



**MADRAS INSTITUTE OF TECHNOLOGY  
ANNA UNIVERSITY**



**DEPARTMENT OF INFORMATION TECHNOLOGY  
IT5612- DATA ANALYTICS AND CLOUD COMPUTING  
LABORATORY**

**RECORD**

**REGISTER NUMBER: 2019506076**

**NAME: ROZEN BERG D**

**SEMESTER: 6/8**

**DEPARTMENT OF INFORMATION TECHNOLOGY  
ANNA UNIVERSITY, MIT CAMPUS**

CHROME PET, CHENNAI – 600 044

## **BONAFIDE CERTIFICATE**

Certified that the bonafide record of the practical work done by  
Mr. Rozen Berg D Register Number (2019506076) of **Sixth Semester, B.Tech**  
**Information Technology** in the **IT5612-Data Analytics and Cloud Computing**  
**Laboratory** during the academic period from March 2022 to June 2022

Submitted for Practical Examination held on

Date: \_\_\_\_\_ Course Instructor \_\_\_\_\_  
S. Eliza Femi Sherley

## **Internal Examiner**

## **External Examiner**

## TABLE OF CONTENTS

<b>S. NO</b>	<b>DATE</b>	<b>TITLE</b>	<b>PAGE NO</b>	<b>SIGNATURE</b>
1A	08/03/2022	Study on NumPy, Pandas, SciPy and Statsmodels Packages	4	
1B	08/03/2022	Read data from Text file, Excel (csv) file and the web using inbuilt packages	7	
2	15/03/2022	Descriptive Analysis	9	
2 A	22/03/2022	Univariate Analysis	26	
2 B	29/03/2022	Univariate time series Analysis	50	
3	29/03/2022	Openstack	63	
3 A	05/04/2022	Openstack Installation	65	
4	09/04/2022	Creation of VMs in Openstack	69	
5	12/04/2022	MongoDB - Basics	73	
6	19/04/2022	Bivariate Analysis	81	

## STUDY ON NUMPY, SCIPY , STATSMODELS AND PANDAS PACKAGES

### Aim:

To explore the functions in Numpy , SciPy , Statsmodels and pandas packages and prepare a study with an example for each function

### NumPy:

S.No.	Function	Description	Example
1	array()	Used to create a Numpy array object	arr=np.array([1,2,3,4,5]) print(arr) <u>Output:</u> [1 2 3 4 5]
2	name[index]	It is used to access array element at specific index	arr=np.array([1,2,3,4,5]) print(arr[0]) <u>Output:</u> 1
3	dtype	Returns data type of the array	arr=np.array([1,2,3,4,5]) print(arr.dtype) <u>Output:</u> int64
4	copy()	Makes a copy of original array	arr=np.array([1,2,3,4,5]) x=arr.copy() arr[0]=4 print(arr) print(x) <u>Output:</u> [4 2 3 4 5] [1 2 3 4 5]
5	tan()	Helps calculate trigonometric tangent for all parameter	np.tan(np.array([pi,pi/2,pi])) <u>Output:</u> array([1.22e-16,1.63e+16,-1.22e-16])

### SciPy:

Scipy is a scientific computation library that uses Numpy underneath

#### SciPy Constants:

```
from scipy import constants
```

```
print(constants.pi)
```

#### Output

3.141592653589793

## **SciPy Optimizers:**

Optimize.root helps us to find the root of the equation

### Example:

```
from scipy.optimize import root  
from math import os  
  
def eqn(x) :  
  
    return x+cos(x)  
  
myroot=root(eqn,0)
```

### Output:

[-0.73908513]

## **SciPy Sparse Data:**

Sparse data is data that has unused elements

### Example:

```
import numpy as np  
from scipy.sparse import csr_matrix  
arr=np.array([0,0,0,0,0,1,1,0,2])  
print(csr_matrix(arr))
```

### Output:

(0,5) 1  
(0,6) 1  
(0,8) 2

## **Statsmodels:**

Statsmodel is a python module that provides classes and functions for the estimation of many different statistical models, as well as for conducting statistical tests and statistical data exploration. An extensive list of result statistics is available for each estimator. The results are tested against existing statistical packages to ensure that they are correct.

