GENERAL ASSEMBLY

UNSUPERVISED LEARNING

Trevor Lindsay

LEARNING OBJECTIVES

- Understand the difference between supervised and unsupervised learning algorithms
- Understand and apply k-means clustering to an unlabeled dataset
- Use the Silhouette Coefficient metric to measure the performance of the k-means algorithm

OPENING

UNSUPERVISED LEARNING

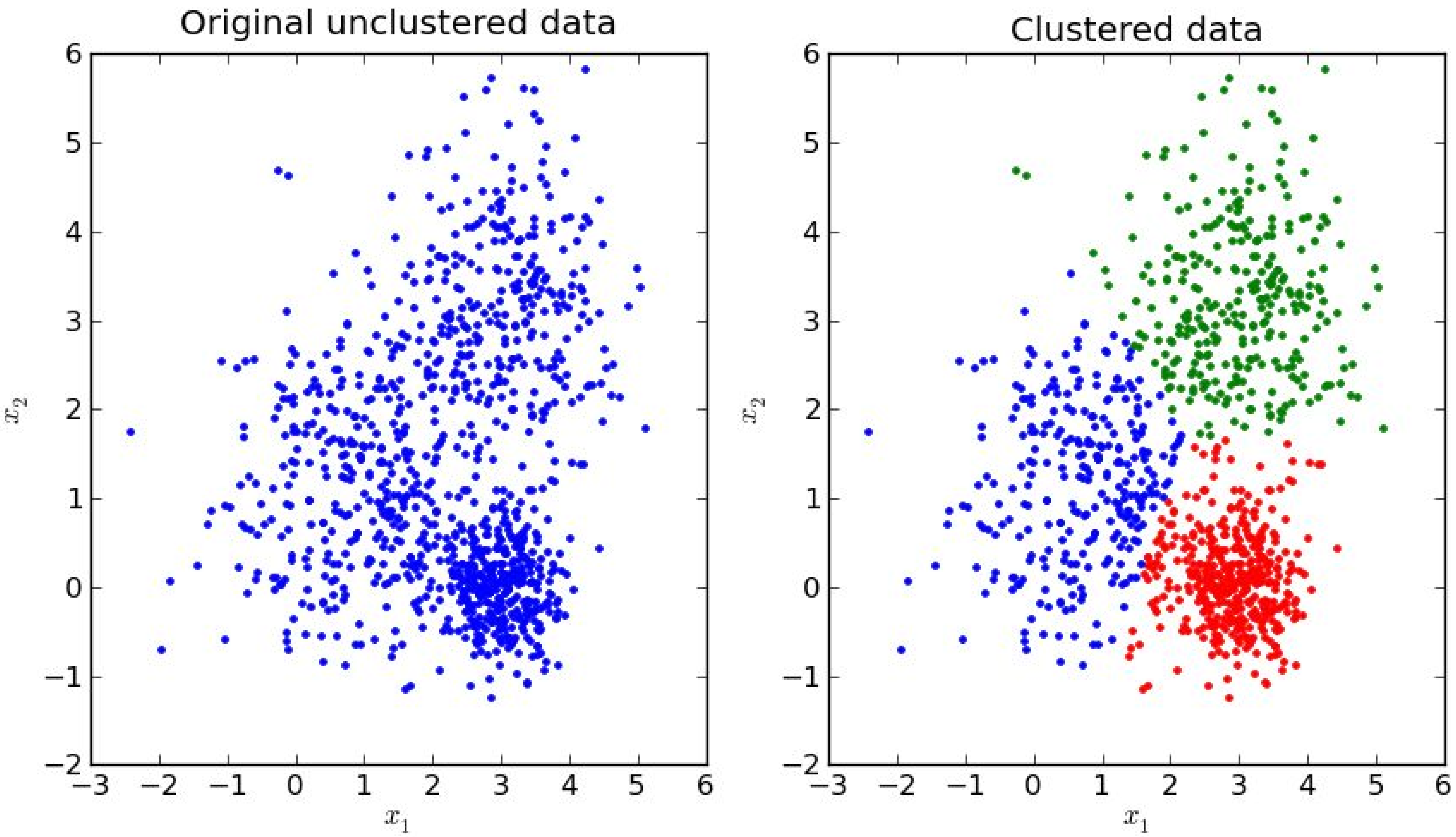
UNSUPERVISED LEARNING

- So far all the algorithms we have used are supervised: each observation (row of data) came with one or more *labels*, either *categorical variables* (classes) or *measurements* (regression)
- Unsupervised learning has a different goal: **finding structure**
- **Clustering** is a common and fundamental example of unsupervised learning
- Clustering algorithms try to find meaningful groups within data

