Clustering and Dimension Reduction

CS 759/859 Natural Language Processing Lecture 5

Samuel Carton, University of New Hampshire

Text Clustering

Last lecture

New toolkit: Pandas

Concepts:

- Supervised learning for classification
- K-nearest neighbors model
- Underfitting/overfitting
- Bias-variance trade-off
- Hyperparameters
- Evaluation metrics
- Model confidence

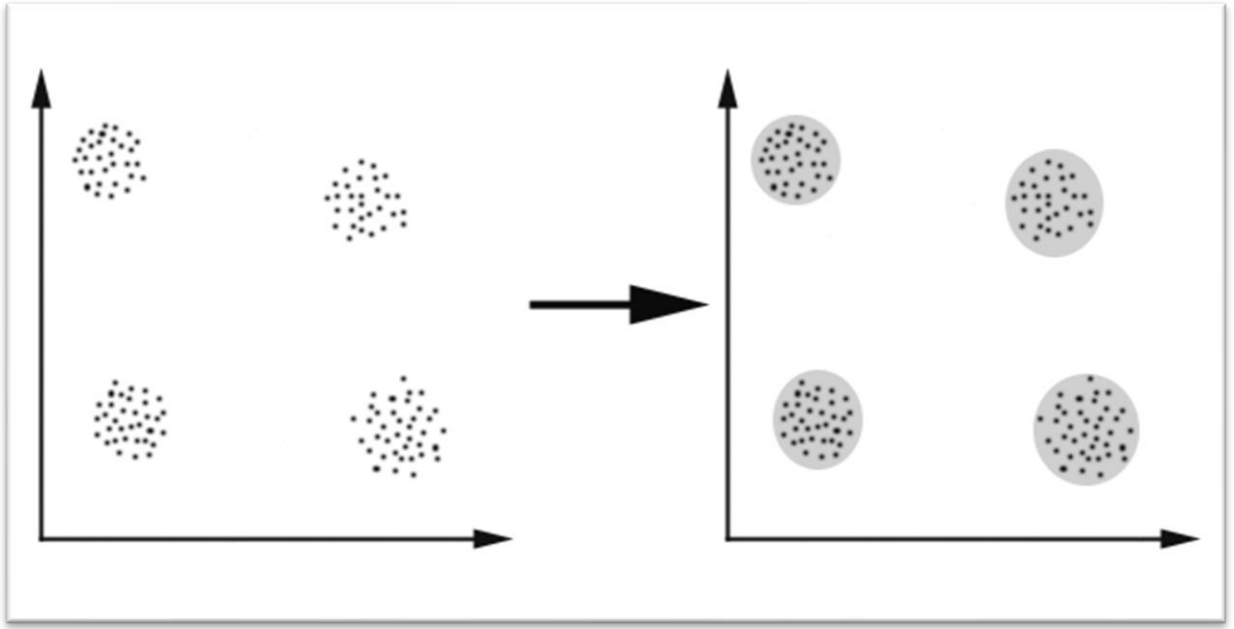
Unsupervised learning for clustering

Clustering: given input text x, cluster x into one of C clusters, along with similar texts

Unsupervised learning: given a dataset X, learn how to predict… something.

- **Clustering**: cluster labels
- **Generation**: new outputs x which belong to the same distribution as the input

Key term: **latent structure**



https://matteucci.faculty.polimi.it/Clustering/tutorial_html/index.html

Clustering for text

When would you want to do clustering on text?

Basically, any time you want to get a high-level understanding of the structure of your corpus.

What are some real-world scenarios where you might want this?

Clustering for text

When would you want to do clustering on text?

Basically, any time you want to get a high-level understanding of the structure of your corpus.

What are some real-world scenarios where you might want this?

- Computational social science/textual analysis
  - E.g. “what are the K basic types of post that exist on this subreddit?”
- Business analytics
  - E.g. “what kinds of things are people saying about my company on Twitter these days?”

Case study: 20-Newsgroups

Classic dataset for text classification and clustering

http://qwone.com/~jason/20Newsgroups/

~20,000 documents, evenly split across 20 newsgroups:

comp.graphics comp.os.ms-windows.misc comp.sys.ibm.pc.hardware comp.sys.mac.hardware comp.windows.x	rec.autos rec.motorcycles rec.sport.baseball rec.sport.hockey	sci.crypt sci.electronics sci.med sci.space
misc.forsale	talk.politics.misc talk.politics.guns talk.politics.mideast	talk.religion.misc alt.atheism soc.religion.christian

20-Newsgroups dataset

Code description

- Downloading and displaying the classic 20-Newsgroup dataset from Scikit-learn

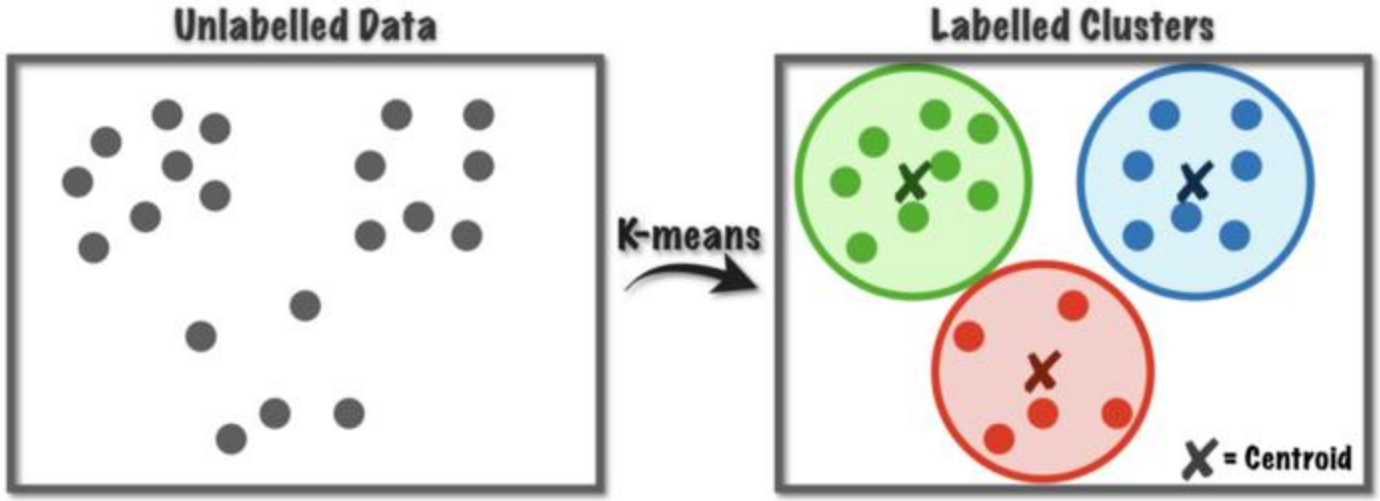
Headings

Clustering

20-Newsgroups dataset

K-means clustering

Basic idea: find K points (aka “means”, aka “centroids”) within the data space that represent centers of cohesive clusters



https://towardsdatascience.com/k-means-a-complete-introduction-1702af9cd8c

K-means clustering

Algorithm:

