**EXPERIMENT 1**

**Question:** Write a program of ML in python to find Mean, Median & Mode

**Aim:** The aim of the program is to calculate & display Mean, Median & Mode

**Procedure:**

#Mean

import numpy as np

a=[36,67,89,45,63,71]

x=np.mean(a)

print(x)

**Output:**

61.833333333333336

#Median

import numpy as np

a=[36,67,89,45,63,71]

x=np.median(a)

print(x)

**Output:**

65.0

#Mode

from scipy import stats

a=[36,67,89,45,63,71,45]

x=stats.mode(a)

print(x)

**Output:**

ModeResult(mode=array([45]), count=array([2]))

**Result: -** Program executed successfully.

**EXPERIMENT 2**

**Question:** Write a program of ML in python to find variance and standard deviation

**Aim:** The aim of the program is to calculate & display variance and standard deviation

**Procedure:**

**#**variance

import numpy as np

a=[36,67,89,45,63,71]

x=np.var(a)

print(x)

**Output:**

300.13888888888886

#std deviation

import numpy as np

a=[36,67,89,45,63,71]

x=np.std(a)

print(x)

**Output:**

17.324516988617283

**Result:** Program executed successfully.

**EXPERIMENT 3**

**Question:** Write a program to Data Distribution in ML Python

**Aim:** The aim of the program is to plot a data distribution graph in ML Python

**Procedure:**

#Draw a histogram

import numpy as np

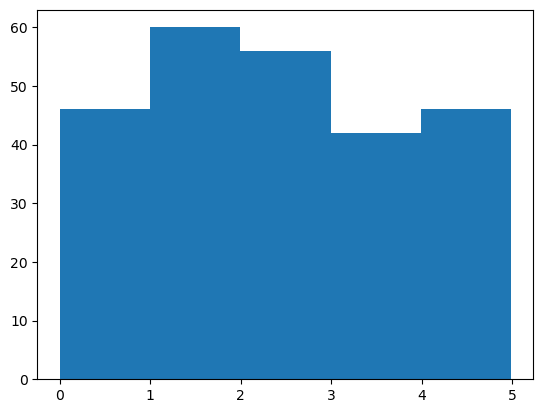
import matplotlib.pyplot as plt

x=np.random.uniform(0.0,5.0,250)

plt.hist(x,5)

plt.show()

**Output:**

****

#Big data distribution

import numpy as np

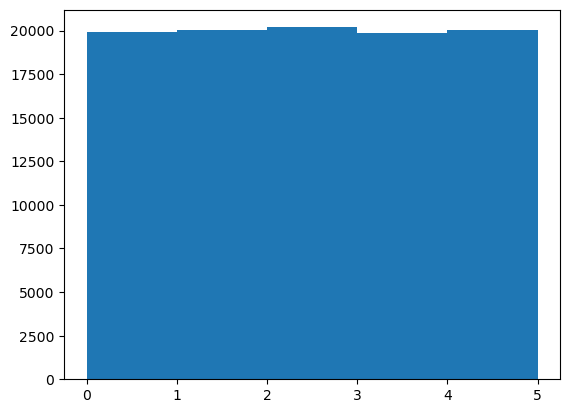
import matplotlib.pyplot as plt

x=np.random.uniform(0.0,5.0,100000)

plt.hist(x,5)

plt.show()

**Output:**

****

**Result:** Program executed successfully.

**EXPERIMENT 4**

**Question:** Write a program to Normal Data Distribution in ML Python

**Aim:** The aim of this program to plot a normal data distribution graph in ML Pyhton

**Procedure:**

#Normal data distribution

import numpy as np

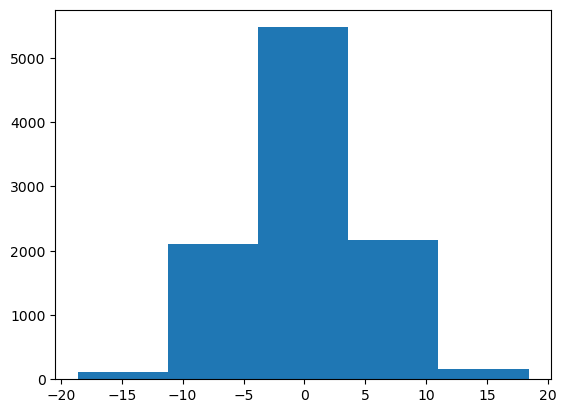
import matplotlib.pyplot as plt

x=np.random.normal(0.0,5.0,10000)

plt.hist(x,5)

plt.show()

