AI Lab Week 5

Session: 2022 – 2026

Submitted by:

Muhammad Saad Akmal 2022-CS-148

Submitted To:

Sir Nauman Shafi

Department of Computer Science

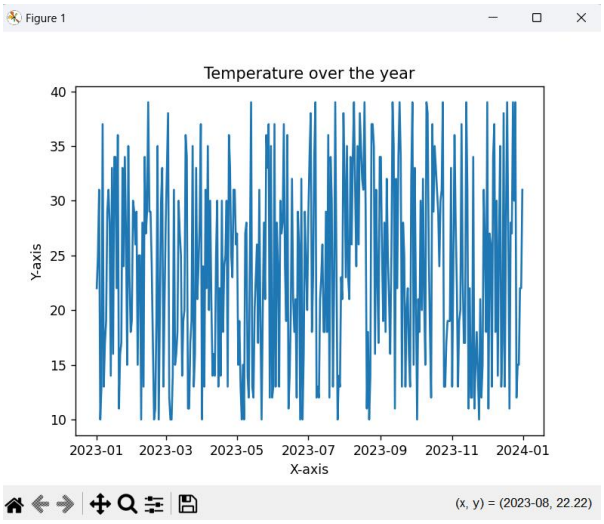
University of Engineering and Technology

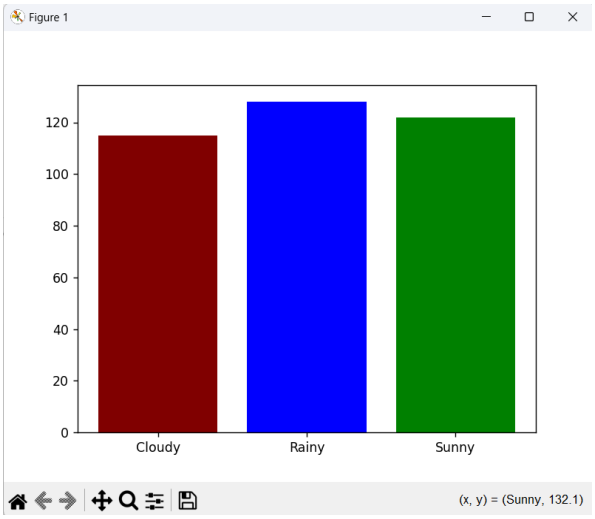
Lahore Pakistan

Case-Study 1:

```
import pandas as pd
import numpy as np
import matplotlib.pylab as plt

np.random.seed(0)


days = 365

temperature = np.random.randint(10, 40, days)
humidity = np.random.randint(30, 90, days)
wind_speed = np.random.randint(0, 20, days)
weather_condition = np.random.choice(['Sunny', 'Rainy', 'Cloudy'], days)

weather_data_gen = pd.DataFrame({
    'Date': pd.date_range(start='2023-01-01', periods=days, freq='D'),
    'Temperature': temperature,
    'Humidity': humidity,
    'Wind Speed': wind_speed,
    'Weather Condition': weather_condition
})

weather_data_gen.to_excel('weather_data.xlsx', index=False)

weather_data = pd.read_excel('weather_data.xlsx')

temp = weather_data['Temperature']

numpy_array = np.array(temp)

mean = np.mean(numpy_array)
median = np.median(numpy_array)
std = np.std(numpy_array)

print(f'Mean: {mean}')
print(f'Meadian: {median}')
print(f'Standard Deviation: {std}')


filtered = weather_data[(weather_data['Temperature'] > 30) & (weather_data['Weather Condition'] == 'Sunny')]
row_count = len(filtered)
print(f'Rows with temp > 30 and wether sunny: {row_count}')

grouped = weather_data.groupby('Weather Condition')
average_humidity = grouped['Humidity'].mean()
print(f'Average Humidity: {average_humidity}')

temperatures = weather_data['Temperature']
days = weather_data['Date']

temp_array = np.array(temperatures)
days_array = np.array(days)

plt.plot(days_array , temp_array)
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.title("Temperature over the year")
plt.show()


weather_condition = weather_data.groupby('Weather Condition')['Weather Condition'].count()

conditions = weather_condition.index
counts = weather_condition.values

plt.bar(conditions, counts, color =['maroon' , 'blue' , 'green'])
plt.show()
```

