BC COMS 2710: Computational Text Analysis

Lecture 18 – Clustering

Announcements – Assignments

- Readings 05:
  - link posted to course site
  - due Sunday
- HW 03:
  - Released last Friday
  - Optional – has anyone looked at it?
- HW04/Tutorial 5.1
  - Releasing today or Friday
  - Based on last Thursday’s and this week’s material

Final Project – Deliverables

- Project ideation – Friday May 28st
  - 5 points
- Project proposal – Sunday June 6th
  - 9 points
- Project presentations – Monday June 14th
  - 6 points
- Project submissions – Friday June 18th
  - 15 points
- http://coms2710.barnard.edu/final_project

Final Project – Proposal due Sunday June 6th

Beefed up version of project ideation

1. Research Question
2. Detailed source of data:
    1. List of twitter user’s, subreddits, etc
3. Detailed methods you plan on applying for exploratory data analysis
    1. Tf-idf, topic modeling, …
4. Prediction

Logistic Regression

Summary of Logistic Regression

- Optimizes $\mathrm{P}(Y\,|X)$ directly
- Define the features
- Learn a vector of weights for each label $y \in Y$
  - Gradient descent, update weights based on error
- Multiple feature vector by weight vector
- Output is $\mathrm{P}(Y = y\,|X)$ after normalizing
- Choose the most probable Y

Slide from Nate Chambers

Parameters

Parameters in …

- Logistic Regression:
  - Weights

- Naive Bayes:
  - Priors – $P(Y)$
  - Likelihoods - $P(x_i|Y)$

- Values that are estimated from data

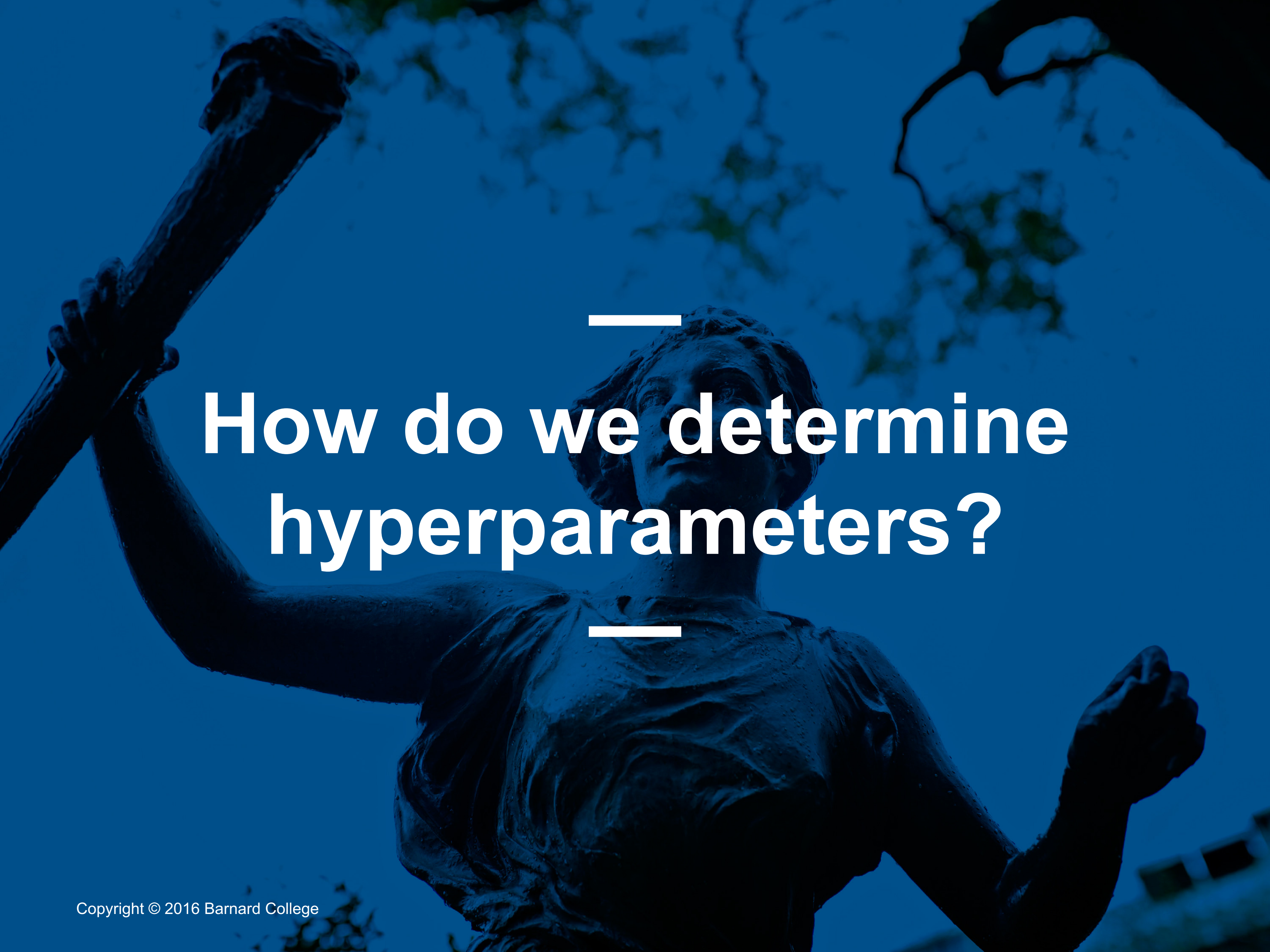
Hyperparameters

- a configuration that is external to the model and whose value cannot be estimated from data
- ***"If you have to specify a value manually, then it is probably a model hyperparameter"***

