GOVERNMENT POLYTECHNIC NAGAMANGALA

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Vth Semester Diploma

**Artificial Intelligence and Machine Learning (20CS51)**

**Assignment: 03**

**NAME: KUSUMA D.R**

**ROLL NO: 158CS22023**

1. Download any two datasets from the internet and perform the

following operations.

a) Analyze the univariate dataset Ex- Mean, Mode, Median,

Range, Std, and Variance and perform Univariate tests

for the dataset.

b) Analyze the multivariate of the dataset Ex- co-variance,

co-relation.

c) Visualize the univariate and multivariate with various

plots.

d) Push the code to your GitHub Repository.

e)Perform any probability calculation.

**Important Note:**

**1. Last Date for Submission:31-07-2024.**

**2. Everyone must perform the above operation using**

**datasets (Previously used).**

**3. Submit the report to the email aimlgptn@gmail.com**

1. Download any two datasets from the internet and perform the

following operations.

a) Analyze the univariate dataset Ex- Mean, Mode, Median,

Range, Std, and Variance and perform Univariate tests

for the dataset.

**DATASET**:01 (“/content/iris (1).csv.xlsx”)

**Description:**

* Sepal.length
* Sepal.width
* Petal.width
* Petal.length
* Variety
* **MEAN**

**df.mean(numeric\_only=True)**

**OUTPUT:**

sepal.length 5.843333

sepal.width 3.057333

petal.length 3.758000

petal.width 1.199333

dtype: float64

* **MEDIAN**

**df.median(numeric\_only=True)**

**OUTPUT:**

sepal.length 5.80

sepal.width 3.00

petal.length 4.35

petal.width 1.30

dtype: float64

* **MODE**

**df.mode(numeric\_only=True)**

**OUTPUT:**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | **sepal.length** | **sepal.width** | **petal.length** | **petal.width** | | --- | --- | --- | --- | --- | | **0** | 5.0 | 3.0 | 1.4 | 0.2 | | **1** | NaN | NaN | 1.5 | NaN | |  |  |  |

* **RANGE**

**df.max(numeric\_only=True) - df.min(numeric\_only=True)**

**OUTPUT:**

sepal.length 3.6

sepal.width 2.4

petal.length 5.9

petal.width 2.4

dtype: float64

* **VARIANCE**

**df.var(numeric\_only=True)**

**OUTPUT:**

sepal.length 0.685694

sepal.width 0.189979

petal.length 3.116278

petal.width 0.581006

dtype: float64

* **STD**

**df.std(numeric\_only=True)**

**OUTPUT:**

sepal.length 0.828066

sepal.width 0.435866

petal.length 1.765298

petal.width 0.762238

dtype: float64

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | * **T -TEST** |  |  |

import pandas as pd

from scipy import stats

df=pd.read\_excel("/content/iris (1).csv.xlsx")

t\_statistic, p\_value = stats.ttest\_ind(df['sepal.length'], df['sepal.width'])

print("t-test")

print("T\_statistic:",t\_statistic)

print("p\_value:",p\_value)

**OUTPUT:**

t-test

T\_statistic: 36.463283934449116

p\_value: 7.027919392009369e-112

* **kruskal-wallis test**

import pandas as pd

from scipy import stats

df=pd.read\_excel("/content/iris (1).csv.xlsx")

h\_statistic, p\_value = stats.kruskal(df['sepal.length'], df['sepal.width'])

print("kruskal-wallis test")

print("H\_statistic:",h\_statistic)

print("p\_value:",p\_value)

**OUTPUT:**

kruskal-wallis test

H\_statistic: 224.44246874951114

p\_value: 9.714284161202739e-51

* **wilcoxon rank-sum test**

import pandas as pd

from scipy import stats

df=pd.read\_excel("/content/iris (1).csv.xlsx")

u\_statistic, p\_value = stats.mannwhitneyu(df['sepal.length'], df['sepal.width'])

print("wilcoxon rank-sum test")

print("U\_statistic:",u\_statistic)

print("p\_value:",p\_value)

**OUTPUT:**

wilcoxon rank-sum test

U\_statistic: 22497.5

p\_value: 9.812123178858578e-51

* **anova test**

import pandas as pd

from scipy import stats

df=pd.read\_excel("/content/iris (1).csv.xlsx")

f\_statistic, p\_value = stats.f\_oneway(df['sepal.length'], df['sepal.width'])

print("one-way ANOVA")

print("F\_statistic:",f\_statistic)

print("p\_value:",p\_value)