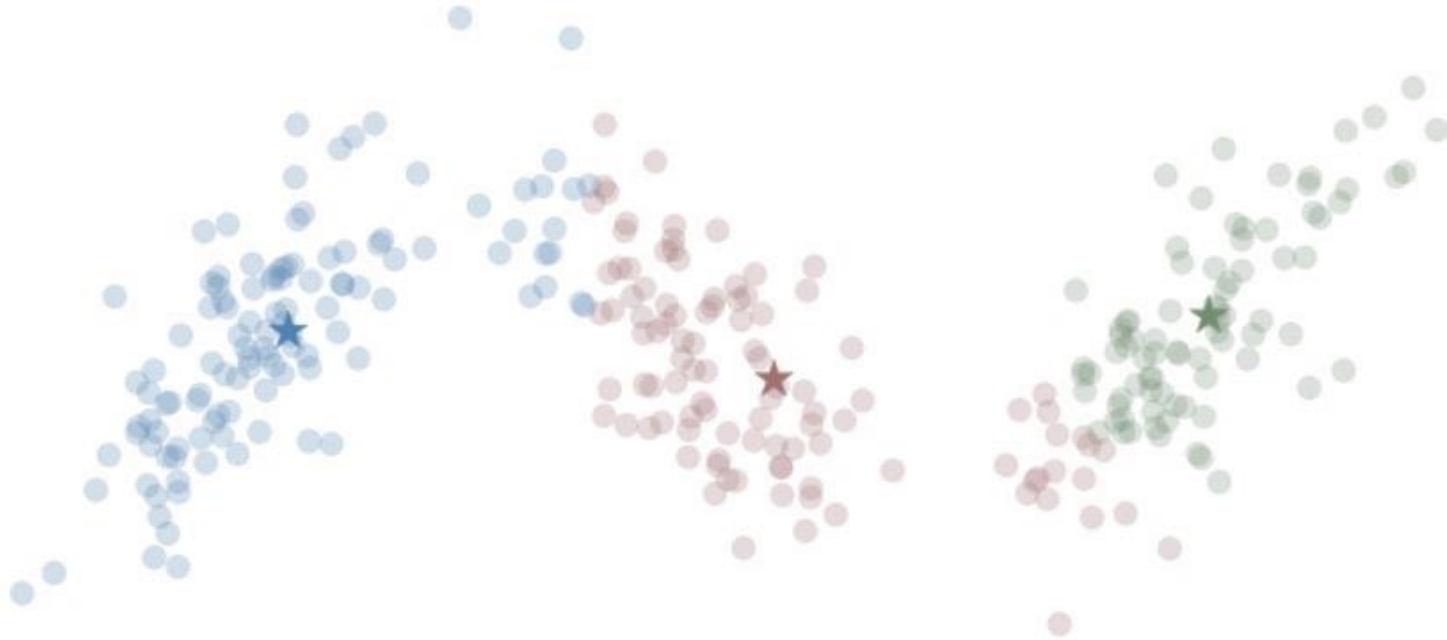
1. Randomly choose K spots within the data space to be initial cluster centers
2. For each point in the data, assign it to the cluster with the closest mean in vector space
   - Implicitly uses Euclidean distance
3. For each cluster center, adjust its position to be the centroid of the data points assigned to it
4. Repeat steps 2 and 3 until some stopping condition is hit
   - Cluster centers stop moving
   - Maximum iterations

