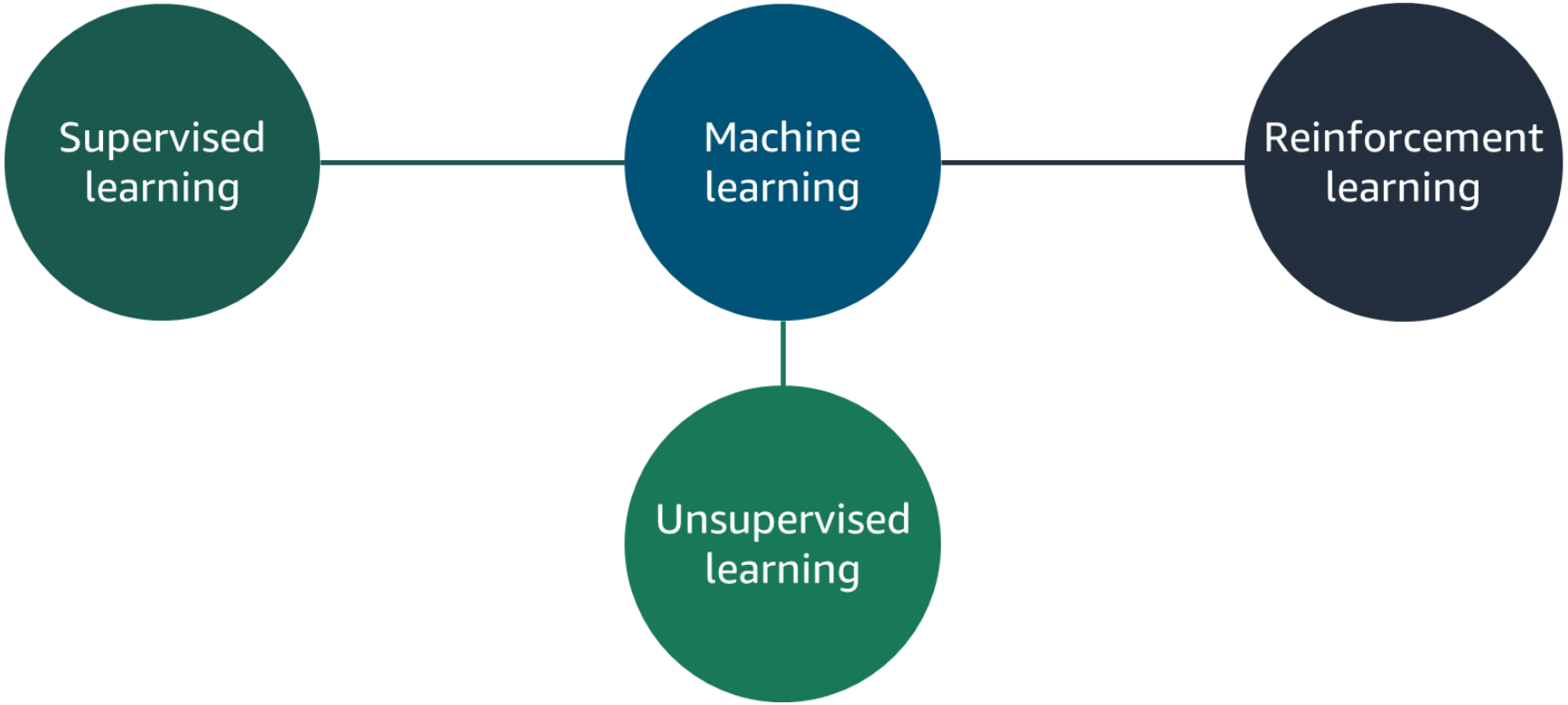
Unsupervised Learning

Outline

- What is Unsupervised Learning?
- Supervised vs. Unsupervised
- Types of unsupervised learning
- Unsupervised learning methods

