ISA 419: Data-Driven Security

13: Clustering

Fadel M. Megahed, PhD

Endres Associate Professor
Farmer School of Business
Miami University

@FadelMegahed
fmegahed
fmegahed@miamioh.edu
Automated Scheduler for Office Hours

Spring 2024

Quick Refresher of Last Class

✅ Explain the difference between supervised and unsupervised learning.

✅ EUnderstand the importance of different preprocessing steps and when they should be used in ML.

✅ Explain typical error measures used for supervised and unsupervised learning tasks.

Learning Objectives for Today's Class

- Define clustering
- Explain the $k-$means clustering algorithm for numeric datasets
- Implement the $k-$means algorithm in Python using the `pycaret` package
- Describe scenarios where other/advanced clustering algorithms are needed

What is Clustering?

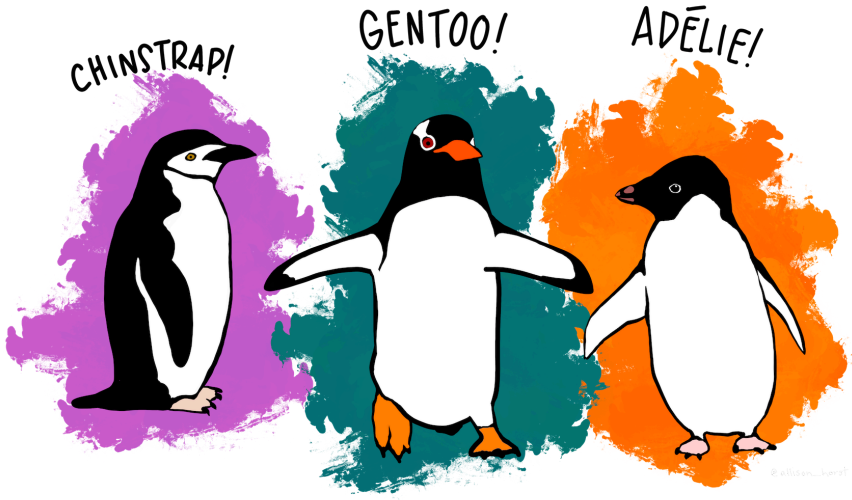
The Problem of Clustering

- Given a **set of (high-dimensional) observations**, with a notion of **distance** between observations, **group the observations** into **some number of clusters**, so that:
  - Members of a cluster are close/similar to each other
  - Members of different clusters are dissimilar
- **Usually:**
  - The observations are in a high-dimensional space
  - Similarity is defined using a distance measure, e.g.,
    - Euclidean, Cosine, Jaccard, edit distance, etc.

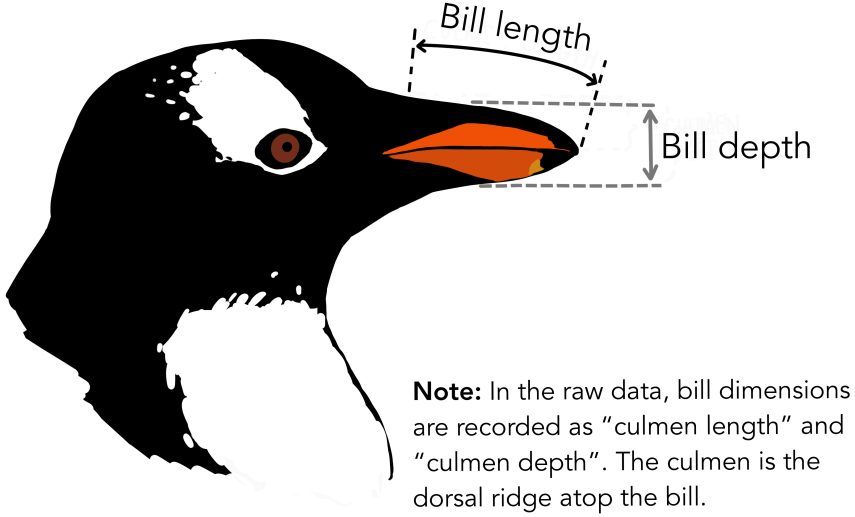
Source: J. Leskovec, A. Rajaraman, J. Ullman: Mining of Massive Datasets, http://www.mmds.org

Clustering in 2D Space: A Fun Example

Meet the Palmer penguins



Anatomical description of the dataset:



