# Hierarchical Clustering on the Iris Dataset

Joan Ball, joan2, CS 466, Fall 2020

Git repository with code and usage guidelines can be found here (https://github.com/joaniffer/466_project).

## Background

Hierarchical clustering is an iterative and traditionally unsupervised method of classification. Along with data as input, hierarchical clustering also requires the specification of how to measure similarity of distance. Two different "distance metrics" (methods to evaluate the distance between two data points) and three different "cluster similarity measures" (methods to evaluate the distance between to clusters) were implemented. Distance metrics included Manhattan (a.k.a. "city block") and vector (a.k.a. Euclidean). Cluster similarity measures included simple (two closest points between two clusters), complete (two farthest points between each cluster), and average (average points of each clusters).

The Iris data set includes 50 data points for each of three different species: Iris setosa, Iris virginica, and Iris versicolor. Each data point includes four measurements: the length and width of the petals and sepals. Additionally, one species, Setosa, is relatively easy to distinguish while the other two species', Virginica and Versicolor, data points are more intermingled with each other.

Because Virginica and Versicolor are harder to distinguish from each other, using clustering (as opposed to supervised classification, such as rule-based) to classify each data point as one of the species becomes an interesting problem. However, it is observable that the differences in some of the measurements between Virginica and Versicolor are greater than others. So, in addition to implementing the hierarchical clustering algorithm to use with the Iris dataset, I will also attempt to use the more "distinguishable measures" to assist in clustering by giving them a higher weight.

## Methods

Code was written in a Jupyter notebook using Python 3.7.3.

From SKLearn, `load_iris` was imported to load the Iris dataset, `normalized_mutual_info_score` was imported to calculate the normalized mutual information (NMI) for resulting clusters, and `preprocessing` was imported to normalize each feature/measurement of the data set. From SciPy, `hierarchy` was imported to visualize the dendrogram calculated with the clustering algorithm. `reduce` from `functools` and `operator` were imported to derive a quick "n choose k" function. `partial` was additionally imported from `functools` to reuse functions that use similar parameters (distance metrics). `matplotlib` was imported to visualize the dendrogram, and `numpy` was imported for array/vector manipulation. `threading` was used to assist the optimization function. Versions of all installed libraries are included in the readme in the repository.

The hierarchical clustering algorithm was implemented based on the pseudocode provided in An Introduction to Bioinformatics (Jones & Pevzner, 2004). Some metadata about the intermediate clusters formed were appended to a "Z" matrix as defined in the usage guidelines for the SciPy `hierarchy.dendrogram` function.

Three different clustering evaluation methods were implemented: purity, Rand index (RI), and NMI. The definitions off which these implementations are based can be found in the chapter "Evaluation of clustering" from Introduction to Information Retrieval (Manning, Raghavan, and Schütze, 2008).

These cluster evaluation functions were applied to the "final" three clusters for simplicity since the Iris dataset includes three species or classifications. Since the hierarchical clustering algorithm continued until there was one cluster that included all data points, another function was implemented to "work backwards" from the final cluster and the Z matrix to find our final three clusters. Since only three clusters are desired, this algorithm is rather straightforward: the one final cluster is split into two, and the subcluster that has a higher distance measure is split again, resulting in three clusters. Special cases include when one subcluster has only one leaf (in this case the other subcluster is split, regardless of distance) and when both subclusters have equal distance (in this case it is chosen that the left subcluster is split). The information needed to find these subclusters and their distances are all recorded in the Z matrix.

To optimize the weights on each of the datapoint measures (sepal and petal length and width) in order to improve clustering, a greedy algorithm was implemented. This algorithm stopped when the difference between the current and previous iteration was below 0.001 and would iterate based on which difference in weighting best improved the specified singular clustering evaluation method. The choice in weighting difference was between an increase or decrease of 1, 0.5, or 0.25 on only one of the weights (six total options for changes to each weight, resulting in 24 different choices).

