Advanced Algorithms

How many McDonald stores are there on the Hong Kong island?

According to GPT 5.4, there are 45 McDonald stores on the Hong Kong island.

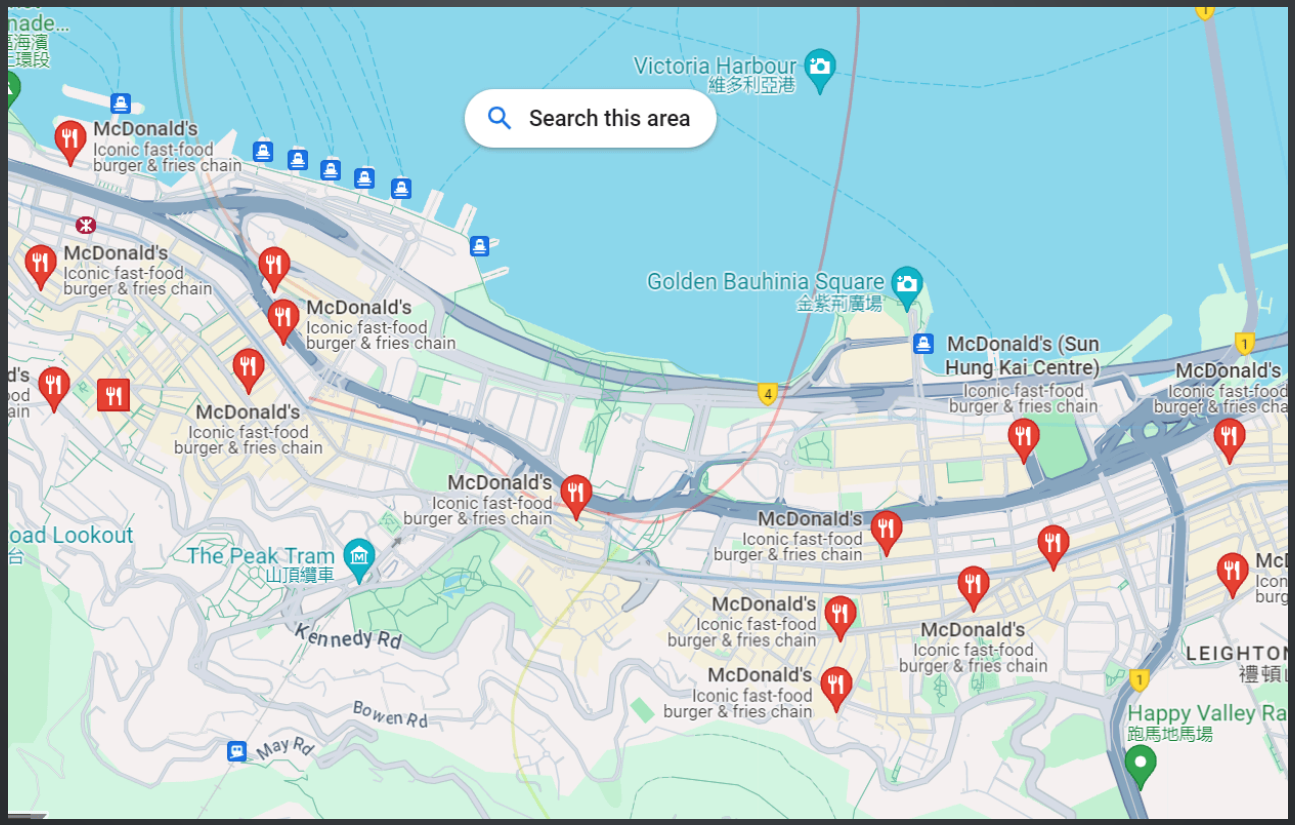
How many McDonald stores are there on Hong Kong island

3 steps completed >

There are **31 McDonald's restaurants on Hong Kong Island** based on the current address list on McDonald's Hong Kong site, counting the branches listed under Central/Western, Wanchai, Eastern, and Southern districts: 9 + 9 + 9 + 4 = 31. `mcdonalds`

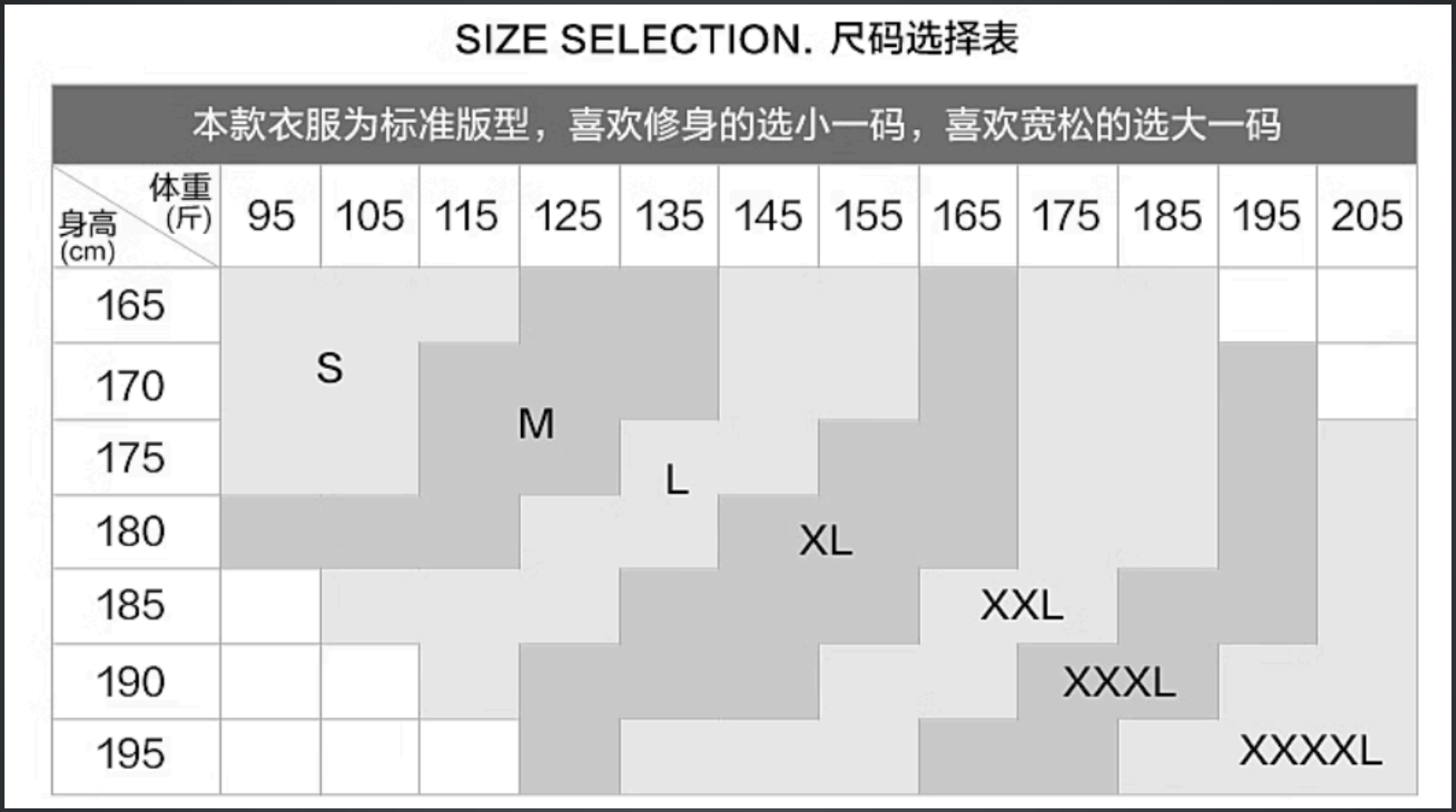
If you were the CEO of McDonald Hong Kong, how would you choose the locations of these stores?

