Numpy Pandas and Matplotlib

CS-371L Artificial Intelligence

Submitted to:

Mr. Nauman Shafi

Submitted by:

Gul-e-Zahra 2022-CS-75

Department of Computer Science
University of Engineering and Technology
Lahore, Pakistan

Contents

List of Figures

1 Weather Data Analysis

1.1 Code

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
dates = pd.date_range(start='2024-01-01', periods=365)
temperature = np.random.randint(10, 40, size=365)
humidity = np.random.randint(30, 90, size=365)
wind_speed = np.random.randint(0, 20, size=365)
weather_conditions = np.random.choice(['Sunny', 'Rainy', 'Cloudy'], size=365)

weather_data = pd.DataFrame({
    'Date': dates,
    'Temperature': temperature,
    'Humidity': humidity,
    'Wind Speed': wind_speed,
    'Weather Condition': weather_conditions
})
print(weather_data)


temperature_array = weather_data['Temperature'].to_numpy()
mean_temp = np.mean(temperature_array)
median_temp = np.median(temperature_array)
std_temp = np.std(temperature_array)

print(f'Mean Temperature: {mean_temp:.2f}°C')
print(f'Median Temperature: {median_temp:.2f}°C')
print(f'Standard Deviation of Temperature: {std_temp:.2f}°C')



sunny_hot_days = weather_data[(weather_data['Temperature'] > 30) & (weather_data['Weather Condition'] == 'Sunny')]
num_sunny_hot_days = sunny_hot_days.shape[0]

print(sunny_hot_days)
print(f"Number of days with temperature above 30°C and Sunny: {num_sunny_hot_days}")

average_humidity = weather_data.groupby('Weather Condition')['Humidity'].mean().reset_index()
average_humidity.columns = ['Weather Condition', ' Average']

print(average_humidity)


plt.figure(figsize=(12, 6))
plt.plot(weather_data['Date'], weather_data['Temperature'])
plt.title('Temperature Variation Over the Year')
plt.xlabel('Date')
plt.ylabel('Temperature (°C)')
plt.show()

weather_condition_counts = weather_data['Weather Condition'].value_counts()