INTRODUCTION

CLUSTERING

CLUSTERING


Original unclustered data
Clustered data
x_1
x_2
−3 −2 −1 0 1 2 3 4 5 6
−2 −1 0 1 2 3 4 5 6

ACTIVITY: THINK-PAIR-SHARE

EXERCISE

ANSWER THE FOLLOWING QUESTIONS ALONE THEN WITH A PARTNER

1. How is unsupervised learning different from classification?
2. Can you think of a real-world clustering application?

DEEP DIVE

K-MEANS CLUSTERING

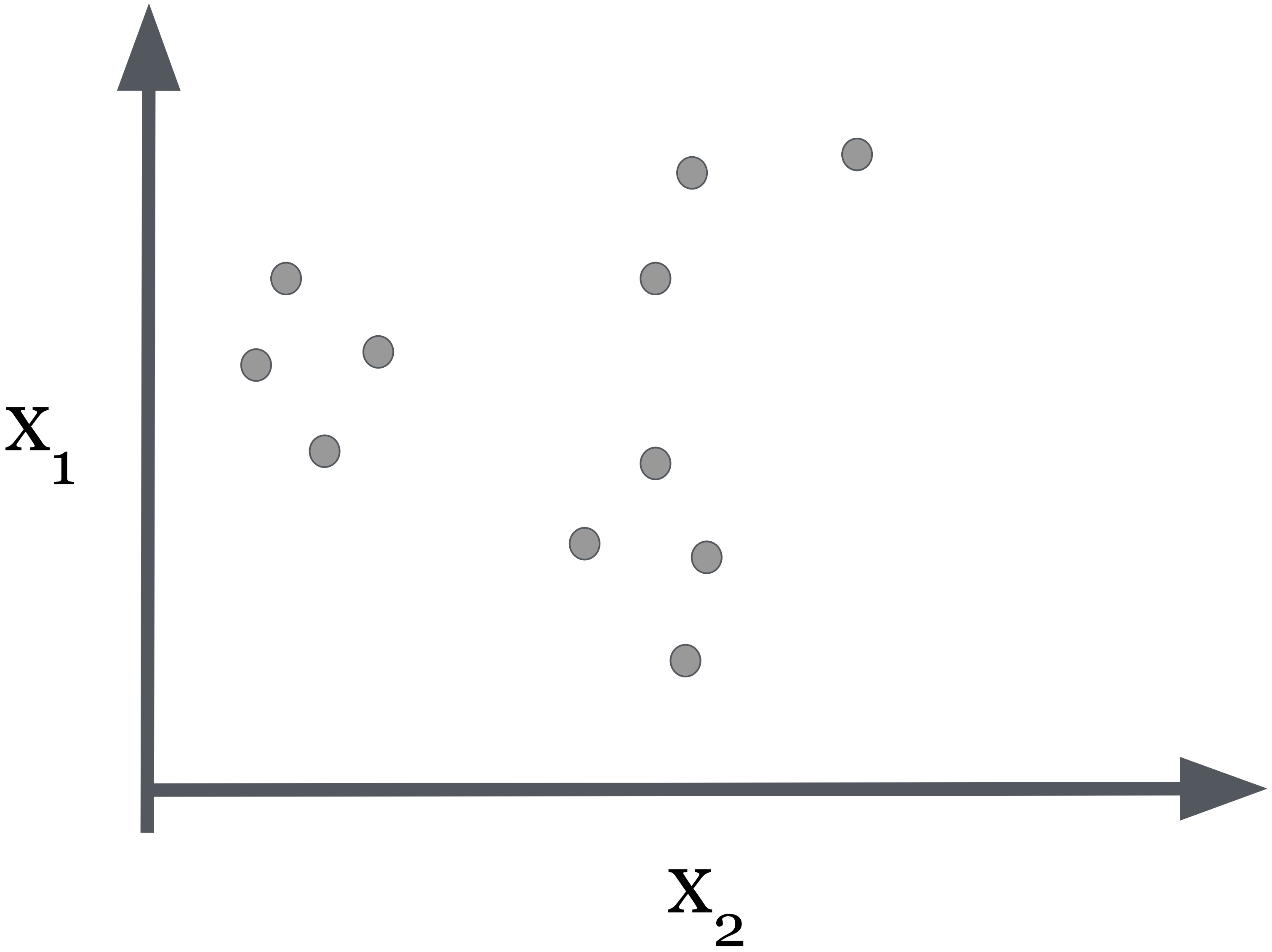
K-MEANS CLUSTERING

- K-means clustering aims to partition n observations into k clusters in which each observation belongs to exactly one cluster
- The mean of the cluster (a.k.a. the *centroid*) serves as a *prototype*
- How does the algorithm work?

