VASAVI COLLEGE OF ENGINEERING

AUTONOMOUS

(Affiliated to Osmania University)

Hyderabad - 500 031.

DEPARTMENT OF : CSE

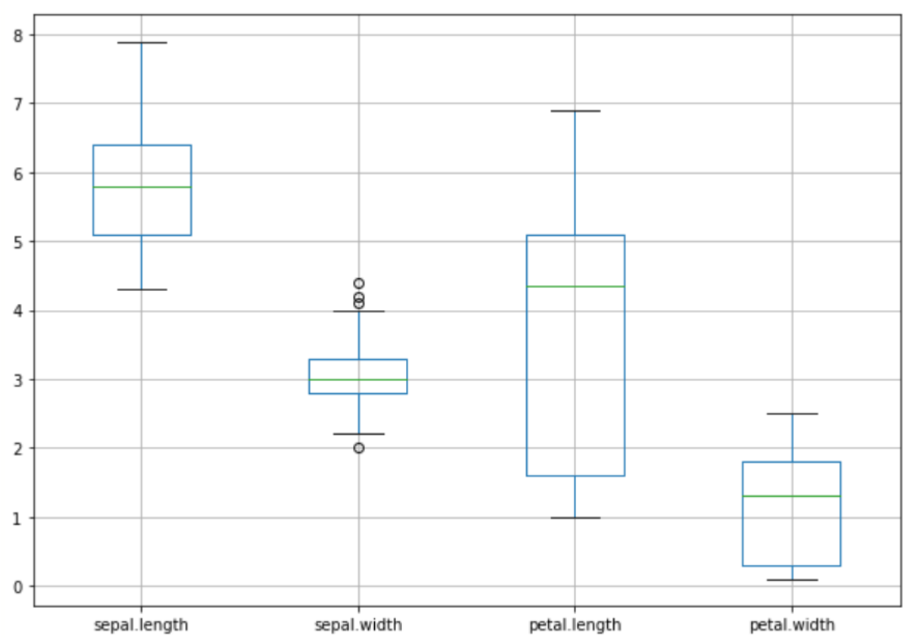
NAME OF THE LABORATORY :

Name P.Pranaya **Roll No. 1602-19-733-037** **Page No:**

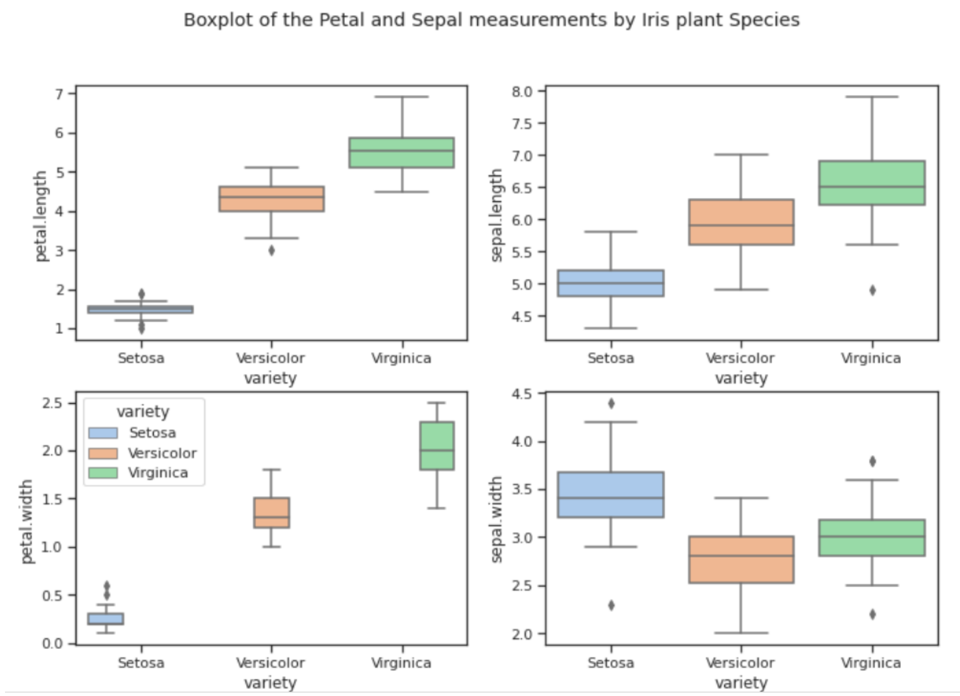
1. Draw boxplot for each attribute of iris data set