Why is there only one McDonald close to Admiralty station?



Clustering

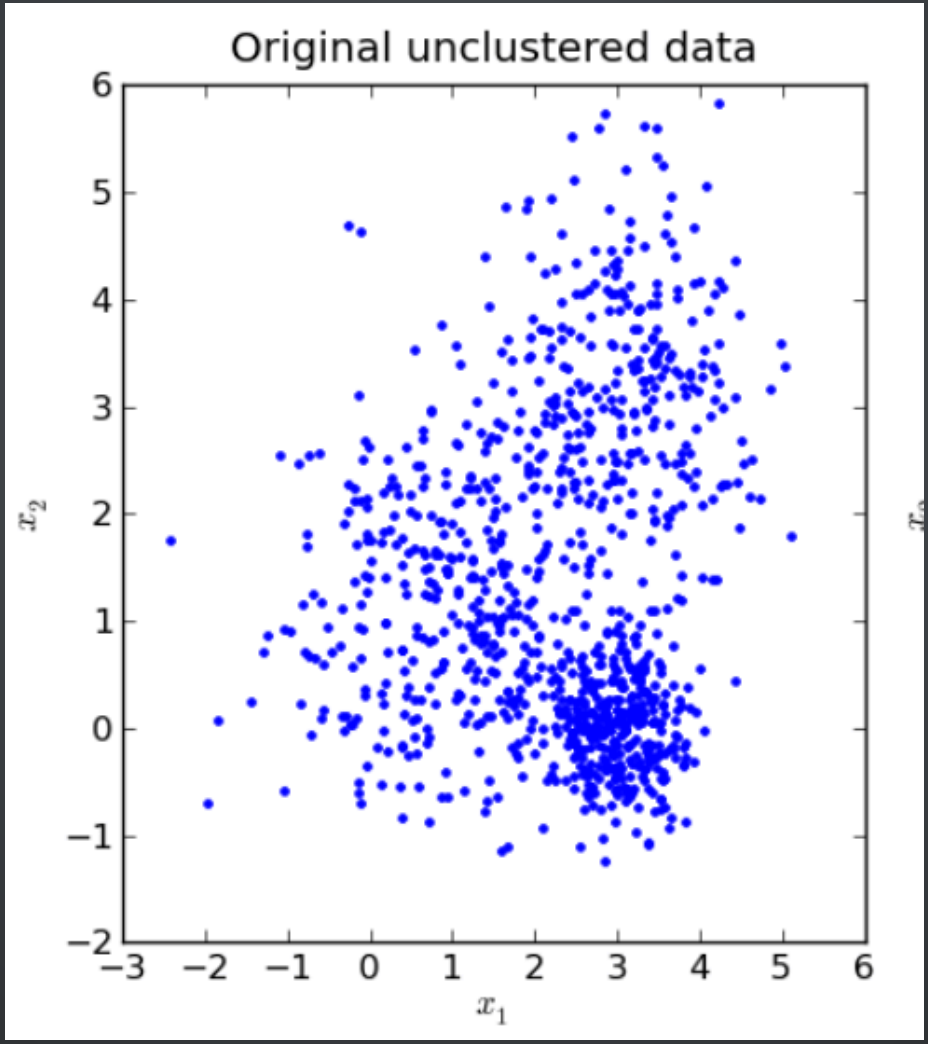
Imaging that you are designing T-shirts for consumers. You are looking at your consumers' weight and height, which would allow you to decide how many sizes to offer and which size fits a particular individual. Instead of classifying your consumers arbitrarily, you can use data to perform the task more precisely.

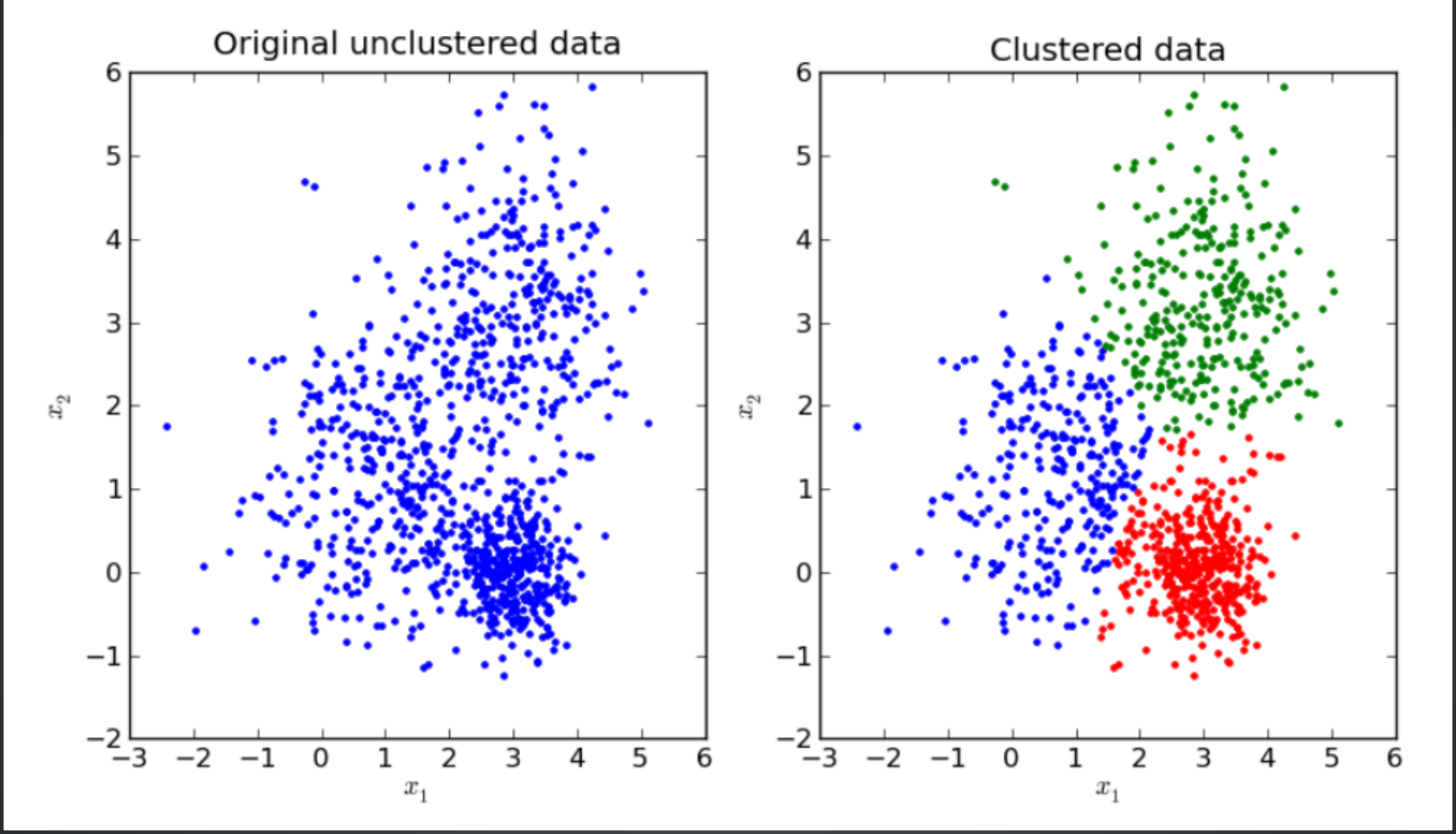


Why offer 7 sizes? Are these sizes optimal?

Image credit: taobao.com

Here is consumer data. How would you classify them into groups?




Original unclustered data
Clustered data
x_1
x_2

The K-means Algorithm

The K-means algorithm an EM (expectation-maximization) algorithm commonly used for classifying objects.