plt.figure(figsize=(8, 5))
weather_condition_counts.plot(kind='bar')
plt.title('Number of Days for Each Weather Condition')
plt.xlabel('Weather Condition')
plt.ylabel('Number of Days')
plt.show()
```

Figure 1: Code of Weather Data Analysis

1.2 Console Output

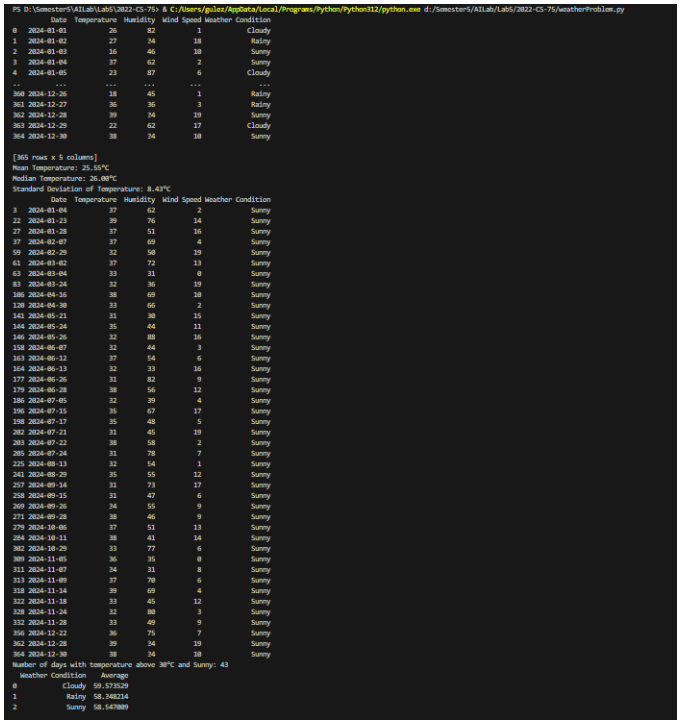


Figure 2: Output of Weather Data Analysis

1.3 Matplotlib usage

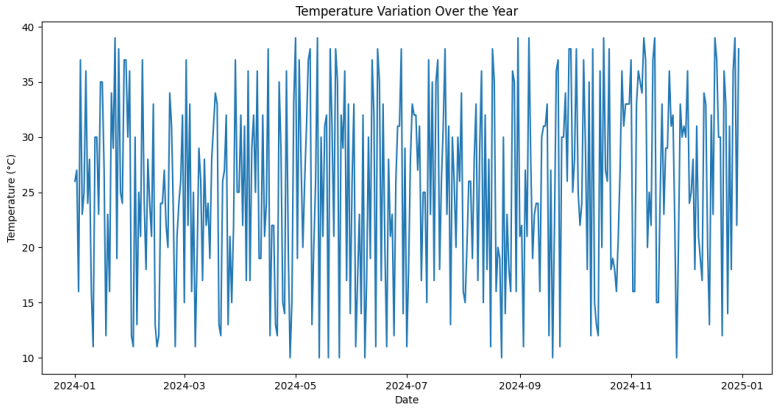


Figure 3: Weather Data Analysis

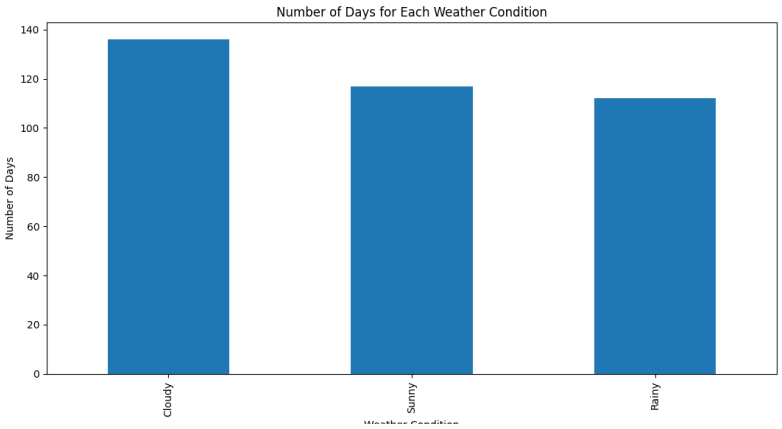
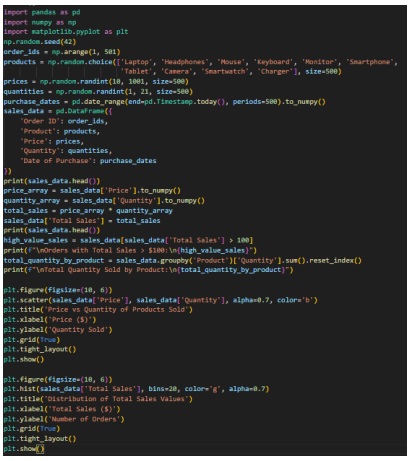


Figure 4: Weather Data Analysis

2 Sales Data Analysis

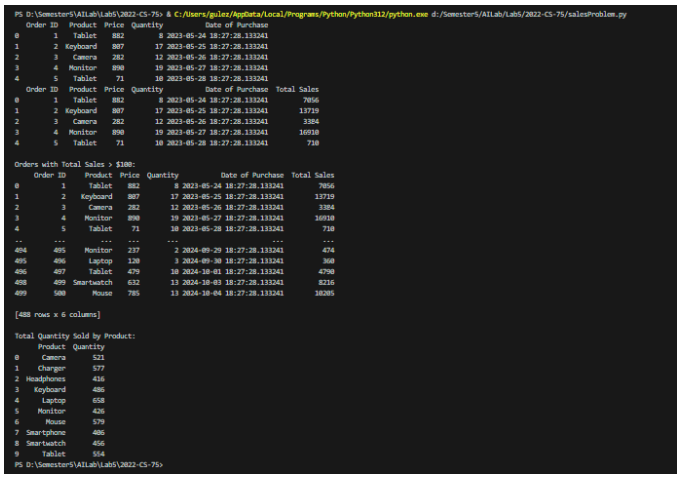
2.1 Code



```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
np.random.seed(42)
order_ids = np.arange(1, 501)
products = np.random.choice(['Laptop', 'Headphones', 'Mouse', 'Keyboard', 'Monitor', 'Smartphone',
                             'Tablet', 'Camera', 'Smartwatch', 'Charger'], size=500)
prices = np.random.randint(10, 1001, size=500)
quantities = np.random.randint(1, 21, size=500)
purchase_dates = pd.date_range(end=pd.Timestamp.today(), periods=500).to_numpy()
sales_data = pd.DataFrame({
    'Order ID': order_ids,
    'Product': products,
    'Price': prices,
    'Quantity': quantities,
    'Date of Purchase': purchase_dates
})
print(sales_data.head())
price_array = sales_data['Price'].to_numpy()
quantity_array = sales_data['Quantity'].to_numpy()
total_sales = price_array * quantity_array
sales_data['Total Sales'] = total_sales
print(sales_data.head())
high_value_sales = sales_data[sales_data['Total Sales'] > 100]
print(f"\nOrders with Total Sales > $100:\n{high_value_sales}")
total_quantity_by_product = sales_data.groupby('Product')['Quantity'].sum().reset_index()
print(f"\nTotal Quantity Sold by Product:\n{total_quantity_by_product}")

plt.figure(figsize=(10, 6))
plt.scatter(sales_data['Price'], sales_data['Quantity'], alpha=0.7, color='b')
plt.title('Price vs Quantity of Products Sold')
plt.xlabel('Price ($)')
plt.ylabel('Quantity Sold')
plt.grid(True)
plt.tight_layout()
plt.show()

plt.figure(figsize=(10, 6))
plt.hist(sales_data['Total Sales'], bins=20, color='g', alpha=0.7)
plt.title('Distribution of Total Sales Values')
plt.xlabel('Total Sales ($)')
plt.ylabel('Number of Orders')
plt.grid(True)
plt.tight_layout()
plt.show()
```

Figure 5: Code of Sales Data Analysis

2.2 Console Output