## Results

Beginning with default (equal) weighting and normalized values, Table 1 describes the results of all combinations of the distance metrics and cluster distance functions.

*Table 1.*
*Default weights, normalized measures*

| | | Purity | RI | NMI |
|---|---|---|---|---|
| Simple | Manhattan | 0.68 | 0.77664 | 0.71746 |
| | Vector | 0.66667 | 0.77181 | 0.73368 |
| Complete | Manhattan | 0.82 | 0.82362 | 0.69523 |
| | Vector | 0.78667 | 0.80215 | 0.65305 |
| Average | Manhattan | 0.68667 | 0.77709 | 0.7131 |
| | Vector | 0.68667 | 0.77709 | 0.7131 |

The dendrograms for Simple-Manhattan, Complete-Manhattan, and Average-Manhattan are included as Figure 1a, Figure 1b, and Figure 1c, respectively, below.

*Figure 1a.*
*Simple-Manhattan, corresponding to Table 1*

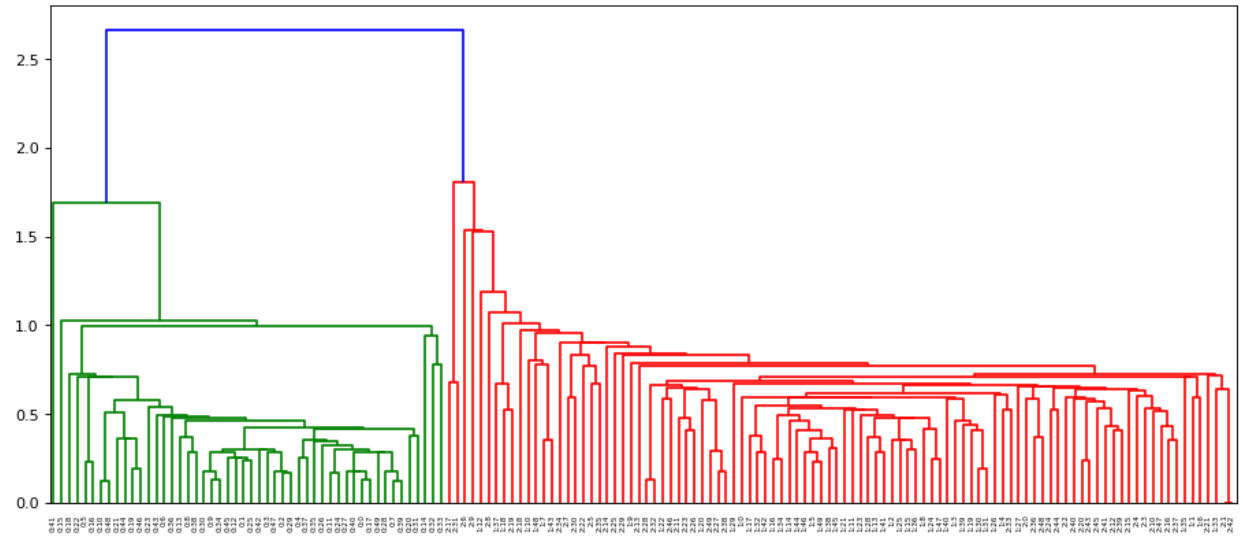



*Figure 1b.*
*Complete-Manhattan, corresponding to Table 1*

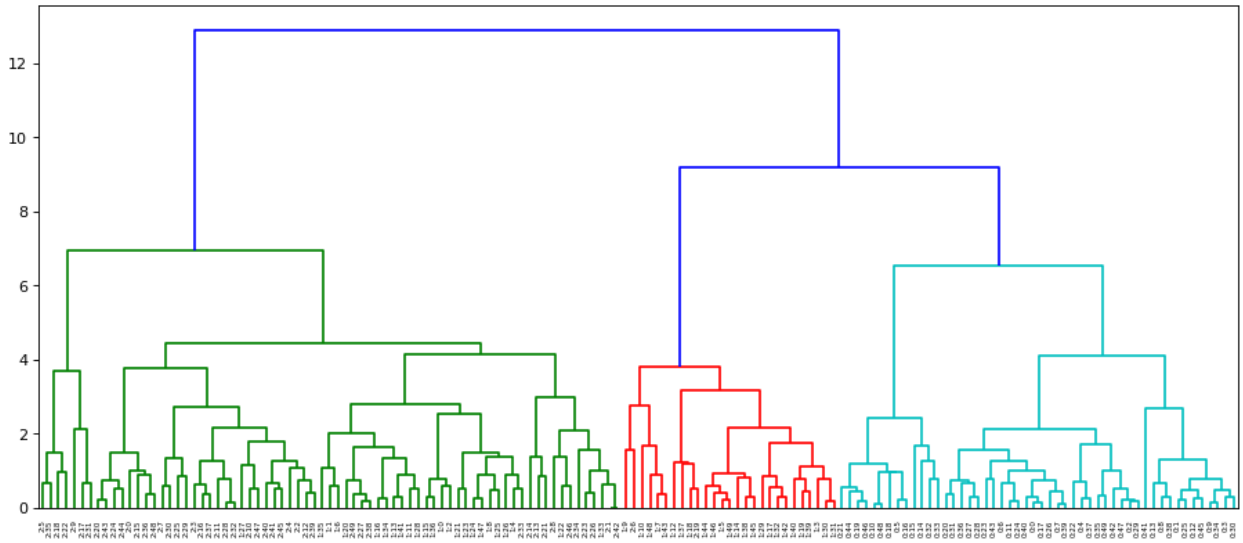
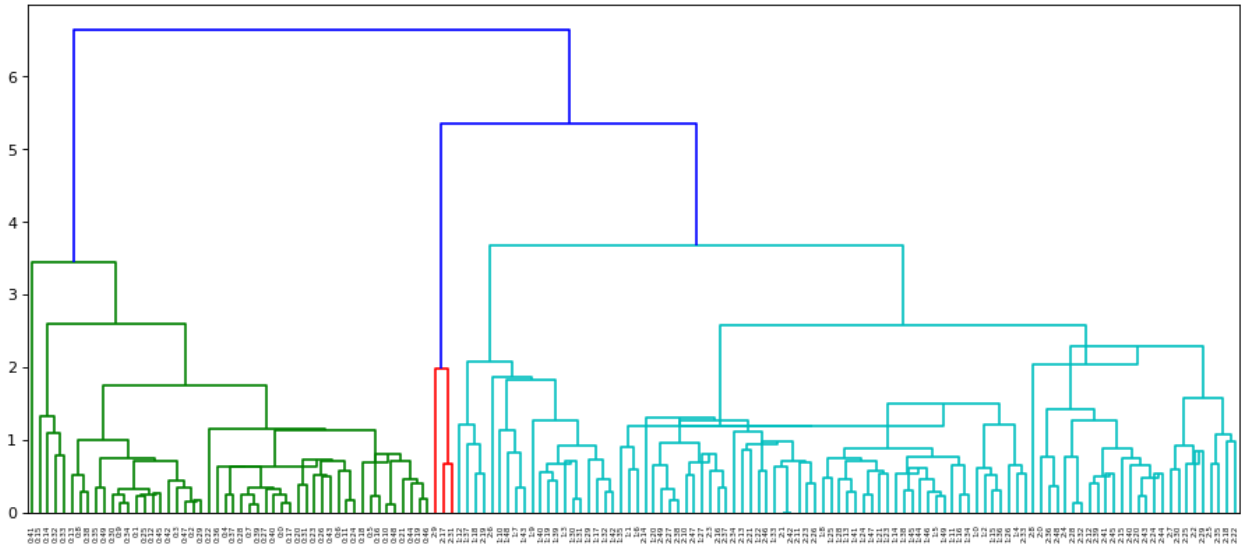



*Figure 1c.*
*Average-Manhattan, corresponding to Table 1*




All above results use default weights only. The following tables present the results for the same distance metrics and cluster similarity measures (again with normalizing the data), but with only taking one measurement into account at a time. These tables should provide additional clarity for the results of programmatic optimization of weights.

