CSCAR CONSULTING FOR STATISTICS, COMPUTING & ANALYTICS RESEARCH
UNIVERSITY OF MICHIGAN

Unsupervised Machine Learning in Python using Scikit-learn

Data Science Skills Series

Márcio Mourão
https://marcio-mourao.github.io

Outline

- Brief introduction to Machine Learning
  - Unsupervised and supervised learning
- Principal Component Analysis (PCA)
  - Review the methodology
  - How to use Python & Scikit-learn to apply the methodology
- K-Means
  - Review the methodology
  - How to use Python & Scikit-learn to apply the methodology

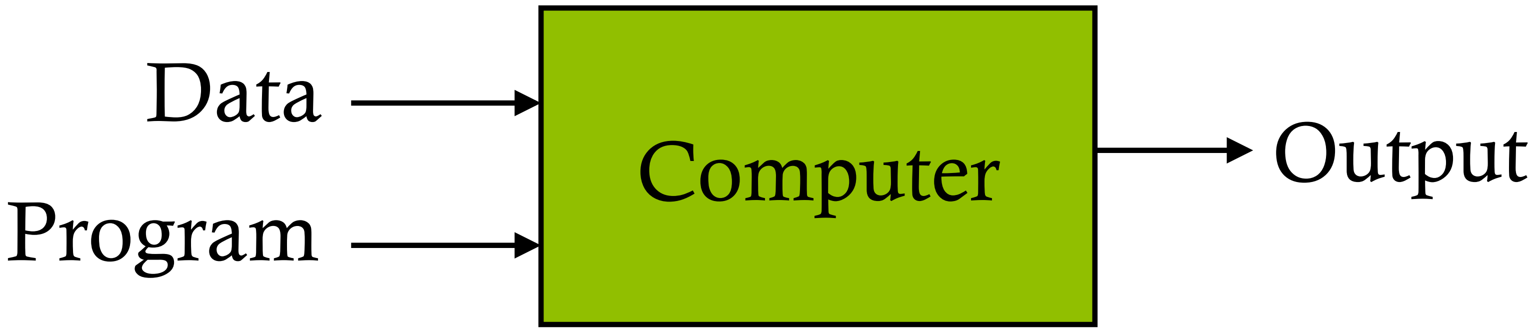
What is Machine Learning (ML)?

- A branch of **artificial intelligence**, concerned with the design and development of algorithms that allow computers to evolve behaviors based on empirical data.

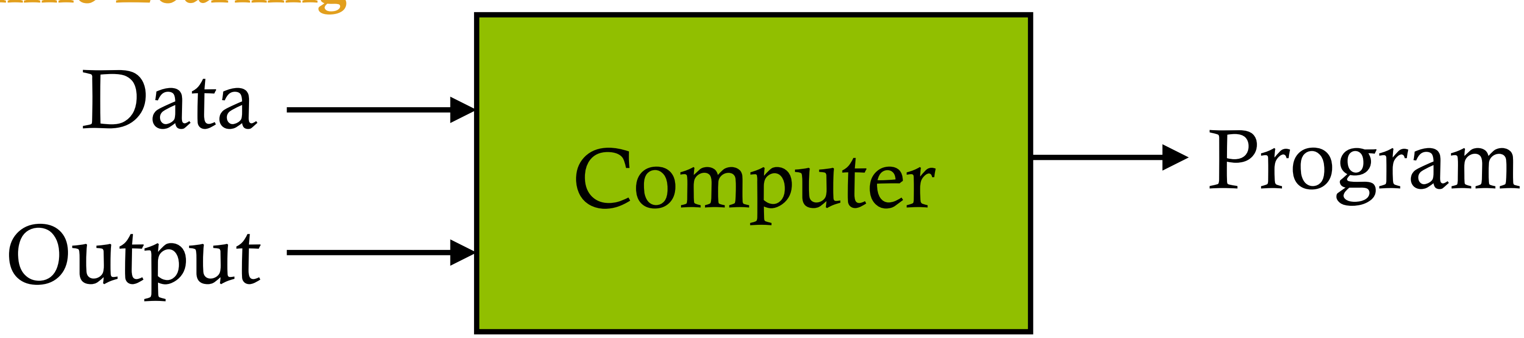
- As intelligence requires knowledge, it is necessary for the computers to acquire knowledge.

What is Machine Learning (ML)?

Traditional Programming



Machine Learning



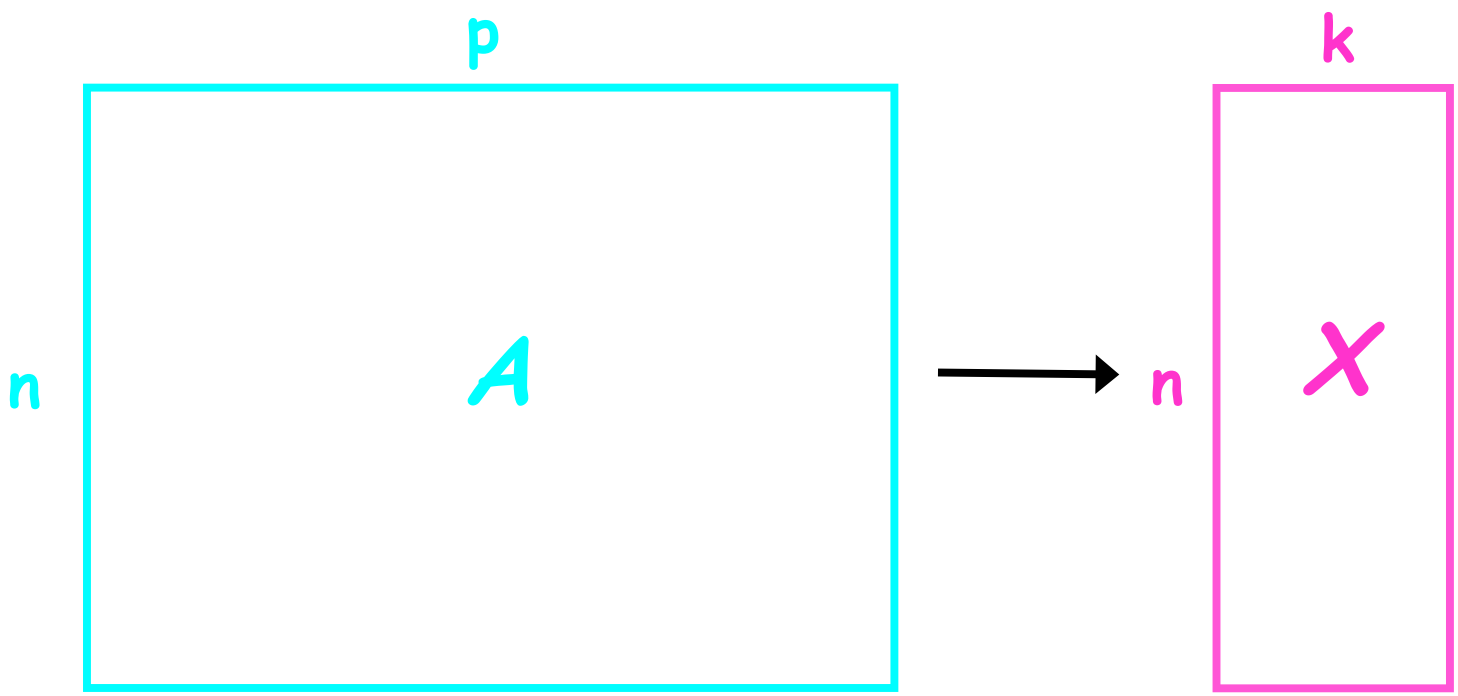
Some algorithms

- **Unsupervised learning (** $\{x_n \in R^d\}_{n=1}^N$ **)**
  - Dimensionality reduction
  - Clustering

