DataSci 400

lesson 7: k-means clustering

Seth Mottaghinejad

today's agenda

- what is **unsupervised learning**?
- **k-means** vs. **k-nearest neighbor**
- how k-means works
  - implementation of the algorithm
  - assumptions about k-means
  - difficulties in **interpreting results**
  - examples where k-means was not properly done

unsupervised learning

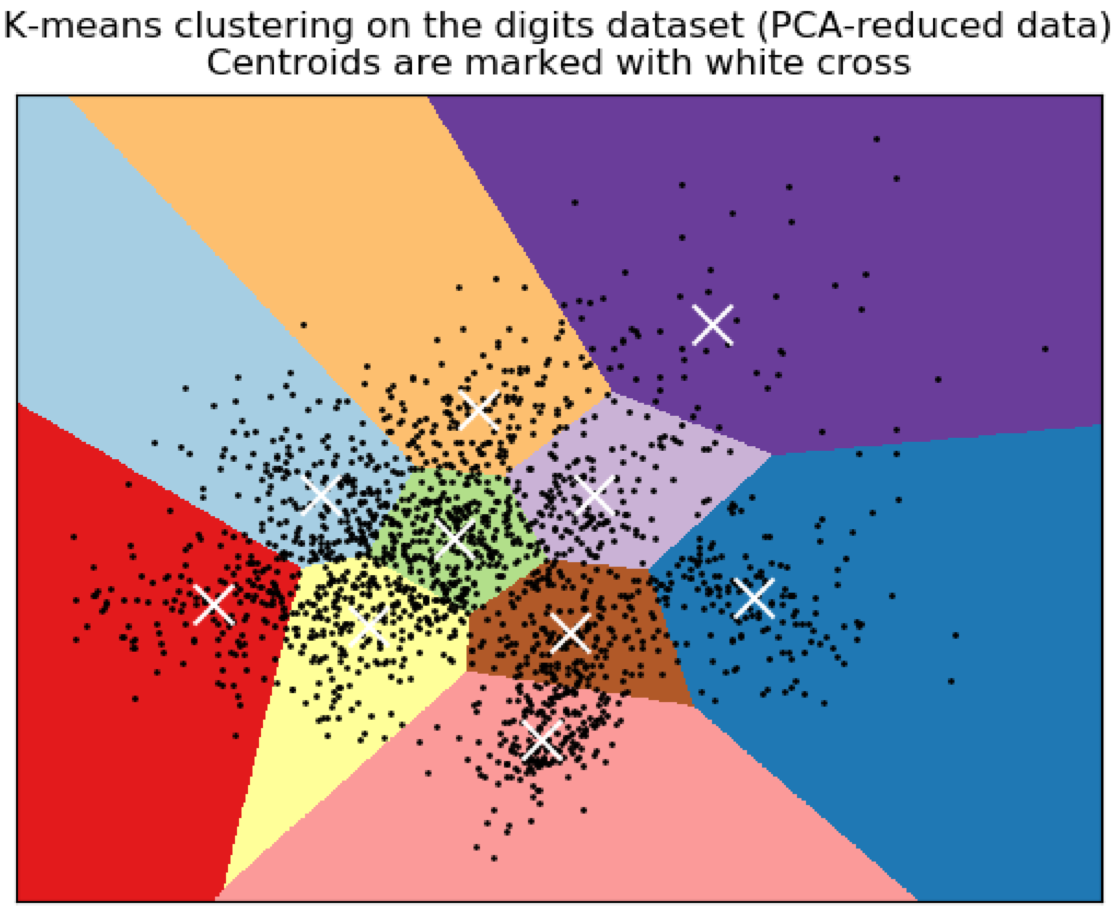
- also called data-mining / pattern recognition / structure discovery
- look at **unlabeled data** and find general patterns
- more subjective and difficult to evaluate and interpret, and hence it is far **less common** than supervised learning
- **clustering** is the most common example
  - k-means clustering
  - variable clustering / dimensionality reduction
  - word clouds (kind of)

