

DAY 1 LAB EXPERIMENTS

Name: M SANTHOSH

Reg.no: 192324014

Dept: B.Tech AI & DS

1. Scenario: You are working on a project that involves analyzing student performance data for a class of 32 students. The data is stored in a NumPy array named `student_scores`, where each row represents a student and each column represents a different subject. The subjects are arranged in the

following order: Math, Science, English, and History. Your task is to calculate the average score for each subject and identify the subject with the highest average score.

Question: How would you use NumPy arrays to calculate the average score for each subject and determine the subject with the highest average score? Assume 4x4 matrix that stores marks of each student in given order.

Solution:

```
import numpy as np
import pandas as pd

student_scores = np.loadtxt("/content/score.csv", delimiter=",", skiprows=1, usecols=(1, 2, 3, 4))

subjects = ["Math", "Science", "English", "History"]

average_scores = np.mean(student_scores, axis=0)

for i in range(len(subjects)):

    print(subjects[i], ":", round(average_scores[i], 2))

highest_subject = subjects[np.argmax(average_scores)]

print("Subject with highest average score:", highest_subject)
```

```
import numpy as np
import pandas as pd

student_scores = np.loadtxt("/content/score.csv", delimiter=",", skiprows=1, usecols=(1, 2, 3, 4))

subjects = ["Math", "Science", "English", "History"]

average_scores = np.mean(student_scores, axis=0)
for i in range(len(subjects)):
    print(subjects[i], ":", round(average_scores[i], 2))

Math : 81.97
Science : 82.56
English : 82.97
History : 82.88

highest_subject = subjects[np.argmax(average_scores)]
print("Subject with highest average score:", highest_subject)

Subject with highest average score: English
```

2.Scenario: You are a data analyst working for a company that sells products online. You have been tasked with analyzing the sales data for the past month. The data is stored in a NumPy array.

Question: How would you find the average price of all the products sold in the past month?

Assume 3x3 matrix with each row representing the sales for a different product

Solution:

```
import pandas as pd

df = pd.read_csv("/content/Sales.csv",
                 usecols=["Month_sales", "Price"])

df["Month_sales"] = df["Month_sales"].astype("category")

avg_price_per_month = (
    df.groupby("Month_sales", observed=True)["Price"].mean()
)

print(avg_price_per_month)
```

```
import pandas as pd
df = pd.read_csv("/content/Sales.csv",
                 usecols=["Month_sales", "Price"])

df["Month_sales"] = df["Month_sales"].astype("category")

avg_price_per_month = (
    df.groupby("Month_sales", observed=True)["Price"].mean()
)

print(avg_price_per_month)
```

| Month_sales | Price |
|----------------|-------------|
| April 2023 | 2799.195185 |
| August 2023 | 2765.507634 |
| December 2023 | 2806.723882 |
| February 2023 | 2610.790533 |
| January 2023 | 2914.791500 |
| January 2024 | 849.366667 |
| July 2023 | 2850.357353 |
| June 2023 | 2737.277283 |
| March 2023 | 2800.217375 |
| May 2023 | 2929.405833 |
| November 2023 | 2337.363579 |
| October 2023 | 2743.478636 |
| September 2023 | 2530.013971 |

Name: Price, dtype: float64

3. Scenario: You are working on a project that involves analyzing a dataset containing information

about houses in a neighborhood. The dataset is stored in a CSV file, and you have imported it into a NumPy array named `house_data`. Each row of the array represents a house, and the columns contain various features such as the number of bedrooms, square footage, and sale price.

Question: Using NumPy arrays and operations, how would you find the average sale price of houses with more than four bedrooms in the neighborhood?

Solution:

```
import numpy as np
```

```
import pandas as pd
```

```
house_data = np.genfromtxt(
```

```
    '/content/sample_data/House_Prediction.csv',
```

```
    delimiter=',',
```

```
    skip_header=1,
```

```
    dtype=float
```

```
)
```

```
filtered = house_data[house_data[:, 1] > 4]
```

```
avg_price = np.mean(filtered[:, -1])
```

```
print("Average Sale Price of houses with more than 4 bedrooms: ",round(avg_price,2))
```

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

```
house_data = np.genfromtxt(
    '/content/sample_data/House_Prediction.csv',
    delimiter=',',
    skip_header=1,
    dtype=float
)
```

```
filtered = house_data[house_data[:, 1] > 4]
avg_price = np.mean(filtered[:, -1])
print("Average Sale Price of houses with more than 4 bedrooms: ",round(avg_price,2))
```

```
Average Sale Price of houses with more than 4 bedrooms: 1566041.68
```

4. **Scenario:** You are working on a project that involves analyzing the sales performance of a

company over the past four quarters. The quarterly sales data is stored in a NumPy array named `sales_data`, where each element represents the sales amount for a specific quarter. Your task is to calculate the total sales for the year and determine the percentage increase in sales from the first quarter to the fourth quarter.