```
PS D:\Semester5\AILab\Labs\2022-CS-75> & C:/Users/gulzz/AppData/Local/Programs/Python/Python312/python.exe d:/Semester5/AILab/Labs/2022-CS-75/salesProblem.py
   Order ID   Product  Price  Quantity          Date of Purchase
0         1    Tablet    882         8 2023-05-24 18:27:28.133241
1         2  Keyboard    807        17 2023-05-25 18:27:28.133241
2         3    Camera    282        12 2023-05-26 18:27:28.133241
3         4   Monitor    890        19 2023-05-27 18:27:28.133241
4         5    Tablet     71        10 2023-05-28 18:27:28.133241
   Order ID   Product  Price  Quantity          Date of Purchase  Total Sales
0         1    Tablet    882         8 2023-05-24 18:27:28.133241         7056
1         2  Keyboard    807        17 2023-05-25 18:27:28.133241        13719
2         3    Camera    282        12 2023-05-26 18:27:28.133241         3384
3         4   Monitor    890        19 2023-05-27 18:27:28.133241        16910
4         5    Tablet     71        10 2023-05-28 18:27:28.133241          710

Orders with Total Sales > $100:
     Order ID     Product  Price  Quantity          Date of Purchase  Total Sales
0           1      Tablet    882         8 2023-05-24 18:27:28.133241         7056
1           2    Keyboard    807        17 2023-05-25 18:27:28.133241        13719
2           3      Camera    282        12 2023-05-26 18:27:28.133241         3384
3           4     Monitor    890        19 2023-05-27 18:27:28.133241        16910
4           5      Tablet     71        10 2023-05-28 18:27:28.133241          710
..        ...         ...    ...       ...                       ...          ...
484       485     Monitor    237         2 2024-09-29 18:27:28.133241          474
485       486      Laptop    120         3 2024-09-30 18:27:28.133241          360
486       487      Tablet    479        10 2024-10-01 18:27:28.133241         4790
488       489  Smartwatch    632        13 2024-10-03 18:27:28.133241         8216
499       500       Mouse    785        13 2024-10-04 18:27:28.133241        10205

[488 rows x 6 columns]

