Clustering Analysis in R

An Introduction for DAT 280

Alex Marsella

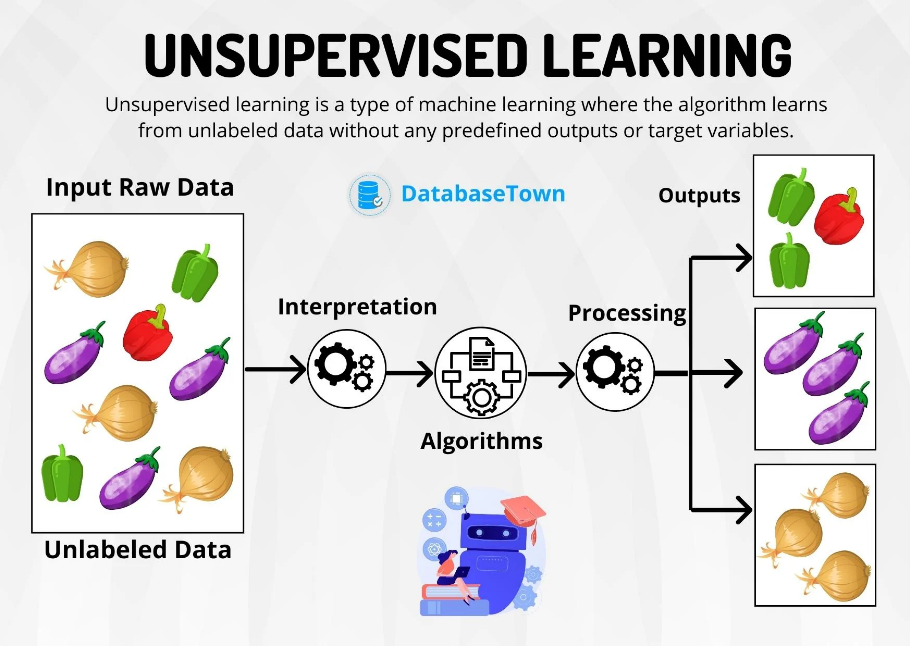
February 22, 2024

Today's gameplan

- I will introduce you to the idea of clustering, a form of machine learning.
- Today's "Lecture" lecture will be more lecture-y, while Monday's "Coding" lecture will be more hands-on.
- I still have an exercise at the end for you to try out.

Introduction to Clustering

Clustering is a type of unsupervised learning method used to group similar entities together. It's a powerful tool in data analysis, allowing us to discover natural groupings in data based on their characteristics.


UNSUPERVISED LEARNING
Unsupervised learning is a type of machine learning where the algorithm learns from unlabeled data without any predefined outputs or target variables.
Input Raw Data
DatabaseTown
Outputs
Interpretation
Processing
Algorithms
Unlabeled Data

Real-World Applications of Clustering

Clustering has a wide range of applications across different fields:

- **Market segmentation**: Grouping customers based on purchasing behavior.
- **Biology**: Classifying plants or animals based on their features.
- **Image segmentation**: Dividing digital images into multiple segments (sets of pixels).
- **Anomaly detection**: Identifying unusual data points that do not fit into any cluster.

Two types of clustering

- **Hierarchical Clustering:** Builds a hierarchy of clusters either through a bottom-up (agglomerative) or top-down (divisive) approach.
- **Partitioning Clustering:** Divides the data into non-overlapping subsets (clusters) such that each data point is in exactly one subset. The most common method is k-means clustering.

K-means Clustering

K-means clustering groups n observations into k clusters in which each observation belongs to the cluster with the nearest mean.

Algorithm Steps

1. **Initialization**: Start with k centroids by *randomly* selecting k points from the dataset.
2. **Assignment**: Assign each point to the nearest centroid.
3. **Update**: Recalculate the centroids as the center of the points assigned to each cluster.
4. **Repeat**: Repeat steps 2 and 3 until the centroids no longer change significantly.