- **Supervised learning (** $\{x_n \in R^d, y_n \in R\}_{n=1}^N$ **)**
  - Decision tree induction
  - Rule induction
  - Instance-based learning
  - Bayesian learning
  - Neural networks
  - Support vector machines
  - Model ensembles
  - Learning theory

Principal Component Analysis (PCA)

Summarization of data with many (p) variables by a smaller set of (k) derived (synthetic, composite) variables.

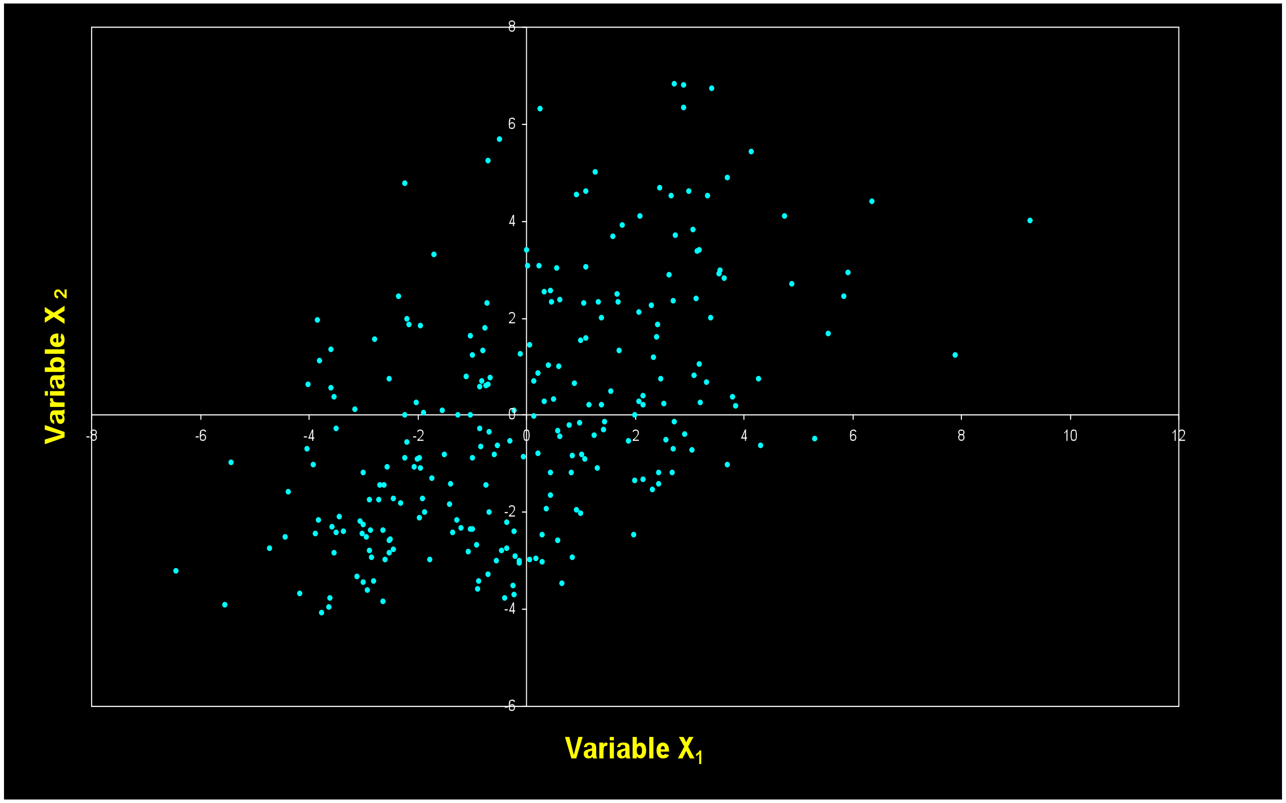


Geometric Rationale of PCA

- The goal of PCA is to rigidly rotate the axes of this p-dimensional space to new positions (principal axes) that have the following properties:

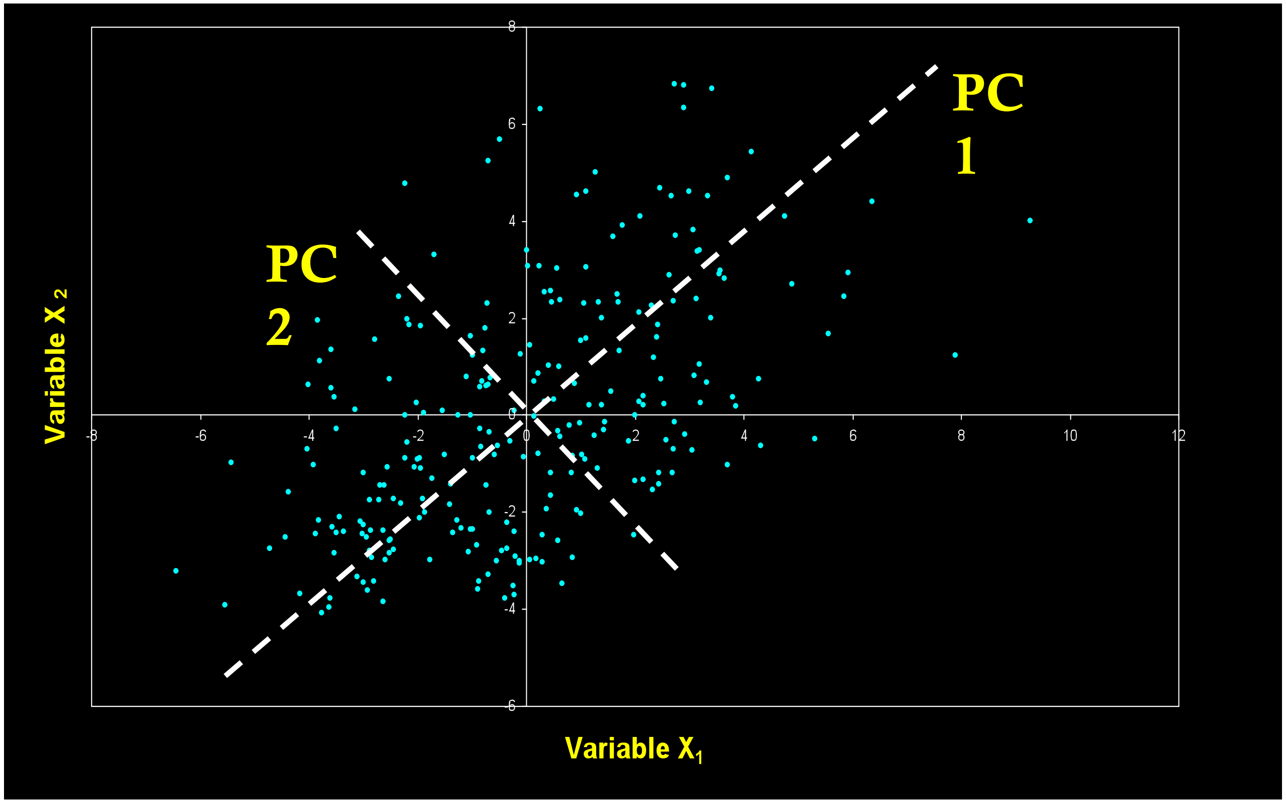
  - The axis are ordered such that principal axis 1 has the highest variance, axis 2 has the next highest variance, ... , and axis p has the lowest variance
  - The covariance among each pair of the principal axes is zero (the principal axes are uncorrelated).