Output:






```
Mean: 23.70958904109589
Meadian: 24.0
Standard Deviation: 8.791218600890723
Rows with temp > 30 and wether sunny: 37
Average Humidity: Weather Condition
Cloudy    59.452174
Rainy     57.796875
Sunny     60.877049
```

Case-Study 2:

```
import pandas as pd
import numpy as np
import matplotlib.pylab as plt

np.random.seed(0)


rows = 500

products_array = ['Tea' , 'Coffee' , 'Drink' , 'Sugar' , 'Cake' , 'Pizza' , 'Burger' , 'Pasta' , 'Shawarma' , 'Wrap']
products = np.random.choice(products_array , rows)
prices = np.random.randint(10, 1000, rows)
quantity = np.random.randint(1, 20, rows)
date_of_purchase = pd.date_range(start='2024-01-01', periods=rows)


sales_data_gen = pd.DataFrame({
    'products' : products,
    'prices' : prices,
    'quantity' : quantity,
    'date_of_purchase' : date_of_purchase
})

sales_data_gen.to_excel('sales_data.xlsx', index=False)

sales_data = pd.read_excel('sales_data.xlsx')

prices = np.array(sales_data['prices'])
quantity = np.array(sales_data['quantity'])

total_sales = np.multiply(prices , quantity)
print(f'Total sales {total_sales}')

sales_data['total_sales'] = total_sales


greater_sales = sales_data[(sales_data['total_sales'] > 100)]
print(greater_sales)


total_quantity_sales = sales_data.groupby('products')['quantity'].sum()
print(total_quantity_sales)


plt.scatter(prices, quantity)
plt.show()


plt.hist(np.array(sales_data['total_sales']))
plt.show()
```

Output:

```
      products  prices  quantity date_of_purchase  total_sales
0       Pizza     424         6       2024-01-01         2544
1         Tea     537        16       2024-01-02         8592
2       Sugar     179        14       2024-01-03         2506
3       Sugar     176        12       2024-01-04         2112
4       Pasta     319        12       2024-01-05         3828
..        ...     ...       ...              ...          ...
495  Shawarma     923         6       2025-05-10         5538
496       Tea     986         3       2025-05-11         2958
497     Pasta     890        18       2025-05-12        16020
498    Burger      53        16       2025-05-13          848
499     Pizza     120         4       2025-05-14          480

[494 rows x 5 columns]
products
Burger      432
Cake        549
Coffee      423
Drink       442
Pasta       545
Pizza       475
Shawarma    538
Sugar       706
Tea         496
Wrap        403
Name: quantity, dtype: int64
```

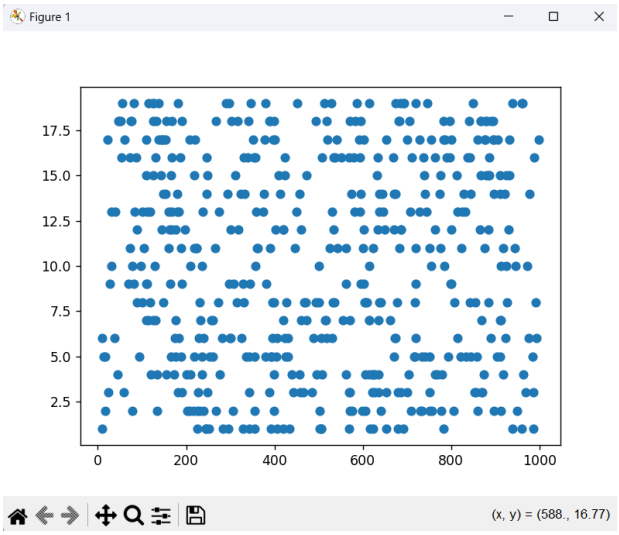


Figure 1
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0
200
400
600
800
1000
(x, y) = (588., 16.77)

