GE: Data Analysis and Visualization Using Python

**Practical**

#1.Write programs in Python using NumPy library to do the following:

#a.Compute the mean, standard deviation, and variance of a two dimensional random integer array along the second axis.

import numpy as np

from numpy import random

x=random.randint(100,size=(3,5))

print (x)

print(x.ndim)

print( "This is the mean of array along second axis")

k=numpy.mean(x,axis=1)

print(k)

print("Standard deviation along second axis")

f=numpy.std(x,axis=1)

print(f)

print(" variance along second axis")

g=numpy.var(x,axis=1)

print(g)

**output**

[[37 86 24 56 90]

[70 8 3 17 41]

[92 79 52 62 81]]

2

This is the mean of array along second axis

[58.6 27.8 73.2]

Standard deviation along second axis

[26.10440576 24.81451188 14.30244734]

variance along second axis

[681.44 615.76 204.56]

#b .create2-dimensional array of size m x n integer elements, also print the shape, type and data type of the array and then   reshape it into an n x m array, where  n and m are user inputs given at the run time

import numpy as np

from numpy import random

m=3

n=4

x=random.randint(50,size=(m,n))

print("shape of 2-D array")

print(x.shape)

print("Type of 2-D array")

print(type(x))

print("dtype of 2-D array")

print(x.dtype)

n=int(input())

m=int(input())

var=x.reshape(n,m)

print(var)

**output**

shape of 2-D array

(3, 4)

Type of 2-D array

<class 'numpy.ndarray'>

dtype of 2-D array

int64

4

3

[[14 16 14]

[34 1 45]

[40 49 21]

[ 0 34 37]]

# C .Test whether the elements of a given 1D array are zero, non-zero and NaN. Record the indices of these

#elements in three separate arrays.

import numpy as np

arr=np.array([1,2,0,4,0,6,3,0,np.nan,np.nan])

print(arr)

ind=np.nonzero(arr)

f=(np.where(arr==0)[0])

print(f)

print(ind)

v=np.isnan(arr)

k =(np.where(v==True)[0])

print(k)

**output**

[ 1. 2. 0. 4. 0. 6. 3. 0. nan nan]

[2 4 7]

(array([0, 1, 3, 5, 6, 8, 9]),)

[8 9]

#d. Create three random arrays of the same size: Array1, Array2 and Array3. Subtract Array 2 from Array3

#and store in Array4. Create another array Array5 having two times the values  in Array1. Find Co

#variance and Correlation of  Array1 with  Array4 and Array5 respectively.

import numpy

from numpy import random

array1=random.randint(15,size=(6))

print(array1)

array2=random.randint(15,size=(6))

print(array2)

array3=random.randint(15,size=(6))

print(array3)

array4=numpy.array(array2-array3)

print(array4)

array5=numpy.array((array1)\*2)

print(array5)

#k=numpy.stack((array1,array4))

print(numpy.cov(array1,array4))

print(numpy.cov(array1,array5))

print(numpy.corrcoef(array1,array4))

output

[ 2 10 3 10 10 8]

[10 5 2 5 5 5]

[12 6 0 8 9 2]

[-2 -1 2 -3 -4 3]

[ 4 20 6 20 20 16]

[[13.76666667 -3.63333333]

[-3.63333333 7.76666667]]

[[13.76666667 27.53333333]

[27.53333333 55.06666667]]

[[ 1. -0.35137715]

[-0.35137715 1. ]]

#e. Create two random arrays of the same size 10: Array1, and Array2. Find the sum of the first half of both

#the arrays and product of the second half of both the arrays.

from numpy import random

array1=random.randint(15,size=10)

array2=random.randint(15,size=10)

print('array1',array1)

print('array2', array2)

sum\_of\_half=numpy.sum(array1[0:6]+array2[0:6])

print("sum of arrays half",sum\_of\_half)

product=numpy.prod(array1[6:]\*array2[6:])

print("product of arrays half",product)

**output**

array1 [ 5 11 3 4 5 8 1 7 3 1]

array2 [ 8 2 5 2 7 0 5 11 3 1]

sum of arrays half 60

product of arrays half 3465

#2. Do the following using PANDAS Series:

#a. Create a series with 5 elements. Display the series sorted on index and also sorted on values seperately

import pandas as pd

data=[1,3,5,2,8]

var=pd.Series(data,index=["c","b","a","d","e"])

print(var)

print(sorted(var))

print(var.sort\_index())

**output**

c 1

b 3

a 5

d 2

e 8

dtype: int64

[1, 2, 3, 5, 8]

a 5

b 3

c 1

d 2

e 8

dtype: int64

#b. Create a  series with N elements with some duplicate values. Find  the minimum and maximum ranks