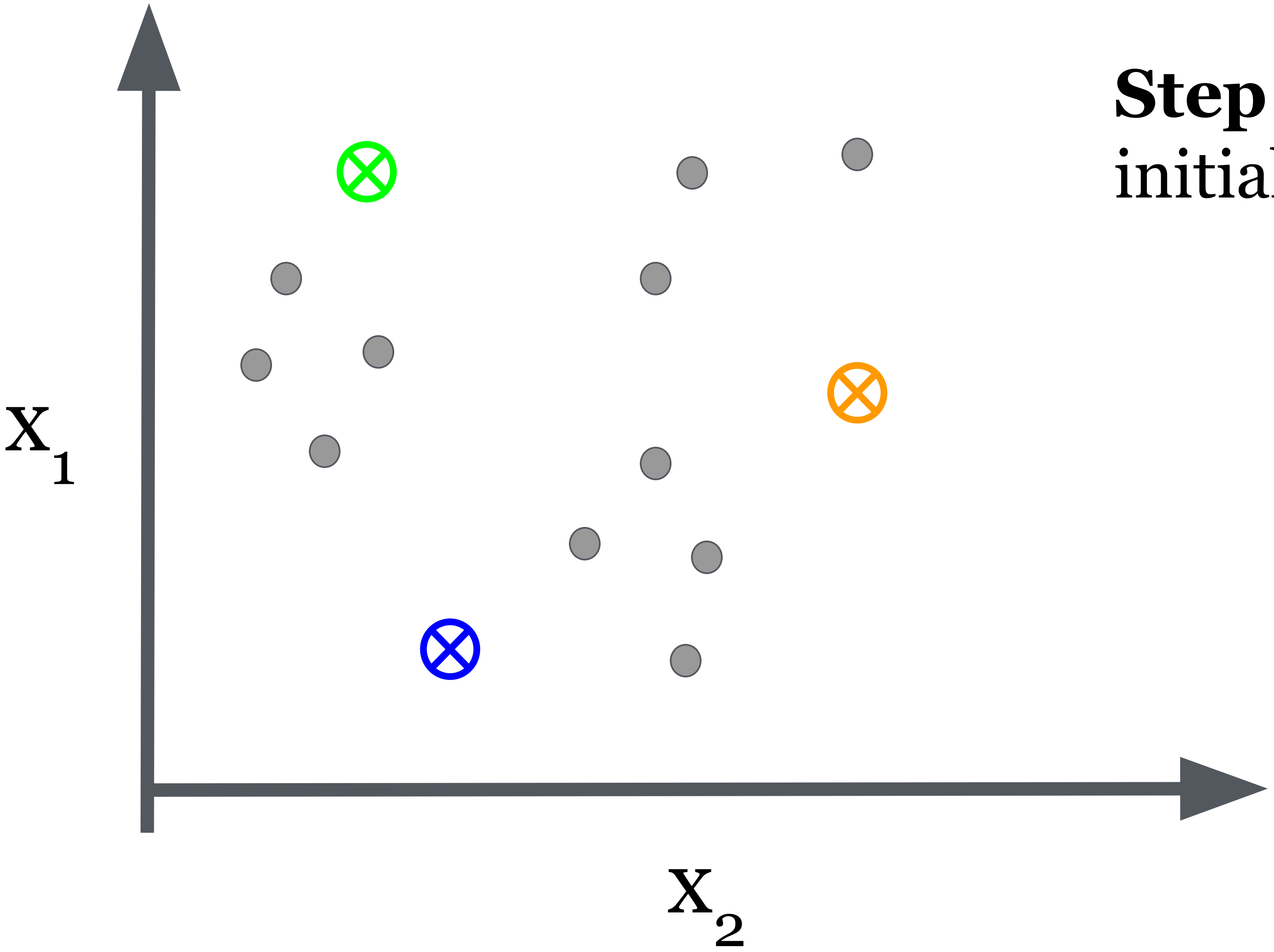
K-MEANS CLUSTERING

1. Choose *k* initial centroids (note that *k* is an input determined by you)
2. For each point, calculate its similarity to each centroid
3. Assign each point to the nearest / most similar centroid based on the chosen measure of similarity
4. Recalculate the centroid positions
5. Repeat steps 2-4 until some stopping criteria is met