k-means characteristics

- there are **no labels**: k-means is **unsupervised**
- clusters are a **construct** we create, not something set in stone
- clusters can be hard to interpret
  - choosing k is mostly subjective
  - lots of **gray areas** when comparing clusters
- there is a **supervised learning** algorithm that is very similar to k-means in how it works, called **k-nearest neighbors**
  - unlike k-means, it can be easily **evaluated**

k-means clutering

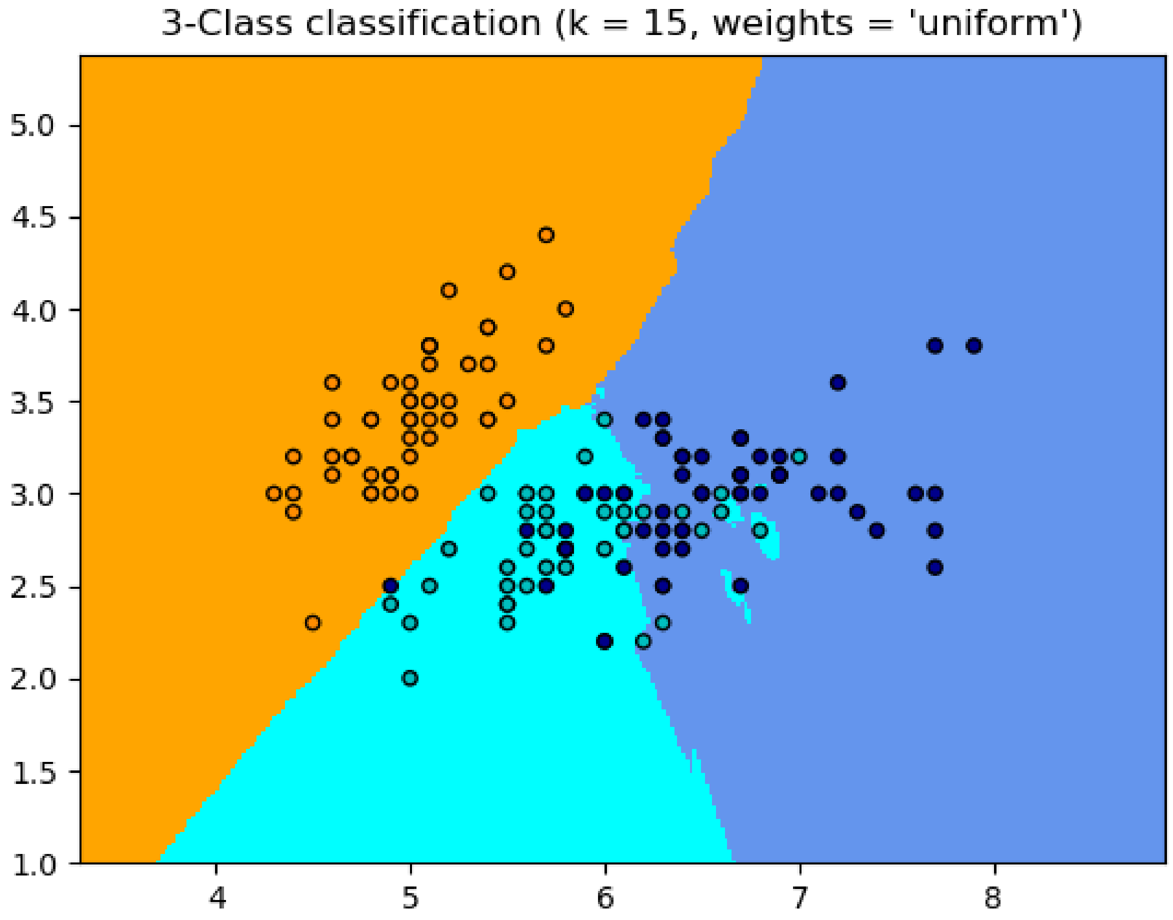
- here we chose $k = 10$
- we have two **numeric features**
- the white crosses are **cluster centroids**
- the colors show which cluster you get **assigned to** based on where you landed