OLS() – It is used to estimate by ordinary least squares

OLS() function can be used for

- a) Regression models
- b) Linear models
- c) Time Series Analysis
- d) Contingency tables

## **Pandas:**

Pandas is a Python library used for working with data sets. It has functions for analyzing, cleaning , exploring and manipulating data.

### Pandas DataFrame

Pandas DataFrame is a 2D data structure

#### Example

```
import pandas as pd  
  
data = {  
    "calories": [420],  
    "duration": [50]  
}  
  
df=pd.DataFrame(data)  
  
print(df)
```

#### Output:

```
0  calories  duration  
420      50
```

#### Read and Analyse:

read\_csv() - It is used to read CSV files

read\_json() - It is used to read JSON files

head() - We can view first rows of data frame

tail() - We can view last rows of data frame

## **Result:**

Study of basic Python packages have been done successfully.

## READ DATA/DATASET

### Aim:

To use inbuilt packages to read data from text file, excel and the web

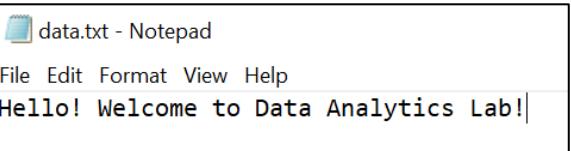
### Code:

#### 1)a)Read data from text file:

```
read=open("data.txt",'r')
for i in read:
    print(i)
```

### Output:

Contents of data.txt and reading:

	<pre>In [1]: read=open("data.txt",'r') for i in read:     print(i) Hello! Welcome to Data Analytics Lab!</pre>
--	--

#### b)Read data using numpy:

```
import numpy as np
data = np.loadtxt('data.txt',dtype='str')
for each in data:
    print(each)
```

### Output:

<pre>In [2]: import numpy as np data = np.loadtxt('data.txt',dtype='str') for each in data:     print(each)</pre>	<pre>Hello! Welcome to Data Analytics Lab!</pre>
---	--

#### 2)Read from web:

```
import urllib.request
url="https://google.com"
req=urllib.request.Request(url)
res=p=urllib.request.urlopen(req)
```

```
data=res.read()  
print(data)
```

## Output:

b`<!doctype html><html itemscope="" itemtype="http://schema.org/WebPage" lang="en-IN"><head><meta content="text/html; charset=UTF-8" http-equiv="Content-Type"><meta content="images/branding/googlelog/1x/googleleg\_standard\_color\_128dp\_50px" itemprop="image" /><title>Google</title><script nonce="c+GPjEk5/uiaGYMx/839Wg==">(function(){window.google=[{KE1:DkcvWvCSCjDe2roPnLyB4Ac},kEXPI:\0,1302536,56873,1709,4349,207,2415,2389,2316,383,246,5,1354,4013,1238,2944,1119571,1197752,670,16,380053,16114,28684,17572,4858,132,9291,3027,4746,12835,4020,978,13282,8039,6430,7432,15309,261,649,5764,1279,2742,149,1103,840,2196,4101,3514,606,2023,1777,520,14670,3228,2844,8,4810,12639,7540,4085,4143,552,8248,9359,3,346,230,1014,1,5444,149,11323,2652,4,1253,275,655,1649,7039,22023,3050,2658,7356,31,18065,9358,7428,651,5146,2560,4094,17,4035,3,3541,1,16807,38,25309,2,14022,1931,784,255,4550,744,5852,9868,595,1160,5679,1020,2811,2719,18242,2,2,5,7772,4568,6252,17116,6308,1252,5835,11862,3106,1542,2790,8,2269,2400,1412,1395,445,2,2,1,6394,566,3829,3051,9909,34,44,703,248,1407,10,1,8591,113,2453,2600,5,466,945,798,1,2,460,2,2579,3550,3878,421,2769,83,936,255,753,883,5450,14,2201,583,420,68,117,2,2,5,1209,1450,2,1,466,140,203,1937,3083,109,1,341,598,321,1295,408,2,1168,238,3,87,1090,719,478,756,329,498,171,303,761,1105,1095,298,666,610,49,2,72,681,33,532,343,46,230,601,473,173,1212,1252,5462973,859,5996732,2800420,882,444,3,1877,1,2562,1,748,141,795,563,1,4265,1,1,2,331,4142,2609,155,17,13,72,139,4,2,2,0,2,169,13,19,46,5,39,96,548,29,2,2,1,2,1,2,7,4,1,2,2,2,2,2,353,513,186,1,1,158,3,2,2,2,2,4,2,3,3,269,1601,141,577,425,32,19,29,1,74,21,2,23951801,4041689,3,450,1964,1008,483,9,1435,159,1358,4726,3,923,32,12,215,489,1803,1532,2442,769966\`,kB!`L:\skq81\`;google.su='wbehyp';google.KHL='en-IN'\});}(function(){\nvar f=this||self;var h=k=[],function l(a){for(var b=a;b!=null;a=&&(l(a.getAttribute('id'))))a=a.parentNode;return b};\nfunction b|h|function m(a){for(var b=null;a&&(l(a.getAttribute('id'))!=b.getAttribute('leid')));a=a.parentNode;return b};\nfunction n(a,b,c,d){var e='';c||=b;search('=&i'+e+'=&i'+l(d),1==b.search('=&lei')&&(d=m(d))&&(e+'=&i'+d));d='';c=&f.\_cshid&1==b.search('=&cshid')&&'88'!=l(h)=&&d!='&cshid='+\\$.cshid;e=c||'+'+(g|`gen\_204`)?atyp=i&t='+'&cad='+'b+e+'&xz='+Date.now():'&https=';window.location.href='http://i.test(c)&'+e});\n});