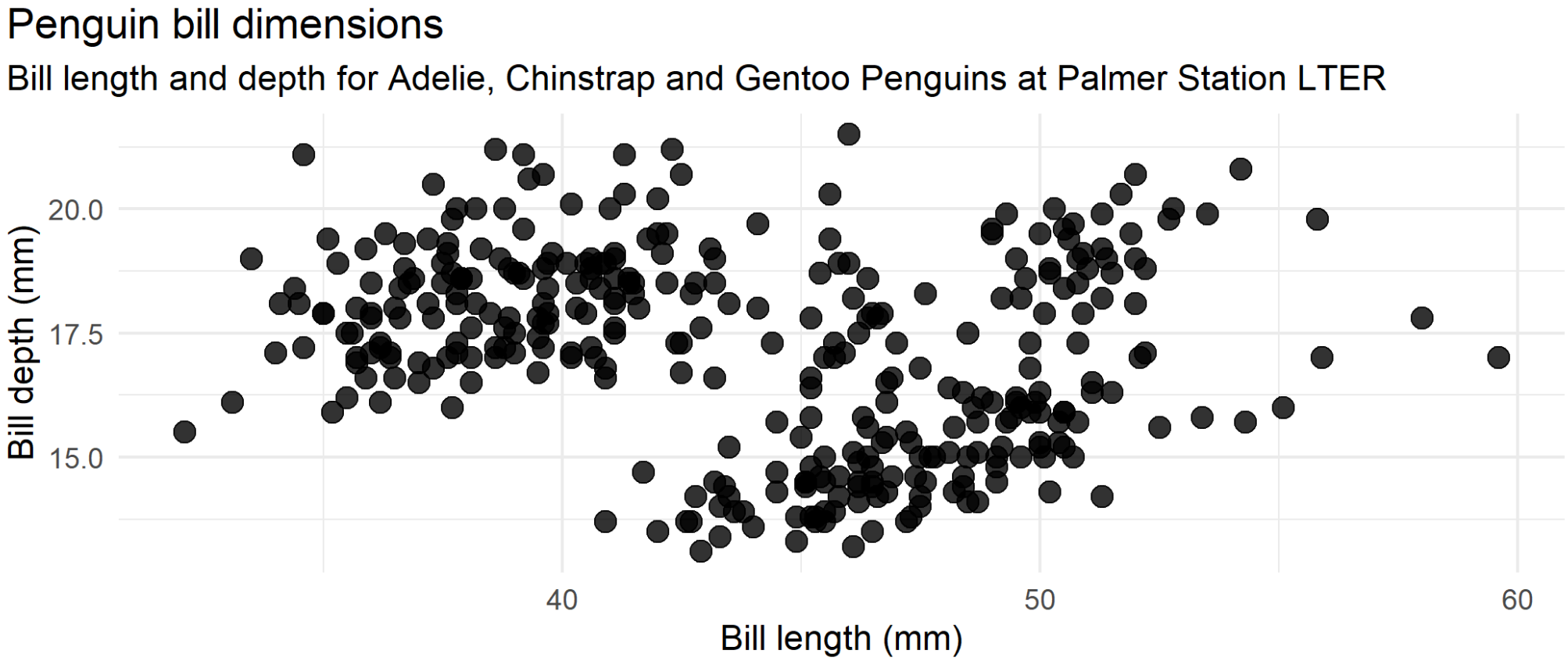
Note: In the raw data, bill dimensions are recorded as "culmen length" and "culmen depth". The culmen is the dorsal ridge atop the bill.

Source: The data are available by CC-0 license in accordance with the Palmer Station LTER Data Policy and the LTER Data Access Policy for Type I data. The artwork is by Allison Horst and available at https://allisonhorst.github.io/palmerpenguins/, and the data is downloaded using the palmerpenguins R package.