K-Means Example

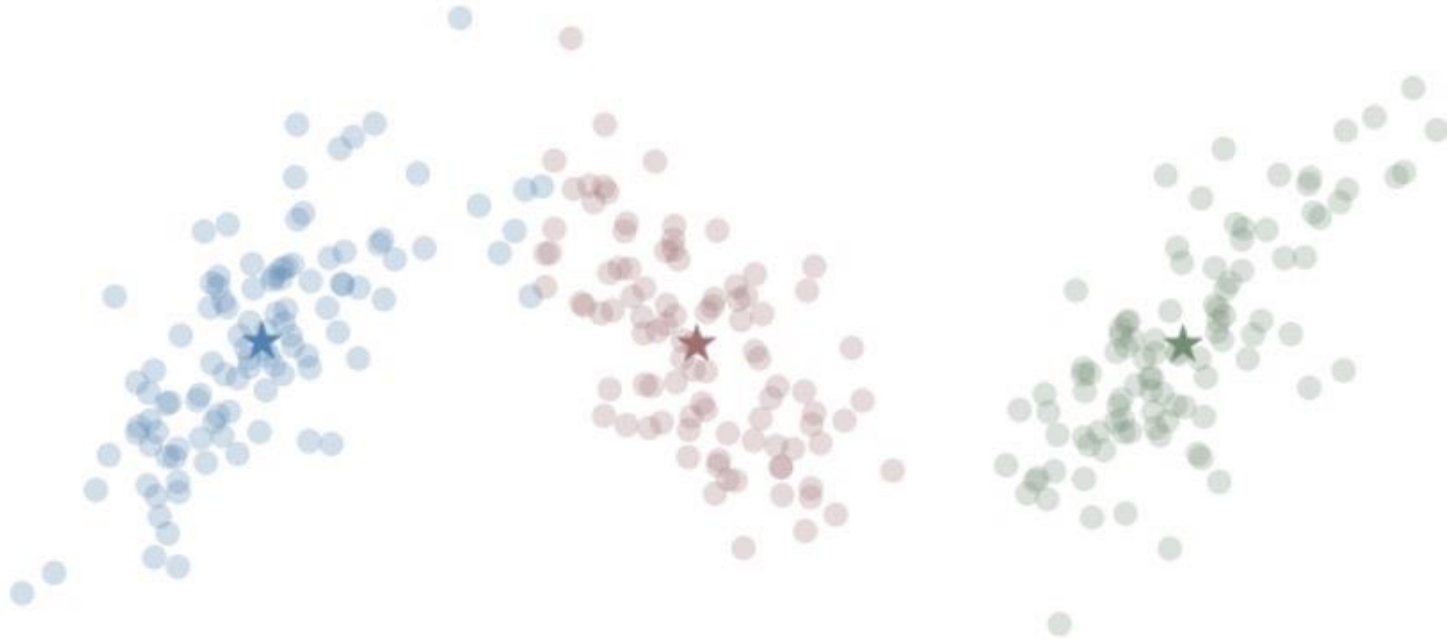
Borrowed from Chenhao Tan

K-Means Example

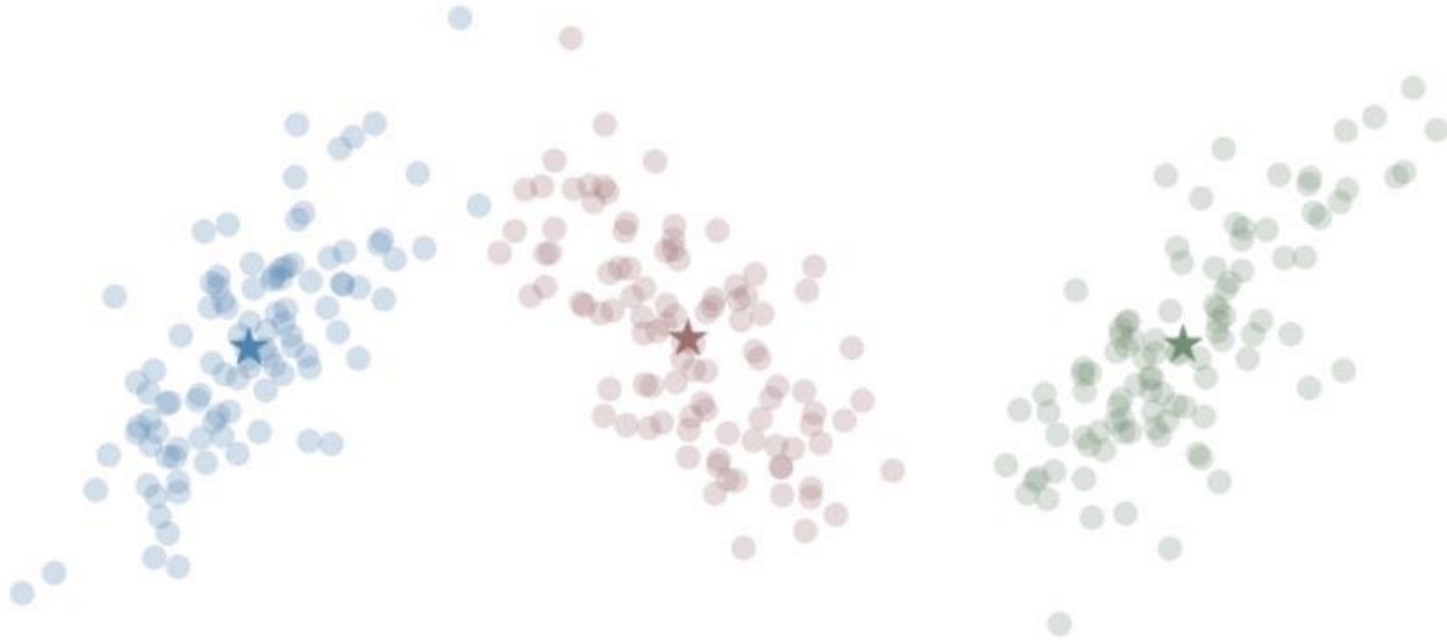
K-Means Example



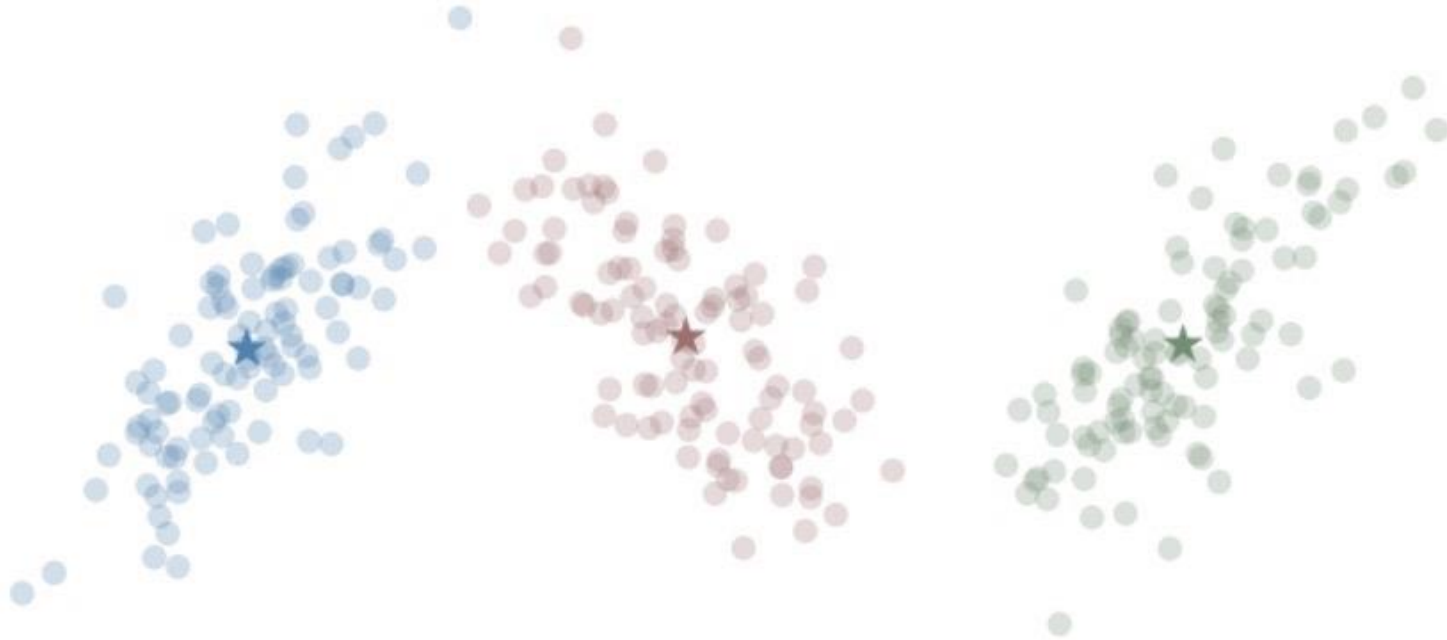
K-Means Example



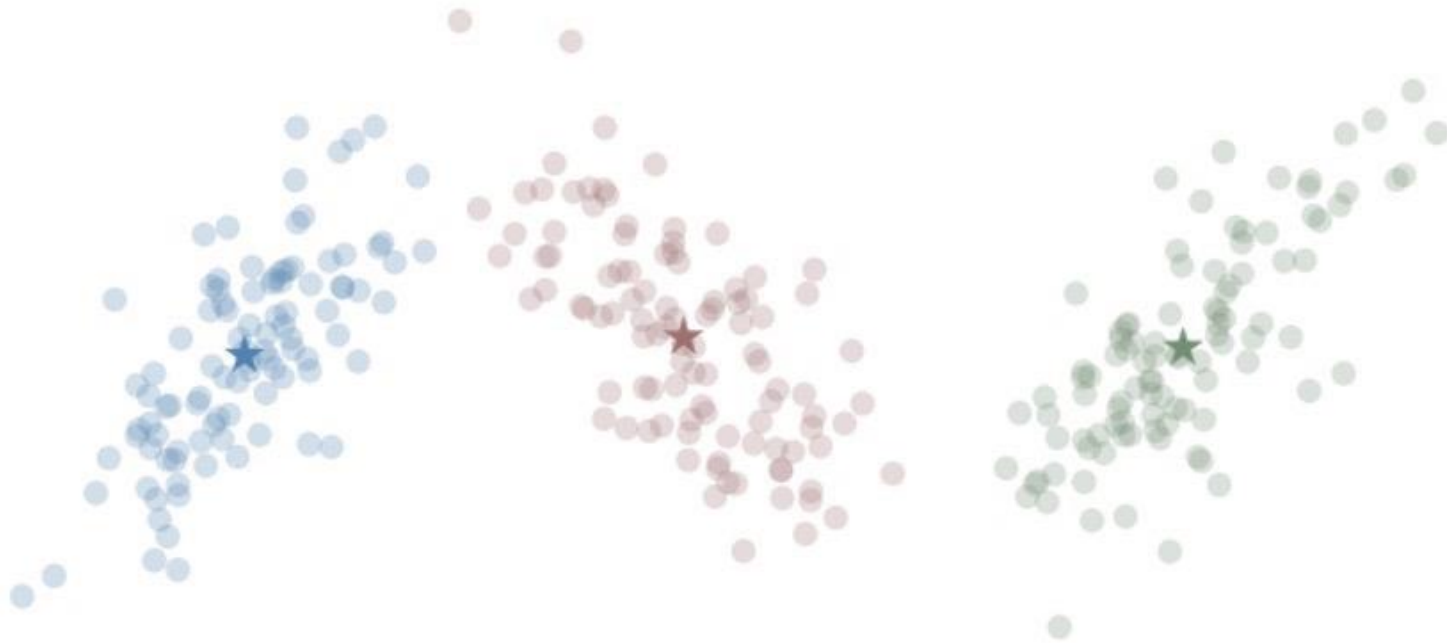
K-Means Example



K-Means Example



K-Means Example



Strengths and weaknesses

Strengths

- Simple
- Intuitive
- Fast

Weaknesses

- Doesn’t work with categorical data
  - Use K-modes instead
- Usually only converges to local minimum
  - Use several random restarts
- Have to determine number of clusters
  - We’ll talk about this in a second
- Can be sensitive to outliers
- Only generates convex clusters

Strengths and weaknesses

Strengths

- Simple
- Intuitive
- Fast

Weaknesses

- Doesn’t work with categorical data
