k-means clustering

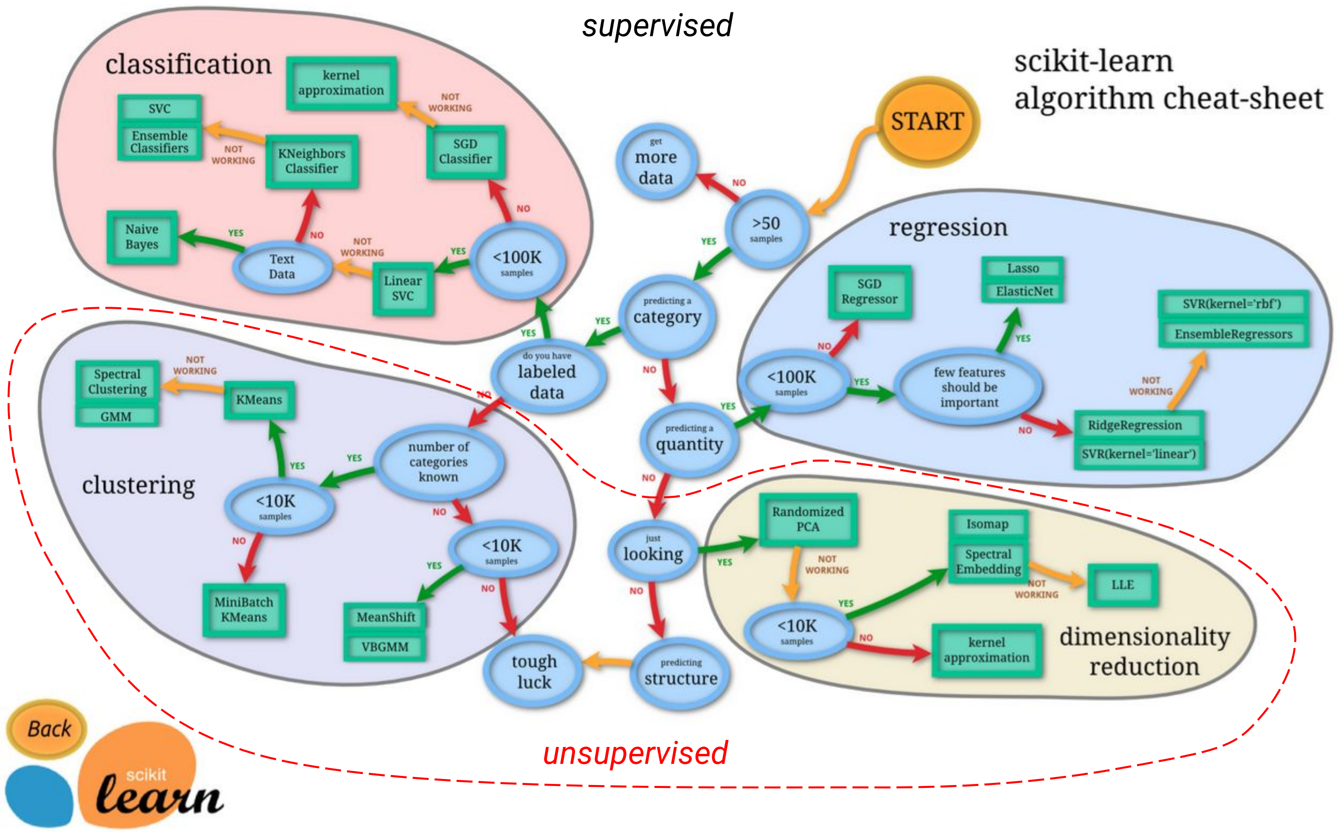
Objectives

After this lecture you will be able to:

- Define *unsupervised learning*
- Define *clustering*
- Provide scenarios where you would want to use *clustering*
- Code your own *k-means* algorithm (if you want)
- Use *elbow* and *silhouette* plots to determine the “right” number of clusters for your data
- Give some practical advice about using *k-means*

Unsupervised learning

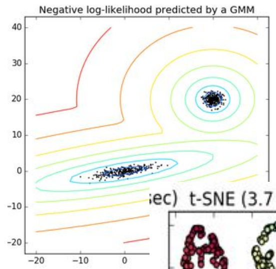
- There is no target/label
- The goal is to find underlying structure or organization
- Determining if you've found the "right" solution is a task unto itself (cross-validation often not applicable)

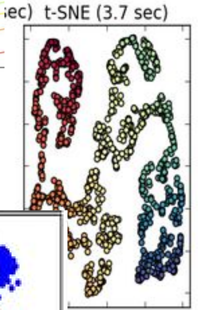


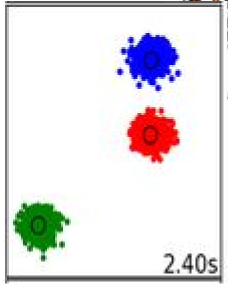
http://scikit-learn.org/stable/tutorial/machine_learning_map/

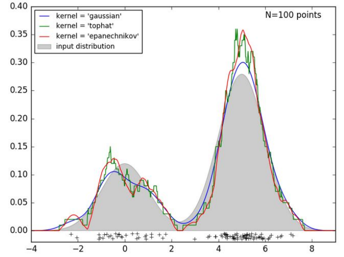
Some unsupervised learning methods

Unsupervised learning method	Description
Gaussian mixture models (GMMs)	A probabilistic model that assumes all the data points are generated from a mixture of Gaussian distributions with unknown parameters.
Manifold learning	An approach to nonlinear dimensionality reduction.
Clustering	The task of grouping objects in such a way that objects in the same cluster are more similar to each other than to those in other clusters.
Decomposing signals into components	Principal Component Analysis (PCA) decomposes a multivariate dataset into successive orthogonal components that explain a maximum amount of the variance. Also Singular Value Decomposition (SVD), Latent Semantic Analysis (LSA), and non-Negative Matrix Factorization (NMF).
Density Estimation	Density estimation is a very simple concept, and most people are already familiar with one common density estimation technique: the histogram.
Neural network models (unsup.)	Restricted Boltzmann machines (RBMs) are unsupervised nonlinear feature learners based on a probabilistic model. RBMs extract features that often give good results when fed into a linear classifier (SVM or a perceptron).





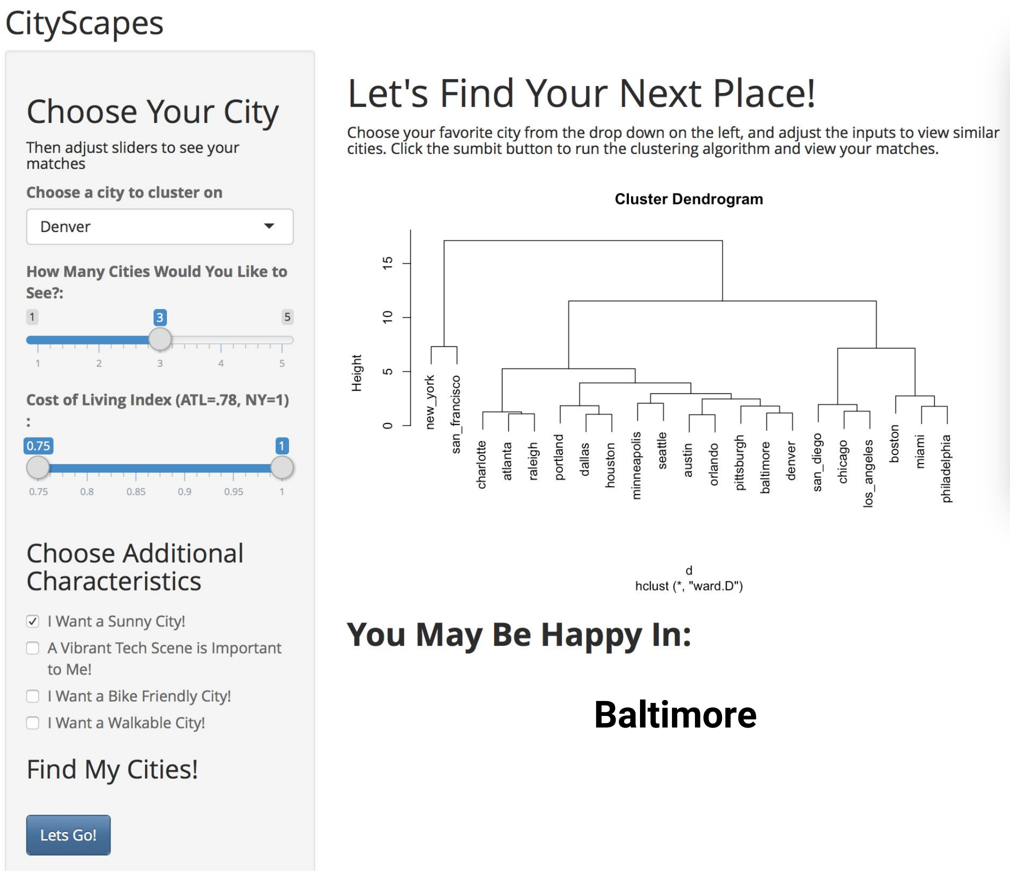




Examples of clustering projects (1)