Before we move any further, let's load in `tidyverse`, `cluster`, and `iris`

```r
library(tidyverse)
library(cluster)
data(iris)
```

K-means clustering in the `iris` dataset

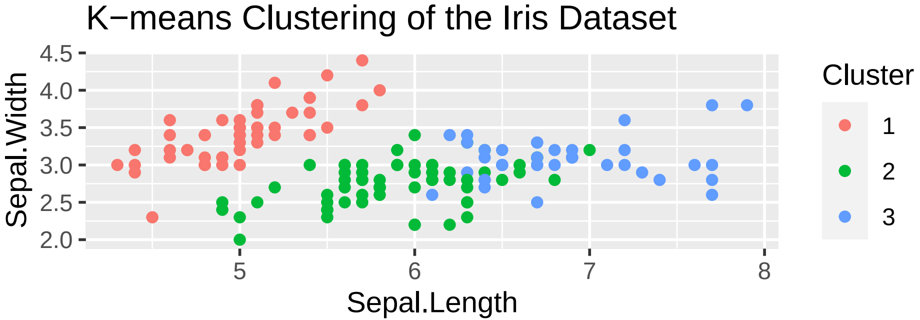
- We will use the `kmeans()` command on columns 1:4 of `iris` and tell it to make 3 clusters.
- This will assign all irises to one of three "clusters" based on the characteristics in those four columns.
  - Sepal width and length, petal width and length
- Note that we do NOT cluster on the categorical aspect of species.
  - Species is nature's clustering, so to speak, we want to see if we can predict that or come up with our own clustering.

```
set.seed(123)  # for reproducibility
kmeans_result <- kmeans(iris[, 1:4], centers = 3)
```

Plotting the result.

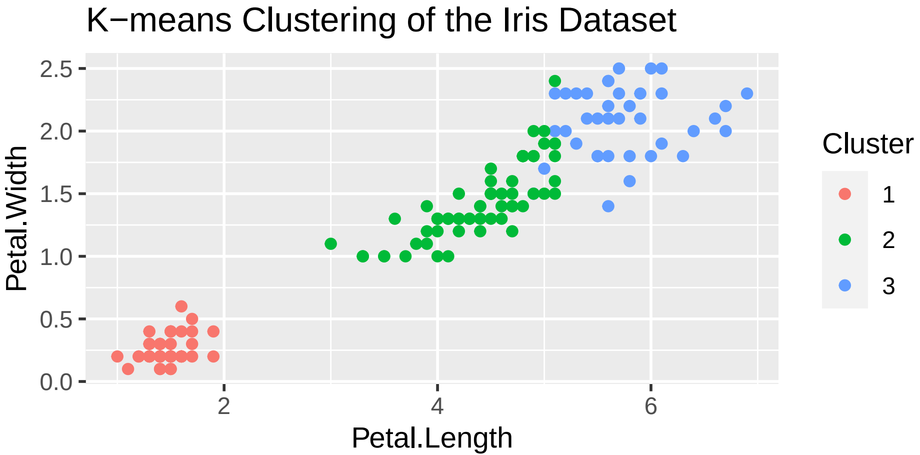
- We color it by the clusters stored in `kmeans_result` even though we are using `iris`
  - note that I wrap it in `factor()` since the clusters are stored as numeric but I want them to be read as categories.

```
ggplot(iris, aes(Sepal.Length, Sepal.Width,
    color = factor(kmeans_result$cluster))) +
    geom_point() + labs(title = "K-means Clustering of the Iri
    color = "Cluster")
```



Plotting the result, again.

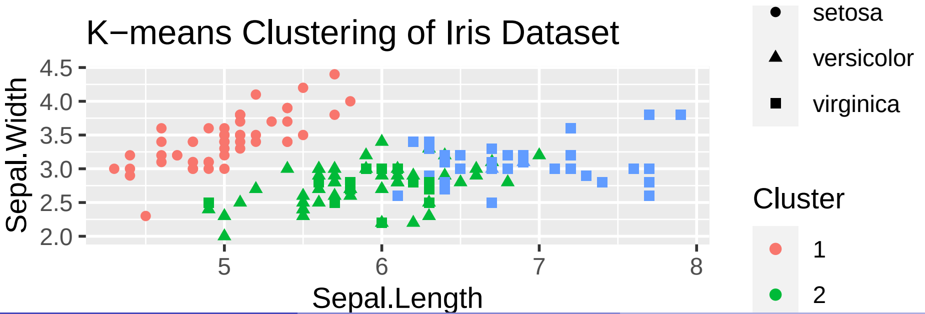
```
ggplot(iris, aes(Petal.Length, Petal.Width,
    color = factor(kmeans_result$cluster))) +
    geom_point() + labs(title = "K-means Clustering of the Iri
    color = "Cluster")
```



Comparing it to the three species of Iris

- Clustering can provide prediction about a class to which something belongs.
- Notice how the clusters are fairly congruent with the species of Iris.

```
ggplot(iris, aes(Sepal.Length, Sepal.Width,
    color = factor(kmeans_result$cluster))) +
    geom_point(aes(shape = Species)) +
    labs(title = "K-means Clustering of Iris Dataset",
        color = "Cluster")
```



