Name: Nooreldean Koteb

Miner Username: IcyEagle

Iris | Rank: 11 **Accuracy:** 95%

Image | Rank: 17 **Accuracy:** 74%

Steps for clustering:

For my solution I first began by creating an average K-means function. I used dictionaries to save centroids, clusters, and the Sum squared error results for each centroid. I then created a bisect function that turns my normal K-means function into a bisect K-means.

K-Means:

Step1: Create random points for centroids.

Step 2: Measure the Sum Squared Error between each data point and each centroid then store it.

Step 3: Measure Sum Squared Error between every data point and every centroid then assign that data point to the closest centroids group.

Step 4: Average the data points in each group and assign that as the new centroid point. If a cluster is empty, then the centroid is assigned a new random point.

Step 5: Repeat Steps 2-4 until new Sum Squared Error is not different from the last run.

Bisect K-Means:

Step 1: Initialize by running the above K-means once with k set to 2.

Step 2: Find largest cluster and call K-means on it with k set to 2.

Step 3: Repeat step 2 until we have K original number of centroids we wanted.

Approach

For my initial centroids I decided to randomly pick points by using the random uniform function that took the data's maximum and minimum values as input. I then decided to implement the bisect K-means method which made initial centroids values less important than it was before.

I implemented the bisect function as an addon to the K-means function since it was repetitive, and I had already implemented K-means. Once my bisect K-means function was complete I started testing various preprocessing methods to try and improve my score.

Feature Selection

Iris dataset:

For the Iris dataset I was able to get 95% on miner by normalizing my data. PCA, Factor Analysis, and TSNE did not improve the results beyond 95% and in most cases hurt the results.

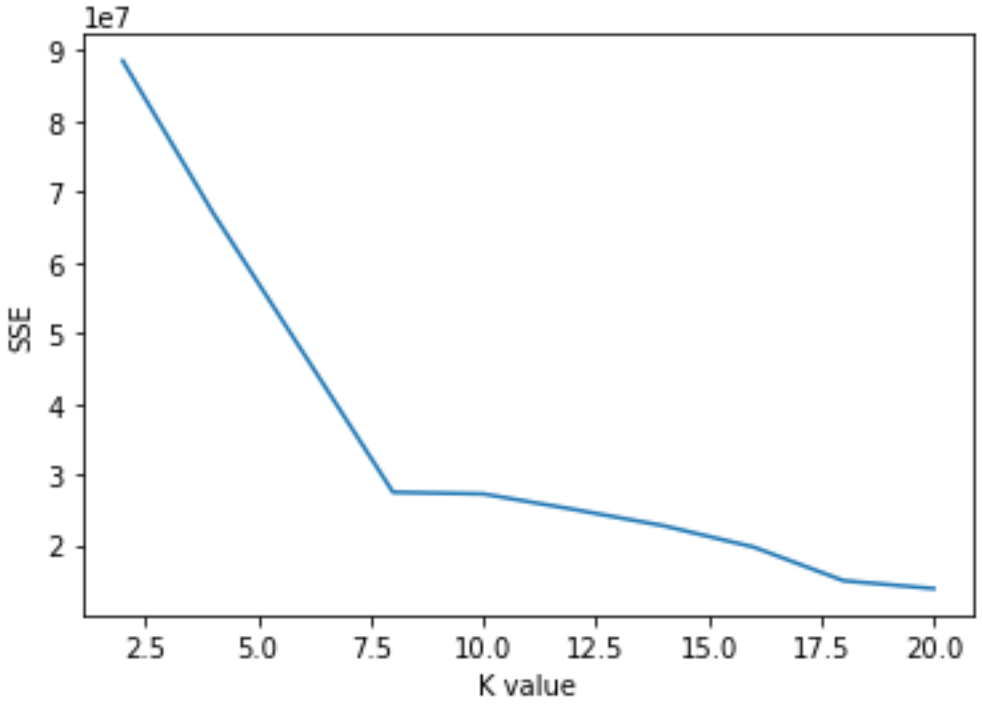
Digit Images:

For feature selection/reduction I combed through the data and removed any columns that were full of only zeros because this meant that there was no data there. I then replaced all zeros with ones and any non-zero value to a zero. This gave me a binary list with zeros where the data used to be and ones where zeros used to be. The data was reduced from 784 to 673 columns. I then used TSNE that took the high dimensional data and converted it to 2-dimensional data.