#assigned to the values using ‘first’ and ‘max’ methods

import pandas as pd

data=[1,2,3,4,4,6,6,7]

var=pd.Series(data)

rank=var.rank()

print(var.rank(method="first"))

print("maximum rank:",rank.max())

print("minimum rank:",rank.min())

**output**

35]

0s

0 1.0

1 2.0

2 3.0

3 4.0

4 5.0

5 6.0

6 7.0

7 8.0

dtype: float64

maximum rank: 8.0

minimum rank: 1.0

#c. Display the index value of the minimum and maximum element of a Series

import pandas as pd

import numpy as np

data=[1,3,4,5,9,7,8,3]

series=pd.Series(data,index=["a","b","c","d","e","f","g","h"])

k=series.max()

l=series.min()

print('maximum index', series.idxmax())

print('minimum index',series.idxmin())

**output**

maximum index e

minimum index a

'''3.

Create a data frame having at least 3 columns and 50 rows to store numeric data generated using a random

function. Replace 10% of the values by null values whose index positions are generated using random function.

Do the following:

a.

Identify and count missing values in a data frame.

b.   Drop the column having more than 5 null values.

c.

Identify the row label having maximum of the sum of all values in a row and drop that row.

d.  Sort the data frame on the basis of the first column.

e.  Remove all duplicates from the first column.

f.

Find the correlation between first and second column and covariance between second and third

column.

g.  Discretize the second column and create 5 bins.  '''

import numpy as np

import pandas as pd

from numpy import random

np.random.seed(0)

Data=random.randint(60,size=(50,3))

dataframe=pd.DataFrame(Data,columns=["a","b","c"])

rows,cols=dataframe.shape

num\_null=int(0.1\*rows\*cols)

Null\_indices=[(np.random.randint(0,rows),np.random.randint(0,cols))for \_ in range(num\_null)]

for rows,cols in Null\_indices:

  dataframe.iloc[rows,cols]=np.nan

  print (dataframe)

count=dataframe.isnull().sum()

print(count)

drop\_column=dataframe.dropna(axis=1,thresh=len(dataframe)-5)

print(drop\_column)

drop\_column=dataframe.sum(axis=1).idxmax(axis=0)

print(drop\_column)

sort\_by=dataframe.sort\_values(by="a")

print(sort\_by)

Remove\_dup=dataframe.drop\_duplicates(subset=["a"])

print(Remove\_dup)

corr\_ab=dataframe["a"].corr(dataframe["b"])

print(corr\_ab)

covar\_ab=dataframe["b"].cov(dataframe["c"])

print(covar\_ab)