Hierarchical Clustering

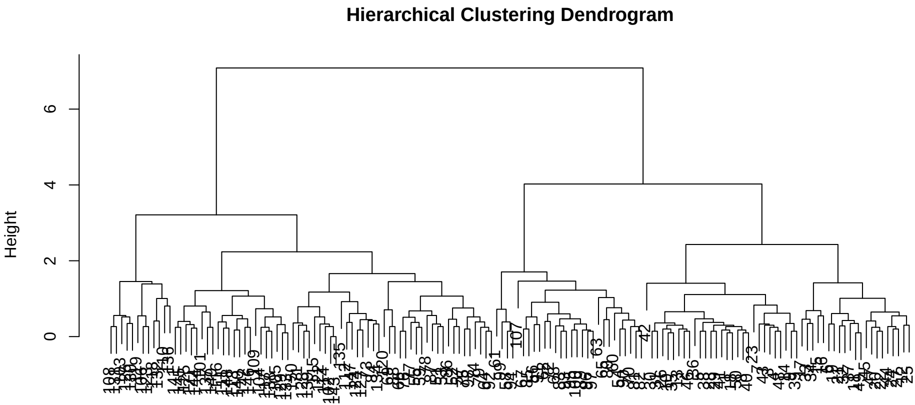
Hierarchical clustering creates a tree of clusters. It doesn't require us to pre-specify the number of clusters. Instead, it produces a dendrogram, allowing us to choose the number of clusters based on the tree.

Agglomerative Hierarchical Clustering

1. **Start**: Treats each point as a separate cluster.
2. **Find**: Identifies the closest two clusters and merges them.
3. **Repeat**: Continue the process until all points are clustered into a single group.
4. **Dendrogram**: A tree-like diagram that records the sequences of merges or splits.

Hierarchical Clustering

```r
dist_mat <- dist(iris[, 1:4])  # Calculate distance matrix
hc <- hclust(dist_mat)  # Perform hierarchical clustering
plot(hc, main = "Hierarchical Clustering Dendrogram",
     xlab = "", sub = "")  # Plot the dendrogram
```



Choosing the right number of clusters

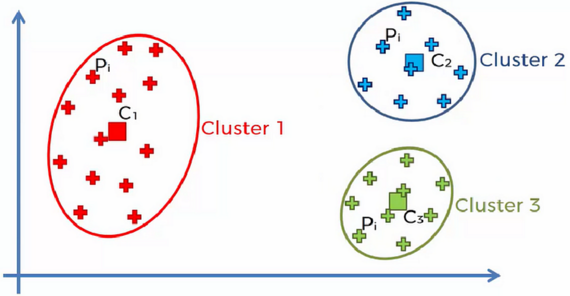
The Elbow Method

The Elbow Method is a popular technique to determine the optimal number of clusters `k` in K-means clustering. It involves the following steps:

1. **Compute Clustering**: Perform K-means clustering on the dataset for a range of values of `k` (e.g., 1 to 10).
2. **Calculate Within-Cluster Sum of Square (WCSS)**: For each `k`, calculate the total within-cluster sum of square (WCSS).
3. **Plot the Curve**: Plot `k` against the WCSS. The plot typically shows a rapid decline in WCSS as `k` increases, which eventually slows, creating an “elbow”.
4. **Determine the Elbow Point**: The point where the rate of decrease sharply changes (the elbow) represents the optimal `k`.

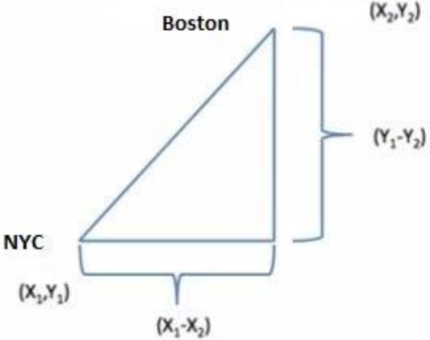
The "Within-Cluster Sum of Squares"

- Sum of the squared distances between an observation in a cluster and its centroid.



$$\mathrm{WCSS} = \sum_{\mathrm{P_i\ in\ Cluster\ 1}} \mathrm{distance}(\mathrm{P_i}, \mathrm{C_1})^2 + \sum_{\mathrm{P_i\ in\ Cluster\ 2}} \mathrm{distance}(\mathrm{P_i}, \mathrm{C_2})^2 + \sum_{\mathrm{P_i\ in\ Cluster\ 3}} \mathrm{distance}(\mathrm{P_i}, \mathrm{C_3})^2$$

$$\sqrt{(X_1 - X_2)^2 + (Y_1 - Y_2)^2}$$



$$\sqrt{(A_1 - A_2)^2 + (B_1 - B_2)^2 + \ldots + (Z_1 - Z_2)^2}$$

Why Use the Elbow Method?

