hierarchical-clustering

February 13, 2025

1 Hierarchical Clustering Demo

```
[1]: import numpy as np
     import matplotlib.pyplot as plt
     import seaborn as sns
```

```
[2]: np.random.seed(42)
```

1.1 Get the dataset to cluster

For demonstration purposes we will again generate random 2D points.

```
[3]: # Generate two globular clusters (25 points in each, 2 dimensions)
     SEPARATION = 3  # Experiment with this value to push the clusters closer or␣
      ↪further apart
     set_1 = np.random.randn(25, 2) + np.array([6 - SEPARATION, 6 - SEPARATION])
     set_2 = np.random.randn(25, 2) + np.array([6 + SEPARATION, 6 + SEPARATION])

     # Stack the sets into a single dataset
     data = np.vstack([set_1, set_2])
```

```
[4]: data
```

```
[4]: array([[ 3.49671415,  2.8617357 ],
            [ 3.64768854,  4.52302986],
            [ 2.76584663,  2.76586304],
            [ 4.57921282,  3.76743473],
            [ 2.53052561,  3.54256004],
            [ 2.53658231,  2.53427025],
            [ 3.24196227,  1.08671976],
            [ 1.27508217,  2.43771247],
            [ 1.98716888,  3.31424733],
            [ 2.09197592,  1.5876963 ],
            [ 4.46564877,  2.7742237 ],
            [ 3.0675282 ,  1.57525181],
            [ 2.45561728,  3.11092259],
            [ 1.84900642,  3.37569802],
            [ 2.39936131,  2.70830625],
            [ 2.39829339,  4.85227818],
            [ 2.98650278,  1.94228907],
            [ 3.82254491,  1.77915635],
            [ 3.2088636 ,  1.04032988],
            [ 1.67181395,  3.19686124],
            [ 3.73846658,  3.17136828],
            [ 2.88435172,  2.6988963 ],
            [ 1.52147801,  2.28015579],
            [ 2.53936123,  4.05712223],
            [ 3.34361829,  1.23695984],
            [ 9.32408397,  8.61491772],
            [ 8.323078  ,  9.61167629],
            [10.03099952,  9.93128012],
            [ 8.16078248,  8.69078762],
            [ 9.33126343,  9.97554513],
            [ 8.52082576,  8.81434102],
            [ 7.89366503,  7.80379338],
            [ 9.81252582, 10.35624003],
            [ 8.92798988, 10.0035329 ],
            [ 9.36163603,  8.35488025],
            [ 9.36139561, 10.53803657],
            [ 8.96417396, 10.56464366],
            [ 6.3802549 ,  9.8219025 ],
            [ 9.08704707,  8.70099265],
            [ 9.09176078,  7.01243109],
            [ 8.78032811,  9.35711257],
            [10.47789404,  8.48172978],
            [ 8.1915064 ,  8.49824296],
            [ 9.91540212,  9.32875111],
            [ 8.4702398 ,  9.51326743],
            [ 9.09707755,  9.96864499],
            [ 8.29794691,  8.67233785],
            [ 8.60789185,  7.53648505],
            [ 9.29612028,  9.26105527],
            [ 9.00511346,  8.76541287]])
```

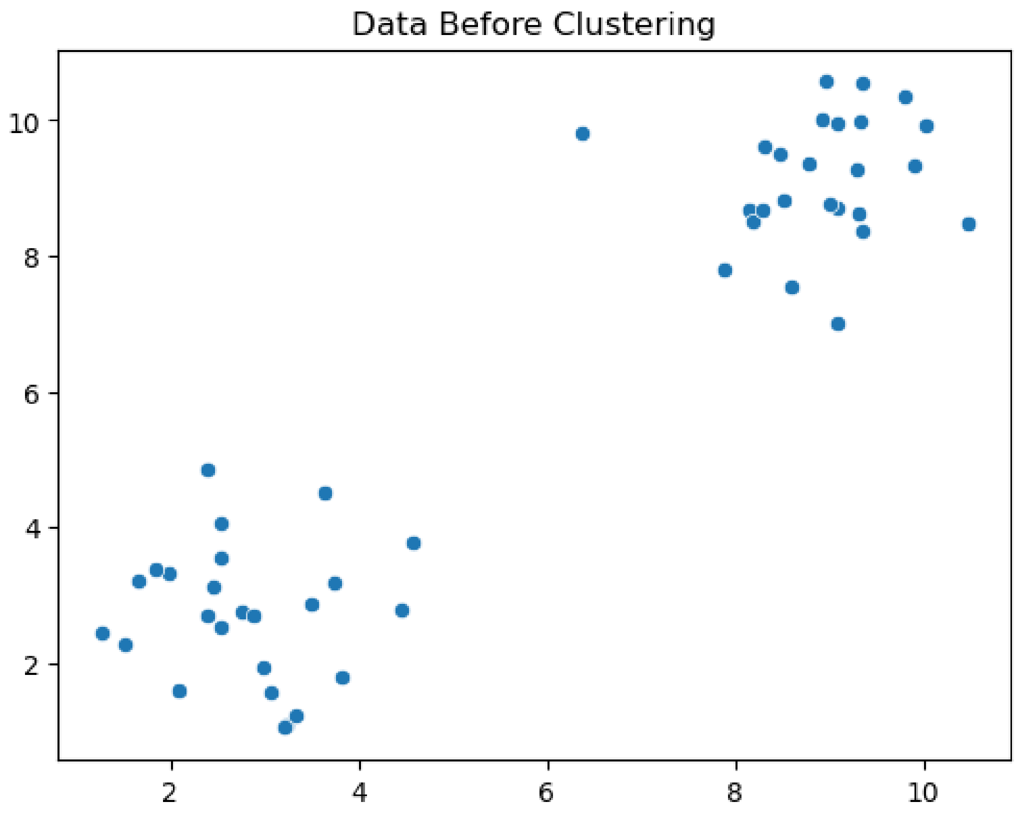
The dataset produced in the cell above contains two globular clusters. You can experiment with adding an additional linear cluster by uncommenting the code in the cell below.

[5]:
```
# # Generate an additional linear cluster
# x = np.linspace(0, 12, 25)  # Evenly spaced X values from 0 to 12 for 25␣
↪points
# y = 12 - x  # Linear Y values to create a line from (0,12) to (12,0)
# # Combine X and Y to form the linear cluster
# linear_cluster = np.column_stack([x, y])
# data = np.vstack([data, linear_cluster])
```

1.1.1 Visualize our data prior to clustering

```
[6]: sns.scatterplot(x=data[:, 0],  # data[:, 0] selects all the x-coordinates in␣
      ↪data
                     y=data[:, 1])  # data[:, 1] selects all the y-coordinates in␣
      ↪data
     plt.title('Data Before Clustering')
```

```
[6]: Text(0.5, 1.0, 'Data Before Clustering')
```



1.2 Perform the hierarchical clustering

Here we generate a linkage matrix using the linkage() function that we imported from SciPy above. Just as X often represents the input data to a machine learning algorithm (in this case the 2D coordinates of each data point) and y is used for the output (the actual or predicted values), Z is the variable name conventionally used to represent the linkage matrix.

In this example, clusters are formed by linking points to nearby points or previously generated clusters. This is known as agglomerative clustering. It is also possible to use divisive clustering where we start with the entire dataset and break it apart a step at a time.

With agglomerative clustering there are different ways to link the data points/previously generated clusters. Ward linkage is often the default choice (as indicated by method='ward' in the code below), but you can experiment with the other approaches by uncommenting the alternatives. You can read more about these methods in the documentation (https://docs.scipy.org/doc/scipy/reference/generated/scipy.cluster.hierarchy.linkage.html)

Each row of Z has the format [idx1, idx2, dist, sample_count], where: - idx1 and idx2 are the indexes of either the original observations (the data points) or previously formed clusters. - dist is the distance between these clusters. - sample_count is the number of original observations in the newly formed cluster.