PCA Example (in 2D)



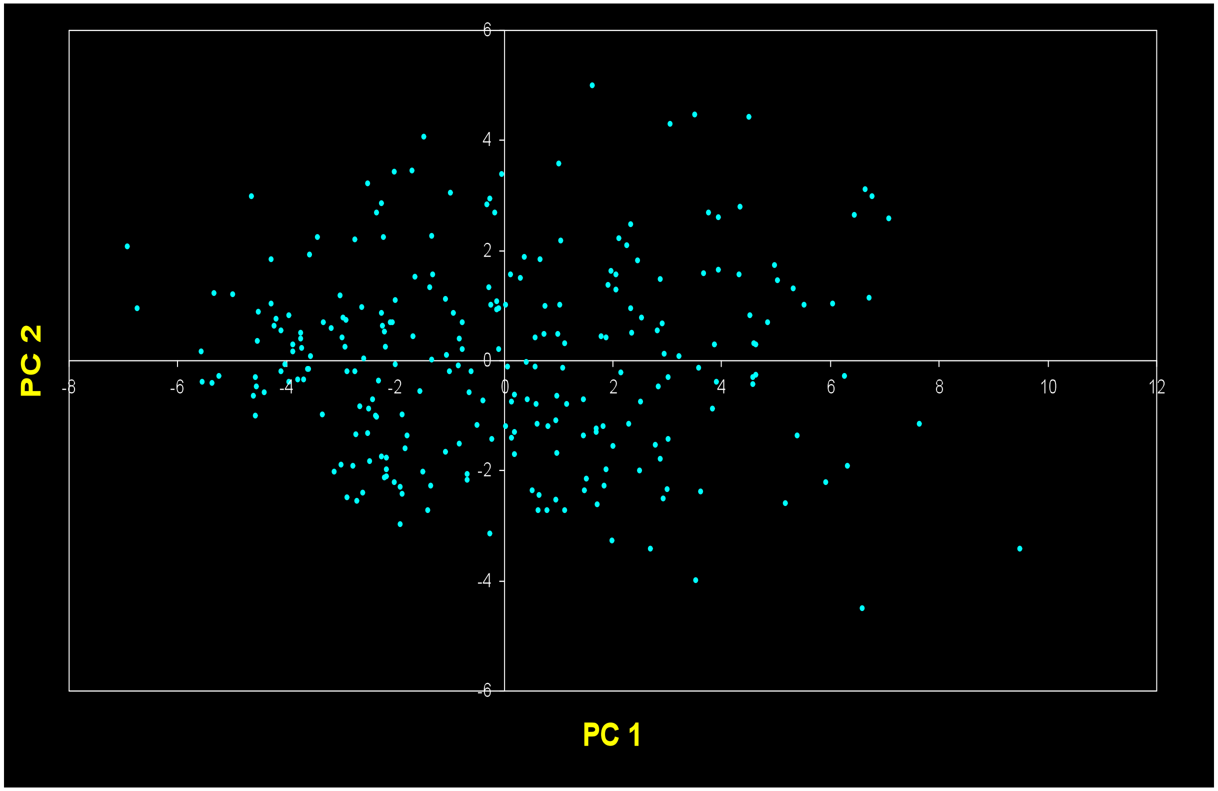
- Variables X_1 and X_2 have positive covariance
- Each variable has a similar variance

PCA Example (in 2D)



- Variables X_1 and X_2 have positive covariance
- Each variable has a similar variance

PCA Example (in 2D)



- PC 1 has the highest possible variance (9.88)
- PC 2 has a variance of (3.03)
- PC 1 and PC 2 have zero covariance.

A summary of the PCA approach

1. Standardize the data

2. Obtain eigenvectors and eigenvalues from the covariance matrix or correlation matrix, or perform Singular Vector Decomposition

3. Sort eigenvalues in descending order and choose the k eigenvectors that correspond to the k largest eigenvalues where k is the number of dimensions of the new feature subspace ($k \leq p$)

4. Construct the projection matrix W from the selected k eigenvectors

5. Transform the original dataset X via W to obtain a k-dimensional feature subspace Y.

Some features

- Assumes relationships among variables are linear
  - cloud of points in p-dimensional space has linear dimensions that can be effectively summarized by the principal axes

- if the structure in the data is nonlinear, the principal axes will not be an efficient and informative summary of the data.

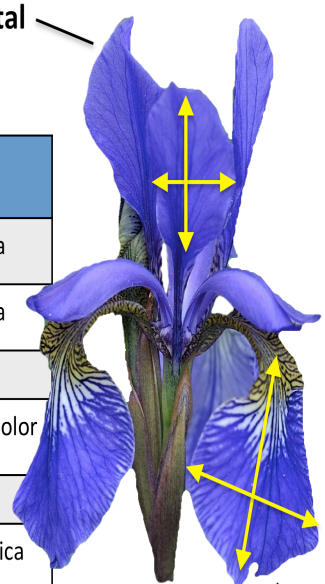
PCA in Python

- Sklearn.decomposition.PCA is the class that implements PCA in Scikit-learn.
- Documentation is available at
  http://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html
- Key options:
  - n_components (default is the min(n_samples, n_features))
  - svd_solver (default 'auto'): The default solver
- Outputs:
  - components_: Principal axes in feature space, representing the directions of maximum variance in the data
  - explained_variance_ratio_: Percentage of variance explained by each of the selected components.

