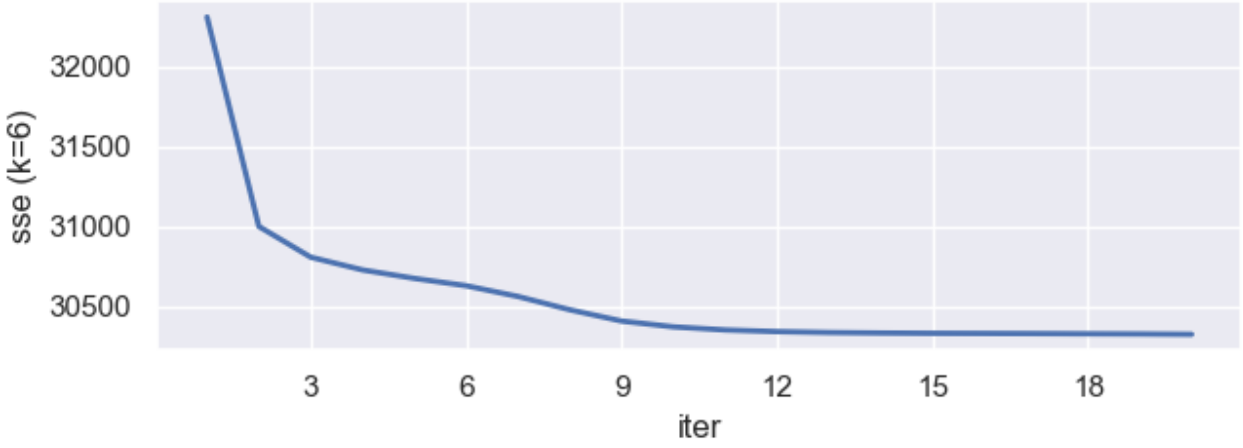
Class: CS 434
By: Logan Saso, Rajat Kulkarni, Evan Medinger
Assignment 4:

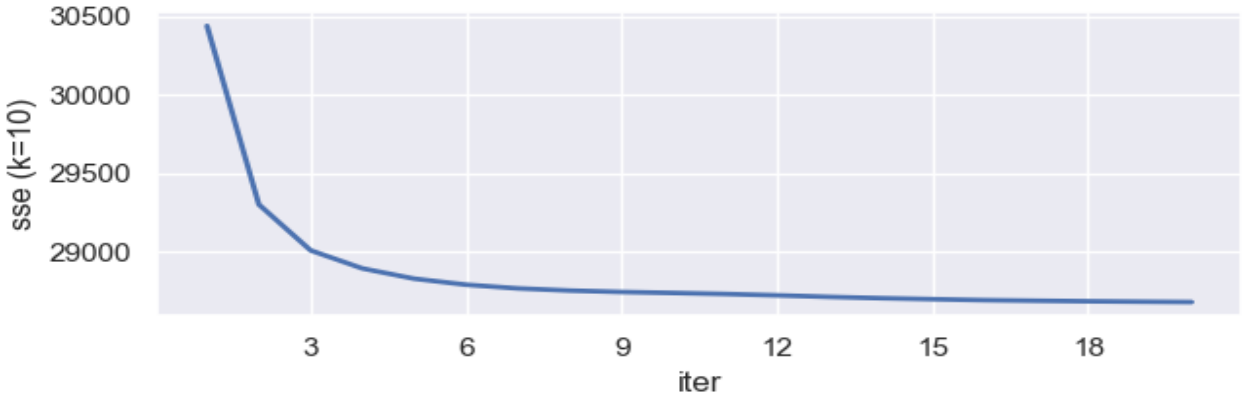
This assignment focused on teaching students how to cluster a vast amount of data that takes in many variables that aren't directly related to each other. Plotting a best-fit curve through all the dimensions would be complex to interpret on a graph. Especially on data with many features. This is where clustering can simplify the data into the most important components.