*Table 2a.*
*Normalized measures*

| | | Weights=[1,0,0,0] | | | Weights=[0,1,0,0] | | |
|---|---|---|---|---|---|---|---|
| | | Purity | RI | NMI | Purity | RI | NMI |
| Simple | Manhattan | 0.37333 | 0.35839 | 0.07206 | 0.34667 | 0.33772 | 0.02493 |
| | Vector | 0.37333 | 0.35839 | 0.07206 | 0.34667 | 0.33772 | 0.02493 |
| Complete | Manhattan | 0.62667 | 0.66523 | 0.37756 | 0.56 | 0.62819 | 0.23293 |
| | Vector | 0.62667 | 0.66523 | 0.37756 | 0.56 | 0.62819 | 0.23293 |
| Average | Manhattan | 0.67333 | 0.67857 | 0.34915 | 0.54 | 0.56644 | 0.22632 |
| | Vector | 0.67333 | 0.67857 | 0.34915 | 0.54 | 0.56644 | 0.22632 |

*Table 2b.*
*Normalized measures*

| | | Weights=[0,0,1,0] | | | Weights=[0,0,0,1] | | |
|---|---|---|---|---|---|---|---|
| | | Purity | RI | NMI | Purity | RI | NMI |
| Simple | Manhattan | 0.67333 | 0.77629 | 0.72348 | 0.66667 | 0.74327 | 0.73368 |
| | Vector | 0.67333 | 0.77629 | 0.72348 | 0.66667 | 0.74327 | 0.73368 |
| Complete | Manhattan | 0.89333 | 0.87973 | 0.79068 | 0.94667 | 0.93414 | 0.83222 |
| | Vector | 0.89333 | 0.87973 | 0.79068 | 0.94667 | 0.93414 | 0.83222 |
| Average | Manhattan | 0.86667 | 0.85682 | 0.76497 | 0.96 | 0.94953 | 0.87052 |
| | Vector | 0.86667 | 0.85682 | 0.76497 | 0.96 | 0.94953 | 0.87052 |

The following figures include the results for varying the weights of petal width and sepal width (corresponding to the right halves of Table 2a and Table 2b) using Manhattan distance and all cluster similarity measures. The width measures are kept here to represent the worst (least discriminative) and best (most discriminative) measures. The varying measure ranged from zero to nineteen, inclusive.