[7]:
```
from scipy.cluster.hierarchy import linkage
```

[8]:
```
# Using the linkage function to perform hierarchical clustering
# 'ward' method minimizes the variance of clusters being merged
# Z = linkage(data, method=     )
# Z = linkage(data, method='average')
# Z = linkage(data, method='median')
# Z = linkage(data, method='complete')  # farthest point between clusters
# Z = linkage(data, method='single')  # nearest distance between clusters
# Z = linkage(data, method='weighted')
Z = linkage(data, method='ward')
Z
```

[8]:
```
array([[6.00000000e+00, 1.80000000e+01, 5.69872204e-02, 2.00000000e+00],
       [3.80000000e+01, 4.90000000e+01, 1.04226106e-01, 2.00000000e+00],
       [2.00000000e+00, 2.10000000e+01, 1.36117600e-01, 2.00000000e+00],
       [2.80000000e+01, 4.60000000e+01, 1.38399692e-01, 2.00000000e+00],
       [8.00000000e+00, 1.30000000e+01, 1.51211942e-01, 2.00000000e+00],
       [3.30000000e+01, 4.50000000e+01, 1.72649375e-01, 2.00000000e+00],
       [2.60000000e+01, 4.40000000e+01, 1.77033605e-01, 2.00000000e+00],
       [4.20000000e+01, 5.30000000e+01, 2.16146225e-01, 3.00000000e+00],
       [5.00000000e+00, 1.40000000e+01, 2.21626110e-01, 2.00000000e+00],
       [2.40000000e+01, 5.00000000e+01, 2.42355704e-01, 3.00000000e+00],
       [2.50000000e+01, 3.40000000e+01, 2.62734933e-01, 2.00000000e+00],
       [7.00000000e+00, 2.20000000e+01, 2.92463704e-01, 2.00000000e+00],
       [1.90000000e+01, 5.40000000e+01, 3.31838784e-01, 3.00000000e+00],
       [2.90000000e+01, 5.50000000e+01, 3.68238700e-01, 3.00000000e+00],
       [1.10000000e+01, 1.60000000e+01, 3.75874272e-01, 2.00000000e+00],
       [0.00000000e+00, 2.00000000e+01, 3.92831480e-01, 2.00000000e+00],
       [3.50000000e+01, 3.60000000e+01, 3.98111758e-01, 2.00000000e+00],
       [4.00000000e+00, 1.20000000e+01, 4.38089204e-01, 2.00000000e+00],
       [3.00000000e+01, 5.70000000e+01, 4.41683650e-01, 4.00000000e+00],
       [2.70000000e+01, 3.20000000e+01, 4.77830181e-01, 2.00000000e+00],
       [4.00000000e+01, 5.60000000e+01, 5.02493123e-01, 3.00000000e+00],
       [5.20000000e+01, 5.80000000e+01, 5.28925813e-01, 4.00000000e+00],
       [5.10000000e+01, 6.00000000e+01, 5.47234823e-01, 4.00000000e+00],
       [4.30000000e+01, 4.80000000e+01, 6.22970886e-01, 2.00000000e+00],
       [3.90000000e+01, 4.70000000e+01, 7.13275333e-01, 2.00000000e+00],
       [1.50000000e+01, 2.30000000e+01, 8.07572371e-01, 2.00000000e+00],
       [6.30000000e+01, 6.60000000e+01, 8.83761769e-01, 5.00000000e+00],
       [1.70000000e+01, 6.40000000e+01, 9.18899811e-01, 3.00000000e+00],
       [3.00000000e+00, 1.00000000e+01, 9.99682421e-01, 2.00000000e+00],
       [6.20000000e+01, 6.70000000e+01, 1.01907869e+00, 5.00000000e+00],
       [5.90000000e+01, 7.70000000e+01, 1.11684490e+00, 6.00000000e+00],
       [9.00000000e+00, 6.10000000e+01, 1.19778780e+00, 3.00000000e+00],
       [3.10000000e+01, 6.80000000e+01, 1.20514757e+00, 5.00000000e+00],
       [6.90000000e+01, 7.30000000e+01, 1.28094882e+00, 4.00000000e+00],
       [6.50000000e+01, 7.80000000e+01, 1.32920510e+00, 4.00000000e+00],
       [1.00000000e+00, 7.50000000e+01, 1.36351641e+00, 3.00000000e+00],
       [4.10000000e+01, 7.20000000e+01, 1.63138597e+00, 5.00000000e+00],
       [7.60000000e+01, 8.30000000e+01, 1.67920958e+00, 9.00000000e+00],
       [7.10000000e+01, 7.90000000e+01, 1.76185358e+00, 9.00000000e+00],
       [7.00000000e+01, 8.20000000e+01, 2.02484867e+00, 8.00000000e+00],
       [7.40000000e+01, 8.60000000e+01, 2.43510853e+00, 7.00000000e+00],
       [8.10000000e+01, 8.80000000e+01, 2.47826502e+00, 1.20000000e+01],
       [3.70000000e+01, 8.90000000e+01, 2.89258364e+00, 9.00000000e+00],
       [8.40000000e+01, 8.50000000e+01, 3.33233720e+00, 7.00000000e+00],
       [9.00000000e+01, 9.20000000e+01, 3.91598824e+00, 1.60000000e+01],
       [9.10000000e+01, 9.30000000e+01, 4.95066492e+00, 1.90000000e+01],
       [8.70000000e+01, 9.40000000e+01, 5.31597367e+00, 2.50000000e+01],
       [8.00000000e+01, 9.50000000e+01, 5.42256655e+00, 2.50000000e+01],
       [9.60000000e+01, 9.70000000e+01, 4.41600852e+01, 5.00000000e+01]])
```

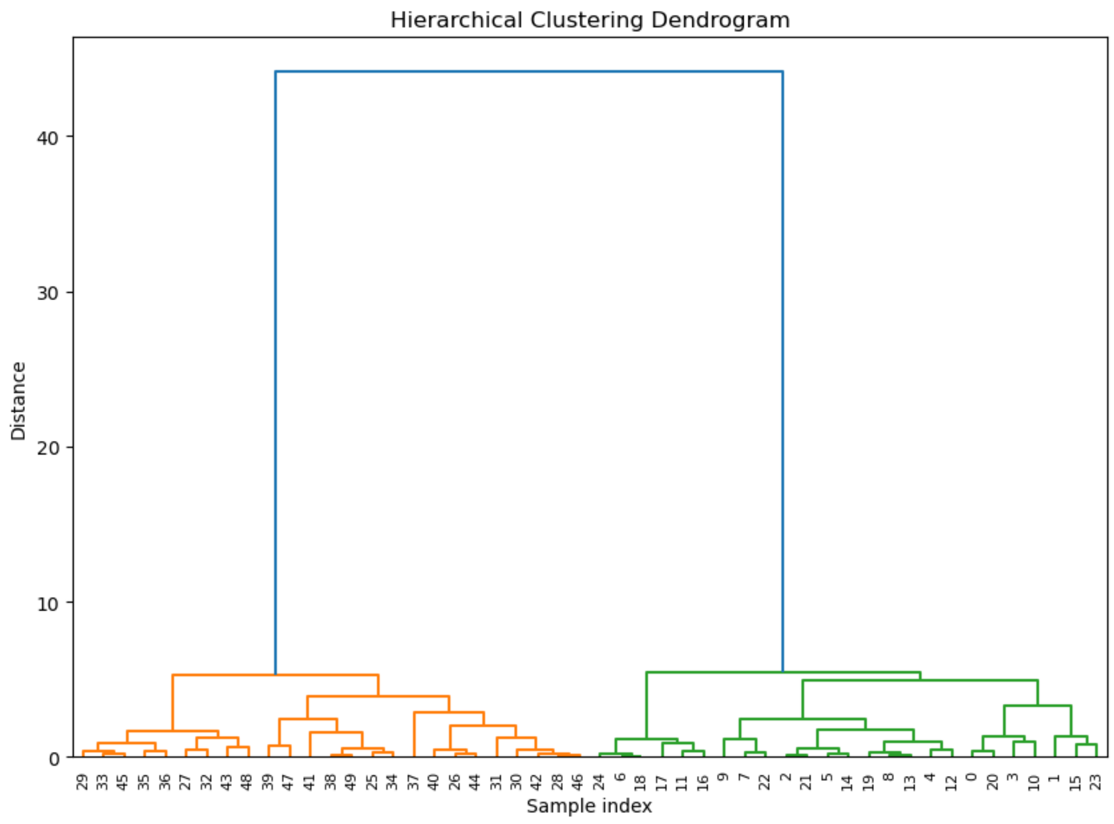
1.2.1 Plot the Dendrogram

A dendrogam shows how the data points have been hierarchically clustered. For example, in the diagram below we can see that (with method='ward') points 6 and 18 were very close together and have been merged into a cluster of two points. Point 24 was very close to that cluster and therefore was merged with 6 and 18 to form a cluster of three points. The dendrogram shows the whole structure.

[9]:
```python
from scipy.cluster.hierarchy import dendrogram
```

[10]:
```python
# Plotting the dendrogram
plt.figure(figsize=(10, 7))
dendrogram(Z, leaf_rotation=90)

plt.title("Hierarchical Clustering Dendrogram")
plt.xlabel("Sample index")
plt.ylabel("Distance")
```

[10]:
```
Text(0, 0.5, 'Distance')
```



1.3 Cut the Dendrogram to Create Clusters

Using the linkage matrix Z, we can use fcluster to cut the dendrogram if we want to get a desired number of clusters. We can experiment with the number of clusters to see what works well.