K-MEANS CLUSTERING

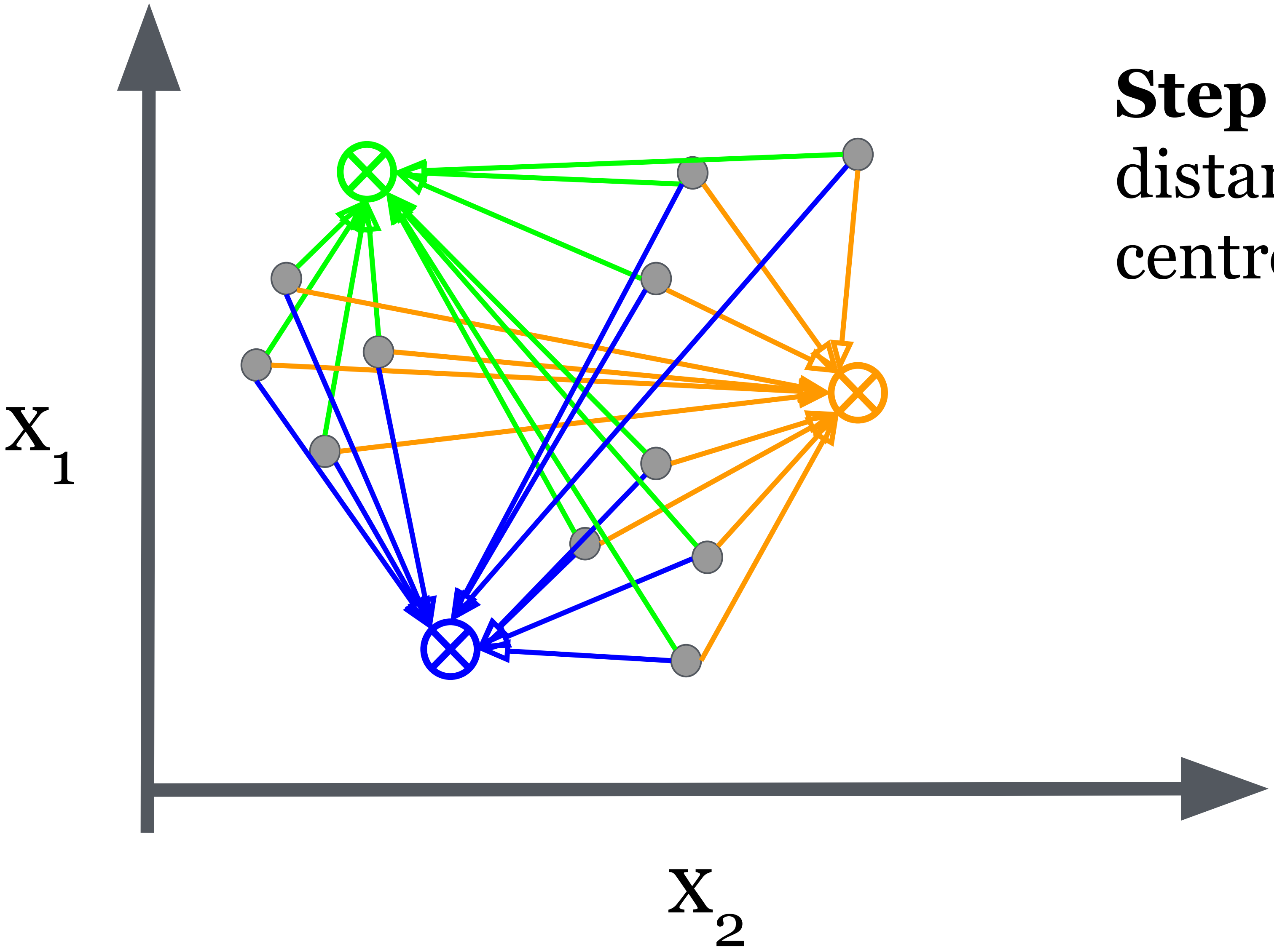


K-MEANS CLUSTERING