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
data = pd.read_csv("iris.csv")
new_data = data[["sepal.length", "sepal.width", "petal.length", "petal.width"]]
plt.figure(figsize = (10, 7))
new_data.boxplot()
sns.set(style="ticks", palette="pastel")
f, axes = plt.subplots(2, 2, sharey=False, figsize=(12, 8))
sns.boxplot(x="variety", y="petal.length",data=data, ax = axes[0,0])
sns.boxplot(x="variety", y="sepal.length", data=data, ax=axes[0,1])
sns.boxplot(x="variety", y="petal.width",hue = "variety",data=data, ax=axes[1,0])
sns.boxplot(x="variety", y="sepal.width", data=data, ax=axes[1,1])
f.suptitle("Boxplot of the Petal and Sepal measurements by Iris plant Species")
plt.show()
```

OUTPUT



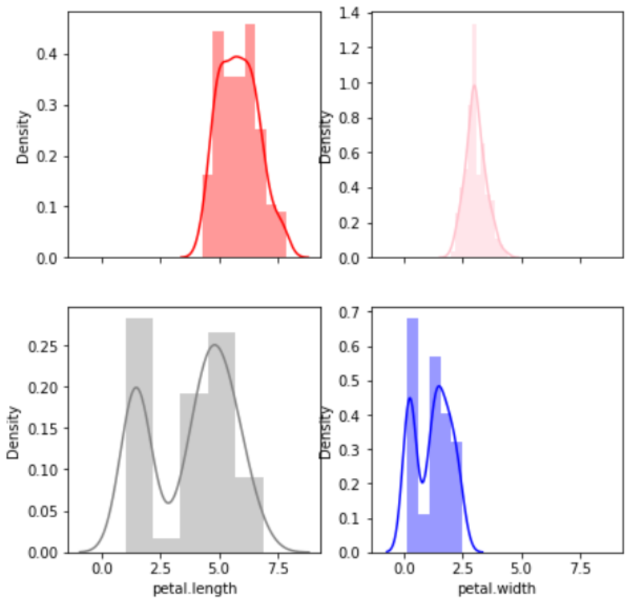
Name P.Pranaya **Roll No.** 1602-19-733-037 **Page No:**



2.Draw data distribution curve for each attribute of iris data set and check whether the attributes are unform distributed or positively / negatively skewed

```
import math,os,random
import pandas as pd
import numpy as np
import seaborn as sns
import scipy.stats as stat
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
col=['sepal.length','sepal.width','petal.length','petal.width','variety']
iris=pd.read_csv("iris.csv",names=col)
f, axes = plt.subplots(2, 2, figsize=(7, 7), sharex=True)

sns.distplot( iris["sepal.length"][1:] , color="red", ax=axes[0, 0])
sns.distplot( iris["sepal.width"][1:] , color="pink", ax=axes[0, 1])
sns.distplot( iris["petal.length"][1:] , color="grey", ax=axes[1, 0])
sns.distplot( iris["petal.width"][1:] , color="blue", ax=axes[1, 1])
```



3.Find mean, median, standard deviation of each attribute of iris data set and check whether the attributes are unform distributed or positively / negatively skewed

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
data = pd.read_csv("iris.csv")
data.describe()

#sepallength
sum_data = data["sepal.length"].sum()
mean_data = data["sepal.length"].mean()
median_data = data["sepal.length"].median()
std=data["sepal.length"].std()
print("Sepal Length:\n Sum:",sum_data, "\nMean:", mean_data, "\nMedian:",median_da
ta,"\nStandardDeviation:",std)
print()
sum_data = data["sepal.width"].sum()
mean_data = data["sepal.width"].mean()
median_data = data["sepal.width"].median()
std=data["sepal.width"].std()
print("Sepal Width:\n Sum:",sum_data, "\nMean:", mean_data, "\nMedian:",median_dat
a,"\nStandardDeviation:",std)
```

```
sum_data = data["petal.length"].sum()
mean_data = data["petal.length"].mean()
median_data = data["petal.length"].median()
std=data["sepal.width"].std()
print("Petal Length:\n Sum:",sum_data, "\nMean:", mean_data, "\nMedian:",median_da
ta,"\nStandardDeviation:",std)
print()
#petal width
sum_data = data["petal.width"].sum()
mean_data = data["petal.width"].mean()
median_data = data["petal.width"].median()
std=data["sepal.width"].std()
print("Petal Width:\n Sum:",sum_data, "\nMean:", mean_data, "\nMedian:",median_dat
a,"\nStandardDeviation:",std)
```

```
Sepal Length:
 Sum: 876.5
Mean: 5.843333333333334
Median: 5.8
StandardDeviation: 0.828066127977863

Sepal Width:
 Sum: 458.6
Mean: 3.0573333333333337
Median: 3.0
StandardDeviation: 0.4358662849366982

Petal Length:
 Sum: 563.7
Mean: 3.7580000000000005
Median: 4.35
StandardDeviation: 0.4358662849366982

Petal Width:
 Sum: 179.90000000000003
Mean: 1.1993333333333336
Median: 1.3
StandardDeviation: 0.4358662849366982
```

