Python Programming and Basic Data Science

Shuvro Pal
Data Scientist, Markopolo.ai
AI Team Lead, Zantrik
ML Facilitator, Crowdsource by Google

Clustering

What is it?

- Clustering is a technique for finding similarity groups in data, called clusters.

- A good clustering method will produce clusters with

  –Inter-clusters distance ⟶ maximized

  –Intra-clusters distance ⟶ minimized

k-means

What is it?

- K-means clustering is an unsupervised learning algorithm used to group similar data points into clusters. It tries to partition the dataset into K clusters, where each data point belongs to the cluster with the nearest centroid.
- **Key idea:** The k-means algorithm aims to group data points that are similar to each other while ensuring that each group (cluster) is as distinct as possible from the others.

Example: Imagine you have a group of fruits with different sizes and colors. K-means clustering helps group similar fruits together based on these features (e.g., grouping apples, oranges, and bananas separately).

k-means

Use cases?

- Customer segmentation in marketing
- Image compression
- Document clustering and topic detection
- Grouping similar products or services

k-means

Real World Examples?

Problem: Images are made up of thousands of pixels, each with an RGB value. Storing or transmitting high-resolution images can be inefficient due to their size.

K-means Solution:

- K-means clustering is used to reduce the number of colors in the image. Each pixel is assigned to the nearest cluster (representing a color).
- Instead of storing the exact RGB value for every pixel, we store only the cluster ID and the centroid (color) for that cluster, drastically reducing the image size.

1. **Before K-means:** A high-resolution image might have millions of unique colors (pixel values).
2. **After K-means:** K-means reduces the number of colors to a smaller set of K colors (e.g., K=16). The resulting image looks similar but takes up far less storage space.

Impact: K-means allows for efficient image compression without significant loss in visual quality. This is used in JPEG compression and web image optimization.

k-means

How it actually works?

Step 1: Choose the number of clusters (K). Decide how many clusters you want to divide your data into (this is a key input to the algorithm).

Step 2: Initialize centroids. Randomly assign K points as the starting centroids.

Step 3: Assign data points to clusters. Each data point is assigned to the nearest centroid.

Step 4: Update centroids. Calculate the new centroids by averaging the data points in each cluster.

Step 5: Repeat steps 3 and 4 until the centroids no longer move significantly (convergence).

k-means

Some math please?

Use the k-means algorithm and Euclidean distance to cluster the following 8 examples into 3(k=3) clusters:

A1=(2,10),

A2=(2,5),

A3=(8,4),

A4=(5,8),

A5=(7,5),

A6=(6,4),

A7=(1,2),

A8=(4,9)

k-means

Solving the math..

Step 1: Suppose that the initial seeds (centers of each cluster) are A1, A4 and A7.

i.e. **seed1=A1=(2,10), seed2=A4=(5,8), seed3=A7=(1,2)**

Step 2: Find the Euclidian distance between each seed and point.

d(a,b)=sqrt((xb-xa)2+(yb-ya)2))

k-means

Solving the math..

A1:

d(A1, seed1)=0 as A1 is seed1

d(A1, seed2)= $\sqrt{13}$ >0

d(A1, seed3)= $\sqrt{65}$ >0

➔ A1 $\in$ cluster1

A2:

d(A2,seed1)= $\sqrt{25}$ = 5

d(A2, seed2)= $\sqrt{18}$ = 4.24

d(A2, seed3)= $\sqrt{10}$ = 3.16 ← smaller

➔ A2 $\in$ cluster3

A3:

d(A3, seed1)= $\sqrt{36}$ = 6

d(A3, seed2)= $\sqrt{25}$ = 5 ← smaller

d(A3, seed3)= $\sqrt{53}$ = 7.28

➔ A3 $\in$ cluster2

A4:

d(A4, seed1)= $\sqrt{13}$

d(A4, seed2)=0 as A4 is seed2

d(A4, seed3)= $\sqrt{52}$ >0

➔ A4 $\in$ cluster2

A5:

d(A5, seed1)= $\sqrt{50}$ = 7.07

A6:

d(A6, seed1)= $\sqrt{52}$ = 7.21

k-means

Solving the math..

d(A5, seed2)= $\sqrt{13}$ = 3.60 ← smaller

d(A5, seed3)= $\sqrt{45}$ = 6.70

→ A5 ∈ cluster2

d(A6, seed2)= $\sqrt{17}$ = 4.12 ← smaller

d(A6, seed3)= $\sqrt{29}$ = 5.38

→ A6 ∈ cluster2

A7:

d(A7, seed1)= $\sqrt{65}$ >0

d(A7, seed2)= $\sqrt{52}$ >0

d(A7, seed3)=0 as A7 is seed3

→ A7 ∈ cluster3

A8:

d(A8, seed1)= $\sqrt{5}$

d(A8, seed2)= $\sqrt{2}$ ← smaller

d(A8, seed3)= $\sqrt{58}$

→ A8 ∈ cluster2

end of epoch1

new clusters: 1: {A1}, 2: {A3, A4, A5, A6, A8}, 3: {A2, A7}

k-means

Solving the math..

