Numpy Pandas and Matplotlib Lab

Submitted to:

Mr. Nauman Shafi

Submitted by:

Faisal Ilyas 2022-CS-63

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

days_in_year = pd.date_range(start='2024-01-01', periods=365)
temps = np.random.randint(5, 38, size=365)
humidity_levels = np.random.randint(25, 85, size=365)
wind_speeds = np.random.randint(1, 25, size=365)
conditions = np.random.choice(['Clear', 'Stormy', 'Overcast'], size=365)

climate_data = pd.DataFrame({
    'Day': days_in_year,
    'Temp (°C)': temps,
    'Humidity (%)': humidity_levels,
    'Wind (km/h)': wind_speeds,
    'Condition': conditions
})

print(climate_data)

temp_array = climate_data['Temp (°C)'].to_numpy()
avg_temp = np.mean(temp_array)
median_temperature = np.median(temp_array)
temperature_std_dev = np.std(temp_array)

print(f'Average Temp: {avg_temp:.2f}°C')
print(f'Median Temp: {median_temperature:.2f}°C')
print(f'Standard Deviation of Temp: {temperature_std_dev:.2f}°C')

hot_clear_days = climate_data[(climate_data['Temp (°C)'] > 32) & (climate_data['Condition'] == 'Clear')]
hot_clear_day_count = hot_clear_days.shape[0]

print(hot_clear_days)
print(f"Number of clear days with temperature above 32°C: {hot_clear_day_count}")

avg_humidity_by_condition = climate_data.groupby('Condition')['Humidity (%)'].mean().reset_index()
avg_humidity_by_condition.columns = ['Condition', 'Avg Humidity']

print(avg_humidity_by_condition)

plt.figure(figsize=(10, 6))
plt.plot(climate_data['Day'], climate_data['Temp (°C)'], color='orange')
plt.title('Yearly Temperature Fluctuations')
plt.xlabel('Day of the Year')
plt.ylabel('Temperature (°C)')
plt.grid(True)
plt.show()

condition_counts = climate_data['Condition'].value_counts()

plt.figure(figsize=(8, 5))
condition_counts.plot(kind='bar', color='skyblue')
plt.title('Weather Condition Frequency Throughout the Year')
plt.xlabel('Condition')
plt.ylabel('Days Count')
plt.tight_layout()
plt.show()
```

Figure 1: Code of Weather Data Analysis

1.2 Console Output

```
          Day  Temp (°C)  Humidity (%)  Wind (km/h) Condition
0   2024-01-01         11            75           1     Clear
1   2024-01-02          5            27          21     Clear
2   2024-01-03         16            84          19  Overcast
3   2024-01-04         32            66          17    Stormy
4   2024-01-05         25            80           9  Overcast
..         ...        ...           ...         ...       ...
360 2024-12-26         18            73           6     Clear
361 2024-12-27         21            34          13    Stormy
362 2024-12-28         31            76          13     Clear
363 2024-12-29         15            35          15    Stormy
364 2024-12-30         23            47           5    Stormy