**output**

number of null value in columns

a 4

b 0

c 11

dtype: int64

a b

0 44.0 47

1 NaN 3

2 3.0 39

3 19.0 21

4 36.0 23

5 24.0 24

6 58.0 1

7 39.0 23

8 24.0 17

9 25.0 13

10 9.0 20

11 16.0 51

12 15.0 47

13 18.0 35

14 49.0 51

15 19.0 19

16 NaN 32

17 9.0 57

18 31.0 10

19 23.0 35

20 50.0 55

21 34.0 0

22 36.0 53

23 38.0 40

24 17.0 15

25 41.0 42

26 31.0 56

27 1.0 39

28 57.0 35

29 55.0 11

30 18.0 27

31 14.0 35

32 12.0 57

33 20.0 11

34 6.0 4

35 52.0 3

36 36.0 52

37 14.0 15

38 35.0 58

39 15.0 13

40 21.0 48

41 5.0 41

42 0.0 31

43 NaN 0

44 50.0 36

45 48.0 29

46 NaN 42

47 48.0 39

48 9.0 0

49 50.0 43

49

a b c

42 0.0 31 5.0

27 1.0 39 NaN

2 3.0 39 9.0

41 5.0 41 NaN

34 6.0 4 NaN

48 9.0 0 10.0

17 9.0 57 NaN

10 9.0 20 NaN

32 12.0 57 42.0

37 14.0 15 20.0

31 14.0 35 53.0

39 15.0 13 53.0

12 15.0 47 0.0

11 16.0 51 NaN

24 17.0 15 4.0

30 18.0 27 0.0

13 18.0 35 24.0

15 19.0 19 14.0

3 19.0 21 NaN

33 20.0 11 4.0

40 21.0 48 49.0

19 23.0 35 NaN

8 24.0 17 37.0

5 24.0 24 12.0

9 25.0 13 8.0

26 31.0 56 1.0

18 31.0 10 52.0

21 34.0 0 0.0

38 35.0 58 23.0

22 36.0 53 5.0

4 36.0 23 6.0

36 36.0 52 40.0

23 38.0 40 NaN

7 39.0 23 46.0

25 41.0 42 58.0

0 44.0 47 NaN

45 48.0 29 3.0

47 48.0 39 21.0

14 49.0 51 29.0

20 50.0 55 28.0

44 50.0 36 34.0

49 50.0 43 58.0

35 52.0 3 NaN

29 55.0 11 46.0

28 57.0 35 38.0

6 58.0 1 38.0

1 NaN 3 59.0

16 NaN 32 1.0

43 NaN 0 49.0

46 NaN 42 13.0

a b c

0 44.0 47 NaN

1 NaN 3 59.0

2 3.0 39 9.0

3 19.0 21 NaN

4 36.0 23 6.0

5 24.0 24 12.0

6 58.0 1 38.0

7 39.0 23 46.0

9 25.0 13 8.0

10 9.0 20 NaN

11 16.0 51 NaN

12 15.0 47 0.0

13 18.0 35 24.0

14 49.0 51 29.0

18 31.0 10 52.0

19 23.0 35 NaN

20 50.0 55 28.0

21 34.0 0 0.0

23 38.0 40 NaN

24 17.0 15 4.0

25 41.0 42 58.0

27 1.0 39 NaN

28 57.0 35 38.0

29 55.0 11 46.0

31 14.0 35 53.0

32 12.0 57 42.0

33 20.0 11 4.0

34 6.0 4 NaN

35 52.0 3 NaN

38 35.0 58 23.0

40 21.0 48 49.0

41 5.0 41 NaN

42 0.0 31 5.0

45 48.0 29 3.0

correlation

0.03472104830197248

coveriance

-13.445344129554641

''' 4.  Consider two excel files having attendance of  two workshos. Each file has three fields ‘Name’, ‘Date, duration

(in minutes) where names are unique within a file. Note that duration may take one of three values (30, 40, 50)

only. Import the data into two data frames and do the following:'''

import pandas as pd

file1=pd.read\_csv('/content/file1.csv')

file2=pd.read\_csv('/content/file2.csv')

print(file1)

print(file2)

#a. Perform merging of the two data frames to find the names of students who had attended both

#workshops.

df=pd.merge(file1,file2,on="Name",how='inner')

print(df['Name'])

#b.  Find names of all students who have attended a single workshop only.

#c. Merge two data frames row-wise and find the total number of records in the data frame.

merge\_df=pd.concat([file1,file2],axis=1)

total=len(merge\_df)

print(total)

print(merge\_df)

#d. Merge two data frames row-wise and use two columns viz. names and dates as multi-row indexes.

#Generate descriptive statistics for this hierarchical data frame.

merge\_df2=pd.concat([file1.set\_index(['Name','Date']),file2.set\_index(['Name','Date'])])

statistics=merge\_df2.groupby(level=[0,1]).describe()

print(statistics)

print(merge\_df2)

output

Name Date Duration(in minutes)