Total Quantity Sold by Product:
      Product  Quantity
0      Camera       521
1     Charger       577
2  Headphones       416
3    Keyboard       486
4      Laptop       658
5     Monitor       426
6       Mouse       579
7  Smartphone       486
8  Smartwatch       456
9      Tablet       514
PS D:\Semester5\AILab\Labs\2022-CS-75>
```

Figure 6: Output of Sales Data Analysis

2.3 Matplotlib usage

Figure 7: Sales Data Analysis

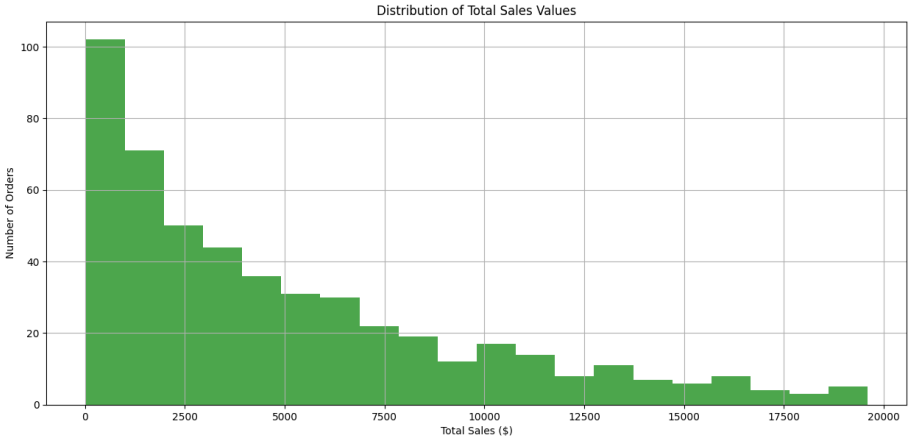
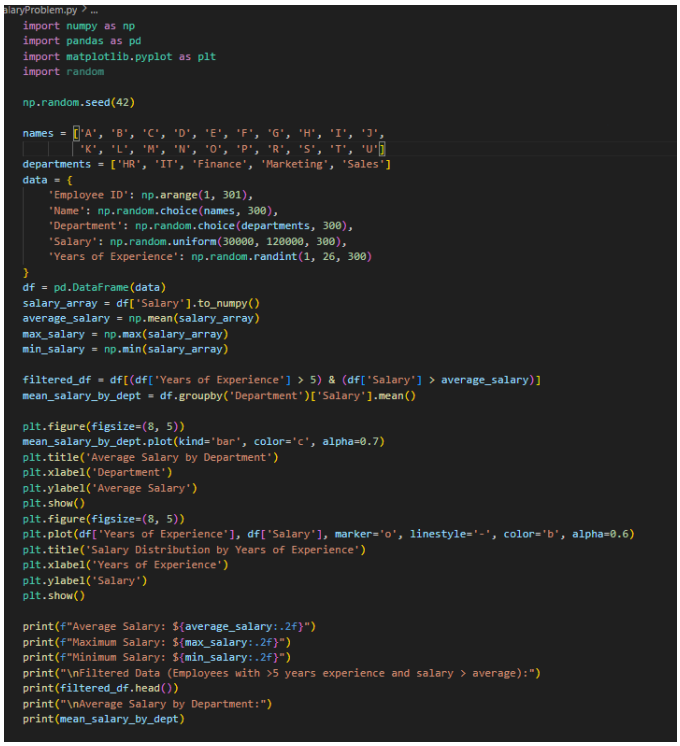


Figure 8: Sales Data Analysis

3 Employee Salary Analysis

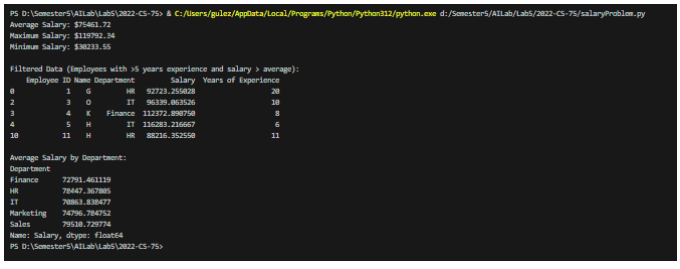
3.1 Code



```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import random

np.random.seed(42)

names = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J',
         'K', 'L', 'M', 'N', 'O', 'P', 'R', 'S', 'T', 'U']
departments = ['HR', 'IT', 'Finance', 'Marketing', 'Sales']
data = {
    'Employee ID': np.arange(1, 301),
    'Name': np.random.choice(names, 300),
    'Department': np.random.choice(departments, 300),
    'Salary': np.random.uniform(30000, 120000, 300),
    'Years of Experience': np.random.randint(1, 26, 300)
}
df = pd.DataFrame(data)
salary_array = df['Salary'].to_numpy()
average_salary = np.mean(salary_array)
max_salary = np.max(salary_array)
min_salary = np.min(salary_array)

filtered_df = df[(df['Years of Experience'] > 5) & (df['Salary'] > average_salary)]
mean_salary_by_dept = df.groupby('Department')['Salary'].mean()

plt.figure(figsize=(8, 5))
mean_salary_by_dept.plot(kind='bar', color='c', alpha=0.7)
plt.title('Average Salary by Department')
plt.xlabel('Department')
plt.ylabel('Average Salary')
plt.show()
plt.figure(figsize=(8, 5))
plt.plot(df['Years of Experience'], df['Salary'], marker='o', linestyle='-', color='b', alpha=0.6)
plt.title('Salary Distribution by Years of Experience')
plt.xlabel('Years of Experience')
plt.ylabel('Salary')
plt.show()

print(f"Average Salary: ${average_salary:.2f}")
print(f"Maximum Salary: ${max_salary:.2f}")
print(f"Minimum Salary: ${min_salary:.2f}")
print("\nFiltered Data (Employees with >5 years experience and salary > average):")
print(filtered_df.head())
print("\nAverage Salary by Department:")
print(mean_salary_by_dept)
```

Figure 9: Code of Employee Salary Analysis

3.2 Console Output



```
PS D:\Semester5\AILab\Lab5\2022-CS-75> & C:/Users/gulez/AppData/Local/Programs/Python/Python312/python.exe d:/Semester5/AILab/Lab5/2022-CS-75/salaryProblem.py
Average Salary: $75461.72
Maximum Salary: $119792.34
Minimum Salary: $30233.55

Filtered Data (Employees with >5 years experience and salary > average):
    Employee ID Name Department        Salary  Years of Experience
0             1    G         HR   92723.255028                   20
2             3    O         IT   96139.863526                   10
3             4    K    Finance  112372.898750                    8
4             5    H         IT  116283.216667                    6
10           11    H         HR   88216.352550                   11

