

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

1	Weather Data Analysis	4
1.1	Code	4
1.2	Console Output	5
1.3	Matplotlib usage	6
2	Sales Data Analysis	7
2.1	Code	7
2.2	Console Output	7
2.3	Matplotlib usage	8
3	Employee Salary Analysis	9
3.1	Code	9
3.2	Console Output	9
3.3	Matplotlib usage	10
4	Exam Score Analysis	11
4.1	Code	11
4.2	Console Output	11
4.3	Matplotlib usage	12
5	Stock Market Analysis	13
5.1	Code	13
5.2	Console Output	13
5.3	Matplotlib usage	14
6	Customer Reviews Analysis	15
6.1	Code	15
6.2	Console Output	16
6.3	Matplotlib usage	16

List of Figures

1	Code of Weather Data Analysis	4
2	Output of Weather Data Analysis	5
3	Weather Data Analysis	6
4	Weather Data Analysis	6
5	Code of Sales Data Analysis	7
6	Output of Sales Data Analysis	7
7	Sales Data Analysis	8
8	Sales Data Analysis	8
9	Code of Employee Salary Analysis	9
10	Output of Employee Salary Analysis	9
11	Employee Salary Analysis	10
12	Employee Salary Analysis	10
13	Code of Exam Score Analysis	11
14	Output of Exam Score Analysis	11

15	Exam Score Analysis	12
16	Exam Score Analysis	12
17	Code of Stock Market Analysis	13
18	Output of Stock Market Analysis	13
19	Exam Score Analysis	14
20	Stock Market Analysis	14
21	Code of Customer Reviews Analysis	15
22	Output of Customer Reviews Analysis	16
23	Customer Reviews Analysis	16
24	Customer Reviews Analysis	17

1 Weather Data Analysis

1.1 Code

```

1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4 dates = pd.date_range(start='2024-01-01', periods=365)
5 temperature = np.random.randint(10, 40, size=365)
6 humidity = np.random.randint(30, 90, size=365)
7 wind_speed = np.random.randint(0, 20, size=365)
8 weather_conditions = np.random.choice(['Sunny', 'Rainy', 'Cloudy'], size=365)
9
10 weather_data = pd.DataFrame({
11     'Date': dates,
12     'Temperature': temperature,
13     'Humidity': humidity,
14     'Wind Speed': wind_speed,
15     'Weather Condition': weather_conditions
16 })
17 print(weather_data)
18
19
20 temperature_array = weather_data['Temperature'].to_numpy()
21 mean_temp = np.mean(temperature_array)
22 median_temp = np.median(temperature_array)
23 std_temp = np.std(temperature_array)
24
25 print(f'Mean Temperature: {mean_temp:.2f}°C')
26 print(f'Median Temperature: {median_temp:.2f}°C')
27 print(f'Standard Deviation of Temperature: {std_temp:.2f}°C')
28
29
30 sunny_hot_days = weather_data[(weather_data['Temperature'] > 30) & (weather_data['Weather Condition'] == 'Sunny')]
31 num_sunny_hot_days = sunny_hot_days.shape[0]
32
33 print(sunny_hot_days)
34 print(f"Number of days with temperature above 30°C and Sunny: {num_sunny_hot_days}")
35
36 average_humidity = weather_data.groupby('Weather Condition')['Humidity'].mean().reset_index()
37 average_humidity.columns = ['Weather Condition', 'Average']
38
39 print(average_humidity)
40
41
42 plt.figure(figsize=(12, 6))
43 plt.plot(weather_data['Date'], weather_data['Temperature'])
44 plt.title('Temperature Variation Over the Year')
45 plt.xlabel('Date')
46 plt.ylabel('Temperature (°C)')
47 plt.show()
48
49
50 weather_condition_counts = weather_data['Weather Condition'].value_counts()
51
52 plt.figure(figsize=(8, 5))
53 weather_condition_counts.plot(kind='bar')
54 plt.title('Number of Days for Each Weather Condition')
55 plt.xlabel('Weather Condition')
56 plt.ylabel('Number of Days')
57 plt.show()
58