Types of machine learning



Source: Amazon Web Services

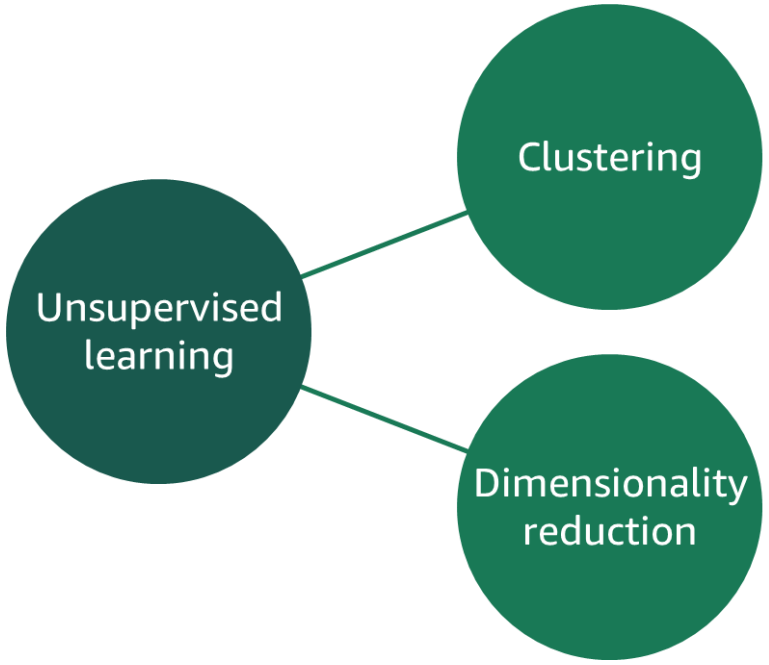
Supervised vs. Unsupervised learning

- Supervised Learning:
  - Use historical labeled data
  - Predict a label on new data
  - Regression or classification

- Unsupervised Learning:
  - Use unlabeled data
  - Discover patterns, clusters, or significant components

Unsupervised learning

The machine is given **unlabeled** data.



- Clustering:
  - Using features, group together data rows into distinct clusters
- Dimensionality Reduction:
  - Using features, discover how to combine and reduce into fewer components

Source: Amazon Web Services

Unsupervised learning

Paradigm shift from supervised to unsupervised learning

- Supervised performance metrics will not apply for unsupervised learning.

- How can we compare to a correct label answer, if there is no label to begin with?