**OUTPUT:**

one-way ANOVA

F\_statistic: 1329.571075284254

p\_value: 7.027919392009998e-112

* **chi\_square test**

import pandas as pd

from scipy import stats

df=pd.read\_excel("/content/iris (1).csv.xlsx")

chi\_statistic, p\_value, dof, expected = stats.chi2\_contingency(df[['sepal.length', 'sepal.width']])

print("chi-square test")

print("chi\_statistic:",chi\_statistic)

print("p\_value:", p\_value)

**OUTPUT:**

chi-square test

chi\_statistic: 13.475202786622289

p\_value: 1.0

**b) Analyze the multivariate of the dataset Ex- co-variance,**

**co-relation.**

* **co-variance**

**df.cov(numeric\_only=True)**

**OUTPUT:**

|  | **sepal.length** | **sepal.width** | **petal.length** | **petal.width** |
| --- | --- | --- | --- | --- |
| **sepal.length** | 0.685694 | -0.042434 | 1.274315 | 0.516271 |
| **sepal.width** | -0.042434 | 0.189979 | -0.329656 | -0.121639 |
| **petal.length** | 1.274315 | -0.329656 | 3.116278 | 1.295609 |
| **petal.width** | 0.516271 | -0.121639 | 1.295609 | 0.581006 |

* **co-relation.**

**df.corr(numeric\_only=True)**

**OUTPUT:**

|  | **sepal.length** | **sepal.width** | **petal.length** | **petal.width** |
| --- | --- | --- | --- | --- |
| **sepal.length** | 1.000000 | -0.117570 | 0.871754 | 0.817941 |
| **sepal.width** | -0.117570 | 1.000000 | -0.428440 | -0.366126 |
| **petal.length** | 0.871754 | -0.428440 | 1.000000 | 0.962865 |
| **petal.width** | 0.817941 | -0.366126 | 0.962865 | 1.00000 |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**c) Visualize the univariate and multivariate with various**

**plots.**

* **Univariate**
* **BARPLOT**

import pandas as pd

df=pd.read\_excel("/content/iris (1).csv.xlsx")

import matplotlib.pyplot as plt

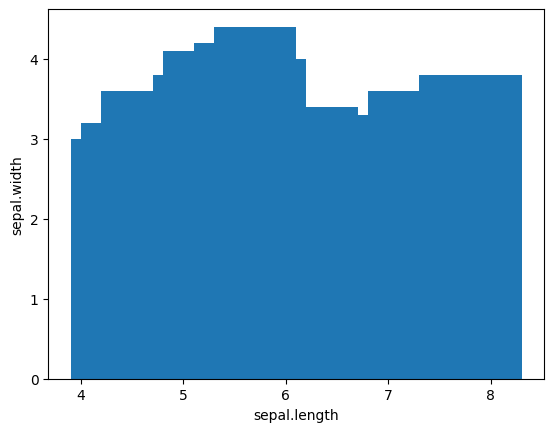
plt.bar(df['sepal.length'],df['sepal.width'])

plt.xlabel("sepal.length")

plt.ylabel("sepal.width")

plt.show()

**OUTPUT:**



* **BOX PLOT**

import pandas as pd

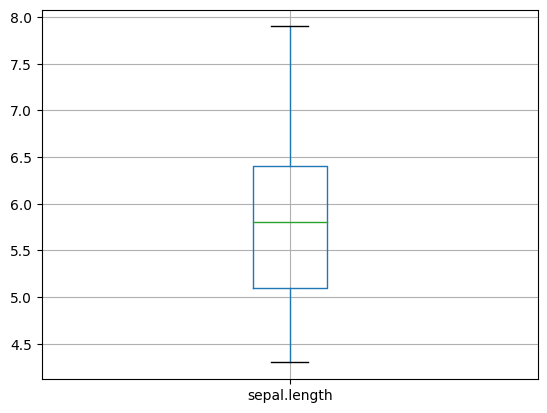
df=pd.read\_excel("/content/iris (1).csv.xlsx")

import matplotlib.pyplot as plt

df.boxplot('sepal.length')

plt.show()

**OUTPUT:**



* **Violinplot**

import pandas as pd

import seaborn as sns

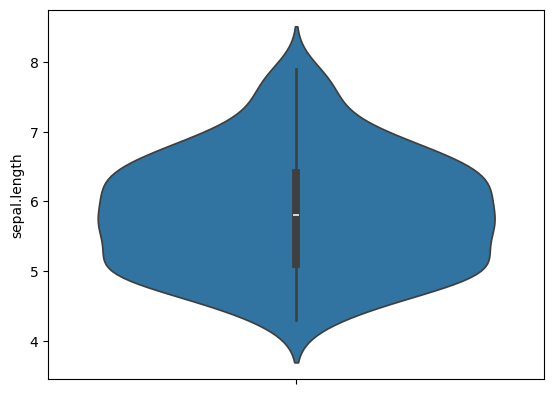
df=pd.read\_excel("/content/iris (1).csv.xlsx")

import matplotlib.pyplot as plt

sns.violinplot(df['sepal.length'])

plt.show()