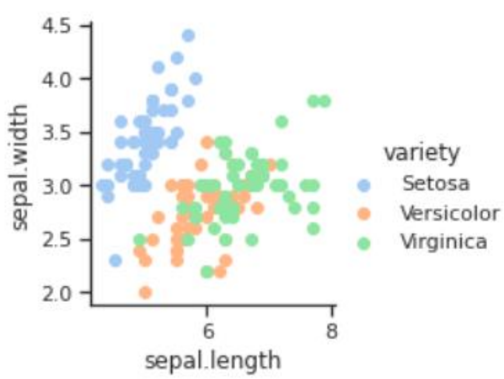
VASAVI COLLEGE OF ENGINEERING

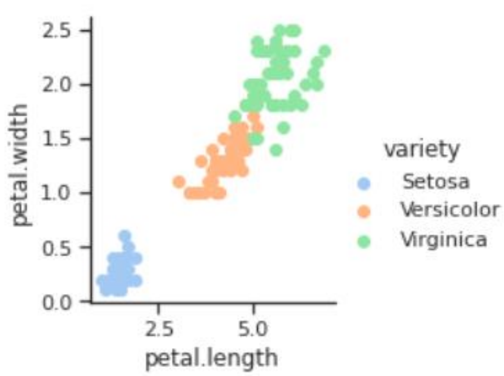
AUTONOMOUS
(Affiliated to Osmania University)
Hyderabad - 500 031.

DEPARTMENT OF : CSE
NAME OF THE LABORATORY :
Name P.Pranaya **Roll No.** 1602-19-733-037 **Page No:**

4.Draw the scatter plot for each pair of attributes of iris data set

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import statsmodels.api as sm
data = pd.read_csv("iris.csv")
sns.FacetGrid(data,hue="variety")\
.map(plt.scatter,"sepal.length","sepal.width")\
.add_legend()
plt.show()
sns.FacetGrid(data,hue="variety")\
.map(plt.scatter,"petal.length","petal.width")\
.add_legend()
plt.show()
```





5.Compute the correlation coefficient for every pair attribute of iris data set

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import statsmodels.api as sm
data = pd.read_csv("iris.csv")
data.groupby("variety").corr()
```

variety		sepal.length	sepal.width	petal.length	petal.width
Setosa	sepal.length	1.000000	0.742547	0.267176	0.278098
	sepal.width	0.742547	1.000000	0.177700	0.232752
	petal.length	0.267176	0.177700	1.000000	0.331630
	petal.width	0.278098	0.232752	0.331630	1.000000
Versicolor	sepal.length	1.000000	0.525911	0.754049	0.546461
	sepal.width	0.525911	1.000000	0.560522	0.663999
	petal.length	0.754049	0.560522	1.000000	0.786668
	petal.width	0.546461	0.663999	0.786668	1.000000
Virginica	sepal.length	1.000000	0.457228	0.864225	0.281108
	sepal.width	0.457228	1.000000	0.401045	0.537728
	petal.length	0.864225	0.401045	1.000000	0.322108
	petal.width	0.281108	0.537728	0.322108	1.000000

6.Find covariance of every pair of attributes of iris data set

```
from sklearn import datasets
import numpy as np
iris = datasets.load_iris()
cov_data = np.corrcoef(iris.data.T)
print(cov_data)
```

```
[[ 1.         -0.11756978  0.87175378  0.81794113]
 [-0.11756978  1.         -0.4284401  -0.36612593]
 [ 0.87175378 -0.4284401   1.          0.96286543]
 [ 0.81794113 -0.36612593  0.96286543  1.        ]]
```

7.Draw the histogram for every attribute of iris data set consider the width of histogram as 50

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

data = pd.read_csv("iris.csv")

plt.figure(figsize = (50, 12))
x = data["sepal.length"]

plt.hist(x, bins = 20, color = "blue")
plt.title("Sepal Length ")
plt.xlabel("Sepal_Length")
plt.ylabel("Count")

plt.figure(figsize = (50, 12))
y = data["sepal.width"]
plt.hist(y, bins = 20, color = "green")
plt.title("Sepal Width ")
plt.xlabel("Sepal_Width")
plt.ylabel("Count")