Clustering in 2D Space: Formulation

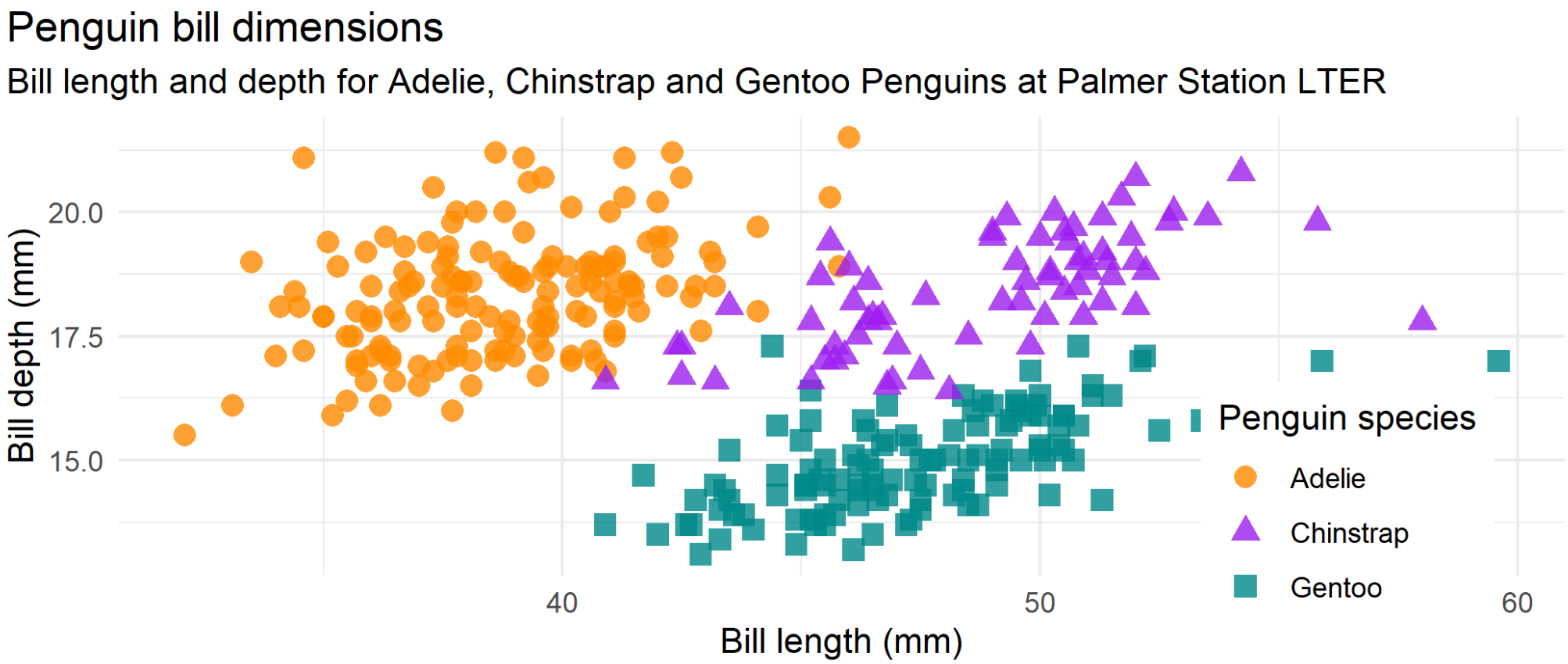
- Given a **set of observations (each containing bill length and depth)**, with a notion of **Euclidean distance** between observations, **group the observations** into **3 clusters**, so that:
  - Members of a cluster are close/similar to each other
  - Members of different clusters are dissimilar
- Note we are assuming that we did not have a "label/type" for each penguin.