[11]:
```
from scipy.cluster.hierarchy import fcluster
```

[12]:
```
NUMBER_OF_CLUSTERS = 2
cluster_labels = fcluster(Z, t=NUMBER_OF_CLUSTERS, criterion='maxclust')
```

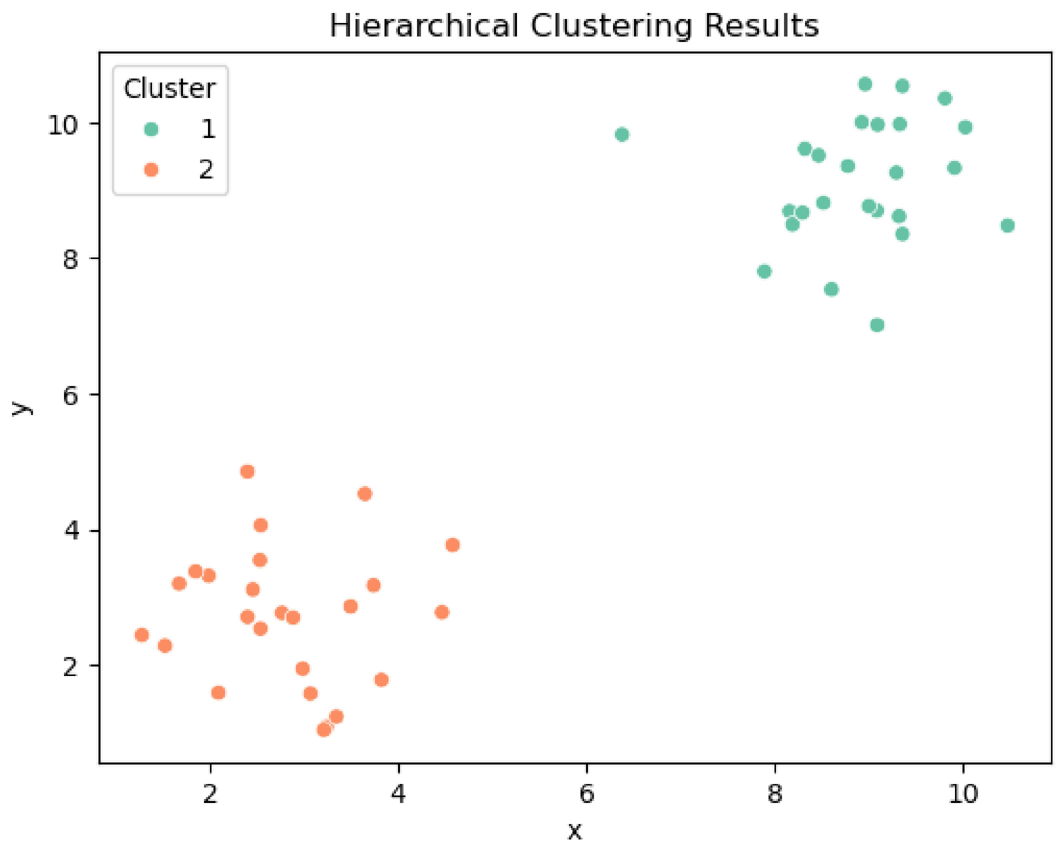
[13]:
```
cluster_labels
```

[13]:
```
array([2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1], dtype=int32)
```

Next we'll create another scatterplot of the data but we'll use the hue argument to highlight the different clusters.

[14]:
```
# Create a scatter plot
sns.scatterplot(x=data[:, 0],
                y=data[:, 1],
                hue=cluster_labels,
                palette='Set2')

plt.title("Hierarchical Clustering Results")
plt.xlabel("x")
plt.ylabel("y")

# Show legend with cluster labels
plt.legend(title='Cluster')
```

```
[14]: <matplotlib.legend.Legend at 0x24ffd229520>
```



2 Cluster the Well-Known Iris Dataset

This dataset is more often used for supervised learning - we predict species of iris based on petal length, petal width, sepal length, and sepal width. But we will use these features to cluster the data and then see if our clusters correspond to the species labels.

2.0.1 Load the Dataset

The iris dataset is very well known and is available from various sources including Kaggle. For convenience we'll load it from Scikit-learn in this demo.

[15]:
```
from sklearn.datasets import load_iris
```

[16]:
```
# I originally read the data from a CSV file that I downloaded from Kaggle.
# This included the species column.
# import pandas as pd
# iris_df = pd.read_csv('./data/iris.csv')

# To simplify the example, I modified the code to load the dataset from sklearn.
# This returns a tuple of the features and the target.
iris_df, species_array = load_iris(as_frame=True, return_X_y=True)
```

[17]:
```
iris_df.head()
```

[17]:
```
   sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)
0                5.1               3.5                1.4               0.2
1                4.9               3.0                1.4               0.2
2                4.7               3.2                1.3               0.2
3                4.6               3.1                1.5               0.2
4                5.0               3.6                1.4               0.2
```

[18]:
```
species_array
```

[18]:
```
0      0
1      0
2      0
3      0
4      0
      ..
145    2
146    2
147    2
148    2
149    2
Name: target, Length: 150, dtype: int32
```

2.0.2 Perform Hierarchical Clustering

[19]:
```
# Using only the feature columns
# X = iris_df.drop(columns=['species'])  # if your dataset contained the␣
 ↪species column
X = iris_df

X
```

[19]:
```
     sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)
0                  5.1               3.5                1.4               0.2
1                  4.9               3.0                1.4               0.2
2                  4.7               3.2                1.3               0.2
3                  4.6               3.1                1.5               0.2
4                  5.0               3.6                1.4               0.2
..                 …                ...                …               …
145                6.7               3.0                5.2               2.3
146                6.3               2.5                5.0               1.9
147                6.5               3.0                5.2               2.0
148                6.2               3.4                5.4               2.3
149                5.9               3.0                5.1               1.8

[150 rows x 4 columns]
```

[20]:
```python
# Use the 'ward' method again for hierarchical clustering
Z = linkage(X, 'ward')

Z
```