### 3) Read from excel:

```
import pandas as pd
```

```
data1=pd.read_excel('dow_jones_index.xlsx')
```

```
print(data1)
```

## Output:

	quarter	stock	date	open	high	low	close	volume
0	1	AA	1/7/2011	\$15.82	\$16.72	\$15.78	\$16.42	239655616
1	1	AA	1/14/2011	\$16.71	\$16.71	\$15.64	\$15.97	242963398
2	1	AA	1/21/2011	\$16.19	\$16.38	\$15.60	\$15.79	138428495
3	1	AA	1/28/2011	\$15.87	\$16.63	\$15.82	\$16.13	151379173
4	1	AA	2/4/2011	\$16.18	\$17.39	\$16.18	\$17.14	154387761
..	...	...	...	...	...	...	...	...
745	2	XOM	5/27/2011	\$80.22	\$82.63	\$80.07	\$82.63	68230855
746	2	XOM	6/3/2011	\$83.28	\$83.75	\$80.18	\$81.18	78616295
747	2	XOM	6/10/2011	\$80.93	\$81.87	\$79.72	\$79.78	92380844
748	2	XOM	6/17/2011	\$80.00	\$80.82	\$78.33	\$79.02	100521400
749	2	XOM	6/24/2011	\$78.65	\$81.12	\$76.78	\$76.78	118679791

## Result:

Reading data using inbuilt packages have been done successfully and the results are verified.

## DESCRIPTIVE ANALYTICS

### 1)Aim:

To use Iris Dataset for exploring various commands for doing descriptive analytics

### Code & Output:

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

### Output:

Out[5]:	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 5 columns

data.info()

### Output:

In [6]:	data.info()
	<pre>&lt;class 'pandas.core.frame.DataFrame'&gt; RangeIndex: 150 entries, 0 to 149 Data columns (total 5 columns):  #   Column      Non-Null Count  Dtype   ---   0   sepal_length  150 non-null    float64  1   sepal_width   150 non-null    float64  2   petal_length  150 non-null    float64  3   petal_width   150 non-null    float64  4   species       150 non-null    object  dtypes: float64(4), object(1) memory usage: 6.0+ KB</pre>

```
data.describe()
```

**Output:**

	sepal_length	sepal_width	petal_length	petal_width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

```
data[data.duplicated()]
```