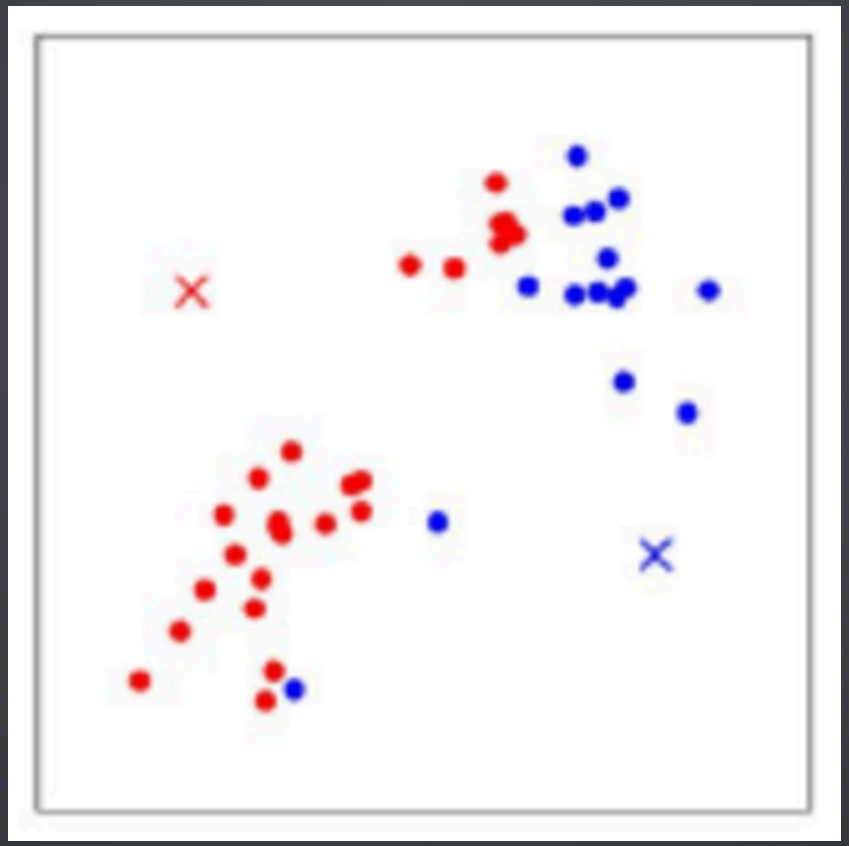
Input: A number of observations $(X_1, X_2, \ldots, X_n)$ and k, the number of groups to be classified

Output: k mutually exclusive and collectively exhaustive groups containing all observations

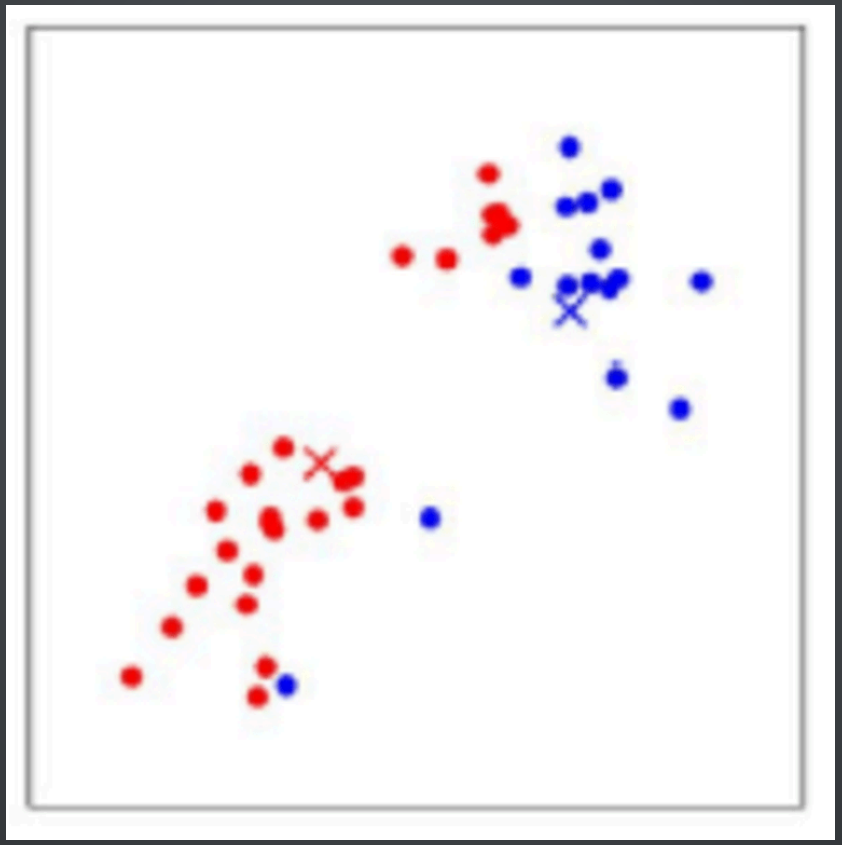
Classify the following observations into $k = 2$ groups:

Step 1: Random choose $k = 2$ "centers" for your clusters.

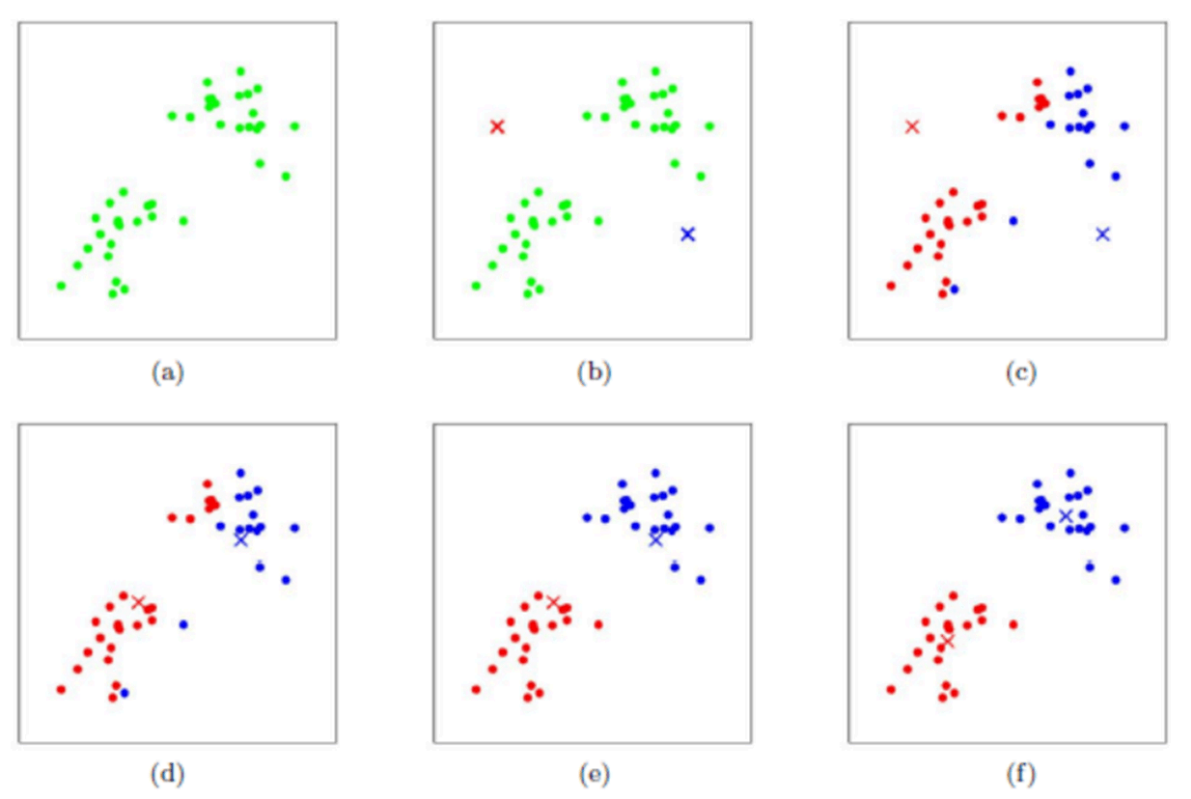
Step 2: Assign each observation to the nearest center.