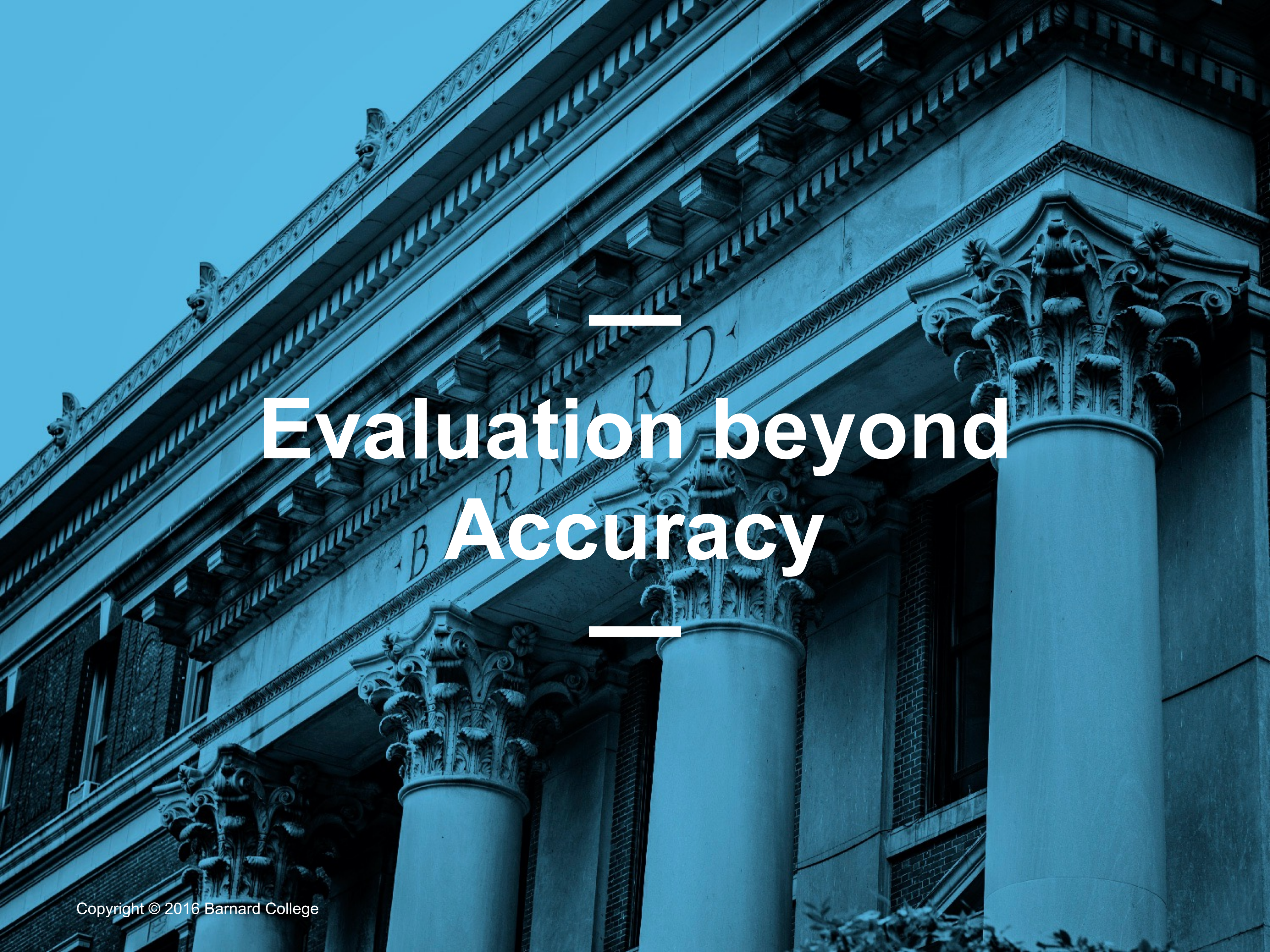
[Machine Learning Mastery]

Hyperparameters in …

- Logistic Regression:
  - max_iters
- Naive Bayes:
  - Value for smoothing
    - Add-one smoothing (Laplacian smoothing)
- LDA
  - ***k*** – the number of topics



How do we determine hyperparameters?



Evaluation beyond Accuracy

Confusion Matrix

	gold positive	gold negative
positive prediction	True Positive **(TP)**	False Positive **(FP)**
negative prediction	False Negative **(FN)**	True Negative **(TN)**

Metrics from the confusion matrix: accuracy

	gold positive	gold negative
positive prediction	True Positive **(TP)**	False Positive **(FP)**
negative prediction	False Negative **(FN)**	True Negative **(TN)**

Accuracy: $\frac{TP+TN}{TP+FP+TN+FN}$

Metrics from the confusion matrix: precision

	gold positive	gold negative
positive prediction	True Positive **(TP)**	False Positive **(FP)**
negative prediction	False Negative **(FN)**	True Negative **(TN)**

% of items the system detected (i.e., items the system labeled as positive) that are in fact positive (according to the human gold labels)

Precision: $\frac{TP}{TP+FN}$

Definition from Dan Jurafsky

Metrics from the confusion matrix: recall

	gold positive	gold negative
positive prediction	True Positive **(TP)**	False Positive **(FP)**
negative prediction	False Negative **(FN)**	True Negative **(TN)**

% of items actually present in the input that were correctly identified by the system.

Recall: $\frac{TP}{TP+FP}$

Definition from Dan Jurafsky

Metrics from the confusion matrix: F1

	gold positive	gold negative
positive prediction	True Positive **(TP)**	False Positive **(FP)**
negative prediction	False Negative **(FN)**	True Negative **(TN)**

Combine precision and recall

$$F_1 = \frac{2PR}{P+R}$$

Clustering

Different Types of Machine Learning

- Supervised Learning
  - Given labeled examples, learn rules

- Unsupervised Learning
  - Given unlabeled example, learn patterns

Slide from Tony Liu

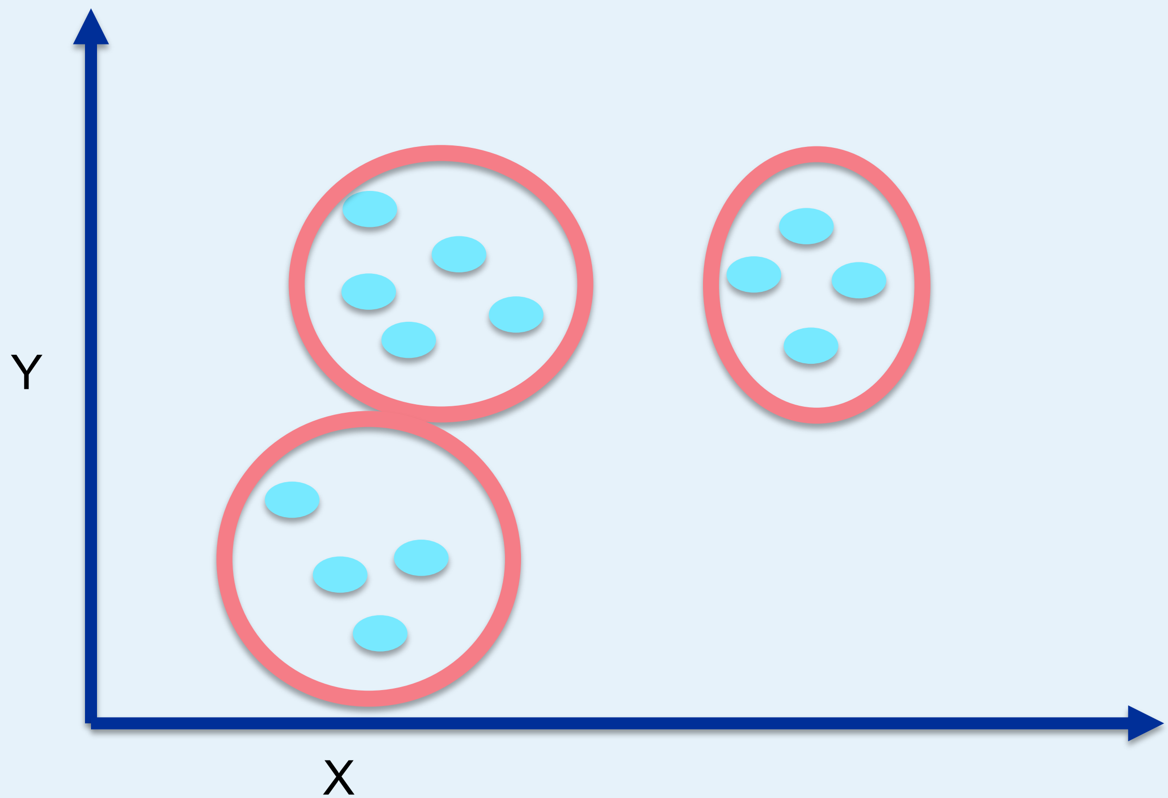
Clustering

- Unsupervised learning
  - Requires data, but no labels
- Detect patterns e.g. in
  - Group emails
  - Group obituaries
  - Group any documents
- Useful when don't know what you're looking for
- Good way to explore your data