  - Use K-modes instead
- Usually only converges to local minimum
  - Use several random restarts
- Have to determine number of clusters
  - We’ll talk about this in a second
- **Can be sensitive to outliers**
- Only generates convex clusters

Weaknesses - Outlier Sensitivity

Weaknesses - Outlier Sensitivity

Weaknesses - Outlier Sensitivity

Weaknesses - Outlier Sensitivity

Strengths and weaknesses

Strengths

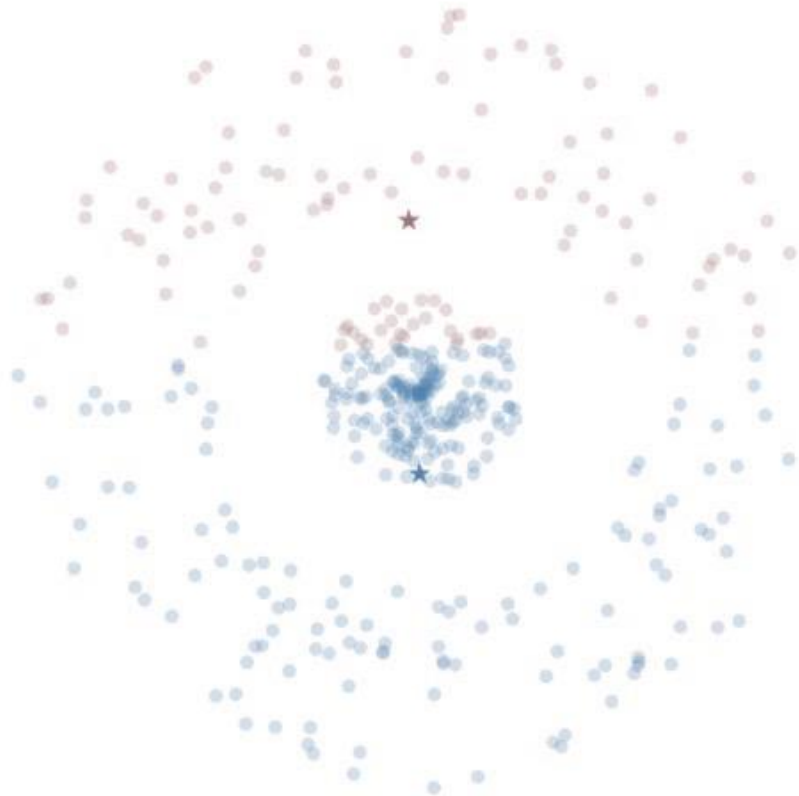
- Simple
- Intuitive
- Fast

Weaknesses

- Doesn’t work with categorical data
  - Use K-modes instead
- Usually only converges to local minimum
  - Use several random restarts
- Have to determine number of clusters
  - We’ll talk about this in a second
- Can be sensitive to outliers
- **Only generates convex clusters**