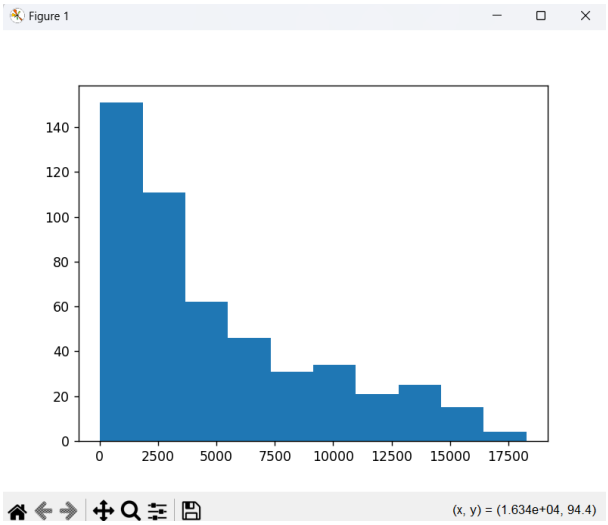
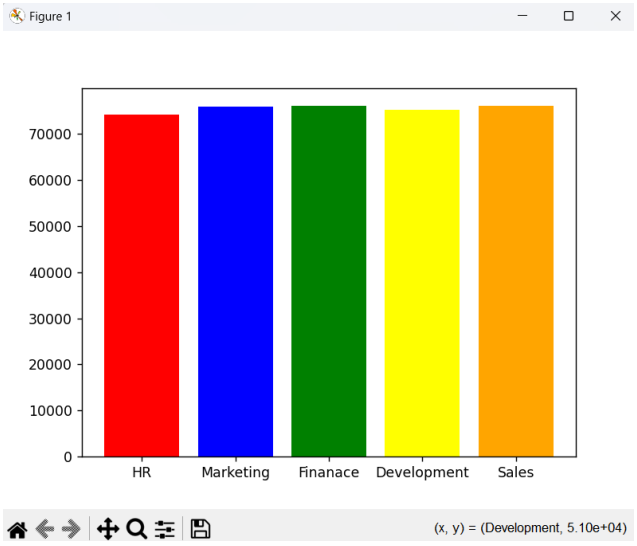


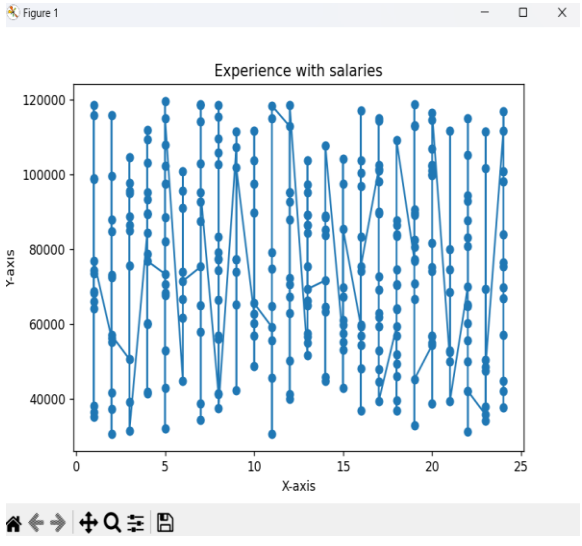
Figure 1
140
120
100
80
60
40
20
0
0
2500
5000
7500
10000
12500
15000
17500
(x, y) = (1.634e+04, 94.4)

