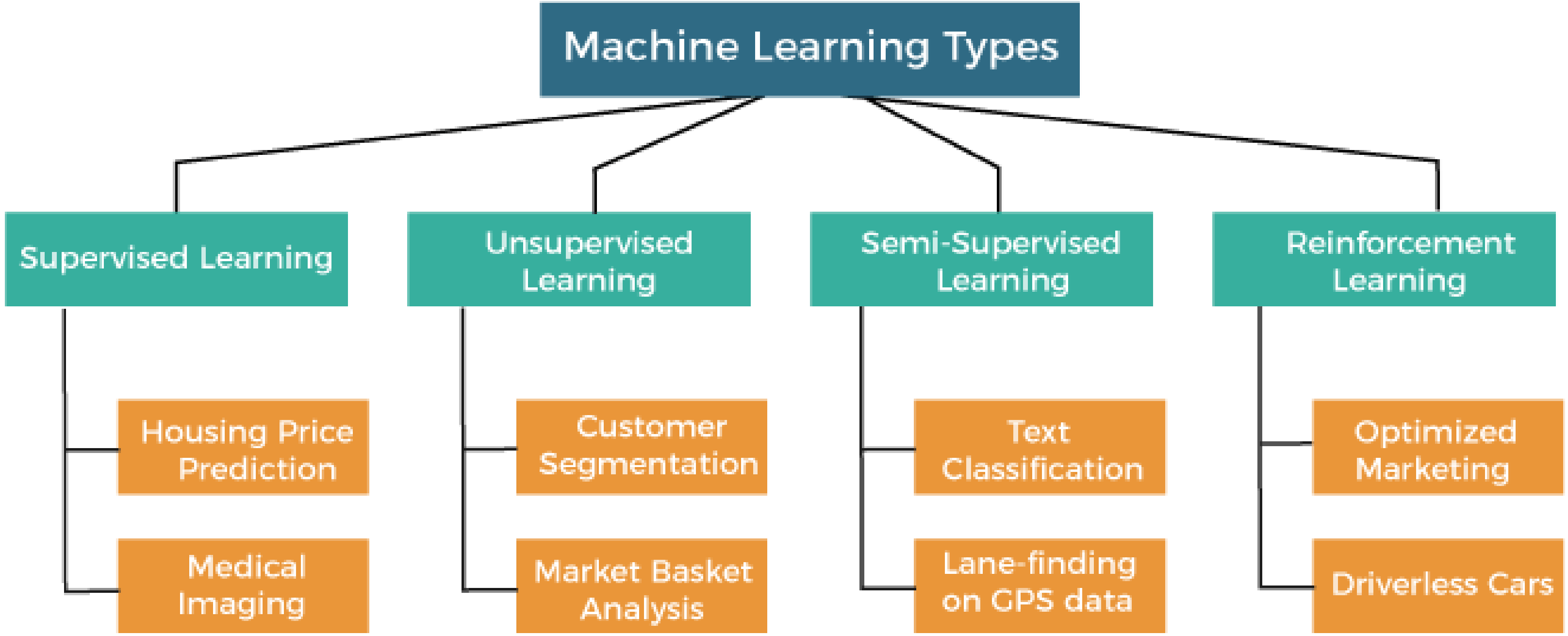
UNSUPERVISED LEARNING PART 1: K-MEANS

MACHINE LEARNING TYPES

- **Supervised**
  - Classification
  - Regression
- **Unsupervised**
  - Clustering
  - Dimensionality Reduction
- **Semi-supervised**
- **Reinforcement**