Weaknesses - Convex Clusters

Weaknesses - Convex Clusters



Preprocessing and clustering

Code description

- Preprocessing the 20-Newsgroup data
- Training a K-means clustering model on it
- Displaying some of the output

Headings

K-means clustering

Preprocessing and vectorization

Building the model

Assessing cluster quality

Extrinsic measures: We have some ground-truth clusters to compare with?

- Why can't we just use classification metrics like accuracy, F1, etc?
- **Mutual information**

Intrinsic measures: No ground-trust cluster assignments available.

- What can we even do? Can we do anything?
- **Silhouette coefficient**

https://scikit-learn.org/stable/modules/clustering.html#clustering-performance-evaluation

Mutual Information

Basic idea: given two discrete random variables, how much does knowing the value of the one tell you about the other?

$$\mathrm{MI}(U,V) = \sum_{i=1}^{|U|}\sum_{j=1}^{|V|} P(i,j)\log\left(\frac{P(i,j)}{P(i)P'(j)}\right)$$

https://scikit-learn.org/stable/modules/clustering.html#mutual-info-score

Actually a measure of pairwise **entropy**

Entropy

A measure of **uncertainty** in a discrete probability distribution

$$H(U) = -\sum_{i=1}^{|U|} P(i)\log(P(i))$$

https://scikit-learn.org/stable/modules/clustering.html#mutual-info-score

High value if distribution is spread out over possible outcomes, low value if it is concentrated in one outcome

Example: think about a fair coin $p_{fair} = (0.5,0.5)$ versus a trick coin$p_{trick} = (0.9,0.1)$

- H(fair) = 0.5*log(0.5)) + 0.5*log(0.5) = -.347 + -.347 = -.693
- H(trick) = 0.9*log(0.9)) + 0.1*log(0.1) = -.094 + -.230 = -.325

Mutual Information

Generalizes entropy to joint distribution of two variables

High value if joint probability is concentrated in one pair of outcomes, low value if it is spread out across pairs

Only defined if we have another variable to compare our clusters to (i.e., when we have ground-truth labels available)

$$\mathrm{MI}(U,V) = \sum_{i=1}^{|U|}\sum_{j=1}^{|V|} P(i,j)\log\left(\frac{P(i,j)}{P(i)P'(j)}\right)$$

https://scikit-learn.org/stable/modules/clustering.html#mutual-info-score

Normalized mutual information

Normalized mutual information normalizes mutual information to fall between 0 (maximum possible pairwise entropy) and 1 (minimum possible pairwise entropy)

Preferable over un-normalized because it can be interpreted visually

$$\mathrm{NMI}(U, V) = \frac{\mathrm{MI}(U, V)}{\mathrm{mean}(H(U), H(V))}$$

https://scikit-learn.org/stable/modules/clustering.html#mutual-info-score

Mutual information

Code description

- Showing mutual information scores for toy examples
- Showing mutual information scores for our clusters generated over the 20-Newsgroup data

Headings

Assessing cluster quality

Mutual information

Toy example

Our clustering

Silhouette coefficient

Intrinsic measure: doesn't require any ground-truth clusters

Basic idea: measures the extent to which data points are **close** to points in the same cluster and **far away** from points in other clusters

- Rewards tight, well-separated clusters

Formula:

$$s = \frac{b - a}{max(a, b)}$$

- **a**: The mean distance between a sample and all other points in the same class.
- **b**: The mean distance between a sample and all other points in the *next nearest cluster*.

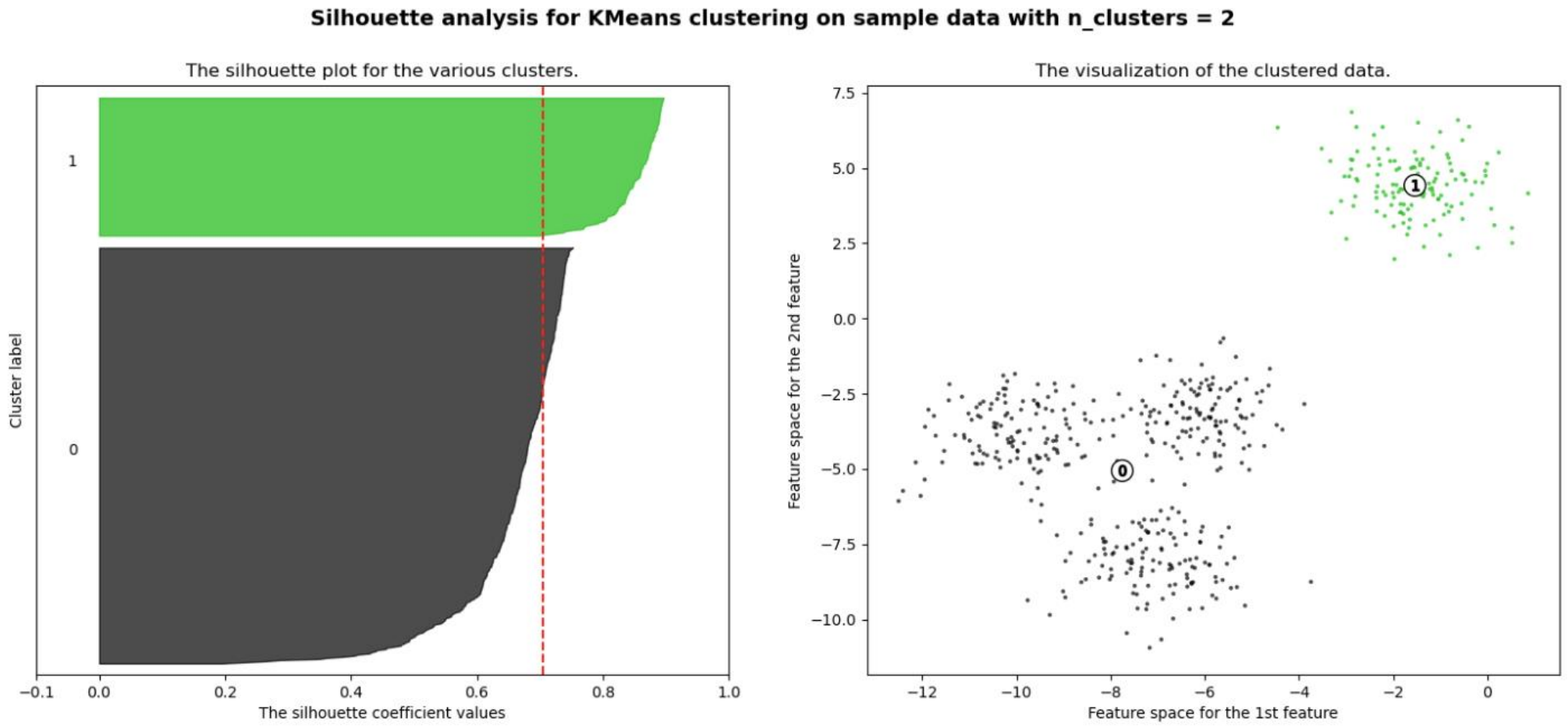
Can be defined for a single sample, or averaged over entire cluster(or entire dataset)