[365 rows x 5 columns]
Average Temp: 21.43°C
Median Temp: 21.00°C
Standard Deviation of Temp: 9.10°C
          Day  Temp (°C)  Humidity (%)  Wind (km/h) Condition
6   2024-01-07         33            72          21     Clear
68  2024-03-09         34            29           2     Clear
99  2024-04-09         36            44          11     Clear
118 2024-04-28         35            25           3     Clear
127 2024-05-07         33            76          17     Clear
138 2024-05-18         34            40           9     Clear
140 2024-05-20         37            26          20     Clear
161 2024-06-10         35            72           2     Clear
176 2024-06-25         37            72          19     Clear
188 2024-07-07         37            79          13     Clear
191 2024-07-10         34            53          11     Clear
231 2024-08-19         33            27          10     Clear
240 2024-08-28         35            34          12     Clear
249 2024-09-06         35            40          10     Clear
251 2024-09-08         34            25          15     Clear
273 2024-09-30         36            41          11     Clear
293 2024-10-20         34            84          23     Clear
296 2024-10-23         37            78          17     Clear
321 2024-11-17         35            83           2     Clear
332 2024-11-28         34            84          21     Clear
Number of clear days with temperature above 32°C: 20
  Condition  Avg Humidity
0     Clear     52.553571
1  Overcast     53.804196
2    Stormy     53.800000
```

Figure 2: Output of Weather Data Analysis

1.3 Matplotlib usage

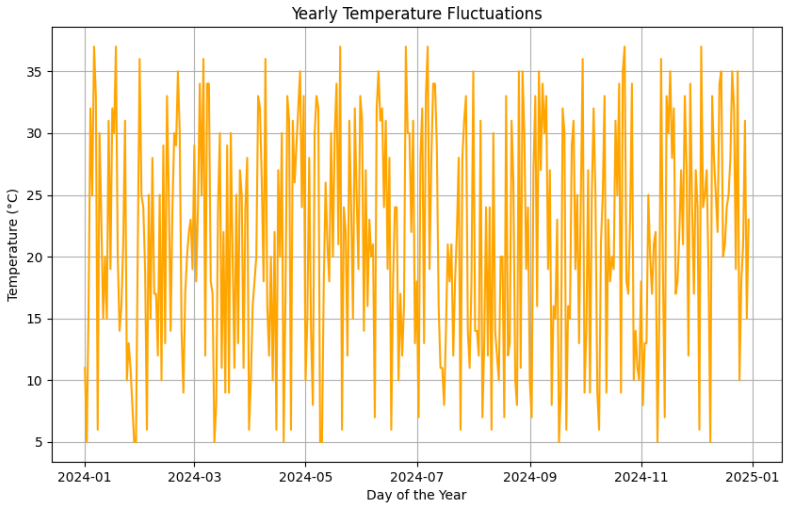


Figure 3: Weather Data Analysis

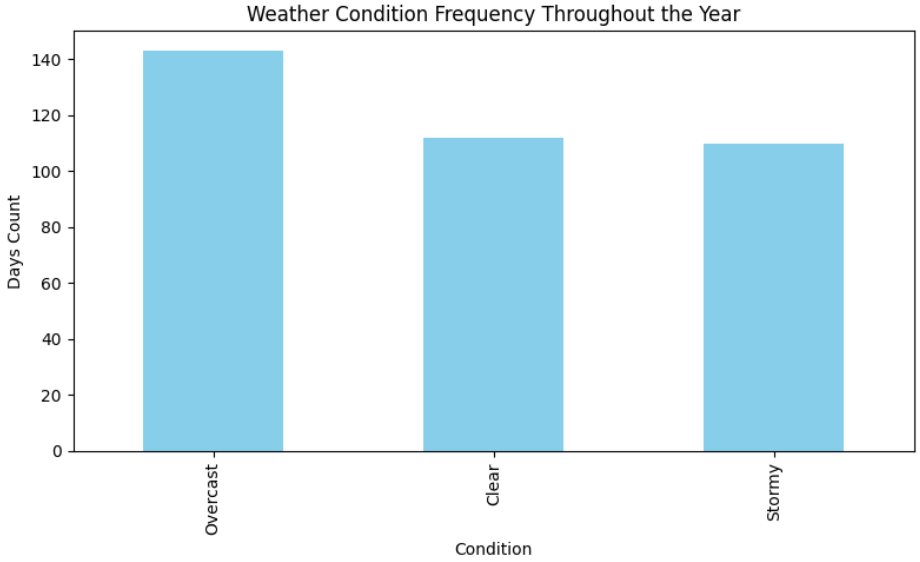


Figure 4: Weather Data Analysis

2 Sales Data Analysis

2.1 Code

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import random
from datetime import datetime, timedelta
np.random.seed(42)
products = ['Sofa', 'Dining Table', 'Chair', 'Coffee Table', 'Bed',
            'Bookshelf', 'Desk', 'Wardrobe', 'Nightstand', 'Armchair']
data = {
    'Order ID': np.arange(1, 501),
    'Product': np.random.choice(products, 500),
    'Price': np.random.uniform(50, 2000, 500),
    'Quantity': np.random.randint(1, 11, 500),
    'Purchase Date': [datetime.now() - timedelta(days=random.randint(1, 365)) for _ in range(500)]
}
df = pd.DataFrame(data)
price_quantity_matrix = df[['Price', 'Quantity']].to_numpy()
df['Total Sales'] = np.prod(price_quantity_matrix, axis=1)
sales_above_100 = df[df['Total Sales'] > 100]
quantity_per_product = df.groupby('Product')['Quantity'].sum()
plt.figure(figsize=(8, 5))
plt.scatter(df['Price'], df['Quantity'], alpha=0.6, color='b')
plt.title('Price vs. Quantity Sold')
plt.xlabel('Price ($)')
plt.ylabel('Quantity Sold')
plt.show()
plt.figure(figsize=(8, 5))
plt.hist(df['Total Sales'], bins=20, color='g', alpha=0.7)
plt.title('Distribution of Total Sales')
plt.xlabel('Total Sales ($)')
plt.ylabel('Frequency')
plt.show()
```

Figure 5: Code of Sales Data Analysis

2.2 Console Output

```
First 5 rows of the DataFrame:
   Order ID       Product       Price  Quantity             Purchase Date  Total Sales
0         1          Desk  251.091544         7 2024-01-01 18:34:55.503703  1757.640810
1         2  Coffee Table  1809.978168         5 2023-12-14 18:34:55.503703  9049.890840
2         3     Wardrobe  1035.242126         3 2024-06-15 18:34:55.503703  3105.726379
3         4           Bed  1661.592059         3 2023-11-04 18:34:55.503703  4984.776177
4         5          Desk   674.096722         4 2023-10-29 18:34:55.503703  2696.386888

Summary Statistics of Total Sales:
count      500.000000
mean      5427.437055
std       4618.777892
min         71.133420
25%       1754.885084
50%       3889.384224
75%       7977.429482
max      19704.483380
Name: Total Sales, dtype: float64

Sales above $100:
     Order ID       Product       Price  Quantity             Purchase Date   Total Sales
0           1          Desk  251.091544         7 2024-01-01 18:34:55.503703   1757.640810
1           2  Coffee Table  1809.978168         5 2023-12-14 18:34:55.503703   9049.890840
2           3     Wardrobe  1035.242126         3 2024-06-15 18:34:55.503703   3105.726379
3           4           Bed  1661.592059         3 2023-11-04 18:34:55.503703   4984.776177
4           5          Desk   674.096722         4 2023-10-29 18:34:55.503703   2696.386888
..        ...           ...          ...       ...                       ...           ...
495       496          Sofa   449.719376         6 2024-08-17 18:34:55.507212   2698.316256
496       497          Desk   621.638874         9 2023-12-29 18:34:55.507212   5594.742666
497       498          Desk  1797.854846         7 2024-07-04 18:34:55.507212  12584.983923
498       499    Nightstand    75.353751         9 2024-09-17 18:34:55.507212    678.183758
499       500         Chair   216.741635         2 2024-01-24 18:34:55.507212    433.483270

[488 rows x 6 columns]

Total Quantity Sold per Product:
Product
Armchair        283
Bed             296
Bookshelf       230
Chair           305
Coffee Table    267
Desk            264
Dining Table    242
Nightstand      271
Sofa            324
Wardrobe        262
Name: Quantity, dtype: int32
```