Step 3: Update the location the centers, which is given by the average location of all points in the corresponding cluster.



Repeat the above process again and again until the centers no longer change.



The K-means algorithm:

1. Select cluster centers $c_1, \ldots, c_k \in \mathbb{R}^d$ arbitrarily.
2. Assign every $x \in \mathcal{X}$ to the cluster $\mathcal{C}_i$ whose cluster center c_i is closest to it, i.e., $\|x - c_i\| \leq \|x - c_j\|$ for all $j \neq i$.
3. Set $c_i = \frac{1}{|\mathcal{C}_i|} \sum_{x \in \mathcal{C}_i} x$.
4. If clusters or centers have changed, goto 2. Otherwise, terminate.

https://www.youtube.com/embed/R2e3Ls9H_fc?enablejsapi=1

Performing K-means with Python:

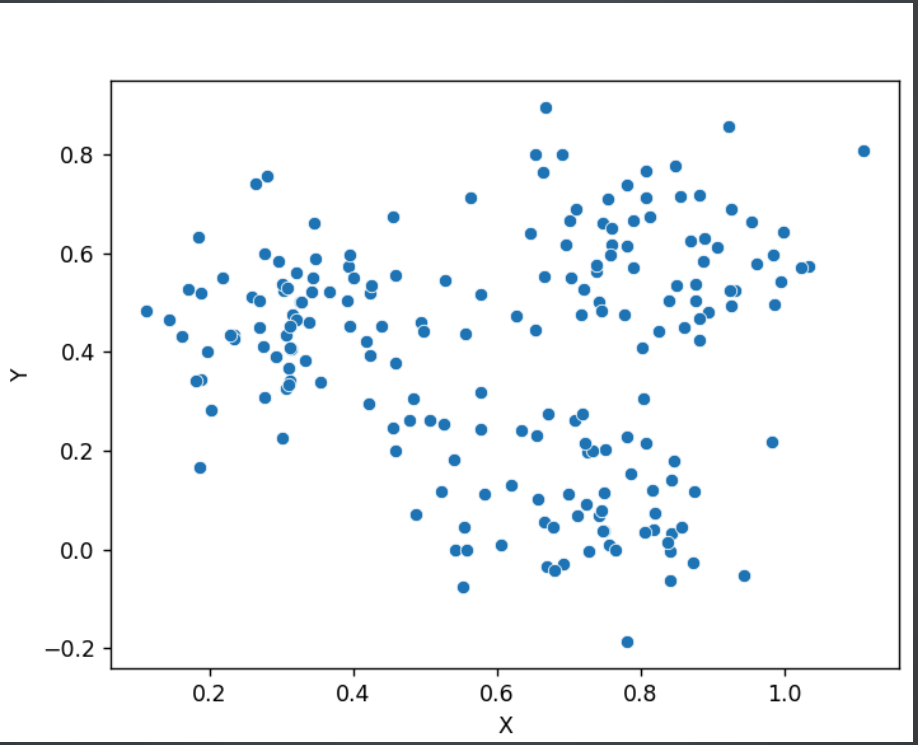
```
1 import pandas as pd
2 import matplotlib.pyplot as plt
3 import seaborn as sns
4 from sklearn.cluster import KMeans
5 from sklearn.metrics import silhouette_score
6 data = pd.read_csv('https://ximarketing.github.io/data/clustering.csv')
7 print(data)
```

	X	Y
0	0.627123	0.473972
1	0.562636	0.714202
2	0.295934	0.583077
3	0.815801	0.121524
4	0.345354	0.662115

The input is a CSV file containing the X and Y coordinates of some individuals.

```
1 sns.scatterplot(data = data, x = 'X', y = 'Y')
2 plt.show()
```

Let's visualize these locations!



How many groups do they belong to?

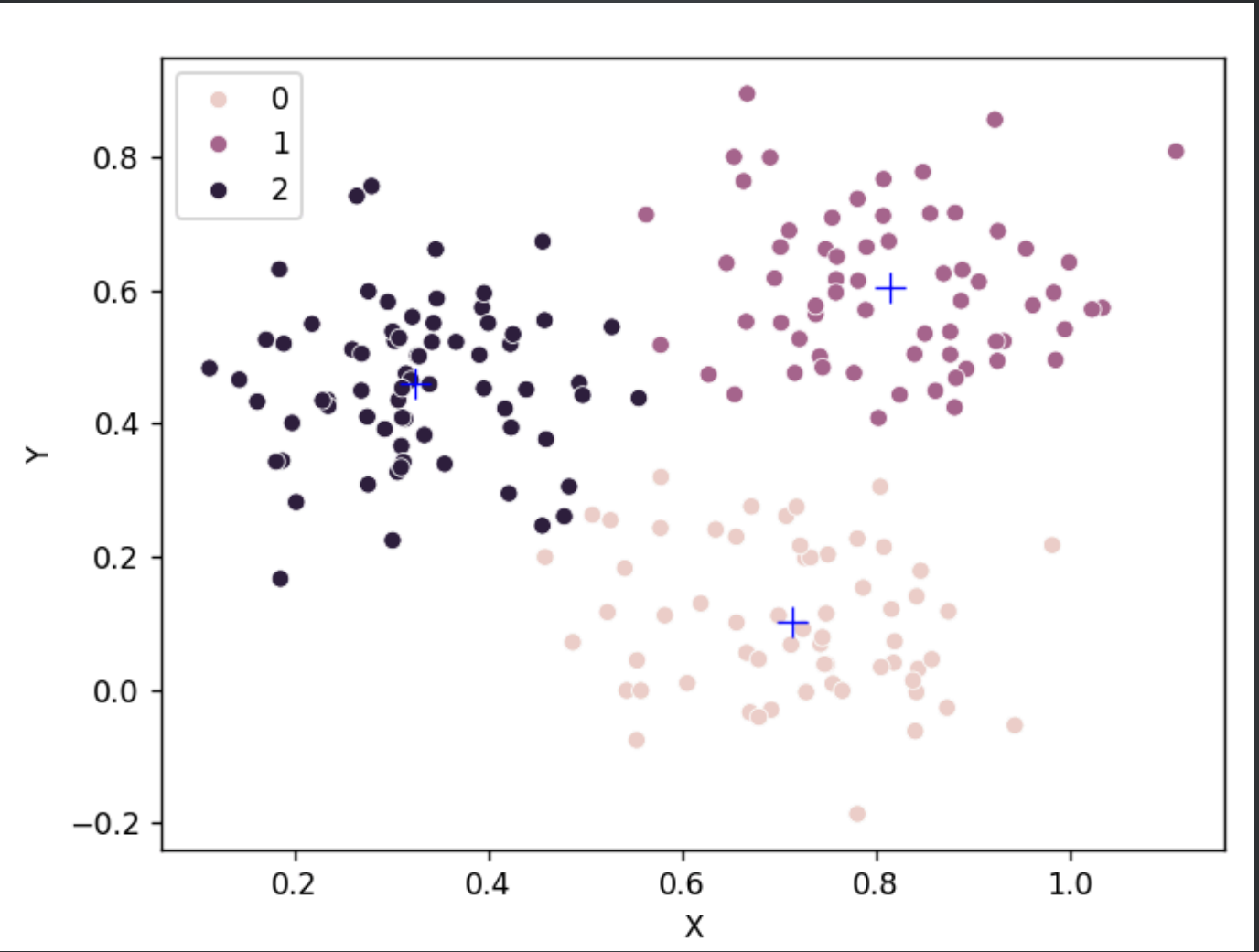
```
kmeans = KMeans(n_clusters = 3, random_state = 0, n_init='auto')
kmeans.fit(data)
sns.scatterplot(x = data['X'], y = data['Y'], hue = kmeans.labels_)
centers = kmeans.cluster_centers_
sns.scatterplot(x = centers[:, 0], y = centers[:, 1],
                color = 'blue', s=100, marker='+')
plt.show()
```

Now, we use the KMeans function to classify the observations. If you don't understand the code, ask AI:

What is the meaning of random state in python's kmeans?