Slide from David Sontag

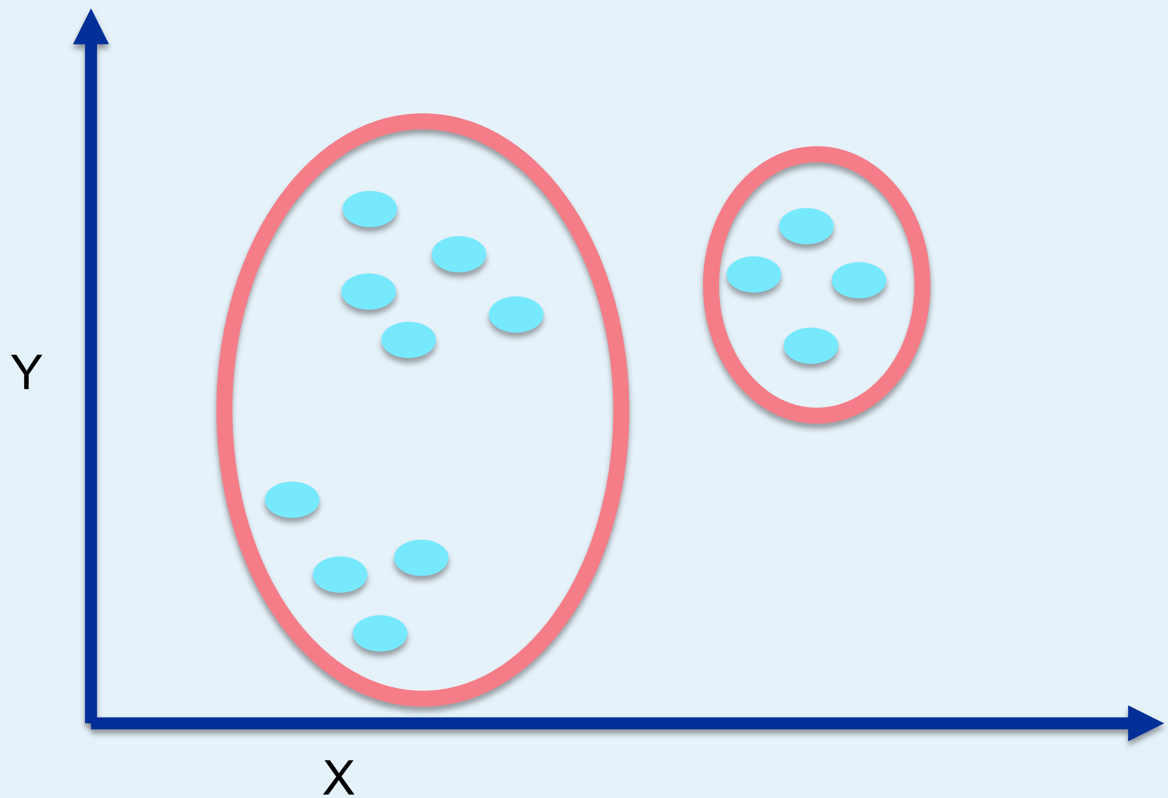
Idea: group together similar instances

- Example: 2D point patterns



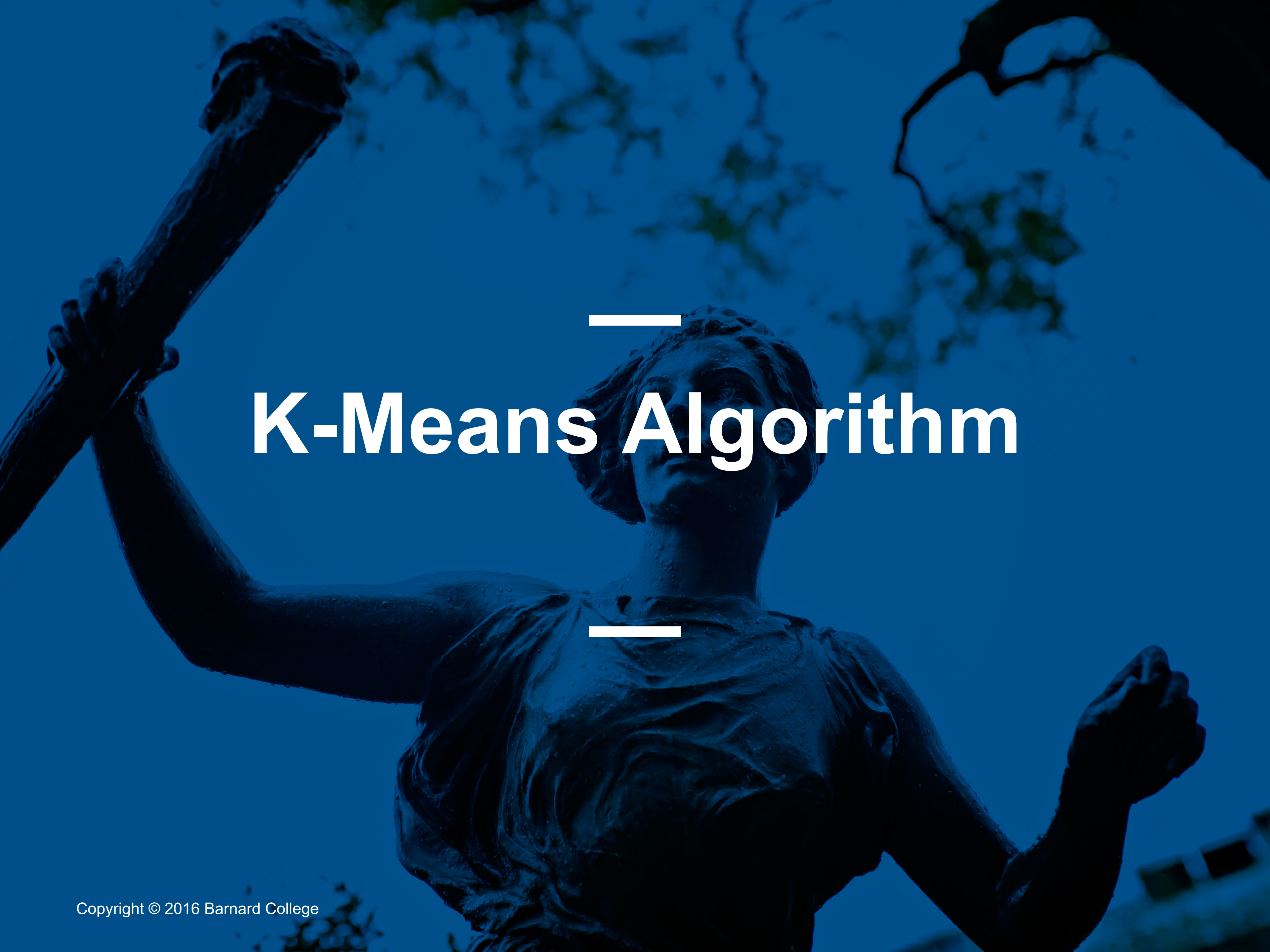
Idea: group together similar instances

- Example: 2D point patterns



Clustering HW02

- HW02 analyzed obits
- Why might we want to cluster obits?
  - Find groups of similar obituaries
  - Find topics of obituaries
  - …