```

Figure 1: Code of Weather Data Analysis

1.2 Console Output

```

PS D:\Semester5\AILab\Lab5\2022-CS-75> & C:\Users\pauls\AppData\Local\Programs\Python\Python312\python.exe d:/Semester5/AILab/Lab5/2022-CS-75/WeatherProblem.py
Data Temperature Humidity Wind Speed Weather Condition
0 2024-01-01 20 82 1 Cloudy
1 2024-01-01 27 84 28 Rainy
2 2024-01-01 10 45 18 Sunny
3 2024-01-04 37 62 2 Sunny
4 2024-01-05 23 87 6 Cloudy
... ..
308 2024-12-26 18 45 1 Rainy
309 2024-12-27 36 36 3 Rainy
312 2024-12-28 39 34 19 Sunny
313 2024-12-29 22 62 17 Cloudy
314 2024-12-30 39 34 18 Sunny

[365 rows x 5 columns]
Mean Temperature: 35.54°C
Median Temperature: 35.89°C
Standard Deviation of Temperature: 8.43°C
Data Temperature Humidity Wind Speed Weather Condition
3 2024-01-04 37 62 2 Sunny
22 2024-01-13 39 76 14 Sunny
27 2024-01-28 37 51 16 Sunny
37 2024-02-07 37 69 4 Sunny
69 2024-02-29 32 58 19 Sunny
63 2024-03-03 37 72 13 Sunny
63 2024-03-04 33 31 0 Sunny
81 2024-03-14 32 36 19 Sunny
186 2024-04-14 36 69 18 Sunny
128 2024-04-30 33 66 2 Sunny
143 2024-05-21 31 38 15 Sunny
144 2024-05-24 35 44 11 Sunny
146 2024-05-26 32 88 16 Sunny
158 2024-06-07 32 44 3 Sunny
163 2024-06-12 37 54 6 Sunny
164 2024-06-13 32 33 16 Sunny
177 2024-06-26 31 62 9 Sunny
179 2024-06-28 36 66 11 Sunny
186 2024-07-05 32 39 4 Sunny
186 2024-07-15 35 67 17 Sunny
188 2024-07-17 35 48 5 Sunny
202 2024-07-21 31 45 19 Sunny
203 2024-07-22 38 58 2 Sunny
206 2024-07-24 31 78 7 Sunny
225 2024-08-13 32 54 3 Sunny
243 2024-08-29 35 55 12 Sunny
257 2024-09-14 31 73 17 Sunny
258 2024-09-15 31 47 6 Sunny
269 2024-09-26 34 55 9 Sunny
271 2024-09-28 38 46 9 Sunny
279 2024-10-05 37 51 13 Sunny
284 2024-10-11 39 41 14 Sunny
302 2024-10-29 33 77 6 Sunny
309 2024-11-05 36 35 0 Sunny
311 2024-11-07 24 31 0 Sunny
313 2024-11-09 37 70 6 Sunny
318 2024-11-14 39 69 4 Sunny
322 2024-11-18 33 45 12 Sunny
328 2024-11-24 32 80 3 Sunny
332 2024-11-28 33 49 9 Sunny
346 2024-12-11 36 71 7 Sunny
362 2024-12-28 39 34 19 Sunny
364 2024-12-30 38 34 18 Sunny

Weather of days with temperature above 30°C and sunny: 43
Weather Condition Average
0 Cloudy 59.57329
1 Rainy 58.345214
2 Sunny 58.547809