1. For this section, this required completing the code of some of the functions in the clustering.py module. After being coded into a working state, this part would search for *k* number of centers in the graph so that each sample can be classified under its own point. Depending on the number of centers, the purity of the clusters can vary, but as more centers are introduced the purity of each cluster increased overall. However, this may come at a cost in the long run. As more centers are defined, it risks that the data will begin to overfit to the training sample.

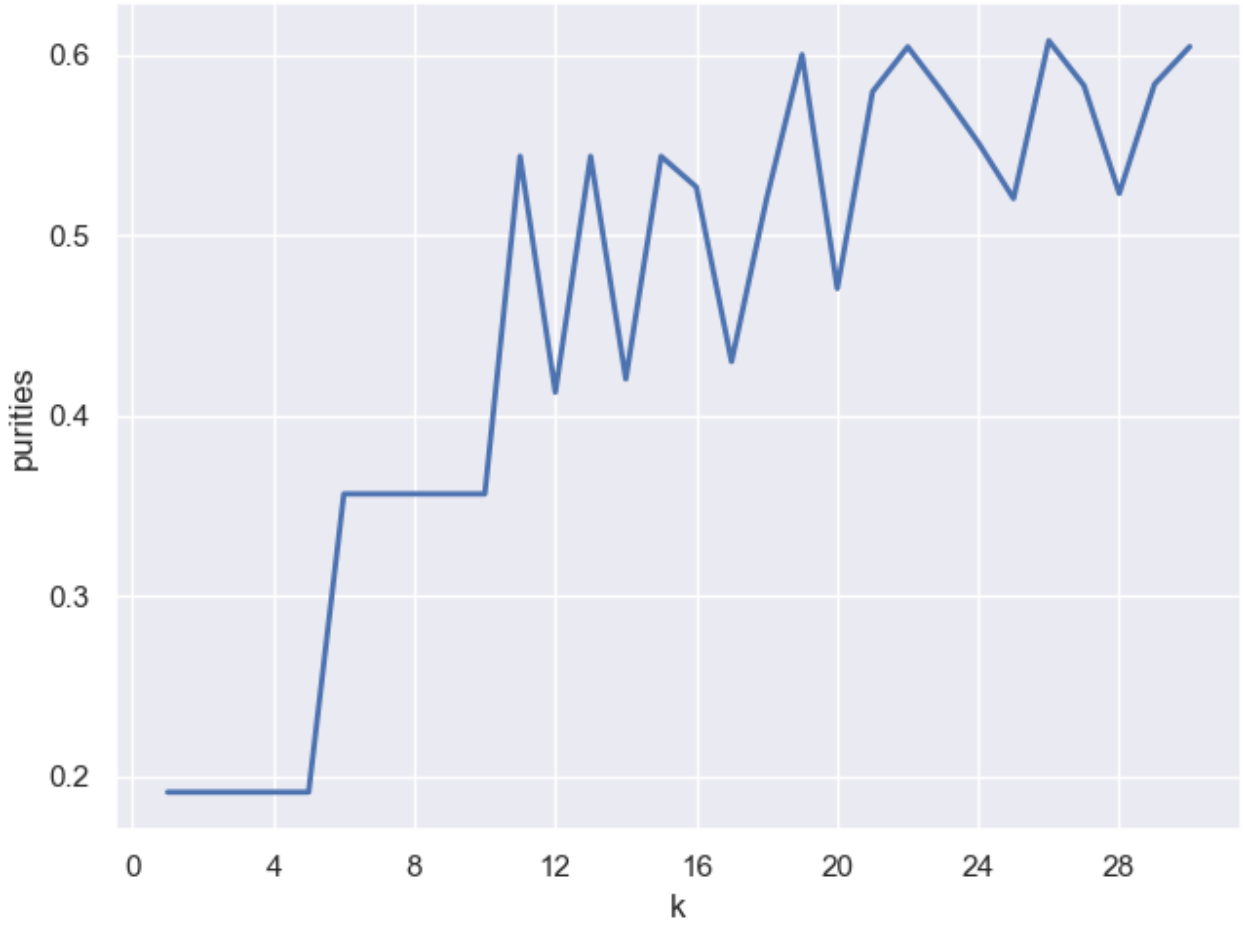
When applying the program to k=6, this graph is generated:



As for manipulating the k value between 1-10, the best graph generated was k =10. The following graph shows the least amount of error over the sample with the best elbow being displayed at either iteration 2 or 3.