**OUTPUT:**



* **multivariate**

**pairplot**

import pandas as pd

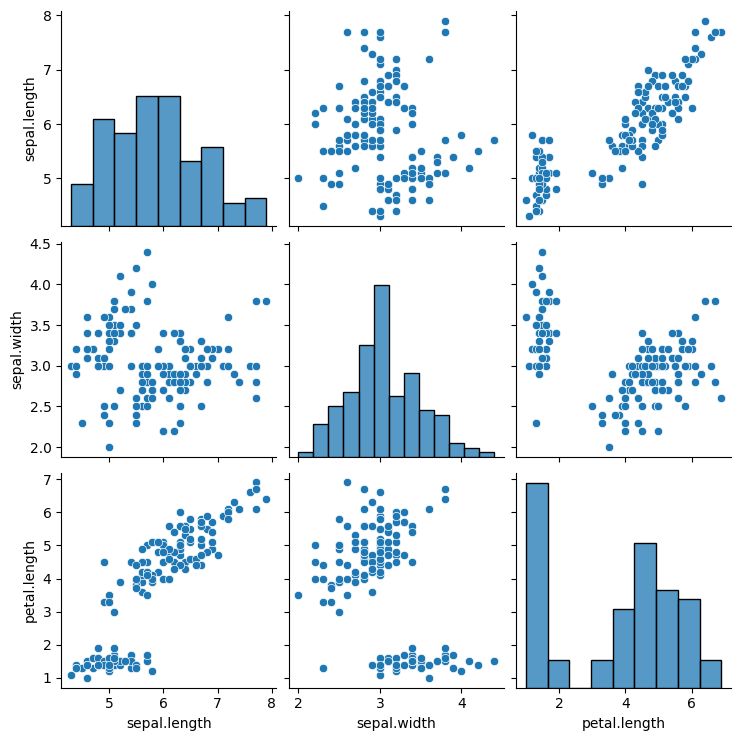
import matplotlib.pyplot as plt

import seaborn as sns

df=pd.read\_excel("/content/iris (1).csv.xlsx")

sns.pairplot(df, vars=['sepal.length', 'sepal.width', 'petal.length'])

**OUTPUT**



**HEATMAP**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

df=pd.read\_excel("/content/iris (1).csv.xlsx")

# Calculate the correlation matrix

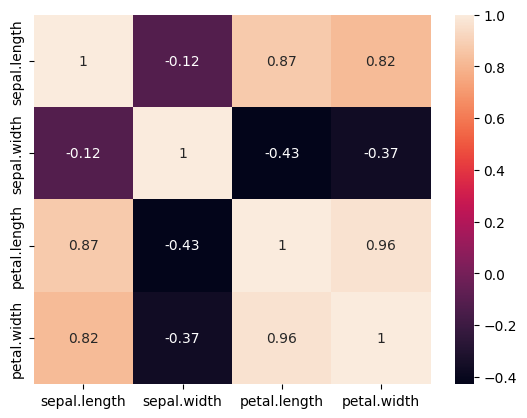
corr\_matrix = df.corr(numeric\_only=True)

# Use seaborn to create the heatmap

sns.heatmap(corr\_matrix, annot=True)

plt.show()

**OUTPUT**



**scatter**

import pandas as pd

df=pd.read\_excel("/content/iris (1).csv.xlsx")

import matplotlib.pyplot as plt

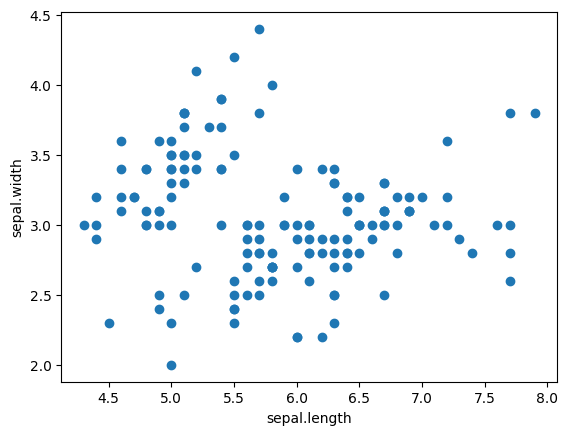
plt.scatter(df['sepal.length'],df['sepal.width'])

plt.xlabel("sepal.length")

plt.ylabel("sepal.width")

plt.show()