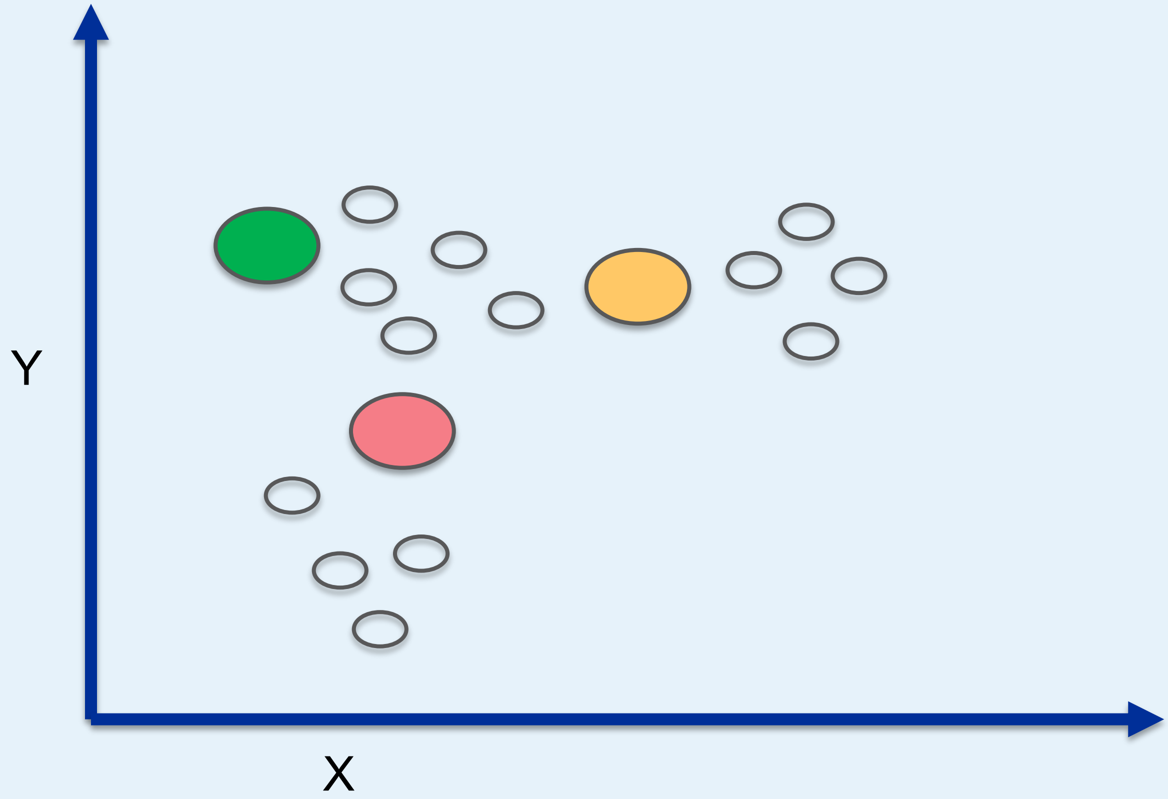
K-Means Algorithm

K-means Algorithms

1. Initialize: Randomly pick K points as cluster centers

Randomly pick K points as centers

- Example: 2D point patterns



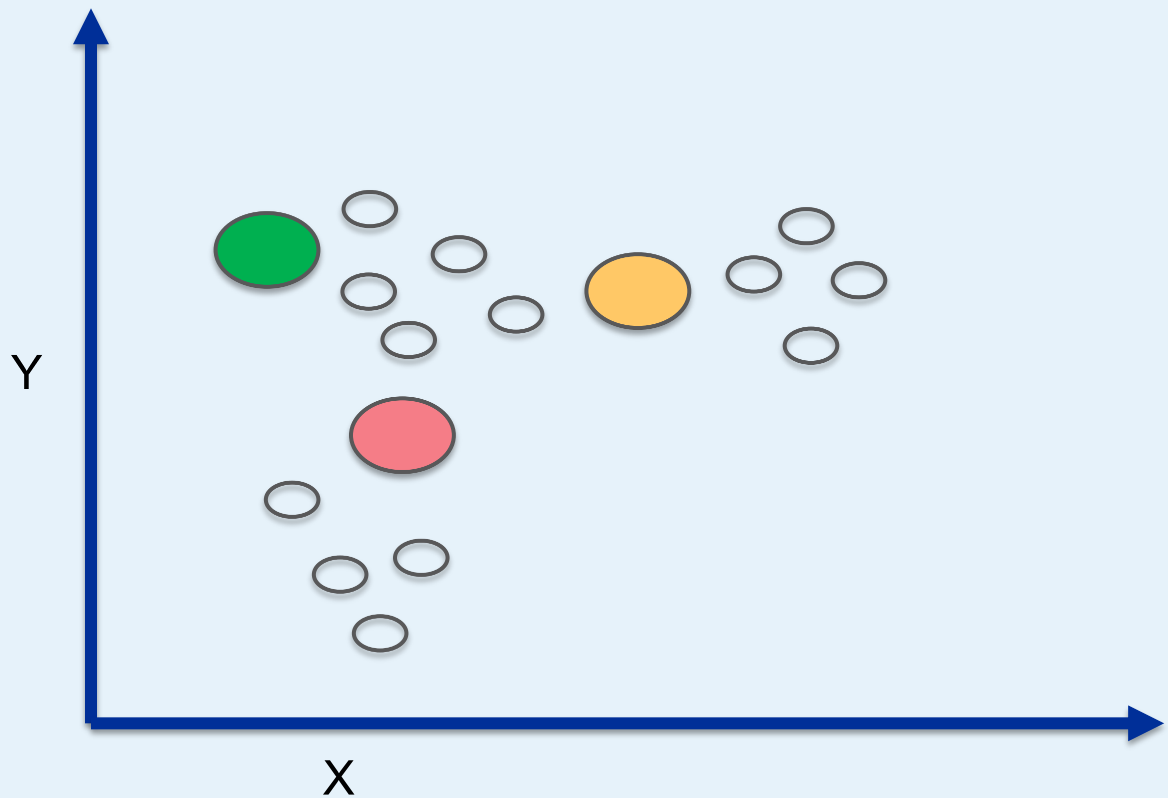
K-means Algorithms

1. Initialize: Randomly pick K points as cluster centers

2. Assign data points to each cluster