- **Simplicity**: Easy to understand and implement.
- **Heuristic**: Helps in making an informed decision based on the WCSS plot.

Limitations

- **Subjectivity**: The exact “elbow” point can sometimes be subjective or not very clear.
- **Not Always Accurate**: May not always provide the best number of clusters for complex datasets.

Coding the setup

- Remove the species column (fifth column) to only include features.
- Create a vector to store WCSS values for 1-10 clusters.
- Create a “for loop” in R to calculate WCSS 10 times for 1-10 clusters.

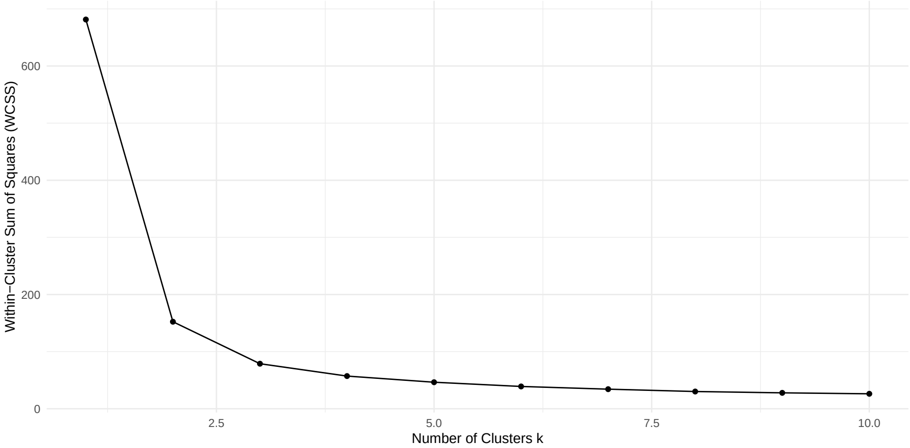
```r
iris_data <- iris[, -5]

wcss <- numeric(10)

for (k in 1:10) {
    # k will take on the value 1,
    # then 2, and so on, repeating
    # to 10
    kmeans_result <- kmeans(iris_data,
        centers = k, nstart = 20)
    wcss[k] <- kmeans_result$tot.withinss
}
```

Creating the plot: Is two clusters enough for Irises?

```
ggplot(, aes(x = 1:10, y = wcss)) + geom_line() +
    geom_point() + xlab("Number of Clusters k") +
    ylab("Within-Cluster Sum of Squares (WCSS)") +
    theme_minimal()
```



Assessing Cluster Quality

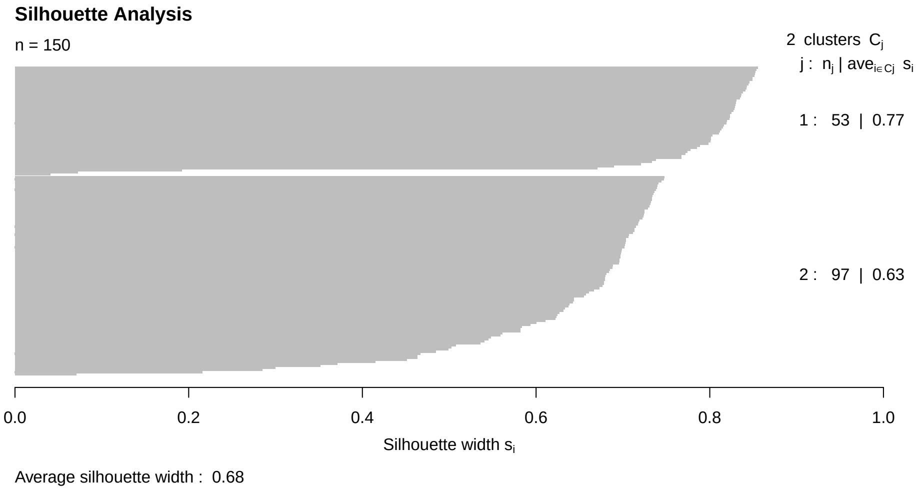
Understanding the quality of the clusters formed is crucial for validating the results of your clustering analysis.