**Output:**

	sepal_length	sepal_width	petal_length	petal_width	species
34	4.9	3.1	1.5	0.1	Iris-setosa
37	4.9	3.1	1.5	0.1	Iris-setosa
142	5.8	2.7	5.1	1.9	Iris-virginica

```
data['species'].value_counts()
```

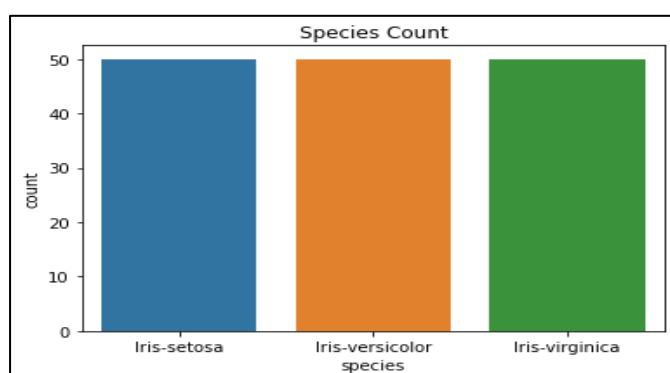
**Output:**

	species	count
	Iris-setosa	50
	Iris-versicolor	50
	Iris-virginica	50

```
plt.title("Species Count")
```

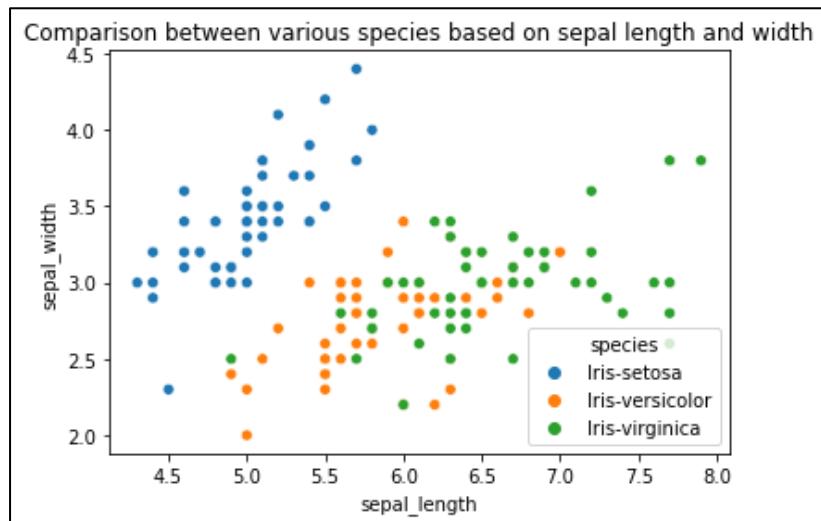
```
sns.countplot(data['species'])
```

**Output:**



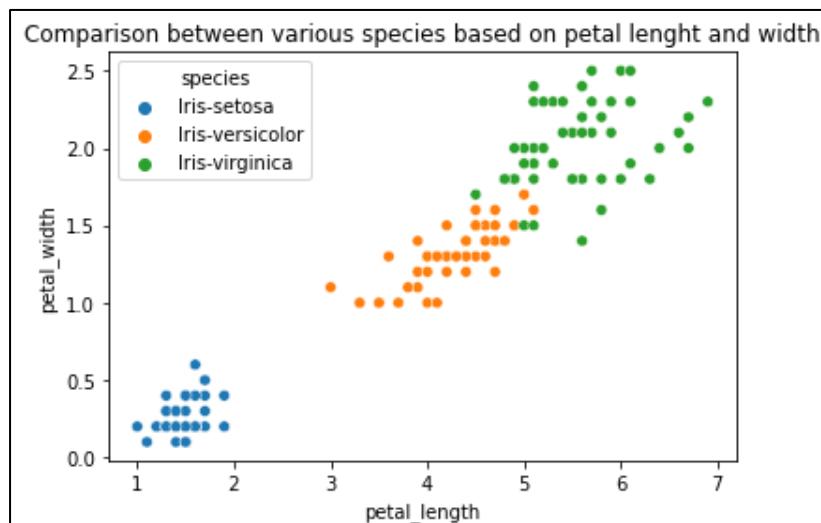
```
plt.title("Comparison between various species based on sepal length and width")
sns.scatterplot(data['sepal_length'],data['sepal_width'],hue=data['species'])
```

