HW5- K-Means Algorithm

Name: B Sai Sahana Bhargavi
Miner Name: word2vec

1.Introduction: To implement K-means algorithm and to evaluate the standard Iris dataset and to deal with image data and to explore methods for dimensionality reduction and to implement the internal evaluation metric for analyzing clustering solutions.

2.Approach:

K-means algorithm:

1. Initialize 'K' centroids with random points of the respective dataset, where K is number of clusters.
2. centroids acts as the mean points of the cluster.
   ```
   centroids= X.sample(n=k).values
   ```
3. distance is computed using cosine similarity between each data point and the centroids, and that data point is assigned to the closest centroid.
   ```
   distances = cdist(data, Initial_centroids ,'cosine')
   ```
4. Recalculate the centroid for each cluster and observe the difference between new centroids and old centroids.
   ```
   new_centroids = pd.DataFrame(X).groupby(by=cluster).mean().values
   if(np.count_nonzero(centroids-new_centroids) == 0)
   ```
5. Repeat the steps from 3 to 4 until the algorithm converges, means the difference is zero between the new and old centroids.

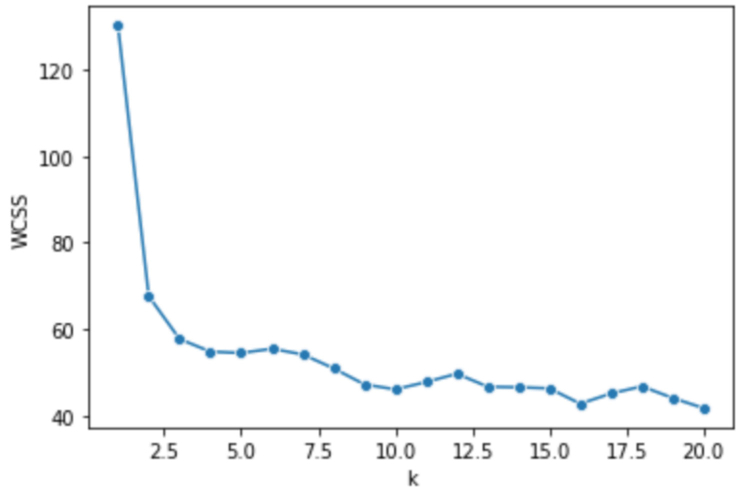
Internal Evaluation metric: `calculate_cost(X, centroids, cluster)`
calculated the sum of squared errors (SSE) for a given value of K using the below formula. SSE is the sum of the squared distances between the cluster's centroid and each member, and plotted this metric on y-axis with value of K increasing from 2 to 20 in steps of 2 for the data.We will notice that as K increases, SSE drops and disortation diminishes.and K value can be found at which the graph drastically decreases and this creates a "elbow effect."

- $$SSE = WSS = \sum_{i} \sum_{x \in C_i} (x - m_i)^2$$

Part 1: Iris

- Data extraction: read the iris dataset (test_data.csv)that contains test data as 150 instances with four features sepal length,sepal width,petal length,petal width in cm without any labels and training data is null.
- K-means algorithm: the algorithm processes the data and divides the records into three clusters 0,1, 2 and the V-measure is computed on the miner for the predicted output cluster file which is turned out be 0.95.
- Internal Evaluation metric:

It can be seen from the below graph that the WSS(within group sum of squares) decreases gradually as the K value increases creating an elbow effect where K can be 3-4 and this is the optimal number of clusters.

- 

Part 2: Image

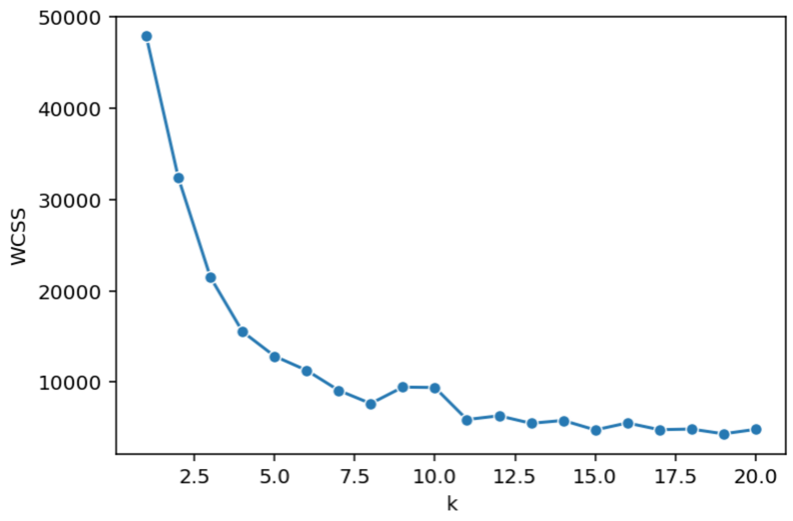
- Data extraction:Input Data (test_img_data.csv) consists of 10,000 images of handwritten digits (0-9). The images were scanned and scaled into 28x28 pixels. For every digit, each pixel can be represented as an integer in the range [0, 255] where 0 corresponds to the pixel being completely white, and 255 corresponds to the pixel being completely black. This gives us a 28x28 matrix of integers for each digit and then flattened each matrix into a 1x784 vector.
- Pre-processing: Standardized the data to have a mean of ~0 and a variance of 1
  X_std = StandardScaler().fit_transform(Testdata_img_df)
  When k-means algorithm run on image data set without applying dimensionality reduction V-measure on miner before applying was 0.53 which is very low,
  Dimensionality reduction:
  As dimensionality increases, data becomes sparse in the space occupied by it, the distance measure becomes less meaningful while clustering points and outlier detection. hence in-order to improve performance, below techniques are applied before clustering
  PCA:The primary goal of this technique is to minimize the dimensionality of highly linked data by transforming the original set of vectors into a new set known as the Principal component and is to find a projection that captures the largest amount of variation in data and 784 features are reduced to 72 .

```
pca = PCA(n_components=72)
pca_result = pca.fit_transform(X_std)
```

  t-SNE:t-distributed stochastic neighborhood embedding. It embeds the points from a higher dimension to a lower dimension trying to preserve the neighborhood of that point. Further, 72 components in pca_result are reduced to 2 components in t_sne.

```
tsne = TSNE(n_components=3, verbose=1, perplexity=8, n_iter=250)
tsne_results = tsne.fit_transform(pca_result)
```

- K-means algorithm: the algorithm processes the reduced data and divides the records into three clusters 0 to 9 clusters and the V-measure is computed on the miner for the predicted output cluster file which is turned out be 0.79.
  `X=pd.DataFrame(tsne_results)`
- Internal evaluation metric:It can be seen from the below graph that the WSS decreases gradually as the K value increases creating an elbow effect where K can be 9-10 and this is the optimal number of clusters.



results are displayed as below in the table:

	Iris dataset	Image dataset
V-measure on miner	0.95	0.79
Rank on miner	14	101
Number of iterations for convergence	1	150
Internal evaluation metric -WSS	3-4	9-10

Conclusion: this homework introduced me how unsupervised learning can be implemented and its capability of classifying unlabeled data into a predetermined number of clusters based on similarities (k) and to avoid curse of dimensionality using dimensionality reduction measures and evaluating clusters using internal metrics such as WSS or SSE. and further it paved a desire to explore more in unsupervised learning techniques such as Hierarchical Clustering.