Silhouette Analysis

- Silhouette analysis measures how similar an object is to its own cluster compared to other clusters.
- The silhouette score ranges from -1 to 1, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters.
  - $s > 0.7$ "strong"
  - $0.7 > s > 0.5$ "reasonable"
  - $0.5 > s > 0.25$ "weak"
  - Close to 0 implies overlapping clusters
  - Negative values imply incorrect assignment to clusters.

Silhouette Analysis

```r
sil <- silhouette(kmeans_result$cluster,
    dist(iris[, 1:4]))
plot(sil, main = "Silhouette Analysis")  # base R plotting is
```



Summary & Final Thoughts

- Clustering is a versatile tool in data analysis, with numerous applications.
- The choice of algorithm depends on the dataset and the specific requirements of the analysis.
- Assessing cluster quality is essential for ensuring meaningful results.
- Continuous exploration and learning are key, as new clustering methods and applications are regularly developed.

Exercise

Apply clustering analysis to segment customers based on their purchasing behavior. This is a common task in marketing and business strategy, helping companies tailor their approaches to different customer groups.

Dataset Description

The dataset, `CustomerData.csv`, represents purchasing data collected from a retail store. It includes the following variables:

- `CustomerID`: Unique identifier for each customer.
- `AnnualIncome`: The annual income of the customer (in thousands).
- `SpendingScore`: A score assigned by the mall based on customer behavior and spending nature (1-100).

Objective

Your task is to segment the customers into distinct groups based on their `AnnualIncome` and `SpendingScore`. You will:

1. Perform exploratory data analysis (EDA) to understand the dataset.
2. Use k-means clustering to identify customer segments.
3. Evaluate the clustering and interpret the customer segments.

Step 1: Load the Dataset

Start by loading the dataset into R and taking a peek. I have set this up for you to expedite it.

```
customer_data <- read.csv("CustomerData.csv")
head(customer_data)
```

```
##   CustomerID AnnualIncome SpendingScore
## 1          1           54            25
## 2          2          121            96
## 3          3           70            61
## 4          4          134            52
## 5          5          142            41
## 6          6           21            88
```

Step 2: Exploratory Data Analysis (EDA)

I have already performed a basic exploratory analysis for you.

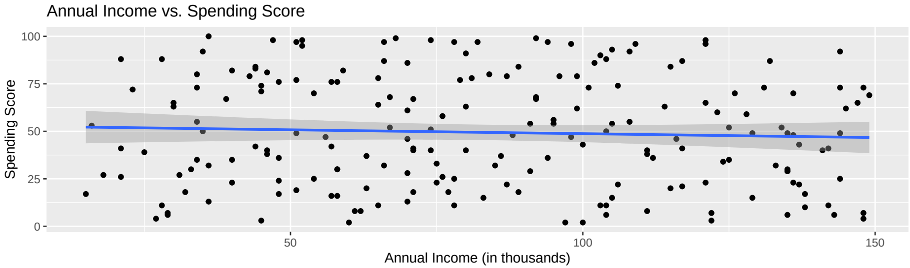
```r
# Summary statistics
summary(customer_data[, c("AnnualIncome",
    "SpendingScore")])
```

```
##   AnnualIncome    SpendingScore
##  Min.   : 15.00   Min.   :  2.00
##  1st Qu.: 51.75   1st Qu.: 24.75
##  Median : 80.00   Median : 47.50
##  Mean   : 83.34   Mean   : 49.44
##  3rd Qu.:114.25   3rd Qu.: 74.50
##  Max.   :149.00   Max.   :100.00
```

Step 2: Exploratory Data Analysis (EDA)

- Notice how there is no discernible trend at all.

```
ggplot(customer_data, aes(x = AnnualIncome,
    y = SpendingScore)) + geom_point() +
    labs(title = "Annual Income vs. Spending Score",
        x = "Annual Income (in thousands)",
        y = "Spending Score") + geom_smooth(method = "lm")
```



Step 3: Find the elbow

Use the elbow method to determine a good number of clusters (k).

Step 4: K-means Clustering

Use k-means clustering to segment the customers. Use four clusters.

Step 5: Visualize it

- Plot the results with a unique trendline for each of the four customer segments.

Step 5: Describe and Interpret

How does the relationship between Annual Income and Spending Score differ by customer type?