BYU

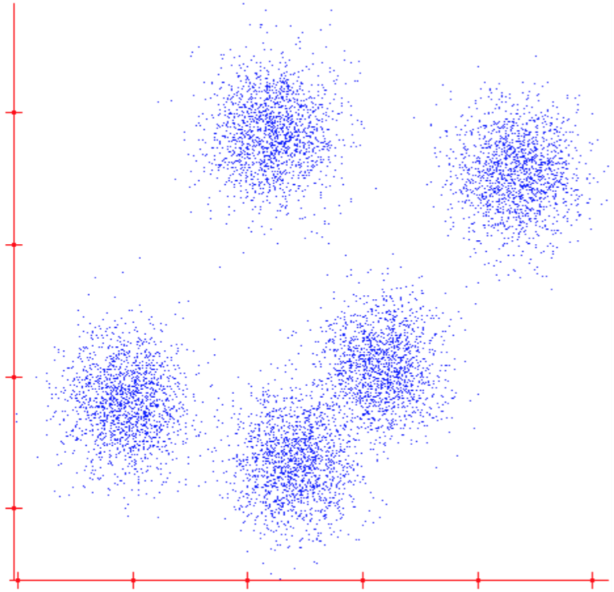
QUIZ

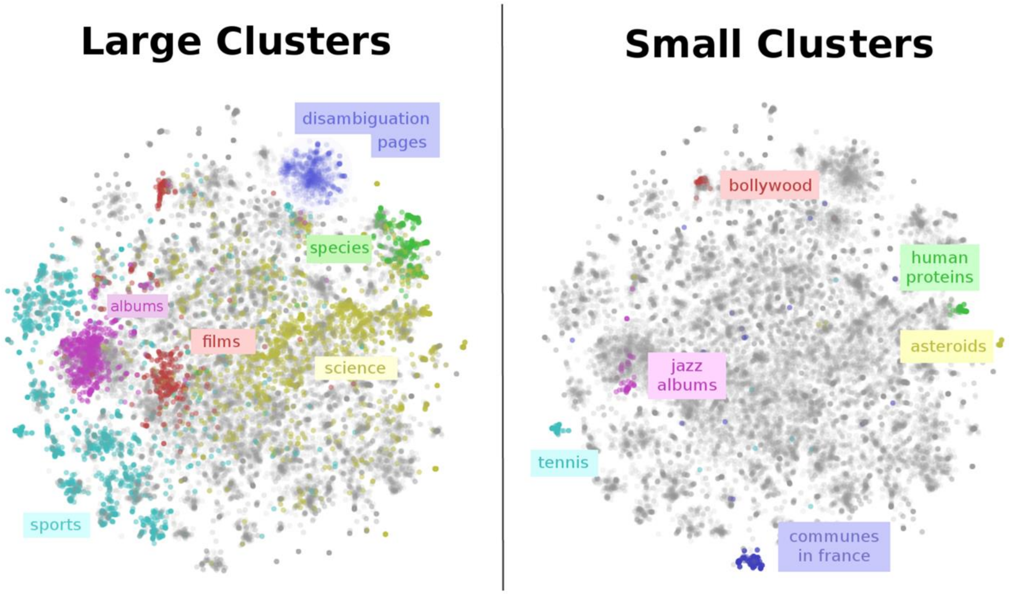
Of the following examples, which would you address using an <u>unsupervised</u> learning algorithm? (Check all that apply.)

- [ ] Given email labeled as spam/not spam, learn a spam filter.
- [x] Given a set of news articles found on the web, group them into set of articles about the same story.
- [x] Given a database of customer data, automatically discover market segments and group customers into different market segments.
- [ ] Given a dataset of patients diagnosed as either having diabetes or not, learn to classify new patients as having diabetes or not.

BYU

CLUSTERING DATA: GROUP SIMILAR THINGS





wikipedia articles

[Goldberger et al.]

APPLICATIONS OF CLUSTER ANALYSIS

- **Customer Segmentation**: In marketing, clustering helps to group customers based on behavior, preferences, or demographics.
  - This segmentation enables personalized marketing, targeted promotions, and better customer service.
- **Document Clustering**: often used in natural language processing to organize large text corpora.
  - It groups similar documents, which is useful for topic modeling, or summarizing news articles.
- **Anomaly Detection**: Can identify outliers in data.
  - Ex: in network security, clustering can detect unusual patterns of behavior that may indicate a cyberattack or fraud.

MORE APPLICATIONS OF CLUSTER ANALYSIS

- **Social Network Analysis**: Clustering helps to find communities within social networks
  - by grouping users with similar connections, interests, or behaviors.

- **Genomics**: clustering is used to classify genes with similar expression patterns,
  - helping in the discovery of gene functions and understanding of diseases.

- **Urban Planning and Geographic Analysis**: Helps in analyzing geographic data
  - Ex: identifying areas with similar land use, demographic profiles, or crime rates, which aids in urban planning.