**Output:**

****

**Result:** Program executed successfully.

**EXPERIMENT 5**

**Question:** Write a program for scatter plot in ML python

**Aim:** The aim of this program is to plot a scatter graph in ML python

**Procedure:**

#Scatter plot

import numpy as np

import matplotlib.pyplot as plt

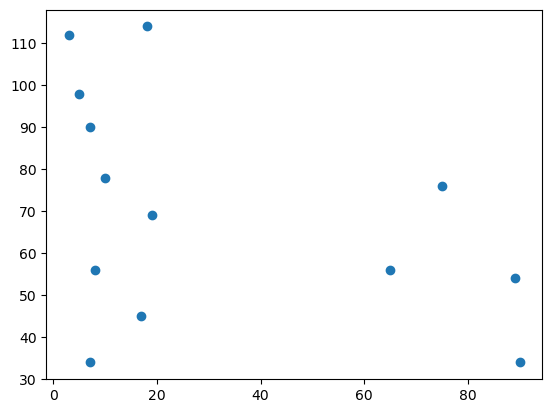
x= [3,7,8,10,5,17,18,19,7,75,89,90,65]

y= [112,34,56,78,98,45,114,69,90,76,54,34,56]

plt.scatter(x,y)

plt.show

**Output:**



#Normal distribution Scatter plot

import numpy as np

import matplotlib.pyplot as plt

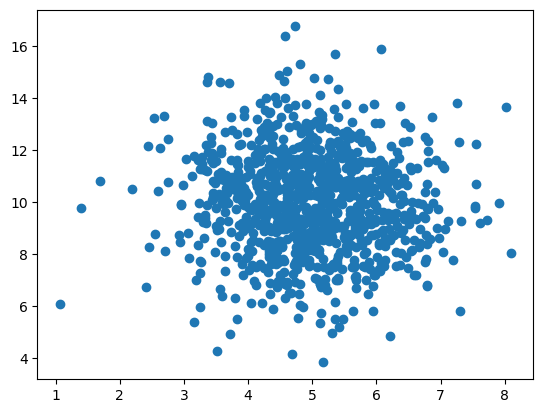
x= np.random.normal(5.0,1.0,1000)

y= np.random.normal(10.0,2.0,1000)

plt.scatter(x,y)

plt.show

**Output:**



**Result: -** We successfully plot the scatter plot.

**EXPERIMENT 6**

**Question:** Write a program to calculate the Linear Reggression in Python

**Aim:** The aim of this program is to calculate the Linear Reggression in Python

**Procedure:**

from sklearn.linear\_model import LinearRegression

x=np.array([[98],[78],[45],[89],[70],[66],[32],[100],[55],[43]])

y=np.array([8,5,2,9,6,5.5,1,10,2,1.6])

model = LinearRegression()

model.fit(x,y)

print("Linear Regression Intercept:", model.intercept\_)

print("Linear Regression Coefficient :\n", model.coef\_)

**Output:**

Linear Regression Intercept: -3.9146236052176615

Linear Regression Coefficient :

[0.13202106]

**Result:**

We successfully got the intercept and coefficient of Linear Regression model

**EXPERIMENT - 7**

**Question:** Write a program to plot Histogram, Box plot, Scatter Plot

**Aim:** The aim of this program is to plot histogram, boxplot and scatter plot.

**Procedure:**

# Histogram

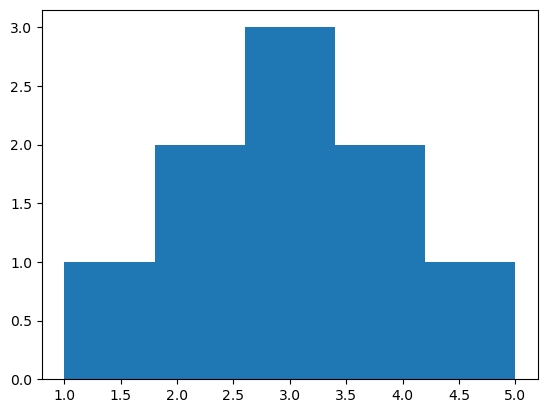
import matplotlib.pyplot as plt

data = [1,2,2,3,3,3,4,4,5]

plt.hist(data,bins=5)

plt.show()