    1. Based on distance between point and cluster's center

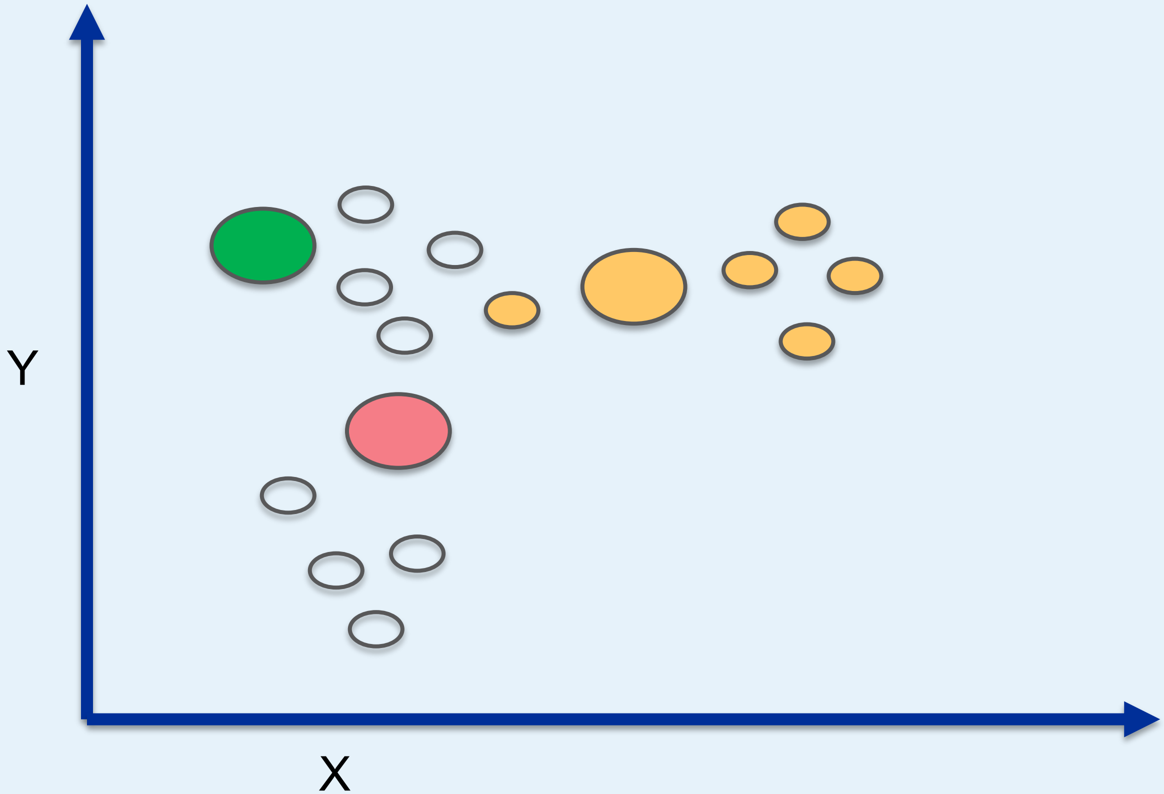
Assign data points to each cluster

- Example: 2D point patterns



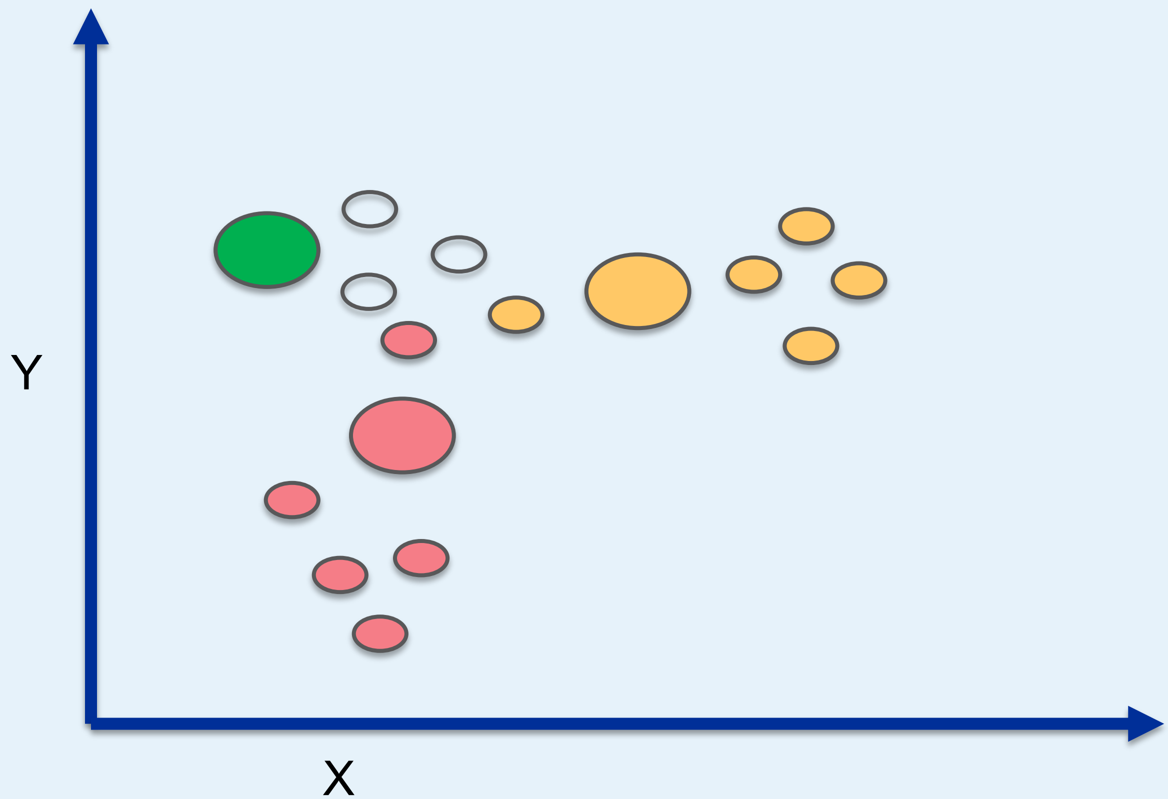
Assign data points to each cluster

- Example: 2D point patterns



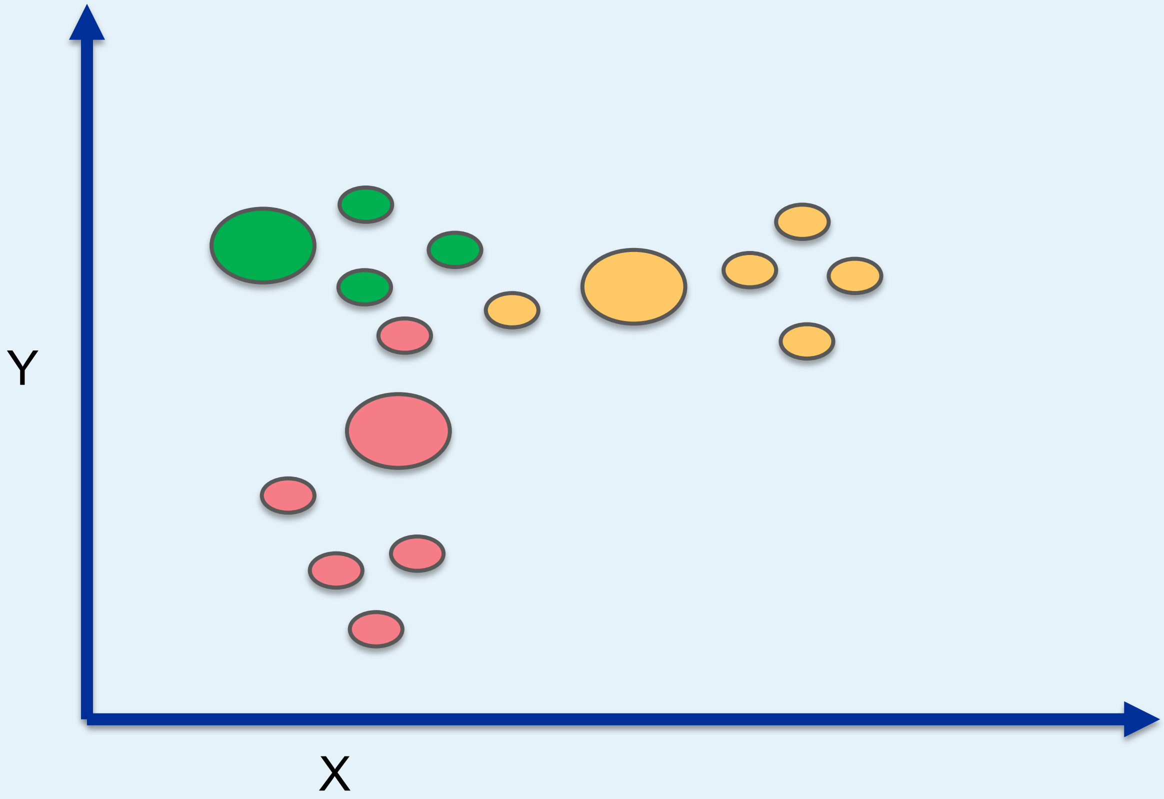
Assign data points to each cluster

- Example: 2D point patterns