Features from:

```
- US Census (API and scraper)
- US Bureau of Economic Affairs (scraper)
- Meetup (API)
- Numbeo (scraper)
- Biggest US Cities (scraper)
- RJ Metrics (scraper + OCR)
- Walkingscore (scraper)
- Bureau of Transportation Statisitics (scraper)
- Quandl (API)
- Priceonomics (scraper)
```



CityScapes

Choose Your City

Then adjust sliders to see your matches

Choose a city to cluster on: Denver

How Many Cities Would You Like to See?: 3 (1–5)

Cost of Living Index (ATL=.78, NY=1): 0.75 – 1

Choose Additional Characteristics

- [✓] I Want a Sunny City!
- [ ] A Vibrant Tech Scene is Important to Me!
- [ ] I Want a Bike Friendly City!
- [ ] I Want a Walkable City!

Find My Cities!

Let's Find Your Next Place!

Choose your favorite city from the drop down on the left, and adjust the inputs to view similar cities. Click the sumbit button to run the clustering algorithm and view your matches.

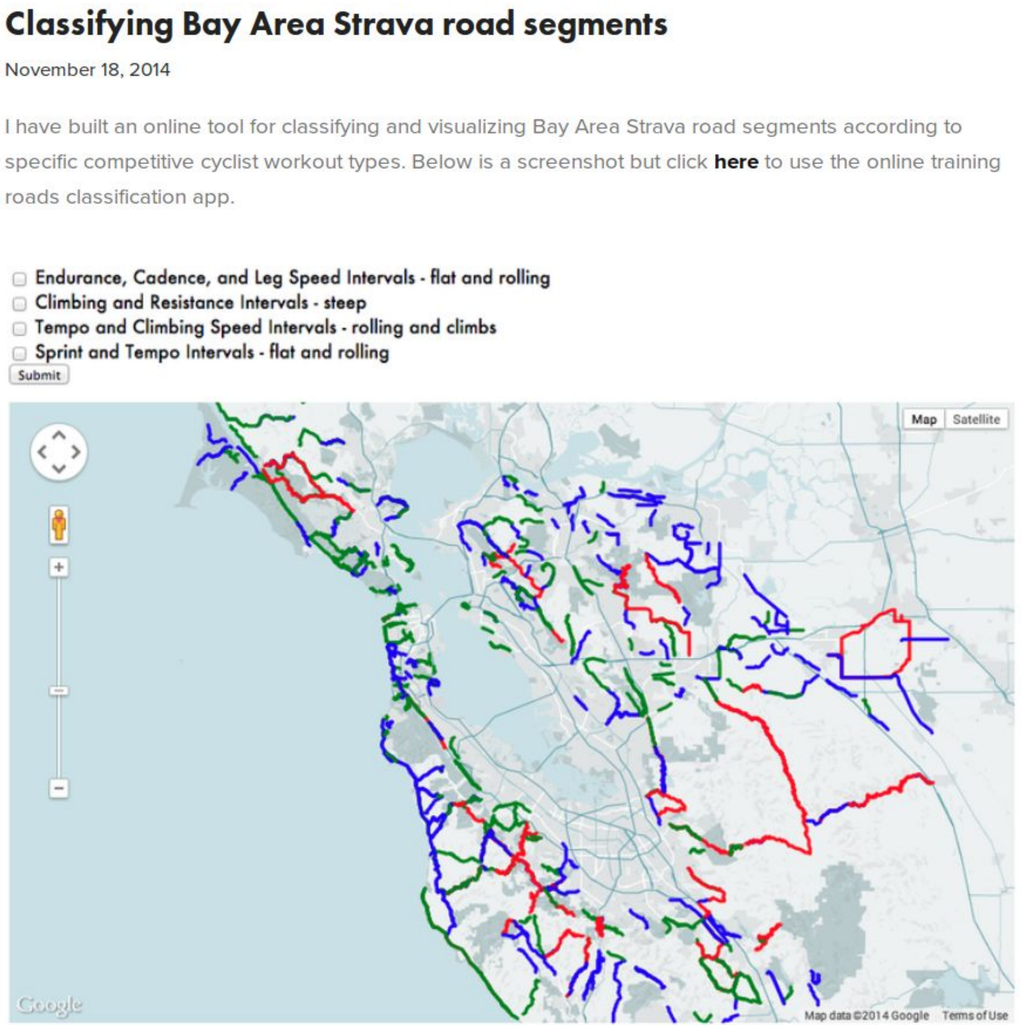
You May Be Happy In:

Baltimore

https://github.com/CLuiz/CityScapes

Examples of clustering projects (2)

http://www.davidbangor.com/blog
https://github.com/rognab/Strava-Segment-Classifier



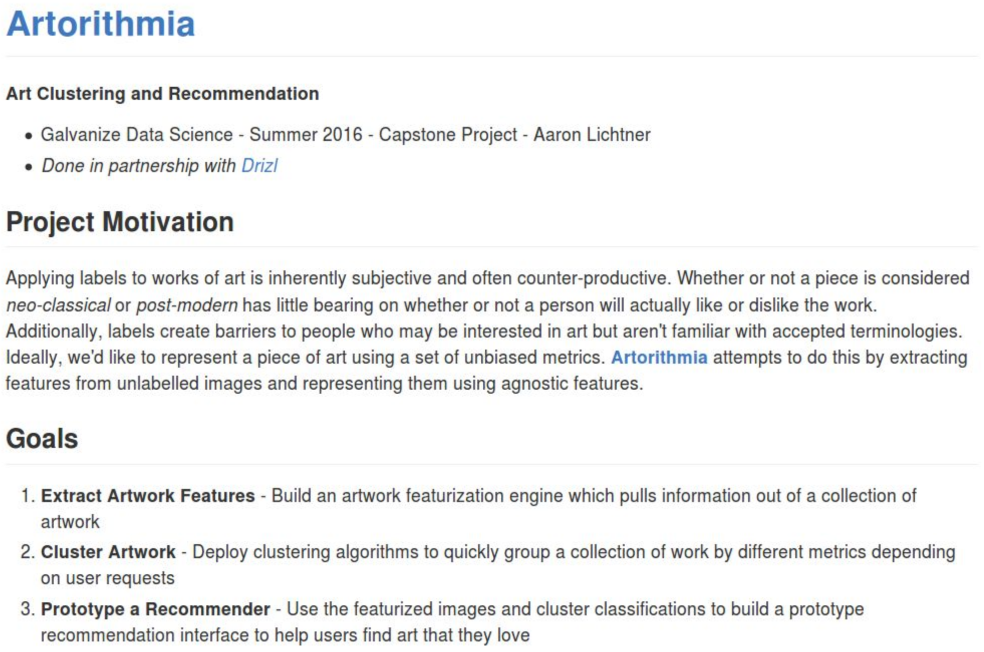
Classifying Bay Area Strava road segments

November 18, 2014

I have built an online tool for classifying and visualizing Bay Area Strava road segments according to specific competitive cyclist workout types. Below is a screenshot but click **here** to use the online training roads classification app.