Clustering in 2D Space: Raw Data



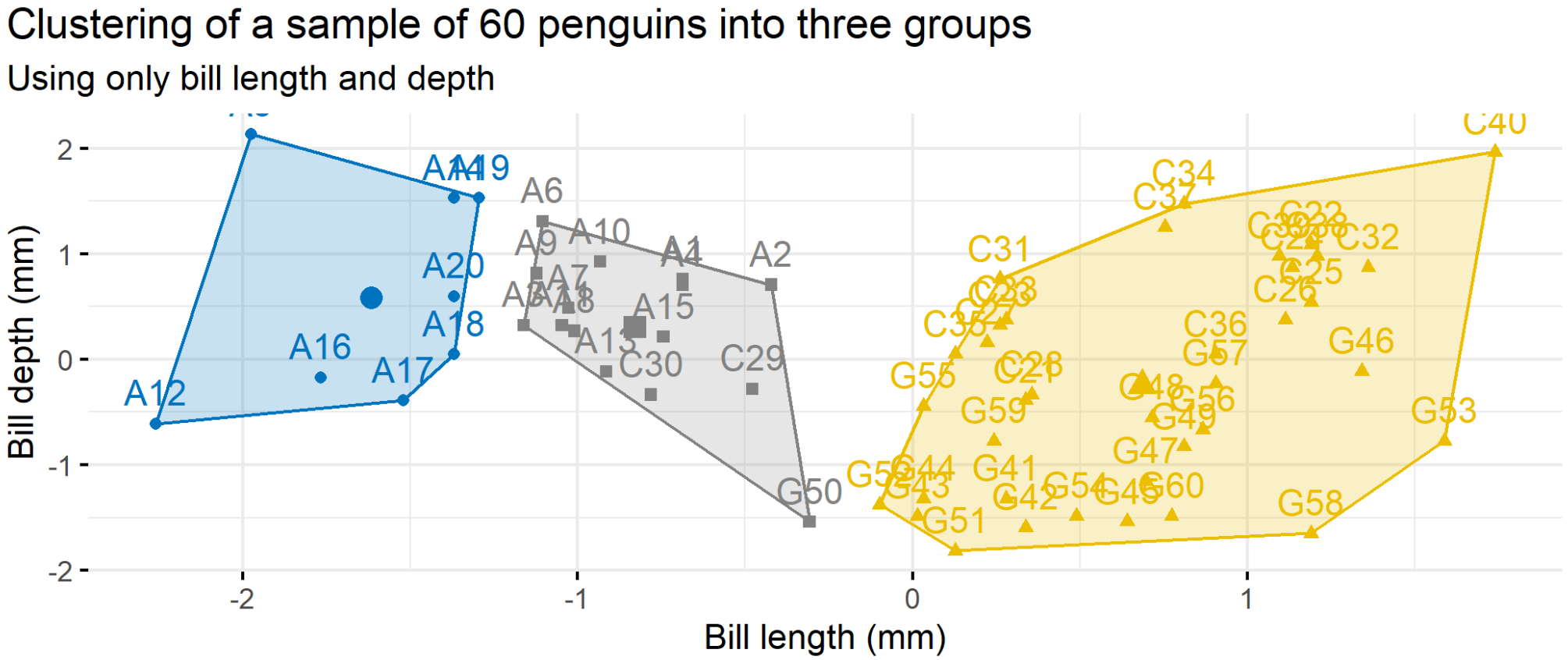
Source: The data are available by CC-0 license in accordance with the Palmer Station LTER Data Policy and the LTER Data Access Policy for Type I data. The artwork is by Allison Horst and available at https://allisonhorst.github.io/palmerpenguins/, and the data is downloaded using the palmerpenguins package.

Clustering in 2D Space: Labeled Raw Data



Source: The data are available by CC-0 license in accordance with the Palmer Station LTER Data Policy and the LTER Data Access Policy for Type I data. The artwork is by Allison Horst and available at https://allisonhorst.github.io/palmerpenguins/, and the data is downloaded using the palmerpenguins package.

Clustering in 2D Space: Clustering Results



Source: The data are available by CC-0 license in accordance with the Palmer Station LTER Data Policy and the LTER Data Access Policy for Type I data. The artwork is by Allison Horst and available at https://allisonhorst.github.io/palmerpenguins/, and the data is downloaded using the palmerpenguins R package.

Comments on the 2D Clustering Problem

Even though the 2D Space clustering problem is the easiest problem to "solve" since we can benefit by plotting the data, **clustering is hard**.

Some important questions:

- With all the variables being numerical, we often assume **Euclidean distance**. This can be problematic when:
  - variables have significantly different scales
  - we are including information that is not pertinent to grouping
- How do you determine the number of clusters (k)?
- How to represent a cluster of many points?
- How do we determine the "nearness" of clusters?