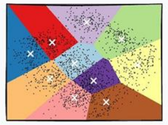
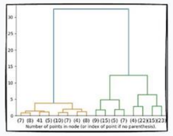
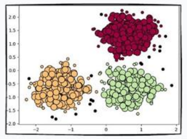
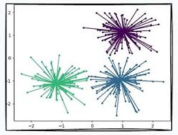
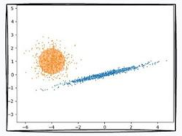
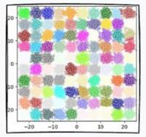
BYU

MAJOR TYPES OF CLUSTERING ALGORITHMS

Our focus for the next two classes

6 Types of Clustering Algorithms in Machine Learning

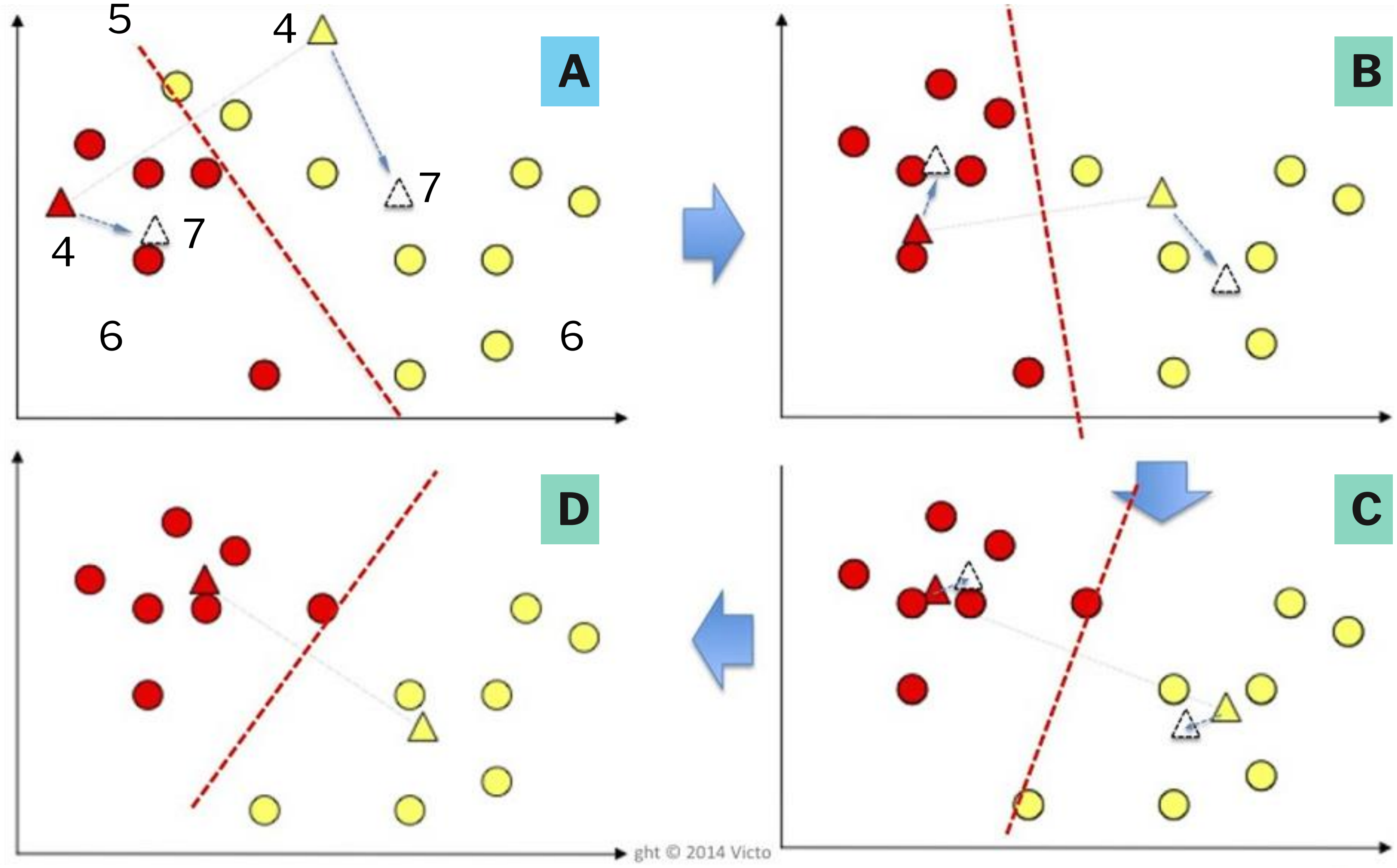
blog.DailyDoseofDS.com

Clustering Algorithm Type		Clustering Methodology	Algorithm(s)
	Centroid-based	Cluster points based on proximity to centroid	KMeans KMeans++ KMedoids
	Connectivity-based	Cluster points based on proximity between clusters	Hierarchical Clustering (Agglomerative and Divisive)
	Density-based	Cluster points based on their density instead of proximity	DBSCAN OPTICS HDBSCAN
	Graph-based	Cluster points based on graph distance	Affinitiy Propagation Spectral Clustering