**Output:**



# Box Plot

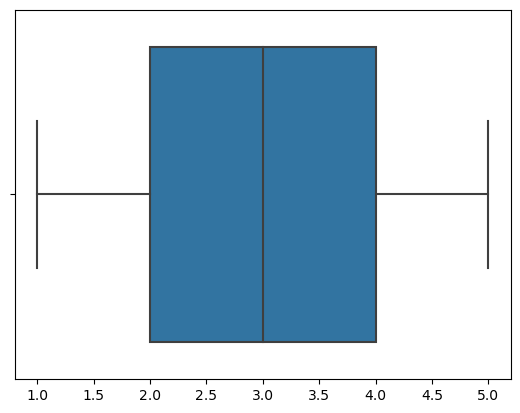
import seaborn as sns

data = [1,2,2,3,3,3,4,4,5]

sns.boxplot(data)

plt.show()

**Output:**



# Scatter Plot

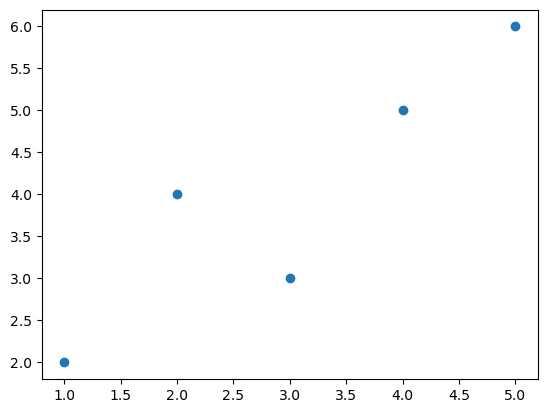
x = [1,2,3,4,5]

y = [2,4,3,5,6]

plt.scatter(x,y)

plt.show()

**Output:**



**Result: -** We successfully plot the histogram, boxplot and scatterplot in python.

**EXPERIMENT - 8**

**Question:** Write a Program to Calculate and visualize a correlation matrix using pandas and seaborn.

**Aim:** The aim of this program is to calculate and visualize a correlation matrix using pandas and seaborn python library.

**Procedure:**

import pandas as pd

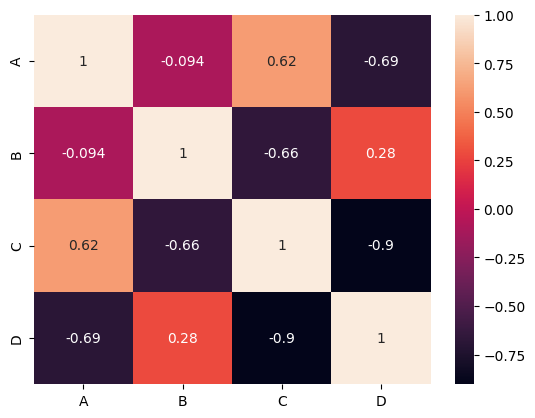
import seaborn as sns

data = {'A':[34,76,20,23],'B':[87,43,40,45],'C':[10,98,30,76],'D':[49,19,60,28]}

df = pd.DataFrame(data)

sns.heatmap(df.corr(),annot=True,)

**Output:**

****

**Result:** We successfully plot the correlation matrix using pandas and seaborn.

**EXPERIMENT - 9**

**Question:** Write a program to handle missing values using pandas (Data Cleaning).

**Aim:** The aim of this program is to handle (fill or drop) missing values using pandas.

**Procedure:**

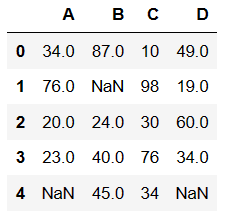
import pandas as pd

data = {'A':[34,76,20,23,None],'B':[87,None,24,40,45],'C':[10,98,30,76,34],'D':[49,19,60,34,None]}

df = pd.DataFrame(data)

print(df)

