May 11, 2023

Manjeet Acharya

Final Project

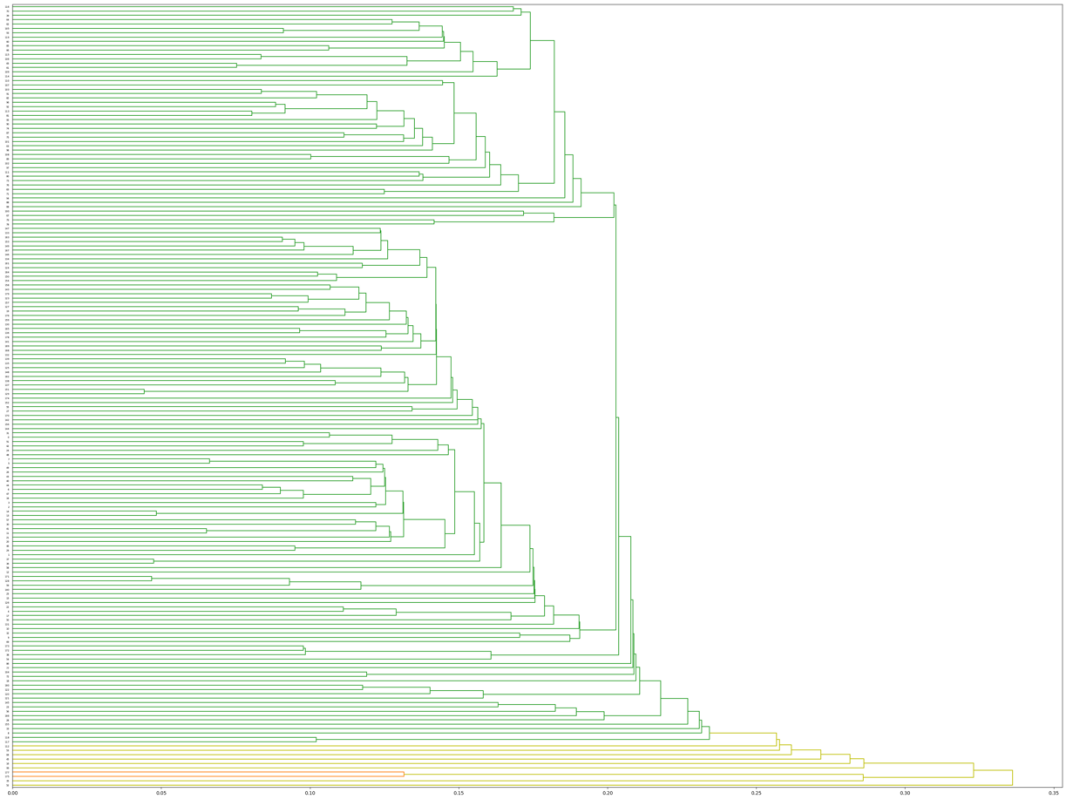
My project was about the hierarchical clustering of UCI seed data set and training a KNN algorithm based on the cluster ID.

I took two different approaches to solving this problem.

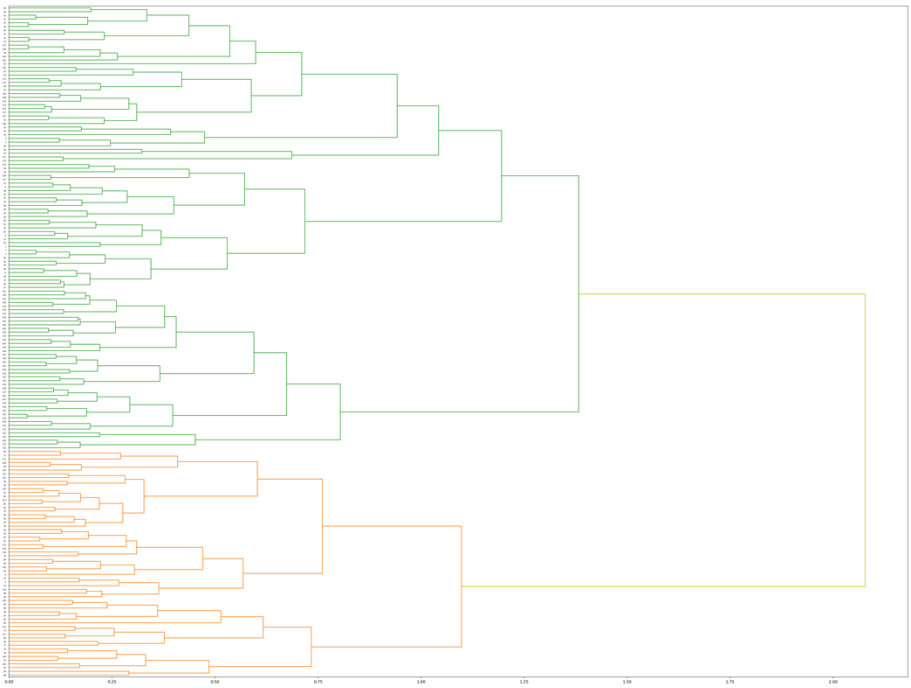
First approach: I separated the test data from the original data that contained labels. I took 10 data points of each class for testing and rest for clustering. For the clustering data, I dropped all the class labels and performed single linkage and complete linkage hierarchical clustering.