I tried to normalize the data, PCA, and Factor Analysis, however they all resulted in a worse score. Prior to TSNE my removal of empty columns and binarization of the data resulted with scores in the 50s, After TSNE it resulted with scores in the 70s.

Evaluation Metric

The evaluation metric used was Sum Squared Error.

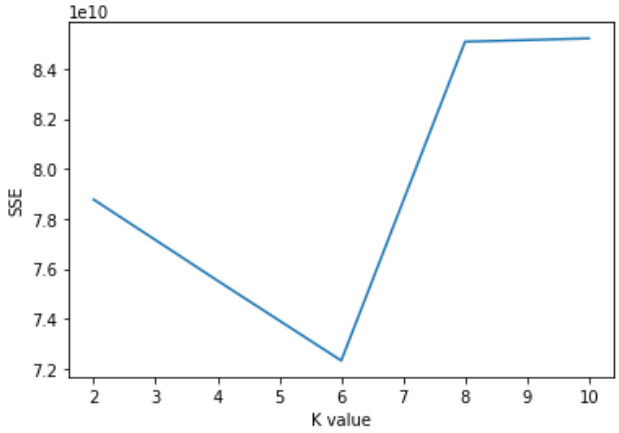


Results

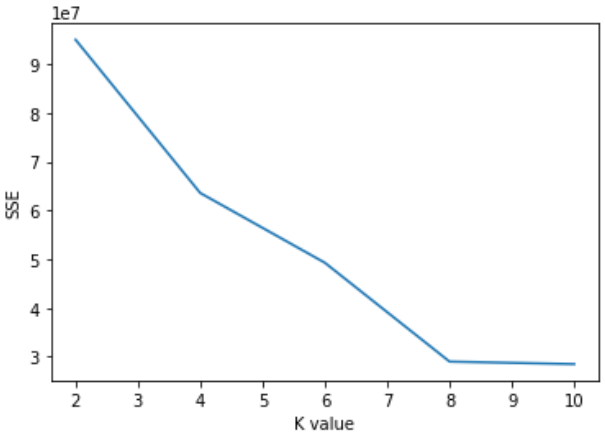
These are my SSE results for the digit Images dataset using bisect K-means. Although using TSNE gave me a higher SSE, it also gave me a much better score on miner. The number next to TSNE is the perplexity. Increasing the perplexity gives a lower SSE score, however my results varied by a few points every time I submitted to miner. TSNE at default 30 perplexity was my final submission giving me 74%

Final SSE	Preprocessing Methods	#
16,27,805	Column removal & binarization	1
20,787,327	TSNE(50), Column removal & binarization	2
26,727,850	TSNE(30), Column removal & binarization	3
28,481,963	PCA(200), TSNE, Column removal, and data binarization	4
85,211,198,159	No preprocessing	5

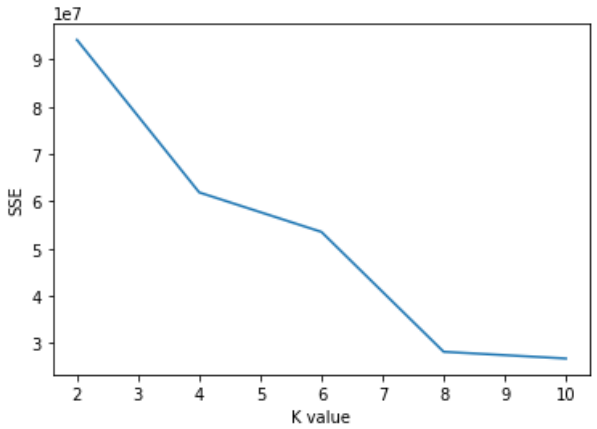
No Preprocessing



PCA(200), TSNE, Column removal, and data binarization



TSNE(50), Column removal & binarization



Column removal & binarization