Average Salary by Department:
Department
Finance      72791.461119
HR           78447.367985
IT           70863.838477
Marketing    74796.784752
Sales        79510.729774
Name: Salary, dtype: float64
PS D:\Semester5\AILab\Lab5\2022-CS-75>
```

Figure 10: Output of Employee Salary Analysis

3.3 Matplotlib usage

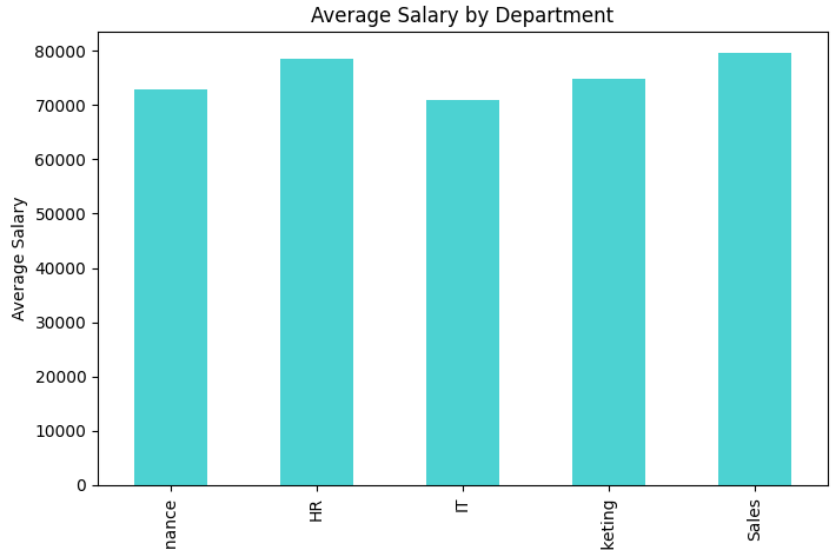


Figure 11: Employee Salary Analysis

Figure 12: Employee Salary Analysis

4 Exam Score Analysis

4.1 Code

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

np.random.seed(42)

names = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J',
         'K', 'L', 'M', 'N', 'O', 'P', 'R', 'S', 'T', 'U']
subjects = ['Math', 'Physics', 'Chemistry', 'Biology', 'English']
data = {
    'Student ID': np.random.randint(1000, 2000, 200),
    'Name': np.random.choice(names, 200),
    'Subject': np.random.choice(subjects, 200),
    'Score': np.random.randint(0, 101, 200),
    'Total Marks': [100] * 200
}
df = pd.DataFrame(data)
scores_array = df['Score'].to_numpy()
mean_score = np.mean(scores_array)
median_score = np.median(scores_array)
std_deviation = np.std(scores_array)
high_scorers = df[df['Score'] > 80]
num_high_scorers = high_scorers.shape[0]
avg_score_by_subject = df.groupby('Subject')['Score'].mean()
plt.figure(figsize=(8, 5))
plt.hist(df['Score'], bins=20, color='skyblue', edgecolor='black', alpha=0.7)
plt.title('Score Distribution Across All Students')
plt.xlabel('Score')
plt.ylabel('Frequency')
plt.show()
plt.figure(figsize=(8, 5))
avg_score_by_subject.plot(kind='bar', color='lightgreen', alpha=0.8)
plt.title('Average Scores by Subject')
plt.xlabel('Subject')
plt.ylabel('Average Score')
plt.show()
print(f"Mean Score: {mean_score:.2f}")
print(f"Median Score: {median_score:.2f}")
print(f"Standard Deviation: {std_deviation:.2f}")
print(f"Number of students scoring above 80%: {num_high_scorers}")
print("\nAverage score by subject:")
print(avg_score_by_subject)
```

Figure 13: Code of Exam Score Analysis

4.2 Console Output

```
PS D:\Semester5\AILab\Lab5\2022-CS-75> & C:/Users/gulez/AppData/Local/Programs/Python/Python312/python.exe d:/Semester5/AILab/Lab5/2022-CS-75/examProblem.py
Mean Score: 48.14
Median Score: 48.00
Standard Deviation: 29.27
Number of students scoring above 80%: 36

Average score by subject:
Subject
Biology      42.952488
Chemistry    56.387097
English      48.510638
Math         50.810811
Physics      44.595238
Name: Score, dtype: float64
PS D:\Semester5\AILab\Lab5\2022-CS-75>
```