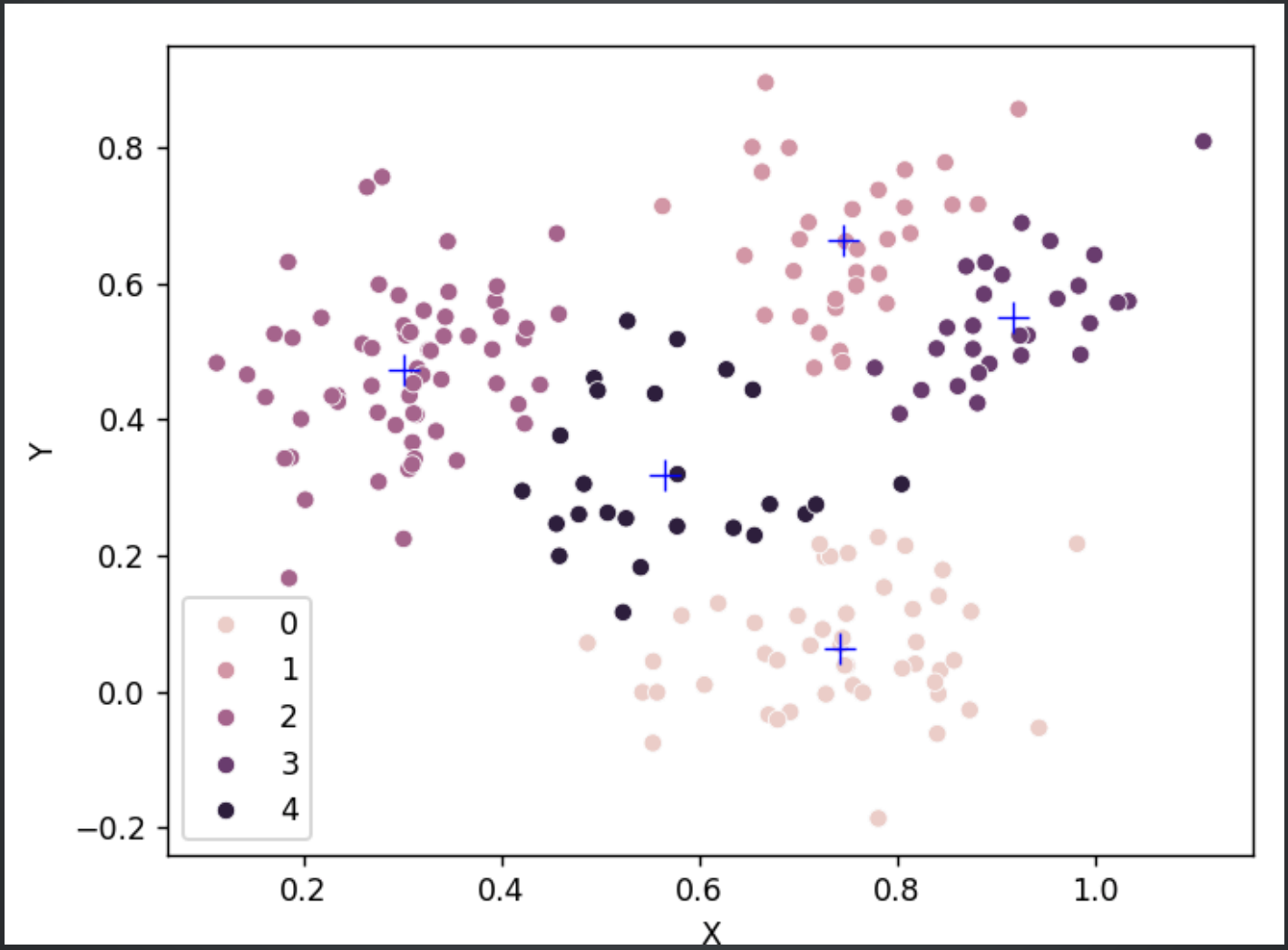
Reviewed 15 sources >

In Python's scikit-learn `KMeans`, `random_state` controls the random numbers used when choosing the initial cluster centers. If you set it to a fixed integer like `0` or `42`, you make the initialization deterministic, so the same data and settings will usually give the same result each time you run the code. `scikit-learn +1`



Does it look good?

Let's try five groups instead...



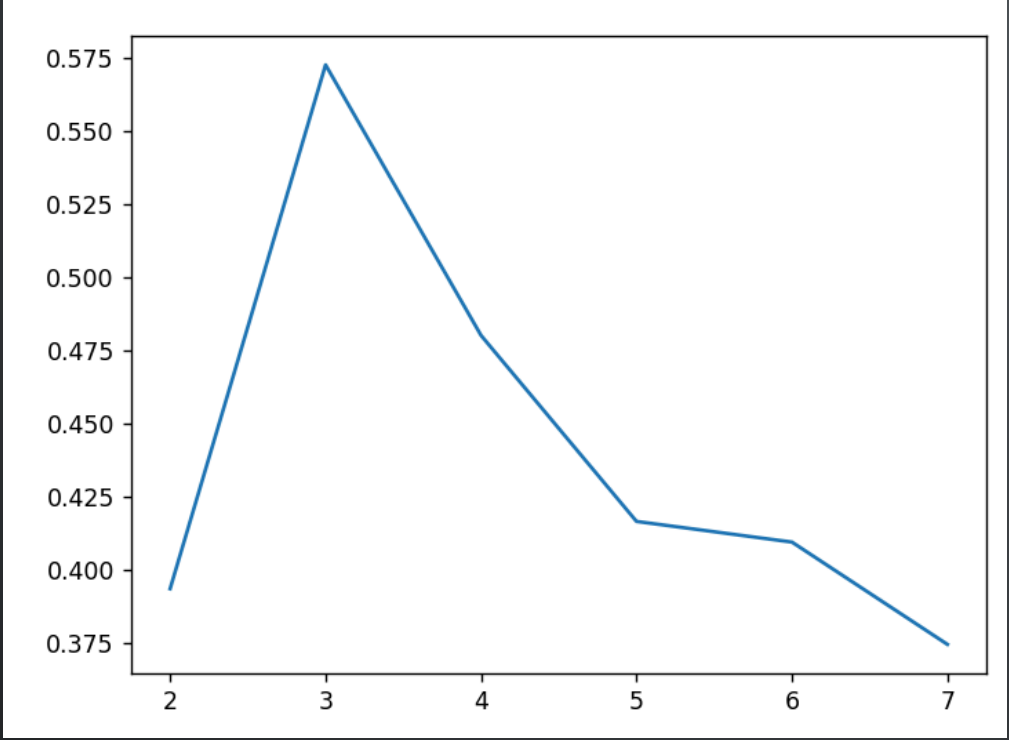
Exercise

Use AI to generate a desktop APP which demonstrates the K-means algorithm

Question:

What is the optimal number of clusters?

```
K = range(2, 8)
fits = []
score = []
for k in K:
    model = KMeans(n_clusters=k, random_state=0, n_init='auto').fit(data)
    fits.append(model)
    score.append(silhouette_score(data, model.labels_, metric='euclidean'))
sns.lineplot(x = K, y = score)
plt.show()
```



3 is the optimal number!