- ☐ Endurance, Cadence, and Leg Speed Intervals - flat and rolling
- ☐ Climbing and Resistance Intervals - steep
- ☐ Tempo and Climbing Speed Intervals - rolling and climbs
- ☐ Sprint and Tempo Intervals - flat and rolling

Examples of clustering projects (3)



Artorithmia

Art Clustering and Recommendation

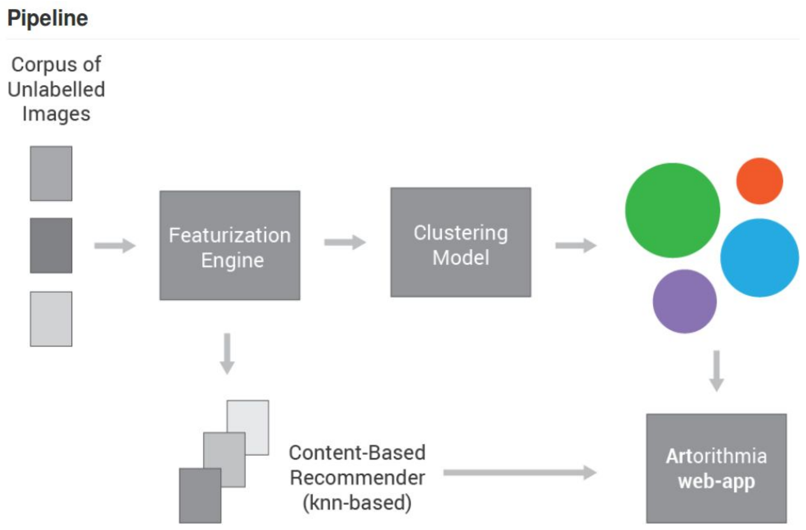
- Galvanize Data Science - Summer 2016 - Capstone Project - Aaron Lichtner
- *Done in partnership with Drizl*

Project Motivation

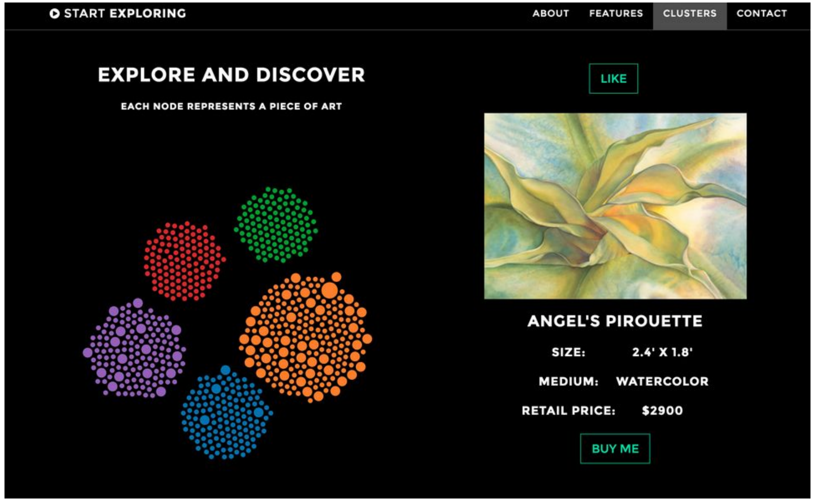
Applying labels to works of art is inherently subjective and often counter-productive. Whether or not a piece is considered *neo-classical* or *post-modern* has little bearing on whether or not a person will actually like or dislike the work. Additionally, labels create barriers to people who may be interested in art but aren't familiar with accepted terminologies. Ideally, we'd like to represent a piece of art using a set of unbiased metrics. **Artorithmia** attempts to do this by extracting features from unlabelled images and representing them using agnostic features.

Goals

1. **Extract Artwork Features** - Build an artwork featurization engine which pulls information out of a collection of artwork
2. **Cluster Artwork** - Deploy clustering algorithms to quickly group a collection of work by different metrics depending on user requests
3. **Prototype a Recommender** - Use the featurized images and cluster classifications to build a prototype recommendation interface to help users find art that they love



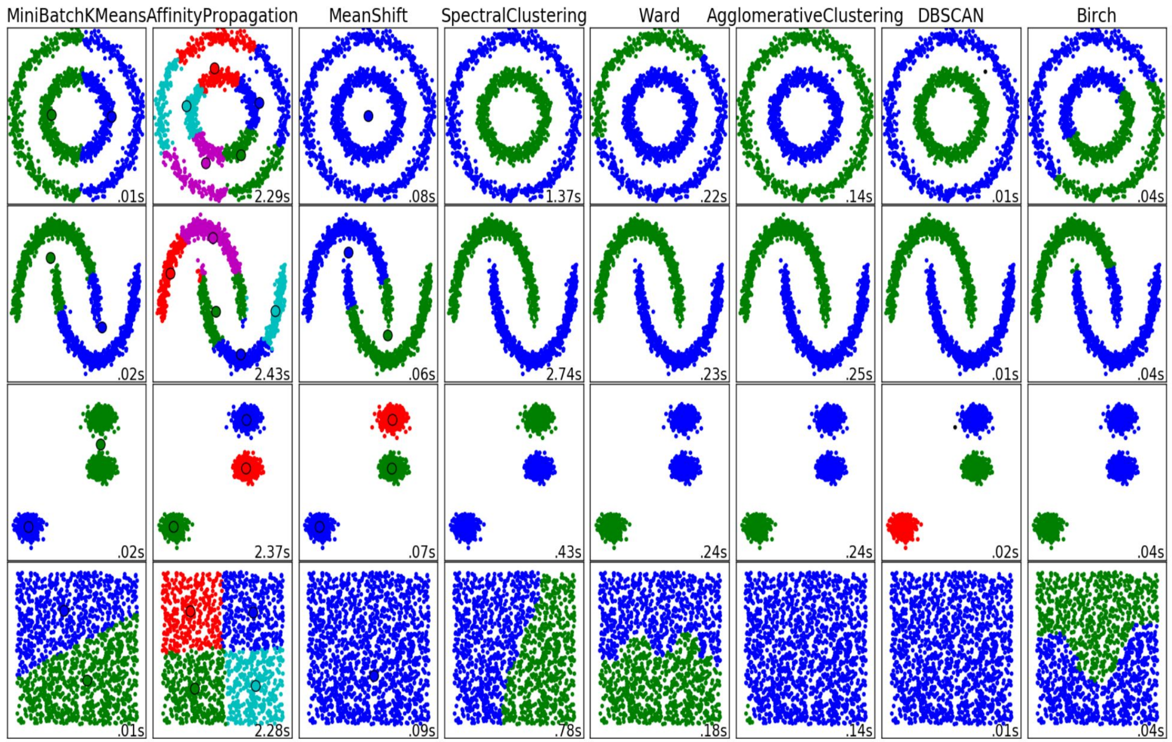
Pipeline
Corpus of Unlabelled Images
Featurization Engine
Clustering Model
Content-Based Recommender (knn-based)
Artorithmia web-app