I also constructed dendrograms for each linkage and found that the complete linkage better fit my data. Additionally, I calculated the accuracies of the clusters based on the original class labels. The accuracy of the complete linkage clustering was higher than the single linkage clustering. For both types of clustering, the accuracies got higher as the number of clusters got higher.

Dendrogram for single linkage:



Dendrogram for complete linkage:



This is the accuracies of my clusters for different values of k during **single linkage** clustering. Here, k is the number of clusters.

```
When k =  2 accuracy is equal to: 0.33888888888888885
When k =  3 accuracy is equal to: 0.35
When k =  4 accuracy is equal to: 0.35000000000000003
When k =  5 accuracy is equal to: 0.35555555555555557
When k =  6 accuracy is equal to: 0.3611111111111111
When k =  7 accuracy is equal to: 0.3666666666666667
When k =  8 accuracy is equal to: 0.3666666666666667
When k =  9 accuracy is equal to: 0.3722222222222222
When k =  10 accuracy is equal to: 0.3777777777777778
```

This is the accuracies of my clusters for different values of k during **complete linkage** clustering. Here, k is the number of clusters.

```
When k =  2 accuracy is equal to: 0.65
When k =  3 accuracy is equal to: 0.8666666666666667
When k =  4 accuracy is equal to: 0.8666666666666667
When k =  5 accuracy is equal to: 0.8666666666666667
When k =  6 accuracy is equal to: 0.8666666666666667
When k =  7 accuracy is equal to: 0.8666666666666667
When k =  8 accuracy is equal to: 0.8666666666666667
When k =  9 accuracy is equal to: 0.8666666666666667
When k =  10 accuracy is equal to: 0.8666666666666667
```

Next, I labeled my clusters based on the original labels and the majority labels of the data points in each cluster. This labeled clusters were now the training set for my KNN algorithm, which uses Euclidian distance as the similarity measure.

For this, I used the complete linkage clusters with 3 clusters as my training data. The KNN algorithm showed the following accuracies for different values of K.

These are accuracies of KNN algorithm for different values of K. Here k is the number of nearest neighbors.