Silhouette analysis to choose K

Basic idea: Visualize silhouette values for each cluster, and choose K such that as many clusters as possible have as many points as possible with high silhouette coefficients.

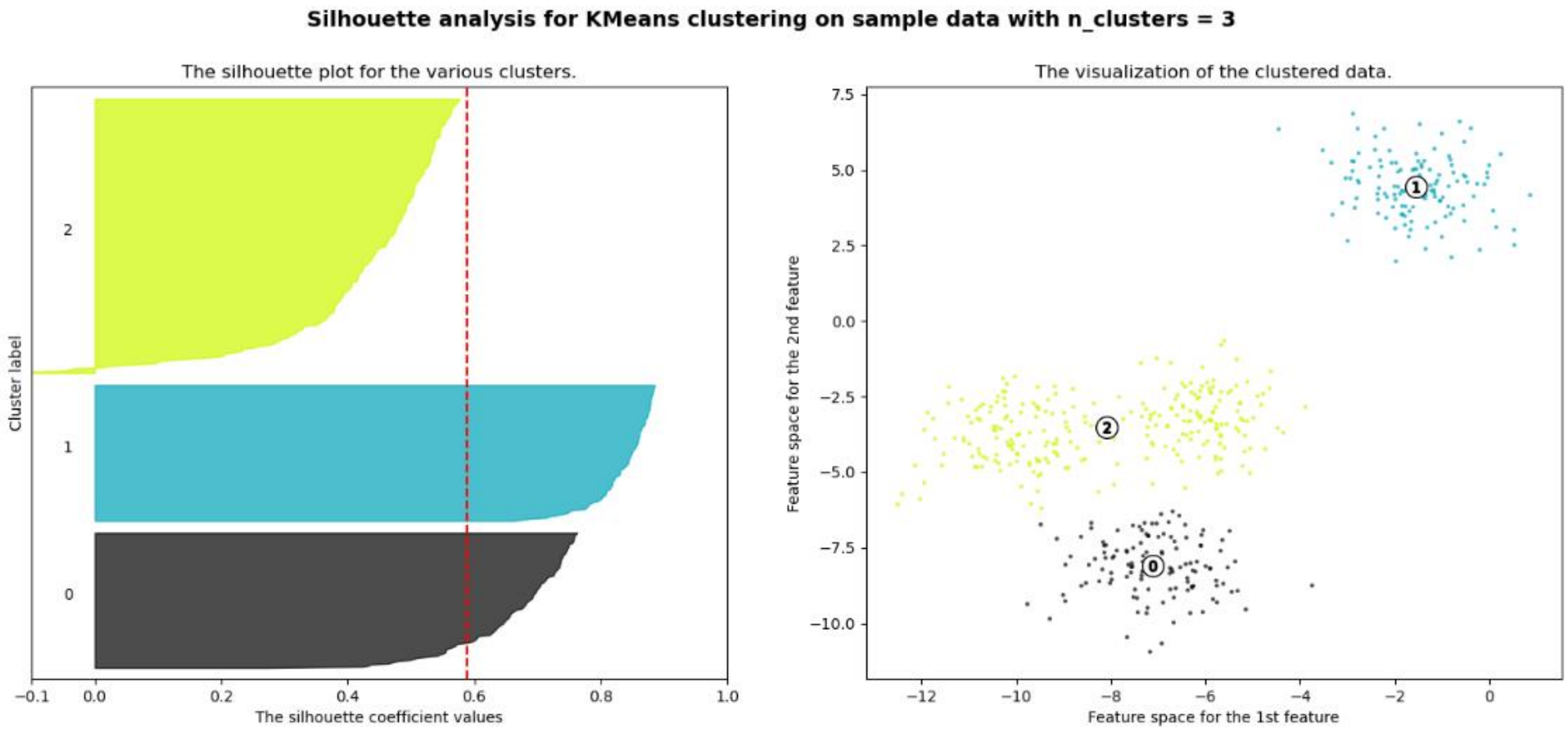
Still a lot of eyeballing involved