*Figure 2a.*

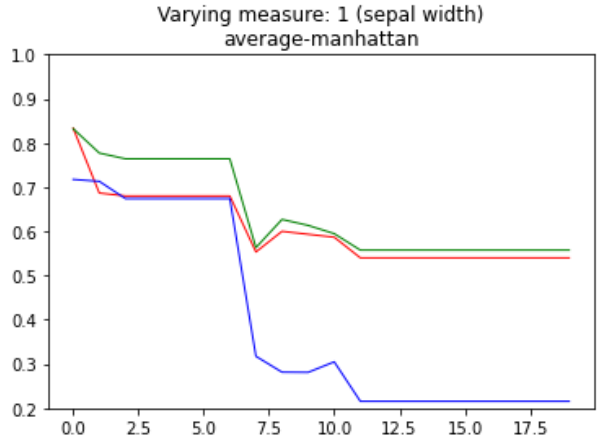



*Figure 2b.*

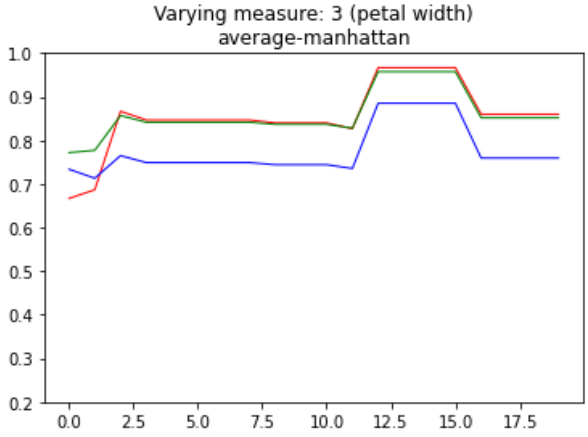



*Figure 2c.*

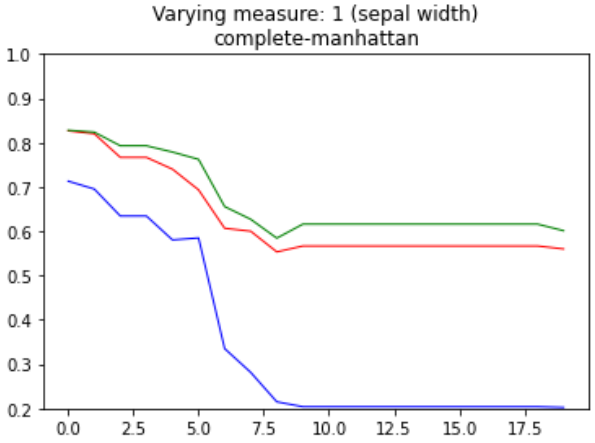



*Figure 2d.*

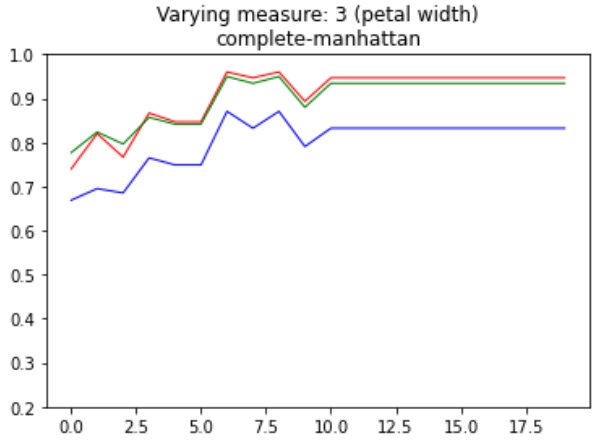



*Figure 2e.*

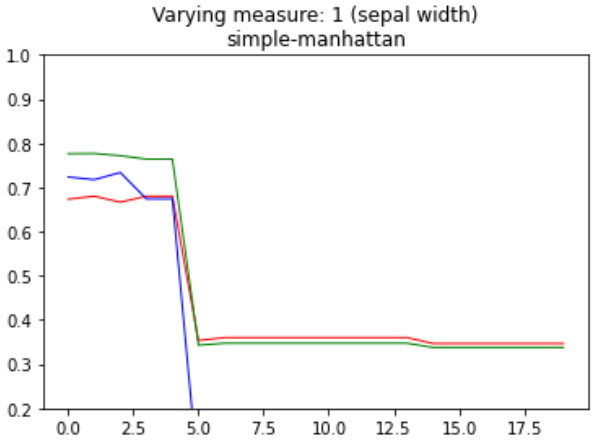
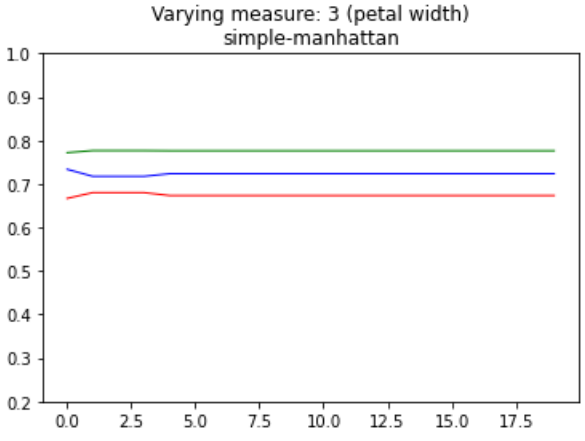



*Figure 2f.*




Key: purity RI NMI

Running the optimization method on all combinations of distance metrics, cluster similarity measures, and cluster evaluation method to optimize for resulted in the following tables: Table 3 with the actual values and Table 4 with the resulting weights. Each triple in Table 3 (the purity, RI, and NMI for each combination of distance metric, cluster similarity measure, and cluster evaluation method to optimize for) is the evaluation of clustering using the weights the weights in the same position of Table 4.