	Distribution-based	Cluster points based on their likelihood of belonging to the same distribution.	Gaussian Mixture Models (GMMs)
	Compression-based	Transform data to a lower dimensional space and then perform clustering	BIRCH

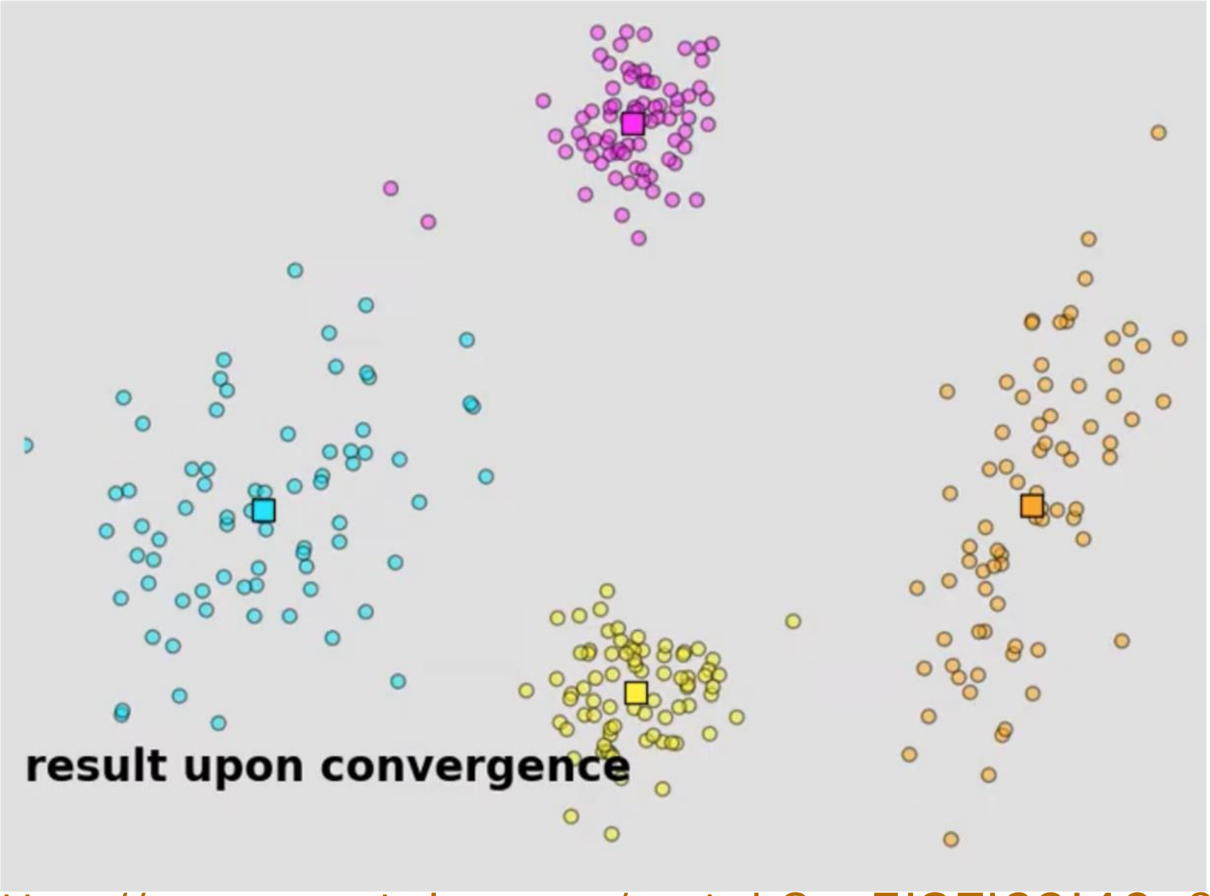
1. **Centroid-based:** Cluster data points based on proximity to centroids.
2. **Connectivity-based:** Cluster points based on proximity between clusters.
3. **Density-based:** Cluster points based on their density. It is more robust to clusters with varying densities and shapes than centroid-based clustering.
4. **Graph-based:** Cluster points based on graph distance.
5. **Distribution-based:** Cluster points based on their likelihood of belonging to the same distribution. Gaussian Mixture Model in one example.
6. **Compression-based:** Transform data to a lower dimensional space and then perform clustering

