithm was successfully implemented for the standard Iris dataset and Image dataset, performed dimensionality reduction using PCA on the pixel encodings, thereby discussing and analyzing technical findings.