Week-4 PSOSM 2023

Question 1: What is the critical challenge that Latanya Sweeney's research on k-anonymity addresses?

a) make datasets larger for analysis
b) eliminate data from datasets
c) anonymise data without losing utility
d) ignore research utility for better privacy

Suppose you're working on a project that involves analysing sales data for a chain of stores. You are provided with a list of daily sales figures for a particular product over some time. The task is to calculate various statistics such as the total sales, average sales, and maximum sales for that product. Answer the following questions [2-4]:

Question 2: Which data structure can significantly enhance the efficiency and simplicity of your calculations?

a. Python numpy list
b. Python numpy array
c. Python variables
d. None of these

Question 3: Fill in the correct function calculate_total_sales(sales_list) which calculates the total sales from a list of daily sales figures using NumPy in the below code

```
import numpy as np

def calculate_total_sales(sales_list):
  <<<write your code here>>>>

    return total_sales

daily_sales = [1200, 1500, 1300, 1400, 1800, 1600]
total_sales = calculate_total_sales(daily_sales)
print("Total sales:", total_sales)
```

a. sales_array = np.array(sales_list)
   total_sales = np.sum(sales_array)
b. sales_array = np.list(sales_list)
   total_sales = np.sum(sales_array)
c. sales_array = np.array(sales_list)

total_sales = np.sum(np.concatenate(sales_array,sales_list))

d. None of the above

Question 4: Fill in the correct function calculate_maximum_sales(sales_list) which calculates the maximum sales from a list of daily sales figures using NumPy in the below code

```
import numpy as np

def calculate_maximum_sales(sales_list):
  <<<write your code here>>>>
   return max_sales

daily_sales = [1200, 1500, 1300, 1400, 1800, 1600]
maximum_sales = calculate_maximum_sales(daily_sales)
print("Maximum sales:", maximum_sales)
```

a. sales_array = np.array(sales_list)
   max_sales = np.max(sales_array)
b. sales_array = np.list(sales_list)
   max_sales = np.max(sales_array)
c. sales_array = np.array(sales_list)
   max_sales = np.max(np.concatenate(sales_array,sales_list))
d. None of the above

Imagine you have a NumPy array representing your monthly expenses for the past year. Each array element corresponds to the total expenses (in dollars) for a specific month. The task is to analyze and extract specific information.

```
import numpy as np

expenses_array = np.array([1200, 1500, 1300, 1400, 1800, 1600, 1400, 1250, 1350, 1500, 1700, 1900])
```

Answer the following questions from [5-7]:

Question 5: How would you extract expenses for the second half of the year (July to December)?

a. second_half_expenses = expenses_array[6:]
b. second_half_expenses = expenses_array[5:]
c. second_half_expenses = expenses_array[:12]
d. second_half_expenses = expenses_array[7:]

Question 6: Identify the months where expenses exceeded $1600.

a. high_expense_months = np.where(expenses_array < 1600)[0]
b. high_expense_months = np.where(expenses_array > 1600)[0]
c. high_expense_months = np.where(expenses_array > 1600))
d. high_expense_months = np.where(expenses_array > 1600)[0] + 1

Question 7: Calculate the average expenses for the first quarter of year.

a. first_quarter_average = np.mean(expenses_array[:4])
b. first_quarter_average = np.mean(expenses_array[:2])
c. first_quarter_average = np.mean(expenses_array[:3])
d. first_quarter_average = np.mean(expenses_array[:6])

Question 8: Choose the correct option for “result”

```
 import pandas as pd
data = {
   'Product': ['A', 'B', 'A', 'B', 'C', 'A', 'C', 'B', 'C', 'A'],
   'Category': ['Electronics', 'Clothing', 'Electronics', 'Clothing', 'Home', 'Electronics', 'Home',
'Clothing', 'Home', 'Electronics'],
   'Price': [500, 40, 600, 35, 100, 550, 80, 30, 90, 480]
}

sales_df = pd.DataFrame(data)
grouped = sales_df.groupby('Category')
result = grouped.agg({
   'Price': ['sum', 'mean']
}).reset_index()
# Rename the columns
result.columns = ['Category', 'Total Revenue', 'Average Price']
print(result)
```

a.

	Category	Total Revenue	Average Price
0	clothing	105	35.00
1	electronics	1630	543.33
2	home	270	90.00

b.

	Category	Total Revenue	Average Price
0	clothing	1630	35.00
1	electronics	105	543.33
2	home	270	90.00

c.

	Category	Total Revenue	Average Price
0	clothing	105	35.00
1	electronics	270	90.00
2	home	1630	543.33

d. None of the above

Question 9: What is the primary goal of k-anonymity in data privacy?

A) Ensure that every attribute in the dataset is unique.
B) Prevent unauthorised access to the dataset.
C) Minimize the amount of data collected.
D) Making it difficult to link specific individuals to their records.

Question 10: What will be the output of the below code:

```
import numpy as np
import matplotlib.pyplot as plt
time = np.linspace(0, 2 * np.pi, 1000)
amplitude = 1.0
frequency = 1.0
sine_wave = amplitude * np.sin(frequency * time)
plt.figure(figsize=(8, 6))
plt.plot(time, sine_wave, label='Sine Wave', color='blue')
plt.title('Sine Wave')
plt.xlabel('Time')
plt.ylabel('Amplitude')
```

```
plt.legend()
plt.grid(True)
plt.show()
```

a. A plot consisting of sine wave
b. A plot consisting of cosine wave
c. Logical error in code
d. None of the above

Solutions

1. Refer to lecture Week 4.1
2. In this example, by converting the sales data into a NumPy array, you can use the np.sum() function to easily calculate the total sales with just one line of code. NumPy's built-in functions are optimized for numerical computations and can provide better performance compared to traditional looping through a Python list. Converting the sales list to a NumPy array also opens up the possibility to perform various other mathematical operations (such as calculating the average, finding the maximum sales, etc.) efficiently and with concise code.

Question 3 to 8 and Q10 -please run the code

Q8: correct answer (code)

```
import pandas as pd
data = {
    'Product': ['A', 'B', 'A', 'B', 'C', 'A', 'C', 'B', 'C', 'A'],
    'Category': ['Electronics', 'Clothing', 'Electronics', 'Clothing', 'Home', 'Electronics', 'Home', 'Clothing', 'Home', 'Electronics'],
    'Price': [500, 40, 600, 35, 100, 550, 80, 30, 90, 480]
}

sales_df = pd.DataFrame(data)
grouped = sales_df.groupby('Category')
result = grouped.agg({
    'Price': ['sum', 'mean']
}).reset_index()
# Rename the columns
result.columns = ['Category', 'Total Revenue', 'Average Price']
print(result)
```

```
      Category  Total Revenue  Average Price
0     Clothing            105           35.0
1  Electronics           2130          532.5
2         Home            270           90.0
```

More cell actions

9. D) Making it difficult to link specific individuals to their records.

K-anonymity is a technique used in data privacy to protect individual identities within a dataset. The primary goal of k-anonymity is to make it difficult to identify specific individuals by ensuring that each record is indistinguishable from at least k-1 other records with respect to certain attributes.

Options A, B, and C are not accurate descriptions of the primary goal of k-anonymity. While ensuring uniqueness, preventing unauthorized access, and minimizing data collection are important considerations for data privacy, they are not the central focus of k-anonymity.