[20]:
```
array([[1.01000000e+02, 1.42000000e+02, 0.00000000e+00, 2.00000000e+00],
       [7.00000000e+00, 3.90000000e+01, 1.00000000e-01, 2.00000000e+00],
       [0.00000000e+00, 1.70000000e+01, 1.00000000e-01, 2.00000000e+00],
       [9.00000000e+00, 3.40000000e+01, 1.00000000e-01, 2.00000000e+00],
       [1.28000000e+02, 1.32000000e+02, 1.00000000e-01, 2.00000000e+00],
       [1.00000000e+01, 4.80000000e+01, 1.00000000e-01, 2.00000000e+00],
       [4.00000000e+00, 3.70000000e+01, 1.41421356e-01, 2.00000000e+00],
       [1.90000000e+01, 2.10000000e+01, 1.41421356e-01, 2.00000000e+00],
       [2.90000000e+01, 3.00000000e+01, 1.41421356e-01, 2.00000000e+00],
       [5.70000000e+01, 9.30000000e+01, 1.41421356e-01, 2.00000000e+00],
       [8.00000000e+01, 8.10000000e+01, 1.41421356e-01, 2.00000000e+00],
       [1.16000000e+02, 1.37000000e+02, 1.41421356e-01, 2.00000000e+00],
       [8.00000000e+00, 3.80000000e+01, 1.41421356e-01, 2.00000000e+00],
       [3.00000000e+00, 4.70000000e+01, 1.41421356e-01, 2.00000000e+00],
       [2.70000000e+01, 2.80000000e+01, 1.41421356e-01, 2.00000000e+00],
       [8.20000000e+01, 9.20000000e+01, 1.41421356e-01, 2.00000000e+00],
       [9.50000000e+01, 9.60000000e+01, 1.41421356e-01, 2.00000000e+00],
       [1.27000000e+02, 1.38000000e+02, 1.41421356e-01, 2.00000000e+00],
       [1.00000000e+00, 4.50000000e+01, 1.41421356e-01, 2.00000000e+00],
       [6.30000000e+01, 9.10000000e+01, 1.41421356e-01, 2.00000000e+00],
       [6.50000000e+01, 7.50000000e+01, 1.41421356e-01, 2.00000000e+00],
       [4.00000000e+01, 1.52000000e+02, 1.73205081e-01, 3.00000000e+00],
       [1.23000000e+02, 1.26000000e+02, 1.73205081e-01, 2.00000000e+00],
       [4.90000000e+01, 1.51000000e+02, 1.73205081e-01, 3.00000000e+00],
       [1.12000000e+02, 1.39000000e+02, 1.73205081e-01, 2.00000000e+00],
       [9.40000000e+01, 9.90000000e+01, 1.73205081e-01, 2.00000000e+00],
       [1.20000000e+01, 1.68000000e+02, 1.82574186e-01, 3.00000000e+00],
       [8.80000000e+01, 1.66000000e+02, 1.82574186e-01, 3.00000000e+00],
       [6.60000000e+01, 8.40000000e+01, 2.00000000e-01, 2.00000000e+00],
       [2.30000000e+01, 2.60000000e+01, 2.00000000e-01, 2.00000000e+00],
       [5.30000000e+01, 8.90000000e+01, 2.00000000e-01, 2.00000000e+00],
       [7.40000000e+01, 9.70000000e+01, 2.00000000e-01, 2.00000000e+00],
       [2.50000000e+01, 1.53000000e+02, 2.08166600e-01, 3.00000000e+00],
       [4.60000000e+01, 1.57000000e+02, 2.16024690e-01, 3.00000000e+00],
       [2.00000000e+00, 1.63000000e+02, 2.16024690e-01, 3.00000000e+00],
       [1.10000000e+02, 1.47000000e+02, 2.23606798e-01, 2.00000000e+00],
       [1.20000000e+02, 1.43000000e+02, 2.23606798e-01, 2.00000000e+00],
       [1.36000000e+02, 1.48000000e+02, 2.44948974e-01, 2.00000000e+00],
       [7.80000000e+01, 1.69000000e+02, 2.44948974e-01, 3.00000000e+00],
       [6.90000000e+01, 1.60000000e+02, 2.44948974e-01, 3.00000000e+00],
       [5.40000000e+01, 5.80000000e+01, 2.44948974e-01, 2.00000000e+00],
       [1.40000000e+02, 1.44000000e+02, 2.44948974e-01, 2.00000000e+00],
       [1.41000000e+02, 1.45000000e+02, 2.44948974e-01, 2.00000000e+00],
       [4.30000000e+01, 1.79000000e+02, 2.58198890e-01, 3.00000000e+00],
       [6.80000000e+01, 8.70000000e+01, 2.64575131e-01, 2.00000000e+00],
       [5.00000000e+01, 5.20000000e+01, 2.64575131e-01, 2.00000000e+00],
       [5.10000000e+01, 5.60000000e+01, 2.64575131e-01, 2.00000000e+00],
       [1.07000000e+02, 1.30000000e+02, 2.64575131e-01, 2.00000000e+00],
       [1.05000000e+02, 1.22000000e+02, 2.64575131e-01, 2.00000000e+00],
       [1.03000000e+02, 1.61000000e+02, 2.70801280e-01, 3.00000000e+00],
       [1.64000000e+02, 1.71000000e+02, 2.75680975e-01, 5.00000000e+00],
       [2.00000000e+01, 3.10000000e+01, 2.82842712e-01, 2.00000000e+00],
       [1.10000000e+01, 1.58000000e+02, 2.94392029e-01, 3.00000000e+00],
       [6.70000000e+01, 1.65000000e+02, 2.94392029e-01, 3.00000000e+00],
       [7.00000000e+01, 1.67000000e+02, 2.94392029e-01, 3.00000000e+00],
       [4.20000000e+01, 1.62000000e+02, 2.94392029e-01, 3.00000000e+00],
       [1.13000000e+02, 1.50000000e+02, 3.05505046e-01, 3.00000000e+00],
       [6.00000000e+00, 1.84000000e+02, 3.13581462e-01, 4.00000000e+00],
       [1.73000000e+02, 2.00000000e+02, 3.14642654e-01, 8.00000000e+00],
       [5.50000000e+01, 9.00000000e+01, 3.16227766e-01, 2.00000000e+00],
       [1.76000000e+02, 1.82000000e+02, 3.16227766e-01, 6.00000000e+00],
       [8.60000000e+01, 1.95000000e+02, 3.21455025e-01, 3.00000000e+00],
       [1.24000000e+02, 1.86000000e+02, 3.31662479e-01, 3.00000000e+00],
       [8.30000000e+01, 1.33000000e+02, 3.31662479e-01, 2.00000000e+00],
       [5.00000000e+00, 1.80000000e+01, 3.31662479e-01, 2.00000000e+00],
       [1.30000000e+01, 2.05000000e+02, 3.36650165e-01, 4.00000000e+00],
       [1.75000000e+02, 1.77000000e+02, 3.41565026e-01, 5.00000000e+00],
       [3.20000000e+01, 3.30000000e+01, 3.46410162e-01, 2.00000000e+00],
       [1.25000000e+02, 1.29000000e+02, 3.46410162e-01, 2.00000000e+00],
       [1.04000000e+02, 1.54000000e+02, 3.51188458e-01, 3.00000000e+00],
       [7.30000000e+01, 1.88000000e+02, 3.53553391e-01, 4.00000000e+00],
       [1.49000000e+02, 2.04000000e+02, 3.58236421e-01, 4.00000000e+00],
       [1.46000000e+02, 1.72000000e+02, 3.60555128e-01, 3.00000000e+00],
       [1.21000000e+02, 2.06000000e+02, 3.62859018e-01, 4.00000000e+00],
       [3.60000000e+01, 1.55000000e+02, 3.69684550e-01, 3.00000000e+00],
       [7.60000000e+01, 1.90000000e+02, 3.74165739e-01, 3.00000000e+00],
       [1.15000000e+02, 1.87000000e+02, 3.74165739e-01, 3.00000000e+00],
       [6.10000000e+01, 7.10000000e+01, 4.00000000e-01, 2.00000000e+00],
       [1.56000000e+02, 2.08000000e+02, 4.06201920e-01, 1.00000000e+01],
       [7.20000000e+01, 2.13000000e+02, 4.12310563e-01, 3.00000000e+00],
       [1.17000000e+02, 1.31000000e+02, 4.12310563e-01, 2.00000000e+00],
       [1.91000000e+02, 2.12000000e+02, 4.14728827e-01, 5.00000000e+00],
       [2.40000000e+01, 2.02000000e+02, 4.22295315e-01, 4.00000000e+00],
       [9.80000000e+01, 1.59000000e+02, 4.39696865e-01, 3.00000000e+00],
       [1.60000000e+01, 2.24000000e+02, 4.39696865e-01, 4.00000000e+00],
       [3.50000000e+01, 2.10000000e+02, 4.39696865e-01, 7.00000000e+00],
       [6.40000000e+01, 7.90000000e+01, 4.47213595e-01, 2.00000000e+00],
       [8.50000000e+01, 1.96000000e+02, 4.58257569e-01, 3.00000000e+00],
       [7.70000000e+01, 1.85000000e+02, 4.65474668e-01, 3.00000000e+00],
       [4.40000000e+01, 1.83000000e+02, 4.67261526e-01, 4.00000000e+00],
       [1.11000000e+02, 1.99000000e+02, 4.81317636e-01, 4.00000000e+00],
       [1.80000000e+02, 1.89000000e+02, 4.81663783e-01, 5.00000000e+00],
       [1.02000000e+02, 2.18000000e+02, 4.89897949e-01, 3.00000000e+00],
       [1.74000000e+02, 1.92000000e+02, 5.14781507e-01, 4.00000000e+00],
       [1.81000000e+02, 2.27000000e+02, 5.29150262e-01, 4.00000000e+00],
       [1.70000000e+02, 2.25000000e+02, 5.32916504e-01, 5.00000000e+00],
       [1.18000000e+02, 1.98000000e+02, 5.38516481e-01, 3.00000000e+00],
       [1.40000000e+01, 1.50000000e+01, 5.47722558e-01, 2.00000000e+00],
       [1.78000000e+02, 2.09000000e+02, 5.74456265e-01, 4.00000000e+00],
       [2.22000000e+02, 2.29000000e+02, 5.80229840e-01, 6.00000000e+00],
       [2.01000000e+02, 2.34000000e+02, 5.94418483e-01, 6.00000000e+00],
       [1.14000000e+02, 2.23000000e+02, 6.05805249e-01, 5.00000000e+00],
       [6.00000000e+01, 2.33000000e+02, 6.13731755e-01, 4.00000000e+00],
       [2.17000000e+02, 2.47000000e+02, 6.24499800e-01, 4.00000000e+00],
       [5.90000000e+01, 2.41000000e+02, 6.30872412e-01, 6.00000000e+00],
       [2.07000000e+02, 2.32000000e+02, 6.36396103e-01, 8.00000000e+00],
       [1.97000000e+02, 2.42000000e+02, 6.40312424e-01, 5.00000000e+00],
       [6.20000000e+01, 2.03000000e+02, 6.42910051e-01, 4.00000000e+00],
       [2.14000000e+02, 2.50000000e+02, 6.62696512e-01, 8.00000000e+00],
       [1.19000000e+02, 1.94000000e+02, 7.09459888e-01, 3.00000000e+00],
       [1.00000000e+02, 2.26000000e+02, 7.24568837e-01, 4.00000000e+00],
       [1.08000000e+02, 2.19000000e+02, 7.32575366e-01, 4.00000000e+00],
       [2.16000000e+02, 2.48000000e+02, 7.34090518e-01, 9.00000000e+00],
       [2.11000000e+02, 2.45000000e+02, 7.34960316e-01, 8.00000000e+00],
       [2.40000000e+02, 2.61000000e+02, 7.63216876e-01, 8.00000000e+00],
       [1.93000000e+02, 2.39000000e+02, 7.76285416e-01, 7.00000000e+00],
       [1.09000000e+02, 1.35000000e+02, 8.06225775e-01, 2.00000000e+00],
       [2.35000000e+02, 2.55000000e+02, 8.18738868e-01, 1.50000000e+01],
       [2.38000000e+02, 2.43000000e+02, 8.21873585e-01, 7.00000000e+00],
       [2.36000000e+02, 2.54000000e+02, 8.26135582e-01, 8.00000000e+00],
       [2.20000000e+01, 2.15000000e+02, 8.55569985e-01, 5.00000000e+00],
       [2.20000000e+02, 2.44000000e+02, 8.64580823e-01, 8.00000000e+00],
       [2.28000000e+02, 2.65000000e+02, 9.09981993e-01, 1.70000000e+01],
       [2.57000000e+02, 2.69000000e+02, 9.28708781e-01, 1.20000000e+01],
       [1.34000000e+02, 2.49000000e+02, 9.29157324e-01, 7.00000000e+00],
       [2.21000000e+02, 2.37000000e+02, 1.00534287e+00, 7.00000000e+00],
       [2.31000000e+02, 2.60000000e+02, 1.04705937e+00, 9.00000000e+00],
       [4.10000000e+01, 2.70000000e+02, 1.10513951e+00, 6.00000000e+00],
       [2.30000000e+02, 2.66000000e+02, 1.15325626e+00, 4.00000000e+00],
       [1.06000000e+02, 2.62000000e+02, 1.21700908e+00, 1.00000000e+01],
       [2.53000000e+02, 2.58000000e+02, 1.25399362e+00, 1.20000000e+01],
       [2.59000000e+02, 2.74000000e+02, 1.30486270e+00, 1.00000000e+01],
       [2.67000000e+02, 2.77000000e+02, 1.37126983e+00, 2.10000000e+01],
       [2.64000000e+02, 2.68000000e+02, 1.41139074e+00, 1.50000000e+01],
       [2.71000000e+02, 2.75000000e+02, 1.49547731e+00, 1.50000000e+01],
       [2.46000000e+02, 2.78000000e+02, 1.59776630e+00, 7.00000000e+00],
       [2.51000000e+02, 2.81000000e+02, 1.75916647e+00, 1.50000000e+01],
       [2.76000000e+02, 2.83000000e+02, 1.76044502e+00, 2.40000000e+01],
       [2.56000000e+02, 2.85000000e+02, 1.84584475e+00, 1.20000000e+01],
       [2.73000000e+02, 2.79000000e+02, 1.91608028e+00, 2.20000000e+01],
       [2.72000000e+02, 2.80000000e+02, 1.91875287e+00, 2.90000000e+01],
       [2.63000000e+02, 2.84000000e+02, 2.05363058e+00, 2.30000000e+01],
       [2.52000000e+02, 2.89000000e+02, 2.81393883e+00, 2.60000000e+01],
       [2.86000000e+02, 2.91000000e+02, 2.86941764e+00, 3.80000000e+01],
       [2.82000000e+02, 2.90000000e+02, 3.82805262e+00, 5.00000000e+01],
       [2.87000000e+02, 2.88000000e+02, 4.84770851e+00, 3.60000000e+01],
       [2.92000000e+02, 2.93000000e+02, 6.39940682e+00, 6.40000000e+01],
       [2.95000000e+02, 2.96000000e+02, 1.23003961e+01, 1.00000000e+02],
       [2.94000000e+02, 2.97000000e+02, 3.24476070e+01, 1.50000000e+02]])
```