```

Figure 2: Output of Weather Data Analysis

1.3 Matplotlib usage

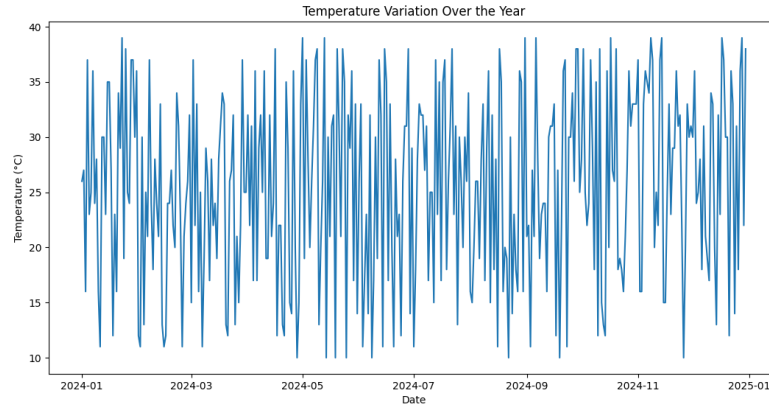


Figure 3: Weather Data Analysis

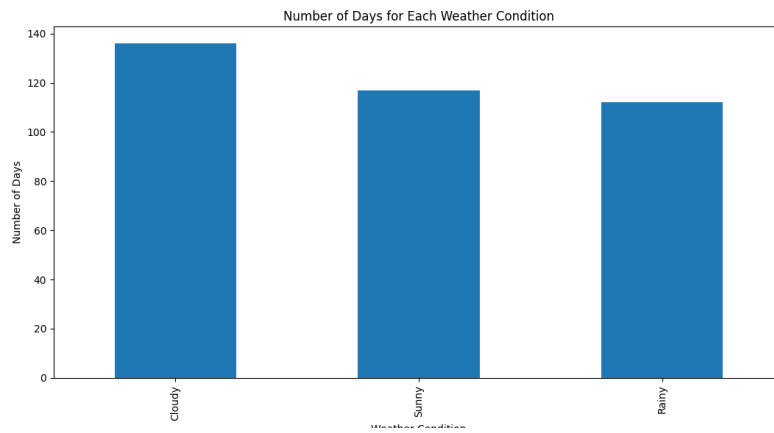


Figure 4: Weather Data Analysis

2 Sales Data Analysis

2.1 Code

```

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

np.random.seed(42)
order_ids = np.arange(1, 500)
products = np.random.choice(['Laptop', 'Headphones', 'Mouse', 'Keyboard', 'Monitor', 'Smartphone',
                             'Tablet', 'Camera', 'Smartwatch', 'Charger'], size=500)
prices = np.random.randint(50, 1000, size=500)
quantities = np.random.randint(1, 20, 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'Number of high value sales: {len(high_value_sales)}')
total_quantity_by_product = sales_data.groupby('Product')['Quantity'].sum().reset_index()
print(f'Total 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\Lab5\2022-CS-75> & C:\Users\jules\AppData\Local\Programs\Python\Python312\python.exe d:\Semester5\AILab\Lab5\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 18 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 7856
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 16918
4 5 Tablet 71 18 2023-05-28 18:27:28.133241 738
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 7856
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 16918
4 5 Tablet 71 18 2023-05-28 18:27:28.133241 738
...
494 495 Monitor 237 2 2024-09-29 18:27:28.133241 474
495 496 Laptop 120 3 2024-09-30 18:27:28.133241 360
496 497 Tablet 479 3 2024-10-01 18:27:28.133241 1437
497 498 Smartwatch 632 13 2024-10-03 18:27:28.133241 8216
498 499 Mouse 785 13 2024-10-04 18:27:28.133241 10205
499 500
[498 rows x 6 columns]
Total Quantity Sold by Product:
Product Quantity
0 Camera 521
1 Charger 579
2 Headphones 416
3 Keyboard 485
4 Laptop 658
5 Monitor 426
6 Mouse 579
7 Smartphone 406
8 Smartwatch 456
9 Tablet 564
PS D:\Semester5\AILab\Lab5\2022-CS-75>