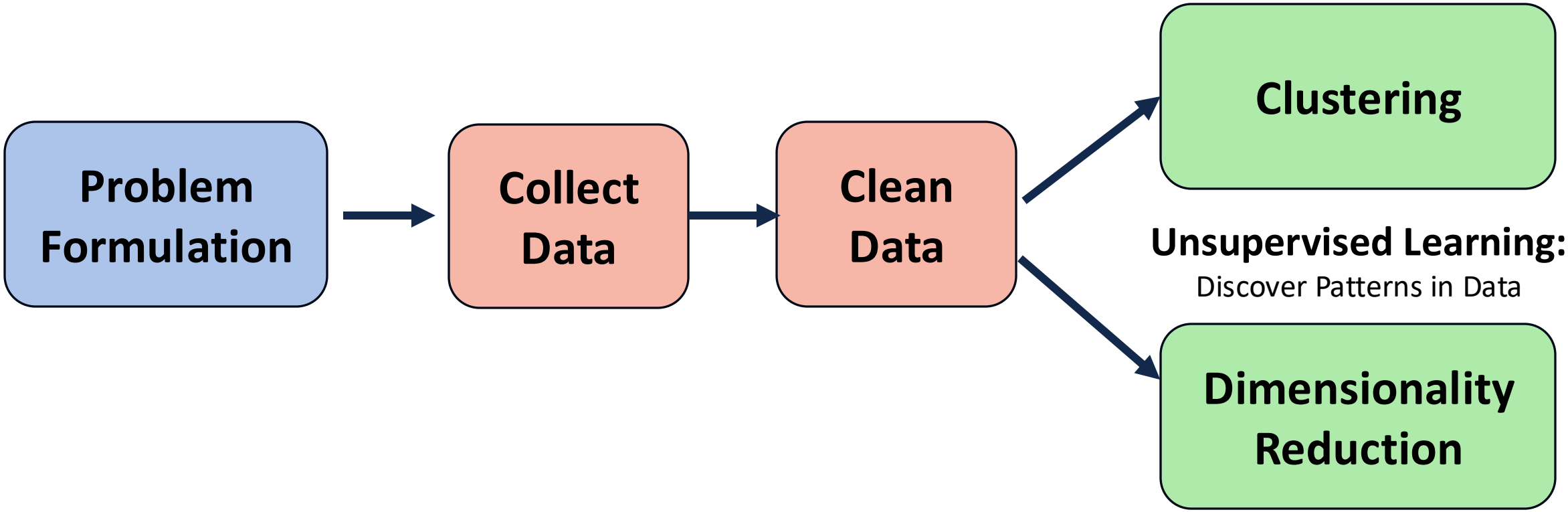
Unsupervised learning

Paradigm shift from supervised to unsupervised learning

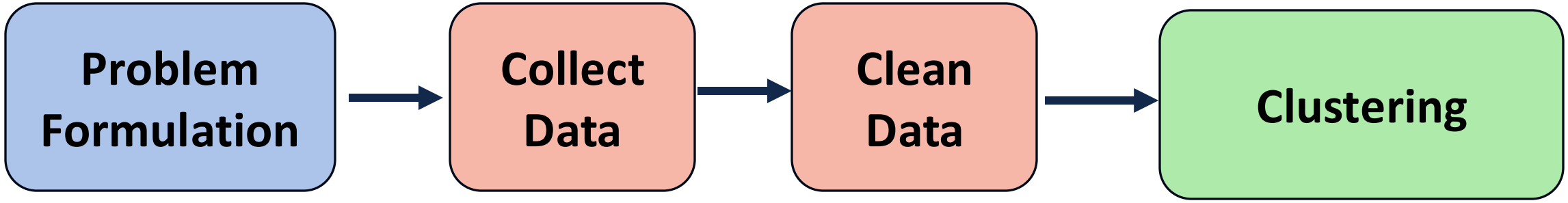
- Instead of metrics like RMSE or Accuracy, we need to figure out other ways of assessing unsupervised model performance.

- Even our understanding of what “performance” actually means need to change with unsupervised learning.

Machine Learning Workflow

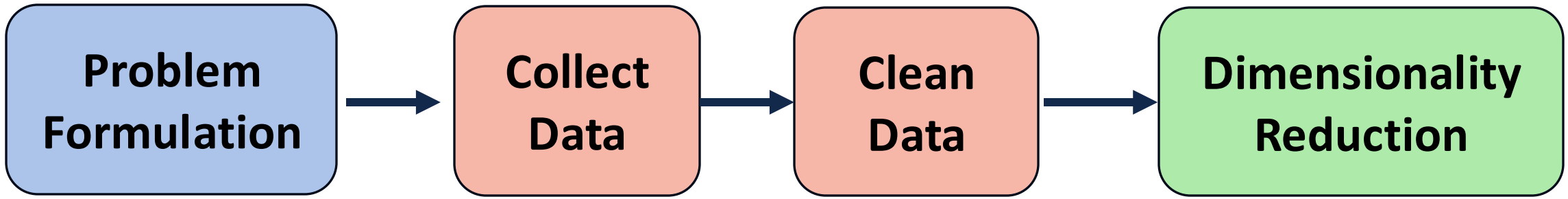


Machine Learning Workflow



Clustering: If we have unlabeled data, can we attempt to cluster or group similar data points together to “discover” possible labels for clusters?

Machine Learning Workflow

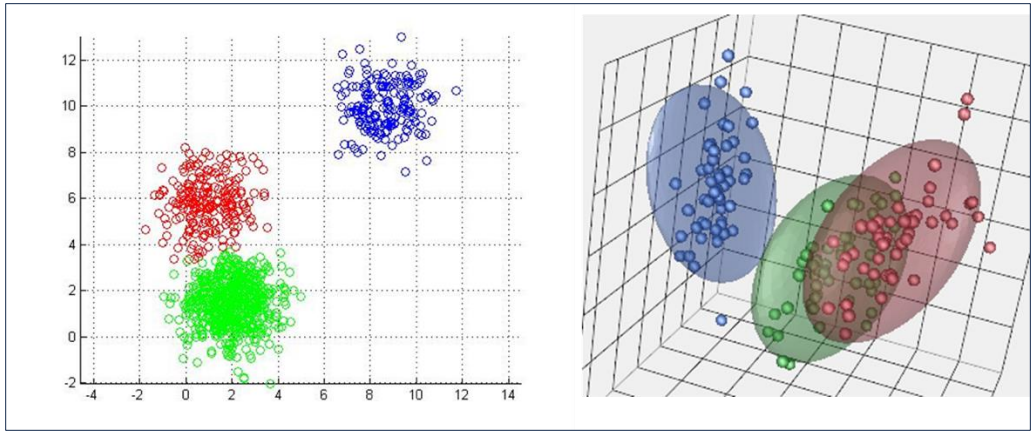


Dimensionality Reduction: If we have unlabeled data, can we attempt to reduce the number of features by combining them into new components? Do these new components give us further insight for the data?