Figure 6: Output of Sales Data Analysis

2.3 Matplotlib usage

Figure 7: Sales Data Analysis

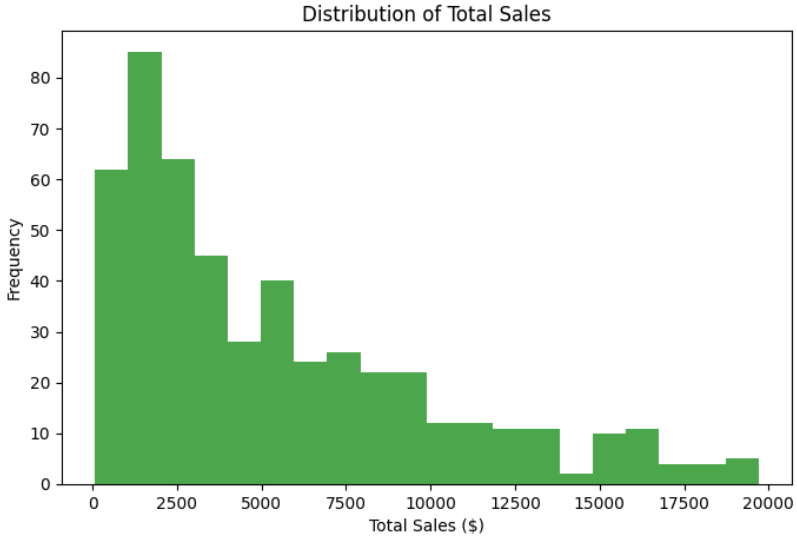


Figure 8: Sales Data Analysis

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)

employee_names = [ 'Zainab','Ayesha', 'Ahmed', 'Hassan', 'Fatima', 'Ali', 'Sara', 'Usman', 'Amina', 'Bilal',
                   'Hina', 'Farhan', 'Rabia', 'Kamran', 'Asma', 'Tariq', 'Nida', 'Sana', 'Shahid', 'Naveed']

departments_list = ['Computer Science', 'Mathematics', 'Physics', 'Chemistry', 'Biology', 'Economics',
                    'Electrical Engineering', 'Mechanical Engineering', 'Civil Engineering', 'Business Administration']

data = {
    'Employee ID': np.arange(1, 301),
    'Employee Name': np.random.choice(employee_names, 300),
    'Department': np.random.choice(departments_list, 300),
    'Annual Salary': np.random.uniform(30000, 120000, 300),
    'Experience (Years)': np.random.randint(1, 26, 300)
}

df = pd.DataFrame(data)

salary_array = df['Annual Salary'].to_numpy()

avg_salary = np.mean(salary_array)
highest_salary = np.max(salary_array)
lowest_salary = np.min(salary_array)

experienced_high_salary_employees = df[(df['Experience (Years)'] > 5) & (df['Annual Salary'] > avg_salary)]

avg_salary_by_department = df.groupby('Department')['Annual Salary'].mean()

plt.figure(figsize=(8, 5))
avg_salary_by_department.plot(kind='bar', color='teal', alpha=0.7)
plt.title('Average Annual Salary by Department')
plt.xlabel('Department')
plt.ylabel('Average Salary ($)')
plt.show()

plt.figure(figsize=(8, 5))
plt.plot(df['Experience (Years)'], df['Annual Salary'], marker='o', linestyle='-', color='blue', alpha=0.6)
plt.title('Annual Salary vs. Years of Experience')
plt.xlabel('Years of Experience')
plt.ylabel('Annual Salary ($)')
plt.show()


print("Summary Statistics of Employee Salaries:")
print(f"Average Salary: ${avg_salary:.2f}")
print(f"Highest Salary: ${highest_salary:.2f}")
print(f"Lowest Salary: ${lowest_salary:.2f}")

print("\nEmployees with more than 5 years of experience and earning above average salary:")
print(experienced_high_salary_employees[['Employee ID', 'Employee Name', 'Department', 'Annual Salary', 'Experience (Years)']].head())

print("\nAverage Salary by Department:")
print(avg_salary_by_department)
```

Figure 9: Code of Employee Salary Analysis

3.2 Console Output

```
Summary Statistics of Employee Salaries:
Average Salary: $75505.58
Highest Salary: $119792.24
Lowest Salary: $30233.55

Employees with more than 5 years of experience and earning above average salary:
    Employee ID Employee Name              Department  Annual Salary  Experience (Years)
1             2        Naveed  Business Administration   88216.352550                  25
15           16        Ayesha   Mechanical Engineering  114647.864442                  10
17           18        Farhan        Civil Engineering   82304.635906                   9
18           19        Farhan                Economics  112725.889665                  22
20           21         Bilal  Business Administration  108899.533733                  14