Question: Using NumPy arrays and arithmetic operations calculate the total sales for the year and determine the percentage increase in sales from the first quarter to the fourth quarter?

Solution:

```
import numpy as np
import pandas as pd
data = np.genfromtxt(
    "/content/Sales.csv",
    delimiter=",",
    dtype=str,
    skip_header=1
)

months = data[:, 1]
sales = data[:, 4].astype(float)
quarters = np.zeros(len(months), dtype=int)

for i, m in enumerate(months):
    if "Jan" in m or "Feb" in m or "Mar" in m:
        quarters[i] = 1
    elif "Apr" in m or "May" in m or "Jun" in m:
        quarters[i] = 2
    elif "Jul" in m or "Aug" in m or "Sep" in m:
        quarters[i] = 3
    else:
        quarters[i] = 4

Q1 = sales[quarters == 1].sum()
Q2 = sales[quarters == 2].sum()
```

```
Q3 = sales[quarters == 3].sum()
```

```
Q4 = sales[quarters == 4].sum()
```

```
total_sales = np.sum(sales)
```

```
percentage_increase = ((Q4 - Q1) / Q1) * 100
```

```
print("Total Sales for the Year:", round(total_sales,2))
```

```
print("Percentage Increase from Q1 to Q4:", round(percentage_increase,2))
```

```
quarters = np.zeros(len(months), dtype=int)

for i, m in enumerate(months):
    if "Jan" in m or "Feb" in m or "Mar" in m:
        quarters[i] = 1
    elif "Apr" in m or "May" in m or "Jun" in m:
        quarters[i] = 2
    elif "Jul" in m or "Aug" in m or "Sep" in m:
        quarters[i] = 3
    else:
        quarters[i] = 4

Q1 = sales[quarters == 1].sum()
Q2 = sales[quarters == 2].sum()
Q3 = sales[quarters == 3].sum()
Q4 = sales[quarters == 4].sum()

total_sales = np.sum(sales)
percentage_increase = ((Q4 - Q1) / Q1) * 100

print("Total Sales for the Year:", round(total_sales,2))
print("Percentage Increase from Q1 to Q4:", round(percentage_increase,2))

Total Sales for the Year: 5019265.23
Percentage Increase from Q1 to Q4: 4.22
```

5. **Scenario:** You are a data analyst working for a car manufacturing company. As part of your analysis, you have a dataset containing information about the fuel efficiency of different car models. The dataset is stored in a NumPy array named `fuel_efficiency`, where each element

represents the fuel efficiency (in miles per gallon) of a specific car model. Your task is to calculate the average fuel efficiency and determine the percentage improvement in fuel efficiency between two car models.

Question: How would you use NumPy arrays and arithmetic operations to calculate the average fuel efficiency and determine the percentage improvement in fuel efficiency between two car models?

Solution:

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

data = np.genfromtxt(
    '/content/fuel.csv',
    delimiter=',',
    dtype=str,
    skip_header=1
)

highway_mpg = data[:, 7].astype(float)
average_mpg = np.mean(highway_mpg)

model1 = "mustang"
model2 = "forester awd"
model_column = data[:, 9]
mpg1 = highway_mpg[model_column == model1]
mpg2 = highway_mpg[model_column == model2]
print("Average Highway MPG:", round(average_mpg,2))

if len(mpg1) > 0 and len(mpg2) > 0:
    old = mpg1[0]
    new = mpg2[0]

    improvement = ((new - old) / old) * 100
    print("Percentage Improvement:", round(improvement,2), "%")
else:
```

```
print("Model not found in dataset")
```

```
highway_mpg = data[:, 7].astype(float)
```

```
average_mpg = np.mean(highway_mpg)
```

```
print("Average Highway MPG:", round(average_mpg,2))
```

```
Average Highway MPG: 28.61
```

```
model1 = "mustang"
```

```
model2 = "forester awd"
```

```
model_column = data[:, 9]
```

```
mpg1 = highway_mpg[model_column == model1]
```

```
mpg2 = highway_mpg[model_column == model2]
```

```
if len(mpg1) > 0 and len(mpg2) > 0:
```

```
    old = mpg1[0]
```

```
    new = mpg2[0]
```

```
    improvement = ((new - old) / old) * 100
```

```
    print("Percentage Improvement:", round(improvement,2), "%")
```

```
else:
```

```
    print("Model not found in dataset")
```

```
Percentage Improvement: -3.45 %
```