An Overview of Clustering Methods

Categories	Abb. name	Volume			Variety		Velocity	Other criterion
		Size of Dataset	Handling High Dimensionality	Handling Noisy Data	Type of Dataset	Clusters Shape	complexity of Algorithm	Input Parameter
Partitional algorithms	K-Means [25]	Large	No	No	Numerical	Non-convex	O(nkd)	1
	K-modes [19]	Large	Yes	No	Categorical	Non-convex	O(n)	1
	K-medoids [33]	Small	Yes	Yes	Categorical	Non-convex	$O(n^2dt)$	1
	PAM [31]	Small	No	No	Numerical	Non-convex	$O(k(n-k)^2)$	1
	CLARA [23]	Large	No	No	Numerical	Non-convex	$O(k(40+k)^2+k(n-k))$	1
	CLARANS [32]	Large	No	No	Numerical	Non-convex	$O(kn^2)$	2
	FCM [6]	Large	No	No	Numerical	Non-convex	O(n)	1
Hierarchical algorithms	BIRCH [40]	Large	No	No	Numerical	Non-convex	O(n)	2
	CURE [14]	Large	Yes	Yes	Numerical	Arbitrary	$O(n^2 \log n)$	2
	ROCK [15]	Large	No	No	Categorical and Numerical	Arbitrary	$O(n^2+nmmma+n^2 \log n)$	1
	Chameleon [22]	Large	Yes	No	All type of data	Arbitrary	$O(n^2)$	3
	ECHIDNA [26]	Large	No	No	Multivariate Data	Non-convex	$O(N * B(1 + \log_B m))$	2
Density-based algorithms	DBSCAN [9]	Large	No	No	Numerical	Arbitrary	O(nlogn) If a spatial index is used Otherwise, it is $O(n^2)$.	2
	OPTICS [5]	Large	No	Yes	Numerical	Arbitrary	O(nlogn)	2
	DBCLASD [39]	Large	No	Yes	Numerical	Arbitrary	$O(3n^2)$	No
	DENCLUE [17]	Large	Yes	Yes	Numerical	Arbitrary	$O(\log\|D\|)$	2
Grid- based algorithms	Wave-Cluster [34]	Large	No	Yes	Special data	Arbitrary	O(n)	3
	STING [37]	Large	No	Yes	Special data	Arbitrary	O(k)	1
	CLIQUE [21]	Large	Yes	No	Numerical	Arbitrary	O(Ck + mk)	2
	OptiGrid [18]	Large	Yes	Yes	Special data	Arbitrary	Between O(nd) and O(nd log n)	3
Model- based algorithms	EM [8]	Large	Yes	No	Special data	Non-convex	O(knp)	3
	COBWEB [12]	Small	No	No	Numerical	Non-convex	$O(n^2)$	1
	CLASSIT [13]	Small	No	No	Numerical	Non-convex	$O(n^2)$	1
	SOMs [24]	Small	Yes	No	Multivariate Data	Non-convex	$O(n^2m)$	2

Source: A. Fahad, et al.,"A Survey of Clustering Algorithms for Big Data: Taxonomy and Empirical Analysis" in IEEE Transactions on Emerging Topics in Computing, vol. 2, no. 03, pp. 267-279, 2014. https://doi.org/10.1109/TETC.2014.2330519

$k-$Means Clustering

The General Idea

The k-means algorithm clusters data by trying to separate samples in n groups of equal variance, minimizing a criterion known as the **inertia** or **within-cluster sum-of-squares** (see below). This algorithm requires the **number of clusters to be specified**.

$$\sum_{i=0}^{n} \min_{\mu_j \in C}(||x_i - \mu_j||^2)$$

Inertia is a measure of how internally coherent clusters are; however, it suffers from various drawbacks:

- Inertia makes the assumption that clusters are convex and isotropic, which is not always the case. It responds poorly to elongated clusters, or manifolds with irregular shapes.
- Inertia is not a normalized metric: we just know that lower values are better and zero is optimal. But in very high-dimensional spaces, Euclidean distances tend to become inflated.

Source: Clustering — scikit-learn 1.0.2 documentation https://scikit-learn.org/stable/modules/clustering.html#k-means