Average Salary by Department:
Department
Biology                    75069.644470
Business Administration    72361.511849
Chemistry                  74883.395828
Civil Engineering          75456.614471
Computer Science           76533.635524
Economics                  78304.660551
Electrical Engineering     77710.952227
Mathematics                77282.685688
Mechanical Engineering     73954.318939
Physics                    75388.044244
Name: Annual Salary, dtype: float64
```

Figure 10: Output of Employee Salary Analysis

3.3 Matplotlib usage

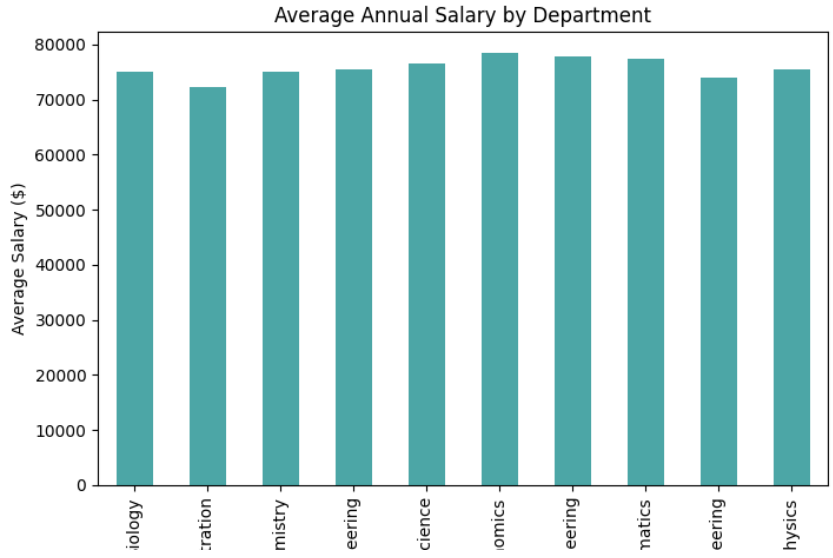
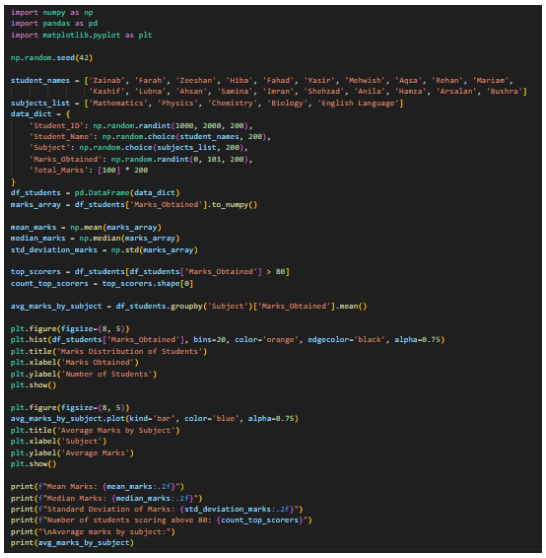


Figure 11: Employee Salary Analysis

Figure 12: Employee Salary Analysis

4 Exam Score Analysis

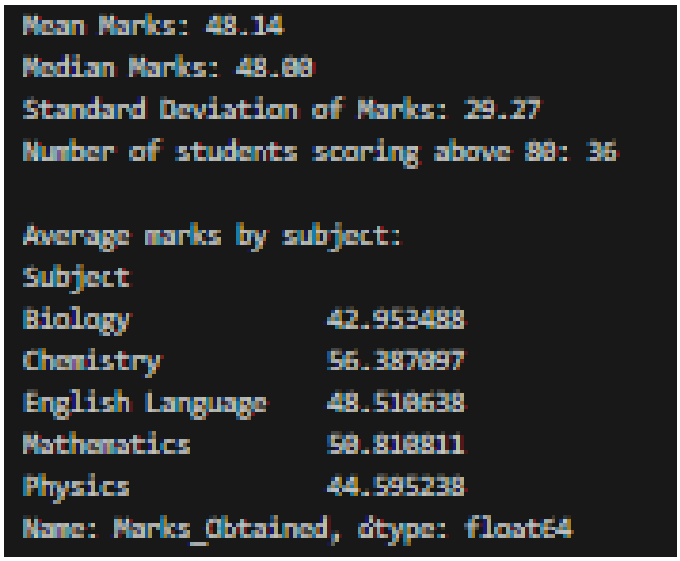
4.1 Code



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

np.random.seed(42)

student_names = ['Zainab', 'Farah', 'Zeeshan', 'Hiba', 'Fahad', 'Yasir', 'Mehwish', 'Aqsa', 'Rehan', 'Mariam',
                 'Kashif', 'Lubna', 'Ahsan', 'Samina', 'Imran', 'Shehzad', 'Anila', 'Hamza', 'Arsalan', 'Bushra']
subjects_list = ['Mathematics', 'Physics', 'Chemistry', 'Biology', 'English Language']
data_dict = {
    'Student_ID': np.random.randint(1000, 2000, 200),
    'Student_Name': np.random.choice(student_names, 200),
    'Subject': np.random.choice(subjects_list, 200),
    'Marks_Obtained': np.random.randint(0, 101, 200),
    'Total_Marks': [100] * 200
}
df_students = pd.DataFrame(data_dict)
marks_array = df_students['Marks_Obtained'].to_numpy()

mean_marks = np.mean(marks_array)
median_marks = np.median(marks_array)
std_deviation_marks = np.std(marks_array)

top_scorers = df_students[df_students['Marks_Obtained'] > 80]
count_top_scorers = top_scorers.shape[0]

avg_marks_by_subject = df_students.groupby('Subject')['Marks_Obtained'].mean()

plt.figure(figsize=(8, 5))
plt.hist(df_students['Marks_Obtained'], bins=20, color='orange', edgecolor='black', alpha=0.75)
plt.title('Marks Distribution of Students')
plt.xlabel('Marks Obtained')
plt.ylabel('Number of Students')
plt.show()

plt.figure(figsize=(8, 5))
avg_marks_by_subject.plot(kind='bar', color='blue', alpha=0.75)
plt.title('Average Marks by Subject')
plt.xlabel('Subject')
plt.ylabel('Average Marks')
plt.show()

print(f"Mean Marks: {mean_marks:.2f}")
print(f"Median Marks: {median_marks:.2f}")
print(f"Standard Deviation of Marks: {std_deviation_marks:.2f}")
print(f"Number of students scoring above 80: {count_top_scorers}")
print("\nAverage marks by subject:")
print(avg_marks_by_subject)
```

Figure 13: Code of Exam Score Analysis

4.2 Console Output



```
Mean Marks: 48.14
Median Marks: 48.00
Standard Deviation of Marks: 29.27
Number of students scoring above 80: 36

Average marks by subject:
Subject
Biology             42.953488
Chemistry           56.387097
English Language    48.518638
Mathematics         50.818811
Physics             44.595238
Name: Marks_Obtained, dtype: float64
```