0 Aman 16.01.2024 40

1 Akash 16.01.2024 30

2 Suman 16.01.2024 50

3 Rohan 12.01.2024 50

4 Shibu 22.01.2024 30

5 Shayam 16.01.2024 40

6 Mounik 12.01.2024 40

7 Prgyaan 27.01.2024 50

8 NaN

Name Date Duration(in minutes) Unnamed: 3

0 Sahil 27.01.2024 30 NaN

1 Vikash 12.01.2024 50 NaN

2 Rupesh 20.01.2024 40 NaN

3 Brijesh 27.01.2024 50 NaN

4 Shibu 22.01.2024 50 NaN

5 Aman 22.01.2024 40 NaN

6 Aashutosh 16.01.2024 30 NaN

7 Akash 22.01.2024 30 NaN

student who have attend the both workshops :-

0 Aman

1 Akash

2 Shibu

Name: Name, dtype: object

Name Date Duration(in minutes) Name Date \

0 Aman 16.01.2024 40 Sahil 27.01.2024

1 Akash 16.01.2024 30 Vikash 12.01.2024

2 Suman 16.01.2024 50 Rupesh 20.01.2024

3 Rohan 12.01.2024 50 Brijesh 27.01.2024

4 Shibu 22.01.2024 30 Shibu 22.01.2024

5 Shayam 16.01.2024 40 Aman 22.01.2024

6 Mounik 12.01.2024 40 Aashutosh 16.01.2024

7 Prgyaan 27.01.2024 50 Akash 22.01.2024

8 NaN

Duration(in minutes) Unnamed: 3

0 30.0 NaN

1 50.0 NaN

2 40.0 NaN

3 50.0 NaN

4 50.0 NaN

5 40.0 NaN

6 30.0 NaN

7 30.0 NaN

8 NaN

total number of records in a data frame :- 9

discriptive statistics :- Unnamed: 3

count mean std min 25% 50% 75% max

Name Date

Aashutosh 16.01.2024 0.0 NaN NaN NaN NaN NaN NaN NaN

Akash 16.01.2024 0.0 NaN NaN NaN NaN NaN NaN NaN

22.01.2024 0.0 NaN NaN NaN NaN NaN NaN NaN

Aman 16.01.2024 0.0 NaN NaN NaN NaN NaN NaN NaN

22.01.2024 0.0 NaN NaN NaN NaN NaN NaN NaN

Brijesh 27.01.2024 0.0 NaN NaN NaN NaN NaN NaN NaN

Mounik 12.01.2024 0.0 NaN NaN NaN NaN NaN NaN NaN

Prgyaan 27.01.2024 0.0 NaN NaN NaN NaN NaN NaN NaN

Rohan 12.01.2024 0.0 NaN NaN NaN NaN NaN NaN NaN

Rupesh 20.01.2024 0.0 NaN NaN NaN NaN NaN NaN NaN

Sahil 27.01.2024 0.0 NaN NaN NaN NaN NaN NaN NaN

Shayam 16.01.2024 0.0 NaN NaN NaN NaN NaN NaN NaN

Shibu 22.01.2024 0.0 NaN NaN NaN NaN NaN NaN NaN

Suman 16.01.2024 0.0 NaN NaN NaN NaN NaN NaN NaN

Vikash 12.01.2024 0.0 NaN NaN NaN NaN NaN NaN NaN

Duration(in minutes) Unnamed: 3

Name Date

Aman 16.01.2024 40 NaN

Akash 16.01.2024 30 NaN

Suman 16.01.2024 50 NaN

Rohan 12.01.2024 50 NaN

Shibu 22.01.2024 30 NaN

Shayam 16.01.2024 40 NaN

Mounik 12.01.2024 40 NaN

Prgyaan 27.01.2024 50 NaN

NaN NaN NaN

Sahil 27.01.2024 30 NaN

Vikash 12.01.2024 50 NaN

Rupesh 20.01.2024 40 NaN

Brijesh 27.01.2024 50 NaN

Shibu 22.01.2024 50 NaN

Aman 22.01.2024 40 NaN

Aashutosh 16.01.2024 30 NaN

Akash 22.01.2024 30 NaN

'''5. Using  Iris data, plot the following with proper legend and axis labels: (Download IRIS data from:

https://archive.ics.uci.edu/ml/datasets/iris or import it from sklearn datasets'''

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.datasets import load\_iris

from scipy import stats

import seaborn as sns

#load iris datasets

iris=load\_iris()

iris\_df=pd.DataFrame(data=np.c\_[iris['data'],iris['target']],columns=iris['feature\_names']+['target'])

print(iris\_df)

iris\_df['Species']=iris\_df['target'].map({0:'setosa',1:'versicolor',2:'virgininca'})

iris\_df.drop('target',axis=1,inplace=True)

#a.  Plot bar chart to show the frequency of each class label in the data.

plt.figure(figsize=(8,6))

sns.countplot(data=iris\_df,x='Species')

plt.title("Frequencty of Each Iris Spacies")

plt.xlabel('Species')

plt.ylabel('Frequecy')

plt.show()

#b.  Draw a scatter plot for Petal width vs sepal width and fit a regression line

plt.figure(figsize=(8,6))

sns.regplot(data=iris\_df,x='sepal width (cm)',y='sepal width (cm)',scatter=True,fit\_reg=True)

plt.title("Scatterplot:petal width vs Sepal width")

plt.xlabel('Sepal width(cm)')

plt.ylabel('Petalwidth(cm)')

plt.show()

#c.  Plot density distribution for feature petal length.

plt.figure(figsize=(8,6))

sns.kdeplot(data=iris\_df['petal length (cm)'],shade=True)

plt.title("Density Distribution of Petal Length")

plt.xlabel('Petal Length (cm)')

plt.ylabel('Density')

plt.show()

#d.  Use a pair plot to show pairwise bivariate distribution in the Iris Dataset.

plt.figure(figsize=(10,8))

sns.pairplot(data=iris\_df,hue='Species',diag\_kind='kde')

plt.suptitle("pairwise Bivariate distribution in Iris Dataset",y=1.02)

plt.show()

#e. Draw heatmap for the four numeric attributes

plt.figure(figsize=(8,6))

sns.heatmap(data=iris\_df.drop('Species',axis=1).corr(),annot=True,cmap='coolwarm',fmt='.2f')

plt.title("correlation Heatmap for Numeric Attributes")

plt.show()