**Output:**



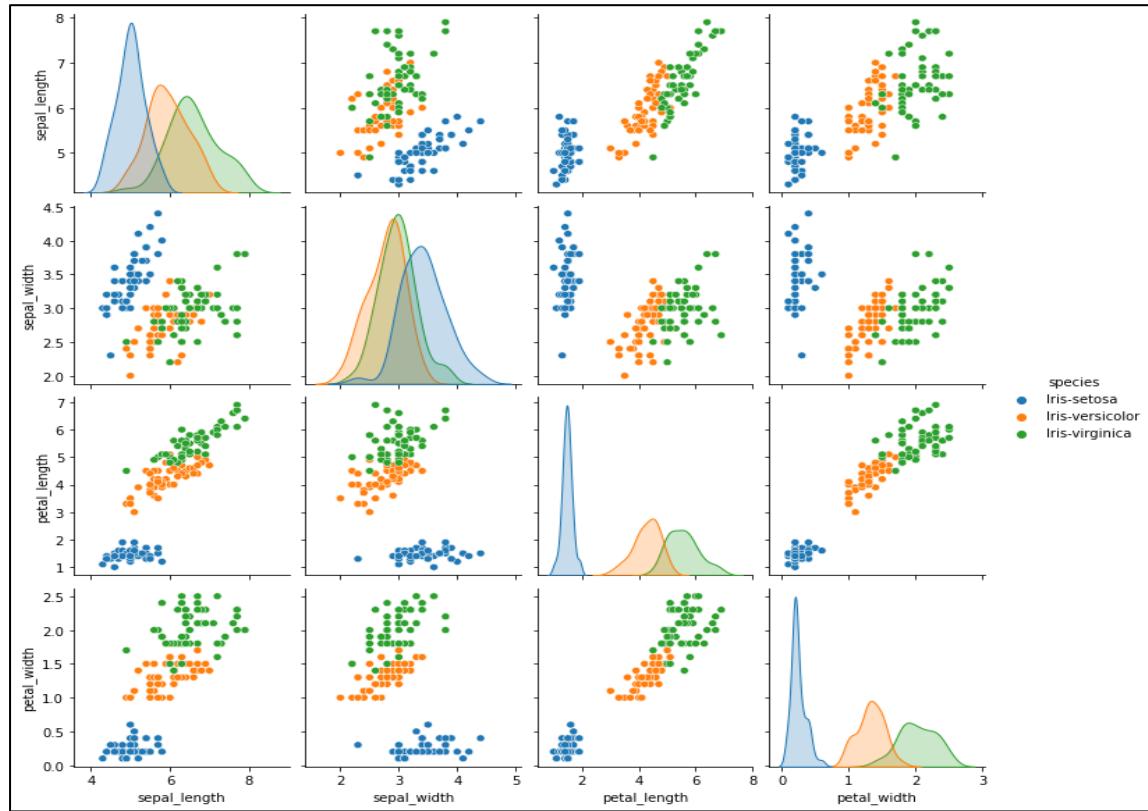
```
plt.title("Comparison between various species based on petal lenght and width")
sns.scatterplot(data['petal_length'],data['petal_width'],hue=data['species'])
```