**OUTPUT**



**d) Push the code to your GitHub Repository.**

**e)Perform any probability calculation**.

import pandas as pd

from scipy.stats import norm

Data =pd.read\_excel('/content/iris (1).csv.xlsx')

Mean\_sepal=Data['sepal.length'].mean()

Std\_sepal=Data['sepal.length'].std()

Value=2

Z\_score=(Value-Mean\_sepal)/Std\_sepal

Probability=1-norm.cdf(Z\_score)

print(f'proability sepal.length greater than ${Value}:{Probability:2%}')

**output:**

proability sepal.length greater than $2:99.999827%

import pandas as pd

df=pd.read\_excel('/content/iris (1).csv.xlsx')

print(df.columns)

total\_count=df['sepal.length'].sum()

df['probability']=df['sepal.length']/total\_count

print(df)

**output:**

]

import pandas as pd

df=pd.read\_excel('/content/iris (1).csv.xlsx')

print(df.columns)

total\_count=df['sepal.length'].sum()

df['probability']=df['sepal.length']/total\_count

print(df)



Index(['sepal.length', 'sepal.width', 'petal.length', 'petal.width',

'variety'],

dtype='object')

sepal.length sepal.width petal.length petal.width variety \

0 5.1 3.5 1.4 0.2 Setosa

1 4.9 3.0 1.4 0.2 Setosa

2 4.7 3.2 1.3 0.2 Setosa

3 4.6 3.1 1.5 0.2 Setosa

4 5.0 3.6 1.4 0.2 Setosa

.. ... ... ... ... ...

145 6.7 3.0 5.2 2.3 Virginica

146 6.3 2.5 5.0 1.9 Virginica

147 6.5 3.0 5.2 2.0 Virginica

148 6.2 3.4 5.4 2.3 Virginica

149 5.9 3.0 5.1 1.8 Virginica

probability

0 0.005819

1 0.005590

2 0.005362

3 0.005248

4 0.005705

.. ...

145 0.007644

146 0.007188

147 0.007416

148 0.007074

149 0.006731

[150 rows x 6 columns

1. Download any two datasets from the internet and perform the

following operations.

a) Analyze the univariate dataset Ex- Mean, Mode, Median,

Range, Std, and Variance and perform Univariate tests

for the dataset

**DATASET**:02 (“/content/lsg\_stats\_2024\_q2.xlsx”)

**Description:**

* City
* Suburb
* Sales 2Q 2023
* Median 2Q 2023
* Sales 2Q 2024
* Median 2Q 2024
* Median Change
* **MEAN**

**df.mean(numeric\_only=True)**

**OUTPUT**

Sales 2Q 2023 1.177694e+01

Median 2Q 2023 9.205435e+05

Sales 2Q 2024 1.271756e+01

Median 2Q 2024 1.043435e+06

Median Change 1.772568e-01

dtype: float64

* **MEDIAN**

**df.median(numeric\_only=True)**

**OUTPUT**

Sales 2Q 2023 8.000000

Median 2Q 2023 780700.000000

Sales 2Q 2024 9.000000

Median 2Q 2024 892500.000000

Median Change 0.157749

dtype: float64

* **MODE**

**df.mode(numeric\_only=True)**

**OUTPUT**

Sales 2Q 2023 Median 2Q 2023 Sales 2Q 2024 Median 2Q 2024 Median Change

0 1.0 660000.0 2.0 755000.0 0.127843

* **VARINACE**

**df.var(numeric\_only=True)**

**OUTPUT**

Sales 2Q 2023 1.383697e+02

Median 2Q 2023 1.885626e+11

Sales 2Q 2024 1.677593e+02

Median 2Q 2024 2.222015e+11

Median Change 6.432886e-02

dtype: float64

* **STD**

**df.std(numeric\_only=True)**

**OUTPUT**

Sales 2Q 2023 11.763066

Median 2Q 2023 434237.970052

Sales 2Q 2024 12.952193

Median 2Q 2024 471382.560217

Median Change 0.253631

dtype: float64

* **RANGE**

**df.max(numeric\_only=True) - df.min(numeric\_only=True)**

**OUTPUT**

Sales 2Q 2023 9.900000e+01

Median 2Q 2023 3.272000e+06

Sales 2Q 2024 1.220000e+02

Median 2Q 2024 2.645500e+06

Median Change 3.133755e+00

dtype: float64

* **ONE WAY ANOVA TEST**

import pandas as pd

import scipy.stats as stats

df=pd.read\_excel("/content/lsg\_stats\_2024\_q2 (2).xlsx")

f\_value, p\_value=stats.f\_oneway(df['Sales 2Q 2024'], df['Median 2Q 2023'], df['Median Change'])

print("one-way ANOVA")

print("F\_value:", f\_value)

print("p\_value:", p\_value)