source: https://scikit-learn.org

k-nearest neighbor

- k is the number of **neighbors** to consider
- the colors of the points shows the **labels**
- the colors of regions show **decision boundaries**
- larger k makes decision boundary **smoother**



source: https://scikit-learn.org

k-means algorithm

1. start with k random "centroids" in the **feature space**, preferably spread out well
2. calculate the **Euclidean distance** of every row to each of the k centroids
3. **assign** each row to whichever centroid it is closest to
4. **recalculate** cluster centroids
5. **repeat** steps 2 through 4 until results stabilize

k-means assumption

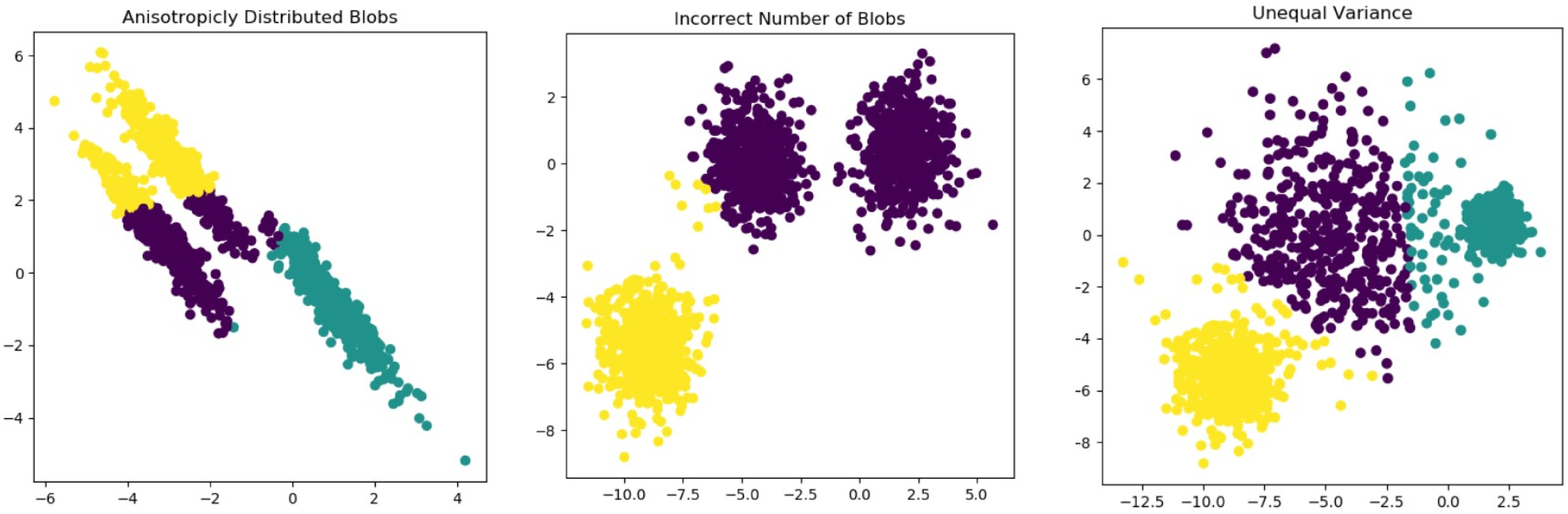
- **Euclidean distance** means that
  - categorical data must be represented numerically
  - numeric data must be **normalized**
- we want to maximize variability **between clusters**
  - i.e. cluster centroids should be far away from each other
- we want to minimize variability **within clusters**
  - i.e. points belonging to the same cluster should be close to the centroid of their cluster

break time

lab time

- in the next slide, you are presented with 3 different situations where k-means **didn't work as intended**
- look at the scatter plot and do your best to **explain** why k-means didn't work as intended in each situation
- propose an **approach** for what to do to **avoid** getting in such a trap
- even though the examples are 2 dimensional, your approach should work even when we have more than 2 features and **cannot** rely on **data visualization**