**Output (Original Dataframe):**

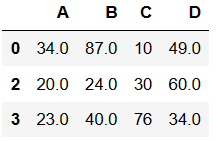
****

1. **Remove Missing Values:**

df\_without\_mv = df.dropna(axis=0)

df\_without\_mv

**Output:**

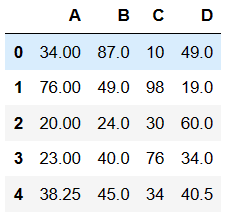


1. **Fill Missing Values:**

**Output:**

df\_filled\_values = df.fillna(df.mean())

df\_filled\_values



**Result:** We successfully handle the missing values using pandas

**EXPERIMENT - 10**

**Question:** Write a program to Normalize data using scikit-learn.

**Aim:** The aim of this program is to normalize data using python scikit-learn library.

**Procedure:**

import pandas as pd

from sklearn.preprocessing import MinMaxScaler

data = {'A':[34,76,20,23],'B':[87,43,40,45],'C':[10,98,30,76],'D':[49,19,60,28]}

df = pd.DataFrame(data)

scaler = MinMaxScaler()

scaled\_data = scaler.fit\_transform(df)

print('Original Data:\n',data)

print('Scaled Data:\n',scaled\_data)

**Output:**

Original Data:

{'A': [34, 76, 20, 23], 'B': [87, 43, 40, 45], 'C': [10, 98, 30, 76], 'D': [49, 19, 60, 28]}

Scaled Data:

[[0.25 1. 0. 0.73170732]

[1. 0.06382979 1. 0. ]

[0. 0. 0.22727273 1. ]

[0.05357143 0.10638298 0.75 0.2195122 ]]

**Result:** We successfully normalized the data using python scikit-learn library.

**EXPERIMENT - 11**

**Question:** Write a Program to Perform K-Means clustering using scikit-learn.

**Aim:** The aim of this program is to perform k-means clustering using scikit-learn.

**Procedure:**

from sklearn.cluster import KMeans

kmean = KMeans(n\_clusters=2)

data = [[28],[76],[51],[48],[34],[40],[87],[67],[75],[39]]

kmean.fit(data)

pred = kmean.predict(data)

print('Prediction: Cluster alloted to each data points are:\n',pred)

**Output:**

Prediction: Cluster alloted to each data points are:

[0 1 0 0 0 0 1 1 1 0]

**Result:** We successfully executed the k-means clustering program, which divide each data points into cluster.

**EXPERIMENT - 12**

**Question:** Write a Program to implement a decision tree classifier using scikit-learn.

**Aim:** The aim of this program is to implement a decision tree classifier using python scikit-learn library.

**Procedure:**

from sklearn.tree import DecisionTreeClassifier

dtree = DecisionTreeClassifier()

X\_train = [[28],[76],[51],[48],[34],[40],[87],[67],[75],[39]]

y\_train = [[0],[1],[1],[0],[0],[0],[1],[1],[1],[0]]

dtree.fit(X\_train,y\_train)

# Testing Data

X\_test = [[46],[54],[50],[42]]

pred = dtree.predict(X\_test)

# Prediction

print('Prediction of Test Data:\n',pred)

**Output:**

Prediction of Test Data:

[0 1 1 0]

**Result:** We successfully implemented a decision tree classifier model, which divide the data points into classes.

**EXPERIMENT - 13**

**Question:** Write a Program to build a random forest classifier using scikit-learn.

**Aim:** The aim of this program is to build a random forest classifier using python scikit-learn library.

**Procedure:**

from sklearn.ensemble import RandomForestClassifier

rfc = RandomForestClassifier()

X\_train = [[28],[76],[51],[48],[34],[40],[87],[67],[75],[39]]

y\_train = [[0],[1],[1],[0],[0],[0],[1],[1],[1],[0]]

rfc.fit(X\_train,y\_train)

# Testing Data

X\_test = [[46],[54],[50],[42]]

pred = rfc.predict(X\_test)

# Prediction

print('Prediction of Test Data:\n',pred)

**Output:**

Prediction of Test Data:

[0 1 1 0]