**OUTPUT**

Avon

F\_statistic: nan

p\_value: nan

* **CHI-SQUARE TEST**

import pandas as pd

from scipy.stats import chi2\_contingency

df=pd.read\_excel('/content/lsg\_stats\_2024\_q2 (2).xlsx')

contingency\_table=pd.crosstab(df['Sales 2Q 2023'],df['Sales 2Q 2024'])

chi2, p, dof, ex = chi2\_contingency(contingency\_table)

print(f"Chi-Square Test of Independent: chi2={chi2}, p={p}, dog={dof}, expected={ex}")

**OUTPUT**

Chi-Square Test of Independent: chi2=334.3616666666667, p=3.7370271813323256e-05, dog=238, expected=[[0.98039216 0.78431373 0.78431373 0.78431373 0.39215686 0.78431373

0.98039216 0.78431373 0.58823529 0.58823529 0.19607843 0.39215686

0.19607843 0.39215686 0.58823529 0.19607843 0.19607843 0.39215686]

[0.49019608 0.39215686 0.39215686 0.39215686 0.19607843 0.39215686

0.49019608 0.39215686 0.29411765 0.29411765 0.09803922 0.19607843

0.09803922 0.19607843 0.29411765 0.09803922 0.09803922 0.19607843]

[0.49019608 0.39215686 0.39215686 0.39215686 0.19607843 0.39215686

0.49019608 0.39215686 0.29411765 0.29411765 0.09803922 0.19607843

0.09803922 0.19607843 0.29411765 0.09803922 0.09803922 0.19607843]

[0.78431373 0.62745098 0.62745098 0.62745098 0.31372549 0.62745098

0.78431373 0.62745098 0.47058824 0.47058824 0.15686275 0.31372549

0.15686275 0.31372549 0.47058824 0.15686275 0.15686275 0.31372549]

[0.39215686 0.31372549 0.31372549 0.31372549 0.15686275 0.31372549

0.39215686 0.31372549 0.23529412 0.23529412 0.07843137 0.15686275

0.07843137 0.15686275 0.23529412 0.07843137 0.07843137 0.15686275]

[0.19607843 0.15686275 0.15686275 0.15686275 0.07843137 0.15686275

0.19607843 0.15686275 0.11764706 0.11764706 0.03921569 0.07843137

0.03921569 0.07843137 0.11764706 0.03921569 0.03921569 0.07843137]

[0.09803922 0.07843137 0.07843137 0.07843137 0.03921569 0.07843137

0.09803922 0.07843137 0.05882353 0.05882353 0.01960784 0.03921569

0.01960784 0.03921569 0.05882353 0.01960784 0.01960784 0.03921569]

[0.09803922 0.07843137 0.07843137 0.07843137 0.03921569 0.07843137

0.09803922 0.07843137 0.05882353 0.05882353 0.01960784 0.03921569

0.01960784 0.03921569 0.05882353 0.01960784 0.01960784 0.03921569]

[0.09803922 0.07843137 0.07843137 0.07843137 0.03921569 0.07843137

0.09803922 0.07843137 0.05882353 0.05882353 0.01960784 0.03921569

0.01960784 0.03921569 0.05882353 0.01960784 0.01960784 0.03921569]

[0.49019608 0.39215686 0.39215686 0.39215686 0.19607843 0.39215686

0.49019608 0.39215686 0.29411765 0.29411765 0.09803922 0.19607843

0.09803922 0.19607843 0.29411765 0.09803922 0.09803922 0.19607843]

[0.09803922 0.07843137 0.07843137 0.07843137 0.03921569 0.07843137

0.09803922 0.07843137 0.05882353 0.05882353 0.01960784 0.03921569

0.01960784 0.03921569 0.05882353 0.01960784 0.01960784 0.03921569]

[0.09803922 0.07843137 0.07843137 0.07843137 0.03921569 0.07843137

0.09803922 0.07843137 0.05882353 0.05882353 0.01960784 0.03921569

0.01960784 0.03921569 0.05882353 0.01960784 0.01960784 0.03921569]

[0.29411765 0.23529412 0.23529412 0.23529412 0.11764706 0.23529412

0.29411765 0.23529412 0.17647059 0.17647059 0.05882353 0.11764706

0.05882353 0.11764706 0.17647059 0.05882353 0.05882353 0.11764706]