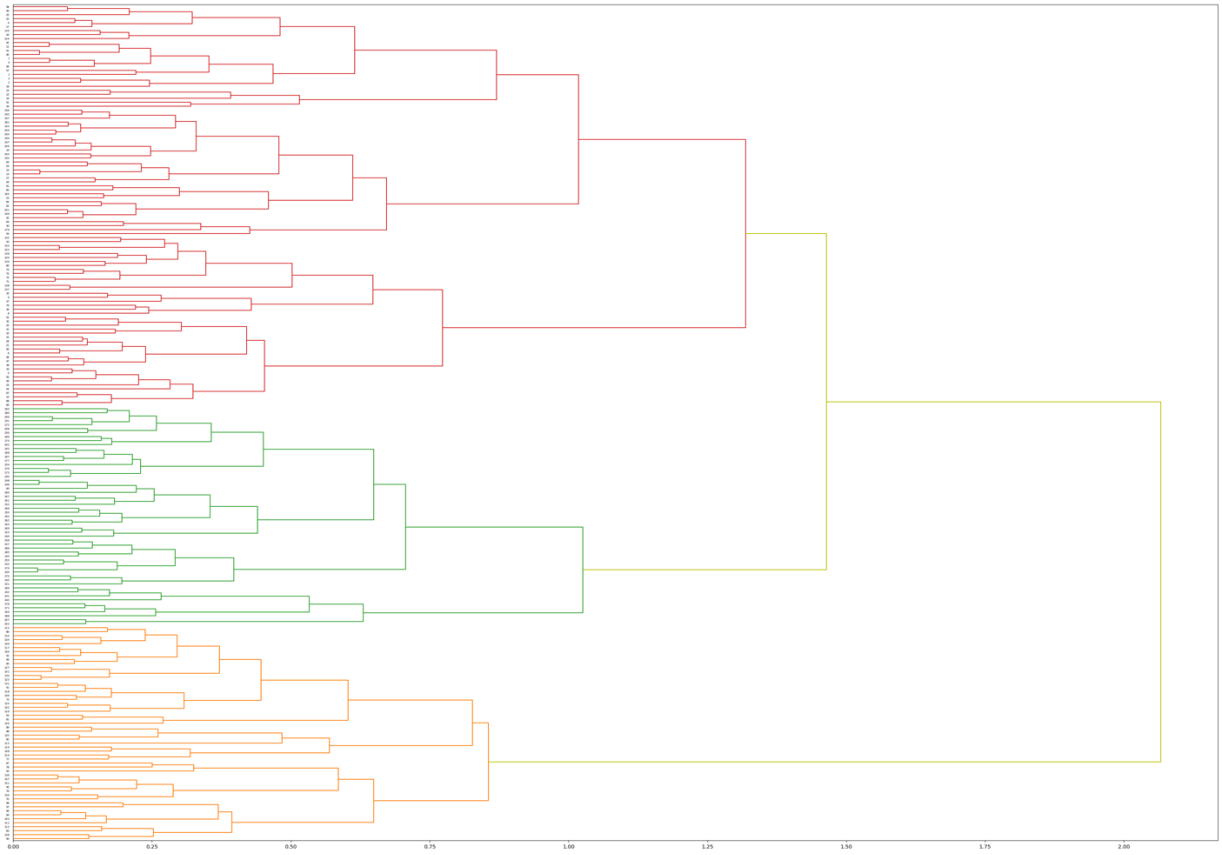
```
For k = 1 the accuracy is : 0.8333333333333334
For k = 2 the accuracy is : 0.8666666666666667
For k = 3 the accuracy is : 0.8666666666666667
For k = 4 the accuracy is : 0.9
For k = 5 the accuracy is : 0.9
For k = 6 the accuracy is : 0.8666666666666667
For k = 7 the accuracy is : 0.8333333333333334
For k = 8 the accuracy is : 0.8333333333333334
For k = 9 the accuracy is : 0.8333333333333334
For k = 10 the accuracy is : 0.8333333333333334
```

Second Approach: I used the entire UCI seed dataset for hierarchical clustering. First, I dropped all the class labels and ran single linkage and complete linkage hierarchical clustering. After running the clustering, I separated the cluster into training and testing set.

Dendrogram for the data set before splitting the test set.

Single Linkage:

Complete Linkage:



I used the cluster ID as labels and trained the KNN algorithm on the training set from clustered dataset. For this, since I was no longer restricted to using the original labels, I used different number of clusters and tested the accuracies of my KNN algorithm.

First, I used **complete linkage and 3 clusters**.

```
For k = 1 the accuracy is : 1.0
For k = 2 the accuracy is : 0.9666666666666667
For k = 3 the accuracy is : 0.9666666666666667
For k = 4 the accuracy is : 0.9666666666666667
For k = 5 the accuracy is : 0.9333333333333333
For k = 6 the accuracy is : 0.9333333333333333
For k = 7 the accuracy is : 0.9333333333333333
For k = 8 the accuracy is : 0.9666666666666667
For k = 9 the accuracy is : 0.9666666666666667
For k = 10 the accuracy is : 0.9666666666666667
For k = 11 the accuracy is : 0.9666666666666667
For k = 12 the accuracy is : 0.9666666666666667
For k = 13 the accuracy is : 0.9666666666666667
For k = 14 the accuracy is : 0.9666666666666667
For k = 15 the accuracy is : 0.9333333333333333
For k = 16 the accuracy is : 0.9333333333333333
```

I found that the accuracy was the highest when the algorithm only took one nearest neighbor. But the overall accuracies of KNN algorithm for this approach was higher than the first approach.