The Steps of the k-means Algorithm

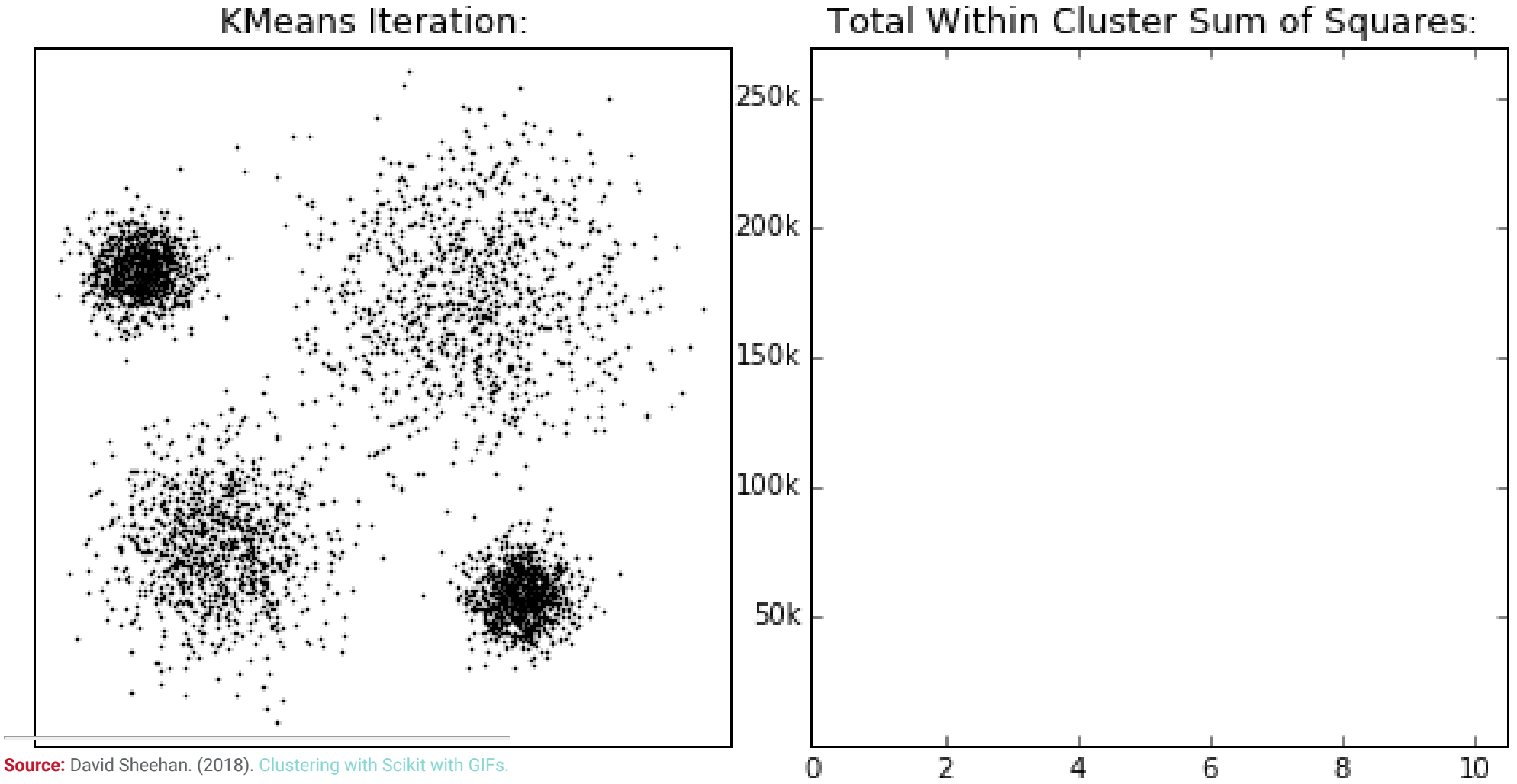
In basic terms, the algorithm has three steps.

- Step 0 chooses the initial centroids, with the most basic method being to choose $k$samples from the dataset X . After initialization, k-means consists of looping between the remaining two steps.
- Step 1 assigns each sample to its nearest centroid.
- Step 2 creates new centroids by taking the mean value of all of the samples assigned to each previous centroid. The difference between the old and the new centroids are computed.

  The algorithm repeats these last two steps the centroids do not move significantly.

Source: Clustering — scikit-learn 1.0.2 documentation https://scikit-learn.org/stable/modules/clustering.html#k-means

The k-means Algorithm: A Visual Example



Source: David Sheehan. (2018). Clustering with Scikit with GIFs.

A Demonstration of k-means Clustering

Use the k-means algorithm to cluster the following observations. Use $k = 2$ and Euclidean distance. **Use this handout to go through the k-means algorithm implementation (by hand).**

Observation	X1	X2
1	1.0	1.0
2	1.5	2.0
3	3.0	4.0
4	5.0	7.0
5	3.5	5.0
6	4.5	5.0
7	3.5	4.5

Practical Issues with k-means Clustering

Data | Prep | k-means (k=3) | Optimal (k) | Viz Clusters

```
## # A tibble: 344 × 8
##    species island    bill_length_mm bill_depth_mm flipper_length_mm body_mass_g
##    <fct>   <fct>              <dbl>         <dbl>             <int>       <int>
##  1 Adelie  Torgersen           39.1          18.7               181        3750
##  2 Adelie  Torgersen           39.5          17.4               186        3800
##  3 Adelie  Torgersen           40.3          18                 195        3250
##  4 Adelie  Torgersen           NA            NA                  NA          NA
##  5 Adelie  Torgersen           36.7          19.3               193        3450
##  6 Adelie  Torgersen           39.3          20.6               190        3650
##  7 Adelie  Torgersen           38.9          17.8               181        3625
##  8 Adelie  Torgersen           39.2          19.6               195        4675
##  9 Adelie  Torgersen           34.1          18.1               193        3475
## 10 Adelie  Torgersen           42            20.2               190        4250
## # ℹ 334 more rows
## # ℹ 2 more variables: sex <fct>, year <int>
```