Silhouette analysis to choose K



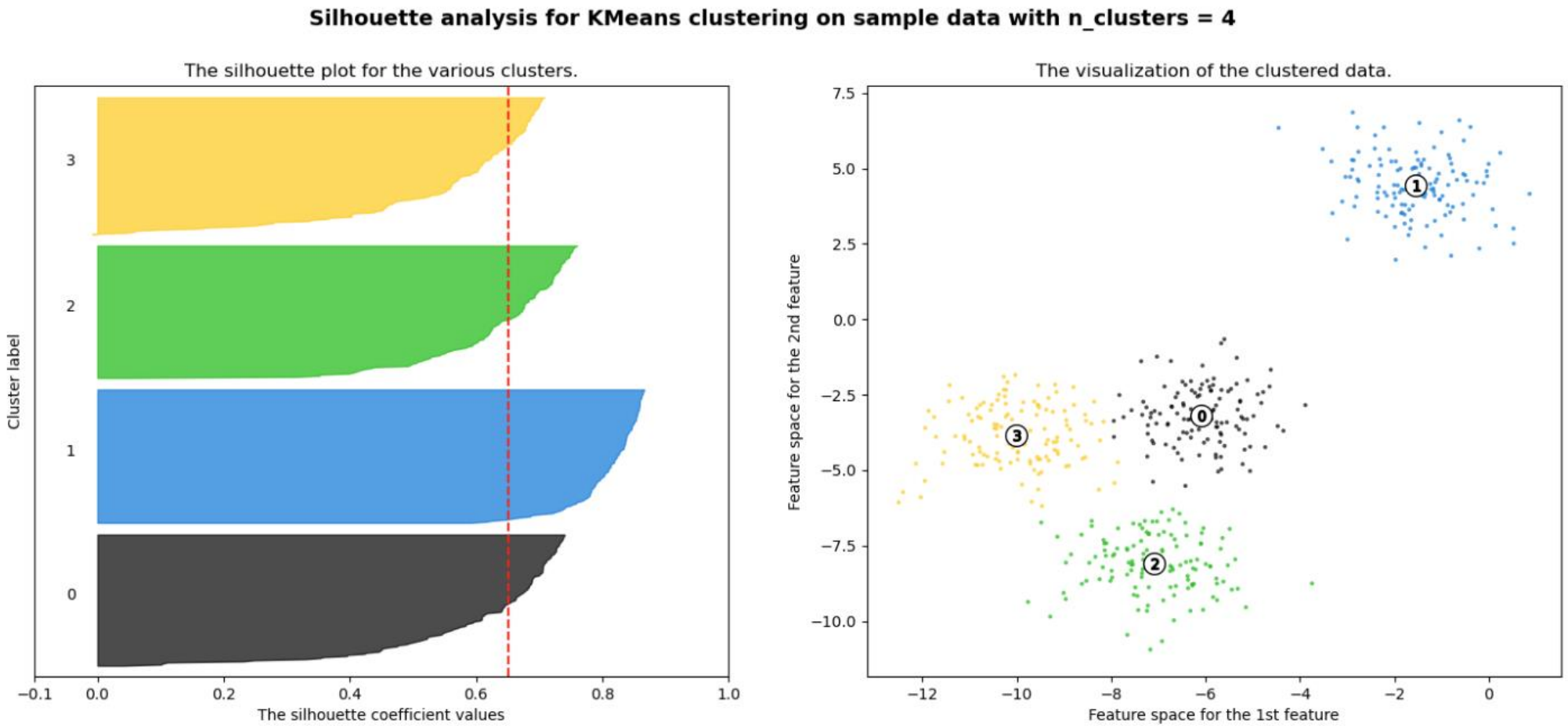
https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html

Silhouette analysis to choose K



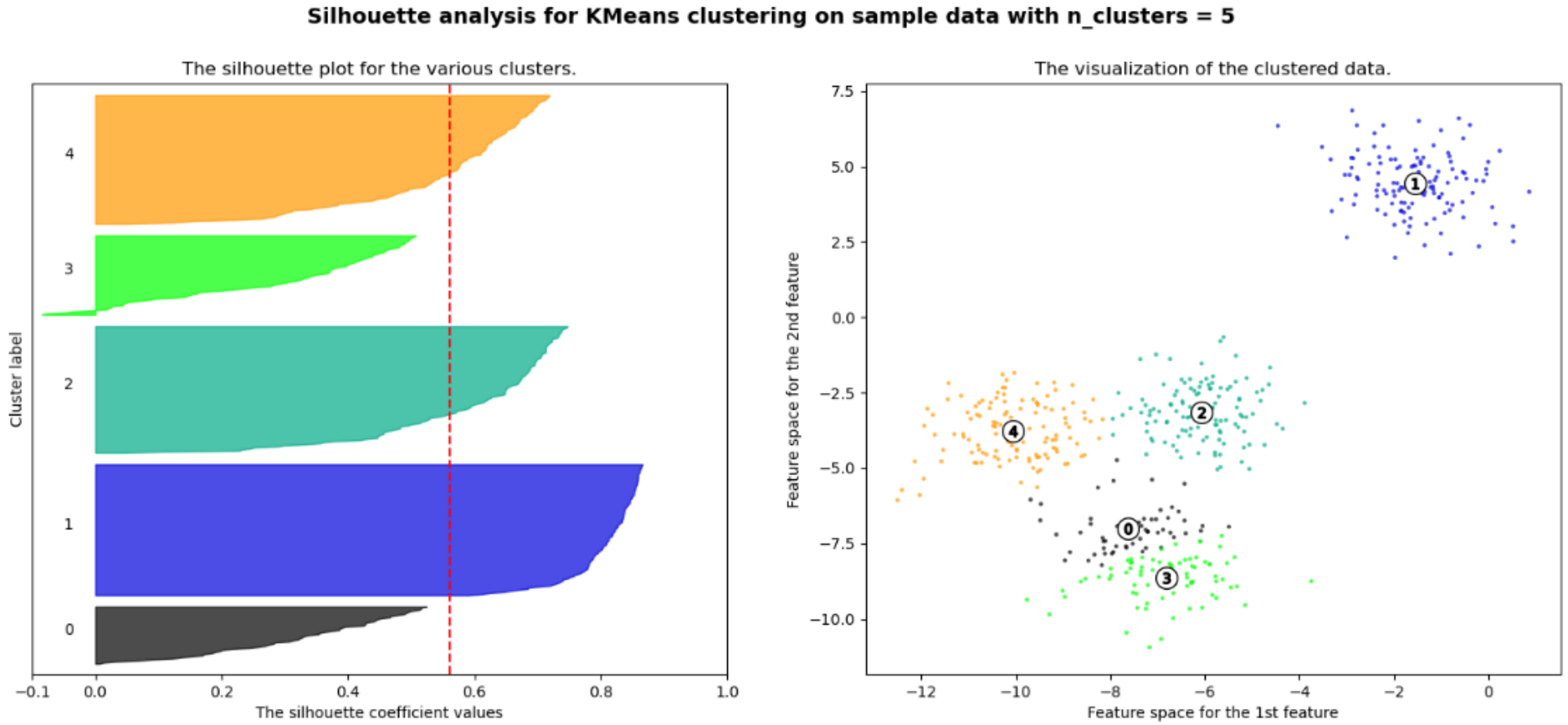
https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html