The complete code is here:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
data = pd.read_csv('https://ximarketing.github.io/data/clustering.csv')
print(data)
sns.scatterplot(data = data, x = 'X', y = 'Y')
plt.show()
kmeans = KMeans(n_clusters = 5, random_state = 0, n_init='auto')
kmeans.fit(data)
sns.scatterplot(x = data['X'], y = data['Y'], hue = kmeans.labels_)
centers = kmeans.cluster_centers_
sns.scatterplot(x = centers[:, 0], y = centers[:, 1],
                color = 'blue', s=100, marker='+')
plt.show()
K = range(2, 8)
fits = []
score = []
for k in K:
    model = KMeans(n_clusters=k, random_state=0, n_init='auto').fit(data)
    fits.append(model)
    score.append(silhouette_score(data, model.labels_, metric='euclidean'))
sns.lineplot(x = K, y = score)
plt.show()
```

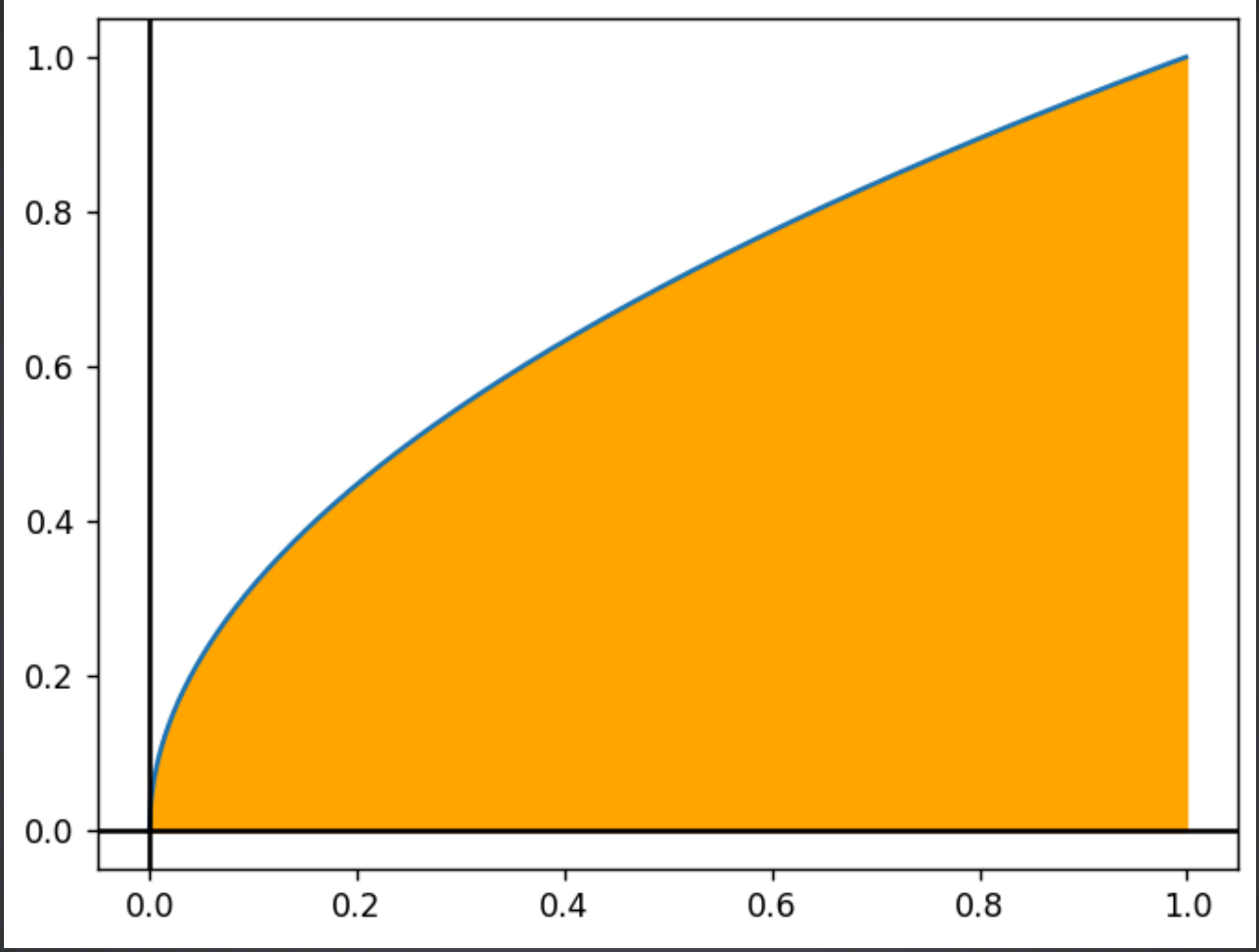
The k-means algorithm can also be used for image compression. How could this be done?

How would you compress this image?