plt.figure(figsize = (50, 12))
z= data["petal.length"]
plt.hist(z, bins = 20, color = "red")
plt.title("Petal Length ")
plt.xlabel("Petal_Length")
plt.ylabel("Count")
plt.figure(figsize = (50, 12))
q = data["petal.width"]
plt.hist(q, bins = 20, color = "yellow")
plt.title("Petal Width ")
plt.xlabel("Petal_Width")
plt.ylabel("Count")
```

VASAVI COLLEGE OF ENGINEERING

AUTONOMOUS

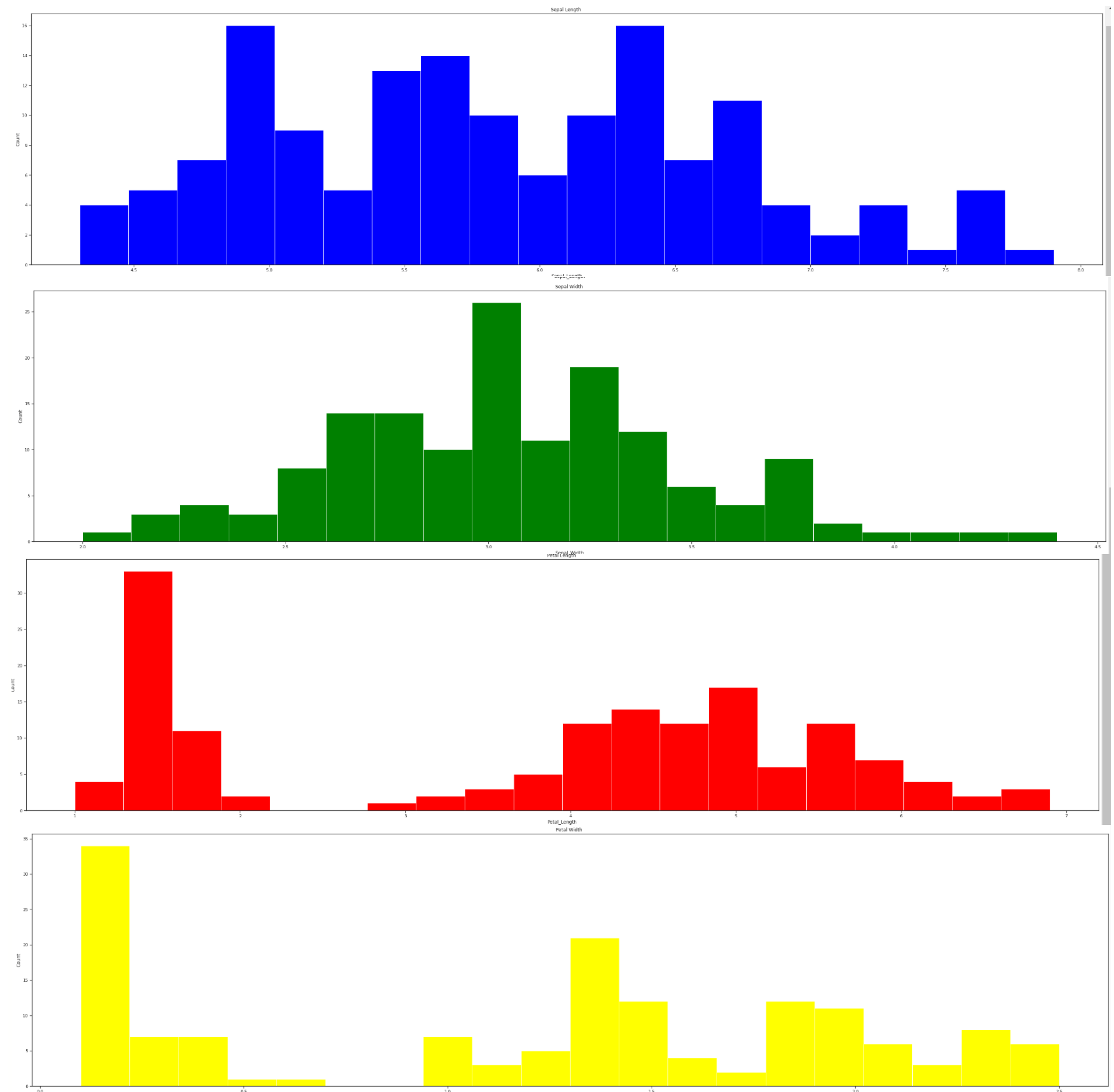
(Affiliated to Osmania University)

Hyderabad - 500 031.

DEPARTMENT OF : CSE

NAME OF THE LABORATORY :

Name P.Pranaya Roll No. 1602-19-733-037 Page No:



8. Construct distance matrix of iris data set using Euclidean, Manhattan distance measures