https://github.com/alichtner/artorithmia

Clustering - some clustering algorithms in sklearn

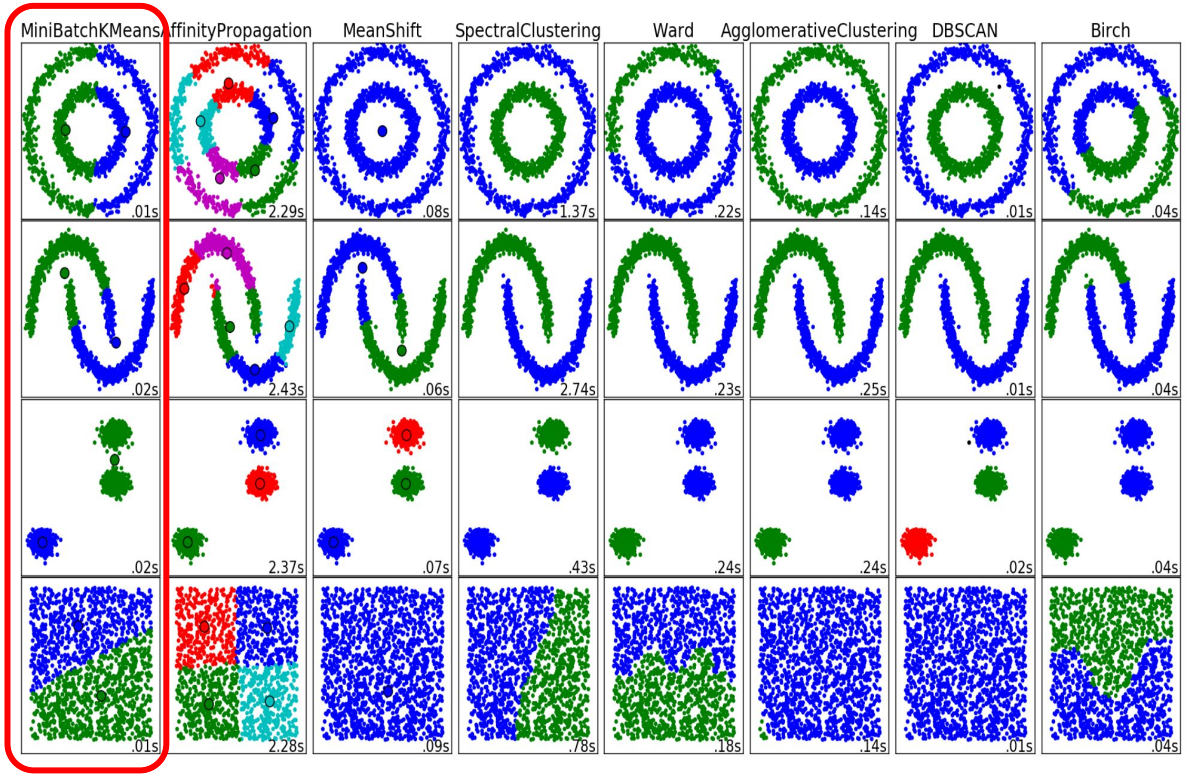
2.3.1. Overview of clustering methods



http://scikit-learn.org/stable/modules/clustering.html

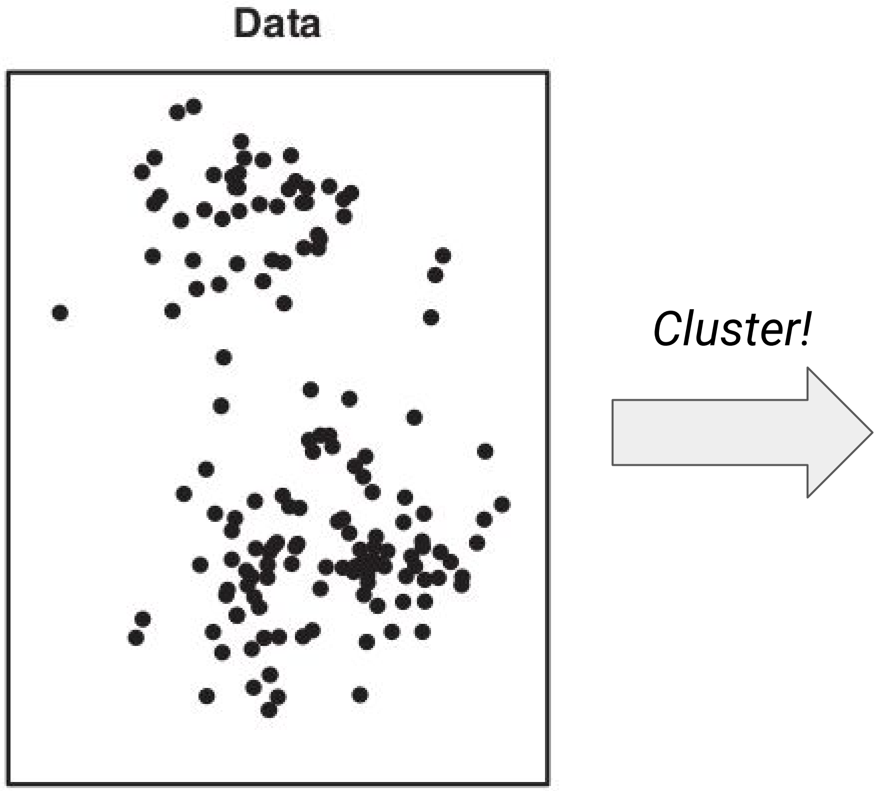
Today: k-means

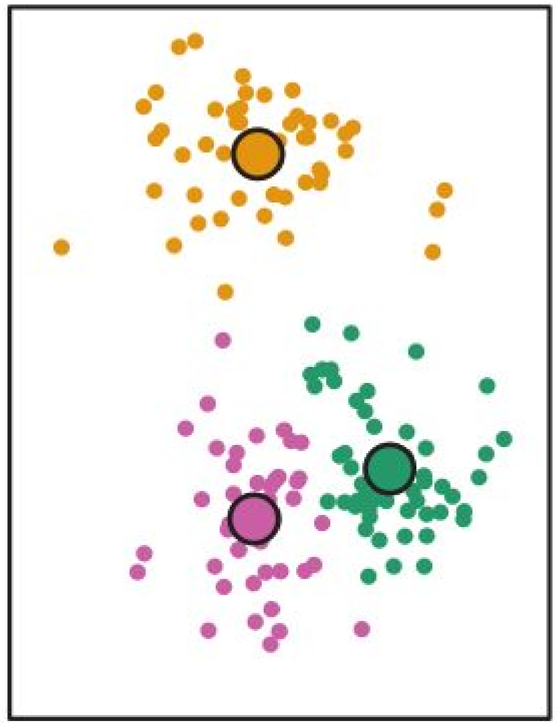
2.3.1. Overview of clustering methods



http://scikit-learn.org/stable/modules/clustering.html

Clustering





k = 3, centroids indicated by large circles

k-means clustering

Would like to partition data into k groups (clusters) so that the total “within cluster variation” (WCV) is the smallest:

$$\underset{C_1,\ldots,C_K}{\text{minimize}} \left\{ \sum_{k=1}^{K} \underline{\text{WCV}(C_k)} \right\}$$

Where WCV for the k-th cluster is the sum of all the pairwise Euclidean distances:

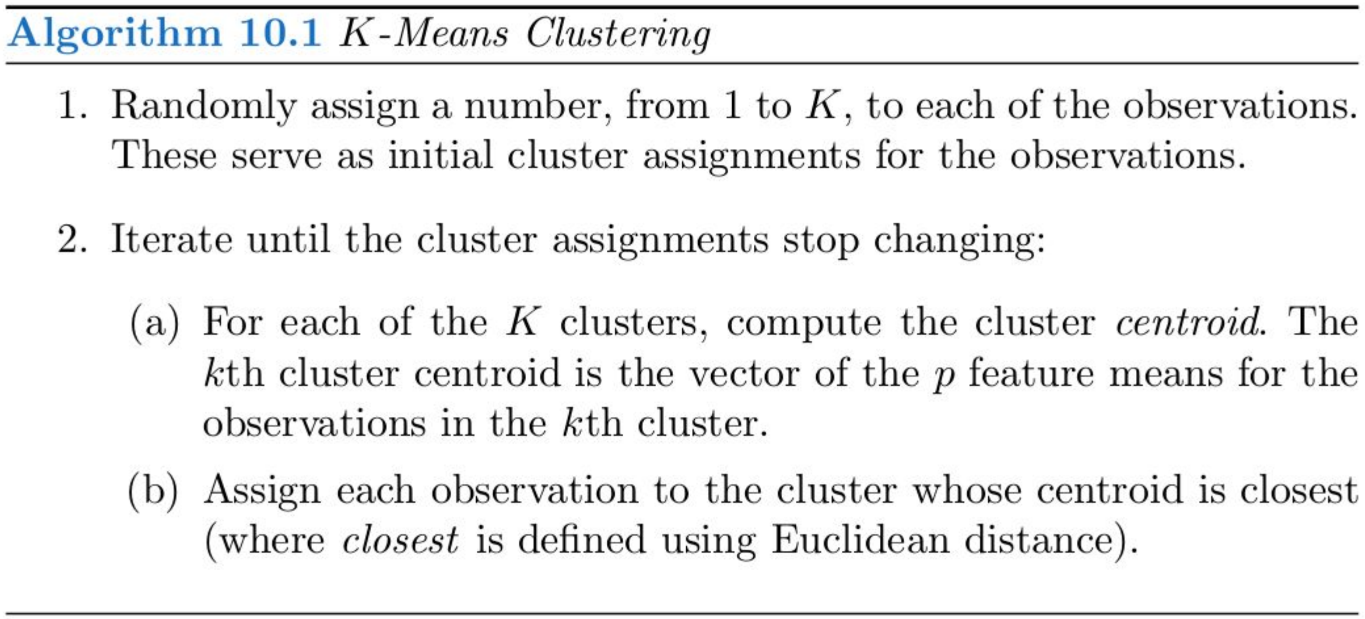
$$\text{WCV}(C_k) = \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2$$

$|C_k|$ is number of observations in k-th cluster

Centroid of cluster C_k

This is the goal (and factors into the cost function), but how to get there?

k-means clustering algorithm



Algorithm 10.1 *K-Means Clustering*

1. Randomly assign a number, from 1 to K, to each of the observations. These serve as initial cluster assignments for the observations.