Figure 14: Output of Exam Score Analysis

4.3 Matplotlib usage

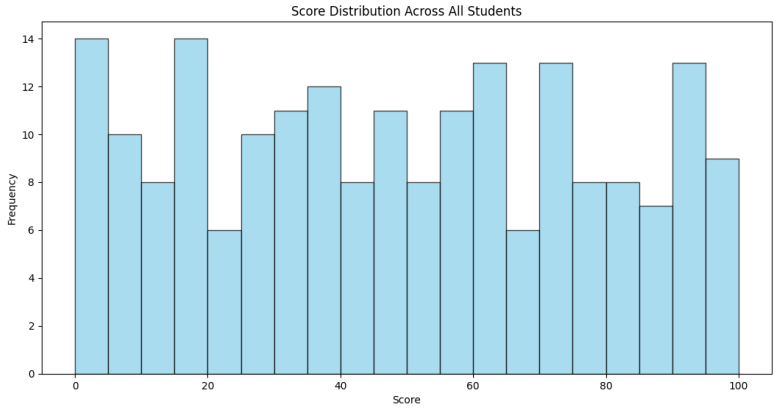


Figure 15: Exam Score Analysis

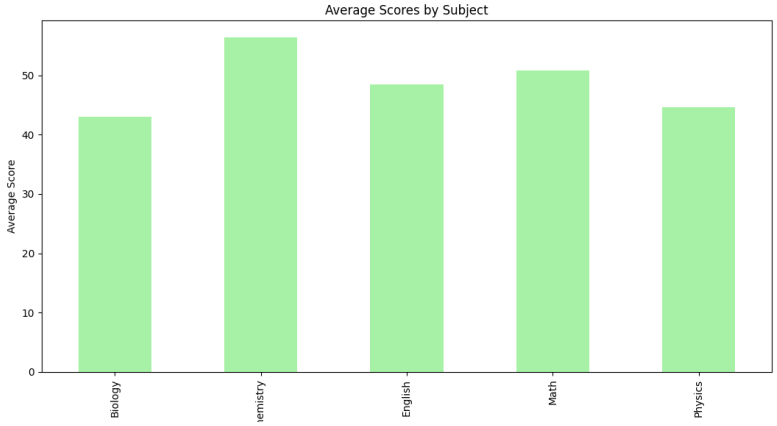


Figure 16: Exam Score Analysis

5 Stock Market Analysis

5.1 Code

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from datetime import datetime, timedelta
import random

np.random.seed(42)
start_date = datetime.now() - timedelta(days=730)
date_list = [start_date + timedelta(days=random.randint(0, 730)) for _ in range(1000)]
companies = ['CompanyA', 'CompanyB', 'CompanyC', 'CompanyD', 'CompanyE']
data = {
    'Date': date_list,
    'Company': np.random.choice(companies, 1000),
    'Open Price': np.random.uniform(50, 500, 1000),
    'Close Price': np.random.uniform(50, 500, 1000),
    'Volume Traded': np.random.randint(1000, 1000000, 1000)
}
df = pd.DataFrame(data)
close_price_array = df['Close Price'].to_numpy()
daily_percentage_change = np.diff(close_price_array) / close_price_array[:-1] * 100
df['Daily Percentage Change'] = np.insert(daily_percentage_change, 0, 0)
price_increase_df = df[df['Daily Percentage Change'] > 2]
total_volume_by_company = df.groupby('Company')['Volume Traded'].sum()
company = 'CompanyA'
company_data = df[df['Company'] == company].sort_values('Date')
plt.figure(figsize=(10, 6))
plt.plot(company_data['Date'], company_data['Close Price'], marker='o', color='b', alpha=0.6)
plt.title(f'Trend of Close Price over Time for {company}')
plt.xlabel('Date')
plt.ylabel('Close Price')
plt.xticks(rotation=45)
plt.show()
avg_percentage_change_by_company = df.groupby('Company')['Daily Percentage Change'].mean()

plt.figure(figsize=(8, 5))
avg_percentage_change_by_company.plot(kind='bar', color='orange', alpha=0.8)
plt.title('Average Daily Percentage Change in Close Price by Company')
plt.xlabel('Company')
plt.ylabel('Average Percentage Change')
plt.show()
print("\nDays when stock price increased by more than 2%:")
print(price_increase_df.head())
print("\nTotal Volume Traded by Company:")
print(total_volume_by_company)
```

Figure 17: Code of Stock Market Analysis

5.2 Console Output

```
PS D:\Semester5\AILab\Lab5\2022-CS-75> & C:/Users/gulez/AppData/Local/Programs/Python/Python312/python.exe d:/Semester5/AILab/Lab5/2022-CS-75/stockProblem.py

Days when stock price increased by more than 2%:
                  Date   Company  Open Price  Close Price  Volume Traded  Daily Percentage Change
4  2023-09-28 18:25:59.128924  CompanyE  368.257385   480.425629         949377               766.430850
8  2024-07-26 18:25:59.128924  CompanyC  240.858882   207.366453         493402                13.042581
9  2024-06-06 18:25:59.128924  CompanyE  381.399986   398.594889         812804                92.217348
14 2023-11-16 18:25:59.128924  CompanyD  493.178540   347.136224         751186               268.204743
15 2023-09-06 18:25:59.128924  CompanyB  427.504139   393.967698         801554                13.490806