#g. Compute correlation coefficients between each pair of features and plot heatmap

correlation\_matrix=iris\_df.drop('Species',axis=1).corr()

plt.figure(figsize=(8,6))

sns.heatmap(data=correlation\_matrix,annot=True,cmap='coolwarm',fmt='.2f')

plt.title('correlation coefficients between Features')

plt.show

**output**

sepal length (cm) sepal width (cm) petal length (cm) petal width (cm) \

0 5.1 3.5 1.4 0.2

1 4.9 3.0 1.4 0.2

2 4.7 3.2 1.3 0.2

3 4.6 3.1 1.5 0.2

4 5.0 3.6 1.4 0.2

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

145 6.7 3.0 5.2 2.3

146 6.3 2.5 5.0 1.9

147 6.5 3.0 5.2 2.0

148 6.2 3.4 5.4 2.3

149 5.9 3.0 5.1 1.8

target

0 0.0

1 0.0

2 0.0

3 0.0

4 0.0

.. ...

145 2.0

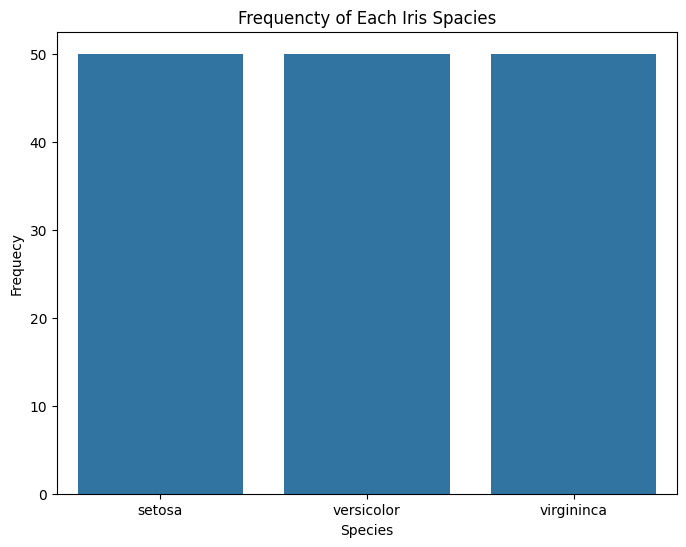
146 2.0

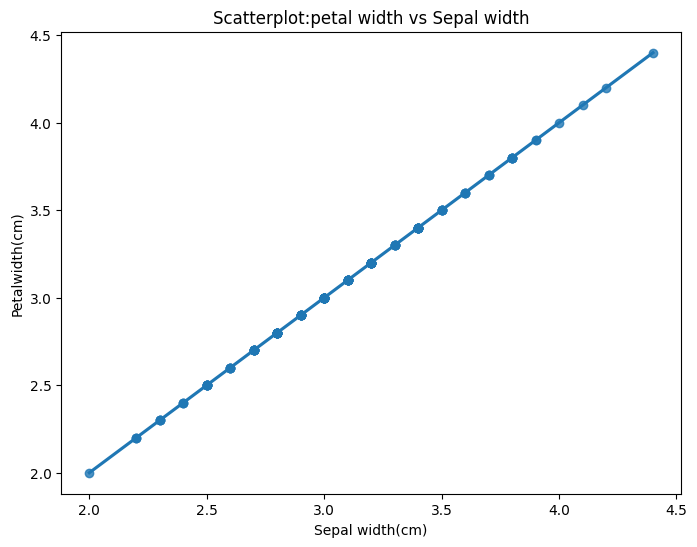
147 2.0

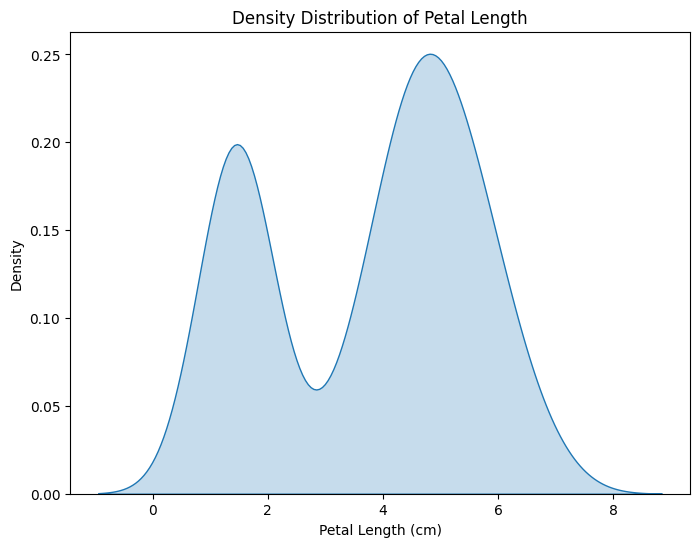
148 2.0

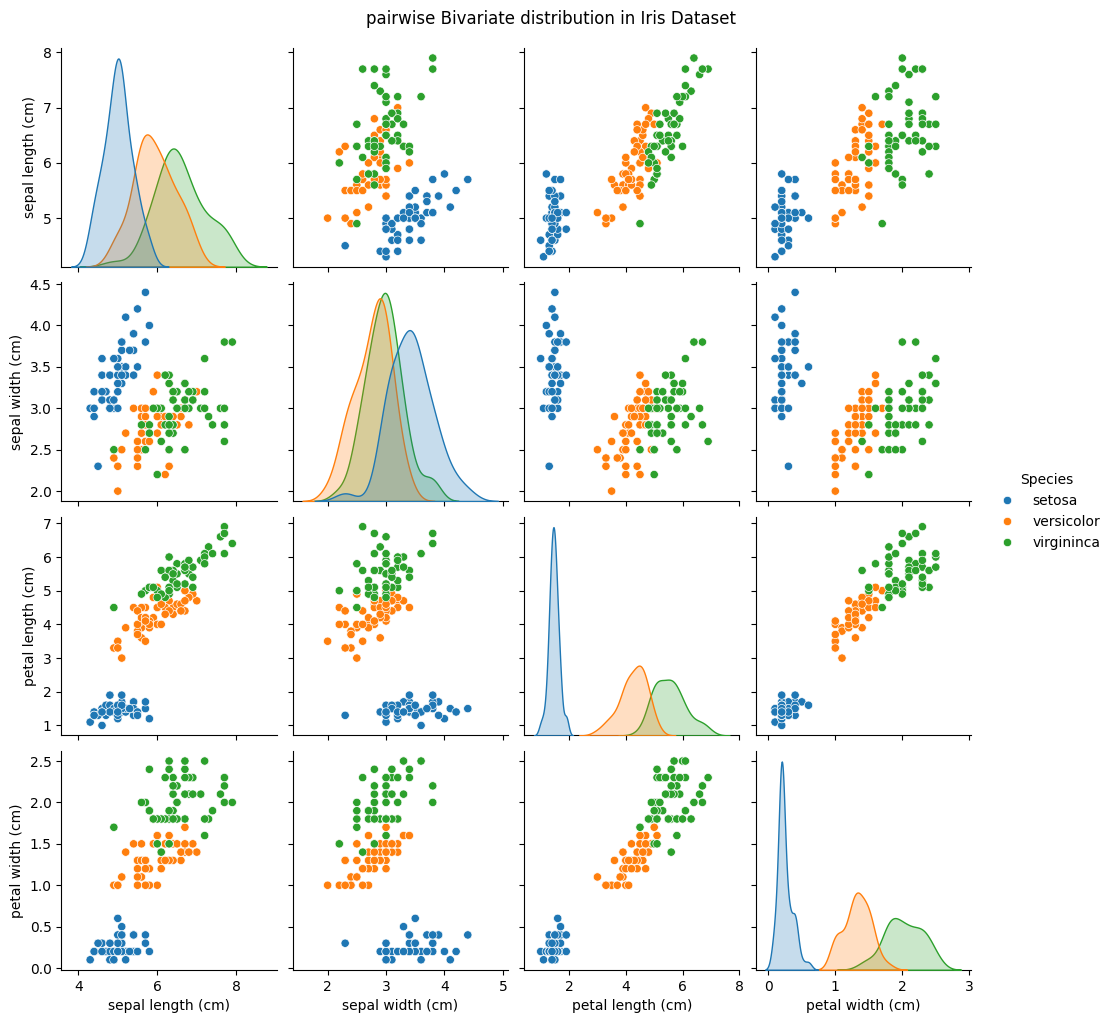
149 2.0

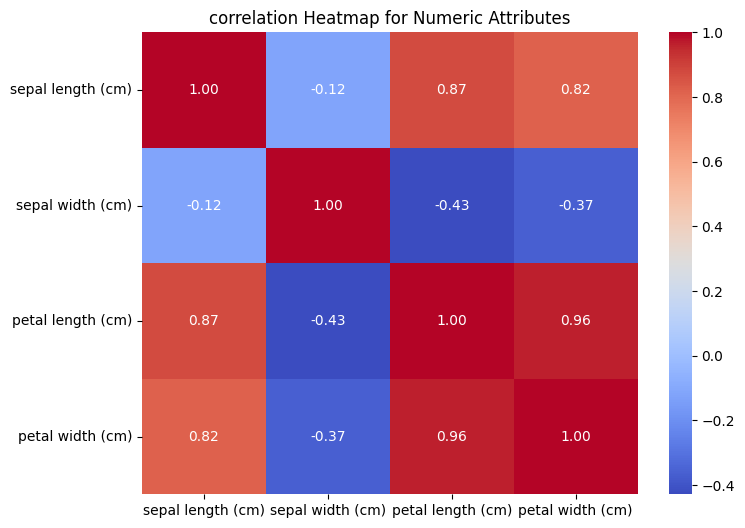
[150 rows x 5 columns]

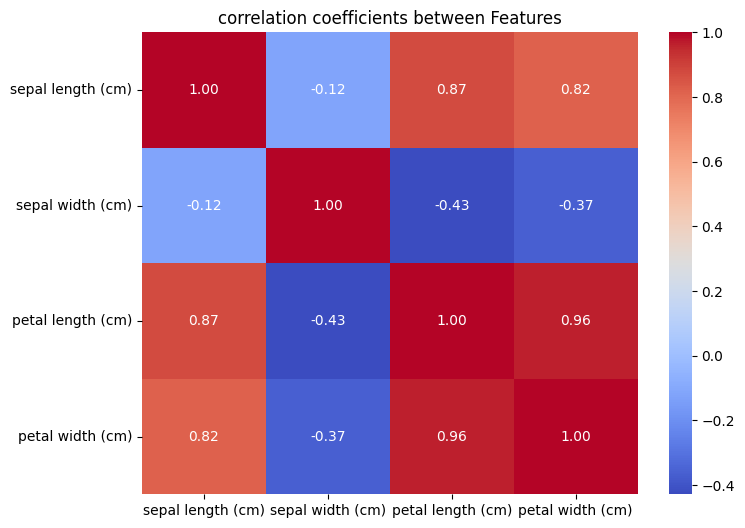












''' 6. Consider the following data frame containing a family name, gender of the family member and her/his monthly