2. Iterate until the cluster assignments stop changing:

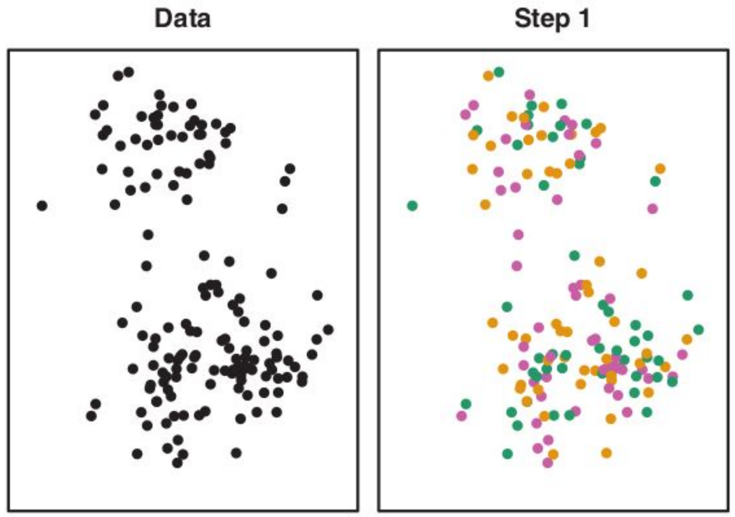
   (a) For each of the K clusters, compute the cluster *centroid*. The kth cluster centroid is the vector of the p feature means for the observations in the kth cluster.

   (b) Assign each observation to the cluster whose centroid is closest (where *closest* is defined using Euclidean distance).

James, G. et al., Introduction to Statistical Learning, 2015

k-means clustering algorithm (visually)

1. Randomly assign each data point to a cluster.



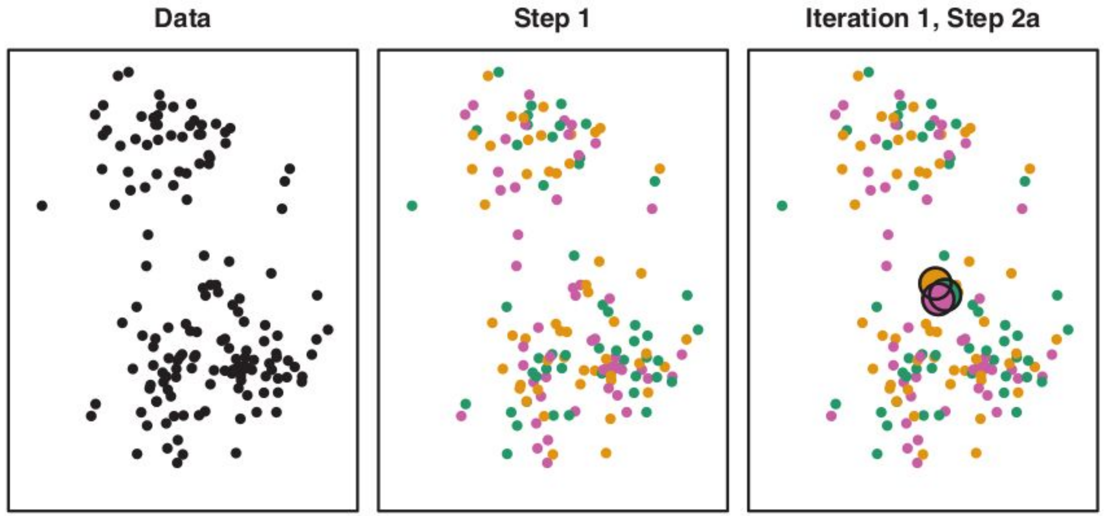
James, G. et al., Introduction to Statistical Learning, 2015

k-means clustering algorithm (visually)

1. Randomly assign each data point to a cluster.

-while cluster assignments change-

2a) Compute the centroid of each cluster.



James, G. et al., Introduction to Statistical Learning, 2015

k-means clustering algorithm (visually)

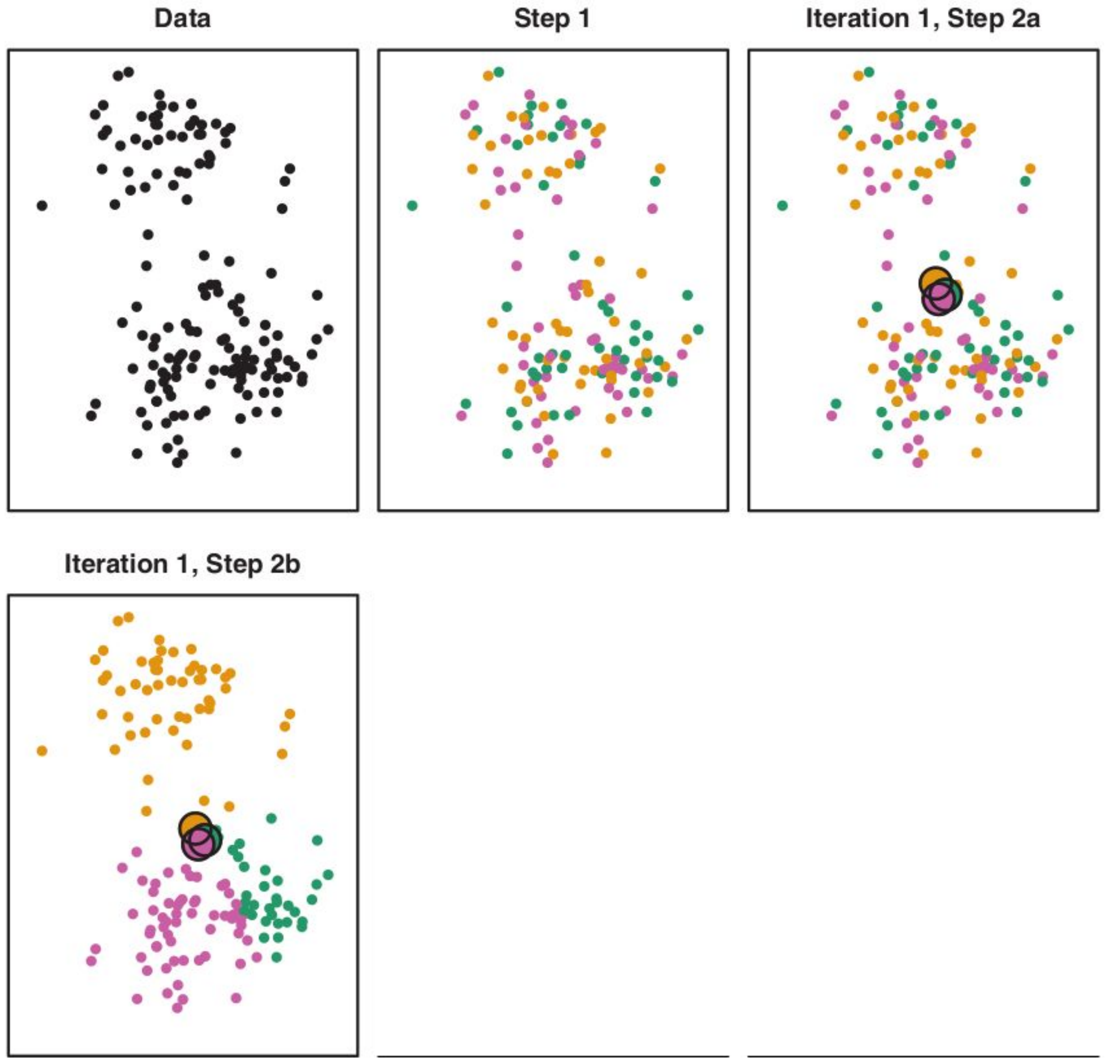
1. Randomly assign each data point to a cluster.