K-MEANS CLUSTERING

1. Standardize samples
2. Choose # clusters (K)
3. Choose # iterations
4. Randomly place initial centroids (Fig. A)
5. Calculate distance to centroid (Fig. A)
6. Assign samples with the nearest centroid color (Fig. A)
7. Move centroids to the mean of cluster (Fig. A)
8. Repeat 5-7 (Figs. B, C, D) until convergence



K-MEANS CLUSTERING ANIMATION



https://www.youtube.com/watch?v=5I3Ei69I40s&t=59s

BYU

K-MEANS DETAILS

- For a very good detailed explanation of how K-Means works under the hood:

  https://www.youtube.com/watch?v=IX-3nGHDhQg

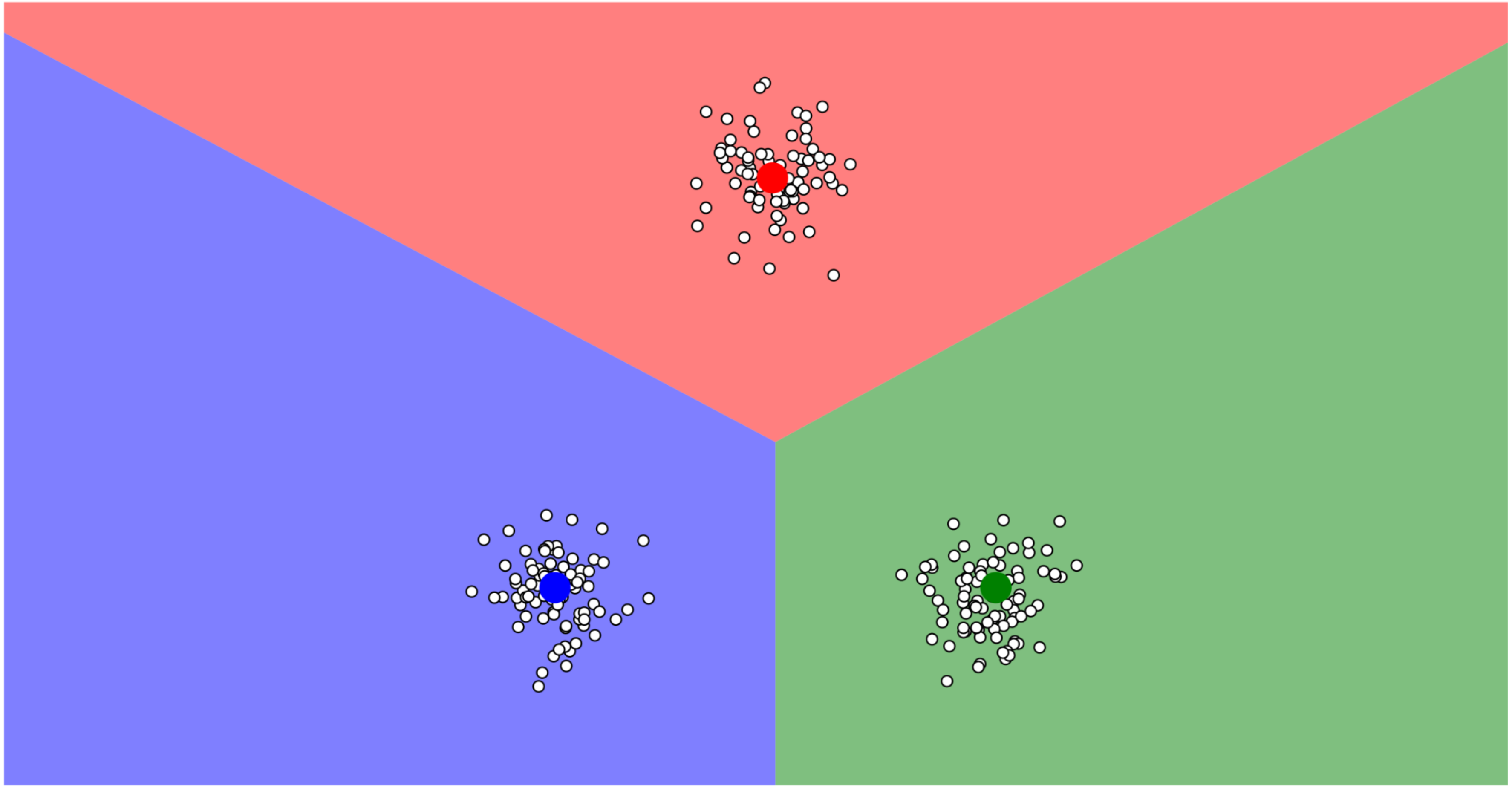
- Another very good video, if you can understand his accent, is:
  https://www.youtube.com/watch?v=iNIZ3IU5Ffw&t=300s