Assign data points to each cluster

- Example: 2D point patterns



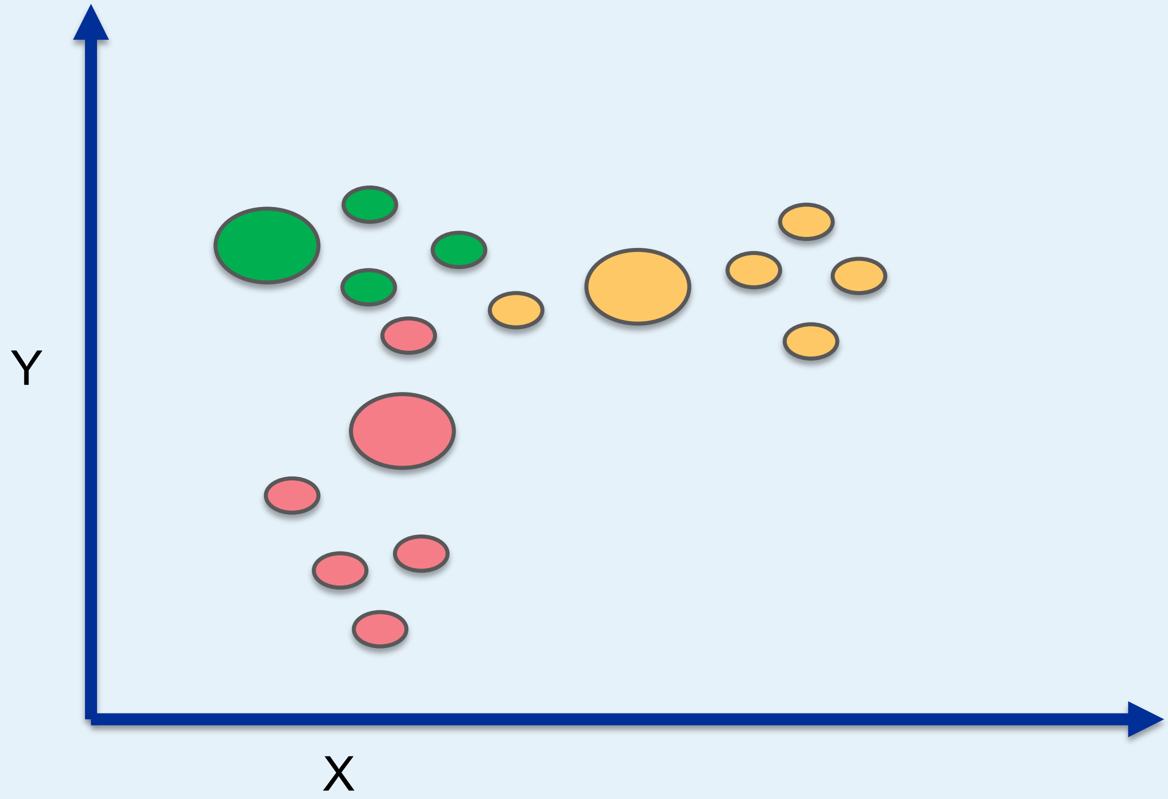
K-means Algorithms

1. Initialize: Randomly pick K points as cluster centers

2. Assign data points to each cluster
   1. Based on distance between point and cluster’s center
3. Update the center of each cluster
   1. The average of its assigned points