Unsupervised Learning Methods

- K-Means Clustering
- Principle Component Analysis (PCA)
- Hierarchical Clustering

K-Means Clustering



K-Means Clustering

- A dimensionality reduction algorithm
- Reduce data down to K dimensions
- Clusters data points into K groups
- Step1: Select K points at random as cluster centers
- Step 2: Assign objects to their closest cluster center according to the distance metric
- Step3: Calculate the new centroid (i.e., mean) of all objects in each cluster
- Step 4: Repeat the above steps until the same points are assigned to each cluster in consecutive rounds.

Image source: https://stanford.edu/~cpiech/cs221/handouts/kmeans.html

Principal Component Analysis (PCA)

Finds eigenvectors in the higher dimensional data

- These define hyperplanes that split the data while preserving the most variance in it.
- The data gets projected onto these hyperplanes, which represent the lower dimensions.

- **Also useful for image compression and facial recognition**

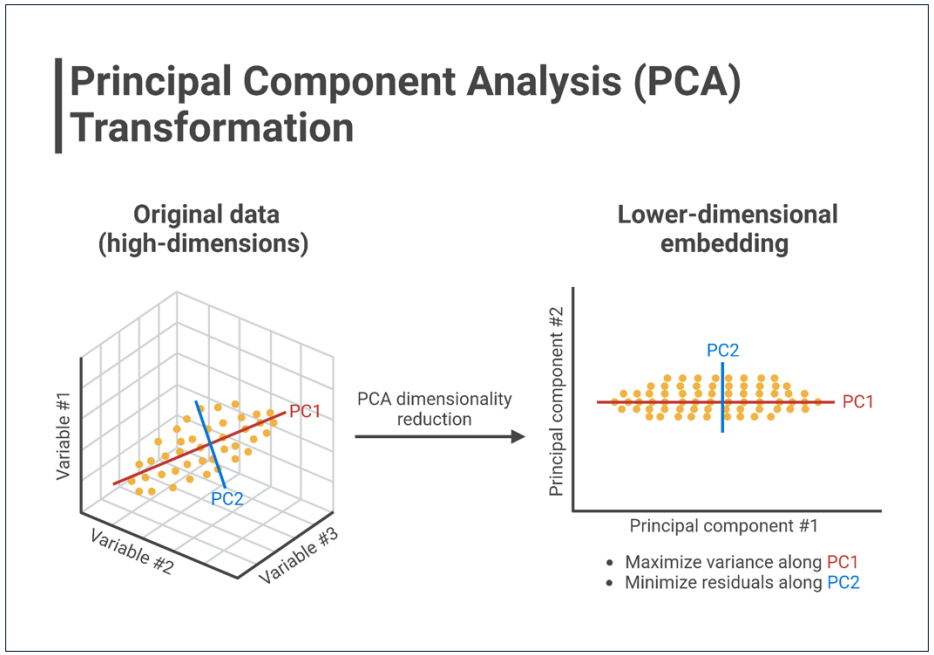


Image source: https://www.biorender.com/template/principal-component-analysis-pca-transformation

Example: Iris Flower Data

- **Iris dataset: comes with Scikit-learn**
- **An Iris flower has petals and sepals (the lower, supportive part)**
- **The length and width of the petals and sepal for many Iris specimens**
  - 4 dimensions for 3 different kinds of flowers
  - Subspecies classification of each flower
- **PCA allows us visualize this in 2 dimensions instead of 4, while still preserving the most information.**