- What are some of the advantages/disadvantages of the K-Means algorithm for clustering? When does it work well? When does it fail?

BYU

Visualizing K-Means

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



EVALUATION METHOD: WITHIN CLUSTER SUM OF SQUARES (WCSS)

- The within-cluster sum of squares (WCSS) measures the compactness of a clustering result by calculating the sum of the squared distances between each data point and the centroid of the cluster it belongs to.
- Here's how to calculate the WCSS:

1. **Identify the clusters**: For each cluster, determine the data points assigned to it and calculate the centroid.
2. **Calculate the squared distance**: For each data point in a cluster, calculate the Euclidean distance to the cluster's centroid. Then, square this distance.
3. **Sum the squared distances**: Sum up the squared distances for all points in the cluster to get the WCSS for that cluster.
4. **Sum across all clusters**: If you have multiple clusters, repeat the above steps for each cluster and sum the results to get the overall WCSS.

```
import numpy as np

def calculate_wcss(data, labels, centroids):
    wcss = 0
    for i, centroid in enumerate(centroids):
        # Get points in cluster i
        cluster_points = data[labels == i]
        # Calculate the sum of squared distances to the centroid
        wcss += np.sum((cluster_points - centroid) ** 2)
    return wcss
```

In this example:

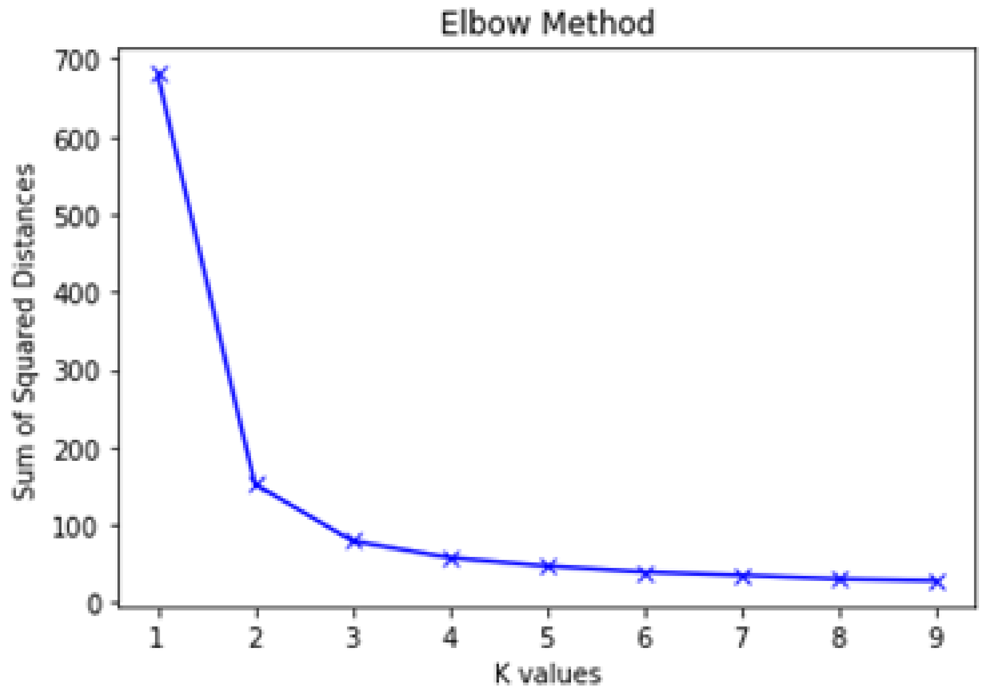
- `data` is your dataset as a NumPy array,
- `labels` are the cluster labels for each data point (indicating cluster assignment),
- `centroids` are the coordinates of each cluster centroid.