```
import csv
import math
A=[]
with open('iris.csv',newline='') as csvfile:
  for row in csv.reader(csvfile):
    A.append(row)
  t=[]
  for i in A:
    t.append(i[:-1])
  t.pop(0)
  print(len(t))
  res=[]
  res1=[]
  for i in range(0,len(t)):
    temp=[]
    temp1=[]
    for j in range(0,i+1):
      if i==j:
        temp.append(0)
        temp1.append(0)
      else:

        sum1=(float(t[i][0])-float(t[j][0]))**2+(float(t[i][1])-
float(t[j][1]))**2+(float(t[i][2])-float(t[j][2]))**2+(float(t[i][3])-
float(t[j][3]))**2
        sum2=abs(float(t[i][0])-float(t[j][0]))+abs(float(t[i][1])-
float(t[j][1]))+abs(float(t[i][2])-float(t[j][2]))+abs(float(t[i][3])-
float(t[j][3]))
        p=math.sqrt(sum1)
        #p=float("{:.2f}".format(p))
        p=round(p,2)
        temp.append(p)
        p1=round(sum1,2)
        temp1.append(p1)
    res.append(temp)
    res1.append(temp1)
  print("Euclidian matrix")
```

```
for i in res:
  print(i)
print("Manhattan matrix")
for j in res1:
  print(j)
```