*Table 3.*

| | | Optimized for purity | | | Optimized for RI | | | Optimized for NMI | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Purity | RI | NMI | Purity | RI | NMI | Purity | RI | NMI |
| Simple | Manhattan | 0.68 | 0.77664 | 0.71746 | 0.68 | 0.77664 | 0.71746 | 0.66667 | 0.77181 | 0.73368 |
| | Vector | 0.67333 | 0.77629 | 0.72348 | 0.67333 | 0.77629 | 0.72348 | 0.66667 | 0.77181 | 0.73368 |
| Complete | Manhattan | 0.96 | 0.94953 | 0.86227 | 0.96 | 0.94953 | 0.86227 | 0.96667 | 0.95749 | 0.88506 |
| | Vector | 0.88 | 0.86792 | 0.75609 | 0.88 | 0.86792 | 0.75609 | 0.90667 | 0.89226 | 0.78567 |
| Average | Manhattan | 0.9 | 0.88591 | 0.79799 | 0.9 | 0.88591 | 0.79799 | 0.88667 | 0.87374 | 0.78374 |
| | Vector | 0.9 | 0.88591 | 0.79799 | 0.9 | 0.88591 | 0.79799 | 0.97333 | 0.96564 | 0.90112 |

*Table 4.*

| | | Optimized for purity | | | | Optimized for RI | | | | Optimized for NMI | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | sepal length | sepal width | petal length | petal width | sepal length | sepal width | petal length | petal width | sepal length | sepal width | petal length | petal width |
| Simple | Manhattan | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| | Vector | 1 | 0.25 | 1 | 1 | 1 | 0.25 | 1 | 1 | 1 | 1 | 1 | 1 |
| Complete | Manhattan | 1 | 1 | 2 | 2 | 1 | 1 | 2 | 2 | 0.5 | 1.25 | 2 | 2 |
| | Vector | 1 | 1 | 1 | 2 | 1 | 1 | 1 | 2 | 1 | 1 | 1.75 | 2.5 |
| Average | Manhattan | 1.5 | 0.75 | 1 | 1 | 1.25 | 0.75 | 1 | 1 | 1 | 1 | 1.5 | 1 |
| | Vector | 1 | 0.25 | 1.25 | 1 | 1 | 0.25 | 1.25 | 1 | 0 | 0 | 1 | 1 |

## Discussion

The clustering algorithm was verified against the Z matrix returned from `hierarchy.linkage`. Cluster evaluation algorithms were verified by using the same input from the "Evaluation of clustering" section from An Introduction to Information Retrieval (Manning, Raghavan, and Schütze, 2008). Correctness of the cluster choosing/hierarchy cutting algorithm was evaluated manually by examining the Z matrix and dendrogram to ensure that the clusters chosen were correct.

Table 1 is fairly straightforward but important in that it represents the primary goal of this project as well as how hierarchical clustering may compare to other classification methods used on the Iris dataset. However, since this method of clustering requires a specification of similarity (not all classification methods have this requirement), direct comparison may be difficult. It is also interesting to note that the data in this table introduces that NMI does not have a strictly positive relationship with

either RI or purity (as can be seen in both distance metrics for the complete measure of cluster similarity produce a higher RI and purity, but lower NMI than the simple and average methods).

Tables 2a and 2b exemplify that not all measures of each flower are as discriminative as others; we see specifically that using the length or width of the sepals alone results in very poor, and sometimes random, classification while using the length or width of the petals alone results in a much stronger classification, where most results are in fact greater than when using all measures equally.