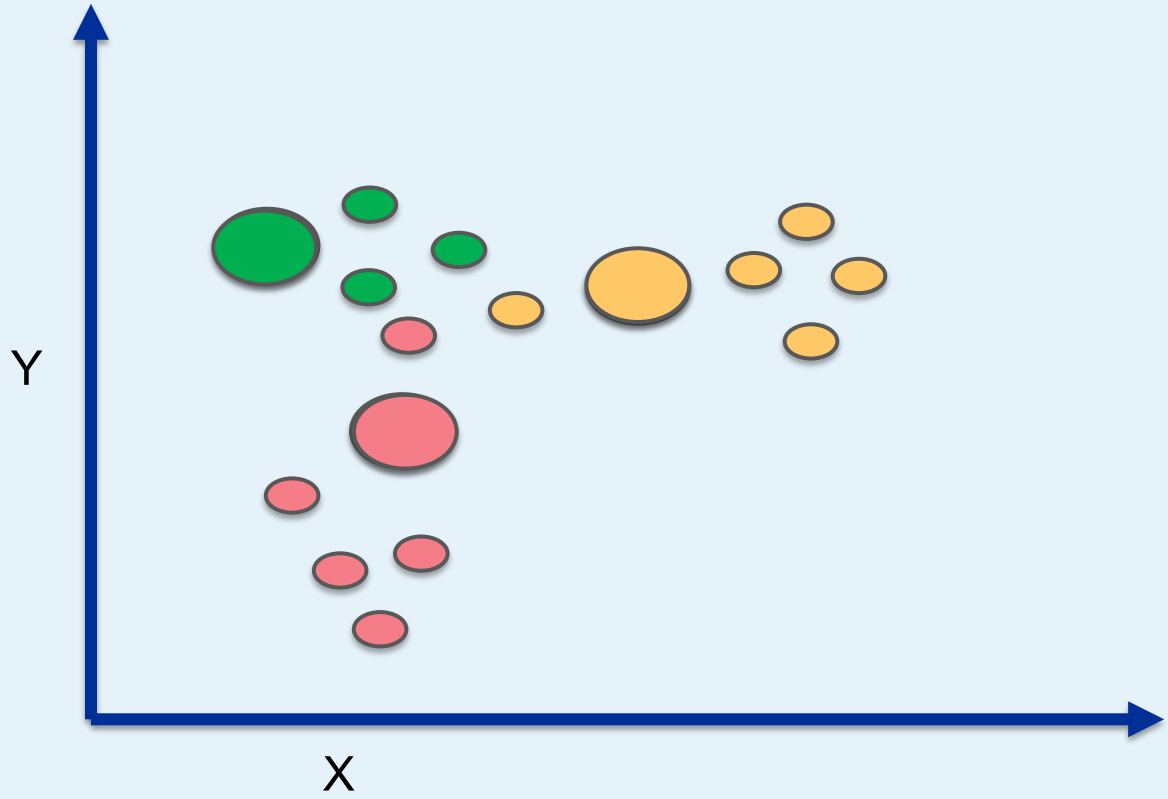
Update Centers

- Example: 2D point patterns



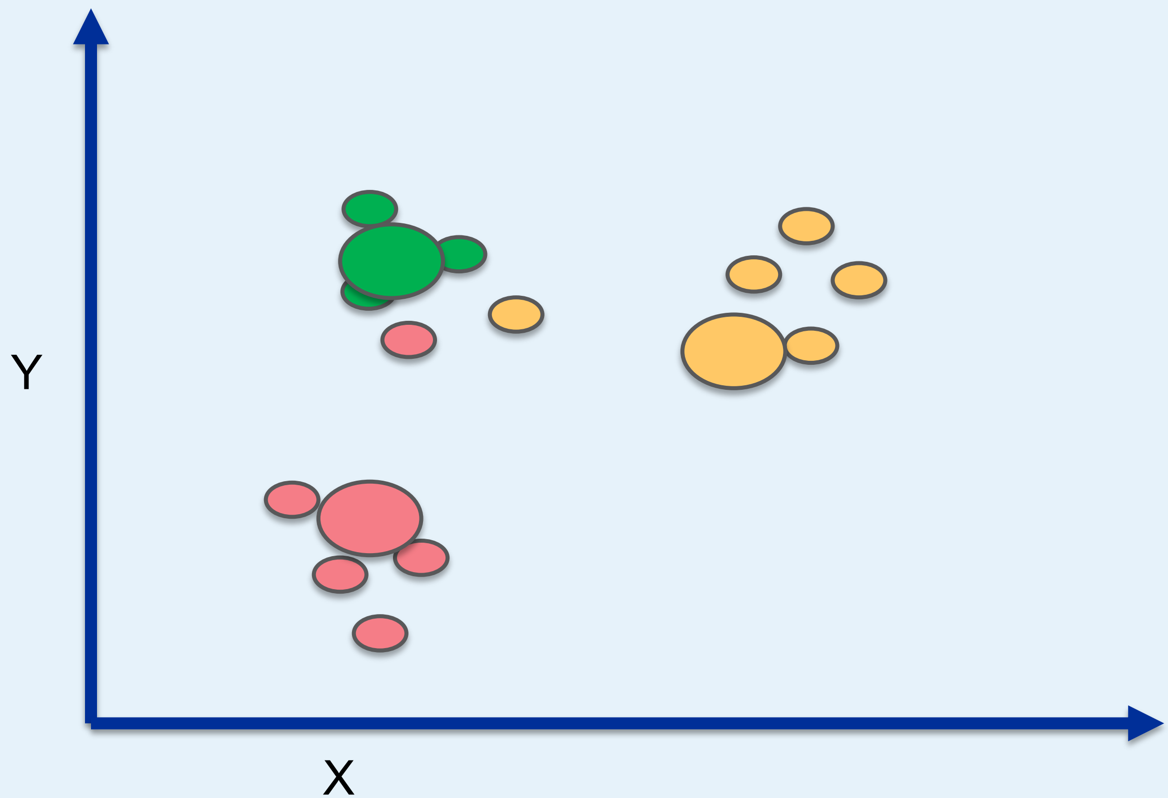
Update Centers

- Example: 2D point patterns



Updated Centers

- Example: 2D point patterns

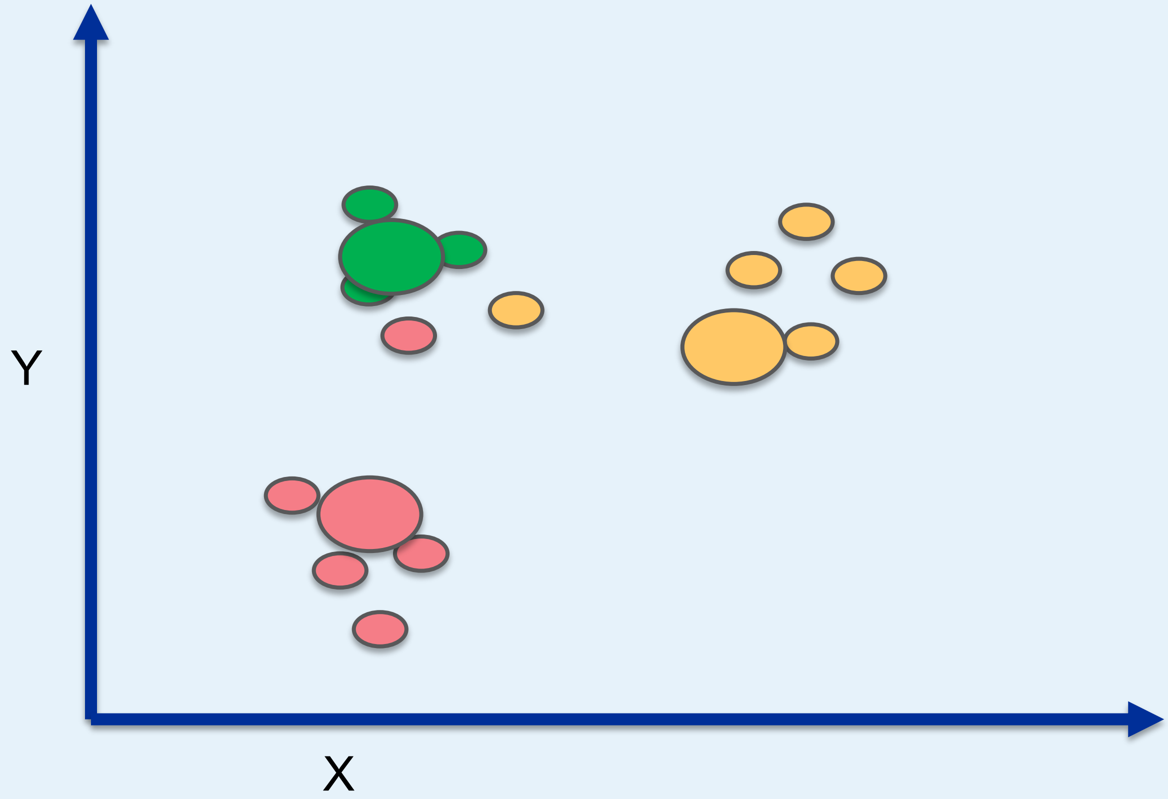


K-means Algorithms

1. Initialize: Randomly pick K points as cluster centers
2. Assign data points to each cluster
    1. Based on distance between point and cluster's center
3. Update the center of each cluster
    1. The average of its assigned points
4. Repeat 2 & 3 until the assignments stop changing