Case-Study 3:

```
import pandas as pd
import numpy as np
import matplotlib.pylab as plt

np.random.seed(0)


rows = 300
names = ['Ahmed', 'Sarah', 'Ali', 'Fatima', 'Zain', 'Aisha', 'Omar', 'Noor', 'Hassan', 'Maria', 'Bilal', 'Zara', 'Saad', 'Sana', 'Imran', 'Layla', 'Usman', 'Hira', 'Yasir', 'Iman']
department = ['HR' , 'Marketing' , 'Finanace' , 'Development' , 'Sales']

employe_id = [i for i in range(rows)]

employee_names = np.random.choice(names , rows)
employee_department = np.random.choice(department , rows)

salary = np.random.randint(30000, 120000, rows)
experience = np.random.randint(1, 25, rows)


employee_data_gen = pd.DataFrame({
    'employee_id' : employe_id,
    'employe_name' : employee_names,
    'employee_department' : employee_department,
    'salary' : salary,
    'exprience' : experience
})

employee_data_gen.to_excel('employee_data.xlsx', index=False)

employee_data = pd.read_excel('employee_data.xlsx')

salary_array = np.array(employee_data['salary'])

average_salary = salary_array.mean()
max_salary = salary_array.max()
min_salary = salary_array.min()

print(f'Average salary: {average_salary}')
print(f'Max salary: {max_salary}')
print(f'Min salary: {min_salary}')


filtered_employees = employee_data[(employee_data['exprience'] > 5) & (employee_data['salary'] > average_salary)]
print(f'Filtered employees {filtered_employees}')


department_group = employee_data.groupby('employee_department')
department_salary_mean = department_group['salary'].mean()
print(f'Department mean salaries {department_salary_mean}')


plt.bar(department , department_salary_mean.values ,color =['red' , 'blue' , 'green' , 'yellow' , 'orange'])
plt.show()

salaries = np.array(employee_data['salary'])
exprience = np.array(employee_data['exprience'].sort_values())

plt.plot(exprience , salaries , marker='o')
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.title("Experience with salaries")
plt.show()
```

Output:



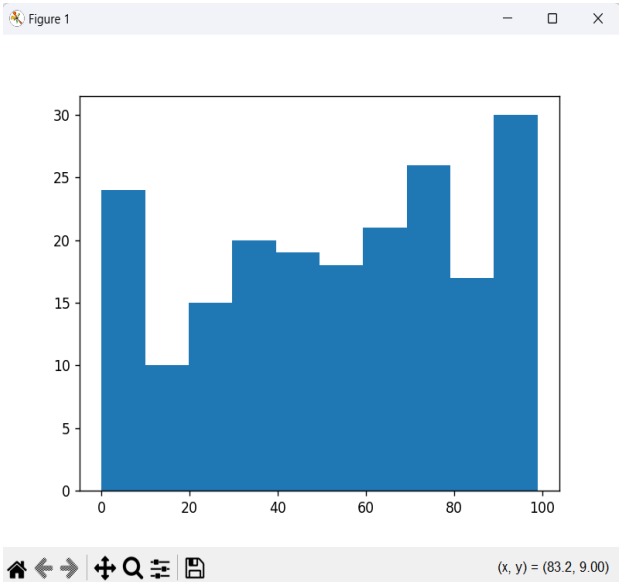


```
Average salary: 75508.22
Max salary: 119740
Min salary: 30592
Filtered employees      employee_id employe_name employee_department  salary  exprience
2               2        Ahmed         Development  115998          8
4               4       Fatima               Sales  118554         15
6               6        Maria            Finanace   99140         21
7               7         Iman               Sales   98816          8
13             13         Omar            Finanace   77042         24
..            ...          ...                 ...     ...        ...
287           287        Ahmed           Marketing  111776         20
288           288         Zara                  HR  117051         21
290           290        Ahmed           Marketing   76617          8
291           291        Bilal            Finanace  101007         22
295           295       Fatima               Sales   98134         13

[112 rows x 5 columns]
Department mean salaries employee_department
Development    74156.303030
Finanace       76038.362069
HR             76087.537037
Marketing      75339.032258
Sales          76136.300000
Name: salary, dtype: float64
Sales          76136.300000
Sales          76136.300000
```

Case-Study 4:

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

np.random.seed(0)


rows = 200
names = ['Ahmed', 'Sarah', 'Ali', 'Fatima', 'Zain', 'Aisha', 'Omar', 'Noor', 'Hassan', 'Maria', 'Bilal', 'Zara', 'Saad', 'Sana', 'Imran', 'Layla', 'Usman', 'Hira', 'Yasir', 'Iman']
subjects = ['PF' , 'TOA' , 'DE' , 'DM' , 'OOP']

student_id = [i for i in range(rows)]
student_names = np.random.choice(names , rows)
subject = np.random.choice(subjects , rows)

marks = np.random.randint(0, 100, rows)


student_data_gen = pd.DataFrame({
    'student_id' : student_id,
    'name' : student_names,
    'subject' : subject,
    'marks' : marks,
})