The Iris dataset

Samples
(instances, observations)

	Sepal length	Sepal width	Petal length	Petal width	Class label
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
			...		
50	6.4	3.5	4.5	1.2	Versicolor
			...		
150	5.9	3.0	5.0	1.8	Virginica



Features
(attributes, measurements, dimensions)

Class labels
(targets)

- This data sets consists of 3 different types of irises' (Setosa, Versicolour, and Virginica) based on four attributes
- The data is stored in a 150x4 numpy.ndarray, the rows being the samples and the columns being: Sepal Length, Sepal Width, Petal Length and Petal Width

K-Means

K-Means problem

- Partitioning Clustering Approach
  - Learn a partition on a data set to produce several non-empty clusters
  - Optimal partition achieved via minimizing the sum of squared distance to its "representative object" in each cluster

$$E = \sum_{k=1}^{K} \sum_{\mathbf{x} \in C_k} d^2(\mathbf{x}, \mathbf{m}_k)$$

e.g., Euclidean distance $d^2(\mathbf{x}, \mathbf{m}_k) = \sum_{n=1}^{N} (x_n - m_{kn})^2$

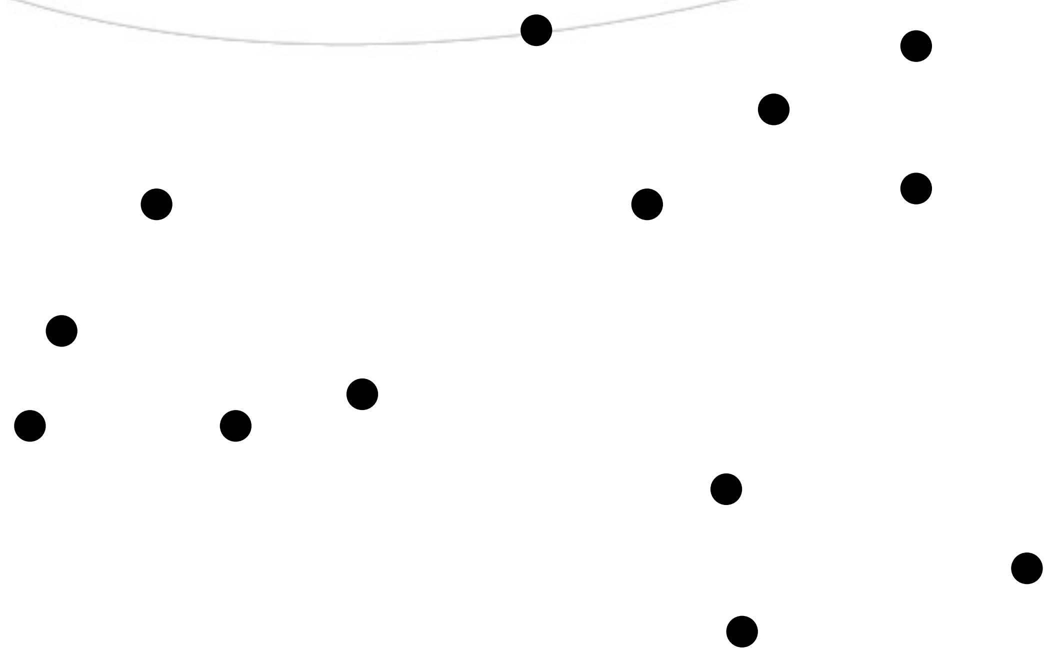
Lloyd's algorithm

Given a set of data points as input

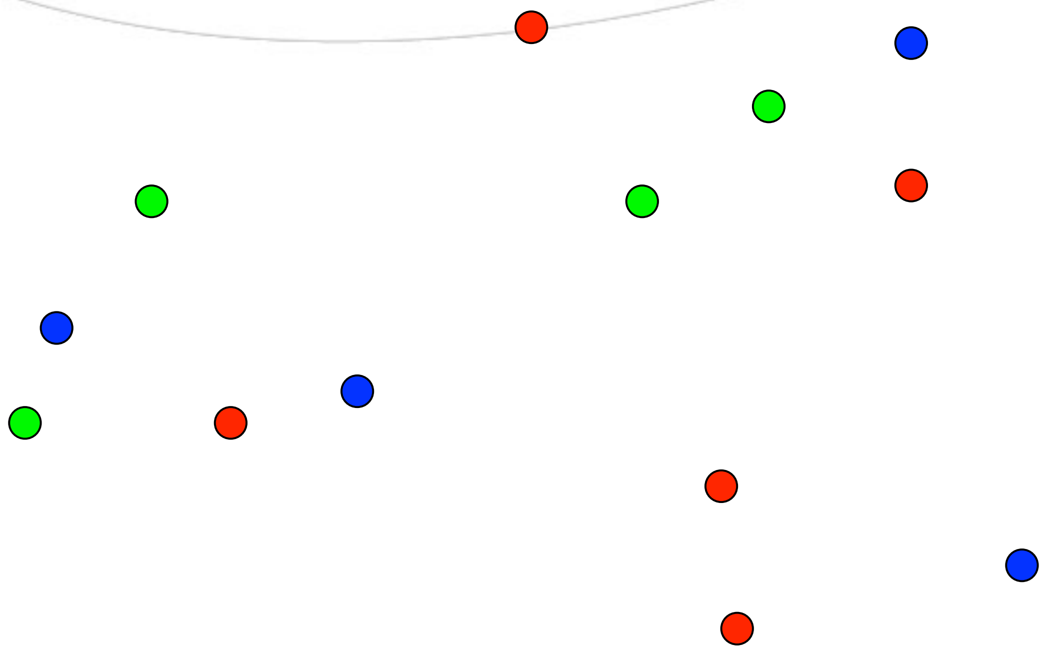
1. Randomly assign each point to one of the *k* clusters

2. Repeat until convergence
    2.1 Calculate <u>model</u> (center) of each of the *k* clusters
    2.2 Assign each point to the cluster with the closest <u>model (center)</u>

K-Means example (in 2D)

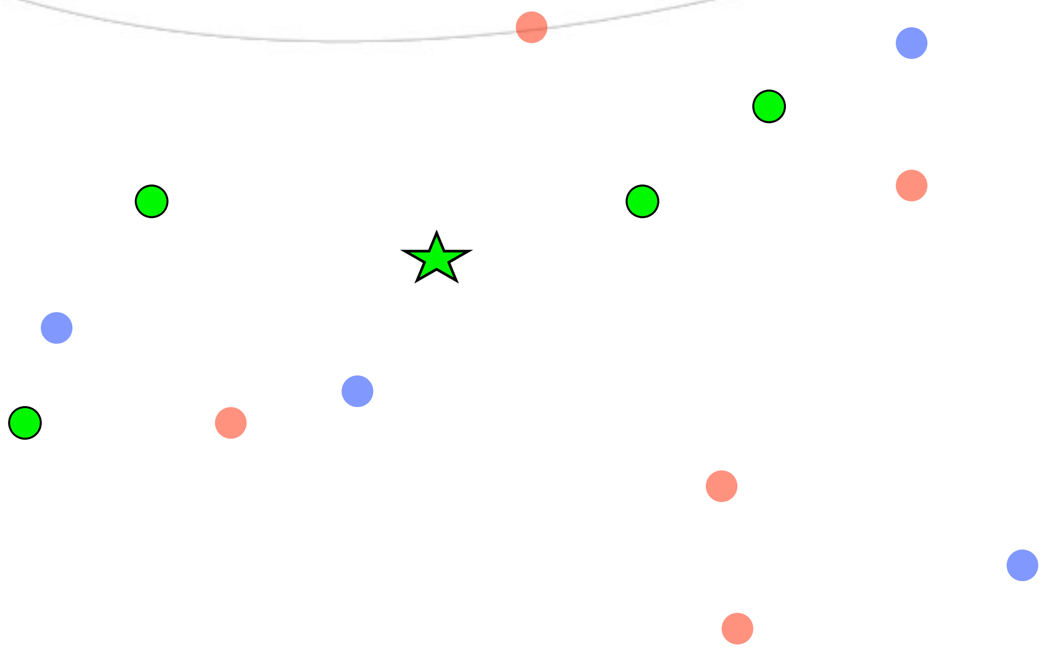


K-Means example (in 2D)