Practical Issues with k-means Clustering

Data | Prep | k-means (k=3) | Optimal (k) | Viz Clusters

```
## # A tibble: 342 × 5
##    species bill_length_mm[,1] bill_depth_mm[,1] flipper_length_mm[,1]
##    <fct>                <dbl>             <dbl>                  <dbl>
##  1 Adelie              -0.883            0.784                 -1.42
##  2 Adelie              -0.810            0.126                 -1.06
##  3 Adelie              -0.663            0.430                 -0.421
##  4 Adelie              -1.32             1.09                  -0.563
##  5 Adelie              -0.847            1.75                  -0.776
##  6 Adelie              -0.920            0.329                 -1.42
##  7 Adelie              -0.865            1.24                  -0.421
##  8 Adelie              -1.80             0.480                 -0.563
##  9 Adelie              -0.352            1.54                  -0.776
## 10 Adelie              -1.12            -0.0259                -1.06
## # ℹ 332 more rows
## # ℹ 1 more variable: body_mass_g <dbl[,1]>
```

Practical Issues with k-means Clustering

Data | Prep | k-means (k=3) | Optimal (k) | Viz Clusters

```
##
##                1   2   3
##   Adelie       0   0 151
##   Chinstrap    0   1  67
##   Gentoo      66  57   0
```

Practical Issues with k-means Clustering

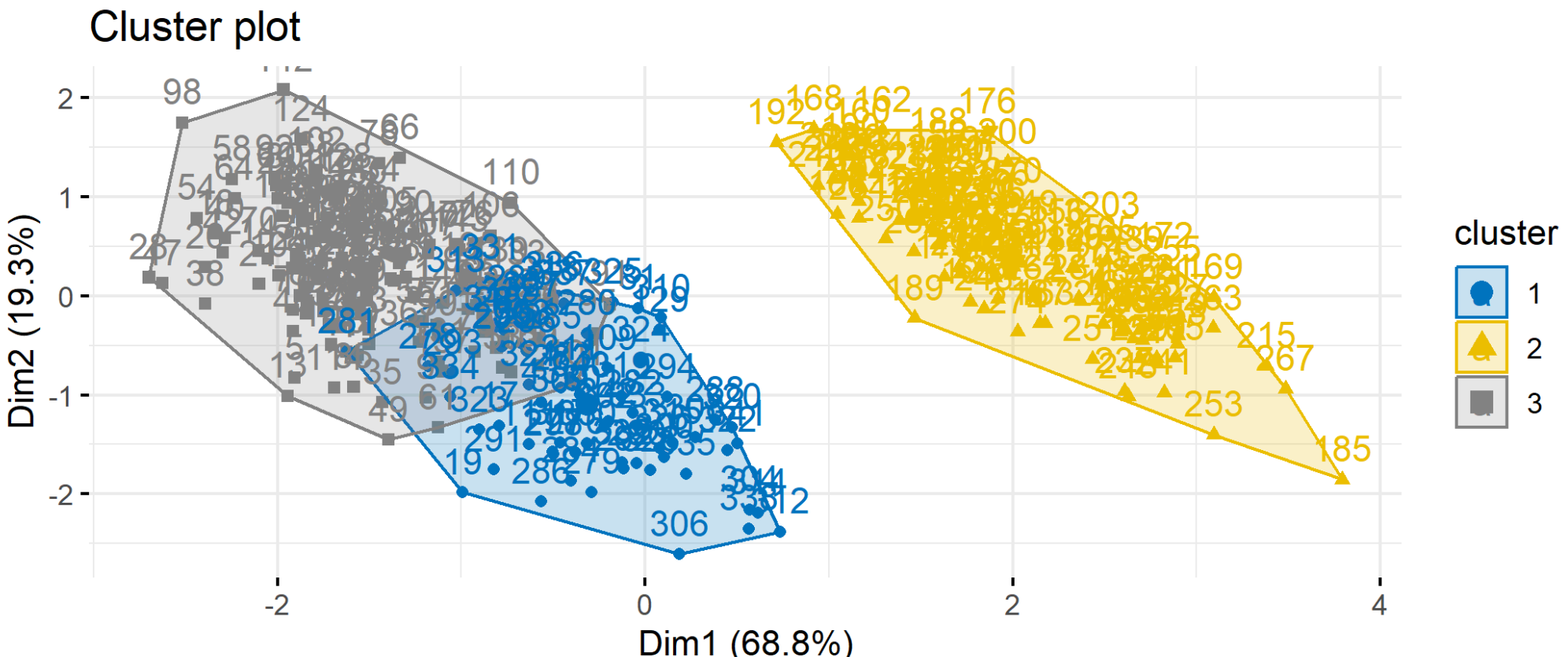
Data | Prep | k-means (k=3) | Optimal (k) | Viz Clusters

```
## *** : The Hubert index is a graphical method of determining the number of clusters.
##                 In the plot of Hubert index, we seek a significant knee that corresponds
##                 significant increase of the value of the measure i.e the significant peak
##                 index second differences plot.
##
```

```
## *** : The D index is a graphical method of determining the number of clusters.
##                 In the plot of D index, we seek a significant knee (the significant peak
##                 second differences plot) that corresponds to a significant increase of th
##                 the measure.
##
## *******************************************************************
## * Among all indices:
## * 8 proposed 2 as the best number of clusters
## * 11 proposed 3 as the best number of clusters
## * 1 proposed 4 as the best number of clusters
## * 3 proposed 5 as the best number of clusters
## * 1 proposed 10 as the best number of clusters
```