income in each record;

Write a program in Python using Pandas to perform the following: '''

import pandas as pd

data={'Name':['Shah', 'Vats','Vats','Kumar' ,'Vats' ,'Kumar','Shah','Shah','Kumar','Vats'],

        'Gender':['Male','Male','Female','Female','Female','Male','Male','Female','Female','Male'],

      'MonthlyIncome (Rs.)':[114000.00, 65000.00,43150.00,69500.00,155000.00,103000.00,55000.00,112400.00,81030.00,71900.00]}

df=pd.DataFrame(data)

print(df)

#a.  Calculate and display familywise gross monthly income.

gross\_Income=df.groupby("Name")["MonthlyIncome (Rs.)"].sum()

print("family gross monthly income: \n", gross\_Income)

#b.  Calculate and display the member with the highest monthly income.

Max\_monthly\_income=df.loc[df['MonthlyIncome (Rs.)'].idxmax()]

print("Highest monthly income\n",Max\_monthly\_income)

#c.  Calculate and display monthly income of all members with income greater than Rs. 60000.00.

monthly\_inocme=df.loc[df['MonthlyIncome (Rs.)']>60000]

print('person whose income is >600000 \n',monthly\_inocme)

#d.  Calculate and display the average monthly income of the female members

monthly\_inocme\_female=df.loc[df['Gender']=='Female']['MonthlyIncome (Rs.)'].mean()

print("Average monthly income of female member :-",monthly\_inocme\_female)

**output**

]

''' 6. Consider the following data frame containing a family name, gender of the family member and her/his monthly

income in each record;

Write a program in Python using Pandas to perform the following: '''

import pandas as pd

data={'Name':['Shah', 'Vats','Vats','Kumar' ,'Vats' ,'Kumar','Shah','Shah','Kumar','Vats'],

        'Gender':['Male','Male','Female','Female','Female','Male','Male','Female','Female','Male'],

      'MonthlyIncome (Rs.)':[114000.00, 65000.00,43150.00,69500.00,155000.00,103000.00,55000.00,112400.00,81030.00,71900.00]}

df=pd.DataFrame(data)

print(df)

#a.  Calculate and display familywise gross monthly income.

gross\_Income=df.groupby("Name")["MonthlyIncome (Rs.)"].sum()

print("family gross monthly income: \n", gross\_Income)

#b.  Calculate and display the member with the highest monthly income.