Step 3: Find the centers of the new clusters: C1= (2, 10),

C2= ((8+5+7+6+4)/5, (4+8+5+4+9)/5) = (6, 6),

C3= ((2+1)/2, (5+2)/2) = (1.5, 3.5)

Step 4: After the 2nd epoch the results would be:

1: {A1, A8}, 2: {A3, A4, A5, A6}, 3: {A2, A7}

with centers C1=(3, 9.5), C2=(6.5, 5.25) and C3=(1.5, 3.5).

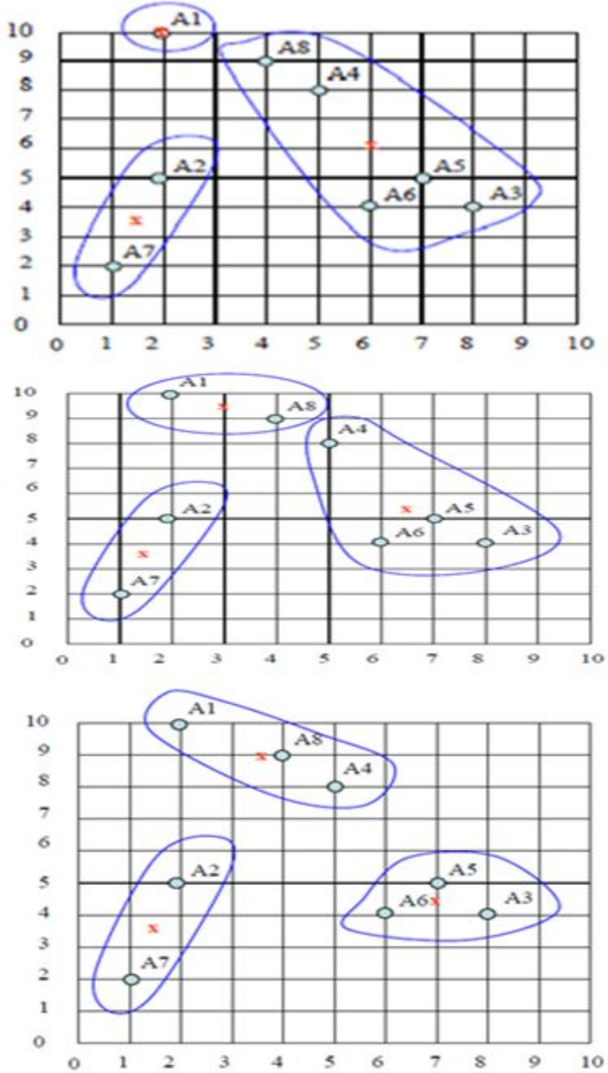
After the 3rd epoch, the results would be:

1: {A1, A4, A8}, 2: {A3, A5, A6}, 3: {A2, A7}

with centers C1=(3.66, 9), C2=(7, 4.33) and C3=(1.5, 3.5).

k-means

Solving the math..



k-means

Why it's worthy?

- **Simple and Fast:** K-means is easy to understand and implement. It's also computationally efficient for large datasets.
- **Scalable:** K-means works well even when the dataset grows large in size or number of features.
- **Versatile:** K-means is widely applicable to different types of data (e.g., customer segmentation, image compression, text analysis).

k-means

Challenges?

Choosing K:

- Choosing the right number of clusters (K) can be tricky. A common technique is the Elbow Method, where you plot the sum of squared errors (SSE) for different values of K and look for the "elbow point," where increasing K no longer significantly improves the fit.

```
| SSE
|  \          Elbow Point
|    \        /
|      \____/
|______________________
       K
```

Cluster Initialization:

- K-means is sensitive to the initial position of centroids. A bad initialization can lead to poor clustering. To solve this, we use K-means++ initialization, which spreads out the initial centroids more intelligently.

Future of AI


সবার জন্য কৃত্তিম বুদ্ধিমত্তা
Java
আপনার সন্তানকে
মেশিন লার্নিং শিক্ষা দিন
SUFIAN SIDDIQUI