Figure 14: Output of Exam Score Analysis

4.3 Matplotlib usage

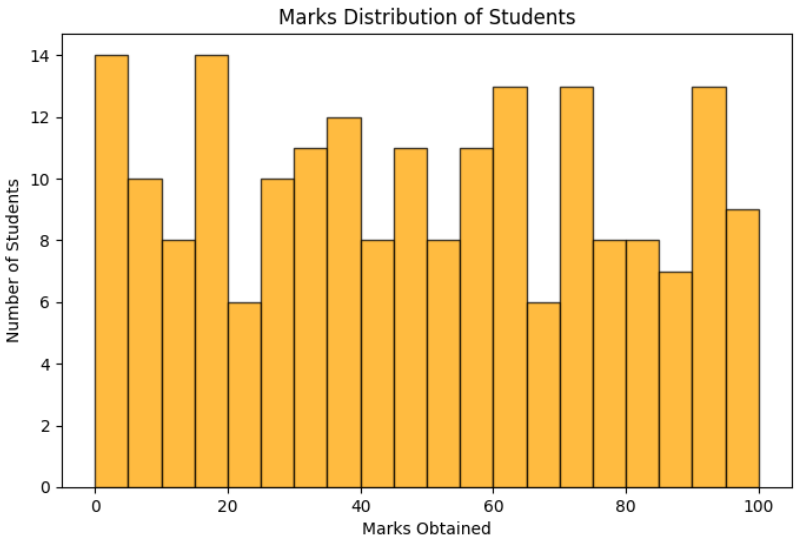


Figure 15: Exam Score Analysis

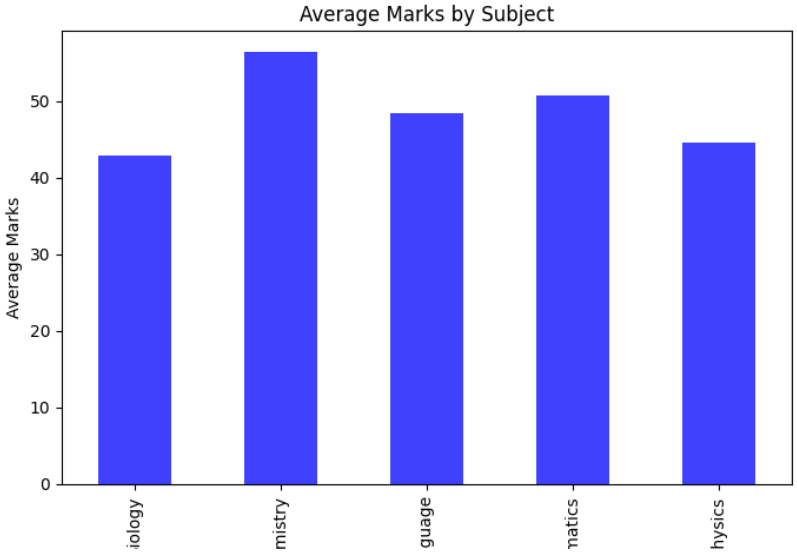


Figure 16: Exam Score Analysis

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_range = [start_date + timedelta(days=random.randint(0, 730)) for _ in range(1000)]
companies_list = ['CorpA', 'CorpB', 'CorpC', 'CorpD', 'CorpZE']
market_data = {
    'Trade Date': date_range,
    'Corporation': np.random.choice(companies_list, 1000),
    'Opening Price': np.random.uniform(60, 450, 1000),
    'Closing Price': np.random.uniform(60, 450, 1000),
    'Trade Volume': np.random.randint(2000, 1500000, 1000)
}
df_stocks = pd.DataFrame(market_data)

closing_prices = df_stocks['Closing Price'].to_numpy()
percent_change = np.diff(closing_prices) / closing_prices[:-1] * 100
df_stocks['Percent Change Daily'] = np.insert(percent_change, 0, 0)

high_change_df = df_stocks[df_stocks['Percent Change Daily'] > 2]
total_traded_volume = df_stocks.groupby('Corporation')['Trade Volume'].sum()

selected_company = 'CorpX'
company_prices = df_stocks[df_stocks['Corporation'] == selected_company].sort_values('Trade Date')

plt.figure(figsize=(10, 6))
plt.plot(company_prices['Trade Date'], company_prices['Closing Price'], marker='x', color='g', alpha=0.7)
plt.title(f'Closing Price Trend for {selected_company}')
plt.xlabel('Trade Date')
plt.ylabel('Closing Price')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

avg_daily_change = df_stocks.groupby('Corporation')['Percent Change Daily'].mean()

plt.figure(figsize=(8, 5))
avg_daily_change.plot(kind='bar', color='purple', alpha=0.85)
plt.title('Average Daily Percent Change per Corporation')
plt.xlabel('Corporation')
plt.ylabel('Average Daily Change (%)')
plt.tight_layout()
plt.show()

print("\nStock price increase by more than 2% on the following days:")
print(high_change_df.head())
print("\nTotal Volume Traded per Corporation:")
print(total_traded_volume)
```

Figure 17: Code of Stock Market Analysis

5.2 Console Output

```
Stock price increase by more than 2% on the following days:
              Trade Date Corporation  Opening Price  Closing Price  Trade Volume  Percent Change Daily
4  2024-09-23 18:39:38.155806      CorpZE     335.823067     433.035545        950377            569.066936
8  2024-04-10 18:39:38.155806       CorpC     225.410962     196.384260        404402             11.885019
9  2023-08-23 18:39:38.155806      CorpZE     347.213252     362.114877        813804             84.390988
14 2022-10-06 18:39:38.155806       CorpD     444.088068     317.518061        752186            222.765244
15 2024-05-08 18:39:38.155806       CorpB     387.170254     358.105338        802554             12.782667