-while cluster assignments change-

2a) Compute the centroid of each cluster.

2b) Reassign each point to belong to the nearest cluster.

James, G. et al., Introduction to Statistical Learning, 2015



k-means clustering algorithm (visually)

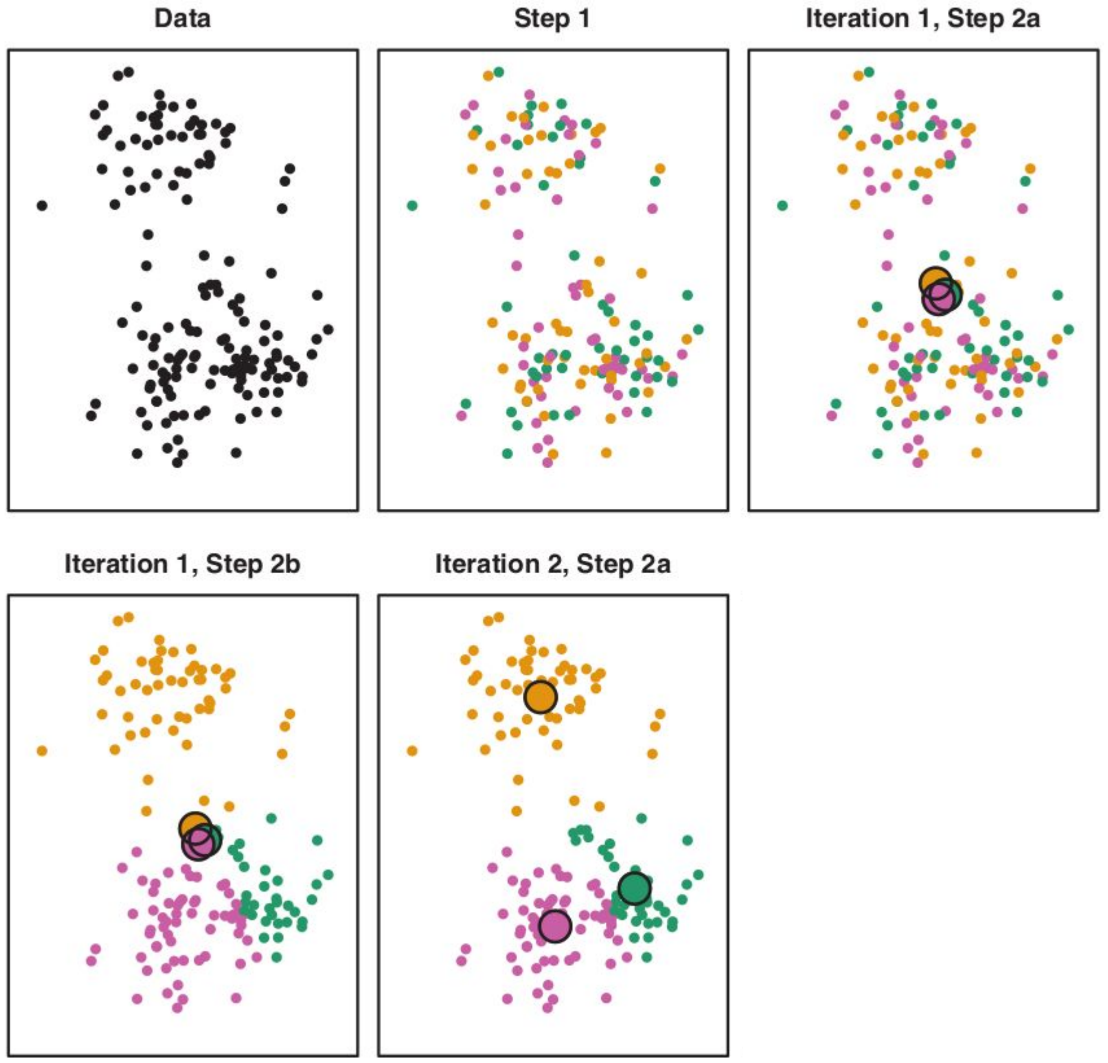
1. Randomly assign each data point to a cluster.

-while cluster assignments change-

2a) Compute the centroid of each cluster.

2b) Reassign each point to belong to the nearest cluster.

James, G. et al., Introduction to Statistical Learning, 2015



k-means clustering algorithm (visually)

1. Randomly assign each data point to a cluster.

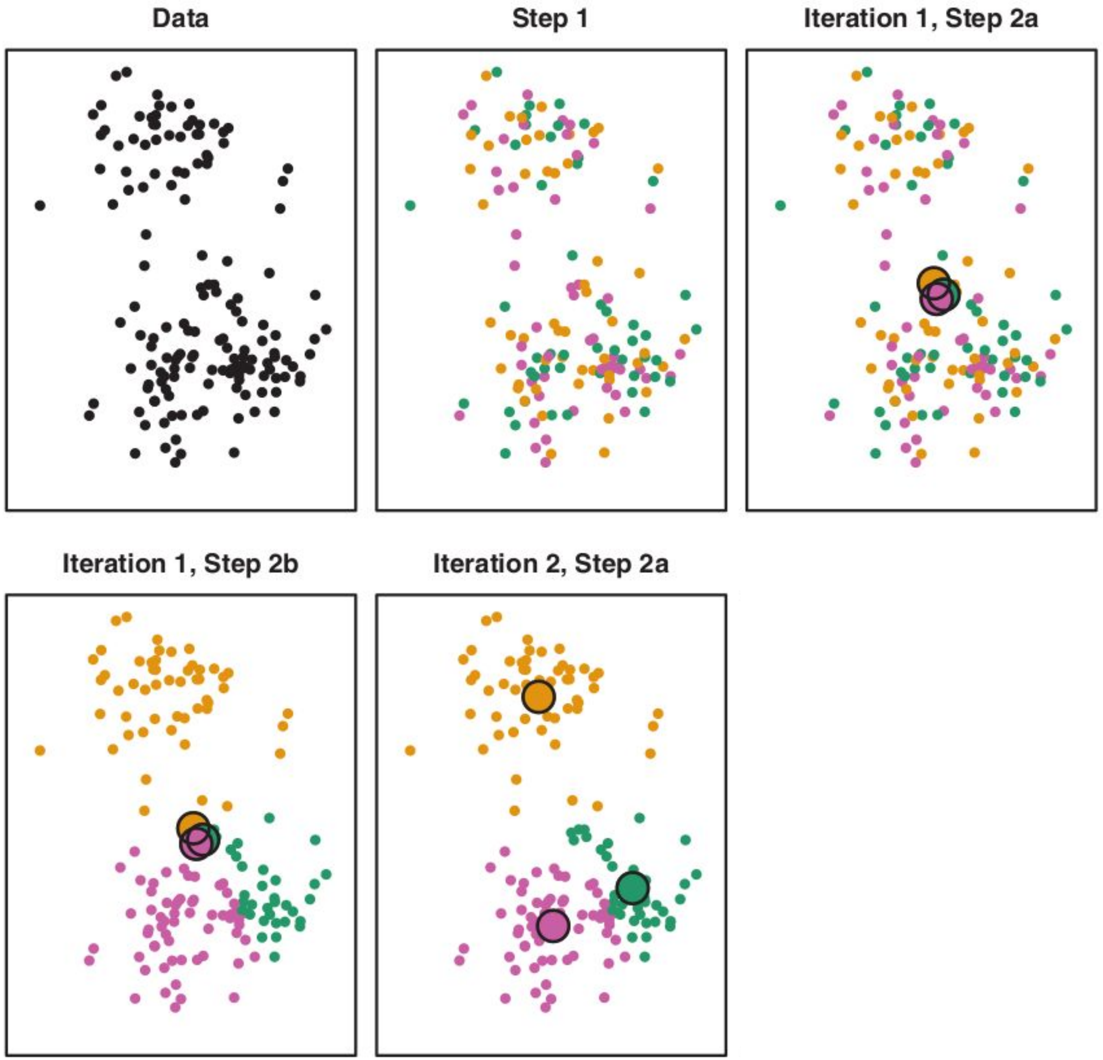
-while cluster assignments change-

2a) Compute the ce of each cluster.