Complete linkage and 5 clusters.

```
For k = 1 the accuracy is : 0.72
For k = 2 the accuracy is : 0.68
For k = 3 the accuracy is : 0.68
For k = 4 the accuracy is : 0.7
For k = 5 the accuracy is : 0.7
For k = 6 the accuracy is : 0.72
For k = 7 the accuracy is : 0.72
For k = 8 the accuracy is : 0.72
For k = 9 the accuracy is : 0.72
For k = 10 the accuracy is : 0.72
For k = 11 the accuracy is : 0.72
For k = 12 the accuracy is : 0.72
For k = 13 the accuracy is : 0.74
For k = 14 the accuracy is : 0.74
For k = 15 the accuracy is : 0.74
For k = 16 the accuracy is : 0.76
For k = 17 the accuracy is : 0.74
For k = 18 the accuracy is : 0.76
For k = 19 the accuracy is : 0.76
```

For this, I couldn't use the single linkage with test data with 10 samples from each cluster because most of my clusters had only one data points in it. So instead, I used 5 clusters and 1 point from each cluster as a test point.

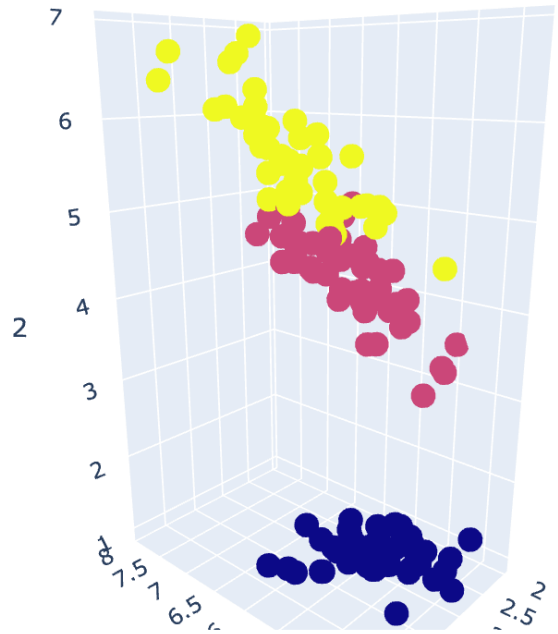
Single linkage with 5 clusters.

```
For k = 1 the accuracy is : 0.4
For k = 2 the accuracy is : 0.4
For k = 3 the accuracy is : 0.2
For k = 4 the accuracy is : 0.2
For k = 5 the accuracy is : 0.2
For k = 6 the accuracy is : 0.2
For k = 7 the accuracy is : 0.2
For k = 8 the accuracy is : 0.2
For k = 9 the accuracy is : 0.2
For k = 10 the accuracy is : 0.2
```

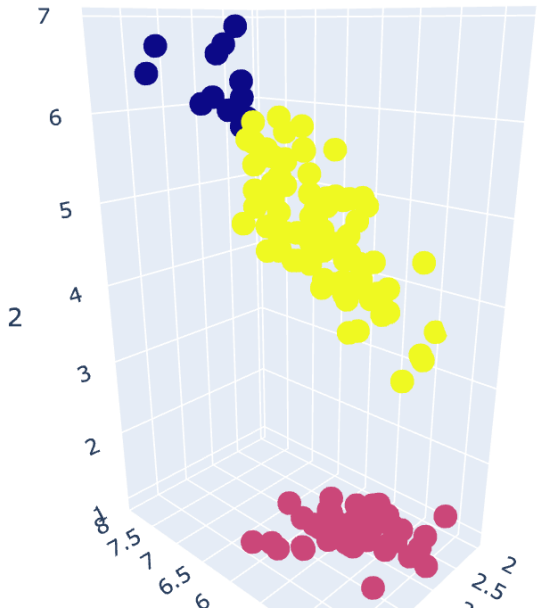
Therefore, for this type of dataset, I found out that it's better to predict the cluster ID of the new data and assign them to their respective clusters first. This seems to be a better approach for classifying the new data.