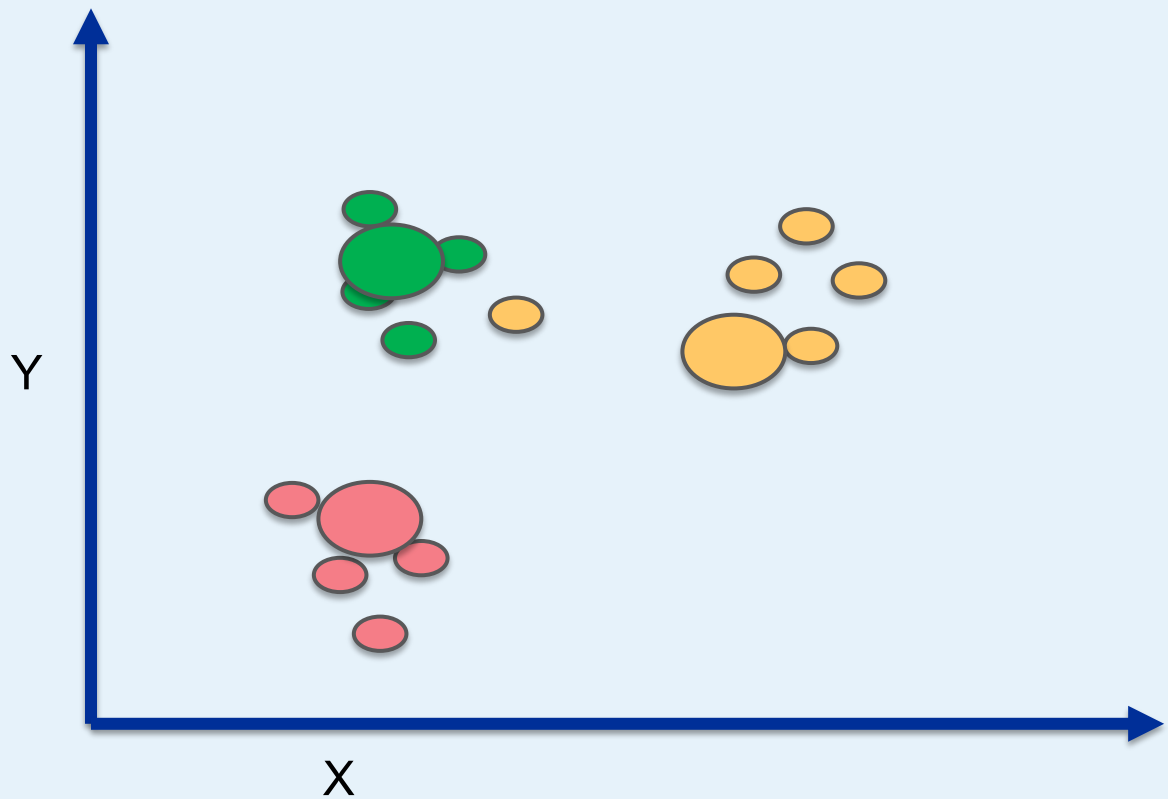
Reassign data points to each cluster

- Example: 2D point patterns



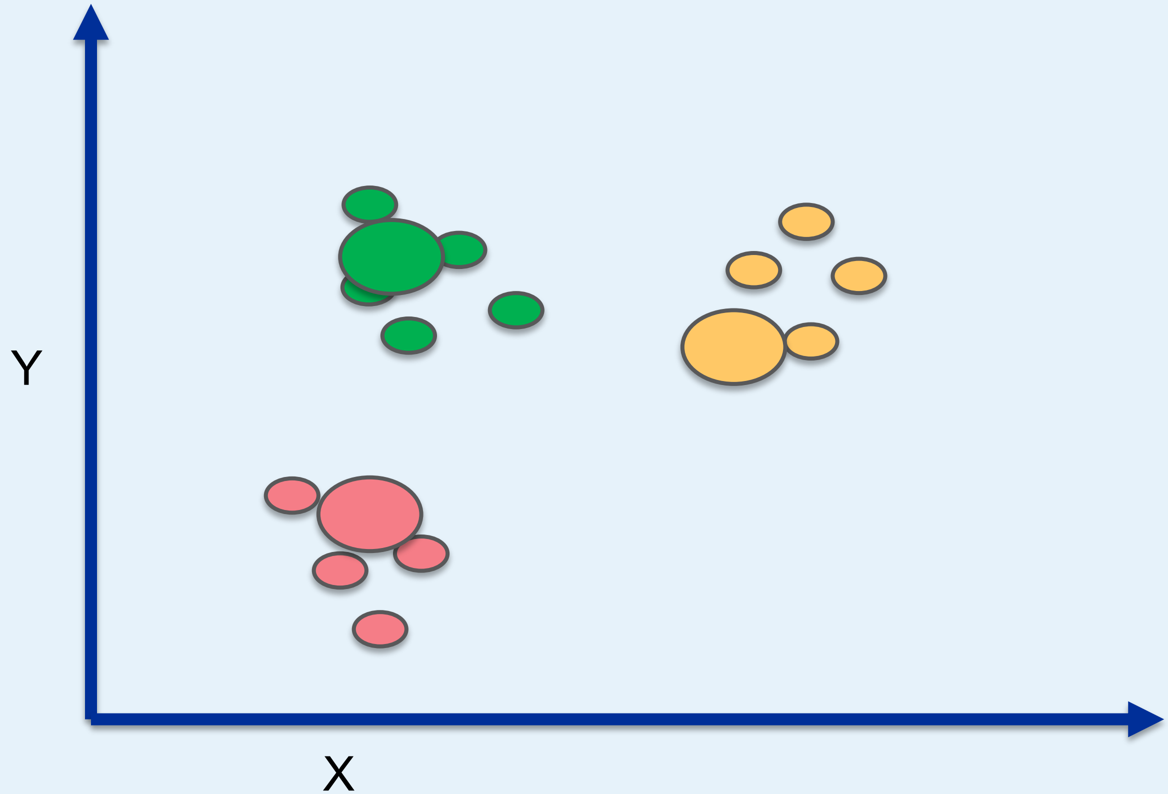
Reassign data points to each cluster

- Example: 2D point patterns



Reassign data points to each cluster

- Example: 2D point patterns



K-means Algorithms

1. Initialize: Randomly pick K points as cluster centers

2. Assign data points to each cluster
    1. Based on distance between point and cluster's center
3. Update the center of each cluster
    1. The average of its assigned points

4. Repeat 2 & 3 until the assignments stop changing

K-means Algorithms

1. Initialize: Randomly pick K points as cluster centers

2. Assign data points to each cluster
   1. Based on **distance between** point and cluster's center
3. Update the center of each cluster
   1. The average of its assigned points

4. Repeat 2 & 3 until the assignments stop changing

How do we quantify similarity/distance?

We need to define similarity/distance

Similarity metrics we've seen so far:

cos similarity

Euclidian distance between two documents x_1 and x_2

$$D = \sqrt{\sum_i (x_{1_i} - x_{2_i})^2}$$

Week 6

- Monday 06/07 - Matrix Factorization
- Tuesday 06/08 – Word Embeddings
- Wednesday 06/09 – Guest Lecture
  - Attendance required
- Thursday 06/10 - TBA