```

Figure 6: Output of Sales Data Analysis

2.3 Matplotlib usage



Figure 7: Sales Data Analysis

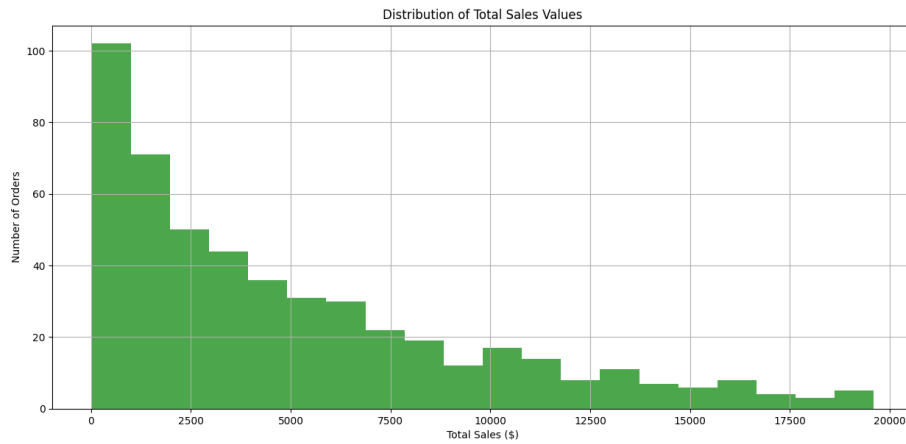


Figure 8: Sales Data Analysis

3 Employee Salary Analysis

3.1 Code

```

salaryProblem.py / ...
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:\Users\AILab\Lab5\2022-CS-75> C:\Users\AILab\AppData\Local\Programs\Python\Python112\python.exe d:\Users\AILab\Lab5\2022-CS-75\salaryProblem.py
Average Salary: $71461.72
Maximum Salary: $119992.34
Minimum Salary: $30213.55

Filtered Data (Employees with >5 years experience and salary > average):
  Employee ID  Name  Department  Salary  Years of Experience
0           1    G         HR    92723.256028             20
2           3    O         IT    96139.083526             10
3           4    K    Finance    113772.088798              8
4           5    H         IT    116283.236667              6
10          11    H         HR    88216.352558             11

Average Salary by Department:
Department
Finance    72791.461110
HR         78447.367865
IT         78861.818477
Marketing  74796.784762
Sales     76018.722774
Name: Salary, dtype: float64
PS D:\Users\AILab\Lab5\2022-CS-75>

```

Figure 10: Output of Employee Salary Analysis

3.3 Matplotlib usage

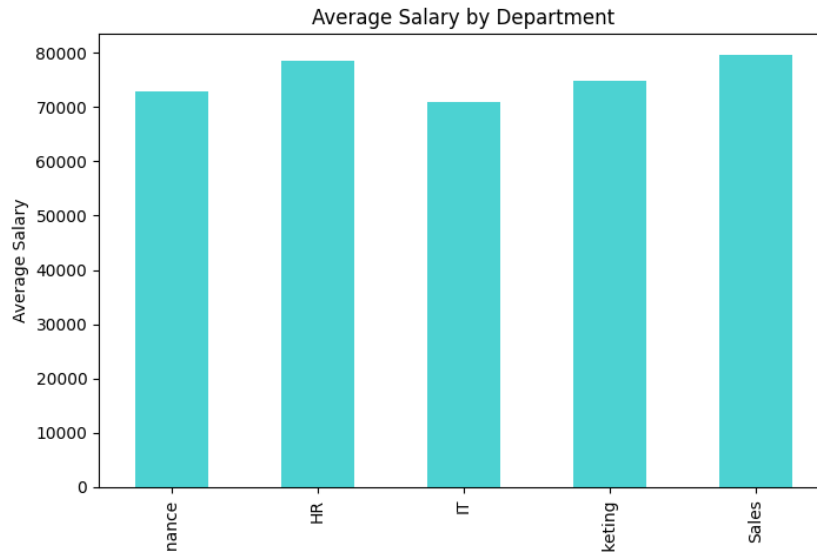


Figure 11: Employee Salary Analysis



Figure 12: Employee Salary Analysis

4 Exam Score Analysis

4.1 Code

```
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/guine/AppData/Local/Programs/Python/Python112/python.exe d:/Semester5/AILab/Lab5/2022-CS-75/examProblem.py
Mean Score: 48.24
Median Score: 48.00
Standard Deviation: 29.27
Number of students scoring above 80: 36

Average score by subject:
Subject
Biology      42.953488
Chemistry    56.387897
English      42.530608
Math         58.830811
Physics      44.105238
Name: Score, dtype: float64
PS D:\Semester5\AILab\Lab5\2022-CS-75>
```