Further, I wanted to visualize how my KNN algorithm behaves for the data points that are deep in the cluster and between the clusters. I especially wanted to see how my KNN algorithm classifies the data points that are between the clusters. I took the Iris data set and only took the first three features to plot the data points on a 3-d space.

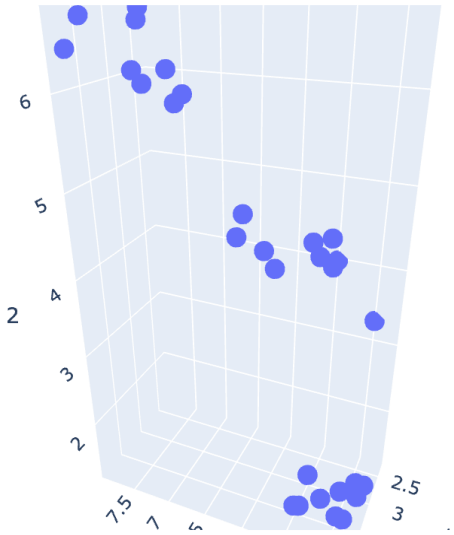
Here's my **original data.**



After that, I ran the clustering algorithm on the entire data set, so here's **our clusters when we have 3 clusters:** *Ignore the color because the clusters have different ID, and the ID does not match with the class from the original dataset.*

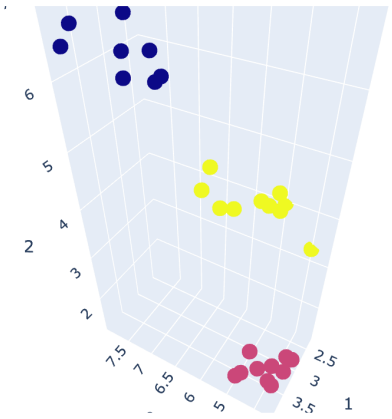


And this is **our test data set**.

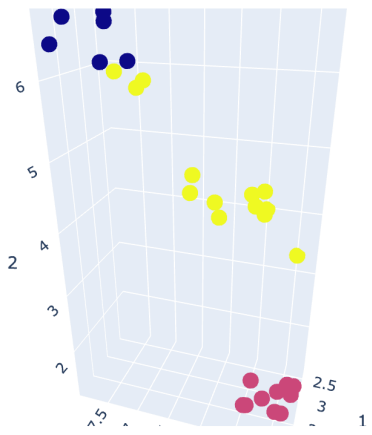


Notice the points that are near 6 on the y-axis. They fall between two clusters, so we will observe how these points are classified for different values of k.

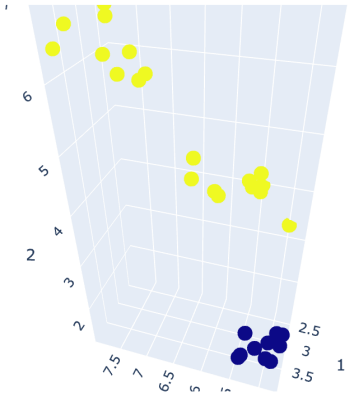
When k = 1



When k = 3



When k = 6



Looking at the clusters and the predicted cluster IDs for the testing data, we can see that k=3 classifies the data points that are in between the clusters much more differently than k=1 or k=6. For the lower values of k, the data points seem to be sucked into the blue cluster. This shows a huge distance between the clusters.

But for k=3 or k=4, the yellow cluster takes some of the points that are near the clue cluster. There's a less distance between the yellow and blue cluster. They're almost overlapping each other.

Finally, when value of k=6 or higher, the data points are completely classified with the yellow cluster ID and the KNN algorithm cannot predict the blue cluster ID.

It seems that the algorithm performs better for the points that are in between the clusters when the value of k is 3 or 4.

In our algorithm for the UCI seed data set, even though the accuracy was the highest for when k=1, we should use the value of k=3 or 4 for a better classifier. Because the classifier is robust to noise when k =3 or 4.