Total Volume Traded per Corporation:
Corporation
CorpA     153070457
CorpB     143364357
CorpC     139023775
CorpD     151972438
CorpZE    157680630
Name: Trade Volume, dtype: int32
```

Figure 18: Output of Stock Market Analysis

5.3 Matplotlib usage

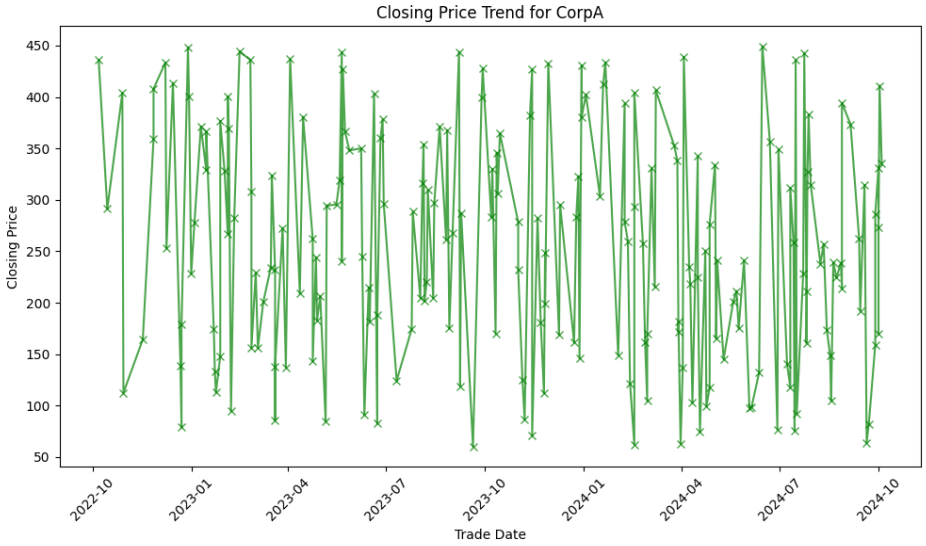


Figure 19: Exam Score Analysis

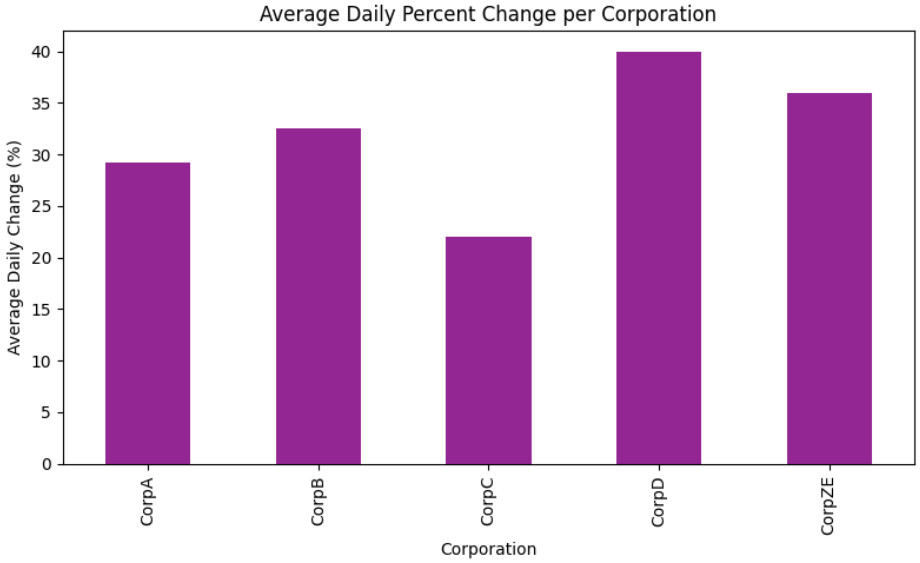


Figure 20: Stock Market Analysis

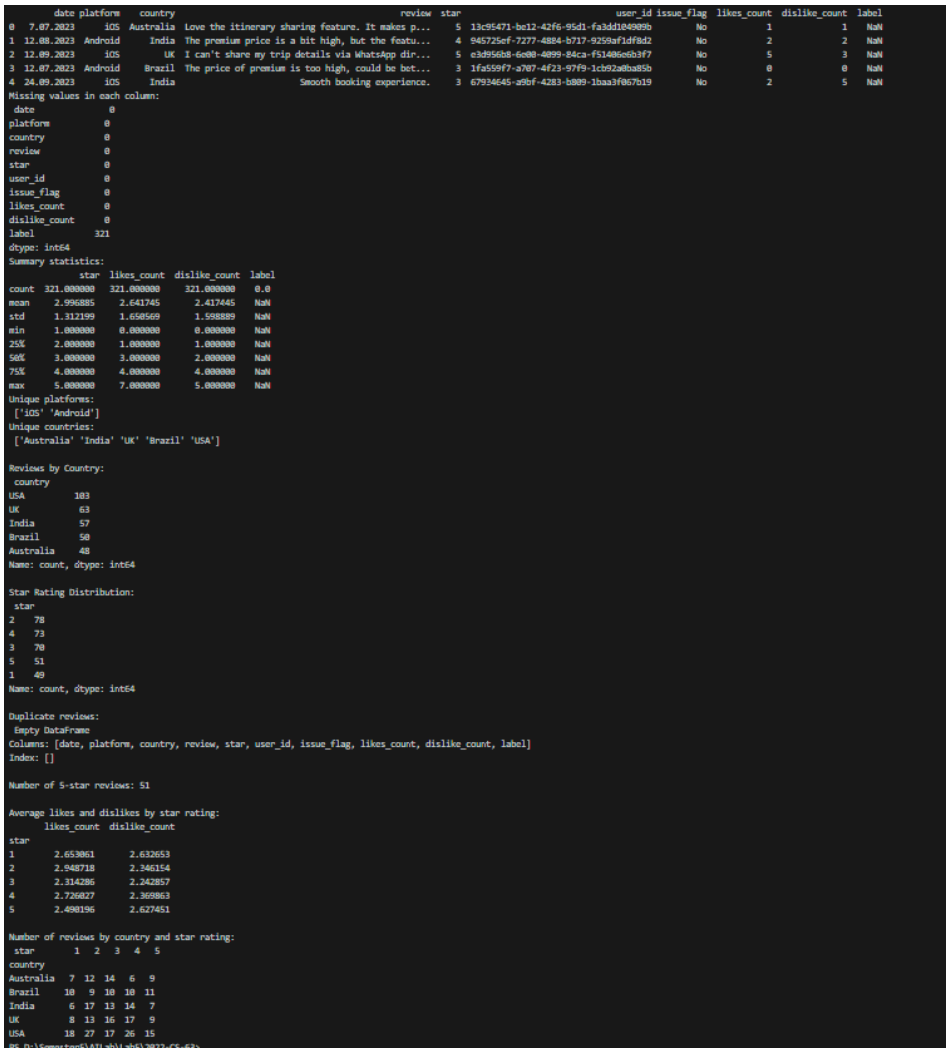
6 App-store reviews Analysis

6.1 Code

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

file_path = 'app_store_reviews.csv'
df = pd.read_csv(file_path, delimiter=';')

print(df.head())

#  Basic operations
print("Missing values in each column:\n", df.isnull().sum())

print("Summary statistics:\n", df.describe())

print("Unique platforms:\n", df['platform'].unique())
print("Unique countries:\n", df['country'].unique())

country_reviews = df['country'].value_counts()
print("\nReviews by Country:\n", country_reviews)

star_distribution = df['star'].value_counts()
print("\nStar Rating Distribution:\n", star_distribution)

df['date'] = pd.to_datetime(df['date'], format='%d.%m.%Y')

duplicates = df[df.duplicated(subset=['user_id', 'review'], keep=False)]
print("\nDuplicate reviews:\n", duplicates)

df = df.drop_duplicates(subset=['user_id', 'review'])


# Analysis

five_star_reviews = df[df['star'] == 5]
print("\nNumber of 5-star reviews:", five_star_reviews.shape[0])

avg_likes_dislikes = df.groupby('star')[['likes_count', 'dislike_count']].mean()
print("\nAverage likes and dislikes by star rating:\n", avg_likes_dislikes)

country_star_reviews = df.groupby(['country', 'star']).size().unstack(fill_value=0)
print("\nNumber of reviews by country and star rating:\n", country_star_reviews)


# Visualizations

plt.figure(figsize=(10, 6))
country_reviews.plot(kind='bar', color='skyblue')
plt.title('Number of Reviews by Country')
plt.xlabel('Country')
plt.ylabel('Number of Reviews')
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()


plt.figure(figsize=(8, 6))
star_distribution.plot(kind='bar', color='coral')
plt.title('Star Rating Distribution')
plt.xlabel('Star Rating')
plt.ylabel('Number of Reviews')
plt.tight_layout()
plt.show()



plt.figure(figsize=(10, 6))
plt.plot(avg_likes_dislikes.index, avg_likes_dislikes['likes_count'], marker='o', color='green', label='Average Likes')
plt.plot(avg_likes_dislikes.index, avg_likes_dislikes['dislike_count'], marker='o', color='red', label='Average Dislikes')
plt.title('Average Likes and Dislikes by Star Rating')
plt.xlabel('Star Rating')
plt.ylabel('Average Count')
plt.legend()
plt.tight_layout()
plt.show()