**Output:**



```
sns.pairplot(data,hue="species")
```

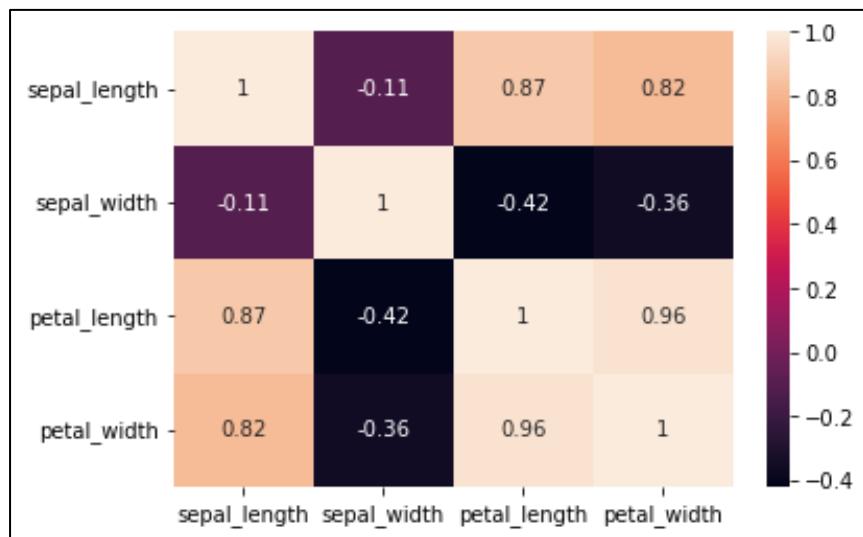
**Output:**



```
sns.heatmap(data.corr(),annot=True)
```

```
plt.plot()
```

**Output:**



```
data.groupby('species').agg(['mean','median'])
```

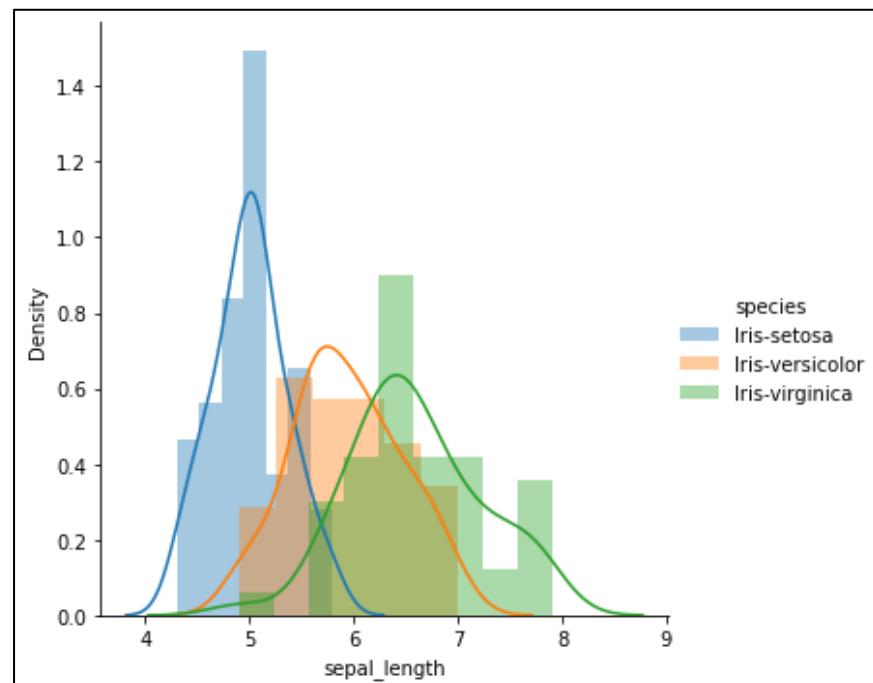
### Output:

In [19]:	data.groupby('species').agg(['mean','median'])							
Out[19]:	sepal_length		sepal_width		petal_length		petal_width	
	mean	median	mean	median	mean	median	mean	median
species								
Iris-setosa	5.006	5.0	3.418	3.4	1.464	1.50	0.244	0.2
Iris-versicolor	5.936	5.9	2.770	2.8	4.260	4.35	1.326	1.3
Iris-virginica	6.588	6.5	2.974	3.0	5.552	5.55	2.026	2.0

```
sns.FacetGrid(data, hue="species",height=5).map(sns.distplot, "sepal_length").add_legend()
```

```
plt.show()
```

### Output:



### **Result:**

Iris Dataset has been successfully used for exploring various commands to do descriptive analytics and the results are verified.