**Result:** We successfully implemented a random forest classifier model, which divide the data points into classes.

**EXPERIMENT - 14**

**Question:** Write a Program to apply PCA for dimensionality reduction using scikit-learn.

**Aim:** The aim of this program is to apply PCA for reduce the data dimensions using python scikit-learn library.

**Procedure:**

from sklearn.decomposition import PCA

import numpy as np

pca = PCA(n\_components=2)

data = np.array([[34,76,20,23],[87,43,40,45],[10,98,30,76],[49,19,60,28]])

transformed\_data = pca.fit\_transform(data)

print('Original Data:\n',data)

print('Data After Appling PCA:\n',transformed\_data)

**Output:**

Original Data:

[[34 76 20 23]

[87 43 40 45]

[10 98 30 76]

[49 19 60 28]]

Data After Appling PCA:

[[ 16.38102936 -10.97233447]

[-35.47704737 27.66701872]

[ 60.25710461 4.67481146]

[-41.1610866 -21.36949571]]

**Result:** We successfully applied PCA to reduce the dimensions of the data using python scikit-learn library.

**EXPERIMENT - 15**

**Question:** Write a Program to create a time series plot using pandas and matplotlib.

**Aim:** The aim of this program is to create a time-series plot using python pandas and matplotlib library.

**Procedure:**

import pandas as pd

import matplotlib.pyplot as plt

data = {'Date': pd.date\_range('2024-01-01', periods=7), 'Value': [90, 115, 80, 120, 85, 97, 150]}

df = pd.DataFrame(data)

plt.figure(figsize=(12,8))

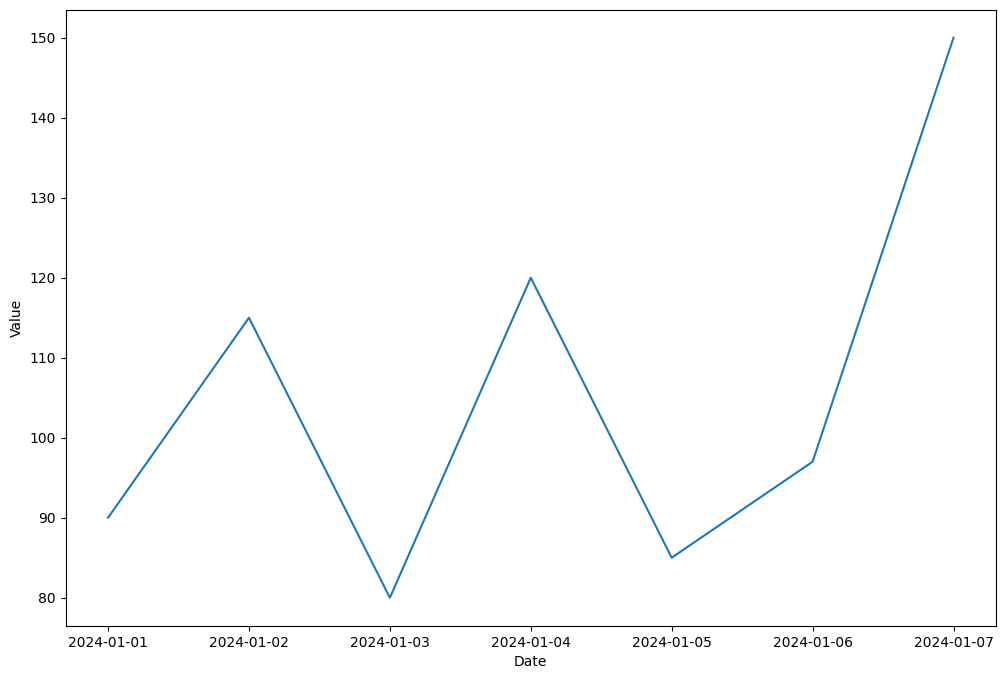
plt.plot(df['Date'], df['Value'])

plt.xlabel('Date')

plt.ylabel('Value')

plt.show()