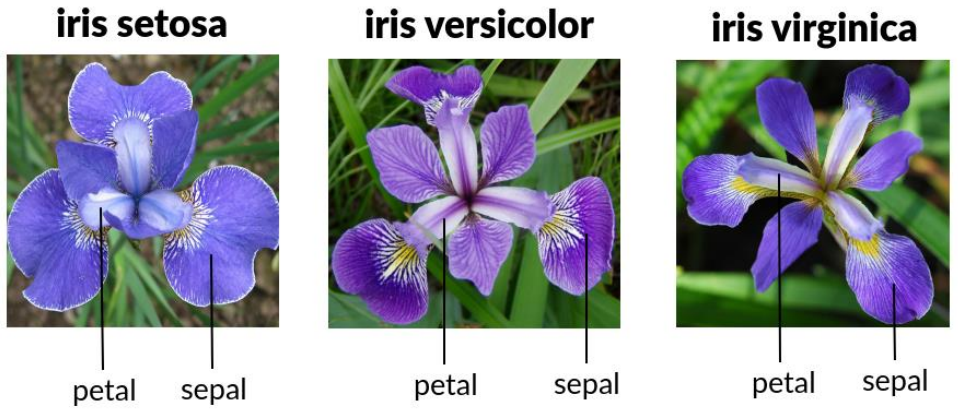


Image source: https://www.analyticsvidhya.com/blog/2022/06/iris-flowers-classification-using-machine-learning/

Example: Iris Flower Data - PCA

Documentation:
https://scikit-learn.org/stable/auto_examples/datasets/plot_iris_dataset.html#sphx-glr-auto-examples-datasets-plot-iris-dataset-py

```
from sklearn.datasets import load_iris
from sklearn.decomposition import PCA
import pylab as pl
from itertools import cycle

iris = load_iris()
num_sample, num_feature = iris.data.shape

print(num_sample)
print(num_feature)
print(list(iris.target_names))
```

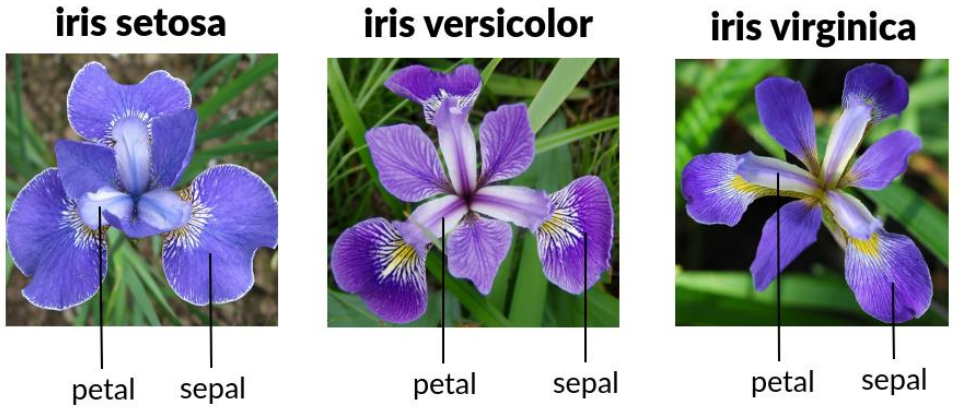


Image source: https://www.analyticsvidhya.com/blog/2022/06/iris-flowers-classification-using-machine-learning/