```

Figure 21: Code of App-store reviews Analysis

6.2 Console Output



```
         date platform    country                                          review  star                               user_id issue_flag  likes_count  dislike_count  label
0   7.07.2023      iOS  Australia  Love the itinerary sharing feature. It makes p...     5  13c95471-be12-42f6-95d1-fa3dd104909b         No            1              1    NaN
1  12.08.2023  Android      India  The premium price is a bit high, but the featu...     4  945725ef-7277-4884-b717-9259af1df8d2         No            2              2    NaN
2  12.09.2023      iOS         UK  I can't share my trip details via WhatsApp dir...     5  e3d956b8-6e00-4099-84ca-f51406e6b3f7         No            5              3    NaN
3  12.07.2023  Android     Brazil  The price of premium is too high, could be bet...     3  1fa559f7-a707-4f23-97f9-1cb92a0ba85b         No            0              0    NaN
4  24.09.2023      iOS      India                        Smooth booking experience.     3  67934645-a9bf-4283-b809-1baa3f067b19         No            2              5    NaN
Missing values in each column:
 date              0
platform          0
country           0
review            0
star              0
user_id           0
issue_flag        0
likes_count       0
dislike_count     0
label           321
dtype: int64
Summary statistics:
             star  likes_count  dislike_count  label
count  321.000000   321.000000     321.000000    0.0
mean     2.996885     2.641745       2.417445    NaN
std      1.312199     1.658569       1.598889    NaN
min      1.000000     0.000000       0.000000    NaN
25%      2.000000     1.000000       1.000000    NaN
50%      3.000000     3.000000       2.000000    NaN
75%      4.000000     4.000000       4.000000    NaN
max      5.000000     7.000000       5.000000    NaN
Unique platforms:
 ['iOS' 'Android']
Unique countries:
 ['Australia' 'India' 'UK' 'Brazil' 'USA']

Reviews by Country:
 country
USA          103
UK            63
India         57
Brazil        50
Australia     48
Name: count, dtype: int64

Star Rating Distribution:
 star
2    78
4    73
3    70
5    51
1    49
Name: count, dtype: int64

Duplicate reviews:
 Empty DataFrame
Columns: [date, platform, country, review, star, user_id, issue_flag, likes_count, dislike_count, label]
Index: []

Number of 5-star reviews: 51

Average likes and dislikes by star rating:
       likes_count  dislike_count
star
1         2.653061       2.632653
2         2.948718       2.346154
3         2.314286       2.242857
4         2.726027       2.369863
5         2.490196       2.627451

Number of reviews by country and star rating:
 star       1   2   3   4   5
country
Australia   7  12  14   6   9
Brazil     10   9  10  10  11
India       6  17  13  14   7
UK          8  13  16  17   9
USA        18  27  17  26  15
```