Silhouette analysis to choose K



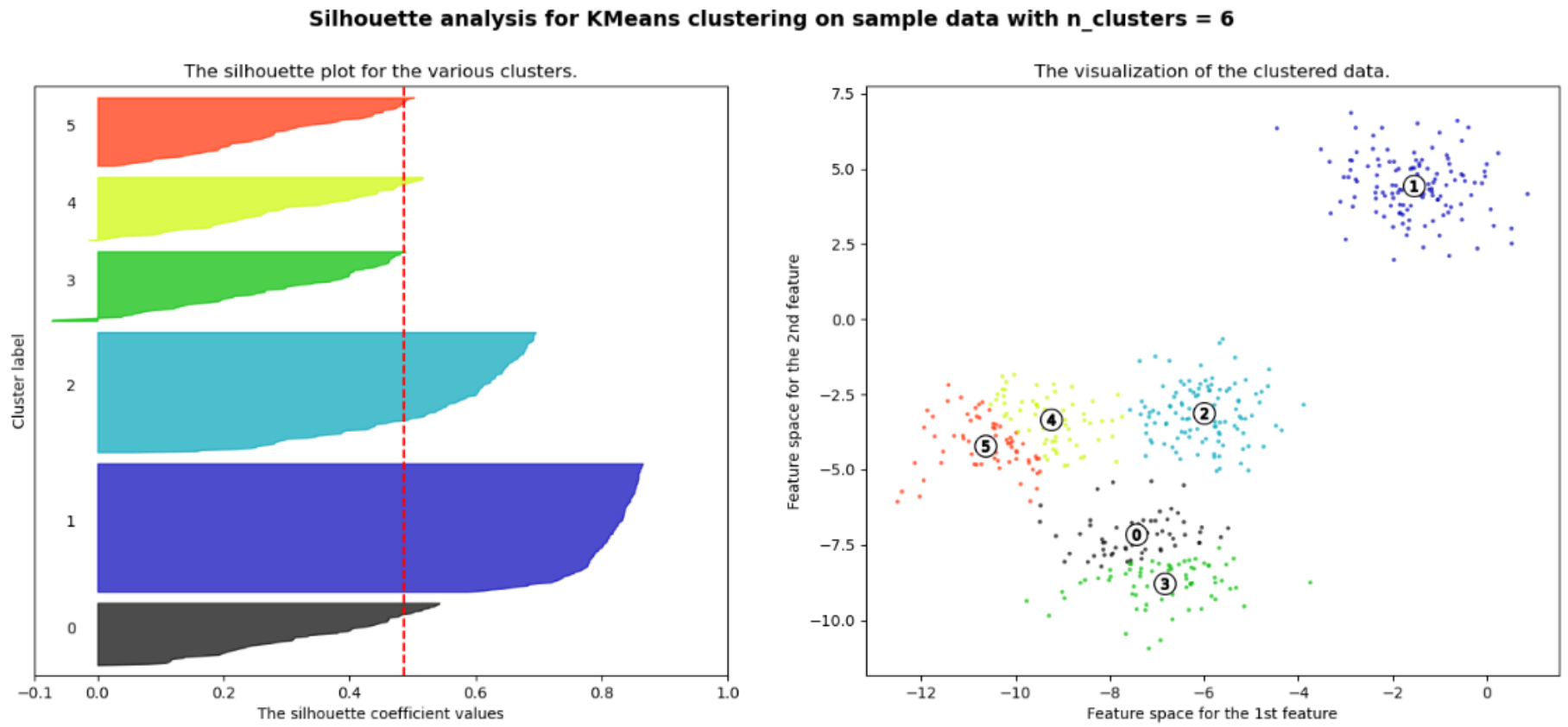
https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html

Silhouette analysis to choose K



https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html

Silhouette analysis to choose K



https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html

Silhouette coefficient

Code description

- Silhouette analysis on toy examples
- Silhouette analysis on our clusters

Headings

Silhouette coefficient

Silhouette analysis

Toy example

Our clustering

Representing individual text clusters

If we want to represent individual text clusters, how should we do it?

Just pick the top N most frequent words?

- Likely to be stop-words or otherwise uninteresting

TF-IDF to the rescue!

- Treat entire cluster as document, find words that occur frequently in that cluster relative to the corpus as a whole

Displaying clusters

Code description

- Using principles of TF-IDF to visualize the learned clusters in our data

Headings

Displaying clusters

Top terms associated with the actual groups

Top terms associated with the clusters we found

Why is evaluating clusters so hard

Compared to (supervised) classification, it seems very difficult to evaluate (unsupervised) clusters. Why?

Because the general problem is underspecified.

What is your overall goal when running your clustering algorithm?

- Customize your evaluation relative to that goal

Case study: Understanding a corpus

Goal: Run clustering algorithm over an unknown text corpus to understand it

- "Looks like 30% of these are job ads, 20% are sports articles, …"

Evaluation:

- Clusters should be individually understandable
  - Create hypothesis for each cluster, sample 20 texts and see if they fit
  - "We find that 17/20 texts in the 'sports article' cluster really are sports articles"
- Clusters should agree with information you do happen to know
  - E.g. "sports articles" cluster should be contained within known "articles" group
  - Mutual information with known groups

Case study: Expanding training data

Goal: Imagine that you have **some** examples of a phenomenon like hate speech, but you're worried it isn't a representative sample, so you want to discover more.

Procedure:

1. Run clustering algorithm
2. See which clusters your known examples are in
3. See if other items in those clusters are

Evaluation

- Existing examples should probably be clustered together (so, mutual information)
- Then maybe manually label samples of clusters they are in, to estimate how many new examples you are finding?

Very ad-hoc, but that's life!

Concluding thoughts

Lots of clustering algorithms out there:
https://scikit-learn.org/stable/modules/clustering.html#clustering

Some methods pick K, others require it as a hyperparameter

Always hard to tell when you have “good” clusters

Combines well with dimension reduction, which we’ll talk about next class

- Especially for visualization purposes