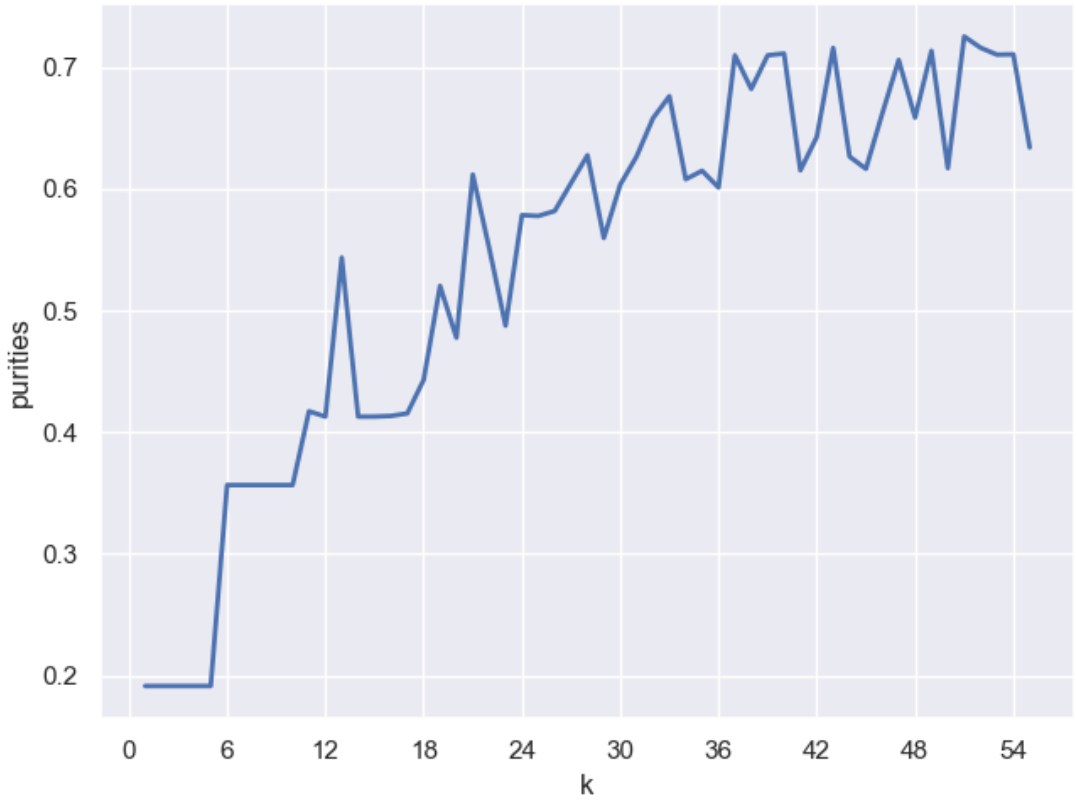


As for the average purity for the samples, as the k increases, the purity of the clusters also increases. The following graph shows this trend:



2. For this section, it requires the functions *decompose* and *visualize* to allow for PCA to work. Based on the calculations made by part 1, this graph is generated:

It was able to refine the components down to 2 notable variables, but some of the samples are seen overlapping each other. When using this data to figure out its purity, this graph is generated:



Compared to the original purity graph from the past section, it shows little variation and seems to support that the PCA doesn't harm the purity of the clusters.