Hierarchical Clustering

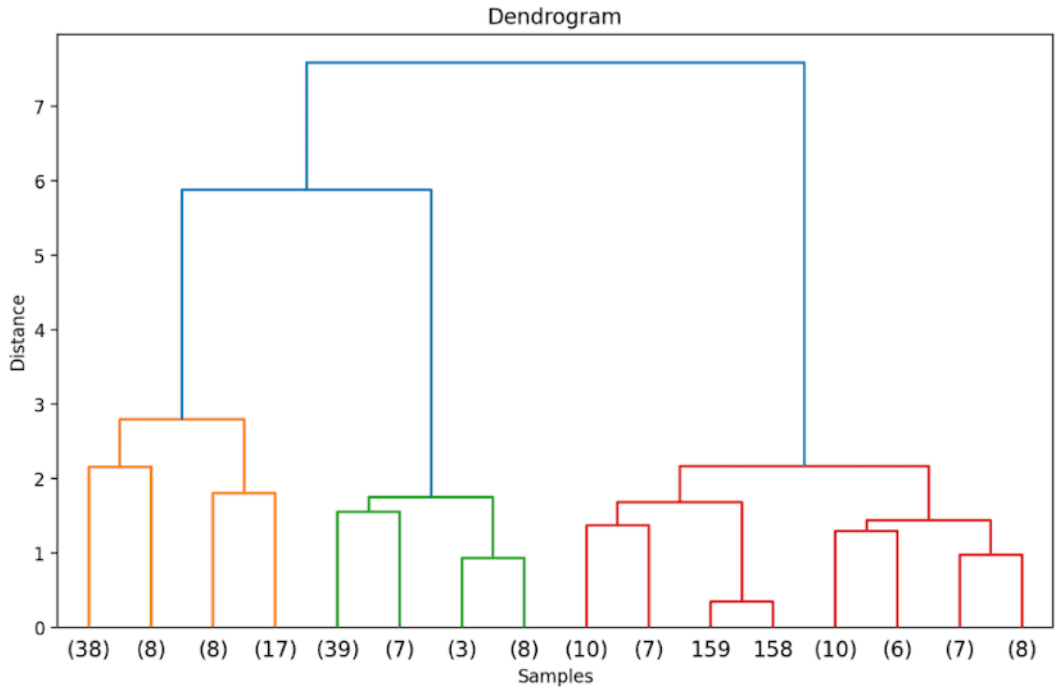
- Like most clustering algorithms, Hierarchical Clustering relies on measuring which data points are most “similar” to other data points.
- “Similarity” is defined by choosing a distance metric.
- So why use Hierarchical Clustering?
  - Easy to understand and visualize
  - Helps users decide how many clusters to choose
  - Not necessary to choose cluster amount before running the algorithm

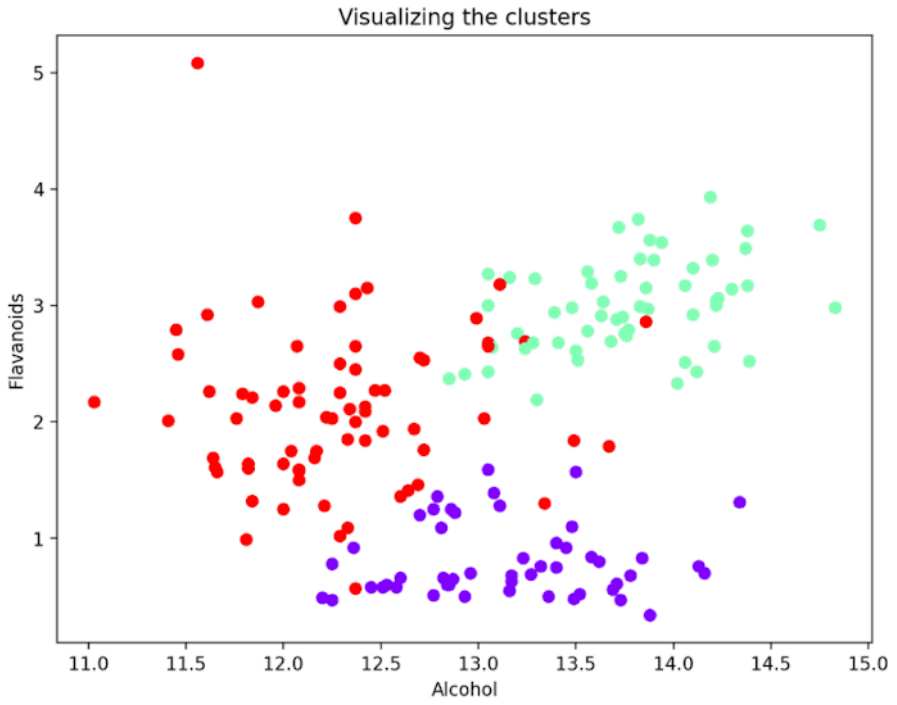
Hierarchical Clustering

- Divides points into potential clusters
- Agglomerative Approach:
  - Each point begins as its own cluster, then clusters are joined.
- Divisive Approach:
  - All points begin in the same cluster, then clusters are split.

Hierarchical Clustering

- Agglomerative Approach





Source: https://www.kdnuggets.com/unveiling-hidden-patterns-an-introduction-to-hierarchical-clustering