BYU

FINDING THE OPTIMAL K VALUE USING ELBOW METHOD

Steps:

- 1. For different values of K, execute the following steps:
- 2. For each cluster, calculate the sum of the squared distance of every point to its centroid.
- 3. Add the sum of squared distances of each cluster to get the total sum of squared distances for that value of K.
- 4. Keep adding the total sum of squared distances for each K to a list.
- 5. Plot the sum of squared distances (using the list created in the previous step) and their K values.
- 6. Select the K at which a sharp change occurs (looks like an elbow of the curve).



ELBOW PLOT CODE

1. Data Generation: Generate sample data with make_blobs for demonstration. In practice, you'd replace X with your dataset.

2. WCSS Calculation: Initialize an empty list wcss and use a loop to run KMeans with different values of k (from 1 to 10). The inertia_ attribute of the KMeans object gives the WCSS for the fitted clusters.

3. Plotting: The plot of WCSS vs. k shows how the WCSS decreases as we increase the number of clusters. The "elbow" in this plot indicates the optimal number of clusters, where increasing k further has diminishing returns on reducing WCSS.

4. Interpreting the Elbow Plot
The elbow point, where the WCSS curve starts to flatten, suggests the optimal number of clusters. Choosing a k beyond this point provides minimal additional clustering improvement.

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.datasets import make_blobs

# Generate sample data
X, _ = make_blobs(n_samples=300, centers=4, cluster_std=1.0, ran

# List to store WCSS values for each k
wcss = []

# Run k-means with k values from 1 to 10 and calculate WCSS
for k in range(1, 11):
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(X)
    wcss.append(kmeans.inertia_)  # inertia_ is the WCSS for the