2.0.3 Extract and Prepare Species Labels for Dendrogram

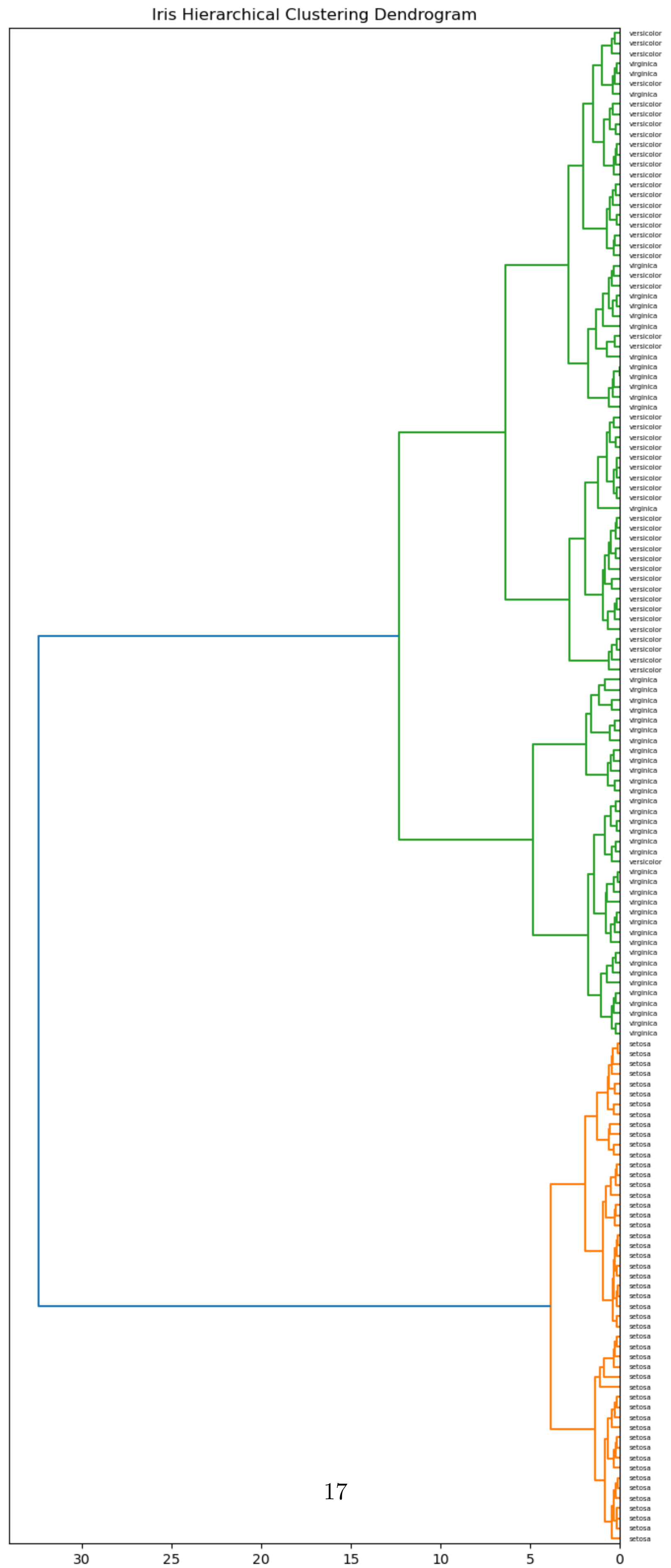
[21]:
```
# This line gets an array of species labels from my original dataset from Kaggle
#species_labels = iris_df['species'].values

# Here we create a list of species labels from the version of the data from␣
 ↪sklearn
species = ['setosa', 'versicolor', 'virginica']
species_labels = [species[i] for i in species_array]
species_labels
```

[21]:
```
['setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'setosa',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'versicolor',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica',
 'virginica']
```

2.0.4 Display Dendrogram with Species Labels

[22]:
```
plt.figure(figsize=(8, 20))
dendrogram(Z,
           labels=species_labels,
           orientation='left')
plt.title("Iris Hierarchical Clustering Dendrogram")
```

[22]:
```
Text(0.5, 1.0, 'Iris Hierarchical Clustering Dendrogram')
```


Iris Hierarchical Clustering Dendrogram
versicolor
versicolor
versicolor
virginica
virginica
versicolor
virginica
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
virginica
versicolor
versicolor
virginica
virginica
virginica
virginica
versicolor
versicolor
virginica
virginica
virginica
virginica
virginica
virginica
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
virginica
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
versicolor
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
30
25
20
15
10
5
0

2.1 Try to achieve better separation of the species

Looking at the dendrogram above, iris-setosa looks well separated, but iris-virginica and iris-versicolor look a bit more mixed. We will try to achieve better separation by normalizing. The goal is to ensure that each feature contributes equally to the distance computations used in clustering. Without normalization, features with larger numerical ranges can disproportionately influence the distance calculations.

2.1.1 Standardization vs. Normalization

Standardization (Z-score normalization) scales the features so they have the properties of a standard normal distribution with a mean of 0 and a standard deviation of 1.

Normalization (Min-Max scaling) scales and shifts the features so that they range from 0 to 1. You can also scale them to any other specific range. Both methods can be effective, but standardization is less affected by outliers and is generally preferred for clustering.