Figures 2a-2f provide further evidence for the inequality of discriminatory capacity for each of the measures. The graphs also support the data presented in Tables 2a and 2b, that a increase in weighting on the petal width generally improves clustering, while an increasing in weighting on the sepal width generally worsens clustering. Further, we see again that NMI does not have a strictly positive relationship with either RI or purity, as seen in the figures which vary sepal width.

Table 3 provides the main results of running the greedy optimization method. In half the cases, optimizing for NMI produced a higher purity and RI when optimizing for either of those methods. In all cases, optimizing for purity and RI produced the same result and additionally took the same iterative steps to arrive there. That optimizing for NMI produced more optimal purity and RI in some cases merely speaks to the fact that this optimization method is greedy.

It would be interesting to see how each of the weights represented in Table 4 applied to the other combinations of distance metrics and cluster similarity methods, especially the more successful optimizations (Complete-Manhattan-NMI and Average-Vector-NMI). It would additionally be interesting to provide different starting points (instead of all weights being equal) that may be closer or farther away from a good weighting. Both of these things are possible with the code in the repository, but the presentation of this data in this report would be superfluous.

Improvements and future work mostly lay in the optimization method. For this project, a greedy method was chosen to save on resources and time, so there is no guarantee any local or global maximum is found (in fact, it's clear none have been found with this greedy method when comparing Table 3 with Table 2d). The lack of a true optimal value does definitely rely on the fact that the algorithm is greedy and not a product of a threshold that is too high; the steps taken for the greedy algorithm often ended when the best choice for the last iteration had the same cluster evaluation value as the previous. However, the algorithm could also benefit from a higher granularity in changing each step as well as possibly a greater range.

If a greedy method were chosen to be improved, improvements could include a change to more than just one value of the weights, including all combinations of pairs, or even all combinations. However, if a higher granularity in change values as well as a greater range were to be included, this would be so time intensive, it may be more beneficial to move on from using a greedy algorithm. For these reasons, it would be recommended to choose a different algorithm to optimize, which may include gradients or other standardized methods.

Additional improvements could include different distance metrics or cluster similarity measures, or a more focused set of them. In the above cases, the final weights returned from the optimization methods for purity and RI were the same in all cases. A further dive into how these evaluation methods (and others) correlate with each other could lead to a different set of measures that together cover a wider set of methods, for example, that if the weights produced from optimizing RI are always the same as optimizing for purity, then only one should be used in evaluation. However, these improvements are more for interestingness, since this project's primary goal was to cluster, and secondly to investigate how clustering could be optimized.

A final further improvement could include a different method for cutting the hierarchy. The way the hierarchy is cut in this project was very straightforward and simple, and, while intuitive, there may be other ways to improve clustering accuracy. While more complex tree cutting techniques tend to be more useful for much larger datasets (and often genomic ones), many suboptimal clustering results from hierarchical clustering on the Iris dataset include small nested clusters, which may benefit from more complex techniques, such as that developed by Langfelder, Zhang, and Horvath (2007), being able to find such clusters.

# References


Iris flower data set. (2020, December 02). Retrieved December 19, 2020, from https://en.wikipedia.org/wiki/Iris_flower_data_set

Jones, N. C., & Pevzner, P. (2004). Hierarchical Clustering. An Introduction to Bioinformatics Algorithms (pp. 343-346). Cambridge: MIT Press.

Langfelder, P., Zhang, B., & Horvath, S. (2007). Defining clusters from a hierarchical cluster tree: The Dynamic Tree Cut package for R [Abstract]. Bioinformatics, 24(5), 719-720. doi:10.1093/bioinformatics/btm563

Manning, C. D., Raghavan, P., & Schütze, H. (2008). Evaluation of clustering. An Introduction to information retrieval. Cambridge: Cambridge University Press. Retrieved December 19, 2020, from https://nlp.stanford.edu/IR-book/html/htmledition/evaluation-of-clustering-1.html