student_data_gen.to_excel('student_data.xlsx', index=False)


student_data = pd.read_excel('student_data.xlsx')


student_marks = np.array(student_data['marks'])

mean = np.mean(student_marks)
median = np.median(student_marks)
# mode = student_marks.mode()
std = np.std(student_marks)


print(f'Mean: {mean}')
print(f'Meadian: {median}')
# print(f'Mode: {mode}')
print(f'Standard Deviation: {std}')



filtered_student = student_data[(student_data['marks'] > 80)]
print(f'Filtered students {filtered_student}')


subjects_mean = student_data.groupby('subject')['marks'].mean()

print(f'Subjects Mean {subjects_mean}')


plt.hist(np.array(student_data['marks']))
plt.show()


plt.bar(subjects_mean.index , subjects_mean.values , color =['red' , 'blue' , 'green' , 'yellow' , 'orange'])
plt.show()
```

Output:

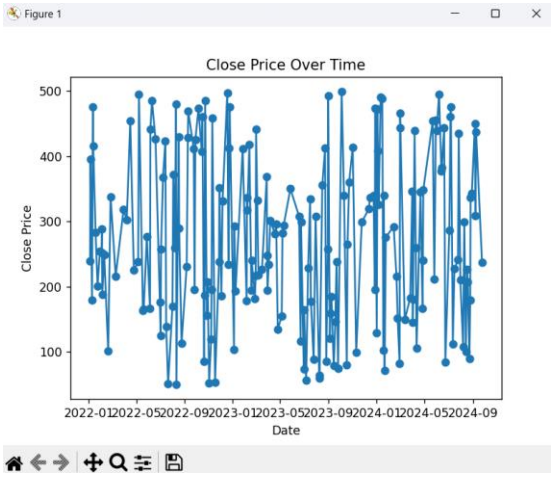



```
Mean: 53.235
Meadian: 57.0
Standard Deviation: 29.75818836891789
Filtered students      student_id    name subject  marks
2              2   Ahmed     OOP     98
3              3  Fatima      DM     97
14            14    Noor      DE     99
18            18    Sana      DE     98
25            25   Layla      DM     96
32            32    Iman      DM     94
34            34   Imran      DM     84
37            37   Sarah     OOP     85
38            38   Maria      PF     91
40            40   Bilal      DM     84
42            42    Zara      DM     83
43            43   Yasir      DE     95
46            46   Ahmed      DE     91
49            49    Omar      DE     83
52            52   Layla      DE     92
59            59   Maria      DM     89
61            61    Omar     TOA     98
62            62    Zara     TOA     87

Subjects Mean subject
DE      55.218750
DM      53.477273
OOP     57.073171
PF      52.350000
TOA     48.674419
Name: marks, dtype: float64
```

Case-Study 5:

```
import pandas as pd
import numpy as np
import matplotlib.pylab as plt

np.random.seed(0)


rows = 1000

date_range = pd.date_range(start='2022-01-01', end='2024-09-30', freq='D')


companies = ['Apple' , 'Amazon' , 'Google' , 'Microsoft' , 'Meta']

dates = np.random.choice(date_range, rows , replace=False)
companies = np.random.choice(companies , rows)
open_price = np.random.randint(50, 500, rows)
closed_price = np.random.randint(50, 500, rows)
volume_traded = np.random.randint(1000, 1000000., rows)


stocks_data_gen = pd.DataFrame({
    'date' : dates,
    'company' : companies,
    'open_price' : open_price,
    'close_price' : closed_price,
    'volume_traded' : volume_traded

})

stocks_data_gen.to_excel('stocks_data.xlsx', index=False)

stocks_data = pd.read_excel('stocks_data.xlsx')


close_price_arr = np.array(stocks_data['close_price'])

open_price_arr = np.array(stocks_data['open_price'])
sub = np.subtract(close_price_arr , open_price_arr)
div = np.divide(sub , open_price_arr)
rounded = np.round(div , decimals=2)
change = np.multiply(rounded,100)

print(f'Change in stocks {change}')

stock_increase = stocks_data[stocks_data['close_price'].diff() > 2]
print(f'Stock price increase {stock_increase}')