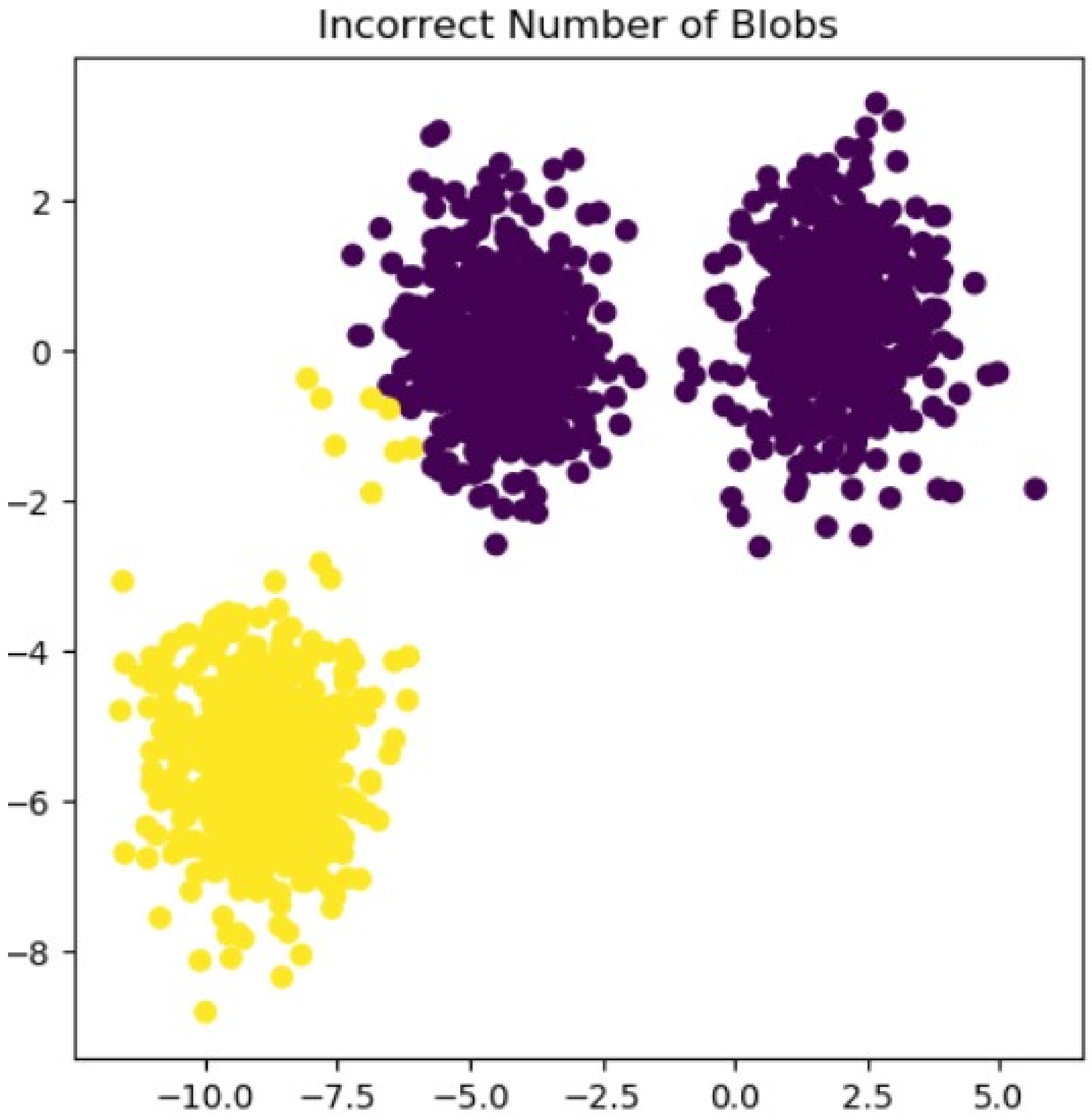
source: https://scikit-learn.org



k-means fail # 1

- we are too **conservative** in our choice of k
- we can catch this by **increasing** k and noticing a big drop in within-cluster **variability**