[0.29411765 0.23529412 0.23529412 0.23529412 0.11764706 0.23529412

0.29411765 0.23529412 0.17647059 0.17647059 0.05882353 0.11764706

0.05882353 0.11764706 0.17647059 0.05882353 0.05882353 0.11764706]

[0.09803922 0.07843137 0.07843137 0.07843137 0.03921569 0.07843137

0.09803922 0.07843137 0.05882353 0.05882353 0.01960784 0.03921569

0.01960784 0.03921569 0.05882353 0.01960784 0.01960784 0.03921569]]

* **KRUSKAL WALLIS TEST**

from scipy.stats import kruskal

df=pd.read\_excel("/content/lsg\_stats\_2024\_q2 (2).xlsx")

h\_stat, p\_val = kruskal("Sales 2Q 2023","Sales 2Q 2024","Median 2Q 2023")

print(f"kruskal-wallis Test:H\_stat={h\_stat},P\_val{p\_val}")

**OUTPUT**

Kruskal-wallis Test:H\_stat=2.0,P\_val0.36787944117144245

**b) Analyze the multivariate of the dataset Ex- co-variance,**

**co-relation.**

* **CO-VARIANCE**

**df.cov(numeric\_only=True)**

**OUTPUT**

Sales 2Q 2023 Median 2Q 2023 Sales 2Q 2024 Median 2Q 2024 Median Change

Sales 2Q 2023 1.383697e+02 -1.763076e+06 1.393708e+02 -1.993901e+06 -0.110682

Median 2Q 2023 -1.763076e+06 1.885626e+11 -1.655730e+06 1.705753e+11 -27183.434215

Sales 2Q 2024 1.393708e+02 -1.655730e+06 1.677593e+02 -1.921951e+06 -0.198508

Median 2Q 2024 -1.993901e+06 1.705753e+11 -1.921951e+06 2.222015e+11 29442.004050

Median Change -1.106817e-01 -2.718343e+04 -1.985081e-01 2.944200e+04 0.064329

**co-relation.**

**df.corr(numeric\_only=True)**

**OUTPUT**

| **Sales 2Q 2023** | **Median 2Q 2023** | **Sales 2Q 2024** | **Median 2Q 2024** | **Median Change** |
| --- | --- | --- | --- | --- |
| **Sales 2Q 2023** | 1.000000 | -0.345162 | 0.899836 | -0.356388 | -0.036750 |
| **Median 2Q 2023** | -0.345162 | 1.000000 | -0.293719 | 0.837696 | -0.247991 |
| **Sales 2Q 2024** | 0.899836 | -0.293719 | 1.000000 | -0.314793 | -0.060004 |
| **Median 2Q 2024** | -0.356388 | 0.837696 | -0.314793 | 1.000000 | 0.246376 |
| **Median Change** | -0.036750 | -0.247991 | -0.060004 | 0.246376 | 1.000000 |
| **c) Visualize the univariat and multivariate with various**    **plots.** |  |  |  |  |  |

* **BAR PLOT**

import pandas as pd

df=pd.read\_excel("/content/lsg\_stats\_2024\_q2.xlsx")

import matplotlib.pyplot as plt

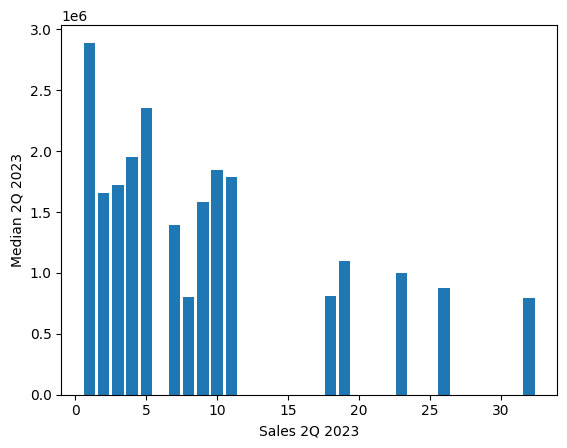
plt.bar(df['Sales 2Q 2023'],df['Median 2Q 2023'])

plt.xlabel("Sales 2Q 2023")

plt.ylabel("Median 2Q 2023")

plt.show()