```
150
Euclidian matrix
[0]
[0.54, 0]
[0.51, 0.3, 0]
[0.65, 0.33, 0.24, 0]
[0.14, 0.61, 0.51, 0.65, 0]
[0.62, 1.09, 1.09, 1.17, 0.62, 0]
[0.52, 0.51, 0.26, 0.33, 0.46, 0.99, 0]
[0.17, 0.42, 0.41, 0.5, 0.22, 0.7, 0.42, 0]
[0.92, 0.51, 0.44, 0.3, 0.92, 1.46, 0.55, 0.79, 0]
[0.47, 0.17, 0.32, 0.32, 0.53, 1.01, 0.48, 0.33, 0.56, 0]
[0.37, 0.87, 0.88, 1.0, 0.42, 0.35, 0.87, 0.5, 1.28, 0.79, 0]
[0.37, 0.46, 0.37, 0.37, 0.35, 0.81, 0.3, 0.22, 0.67, 0.35, 0.68, 0]
[0.59, 0.14, 0.26, 0.26, 0.64, 1.16, 0.49, 0.47, 0.42, 0.17, 0.93, 0.46, 0]
[0.99, 0.68, 0.5, 0.52, 0.97, 1.57, 0.62, 0.91, 0.35, 0.73, 1.37, 0.82, 0.58, 0]
[0.88, 1.36, 1.36, 1.53, 0.92, 0.68, 1.36, 1.04, 1.79, 1.31, 0.58, 1.23, 1.43, 1.81, 0]
[1.1, 1.63, 1.59, 1.71, 1.09, 0.62, 1.49, 1.24, 2.0, 1.56, 0.79, 1.36, 1.69, 2.04, 0.55, 0]
[0.55, 1.05, 1.01, 1.17, 0.55, 0.4, 0.95, 0.7, 1.43, 1.01, 0.35, 0.86, 1.13, 1.47, 0.47, 0.62, 0]
[0.1, 0.55, 0.52, 0.66, 0.17, 0.59, 0.51, 0.2, 0.93, 0.5, 0.39, 0.39, 0.62, 1.01, 0.89, 1.09, 0.52, 0]
[0.74, 1.17, 1.24, 1.32, 0.79, 0.33, 1.21, 0.84, 1.61, 1.1, 0.39, 0.99, 1.26, 1.73, 0.56, 0.64, 0.52, 0.73, 0]
[0.33, 0.84, 0.75, 0.87, 0.26, 0.39, 0.65, 0.42, 1.15, 0.75, 0.33, 0.52, 0.88, 1.22, 0.79, 0.85, 0.39, 0.32, 0.63, 0]
[0.44, 0.71, 0.83, 0.88, 0.54, 0.54, 0.86, 0.45, 1.16, 0.62, 0.36, 0.61, 0.79, 1.32, 0.88, 1.08, 0.67, 0.45, 0.51, 0.55, 0]
[0.3, 0.76, 0.7, 0.81, 0.26, 0.41, 0.6, 0.37, 1.09, 0.7, 0.36, 0.48, 0.82, 1.17, 0.84, 0.92, 0.41, 0.24, 0.65, 0.14, 0.51, 0]
[0.65, 0.78, 0.51, 0.71, 0.57, 1.12, 0.46, 0.67, 0.83, 0.77, 0.95, 0.66, 0.75, 0.69, 1.28, 1.46, 0.93, 0.66, 1.32, 0.74, 1.08, 0.74, 0]
[0.47, 0.56, 0.65, 0.65, 0.53, 0.68, 0.62, 0.39, 0.91, 0.53, 0.62, 0.45, 0.66, 1.12, 1.15, 1.27, 0.79, 0.41, 0.81, 0.57, 0.44, 0.46, 0.96, 0]
[0.59, 0.65, 0.64, 0.54, 0.57, 0.83, 0.55, 0.45, 0.81, 0.52, 0.78, 0.3, 0.65, 1.03, 1.36, 1.42, 1.0, 0.6, 1.01, 0.65, 0.63, 0.62, 0.94, 0.48, 0]
[0.55, 0.22, 0.47, 0.42, 0.63, 1.01, 0.61, 0.41, 0.64, 0.2, 0.81, 0.45, 0.3, 0.87, 1.34, 1.58, 1.05, 0.56, 1.07, 0.82, 0.57, 0.74, 0.94, 0.45, 0.54, 0]
[0.32, 0.5, 0.51, 0.55, 0.35, 0.65, 0.46, 0.22, 0.83, 0.45, 0.55, 0.28, 0.57, 0.99, 1.1, 1.22, 0.71, 0.26, 0.82, 0.44, 0.46, 0.33, 0.77, 0.2, 0.41, 0.45, 0]
Manhattan matrix
[0]
[0.29, 0]
[0.26, 0.09, 0]
[0.42, 0.11, 0.06, 0]
[0.02, 0.37, 0.26, 0.42, 0]
[0.38, 1.19, 1.18, 1.36, 0.38, 0]
[0.27, 0.26, 0.07, 0.11, 0.21, 0.99, 0]
[0.03, 0.18, 0.17, 0.25, 0.05, 0.49, 0.18, 0]
[0.85, 0.26, 0.19, 0.09, 0.85, 2.13, 0.3, 0.62, 0]
[0.22, 0.03, 0.1, 0.1, 0.28, 1.02, 0.23, 0.11, 0.31, 0]
[0.14, 0.75, 0.78, 1.0, 0.18, 0.12, 0.75, 0.25, 1.65, 0.62, 0]
[0.14, 0.21, 0.14, 0.14, 0.12, 0.66, 0.09, 0.05, 0.45, 0.12, 0.46, 0]
[0.35, 0.02, 0.07, 0.07, 0.41, 1.35, 0.24, 0.22, 0.18, 0.03, 0.87, 0.21, 0]
[0.99, 0.46, 0.25, 0.27, 0.95, 2.47, 0.38, 0.82, 0.12, 0.53, 1.87, 0.67, 0.34, 0]
[0.78, 1.85, 1.86, 2.34, 0.84, 0.46, 1.85, 1.09, 3.21, 1.72, 0.34, 1.52, 2.05, 3.27, 0]
[1.22, 2.65, 2.52, 2.94, 1.18, 0.38, 2.23, 1.53, 3.99, 2.42, 0.62, 1.86, 2.87, 4.17, 0.3, 0]
[0.3, 1.11, 1.02, 1.36, 0.3, 0.16, 0.91, 0.49, 2.05, 1.02, 0.12, 0.74, 1.27, 2.15, 0.22, 0.38, 0]
[0.01, 0.3, 0.27, 0.43, 0.03, 0.35, 0.26, 0.04, 0.86, 0.25, 0.15, 0.15, 0.38, 1.02, 0.79, 1.19, 0.27, 0]
[0.55, 1.38, 1.53, 1.75, 0.63, 0.11, 1.46, 0.7, 2.6, 1.21, 0.15, 0.99, 1.58, 3.0, 0.31, 0.41, 0.27, 0.54, 0]
[0.11, 0.7, 0.57, 0.75, 0.07, 0.15, 0.42, 0.18, 1.32, 0.57, 0.11, 0.27, 0.78, 1.48, 0.63, 0.73, 0.15, 0.1, 0.4, 0]
[0.19, 0.5, 0.69, 0.77, 0.29, 0.29, 0.74, 0.2, 1.34, 0.39, 0.13, 0.37, 0.62, 1.74, 0.77, 1.17, 0.45, 0.2, 0.26, 0.3, 0]
[0.09, 0.58, 0.49, 0.65, 0.07, 0.17, 0.36, 0.14, 1.18, 0.49, 0.13, 0.23, 0.68, 1.38, 0.71, 0.85, 0.17, 0.06, 0.42, 0.02, 0.26, 0]
[0.42, 0.61, 0.26, 0.5, 0.32, 1.26, 0.21, 0.45, 0.69, 0.6, 0.9, 0.44, 0.57, 0.47, 1.64, 2.14, 0.86, 0.43, 1.75, 0.55, 1.17, 0.55, 0]
[0.22, 0.31, 0.42, 0.42, 0.28, 0.46, 0.39, 0.15, 0.83, 0.28, 0.38, 0.2, 0.43, 1.25, 1.32, 1.62, 0.62, 0.17, 0.65, 0.33, 0.19, 0.21, 0.92, 0]
[0.35, 0.42, 0.41, 0.29, 0.33, 0.69, 0.3, 0.2, 0.66, 0.27, 0.61, 0.09, 0.42, 1.06, 1.85, 2.01, 1.01, 0.36, 1.02, 0.42, 0.4, 0.38, 0.89, 0.23, 0]
[0.3, 0.05, 0.22, 0.18, 0.4, 1.02, 0.37, 0.17, 0.41, 0.04, 0.66, 0.2, 0.09, 0.75, 1.8, 2.5, 1.1, 0.31, 1.15, 0.67, 0.33, 0.55, 0.88, 0.2, 0.29, 0]
[0.1, 0.25, 0.26, 0.3, 0.12, 0.42, 0.21, 0.05, 0.69, 0.2, 0.3, 0.08, 0.33, 0.99, 1.2, 1.5, 0.5, 0.07, 0.67, 0.19, 0.21, 0.11, 0.6, 0.04, 0.17, 0.2, 0]
```

9.Construct dissimilarity matrix for weather nominal data set