Figure 14: Output of Exam Score Analysis

4.3 Matplotlib usage

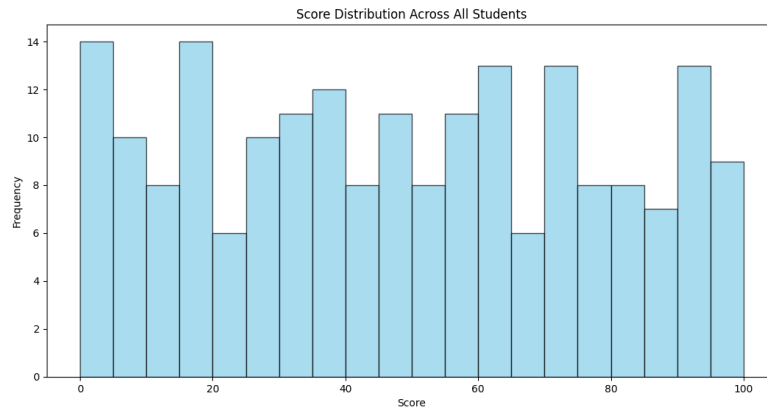


Figure 15: Exam Score Analysis

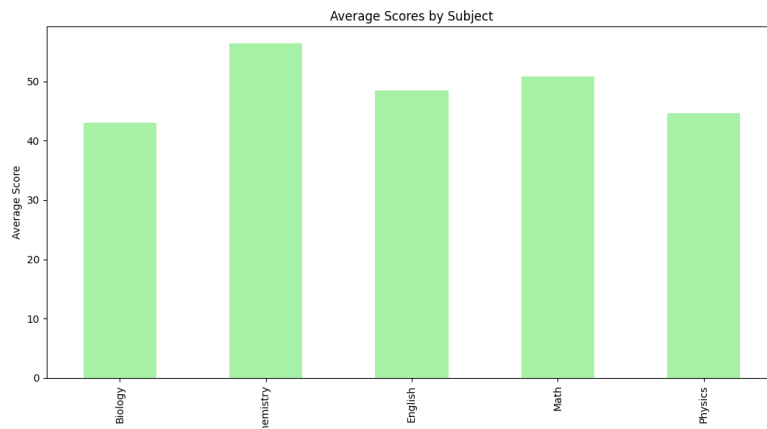


Figure 16: Exam Score Analysis

5 Stock Market Analysis

5.1 Code

```

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  2023-09-20 18:25:59.128924  CompanyE    368.257385    488.425629      940377          766.438958
8  2024-07-26 18:25:59.128924  CompanyE    348.858082    387.265483       493482          11.842581
9  2024-06-06 18:25:59.128924  CompanyE    381.399980    388.504889       812884          92.217248
14 2023-11-16 18:25:59.128924  CompanyD    493.178548    547.136224       751186       208.284743
15 2023-09-06 18:25:59.128924  CompanyE    427.564139    393.967698       801554        13.408886

Total Volume Traded by Company:
Company
CompanyA    385331742
CompanyB    89984723
CompanyC    97178446
CompanyD    382338826
CompanyE    183643489
Name: Volume Traded, dtype: int32
PS D:\Semester5\AILab\Lab5\2022-CS-75>