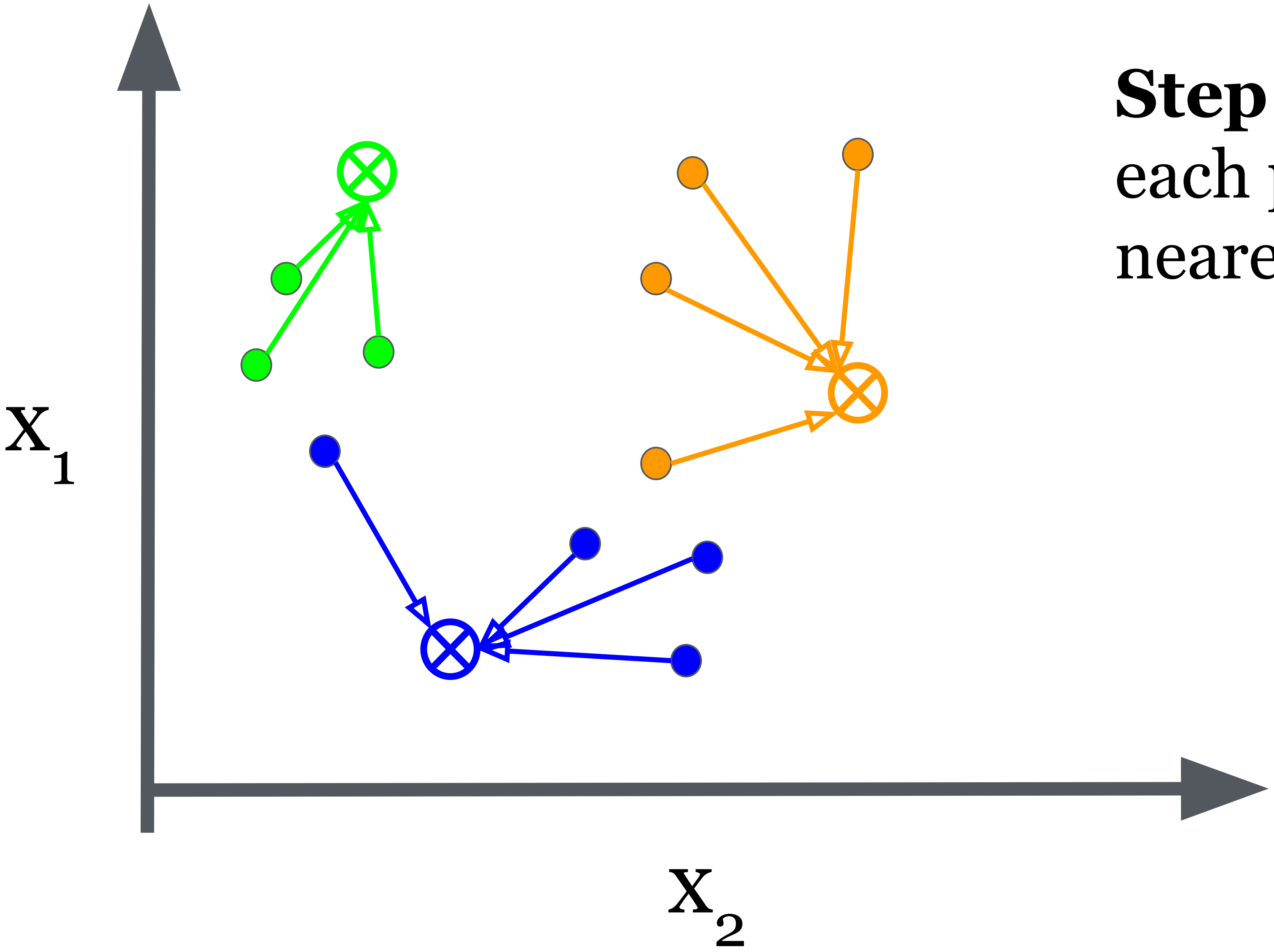
Step 1: Choose *k* initial centroids.