## 2)Aim:

To use personal (Dow Jones Index) dataset for exploring various commands for doing descriptive analytics

## Code & Output:

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

```
data=pd.read_csv("dow_jones_index.csv")
```

```
data
```

## Output:

Out[2]:	quarter	stock	date	open	high	low	close	volume	percent_change_price	percent_change_volume_over_last_wk	pr
0	1	AA	1/7/2011	\$15.82	\$16.72	\$15.78	\$16.42	239655616	3.79267	NaN	
1	1	AA	1/14/2011	\$16.71	\$16.71	\$15.64	\$15.97	242963398	-4.42849	1.380223	
2	1	AA	1/21/2011	\$16.19	\$16.38	\$15.60	\$15.79	138428495	-2.47066	-43.024959	
3	1	AA	1/28/2011	\$15.87	\$16.63	\$15.82	\$16.13	151379173	1.63831	9.355500	
4	1	AA	2/4/2011	\$16.18	\$17.39	\$16.18	\$17.14	154387761	5.93325	1.987452	
...	...	...	...	...	...	...	...	...	...	...	
745	2	XOM	5/27/2011	\$80.22	\$82.63	\$80.07	\$82.63	68230855	3.00424	-21.355713	
746	2	XOM	6/3/2011	\$83.28	\$83.75	\$80.18	\$81.18	78616295	-2.52161	15.221032	
747	2	XOM	6/10/2011	\$80.93	\$81.87	\$79.72	\$79.78	92380844	-1.42098	17.508519	
748	2	XOM	6/17/2011	\$80.00	\$80.82	\$78.33	\$79.02	100521400	-1.22500	8.811952	
749	2	XOM	6/24/2011	\$78.65	\$81.12	\$76.78	\$76.78	118679791	-2.37762	18.064204	

```
data['high'] = data['high'].str.replace('$', '')
```

```
data['open'] = data['open'].str.replace('$', '')
```

```
data['low'] = data['low'].str.replace('$', '')
```

```
data['close'] = data['close'].str.replace('$', '')
```

```
data['next_weeks_open'] = data['next_weeks_open'].str.replace('$', '')
```

```
data['next_weeks_close'] = data['next_weeks_close'].str.replace('$', '')
```

```
data
```

## Output:

quarter	stock	date	open	high	low	close	volume	percent_change_price	percent_change_volume_over_last_wk	previous_weeks_volume	
0	1	AA	1/7/2011	15.82	16.72	15.78	16.42	239655616	3.79267	NaN	NaN
1	1	AA	1/14/2011	16.71	16.71	15.64	15.97	242963398	-4.42849	1.380223	239655616.0
2	1	AA	1/21/2011	16.19	16.38	15.60	15.79	138428495	-2.47066	-43.024959	242963398.0
3	1	AA	1/28/2011	15.87	16.63	15.82	16.13	151379173	1.63831	9.355500	138428495.0
4	1	AA	2/4/2011	16.18	17.39	16.18	17.14	154387761	5.93325	1.987452	151379173.0
...	...	...	...	...	...	...	...	...	...	...	
745	2	XOM	5/27/2011	80.22	82.63	80.07	82.63	68230855	3.00424	-21.355713	86758820.0
746	2	XOM	6/3/2011	83.28	83.75	80.18	81.18	78616295	-2.52161	15.221032	68230855.0
747	2	XOM	6/10/2011	80.93	81.87	79.72	79.78	92380844	-1.42098	17.508519	78616295.0
748	2	XOM	6/17/2011	80.00	80.82	78.33	79.02	100521400	-1.22500	8.811952	92380844.0
749	2	XOM	6/24/2011	78.65	81.12	76.78	76.78	118679791	-2.37762	18.064204	100521400.0

```
data.info()
```