Max\_monthly\_income=df.loc[df['MonthlyIncome (Rs.)'].idxmax()]

print("Highest monthly income\n",Max\_monthly\_income)

#c.  Calculate and display monthly income of all members with income greater than Rs. 60000.00.

monthly\_inocme=df.loc[df['MonthlyIncome (Rs.)']>60000]

print('person whose income is >600000 \n',monthly\_inocme)

#d.  Calculate and display the average monthly income of the female members

monthly\_inocme\_female=df.loc[df['Gender']=='Female']['MonthlyIncome (Rs.)'].mean()

print("Average monthly income of female member :-",monthly\_inocme\_female)

Name Gender MonthlyIncome (Rs.)

0 Shah Male 114000.0

1 Vats Male 65000.0

2 Vats Female 43150.0

3 Kumar Female 69500.0

4 Vats Female 155000.0

5 Kumar Male 103000.0

6 Shah Male 55000.0

7 Shah Female 112400.0

8 Kumar Female 81030.0

9 Vats Male 71900.0

family gross monthly income:

Name

Kumar 253530.0

Shah 281400.0

Vats 335050.0

Name: MonthlyIncome (Rs.), dtype: float64

Highest monthly income

Name Vats

Gender Female

MonthlyIncome (Rs.) 155000.0

Name: 4, dtype: object

person whose income is >600000

Name Gender MonthlyIncome (Rs.)

0 Shah Male 114000.0

1 Vats Male 65000.0

3 Kumar Female 69500.0

4 Vats Female 155000.0

5 Kumar Male 103000.0

7 Shah Female 112400.0

8 Kumar Female 81030.0

9 Vats Male 71900.0

Average monthly income of female member :- 92216.0

'''7. Using Titanic dataset, to do the following:  '''

import pandas as pd

df=pd.read\_csv('titanic.csv')

print(df)

#a. Find total number of passengers with age less than 30

Age=df.loc[df["Age"]<30]

print("total number:",len(Age))

#b. Find total fare paid by passengers of first class

first\_class=df.loc[df["Pclass"]==1]["Fare"].sum()

print('sum of all fare of first class',first\_class)

#c. Compare number of survivors of each passenger class

survived=df.groupby('Pclass')['Survived'].sum()

print(survived)

#d. Compute descriptive statistics for any numeric attribute genderwise

statistic\_dis=df.groupby('Sex').describe()

print('discription statistic \n',statistic\_dis)

**output**

PassengerId Survived Pclass \

0 1 0 3

1 2 1 1

2 3 1 3

3 4 1 1

4 5 0 3

.. ... ... ...

886 887 0 2

887 888 1 1

888 889 0 3

889 890 1 1

890 891 0 3

Name Sex Age SibSp \

0 Braund, Mr. Owen Harris male 22.0 1

1 Cumings, Mrs. John Bradley (Florence Briggs Th... female 38.0 1

2 Heikkinen, Miss. Laina female 26.0 0

3 Futrelle, Mrs. Jacques Heath (Lily May Peel) female 35.0 1

4 Allen, Mr. William Henry male 35.0 0

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

886 Montvila, Rev. Juozas male 27.0 0

887 Graham, Miss. Margaret Edith female 19.0 0

888 Johnston, Miss. Catherine Helen "Carrie" female NaN 1

889 Behr, Mr. Karl Howell male 26.0 0

890 Dooley, Mr. Patrick male 32.0 0

Parch Ticket Fare Cabin Embarked

0 0 A/5 21171 7.2500 NaN S

1 0 PC 17599 71.2833 C85 C

2 0 STON/O2. 3101282 7.9250 NaN S

3 0 113803 53.1000 C123 S

4 0 373450 8.0500 NaN S

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

886 0 211536 13.0000 NaN S

887 0 112053 30.0000 B42 S

888 2 W./C. 6607 23.4500 NaN S

889 0 111369 30.0000 C148 C

890 0 370376 7.7500 NaN Q

[891 rows x 12 columns]

total number: 384

sum of all fare of first class 18177.4125

Pclass

1 136

2 87

3 119

Name: Survived, dtype: int64

discription statistic

PassengerId \

count mean std min 25% 50% 75% max

Sex

female 314.0 431.028662 256.846324 2.0 231.75 414.5 641.25 889.0

male 577.0 454.147314 257.486139 1.0 222.00 464.0 680.00 891.0

Survived ... Parch Fare \

count mean ... 75% max count mean std min

Sex ...

female 314.0 0.742038 ... 1.0 6.0 314.0 44.479818 57.997698 6.75

male 577.0 0.188908 ... 0.0 5.0 577.0 25.523893 43.138263 0.00

25% 50% 75% max

Sex

female 12.071875 23.0 55.00 512.3292

male 7.895800 10.5 26.55 512.3292

[2 rows x 56 columns]