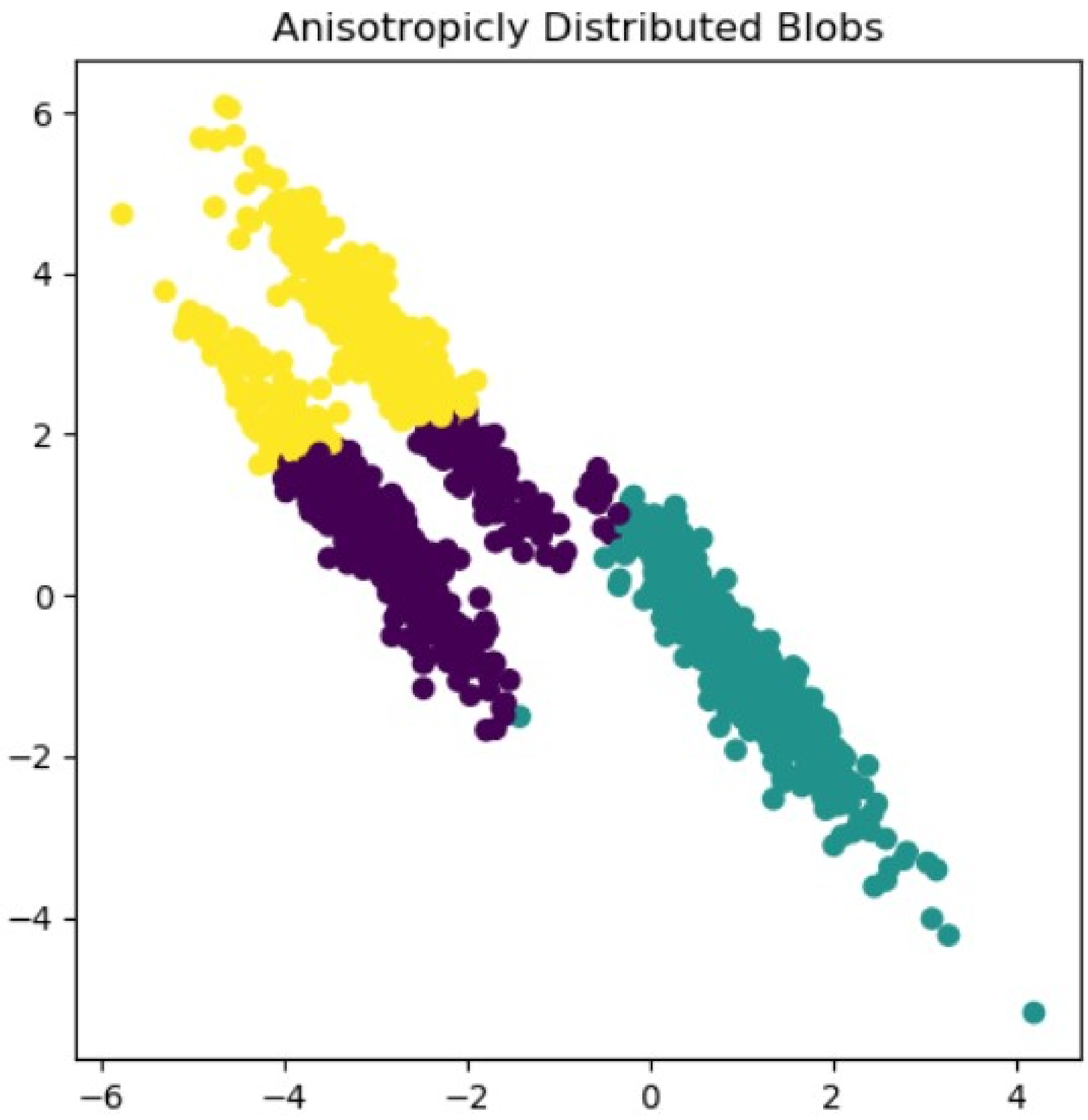
source: https://scikit-learn.org



k-means fail # 2

- data distributions follow slanted shapes
- we can avoid this by **not** including features that are **highly correlated**
- we can try certain **transformations**, e.g. rotation or PCA