```
import csv
import math
A=[]
with open('weather.nominal.csv',newline='') as csvfile:
  for row in csv.reader(csvfile):
    A.append(row)
  t=[]
  for i in A:
    t.append(i[:-1])
  t.pop(0)
  col=len(t[0])

  res=[]
  for i in range(0,len(t)):
    temp=[]
    for j in range(0,i+1):
      m=0
      if i==j:
        temp.append(0)
        continue
      for k in range(col):
        if t[i][k]==t[j][k]:
          m=m+1
      h=(col-m)/col
      temp.append(round(h,2))
    res.append(temp)
  for i in res:
    print(i)
```

```
[0]
[0.25, 0]
[0.25, 0.5, 0]
[0.5, 0.75, 0.5, 0]
[0.75, 1.0, 0.75, 0.5, 0]
[1.0, 0.75, 1.0, 0.75, 0.25, 0]
[1.0, 0.75, 0.75, 1.0, 0.5, 0.25, 0]
[0.25, 0.5, 0.5, 0.25, 0.75, 1.0, 1.0, 0]
[0.5, 0.75, 0.75, 0.75, 0.25, 0.5, 0.5, 0.5, 0]
[0.75, 1.0, 0.75, 0.25, 0.25, 0.5, 0.75, 0.5, 0.5, 0]
[0.75, 0.5, 1.0, 0.75, 0.75, 0.5, 0.5, 0.5, 0.5, 0.5, 0]
[0.75, 0.5, 0.5, 0.5, 1.0, 0.75, 0.5, 0.5, 1.0, 0.75, 0.5, 0]
[0.5, 0.75, 0.25, 0.75, 0.5, 0.75, 0.5, 0.75, 0.5, 0.5, 0.75, 0.75, 0]
[0.75, 0.5, 0.75, 0.25, 0.75, 0.5, 0.75, 0.5, 1.0, 0.5, 0.5, 0.25, 1.0, 0]
```

10.Consider each attribute of iris data set and divide values as equi-depth bins of size 20 each and smooth bins using bin means and bin boundaries

```
import numpy as np
import math
from sklearn.datasets import load_iris
from sklearn import datasets, linear_model, metrics
dataset = load_iris()
a = dataset.data
b = np.zeros(150)
for i in range (150):
  b[i]=a[i,1]
b=np.sort(b)
bin1=np.zeros((30,5))
bin2=np.zeros((30,5))
bin3=np.zeros((30,5))
# Bin mean
for i in range (0,150,5):
  k=int(i/5)
  mean=(b[i] + b[i+1] + b[i+2] + b[i+3] + b[i+4])/5
  for j in range(5):
    bin1[k,j]=mean
print("Bin Mean: \n",bin1)
```

```
# Bin boundaries
for i in range (0,150,5):
  k=int(i/5)
  for j in range (5):
    if (b[i+j]-b[i]) < (b[i+4]-b[i+j]):
      bin2[k,j]=b[i]
    else:
      bin2[k,j]=b[i+4]
print("Bin Boundaries: \n",bin2)
# Bin median
for i in range (0,150,5):
  k=int(i/5)
  for j in range (5):
    bin3[k,j]=b[i+2]
print("Bin Median: \n",bin3)
```

```
Bin Mean:
 [[2.18 2.18 2.18 2.18 2.18]
 [2.34 2.34 2.34 2.34 2.34]
 [2.48 2.48 2.48 2.48 2.48]
 [2.52 2.52 2.52 2.52 2.52]
 [2.62 2.62 2.62 2.62 2.62]
 [2.7  2.7  2.7  2.7  2.7 ]
 [2.74 2.74 2.74 2.74 2.74]
 [2.8  2.8  2.8  2.8  2.8 ]
 [2.8  2.8  2.8  2.8  2.8 ]
 [2.86 2.86 2.86 2.86 2.86]
 [2.9  2.9  2.9  2.9  2.9 ]
 [2.96 2.96 2.96 2.96 2.96]
 [3.   3.   3.   3.   3.  ]
 [3.   3.   3.   3.   3.  ]
 [3.   3.   3.   3.   3.  ]
 [3.   3.   3.   3.   3.  ]
 [3.04 3.04 3.04 3.04 3.04]
 [3.1  3.1  3.1  3.1  3.1 ]
 [3.12 3.12 3.12 3.12 3.12]
 [3.2  3.2  3.2  3.2  3.2 ]
 [3.2  3.2  3.2  3.2  3.2 ]
 [3.26 3.26 3.26 3.26 3.26]
 [3.34 3.34 3.34 3.34 3.34]
 [3.4  3.4  3.4  3.4  3.4 ]
 [3.4  3.4  3.4  3.4  3.4 ]
 [3.5  3.5  3.5  3.5  3.5 ]
```