2b) Reassi ch point to belong to the ne cluster.

After many steps

James, G. et al., Introduction to Statistical Learning, 2015



k-means clustering algorithm (visually)

1. Randomly assign each data point to a cluster.

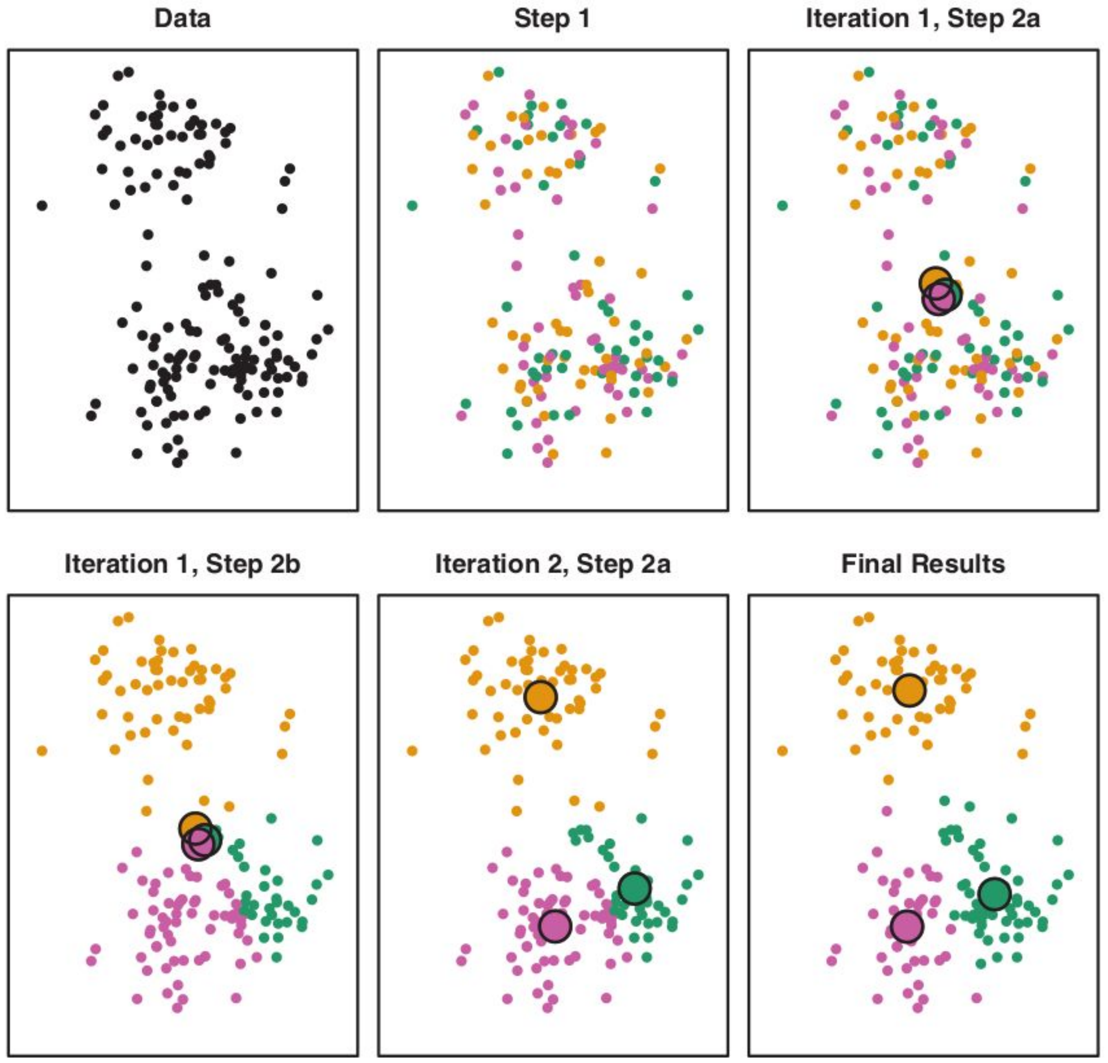
-while cluster assignments change-

2a) Compute the ce of each cluster.

2b) Reassi h point to belong to the ne cluster.

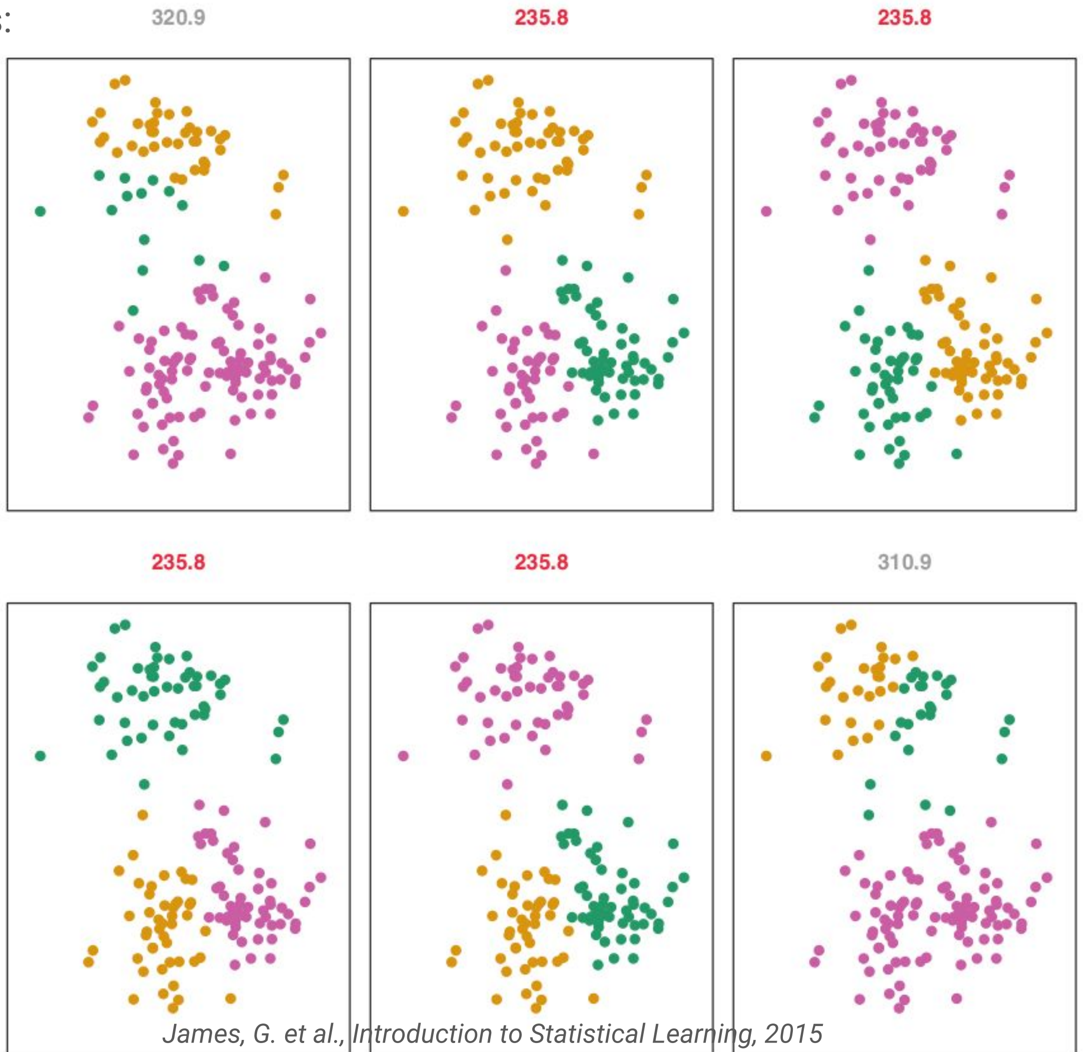
After many steps

James, G. et al., Introduction to Statistical Learning, 2015



k-means convergence

6 solutions:



Value above each chart is:

$$\underset{C_1,\ldots,C_K}{\text{minimize}}\left\{\sum_{k=1}^{K}\frac{1}{|C_k|}\sum_{i,i'\in C_k}\sum_{j=1}^{p}(x_{ij}-x_{i'j})^2\right\}$$

Values vary because initializations are random, and so are only guaranteed to find *local* minima.