**OUTPUT**



* **BOX PLOT**

import pandas as pd

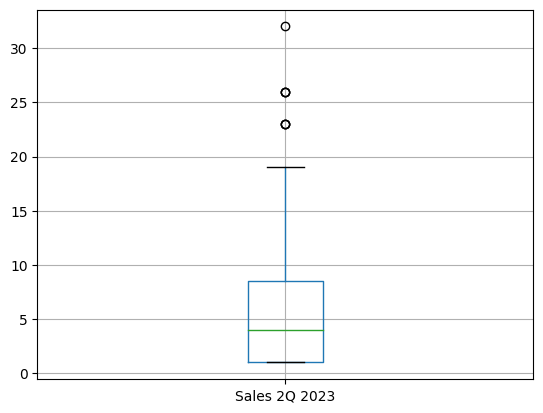
df=pd.read\_excel("/content/lsg\_stats\_2024\_q2.xlsx")

import matplotlib.pyplot as plt

df.boxplot('Sales 2Q 2023')

plt.show()

**OUTPUT**



* VIOLINPLOT

import pandas as pd

import seaborn as sns

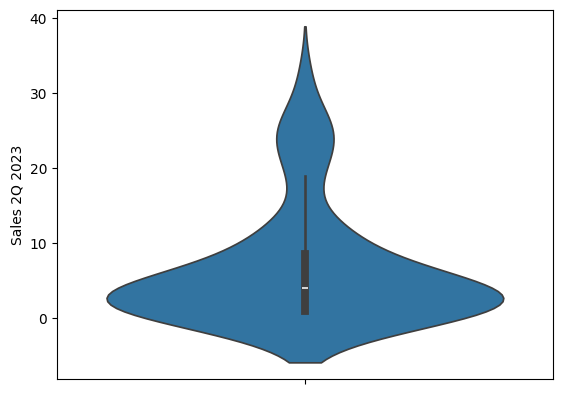
df=pd.read\_excel("/content/lsg\_stats\_2024\_q2.xlsx")

import matplotlib.pyplot as plt

sns.violinplot(df['Sales 2Q 2023'])

plt.show()

**OUTPUT**

* 
* **SCATTER PLOT**

import pandas as pd

df=pd.read\_excel("/content/lsg\_stats\_2024\_q2.xlsx")

import matplotlib.pyplot as plt

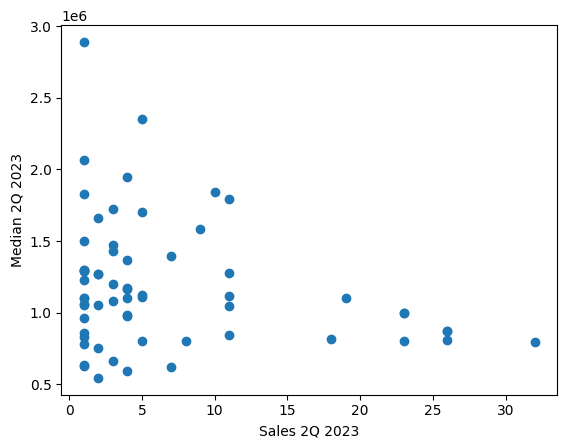
plt.scatter(df['Sales 2Q 2023'],df['Median 2Q 2023'])

plt.xlabel("Sales 2Q 2023")

plt.ylabel("Median 2Q 2023")

plt.show()

**OUTPUT**



* **HEATMAP**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

df=pd.read\_excel("/content/lsg\_stats\_2024\_q2.xlsx")

# Calculate the correlation matrix

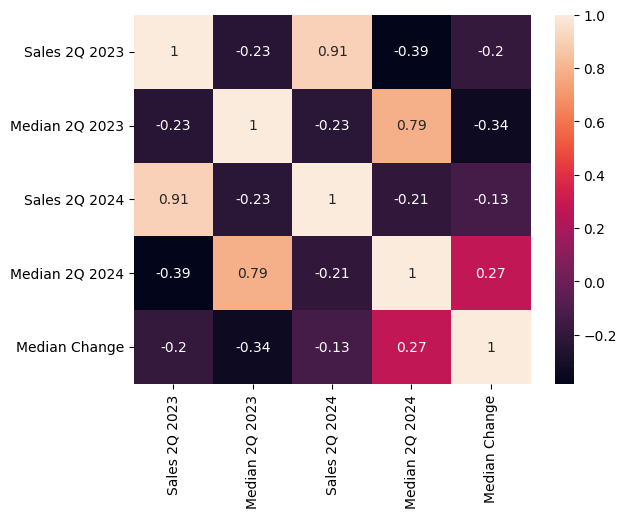
corr\_matrix = df.corr(numeric\_only=True)

# Use seaborn to create the heatmap

sns.heatmap(corr\_matrix, annot=True)

plt.show()