K-MEANS CLUSTERING



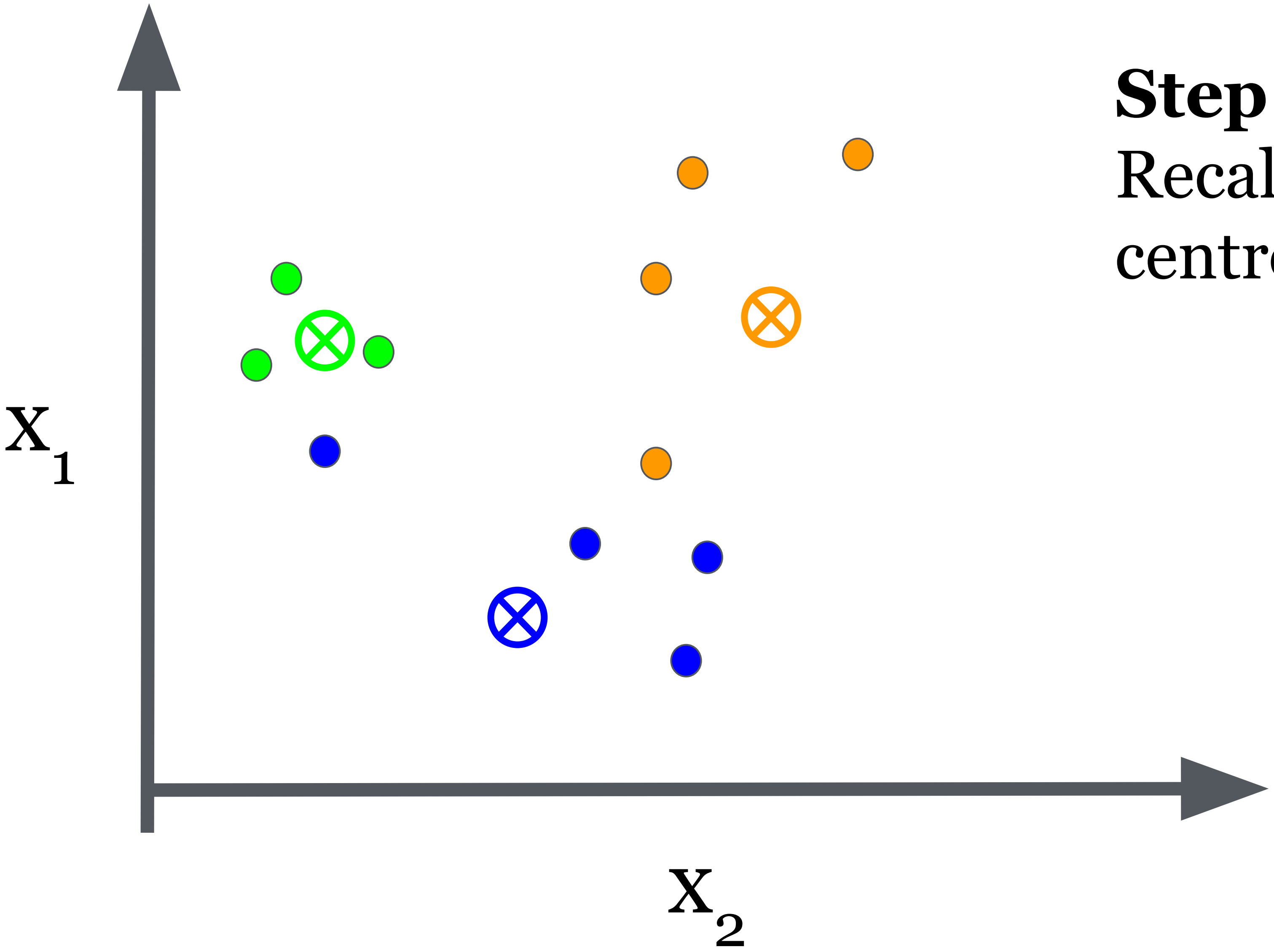
Step 2: Find distance to each centroid

K-MEANS CLUSTERING



Step 3: Assign each point to the nearest centroid

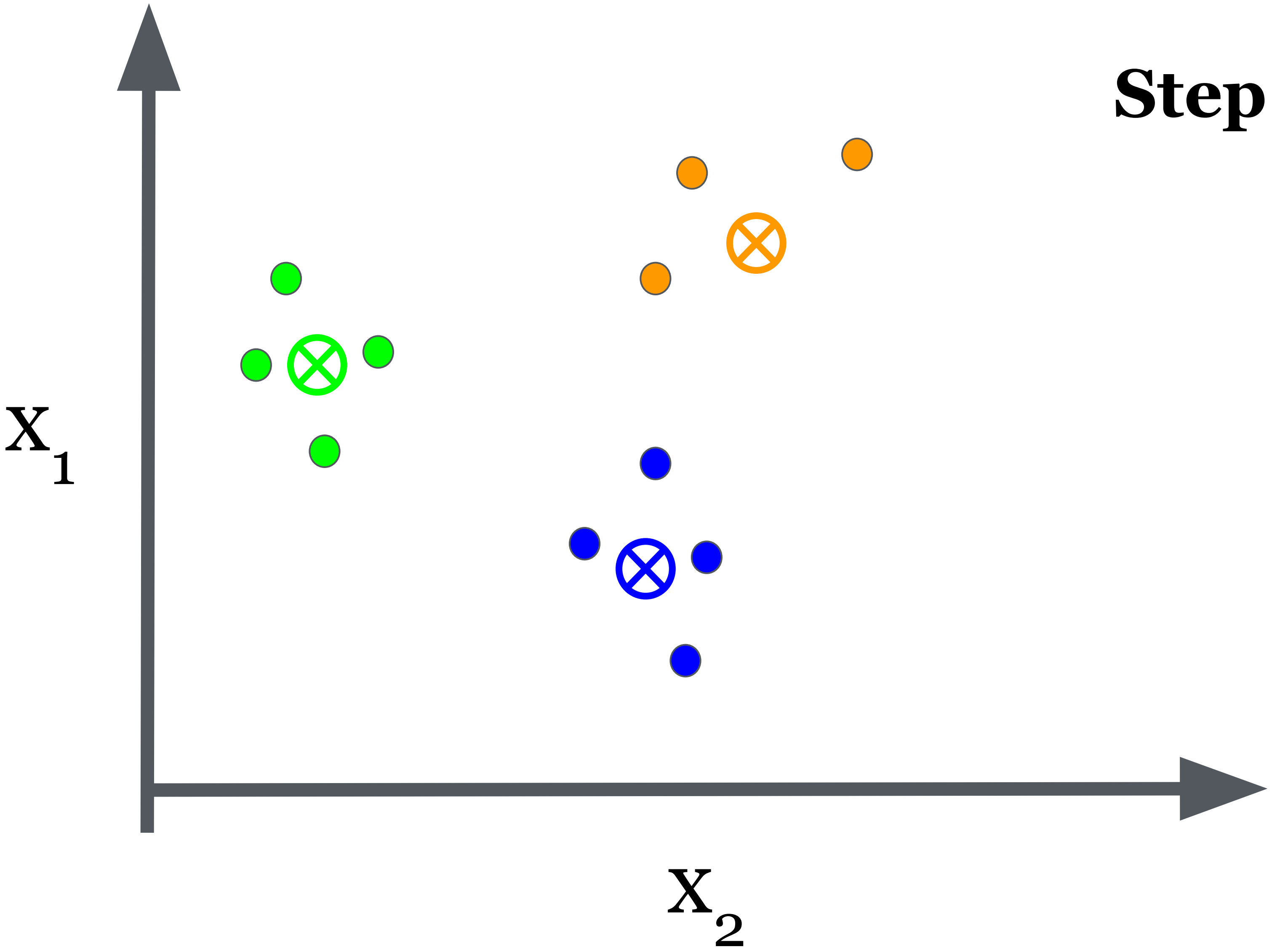
K-MEANS CLUSTERING



Step 4:
Recalculate the centroid positions

K-MEANS CLUSTERING

Step 5: Repeat



ASSESSING SIMILARITY

- How do you determine which centroid a given point is most similar to?
- The similarity criterion is determined by the measure we choose
- In the case of k-means clustering, the most common similarity measure is Euclidean distance

RECOMPUTING THE CENTER

- How de we recompute the position of the centers at each iteration of the algorithm?
- The centroid is calculated as the geometric center of the points in the cluster
- This is done by taking the average of each index of vectors
  - Centroid of [1, 4, 2] and [6, 4, 2]
  - [(1 + 6) / 2, (4 + 4) / 2, (2 + 2) / 2] = [3.5, 4, 2]

CONVERGENCE

- We iterate until some stopping criteria is/are met; in general, suitable convergence is achieved in a small number of steps
- The most common stopping criteria is no change in the assignment of data points to clusters

DEMO

https://www.naftaliharris.com/blog/visualizing-k-means-clustering/