# Plot the WCSS values for each k to find the "elbow" point
plt.figure(figsize=(8, 5))
plt.plot(range(1, 11), wcss, marker='o')
plt.title('Elbow Method for Optimal k')
plt.xlabel('Number of clusters (k)')
plt.ylabel('Within-Cluster Sum of Squares (WCSS)')
plt.show()
```

SILHOUETTE METHOD FOR EVALUATION

- The silhouette method is a technique for evaluating the quality of clustering, which can help determine the optimal number of clusters for a given dataset.
  - It quantifies how well each data point fits into its assigned cluster compared to other clusters.
  - **Silhouette Coefficient:** The silhouette coefficient is a measure of this fit, ranging from -1 to +1.
  - A higher score indicates better clustering.
  - The silhouette score combines the concepts of **cohesion** (how similar a point is to other points in its cluster) and **separation** (how dissimilar a point is to points in other clusters). [1]
- **Here's how the silhouette method works:**
  - Calculate the silhouette coefficient **for each data point.** The silhouette coefficient for a data point *i* is calculated as:
  - *(b(i) - a(i)) / max(a(i), b(i))*
  - where:
    - *a(i)* is the average distance between point *i* and all other points in the same cluster.
    - *b(i)* is the average distance between point *i* and all points in the nearest neighboring cluster.
  - **Calculate the average silhouette coefficient for all data points.** This average score represents the overall quality of the clustering.
  - **Repeat steps 1 and 2 for different numbers of clusters.** By comparing the average silhouette coefficients for different *k* values, you can identify the number of clusters that produce the highest score, indicating the most optimal clustering structure.

BYU

SILHOUETTE METHOD EXAMPLE

Suppose you have a dataset and want to determine the optimal number of clusters using the silhouette method. You apply a clustering algorithm, such as K-Means, with different k values (e.g., $k = 2, 3, 4, 5$). For each k, you calculate the silhouette coefficient for each data point and then average those coefficients. Let's say you obtain the following average silhouette scores:

- $k = 2$: 0.45
- $k = 3$: 0.62
- $k = 4$: 0.55
- $k = 5$: 0.48

Observing the silhouette scores, the highest score occurs at $k = 3$. This suggests that 3 is the most suitable number of clusters for this dataset.

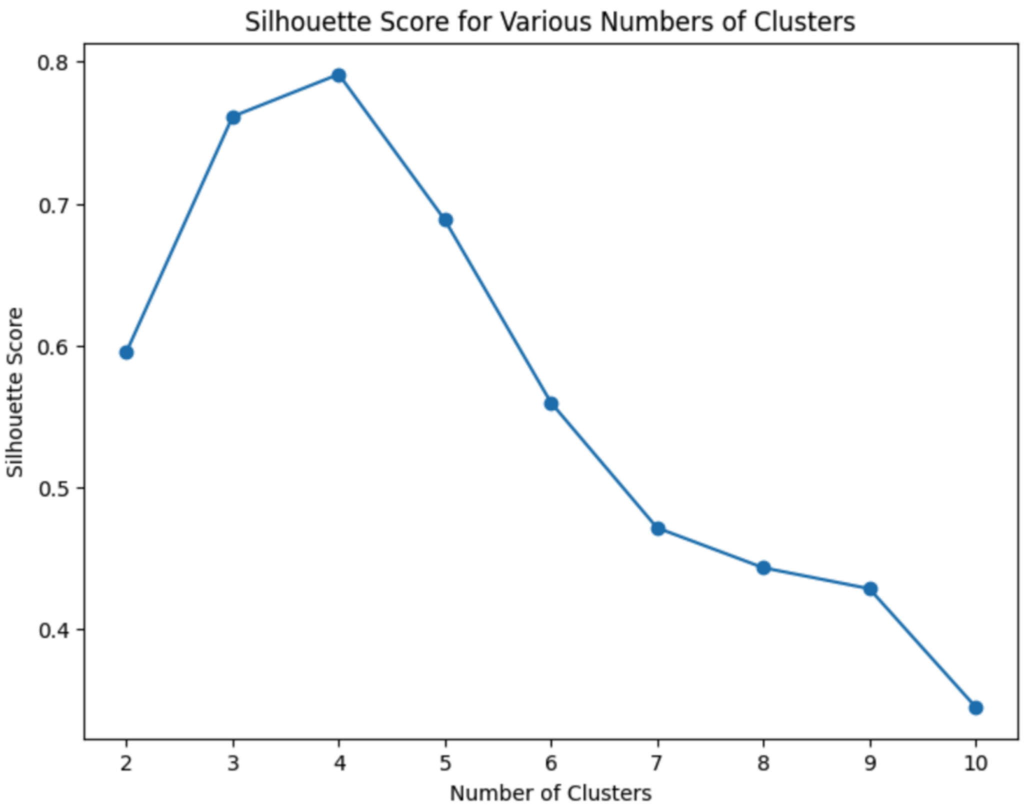
- **Advantages:**
  - The silhouette method quantifies clustering quality, offering greater objectivity than visual methods like the elbow method.
  - It considers both cohesion and separation, giving a more comprehensive evaluation of cluster structure.
- **Limitations:**
  - The calculation of silhouette coefficients **can be computationally expensive**, particularly for large datasets.
  - As with any clustering evaluation metric, the silhouette method is not a foolproof solution and should be used in conjunction with other methods and domain knowledge.

BYU

SILHOUETTE METHOD PYTHON CODE

```
from sklearn.datasets import make_blobs
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
import matplotlib.pyplot as plt

# Generate synthetic dataset with 4 centers
X, _ = make_blobs(n_samples=500, centers=4, cluster_std=1.0, random_state=42)

# List to store silhouette scores
silhouette_scores = []

# Range of clusters to test
range_n_clusters = range(2, 11)

# Compute silhouette scores for each number of clusters
for n_clusters in range_n_clusters:
    kmeans = KMeans(n_clusters=n_clusters, random_state=42)
    cluster_labels = kmeans.fit_predict(X)
    silhouette_avg = silhouette_score(X, cluster_labels)
    silhouette_scores.append(silhouette_avg)
    print(f"For n_clusters = {n_clusters}, the average silhouette score is: {silhouette_avg:.3f}")

# Plot silhouette scores
plt.figure(figsize=(8, 6))
plt.plot(range_n_clusters, silhouette_scores, marker='o')
plt.xlabel("Number of Clusters")
plt.ylabel("Silhouette Score")
plt.title("Silhouette Score for Various Numbers of Clusters")
plt.show()
```

Screenshot



BYU

WSCC VS. SILHOUETTE

When to Use Each?

Metric	Measures	Best For	Limitations
WCSS	Intra-cluster compactness	Elbow Method, K-Means	Does not consider inter-cluster separation
Silhouette Score	Compactness + Separation	Comparing cluster quality across different algorithms	Computationally expensive

BYU