**OUTPUT**



* **PAIRPLOT**

import pandas as pd

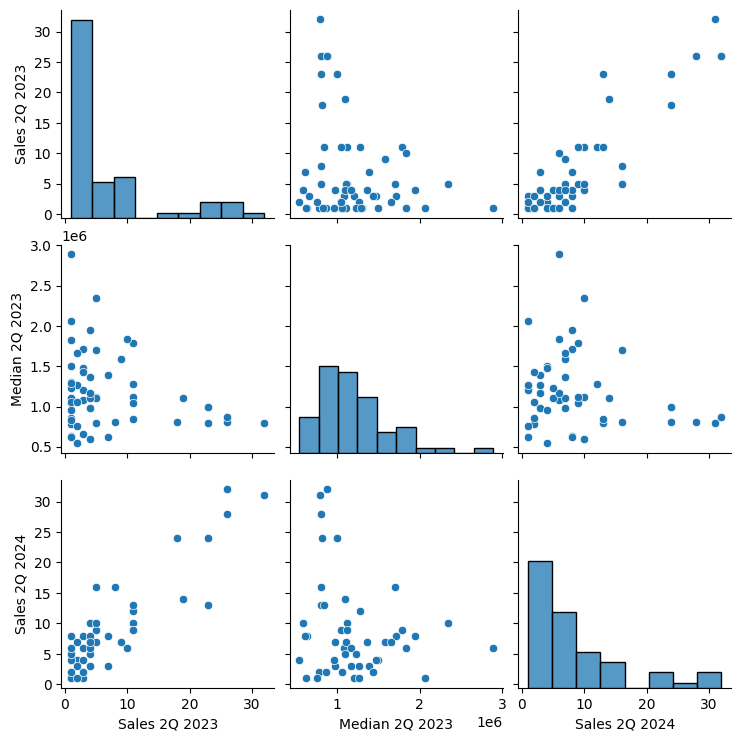
import matplotlib.pyplot as plt

import seaborn as sns

df=pd.read\_excel("/content/lsg\_stats\_2024\_q2.xlsx")

sns.pairplot(df, vars=['Sales 2Q 2023', 'Median 2Q 2023','Sales 2Q 2024'])

**OUTPUT**



**d) Push the code to your GitHub Repository.**

**e)Perform any probability calculation**.

import pandas as pd

from scipy.stats import norm

Data =pd.read\_excel('/content/lsg\_stats\_2024\_q2 (1).xlsx')

Mean\_Sales=Data['Sales 2Q 2023'].mean()

Std\_Sales=Data['Sales 2Q 2023'].std()

Value=30

Z\_score=(Value-Mean\_Sales)/Std\_Sales

Probability=1-norm.cdf(Z\_score)

print(f'proability Sales 2Q 2023 greater than ${Value}:{Probability:2%}')

**output:**

proability Sales 2Q 2023 greater than $30:6.066973%

import pandas as pd

df=pd.read\_excel('/content/lsg\_stats\_2024\_q2 (1).xlsx')

print(df.columns)

total\_count=df['Sales 2Q 2023'].sum()

df['probability']=df['Sales 2Q 2023']/total\_count

print(df)

**output:**

Index(['City', 'Suburb', 'Sales 2Q 2023', 'Median 2Q 2023', 'Sales 2Q 2024',

'Median 2Q 2024', 'Median Change'],

dtype='object')

City Suburb Sales 2Q 2023 Median 2Q 2023 \

0 ADELAIDE ADELAIDE 4.0 978000.0

1 ADELAIDE NORTH ADELAIDE 3.0 1080000.0

2 ADELAIDE HILLS ALDGATE 11.0 1119045.0

3 ADELAIDE HILLS ASHTON NaN NaN

4 ADELAIDE HILLS BALHANNAH 1.0 635000.0

.. ... ... ... ...

477 WEST TORRENS THEBARTON 4.0 875000.0

478 WEST TORRENS TORRENSVILLE 13.0 1058128.0

479 WEST TORRENS UNDERDALE 3.0 1104000.0

480 WEST TORRENS WEST BEACH 21.0 1082000.0

481 WEST TORRENS WEST RICHMOND 3.0 580000.0

Sales 2Q 2024 Median 2Q 2024 Median Change probability

0 7.0 1750000.0 0.789366 0.000851

1 6.0 1325000.0 0.226852 0.000638

2 10.0 1240000.0 0.108088 0.002341

3 2.0 850500.0 NaN NaN

4 8.0 892500.0 0.405512 0.000213

.. ... ... ... ...

477 4.0 1027500.0 0.174286 0.000851

478 12.0 1037000.0 -0.019967 0.002767

479 8.0 1070000.0 -0.030797 0.000638

480 12.0 1250000.0 0.155268 0.004469

481 2.0 790000.0 0.362069 0.000638

[482 rows x 8 columns]