Need to run several times (or k-means++ - show algorithm)

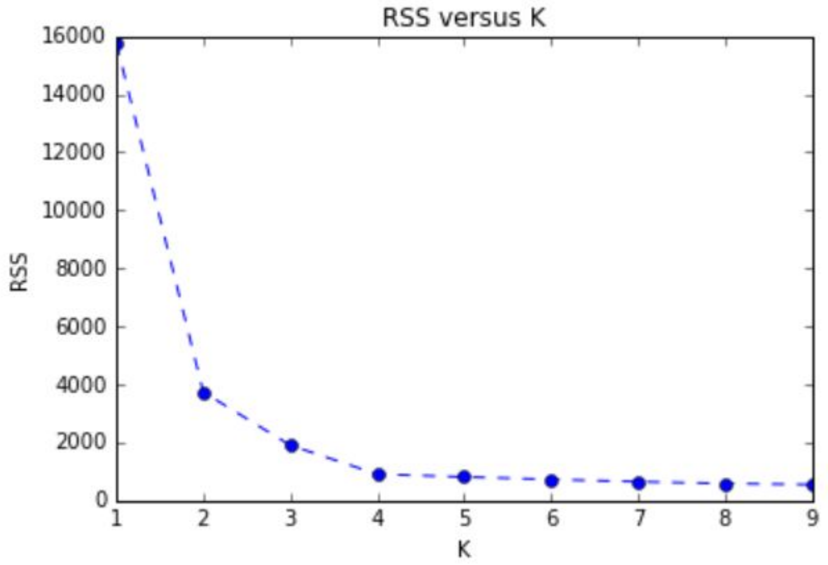
James, G. et al., Introduction to Statistical Learning, 2015

k-means solution - the right value of k?

- No easy answer, and admittedly a little fuzzy
- Looking for something useful/interpretable, and need enough groups to get that information

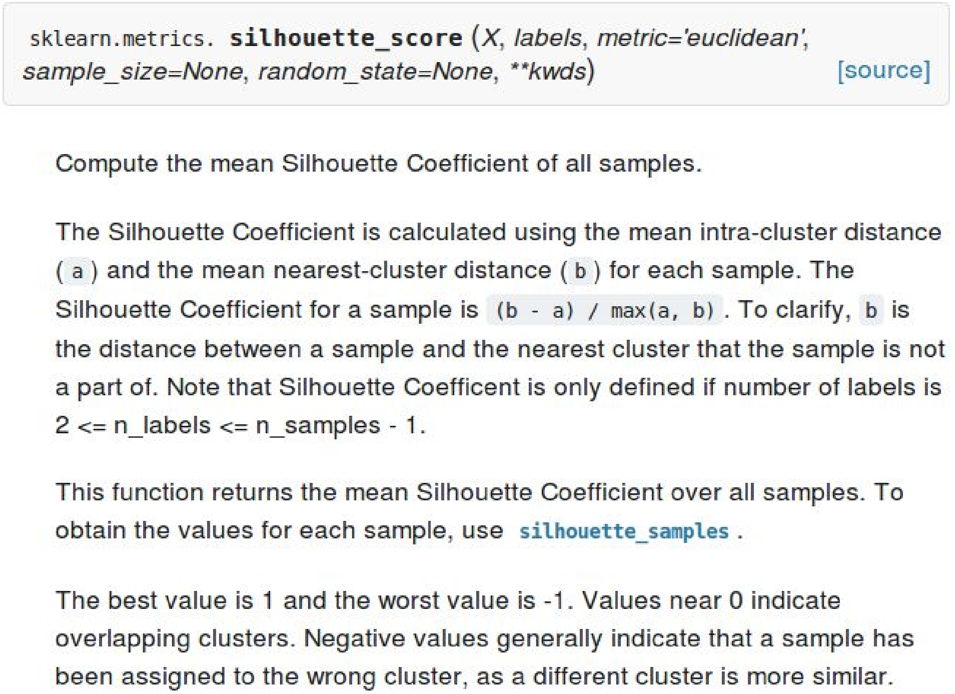
One approach (Elbow method), pick k that drastically reduces RSS:

$$\text{RSS} = \sum_{k=1}^{K} \sum_{C(i)=k} ||x_i - \bar{x}_k||^2$$



k-means solution - the right value of k?

Another approach, the silhouette score:



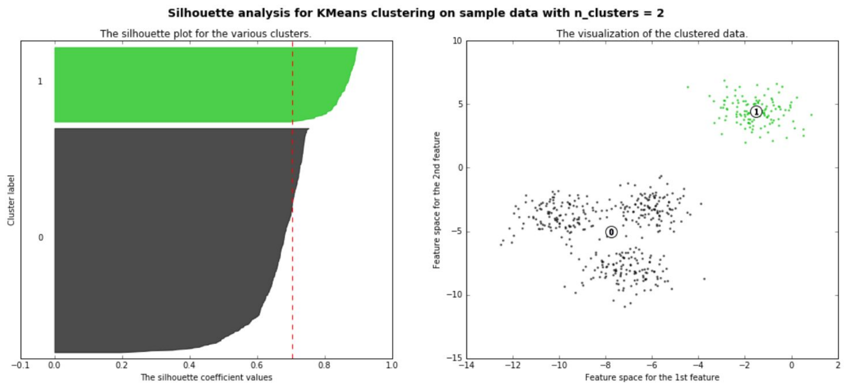
sklearn.metrics. **silhouette_score** (*X*, *labels*, *metric='euclidean'*, *sample_size=None*, *random_state=None*, ***kwds*) [source]

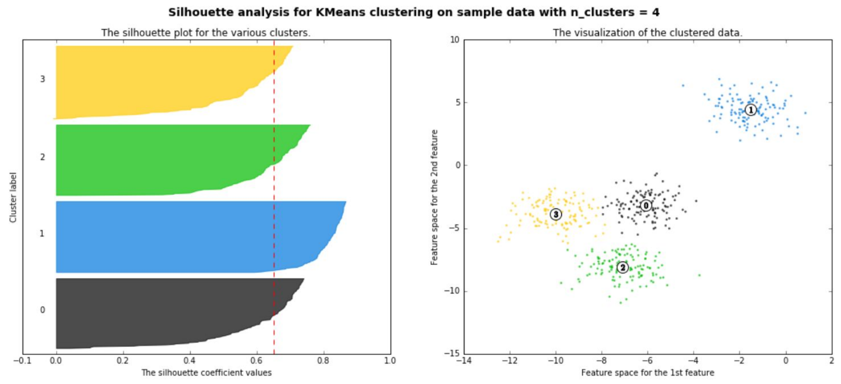
Compute the mean Silhouette Coefficient of all samples.

The Silhouette Coefficient is calculated using the mean intra-cluster distance (`a`) and the mean nearest-cluster distance (`b`) for each sample. The Silhouette Coefficient for a sample is `(b - a) / max(a, b)`. To clarify, `b` is the distance between a sample and the nearest cluster that the sample is not a part of. Note that Silhouette Coefficent is only defined if number of labels is 2 <= n_labels <= n_samples - 1.

This function returns the mean Silhouette Coefficient over all samples. To obtain the values for each sample, use `silhouette_samples` .

The best value is 1 and the worst value is -1. Values near 0 indicate overlapping clusters. Negative values generally indicate that a sample has been assigned to the wrong cluster, as a different cluster is more similar.





k-means - practical considerations

- Standardize features? Based on a distance metric, so *yes*.
- Susceptible to outliers? Euclidean distance, so *yes*.
- Susceptible to Curse of Dimensionality? Euclidean distance, so *yes*.
- How to handle categoricals? Look into k-medoids, kmodes (https://pypi.python.org/pypi/kmodes/).
- Fast! (MiniBatchKmeans for large datasets). But finds local minima. (try kmeans++).
- How many clusters, k?
- Other clustering algorithms available, e.g. DBScan

Go through k-means-FB jupyter notebook