Figure 22: Output of App-store reviews Analysis

6.3 Matplotlib usage

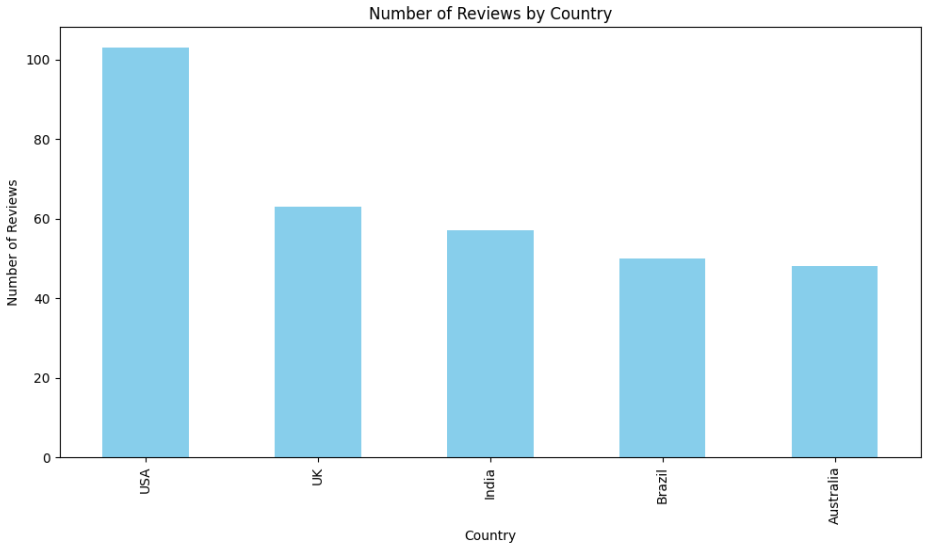


Figure 23: App-store reviews Analysis

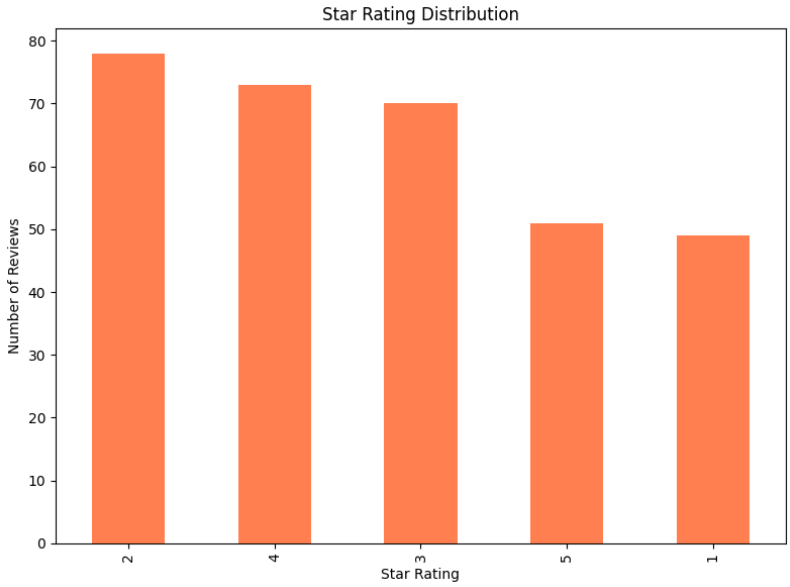


Figure 24: App-store reviews Analysis

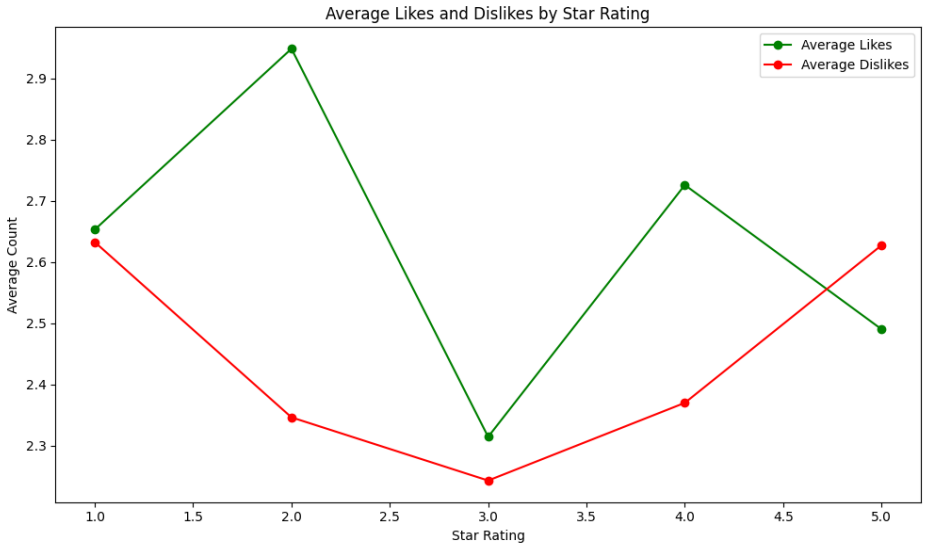


Figure 25: App-store reviews Analysis