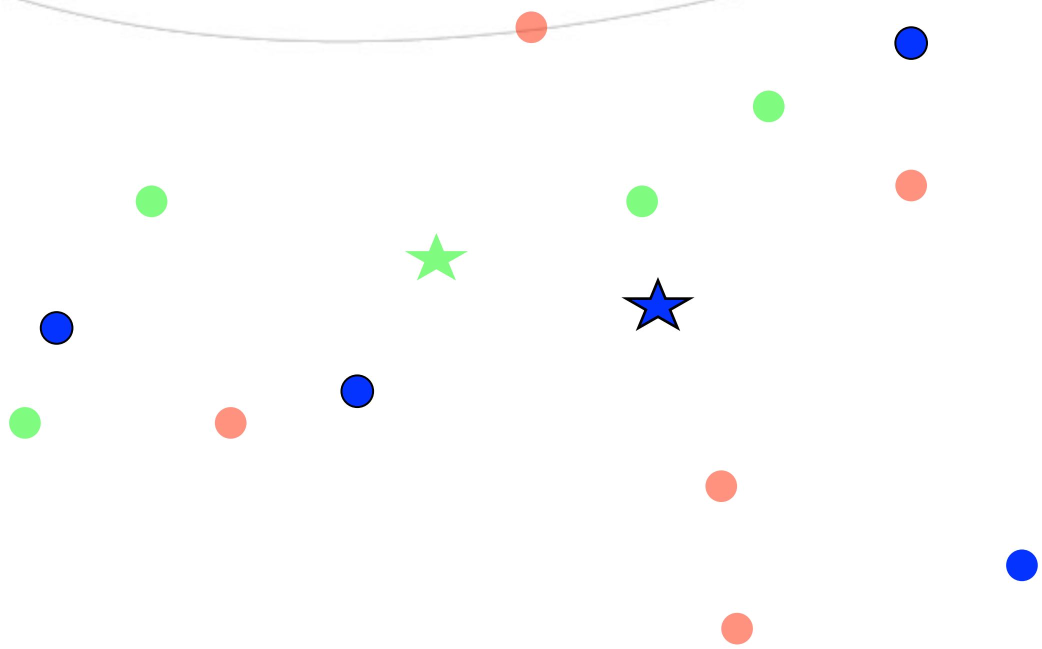
1. Randomly assign each point to one of the k clusters (assume k=3)

K-Means example (in 2D)



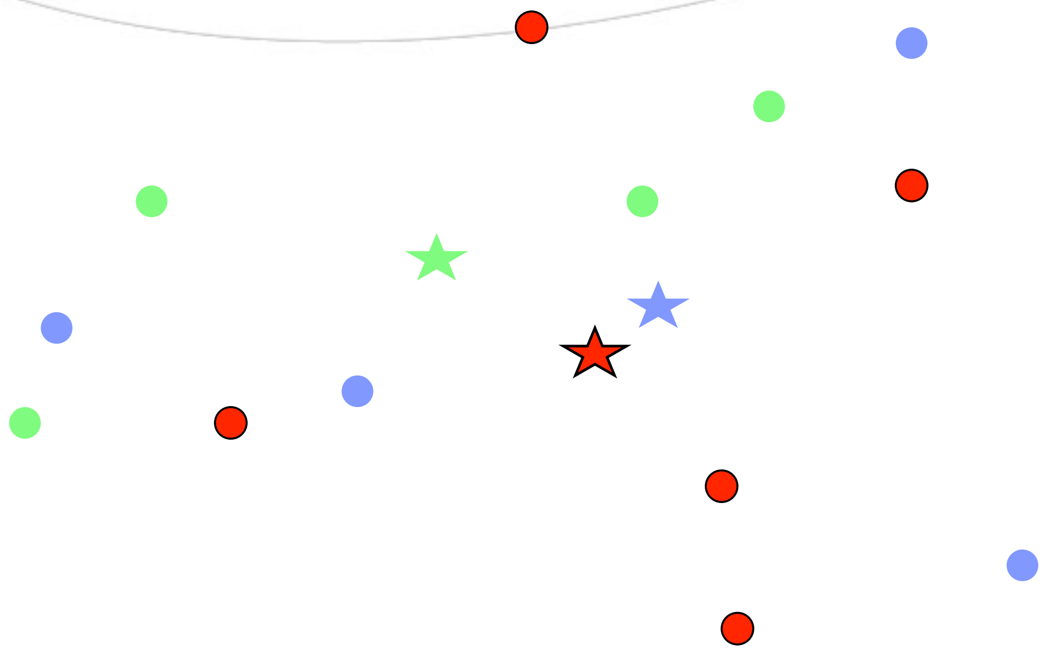
2.1 Calculate center of each cluster

K-Means example (in 2D)



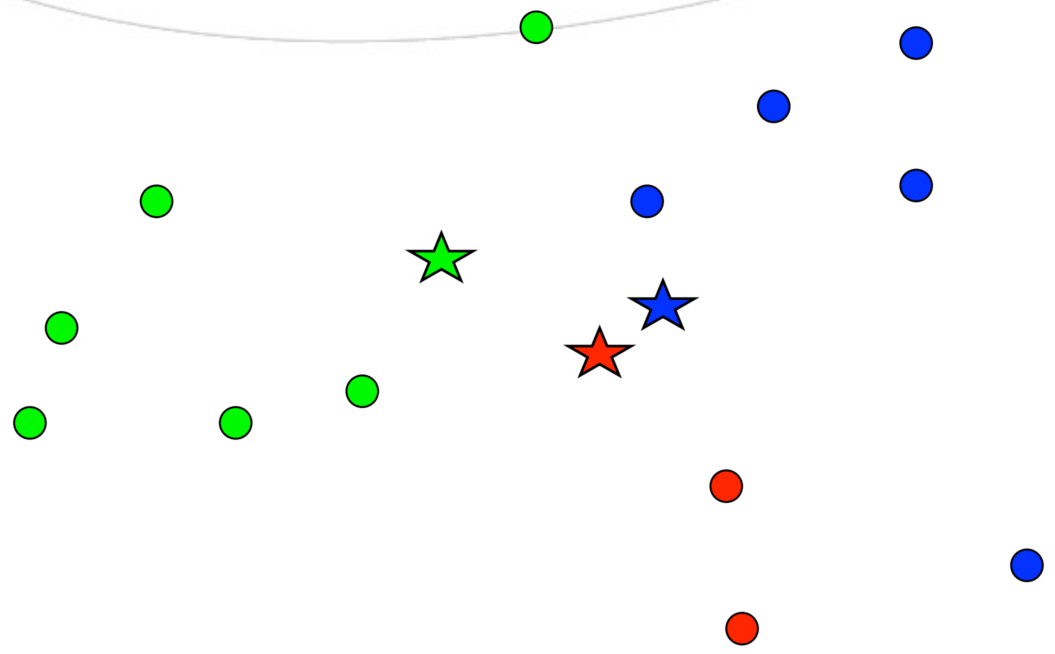
2.1 Calculate center of each cluster

K-Means example (in 2D)