We will use Scikit-Learn to perform standardization and then see if that gives a better clustering.

```
[23]: from sklearn.preprocessing import StandardScaler
```

2.1.2 Apply standardization

```
[24]: scaler = StandardScaler()
      X_scaled = scaler.fit_transform(X)
```

```
[25]: X_scaled
```

```
[25]: array([[-9.00681170e-01,  1.01900435e+00, -1.34022653e+00,
         -1.31544430e+00],
        [-1.14301691e+00, -1.31979479e-01, -1.34022653e+00,
         -1.31544430e+00],
        [-1.38535265e+00,  3.28414053e-01, -1.39706395e+00,
         -1.31544430e+00],
        [-1.50652052e+00,  9.82172869e-02, -1.28338910e+00,
         -1.31544430e+00],
        [-1.02184904e+00,  1.24920112e+00, -1.34022653e+00,
         -1.31544430e+00],
        [-5.37177559e-01,  1.93979142e+00, -1.16971425e+00,
         -1.05217993e+00],
        [-1.50652052e+00,  7.88807586e-01, -1.34022653e+00,
         -1.18381211e+00],
        [-1.02184904e+00,  7.88807586e-01, -1.28338910e+00,
         -1.31544430e+00],
        [-1.74885626e+00, -3.62176246e-01, -1.34022653e+00,
         -1.31544430e+00],
        [-1.14301691e+00,  9.82172869e-02, -1.28338910e+00,
         -1.44707648e+00],
        [-5.37177559e-01,  1.47939788e+00, -1.28338910e+00,
         -1.31544430e+00],
        [-1.26418478e+00,  7.88807586e-01, -1.22655167e+00,
         -1.31544430e+00],
        [-1.26418478e+00, -1.31979479e-01, -1.34022653e+00,
         -1.44707648e+00],
        [-1.87002413e+00, -1.31979479e-01, -1.51073881e+00,
         -1.44707648e+00],
        [-5.25060772e-02,  2.16998818e+00, -1.45390138e+00,
         -1.31544430e+00],
        [-1.73673948e-01,  3.09077525e+00, -1.28338910e+00,
         -1.05217993e+00],
        [-5.37177559e-01,  1.93979142e+00, -1.39706395e+00,
         -1.05217993e+00],
        [-9.00681170e-01,  1.01900435e+00, -1.34022653e+00,
         -1.18381211e+00],
        [-1.73673948e-01,  1.70959465e+00, -1.16971425e+00,
         -1.18381211e+00],
        [-9.00681170e-01,  1.70959465e+00, -1.28338910e+00,
         -1.18381211e+00],
        [-5.37177559e-01,  7.88807586e-01, -1.16971425e+00,
         -1.31544430e+00],
        [-9.00681170e-01,  1.47939788e+00, -1.28338910e+00,
         -1.05217993e+00],
        [-1.50652052e+00,  1.24920112e+00, -1.56757623e+00,
         -1.31544430e+00],
        [-9.00681170e-01,  5.58610819e-01, -1.16971425e+00,
         -9.20547742e-01],
        [-1.26418478e+00,  7.88807586e-01, -1.05603939e+00,
         -1.31544430e+00],
        [-1.02184904e+00, -1.31979479e-01, -1.22655167e+00,
         -1.31544430e+00],
        [-1.02184904e+00,  7.88807586e-01, -1.22655167e+00,
         -1.05217993e+00],
        [-7.79513300e-01,  1.01900435e+00, -1.28338910e+00,
         -1.31544430e+00],
        [-7.79513300e-01,  7.88807586e-01, -1.34022653e+00,
         -1.31544430e+00],
        [-1.38535265e+00,  3.28414053e-01, -1.22655167e+00,
         -1.31544430e+00],
        [-1.26418478e+00,  9.82172869e-02, -1.22655167e+00,
         -1.31544430e+00],
        [-5.37177559e-01,  7.88807586e-01, -1.28338910e+00,
         -1.05217993e+00],
        [-7.79513300e-01,  2.40018495e+00, -1.28338910e+00,
         -1.44707648e+00],
        [-4.16009689e-01,  2.63038172e+00, -1.34022653e+00,
         -1.31544430e+00],
        [-1.14301691e+00,  9.82172869e-02, -1.28338910e+00,
         -1.31544430e+00],
        [-1.02184904e+00,  3.28414053e-01, -1.45390138e+00,
         -1.31544430e+00],
        [-4.16009689e-01,  1.01900435e+00, -1.39706395e+00,
         -1.31544430e+00],
        [-1.14301691e+00,  1.24920112e+00, -1.34022653e+00,
         -1.44707648e+00],
        [-1.74885626e+00, -1.31979479e-01, -1.39706395e+00,
         -1.31544430e+00],
        [-9.00681170e-01,  7.88807586e-01, -1.28338910e+00,
         -1.31544430e+00],
        [-1.02184904e+00,  1.01900435e+00, -1.39706395e+00,
         -1.18381211e+00],
        [-1.62768839e+00, -1.74335684e+00, -1.39706395e+00,
         -1.18381211e+00],
        [-1.74885626e+00,  3.28414053e-01, -1.39706395e+00,
         -1.31544430e+00],
        [-1.02184904e+00,  1.01900435e+00, -1.22655167e+00,
         -7.88915558e-01],
        [-9.00681170e-01,  1.70959465e+00, -1.05603939e+00,
         -1.05217993e+00],
        [-1.26418478e+00, -1.31979479e-01, -1.34022653e+00,
         -1.18381211e+00],
        [-9.00681170e-01,  1.70959465e+00, -1.22655167e+00,
         -1.31544430e+00],
        [-1.50652052e+00,  3.28414053e-01, -1.34022653e+00,
         -1.31544430e+00],
        [-6.58345429e-01,  1.47939788e+00, -1.28338910e+00,
         -1.31544430e+00],
        [-1.02184904e+00,  5.58610819e-01, -1.34022653e+00,
         -1.31544430e+00],
        [ 1.40150837e+00,  3.28414053e-01,  5.35408562e-01,
          2.64141916e-01],
        [ 6.74501145e-01,  3.28414053e-01,  4.21733708e-01,
          3.95774101e-01],
        [ 1.28034050e+00,  9.82172869e-02,  6.49083415e-01,
          3.95774101e-01],
        [-4.16009689e-01, -1.74335684e+00,  1.37546573e-01,
          1.32509732e-01],
        [ 7.95669016e-01, -5.92373012e-01,  4.78571135e-01,
          3.95774101e-01],
        [-1.73673948e-01, -5.92373012e-01,  4.21733708e-01,
          1.32509732e-01],
        [ 5.53333275e-01,  5.58610819e-01,  5.35408562e-01,
          5.27406285e-01],
        [-1.14301691e+00, -1.51316008e+00, -2.60315415e-01,
         -2.62386821e-01],
        [ 9.16836886e-01, -3.62176246e-01,  4.78571135e-01,
          1.32509732e-01],
        [-7.79513300e-01, -8.22569778e-01,  8.07091462e-02,
          2.64141916e-01],
        [-1.02184904e+00, -2.43394714e+00, -1.46640561e-01,
         -2.62386821e-01],
        [ 6.86617933e-02, -1.31979479e-01,  2.51221427e-01,
          3.95774101e-01],
        [ 1.89829664e-01, -1.97355361e+00,  1.37546573e-01,
         -2.62386821e-01],
        [ 3.10997534e-01, -3.62176246e-01,  5.35408562e-01,
          2.64141916e-01],
        [-2.94841818e-01, -3.62176246e-01, -8.98031345e-02,
          1.32509732e-01],
        [ 1.03800476e+00,  9.82172869e-02,  3.64896281e-01,
          2.64141916e-01],
        [-2.94841818e-01, -1.31979479e-01,  4.21733708e-01,
          3.95774101e-01],
        [-5.25060772e-02, -8.22569778e-01,  1.94384000e-01,
         -2.62386821e-01],
        [ 4.32165405e-01, -1.97355361e+00,  4.21733708e-01,
          3.95774101e-01],
        [-2.94841818e-01, -1.28296331e+00,  8.07091462e-02,
         -1.30754636e-01],
        [ 6.86617933e-02,  3.28414053e-01,  5.92245988e-01,
          7.90670654e-01],
        [ 3.10997534e-01, -5.92373012e-01,  1.37546573e-01,
          1.32509732e-01],
        [ 5.53333275e-01, -1.28296331e+00,  6.49083415e-01,
          3.95774101e-01],
        [ 3.10997534e-01, -5.92373012e-01,  5.35408562e-01,
          8.77547895e-04],
        [ 6.74501145e-01, -3.62176246e-01,  3.08058854e-01,
          1.32509732e-01],
        [ 9.16836886e-01, -1.31979479e-01,  3.64896281e-01,
          2.64141916e-01],
        [ 1.15917263e+00, -5.92373012e-01,  5.92245988e-01,
          2.64141916e-01],
        [ 1.03800476e+00, -1.31979479e-01,  7.05920842e-01,
          6.59038469e-01],
        [ 1.89829664e-01, -3.62176246e-01,  4.21733708e-01,
          3.95774101e-01],
        [-1.73673948e-01, -1.05276654e+00, -1.46640561e-01,
         -2.62386821e-01],
        [-4.16009689e-01, -1.51316008e+00,  2.38717193e-02,
         -1.30754636e-01],
        [-4.16009689e-01, -1.51316008e+00, -3.29657076e-02,
         -2.62386821e-01],
        [-5.25060772e-02, -8.22569778e-01,  8.07091462e-02,
          8.77547895e-04],
        [ 1.89829664e-01, -8.22569778e-01,  7.62758269e-01,
          5.27406285e-01],
        [-5.37177559e-01, -1.31979479e-01,  4.21733708e-01,
          3.95774101e-01],
        [ 1.89829664e-01,  7.88807586e-01,  4.21733708e-01,
          5.27406285e-01],
        [ 1.03800476e+00,  9.82172869e-02,  5.35408562e-01,
          3.95774101e-01],
        [ 5.53333275e-01, -1.74335684e+00,  3.64896281e-01,
          1.32509732e-01],
        [-2.94841818e-01, -1.31979479e-01,  1.94384000e-01,
          1.32509732e-01],
        [-4.16009689e-01, -1.28296331e+00,  1.37546573e-01,
          1.32509732e-01],
        [-4.16009689e-01, -1.05276654e+00,  3.64896281e-01,
          8.77547895e-04],
        [ 3.10997534e-01, -1.31979479e-01,  4.78571135e-01,
          2.64141916e-01],
        [-5.25060772e-02, -1.05276654e+00,  1.37546573e-01,
          8.77547895e-04],
        [-1.02184904e+00, -1.74335684e+00, -2.60315415e-01,
         -2.62386821e-01],
        [-2.94841818e-01, -8.22569778e-01,  2.51221427e-01,
          1.32509732e-01],
        [-1.73673948e-01, -1.31979479e-01,  2.51221427e-01,
          8.77547895e-04],
        [-1.73673948e-01, -3.62176246e-01,  2.51221427e-01,
          1.32509732e-01],
        [ 4.32165405e-01, -3.62176246e-01,  3.08058854e-01,
          1.32509732e-01],
        [-9.00681170e-01, -1.28296331e+00, -4.30827696e-01,
         -1.30754636e-01],
        [-1.73673948e-01, -5.92373012e-01,  1.94384000e-01,
          1.32509732e-01],
        [ 5.53333275e-01,  5.58610819e-01,  1.27429511e+00,
          1.71209594e+00],
        [-5.25060772e-02, -8.22569778e-01,  7.62758269e-01,
          9.22302838e-01],
        [ 1.52267624e+00, -1.31979479e-01,  1.21745768e+00,
          1.18556721e+00],
        [ 5.53333275e-01, -3.62176246e-01,  1.04694540e+00,
          7.90670654e-01],
        [ 7.95669016e-01, -1.31979479e-01,  1.16062026e+00,
          1.31719939e+00],
        [ 2.12851559e+00, -1.31979479e-01,  1.61531967e+00,
          1.18556721e+00],
        [-1.14301691e+00, -1.28296331e+00,  4.21733708e-01,
          6.59038469e-01],
        [ 1.76501198e+00, -3.62176246e-01,  1.44480739e+00,
          7.90670654e-01],
        [ 1.03800476e+00, -1.28296331e+00,  1.16062026e+00,
          7.90670654e-01],
        [ 1.64384411e+00,  1.24920112e+00,  1.33113254e+00,
          1.71209594e+00],
        [ 7.95669016e-01,  3.28414053e-01,  7.62758269e-01,
          1.05393502e+00],
        [ 6.74501145e-01, -8.22569778e-01,  8.76433123e-01,
          9.22302838e-01],
        [ 1.15917263e+00, -1.31979479e-01,  9.90107977e-01,
          1.18556721e+00],
        [-1.73673948e-01, -1.28296331e+00,  7.05920842e-01,
          1.05393502e+00],
        [-5.25060772e-02, -5.92373012e-01,  7.62758269e-01,
          1.58046376e+00],
        [ 6.74501145e-01,  3.28414053e-01,  8.76433123e-01,
          1.44883158e+00],
        [ 7.95669016e-01, -1.31979479e-01,  9.90107977e-01,
          7.90670654e-01],
        [ 2.24968346e+00,  1.70959465e+00,  1.67215710e+00,
          1.31719939e+00],
        [ 2.24968346e+00, -1.05276654e+00,  1.78583195e+00,
          1.44883158e+00],
        [ 1.89829664e-01, -1.97355361e+00,  7.05920842e-01,
          3.95774101e-01],
        [ 1.28034050e+00,  3.28414053e-01,  1.10378283e+00,
          1.44883158e+00],
        [-2.94841818e-01, -5.92373012e-01,  6.49083415e-01,
          1.05393502e+00],
        [ 2.24968346e+00, -5.92373012e-01,  1.67215710e+00,
          1.05393502e+00],
        [ 5.53333275e-01, -8.22569778e-01,  6.49083415e-01,
          7.90670654e-01],
        [ 1.03800476e+00,  5.58610819e-01,  1.10378283e+00,
          1.18556721e+00],
        [ 1.64384411e+00,  3.28414053e-01,  1.27429511e+00,
          7.90670654e-01],
        [ 4.32165405e-01, -5.92373012e-01,  5.92245988e-01,
          7.90670654e-01],
        [ 3.10997534e-01, -1.31979479e-01,  6.49083415e-01,
          7.90670654e-01],
        [ 6.74501145e-01, -5.92373012e-01,  1.04694540e+00,
          1.18556721e+00],
        [ 1.64384411e+00, -1.31979479e-01,  1.16062026e+00,
          5.27406285e-01],
        [ 1.88617985e+00, -5.92373012e-01,  1.33113254e+00,
          9.22302838e-01],
        [ 2.49201920e+00,  1.70959465e+00,  1.50164482e+00,
          1.05393502e+00],
        [ 6.74501145e-01, -5.92373012e-01,  1.04694540e+00,
          1.31719939e+00],
        [ 5.53333275e-01, -5.92373012e-01,  7.62758269e-01,
          3.95774101e-01],
        [ 3.10997534e-01, -1.05276654e+00,  1.04694540e+00,
          2.64141916e-01],
        [ 2.24968346e+00, -1.31979479e-01,  1.33113254e+00,
          1.44883158e+00],
        [ 5.53333275e-01,  7.88807586e-01,  1.04694540e+00,
          1.58046376e+00],
        [ 6.74501145e-01,  9.82172869e-02,  9.90107977e-01,
          7.90670654e-01],
        [ 1.89829664e-01, -1.31979479e-01,  5.92245988e-01,
          7.90670654e-01],
        [ 1.28034050e+00,  9.82172869e-02,  9.33270550e-01,
          1.18556721e+00],
        [ 1.03800476e+00,  9.82172869e-02,  1.04694540e+00,
          1.58046376e+00],
        [ 1.28034050e+00,  9.82172869e-02,  7.62758269e-01,
          1.44883158e+00],
        [-5.25060772e-02, -8.22569778e-01,  7.62758269e-01,
          9.22302838e-01],
        [ 1.15917263e+00,  3.28414053e-01,  1.21745768e+00,
          1.44883158e+00],
        [ 1.03800476e+00,  5.58610819e-01,  1.10378283e+00,
          1.71209594e+00],
        [ 1.03800476e+00, -1.31979479e-01,  8.19595696e-01,
          1.44883158e+00],
        [ 5.53333275e-01, -1.28296331e+00,  7.05920842e-01,
          9.22302838e-01],
        [ 7.95669016e-01, -1.31979479e-01,  8.19595696e-01,
          1.05393502e+00],
        [ 4.32165405e-01,  7.88807586e-01,  9.33270550e-01,
          1.44883158e+00],
        [ 6.86617933e-02, -1.31979479e-01,  7.62758269e-01,
          7.90670654e-01]])
```