Total Volume Traded by Company:
Company
CompanyA    185311742
CompanyB     89984723
CompanyC     97178445
CompanyD    102198826
CompanyE    101643489
Name: Volume Traded, dtype: int32
PS D:\Semester5\AILab\Lab5\2022-CS-75>
```

Figure 18: Output of Stock Market Analysis

5.3 Matplotlib usage

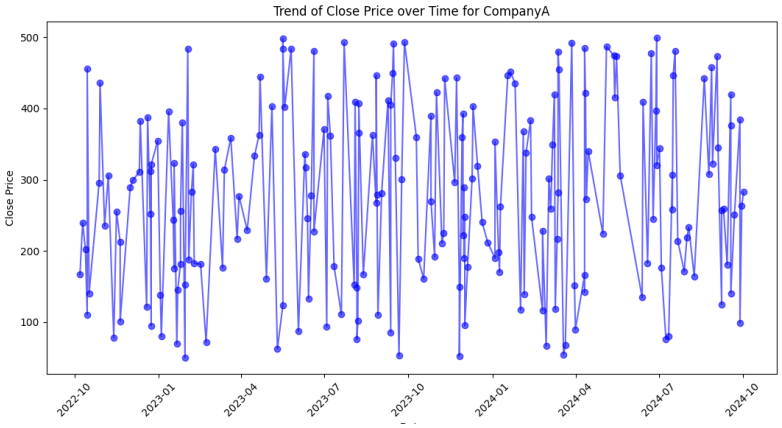


Figure 19: Exam Score Analysis

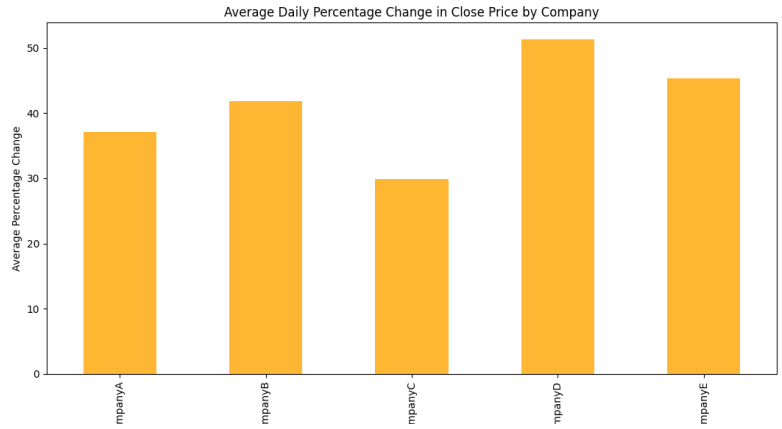


Figure 20: Stock Market Analysis

6 Customer Reviews Analysis

6.1 Code

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# Path to your CSV file
file_path = r'D:\\Semester5\\AILab\\Lab5\\2022-CS-75\\Customer_support_data.csv'

# Load the data from the CSV file
df = pd.read_csv(file_path)

# Numpy Task
# Extract the 'CSAT Score' column
csat_scores = df['CSAT Score'].to_numpy()

# Calculate mean, median, and standard deviation
mean_csat = np.mean(csat_scores)
median_csat = np.median(csat_scores)
std_dev_csat = np.std(csat_scores)

# Display the results
print("Mean CSAT Score:", mean_csat)
print("Median CSAT Score:", median_csat)
print("Standard Deviation of CSAT Score:", std_dev_csat)

# Pandas Task
#1
filtered_df = df[df['CSAT Score'] > 4]
# Display the filtered data
print("filtered ",filtered_df)
#2
# Count the number of issues handled by each agent
agent_issue_count = df['Agent_name'].value_counts()
# Display the count of issues handled by each agent
print(agent_issue_count)
#3
# Group by 'category' and calculate the average CSAT Score
average_csat_by_category = df.groupby('category')['CSAT Score'].mean()
# Display the average CSAT Score for each category
print(average_csat_by_category)


# Matplot Task 1
# Plot a histogram for the 'CSAT Score'
plt.figure(figsize=(8, 6))
plt.hist(df['CSAT Score'], bins=5, edgecolor='black', color='skyblue')
plt.title('Distribution of CSAT Scores')
plt.xlabel('CSAT Score')
plt.ylabel('Frequency')
plt.grid(True)
# Display the histogram
plt.show()

# Matplot Task 2
# Group by 'category' and calculate the average CSAT Score
average_csat_by_category = df.groupby('category')['CSAT Score'].mean()
# Create a bar chart
plt.figure(figsize=(10, 6))
average_csat_by_category.plot(kind='bar', color='skyblue', edgecolor='black')
plt.title('Average CSAT Score by Category')
plt.xlabel('Category')
plt.ylabel('Average CSAT Score')
plt.xticks(rotation=45)
plt.grid(axis='y')
# Display the bar chart
plt.tight_layout()
plt.show()
```