**Output:**



**Result:** We successfully plot the time-series graph using python pandas and matplotlib library.

**EXPERIMENT - 16**

**Question:** Write a Program to load a dataset using pandas and Basic exploration of dataset properties.

**Aim:** The aim of this program is to load a dataset using python pandas library and do basic exploration of dataset.

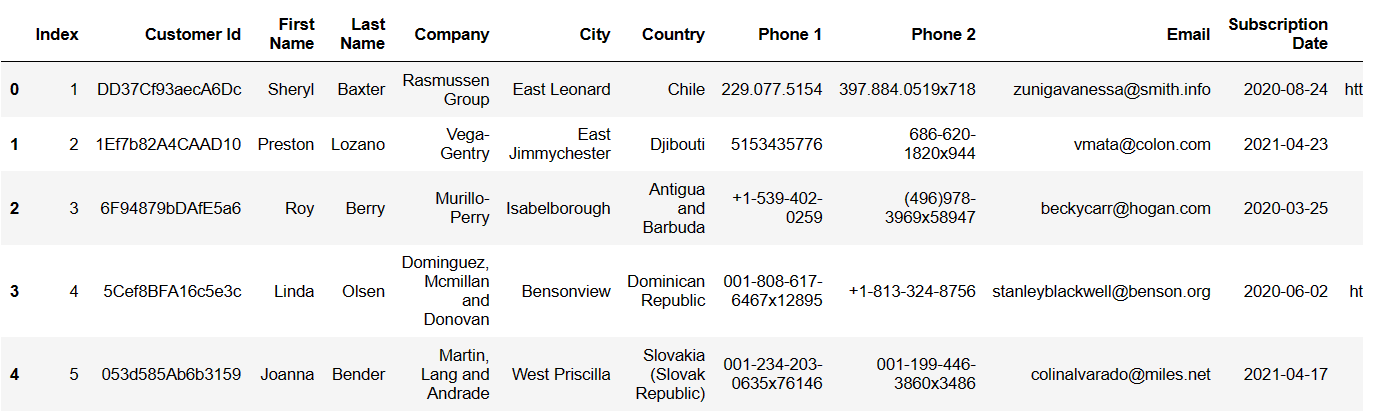
**Procedure:**

import pandas as pd

df = pd.read\_csv('customers-100.csv')

df.head()

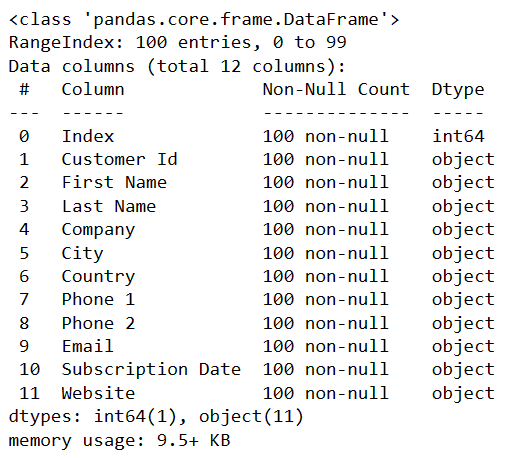
**Output 1 (Top 5 records of dataset):**

****

**Dataset Information:**

df.info()

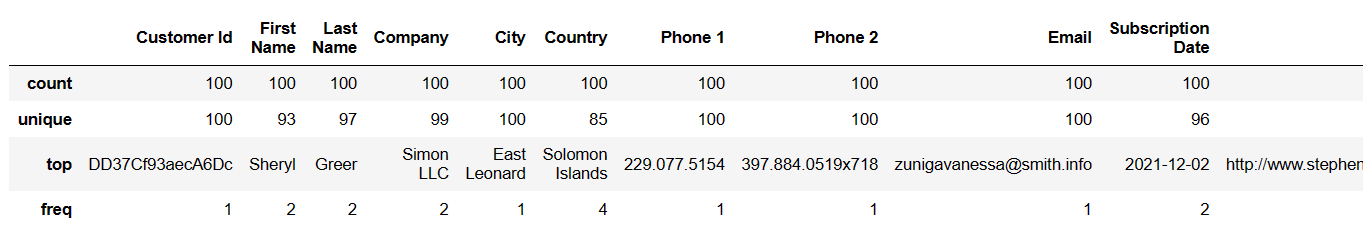
**Output 2 (Information of dataset):**



**Dataset Description:**

df.describe(include=’object’)

**Output 3 (Description of dataset):**

****

**Result:** We successfully load the dataset using python pandas library and do some basic exploration of dataset.