```
Bin Boundaries:
 [[2.  2.3 2.3 2.3 2.3]
 [2.3 2.3 2.3 2.4 2.4]
 [2.4 2.5 2.5 2.5 2.5]
 [2.5 2.5 2.5 2.5 2.6]
 [2.6 2.6 2.6 2.6 2.7]
 [2.7 2.7 2.7 2.7 2.7]
 [2.7 2.7 2.7 2.8 2.8]
 [2.8 2.8 2.8 2.8 2.8]
 [2.8 2.8 2.8 2.8 2.8]
 [2.8 2.8 2.9 2.9 2.9]
 [2.9 2.9 2.9 2.9 2.9]
 [2.9 2.9 3.  3.  3. ]
 [3.  3.  3.  3.  3. ]
 [3.  3.  3.  3.  3. ]
 [3.  3.  3.  3.  3. ]
 [3.  3.  3.  3.  3. ]
 [3.  3.  3.  3.1 3.1]
 [3.1 3.1 3.1 3.1 3.1]
 [3.1 3.1 3.1 3.1 3.2]
 [3.2 3.2 3.2 3.2 3.2]
 [3.2 3.2 3.2 3.2 3.2]
 [3.2 3.2 3.3 3.3 3.3]
 [3.3 3.3 3.3 3.4 3.4]
 [3.4 3.4 3.4 3.4 3.4]
 [3.4 3.4 3.4 3.4 3.4]
 [3.5 3.5 3.5 3.5 3.5]
Bin Median:
 [[2.2 2.2 2.2 2.2 2.2]
 [2.3 2.3 2.3 2.3 2.3]
 [2.5 2.5 2.5 2.5 2.5]
 [2.5 2.5 2.5 2.5 2.5]
 [2.6 2.6 2.6 2.6 2.6]
 [2.7 2.7 2.7 2.7 2.7]
 [2.7 2.7 2.7 2.7 2.7]
 [2.8 2.8 2.8 2.8 2.8]
 [2.8 2.8 2.8 2.8 2.8]
 [2.9 2.9 2.9 2.9 2.9]
 [2.9 2.9 2.9 2.9 2.9]
 [3.  3.  3.  3.  3. ]
 [3.  3.  3.  3.  3. ]
 [3.  3.  3.  3.  3. ]
 [3.  3.  3.  3.  3. ]
 [3.  3.  3.  3.  3. ]
 [3.  3.  3.  3.  3. ]
 [3.1 3.1 3.1 3.1 3.1]
 [3.1 3.1 3.1 3.1 3.1]
 [3.2 3.2 3.2 3.2 3.2]
 [3.2 3.2 3.2 3.2 3.2]
```

11.Form a binary data set with 10 attributes and 20 records and find the dissimilarity matrix by considering attributes are symmetric and find dissimilarity matrix by considering attributes are asymmetric(1 is more important than 0)

```
a = [0, 1, 1, 0]
b = [1, 1, 1, 0]
def jaccard_similarity(A, B):
    q=len([i for i, j in zip(a, b) if i==1 and j==1])
    r=len([i for i, j in zip(a, b) if i==1 and j==0])
    s=len([i for i, j in zip(a, b) if i==0 and j==1])
    t=len([i for i, j in zip(a, b) if i==0 and j==0])
    disim_symmetric=(r+s)/(q+r+s+t)
    disim_asym=(q)/(q+r+s)
    print("disim_symmetric",disim_symmetric)
    print("disim_asym",1-disim_asym)
jaccard_similarity(a,b)
```

```
disim_symmetric 0.25
disim_asym 0.33333333333333337
```

12.Draw quantile plot for every iris data set attributes

```
import statsmodels.api as sm
import matplotlib.pyplot as plt
from sklearn import datasets
import numpy as np
iris = datasets.load_iris()
i=iris['data']
sm.qqplot(i,line="45",fit=True)
plt.show()
```

VASAVI COLLEGE OF ENGINEERING

AUTONOMOUS

(Affiliated to Osmania University)

Hyderabad - 500 031.

DEPARTMENT OF : CSE

NAME OF THE LABORATORY :

Name P. Pranaya **Roll No.** 1602-19-733-037 **Page No:**