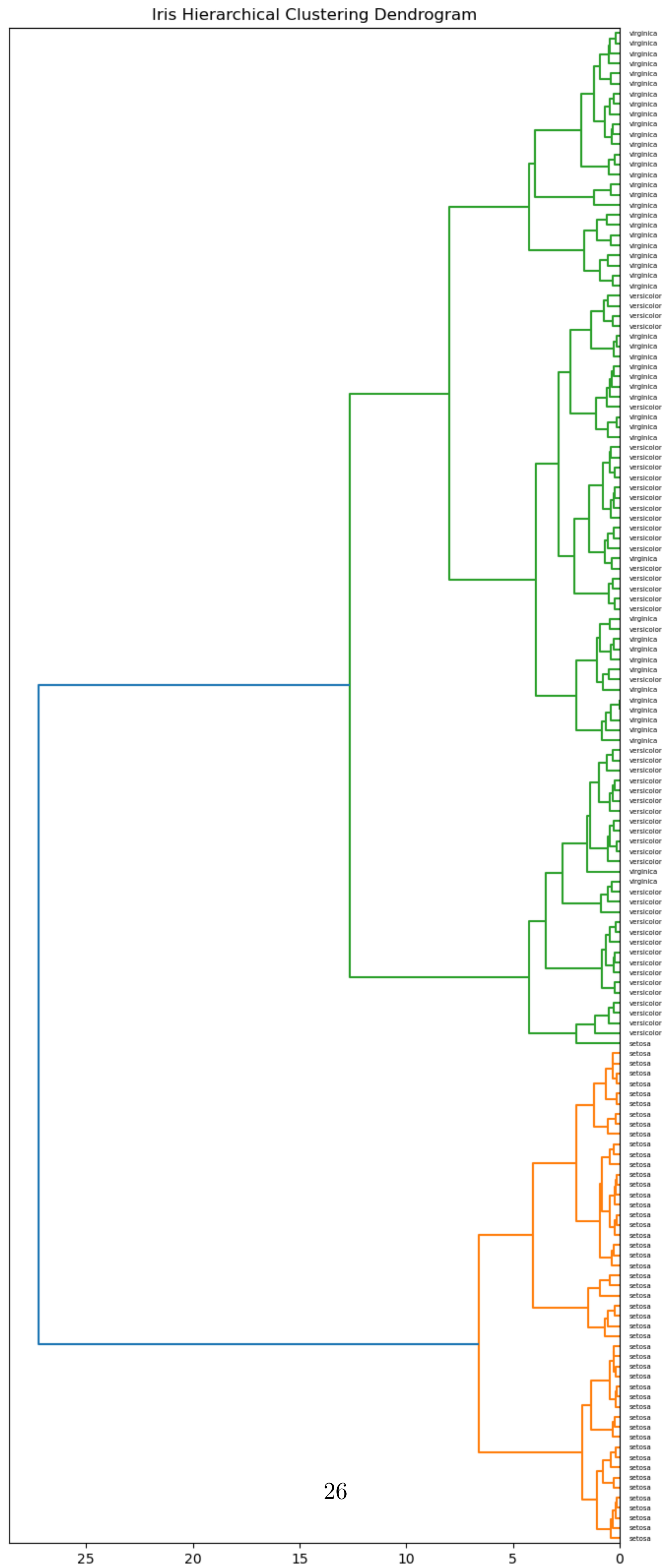
2.1.3 Perform hierarchical clustering on the scaled data