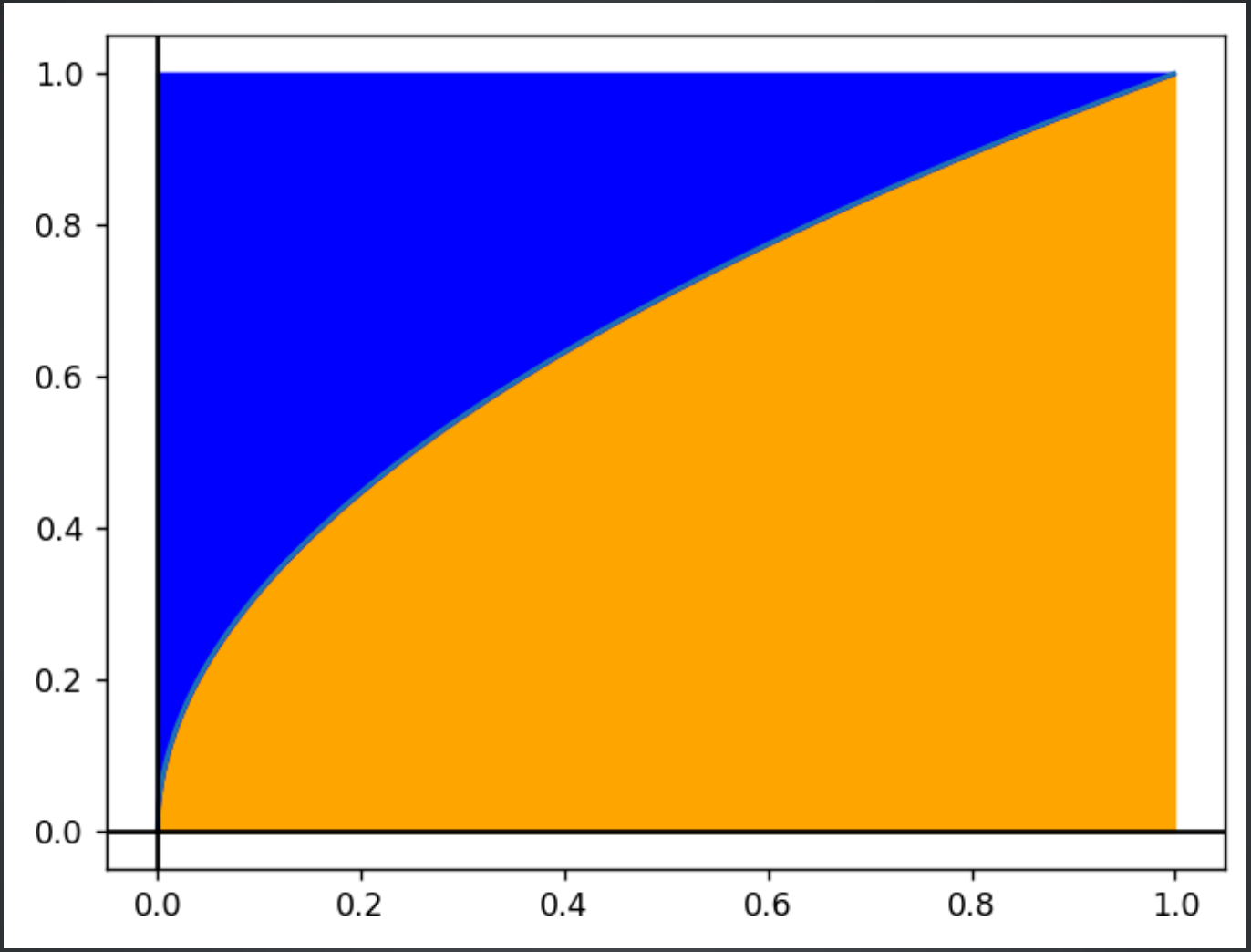
Calculate the following integral:

$$\int_0^1 \sqrt{x} dx$$

Monte Carlo Algorithm

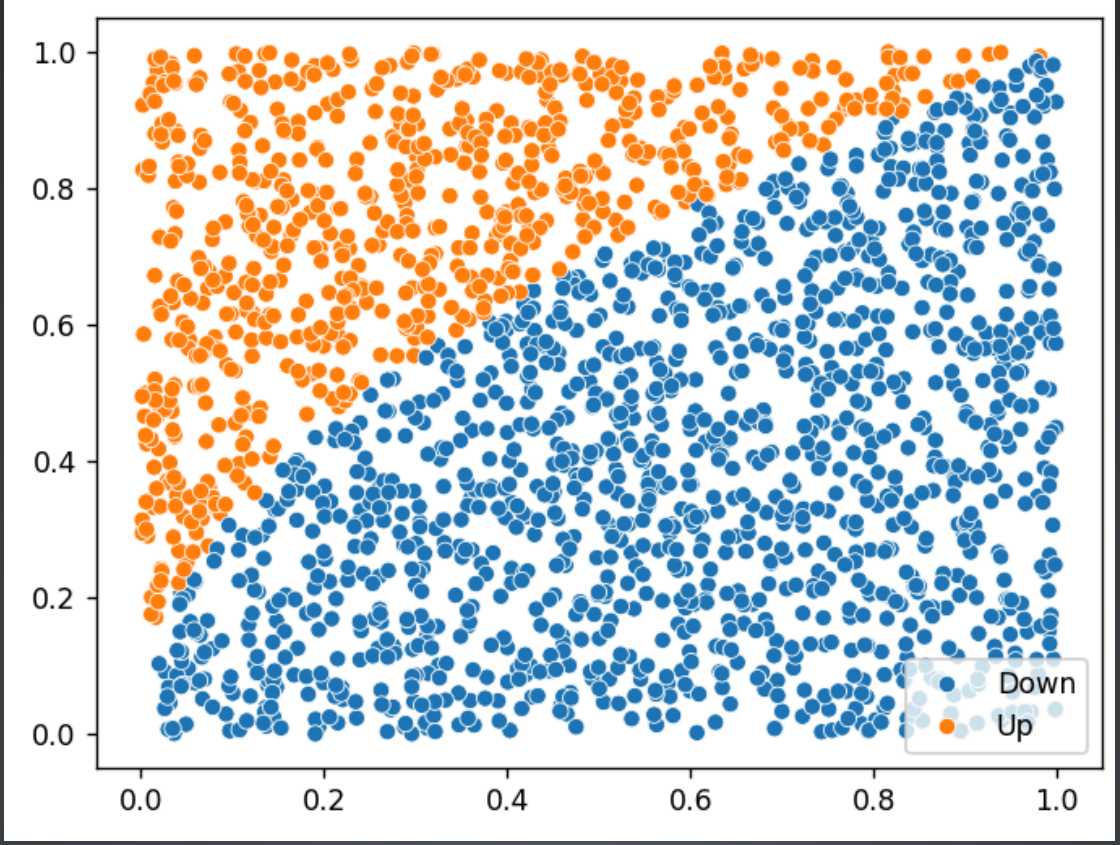


Integral is the area below the line $f(x) = \sqrt{x}$



The area size of blue + orange = 1

We want to calculate the fraction of the orange area



The idea is as follows. We random draw a large number of points in the area $[0, 1] \times [0, 1]$, and count how many points are above $\sqrt{x}$. The size can be calculated as follows:

$$\text{size of blue area} \approx \frac{\text{number of blue points}}{\text{total number of points}} \times 1$$

Algorithm:

1. Randomly choose K points that are uniformly distributed over $[0,1] \times [0,1]$. Denote these points by $(x_1, x_2), \ldots, (x_K, y_K)$.
2. Calculate how many points satisfy $y_i < \sqrt{x_i}$, denote by count_below.
3. The size of the area is count_below / K.

The complete code is here:

```
 1 import random
 2 import math
 3 K = 10000
 4 count_below = 0
 5 for i in range(0, K):
 6     x = random.uniform(0, 1)
 7     y = random.uniform(0, 1)
 8     if y <= math.sqrt(x):
 9         count_below = count_below + 1
10 print (count_below/K)
```

Exercise: Calculate the value of π.

Exercise: Calculate the value of π.

In 1733, the French mathematician Buffon first came up with a probabilistic method to calculate π numerically. He threw needles to parallel lines, and found that the probability that a needle crosses a line is a function of π. Based on that, he calculated the value of π.

In Chinese, this is known as “布丰投针”.

https://www.youtube.com/embed/kazgQXaeOHk?enablejsapi=1

The code for calculating π:

```
1 import random
2 K = 100000
3 count_inside = 0
4 for i in range(0, K):
5     x = random.uniform(-1, 1)
6     y = random.uniform(-1, 1)
7     if x**2+y**2<1:
8         count_inside = count_inside + 1
9 print(count_inside/K*4)
```

Exercise

Use AI to generate a desktop APP which demonstrates how to calculate π numerically.

Dynamic Programming

Question

You are climbing a staircase. It takes 10 steps to reach the top.

Each time you can either climb 1 or 2 steps. In how many distinct ways can you climb to the top?

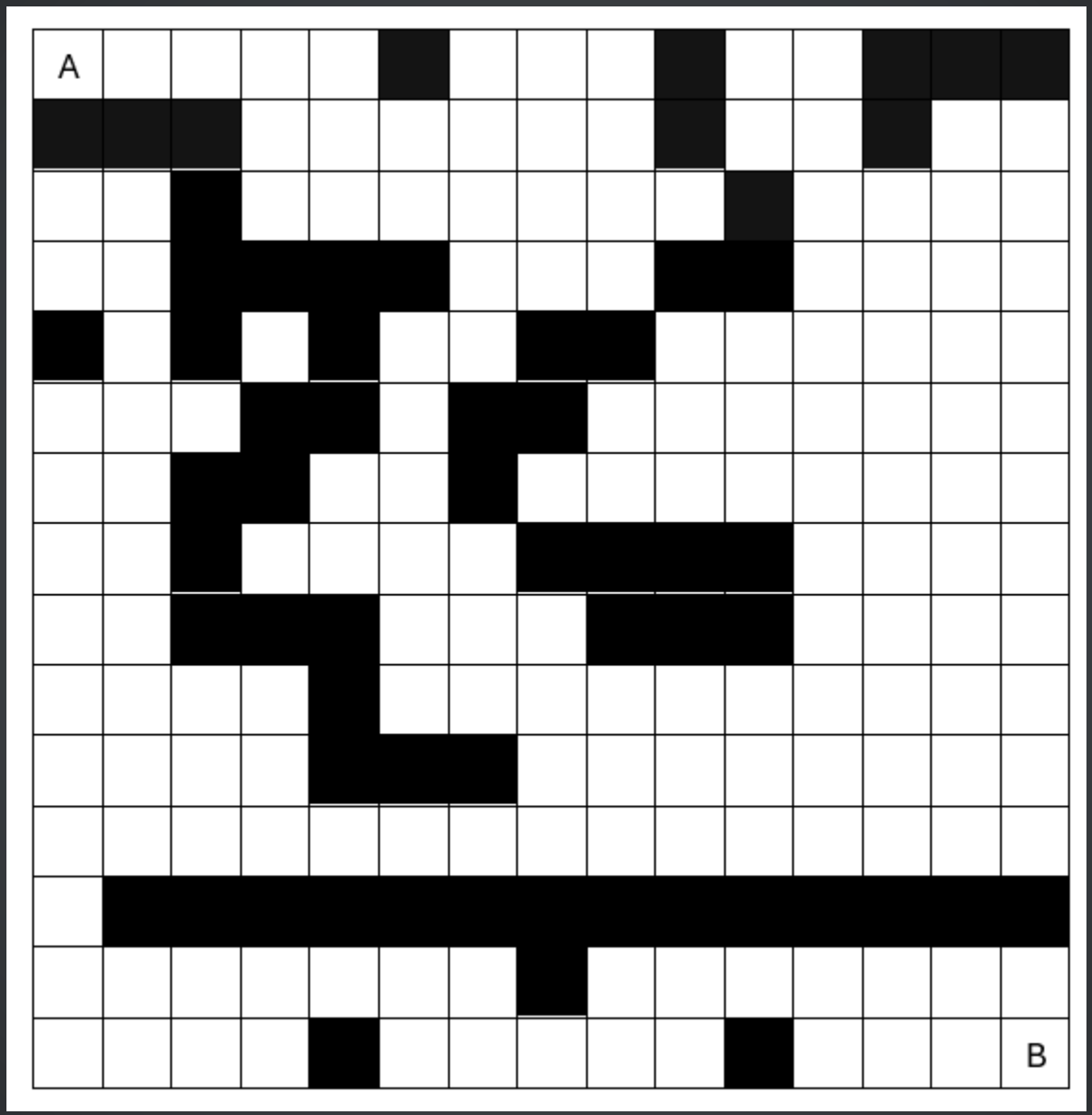
A More General Question

You are climbing a staircase. It takes n steps to reach the top.

Each time you can either climb 1 or 2 steps. In how many distinct ways can you climb to the top?

https://www.youtube.com/embed/tOYZcy2IzJA?enablejsapi=1

Exercise: Answer the question for $n = 10$.


A
B

This is known as Dijkstra's algorithm.

Its developer, Dijkstra, won the Turing Award in 1972.

Exercise: Write the code yourself

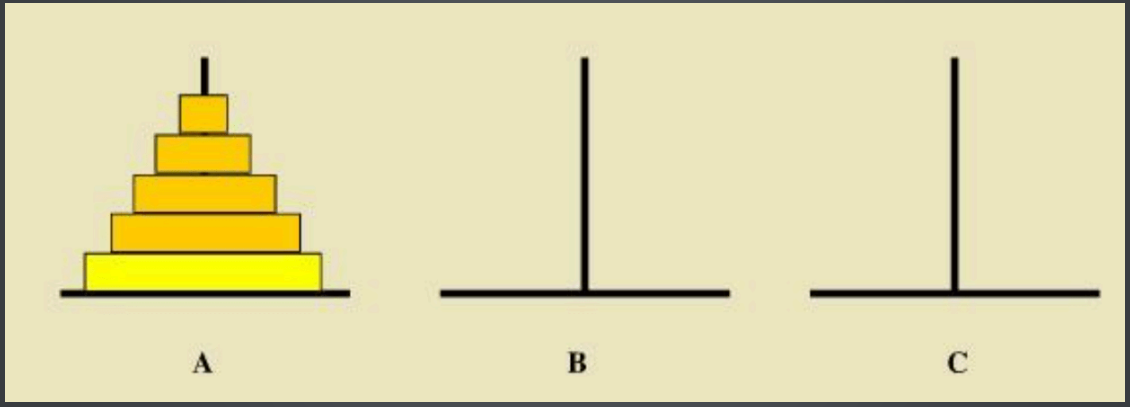
```
input = [[0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1],
         [1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0],
         [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
         [0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0],
         [1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0],
         [0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0],
         [0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0],
         [0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0],
         [0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0],
         [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
         [0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0],
         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
         [0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
         [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
         [0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0]]
```

Here, 1 means a stone and 0 means a feasible block

Exercise

Use AI to generate an interactive APP which demonstrates the algorithm as a game

Tower of Hanoi (汉诺塔)

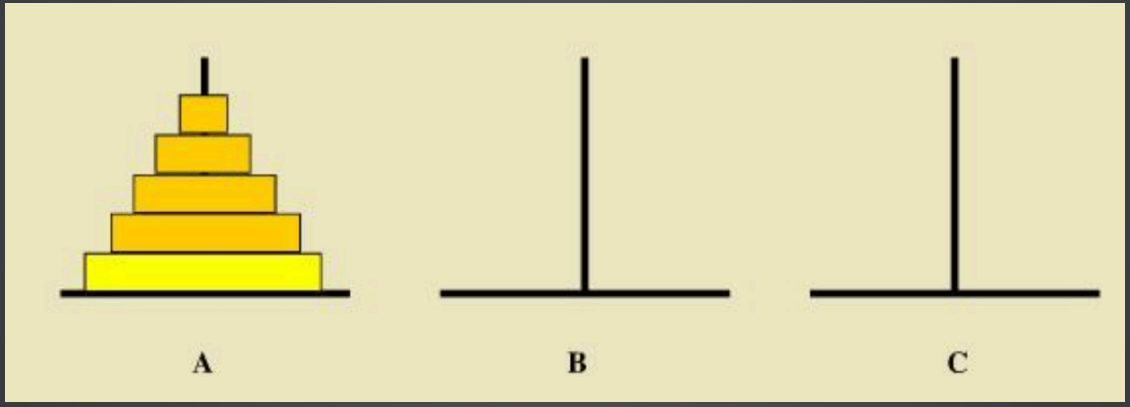


Tower of Hanoi

You want to move n disks from rod A to rod C. Here are the rules:

- Only one disk can be moved at each round.
- Each move consists of taking the upper disk from one of the stacks and placing it on top of another stack or on an empty rod.
- No disk may be placed on top of a disk that is smaller than it.

How to solve the problem?



Divide-and-Conquer

Exercise: Write a program which solves the Hanoi problem given any n. The output is a few steps such as $A \to B$, $B \to C$, $B \to A$, etc.

```python
def tower_of_hanoi(n, source, target, auxiliary):
    if n == 1:
        print(f"Move disk 1 from rod {source} to rod {target}")
        return
    tower_of_hanoi(n-1, source, auxiliary, target)
    print(f"Move disk {n} from rod {source} to rod {target}")
    tower_of_hanoi(n-1, auxiliary, target, source)

n = 3  # Number of disks
source_rod = 'A'
target_rod = 'C'
auxiliary_rod = 'B'

print(f"Solving Tower of Hanoi problem for {n} disks:")
tower_of_hanoi(n, source_rod, target_rod, auxiliary_rod)
```

Exercise: Can you propose another algorithm which uses the idea of divide-and-conquer?

Exercise