```

Figure 18: Output of Stock Market Analysis

5.3 Matplotlib usage

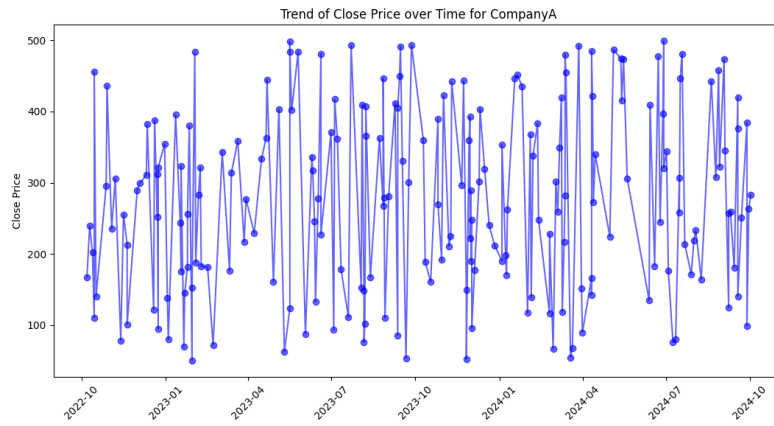


Figure 19: Exam Score Analysis

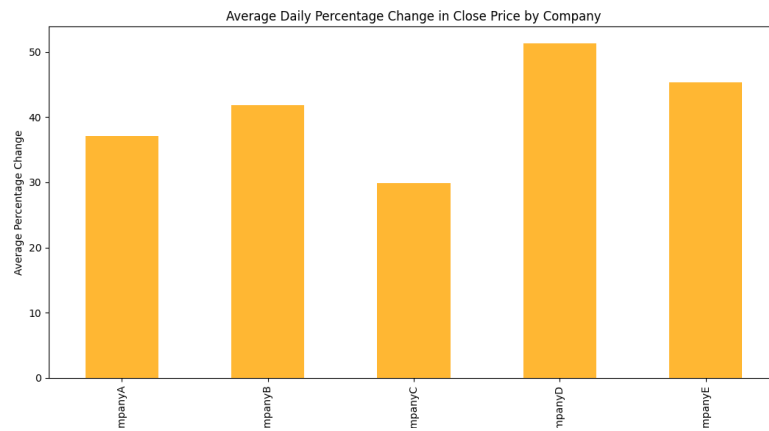


Figure 20: Stock Market Analysis

6 Customer Reviews Analysis

6.1 Code

```

1  import numpy as np
2  import pandas as pd
3  import matplotlib.pyplot as plt
4
5  # Path to your CSV file
6  file_path = r'D:\\Semester5\\AILab\\Lab5\\2022-CS-75\\Customer_support_data.csv'
7
8  # Load the data from the CSV file
9  df = pd.read_csv(file_path)
10
11 # Numpy Task
12 # Extract the 'CSAT Score' column
13 csat_scores = df['CSAT Score'].to_numpy()
14
15 # Calculate mean, median, and standard deviation
16 mean_csat = np.mean(csat_scores)
17 median_csat = np.median(csat_scores)
18 std_dev_csat = np.std(csat_scores)
19
20 # Display the results
21 print("Mean CSAT Score:", mean_csat)
22 print("Median CSAT Score:", median_csat)
23 print("Standard Deviation of CSAT Score:", std_dev_csat)
24
25 # Pandas Task
26 #1
27 filtered_df = df[df['CSAT Score'] > 4]
28 # Display the filtered data
29 print("filtered ", filtered_df)
30 #2
31 # Count the number of issues handled by each agent
32 agent_issue_count = df['Agent_name'].value_counts()
33 # Display the count of issues handled by each agent
34 print(agent_issue_count)
35 #3
36 # Group by 'category' and calculate the average CSAT Score
37 average_csat_by_category = df.groupby('category')['CSAT Score'].mean()
38 # Display the average CSAT Score for each category
39 print(average_csat_by_category)
40
41 # Matplot Task 1
42 # Plot a histogram for the 'CSAT Score'
43 plt.figure(figsize=(8, 6))
44 plt.hist(df['CSAT Score'], bins=5, edgecolor='black', color='skyblue')
45 plt.title('Distribution of CSAT Scores')
46 plt.xlabel('CSAT Score')
47 plt.ylabel('Frequency')
48 plt.grid(True)
49 # Display the histogram
50 plt.show()
51
52 # Matplot Task 2
53 # Group by 'category' and calculate the average CSAT Score
54 average_csat_by_category = df.groupby('category')['CSAT Score'].mean()
55 # Create a bar chart
56 plt.figure(figsize=(10, 6))
57 average_csat_by_category.plot(kind='bar', color='skyblue', edgecolor='black')
58 plt.title('Average CSAT Score by Category')
59 plt.xlabel('Category')
60 plt.ylabel('Average CSAT Score')
61 plt.xticks(rotation=45)
62 plt.grid(axis='y')
63 # Display the bar chart
64 plt.tight_layout()
65 plt.show()
66

```

Figure 21: Code of Customer Reviews Analysis

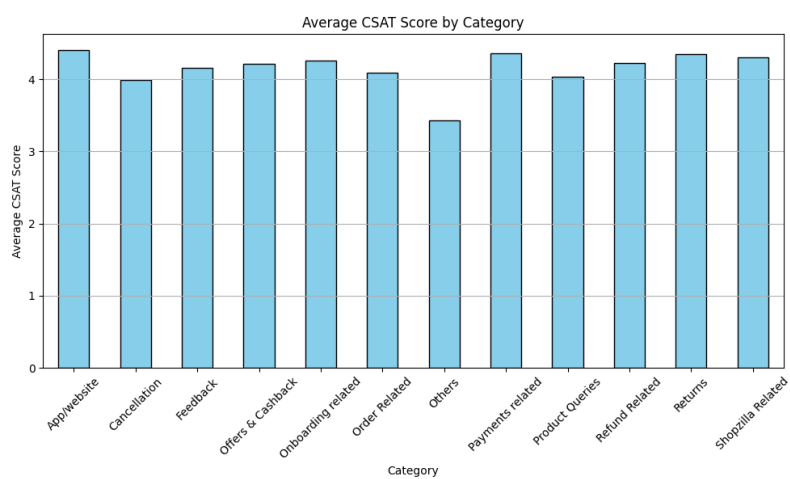


Figure 24: Customer Reviews Analysis