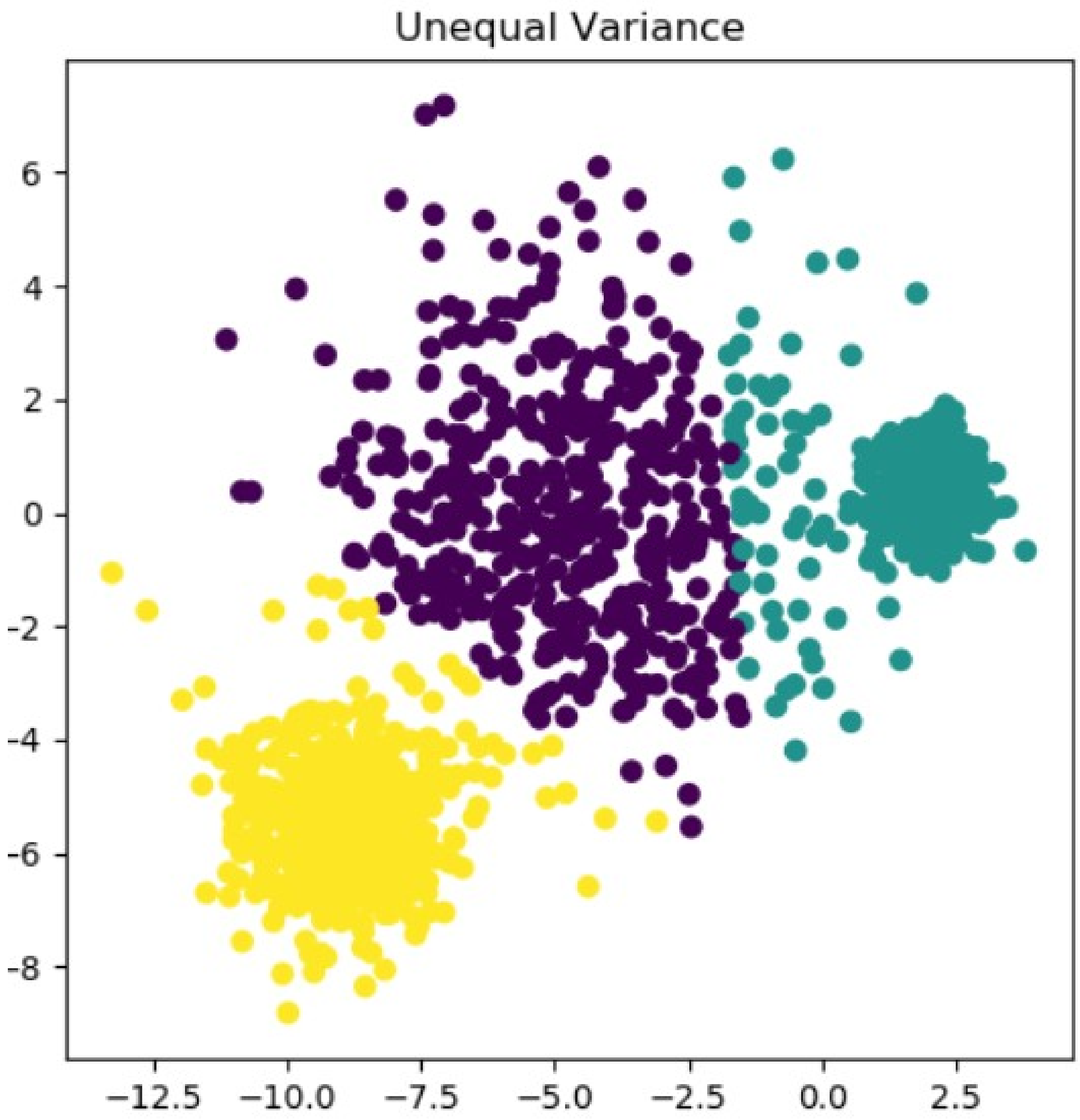
source: https://scikit-learn.org



k-means fail # 3

- the middle cluster looks like it should own more of the points around it
- this is a **tough** one
- we can try to rerun k-means many times and see which cluster the **borderline points** get assigned to most

source: https://scikit-learn.org



the end