Practical Issues with k-means Clustering

Data | Prep | k-means (k=3) | Optimal (k) | Viz Clusters



Summary of Practical Issues

- Rescale numeric data prior to k-means implementation. The scaling can be:
  - a z-transformation similar to what we did in the example
  - a 0-1 scaling
  - converting count data into percentage or counts per a certain number of the population
  - etc.
- Use more than one metric to determine k when using k-means clustering
- Your cluster solution is not the end result, you will need to:
  - visualize it in appropriate way (simple representation as in the previous slide, spatially, time-based, etc.)
  - Attempt to explain the cluster membership using an appropriate binomial/multinomial model (e.g., see this analysis)

Clustering Implementation in Python

The `pycaret` Package

```
import pandas as pd
from pycaret.clustering import *
from janitor import clean_names

portmap_df = (
    pd.read_csv('https://raw.githubusercontent.com/fmegahed/isa419/main/data/portmap.csv')
    .sample(n = 1000, random_state = 123)
    .clean_names()
    .dropna()
    .reset_index(drop = True)
    .loc[:, ['_flow_duration',    '_total_fwd_packets',    '_total_backward_packets',    'to
)

s = setup(
    portmap_df[subset_features], session_id = 2024,
    ignore_features= ['_label'],
    preprocess=True,
    normalize = True, normalize_method= 'zscore')

kmeans = create_model('kmeans', num_clusters = 2)
kmeans_results = assign_model(kmeans)

plot_model(kmeans, plot = 'elbow')
plot_model(kmeans, plot = 'cluster', feature = '_label')
```

Other Clustering Methods

Other Clustering Methods

- **Hierarchical Clustering**: This method does not require the number of clusters to be specified. It is a method of cluster analysis which seeks to build a hierarchy of clusters. Strategies for hierarchical clustering generally fall into two types:
  - **Agglomerative**: This is a "bottom-up" approach: each observation starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy.
  - **Divisive**: This is a "top-down" approach: all observations start in one cluster, and splits are performed recursively as one moves down the hierarchy.
- **DBSCAN**: This is a density-based clustering algorithm: given a set of points in some space, it groups together points that are closely packed together (points with many nearby neighbors), marking as outliers points that lie alone in low-density regions.

Other Clustering Methods

- **Memory-based Clustering**: This is a family of clustering algorithms that do not require the entire dataset to be in memory. Examples include:
  - **BIRCH**: This is a hierarchical clustering algorithm that can be used to cluster very large datasets.
  - **CLARANS**: This is a clustering algorithm that can be used to cluster very large datasets.

Recap

Summary of Main Points

By now, you should be able to do the following:

- Define clustering
- Explain the $k-$means clustering algorithm for numeric datasets
- Implement the $k-$means algorithm in Python using the `pycaret` package
- Describe scenarios where other/advanced clustering algorithms are needed

Review and Clarification

- **Class Notes**: Take some time to revisit your class notes for key insights and concepts.
- **Zoom Recording**: The recording of today's class will be made available on Canvas approximately 3-4 hours after the end of class.
- **Questions**: Please don't hesitate to ask for clarification on any topics discussed in class. It's crucial not to let questions accumulate.