volume_traded = stocks_data.groupby('company')['volume_traded'].sum()
print(f'Stocks sold {volume_traded}')



filtered = stocks_data[stocks_data['company'] == 'Apple']

dates = filtered['date'].sort_values()

close_price = filtered['close_price']


plt.plot(dates , close_price , marker='o')
plt.xlabel('Date')
plt.ylabel('Close Price')
plt.title('Close Price Over Time')
plt.show()


stocks_data['change'] = stocks_data.groupby('company')['close_price'].pct_change() * 100


avg_change = stocks_data.groupby('company')['change'].mean()


plt.bar(avg_change.index , avg_change.values)


plt.xlabel('Company')
plt.ylabel('Average Percentage Change in Close Price')
plt.title('Average Percentage Change in Close Price for Different Companies')

plt.show()
```

Output:



	Stock price increase	date	company	open_price	close_price	volume_traded
1	2023-08-29	Microsoft	261	435	625315	
3	2022-12-23	Apple	140	395	280010	
6	2022-01-28	Amazon	484	397	473094	
8	2022-11-03	Google	400	213	217657	
10	2023-06-07	Meta	249	240	397195	
..	...	...	...	...	...	
990	2023-08-23	Apple	316	237	462517	
992	2024-01-25	Microsoft	52	402	729324	
993	2022-10-05	Microsoft	352	473	805400	
998	2024-02-03	Meta	281	161	670901	
999	2024-04-15	Microsoft	457	393	1228	

[510 rows x 5 columns]

Stocks sold company	
Amazon	97941199
Apple	95390936
Google	97567684
Meta	99418728
Microsoft	110445101

```
Change in stocks [ 48.  67. -17. 182. 313. -21. -18. -82. -47. -76.  -4.  15.  43. -20.
 -23. -49. 307. -27.  85. 245.  -4. -14.  29.  21.  27.  93.  74. -45.
  51. 134. -11.  80.  60. -46. -12. 506. -13.   9.  39.   2. -57. -23.
  93.  44.  49.  58.  22. -31.  49.  41.   0. -47. 257. -80.   4. -37.
 -84. -60.  25. 250. -42.  27.   0.  34.  36. 293. -88. -54. 184. -69.
 -12. 140.  79. -29. 776.  65. -29. 100.  14. 189.  83. -59. -29.  97.
 -74. -65. -58. 300. -46. -21. -23. -29.  89. -23. -73.  31. 206. 506.
 -10.   6.  37.  94. -71. -81. -23. -34.   6. -62.  71.  -6. -43. -27.
  89. -62.  20. -73.  53. -11. 521.  -4.  17.  19. 237. -82. 373. -80.
   2. -25. -88.  16. -25.  88. 159. -26. 180. -85. 209. 100.   4. -79.
   7. -51. -28.  91.  15. -28. -70. -52. -13.  32.  88. -46. 190. 116.
 313.  43. 106. 215.  34. 318. -36.  56. -71.  77. 629. -12. -60.   7.
 -85. -40.   3. -45.  98. -27.  51.  71. -21. -33. 515. -41. -43. 659.
 -74.  52.  -8.  59.  42. -46.  29.  87. 100.  -7.   4. -55. -54.  44.
 -44.  46. 298. -16. -23.  48. 128.  29. 848. 116.  14. 264. -84.  36.
  14. -47. -54.  27.  82.  45.  56.  97. -75. -32. 119. -19. 191. -39.
 -16. -57. -72. -12. -37. 138.  68. -72.  44. 623.  54.  78. -54. -54.
  -9. -62.   4. 414.  17.   9. -44.  66.  96. 225.  -5. -43. 245. 596.
 198.  57.  22. 382.   4. -15. 331.  10. -14. 109. 578. -40.  14. -87.
  43.  35.  28. 275. -31. -68. 143.  31. -27.   3. -60. 445. -15.  -4.
 145.  37. -64. -20. -61. -32.   5. 302. -87. -63. -70. -20. 255. -72.
 -51. -22. 112. -19.   7. -84. -50. 263.  28.  99.  -6.  -7. 178. 138.
 253.  87. -11. -43. -44.  42. -67. 224.  91. -54.  22.  16. -15.  62.
```