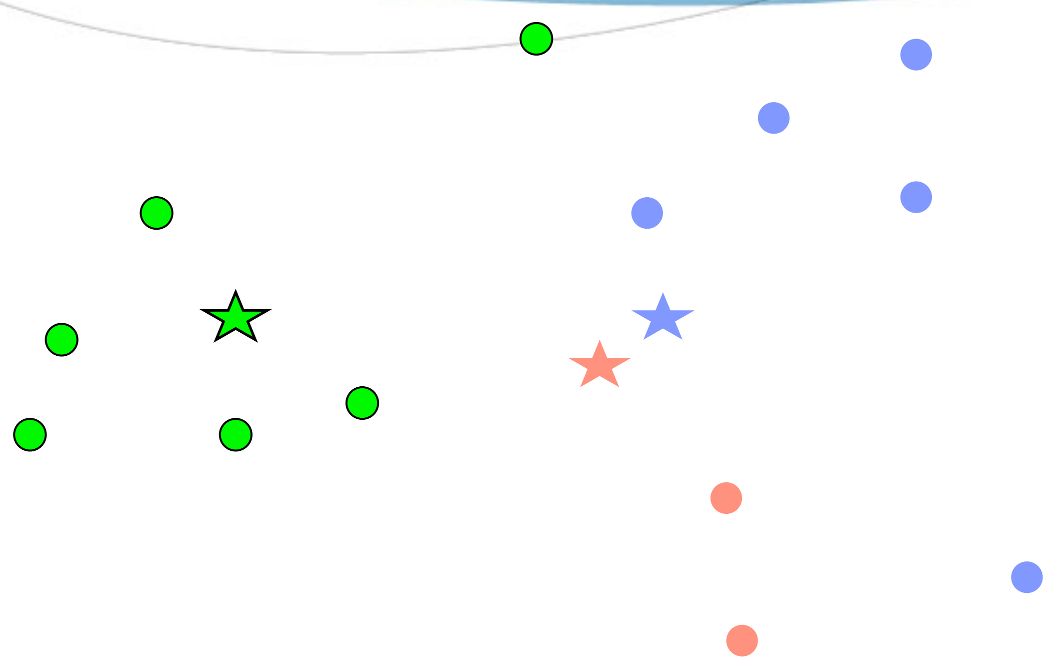
2.1 Calculate center of each cluster

K-Means example (in 2D)



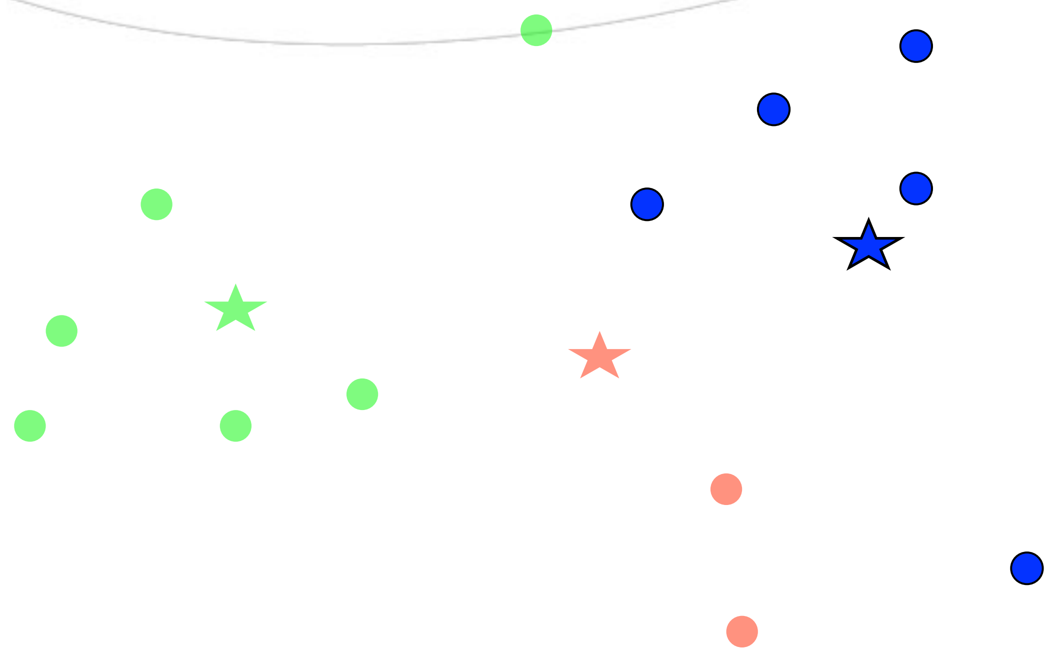
2.2 Assign each point to closest cluster center

K-Means example (in 2D)



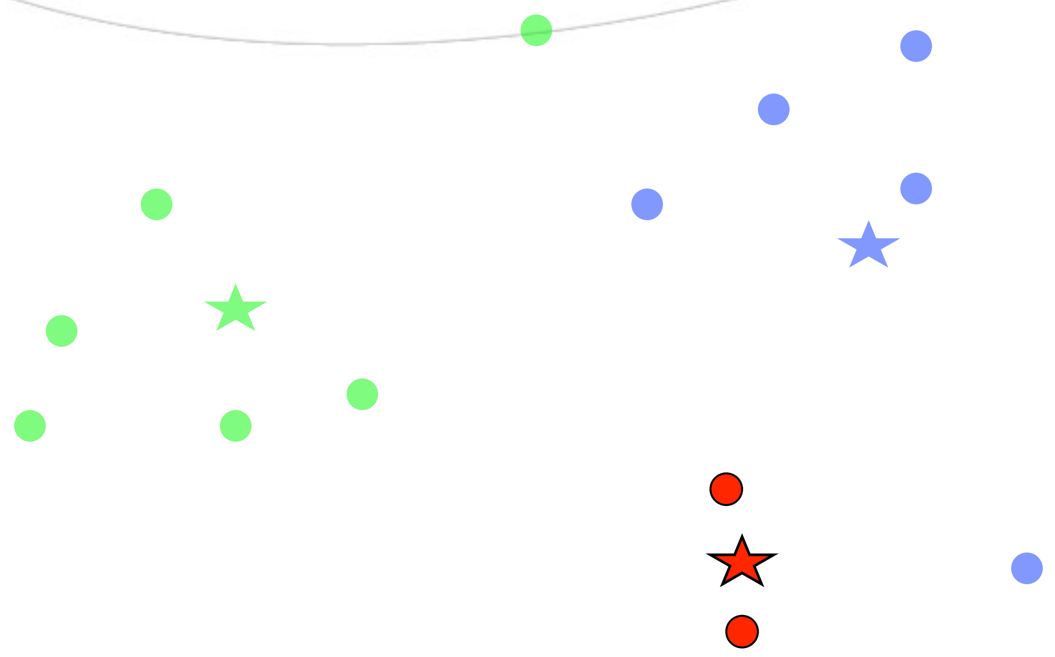
2.1 Calculate center of each cluster

K-Means example (in 2D)