GUIDED PRACTICE

K-MEANS CLUSTERING WITH SCIKIT-LEARN

K-MEANS CLUSTERING WITH SCIKIT-LEARN

```python
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns

from sklearn import metrics
from sklearn.cluster import KMeans
from sklearn.datasets import make_classification

X, y = make_classification(
    n_samples=100,
    n_features=2,
    n_redundant=0,
    n_classes=4,
    n_clusters_per_class=1,
    random_state=40,
)

df = pd.DataFrame(X, columns=['feature1', 'feature2'])
```

ACTIVITY: SKILL CHECK

EXERCISE

COMPLETE THE FOLLOWING TASKS

1. With a partner, plot the data as a scatter plot with feature 1 on the x-axis and feature 2 on the y-axis.

2. **Bonus:** use the label (y) to color the data points

K-MEANS CLUSTERING WITH SCIKIT-LEARN

```
color_map = {0: 'red', 1: 'blue', 2: 'black', 3: 'green'}
colors = y.map(color_map).values

df.plot(x='feature1', y='feature2', kind='scatter', figsize=(10,8), c=colors, s=50)
```

K-MEANS CLUSTERING WITH SCIKIT-LEARN

```
estimator = KMeans(n_clusters=4)
estimator.fit(df)

labels = estimator.labels_
```

K-MEANS CLUSTERING WITH SCIKIT-LEARN

```
color_map = {0: 'red', 1: 'blue', 2: 'black', 3: 'green'}
colors = pd.Series(labels).map(color_map).values


df.plot(x='feature1', y='feature2', kind='scatter', figsize=(10,8), c=colors, s=50)
```

ACTIVITY: THINK-PAIR-SHARE

EXERCISE

ANSWER THE FOLLOWING QUESTIONS ALONE THEN WITH A PARTNER

1. Run the k-means clustering model again, but with only 2 clusters then with 6 clusters

2. How do we assign meaning to the clusters we find?

3. Do clusters always have meaning?

K-MEANS CLUSTERING

- Assumptions are important! k-Means assumes:
  - *k* is the correct number of clusters
  - the data is isotropically distributed (circular/spherical distribution)
  - the variance is the same for each variable
  - clusters are roughly the same size
- Nice counterexamples / cases where assumptions are not met:
  - http://varianceexplained.org/r/kmeans-free-lunch/
  - Scikit-Learn Examples

GUIDED PRACTICE

CLUSTERING METRICS

CLUSTERING METRICS

- As usual we need a metric to evaluate model fit
- For clustering we use a metric called the Silhouette Coefficient
    - a is the mean distance between a sample and all other points in the cluster
    - b is the mean distance between a sample and all other points in the nearest cluster
    - Ranges between 1 and -1
    - Average over all points to judge the cluster algorithm

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