[26]:
```
Z_scaled = linkage(X_scaled, method='ward')
# Z_scaled = linkage(X_scaled, method='complete')
# Z_scaled = linkage(X_scaled, method='centroid')
# Z_scaled = linkage(X_scaled, method='average')
# Z_scaled = linkage(X_scaled, method='median')
# Z_scaled = linkage(X_scaled, method='single')
# Z_scaled = linkage(X_scaled, method='weighted')
```

2.1.4 Plot the dendrogram

[27]:
```
plt.figure(figsize=(8, 20))
dendrogram(Z_scaled,
           labels=species_labels,
           orientation='left')
plt.title("Iris Hierarchical Clustering Dendrogram")
```

[27]:
```
Text(0.5, 1.0, 'Iris Hierarchical Clustering Dendrogram')
```


Iris Hierarchical Clustering Dendrogram
virginica
versicolor
setosa
25
20
15
10
5
0

Did we achieve a better clustering that time? It's hard to tell just by eye-balling the dendrogram, but it doesn't look better to me. We can use the adjusted rand index (ARI) to compare our clusters with the original classes.

[28]:
```
from sklearn.metrics import adjusted_rand_score
```

Get the cluster labels for for the unscaled and scaled clusterings.

[29]:
```
predicted_labels_unscaled = fcluster(Z, t=3, criterion='maxclust')
predicted_labels_scaled = fcluster(Z_scaled, t=3, criterion='maxclust')
```

Calculate ARI.

[30]:
```
ari_unscaled = adjusted_rand_score(species_labels, predicted_labels_unscaled)
ari_scaled = adjusted_rand_score(species_labels, predicted_labels_scaled)

print("Adjusted Rand Index for the unscaled dataset:", ari_unscaled)
print("Adjusted Rand Index for the scaled dataset:", ari_scaled)
```

```
Adjusted Rand Index for the unscaled dataset: 0.7311985567707746
Adjusted Rand Index for the scaled dataset: 0.6153229932145449
```

Oh dear, we've made it worse!

The problem isn't that scaling the data was a bad idea. The problem now is that we're treating all of the columns equally, whereas an expert on irises would be able to tell you that the petal lengths and widths tend to be more useful than sepal lengths and widths for distinguishing between different iris species.

2.2 Apply feature weighting

We will now apply this domain knowledge by reducing the weight of sepal length and sepal width.

[31]:
```
# Here we set the first two features (sepal length and sepal width) to be a␣
  ↪fifth as important
# as the last two (petal length and petal width).

feature_weights = np.array([0.2, 0.2, 1., 1.])
```

[32]:
```
# Weights based on feature importances from decision tree in classifying-irises.
  ↪ipynb
# feature_weights = np.array([0.03550967, 0.03550967, 1., 0.78715117])

# Weights based on feature importances from random forest
# feature_weights = np.array([0.22165819, 0.08485548, 0.99763139, 1.])
```

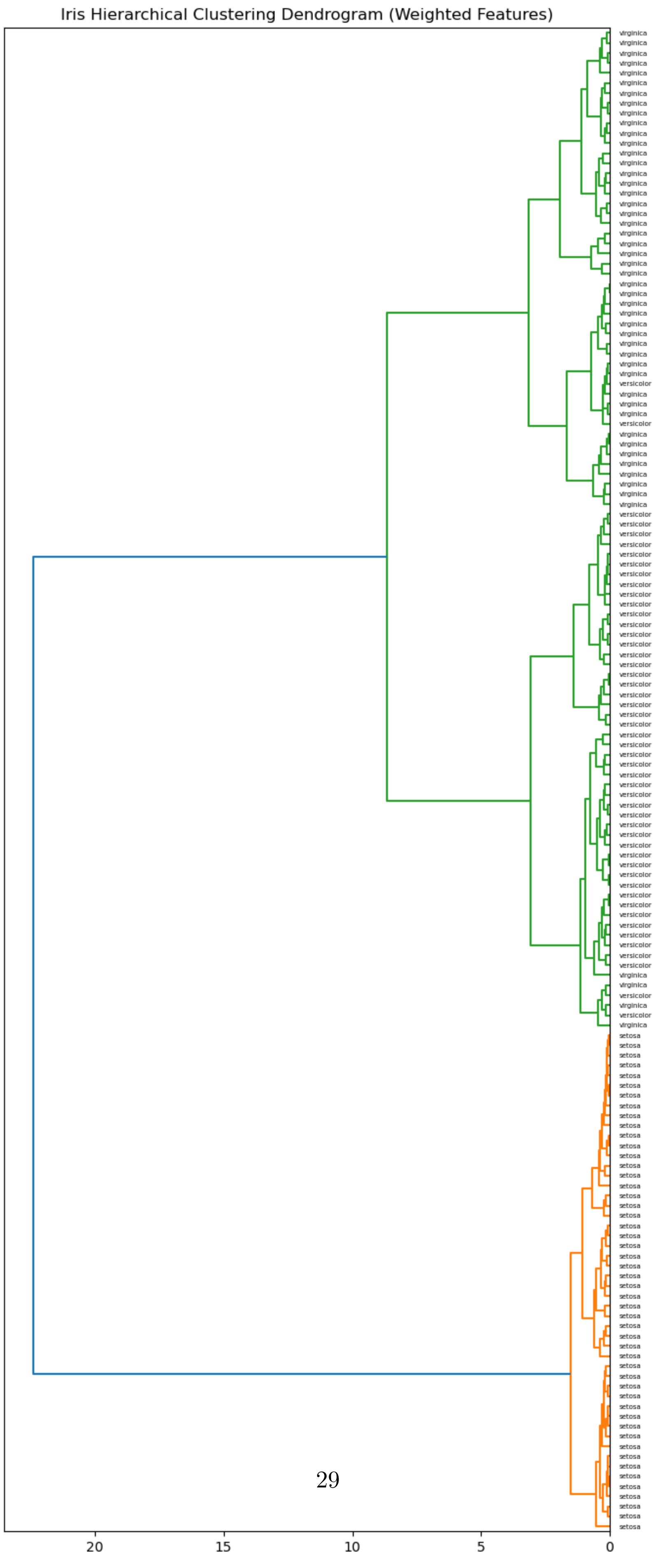
[33]:
```
X_weighted = X_scaled * feature_weights
```

2.2.1 Perform hierarchical clustering on the weighted data

[34]:
```
Z_weighted = linkage(X_weighted, method='ward')
```

[35]:
```
# Plot the dendrogram
plt.figure(figsize=(8, 20))
dendrogram(Z_weighted,
           labels=species_labels,
           orientation='left')
plt.title("Iris Hierarchical Clustering Dendrogram (Weighted Features)")
```

[35]:
```
Text(0.5, 1.0, 'Iris Hierarchical Clustering Dendrogram (Weighted Features)')
```


Iris Hierarchical Clustering Dendrogram (Weighted Features)
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
versicolor
virginica
virginica
virginica
versicolor
virginica
virginica
virginica
virginica
virginica
virginica
virginica
virginica
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
versicolor
virginica
virginica
versicolor
virginica
versicolor
virginica
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
setosa
20
15
10
5
0

2.2.2 Calculate and print ARI

[36]:
```
predicted_labels_weighted = fcluster(Z_weighted, t=3, criterion='maxclust')

ari_weighted = adjusted_rand_score(species_labels, predicted_labels_weighted)
print("Adjusted Rand Index (Weighted):", ari_weighted)
```

```
Adjusted Rand Index (Weighted): 0.8856970310281228
```

We've made a big improvement now - our clusters more closely correspond with the species labels. Of course, we might not have labels in a real-world unsupervised learning situation. This example highlights the value of domain knowledge in data analysis and machine learning. Without domain knowledge, we wouldn't have known that it would be a good idea to to weight the petal features more heavily than the sepal features.