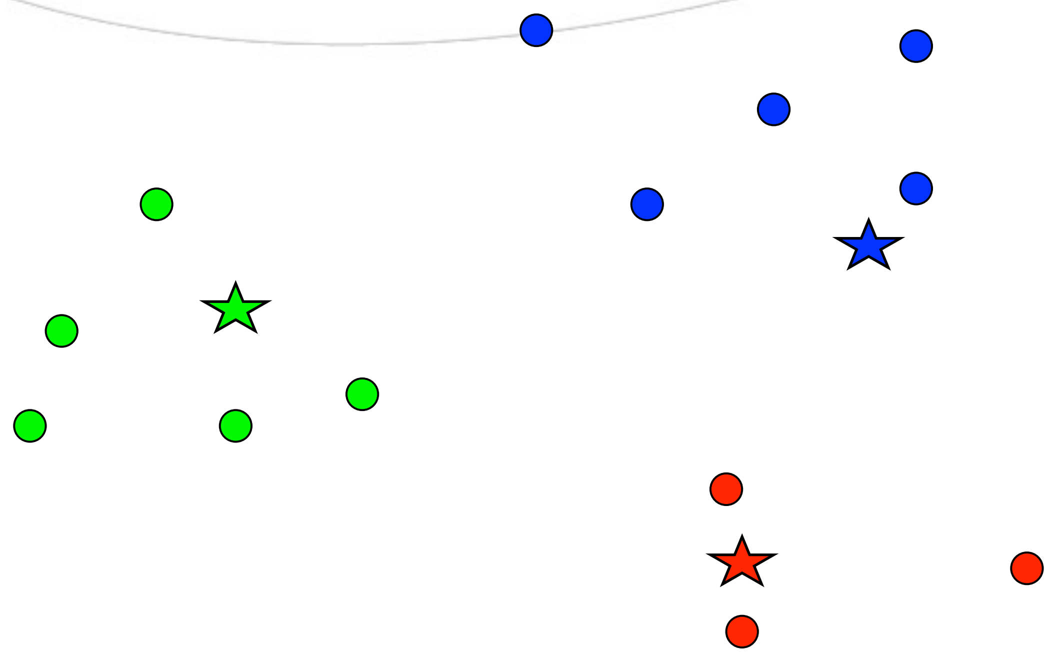
2.1 Calculate center of each cluster

K-Means example (in 2D)



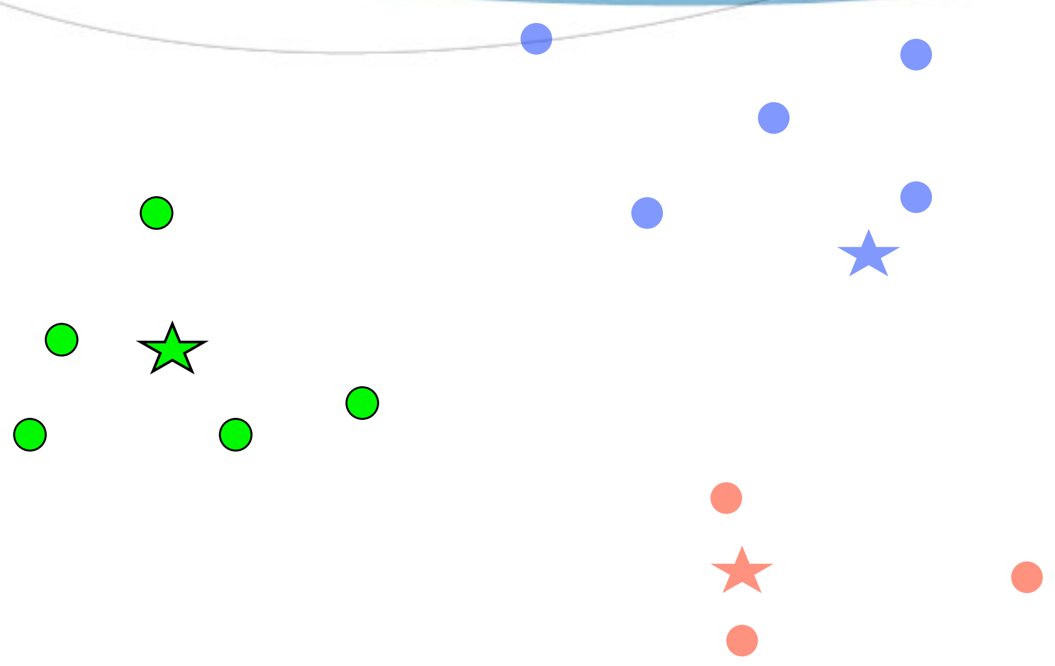
2.1 Calculate center of each cluster

K-Means example (in 2D)



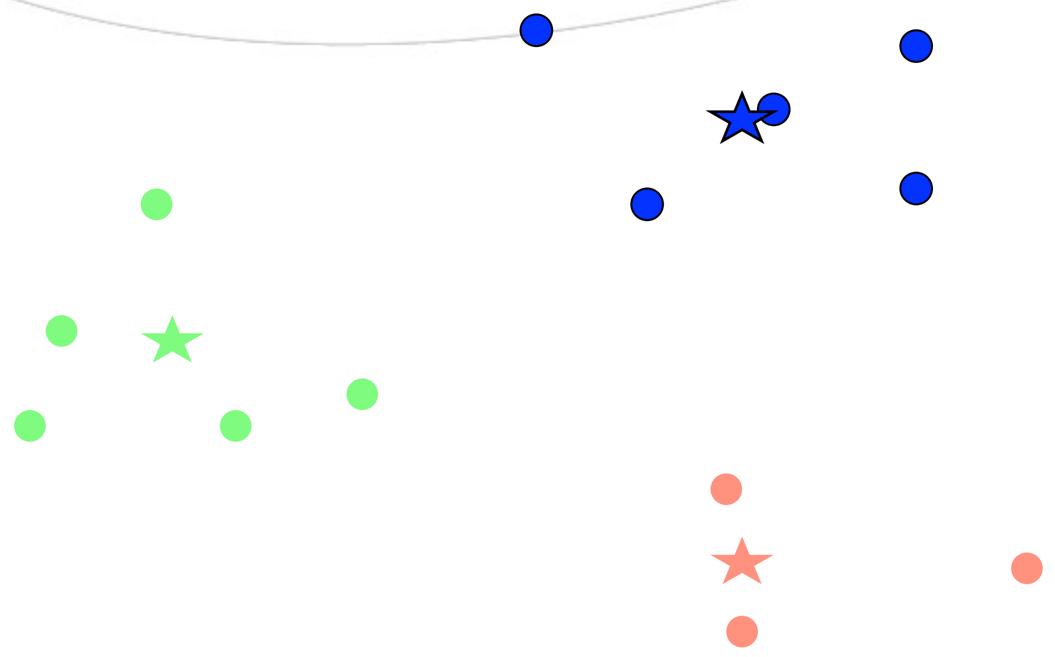
2.2 Assign each point to closest cluster center