Figure 21: Code of Customer Reviews Analysis

6.2 Console Output

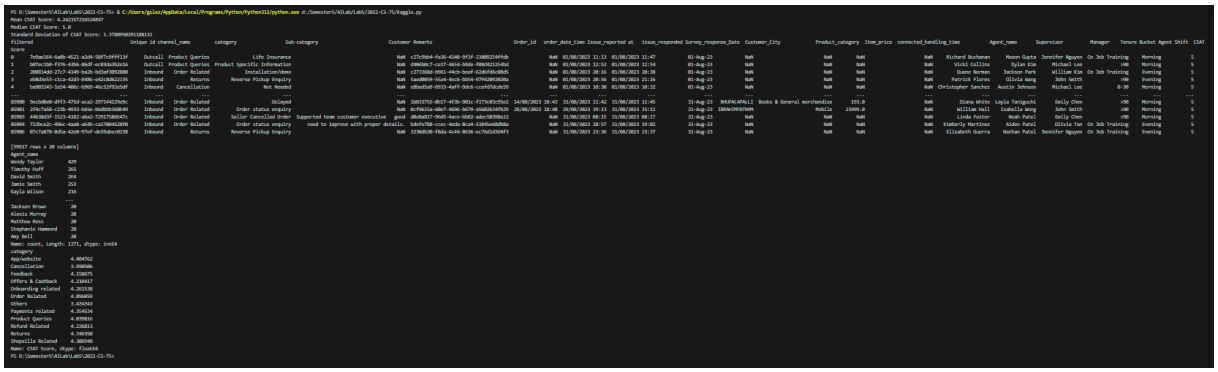


Figure 22: Output of Customer Reviews Analysis

6.3 Matplotlib usage

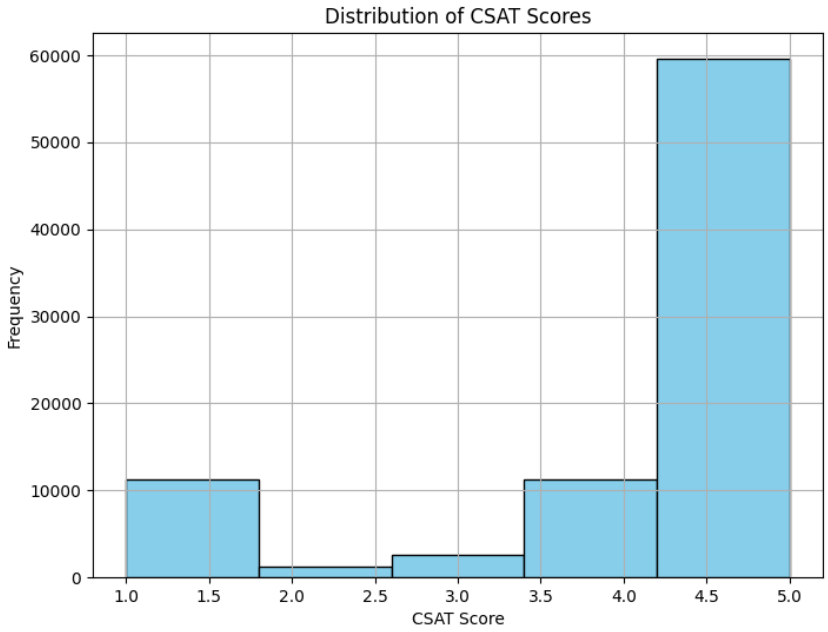


Figure 23: Customer Reviews Analysis

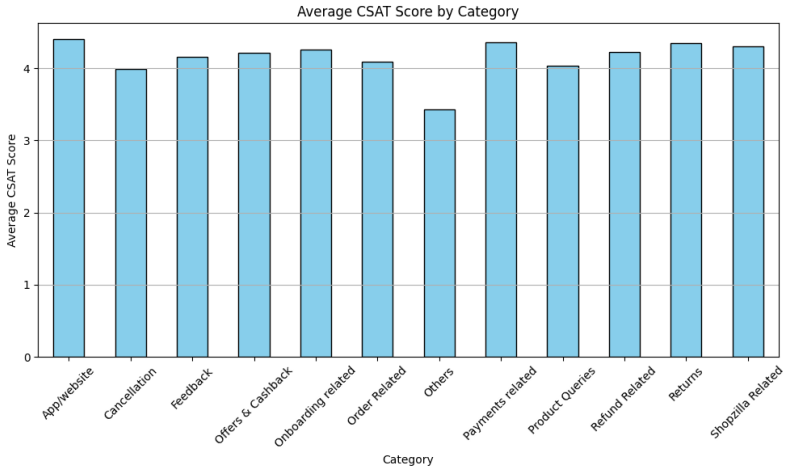


Figure 24: Customer Reviews Analysis