K-Means example (in 2D)



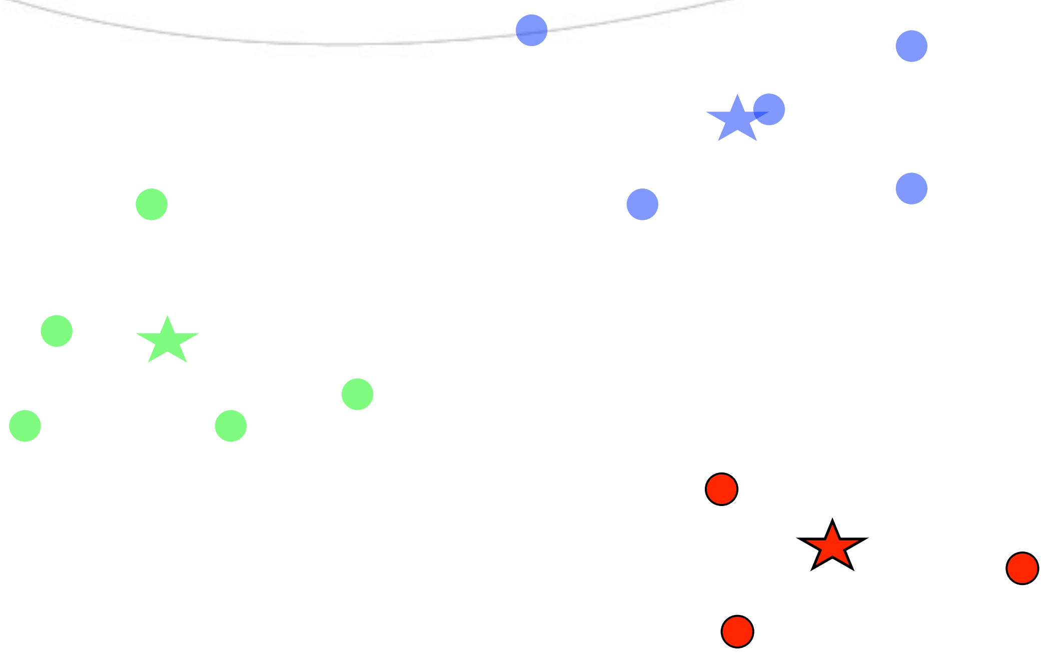
2.1 Calculate center of each cluster

K-Means example (in 2D)



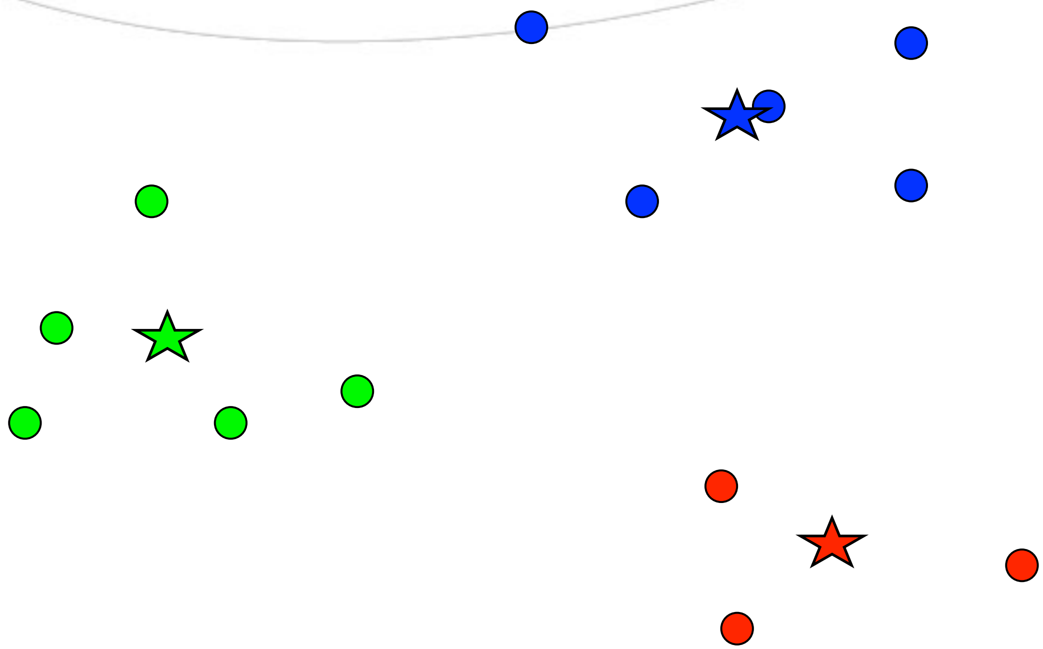
2.1 Calculate center of each cluster

K-Means example (in 2D)



2.1 Calculate center of each cluster

K-Means example (in 2D)



2.2 Assign each point to closest cluster center (no changes)

We have reached **convergence**

Some features

- *Relatively efficient*: $O(tkn)$, where n is # objects, k is # clusters, and t is # iterations. Normally, k, $t << n$
- Often terminates at a *local optimum*. It is good to restart it several times
- Applicable only when *mean* is defined
- Need to specify k, the *number* of clusters, in advance

K-Means in Python

- Sklearn.cluster.Kmeans is the class that implements Kmeans clustering in Scikit-learn.
- Documentation is available at http://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html
- Key options:
  - n_clusters (default 8)
  - n_init (default 10): The number of times to run the algorithm with different centroid seeds
  - init (default "k-means++", other possible values "random" or an ndarray): Methods of initialization for cluster centers
- Outputs:
  - cluster_centers_: Coordinates of the cluster centers
  - labels_: Cluster classification for the observations
  - inertia_: The sum of the squares of distances from the samples to their closest cluster center

Measuring Clustering Effectiveness

- If we don't know the true labels, then we will evaluate the clustering in terms of distances:
  - Inertia
  - Silhouette coefficient: the mean value of (b-a) / max (a , b) where:
    - a is the mean distance between a sample and all other points in the same class
    - b is the mean distance between a sample and all other points in the *next nearest cluster*
  - Calinsky-Harabasz index: the ratio of the overall between-clusters variance to the overall within-cluster variance.

- Documentation is available at *http://scikit-learn.org/stable/modules/clustering.html#clustering-evaluation*

Handwritten Digits Dataset

- The dataset contains images of handwritten digits: 10 classes where each class is one digit from 0 to 9.
- The attributes consist of an 8x8 image of integer pixels in the range 0..16.
- The data is stored in a 1797x64 numpy.ndarray, the rows being the samples and the columns being the 64 attributes.

References

- http://scikit-learn.org/
- http://sebastianraschka.com/Articles/2015_pca_in_3_steps.html
- http://cs.wellesley.edu/~cs315/315_PPTs/...L17-Clustering/k-Means-Clustering-Example.ppt
- https://cs.brown.edu/courses/cs143/lectures/17.ppt