**Output:**

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 750 entries, 0 to 749
Data columns (total 16 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   quarter          750 non-null    int64  
 1   stock             750 non-null    object  
 2   date              750 non-null    object  
 3   open               750 non-null    object  
 4   high              750 non-null    object  
 5   low               750 non-null    object  
 6   close              750 non-null    object  
 7   volume             750 non-null    int64  
 8   percent_change_price  750 non-null    float64 
 9   percent_change_volume_over_last_wk 720 non-null    float64 
 10  previous_weeks_volume      720 non-null    float64 
 11  next_weeks_open          750 non-null    object  
 12  next_weeks_close         750 non-null    object  
 13  percent_change_next_weeks_price 750 non-null    float64 
 14  days_to_next_dividend   750 non-null    int64  
 15  percent_return_next_dividend 750 non-null    float64 
dtypes: float64(5), int64(3), object(8)
memory usage: 93.9+ KB
```

```
data.shape
```

**Output:**

```
In [5]: data.shape
Out[5]: (750, 16)
```

```
data.describe()
```

**Output:**

```
In [6]: data.describe()
Out[6]:    quarter      volume  percent_change_price  percent_change_volume_over_last_wk  previous_weeks_volume  percent_change_next_weeks_price
count  750.000000  7.500000e+02            750.000000                720.000000        7.200000e+02            750.000000
mean   1.520000  1.175478e+08            0.050262                  5.593627        1.173876e+08            0.238468
std    0.499933  1.584381e+08            2.517809                 40.543478        1.592322e+08            2.679538
min    1.000000  9.718851e+06            -15.422900                 -61.433175        9.718851e+06            -15.422900
25%    1.000000  3.086624e+07            -1.288053                 -19.804284        3.067832e+07            -1.222068
50%    2.000000  5.306088e+07            0.000000                  0.512586        5.294556e+07            0.101193
75%    2.000000  1.327218e+08            1.650888                 21.800622        1.333230e+08            1.845562
max    2.000000  1.453439e+09            9.882230                 327.408924        1.453439e+09            9.882230
```

```
data['stock'].value_counts()
```

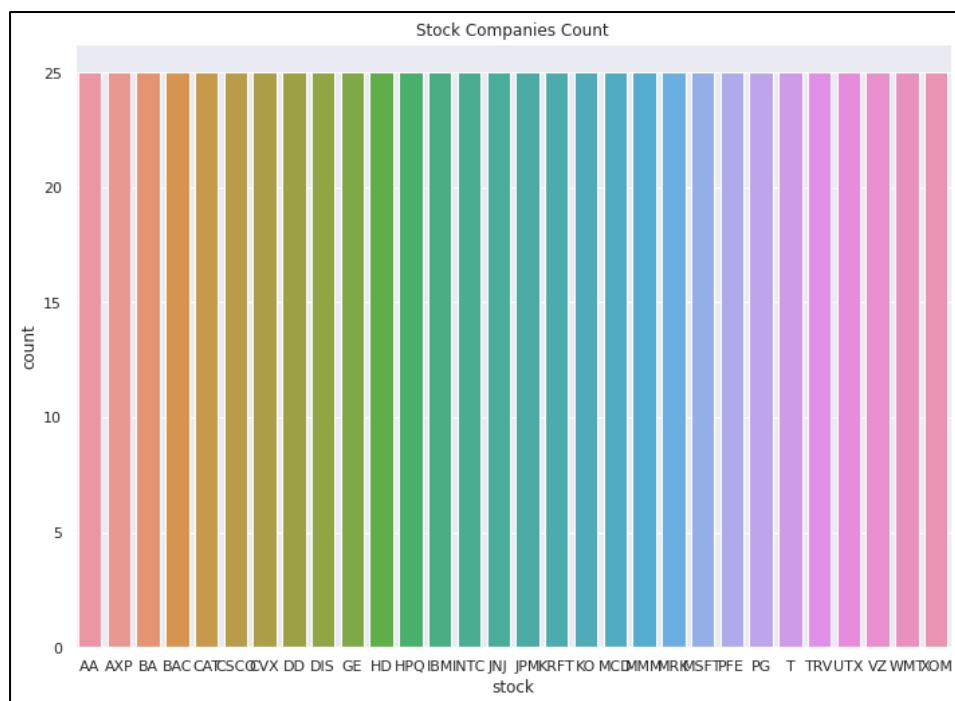
**Output:**

In [12]:	data[ 'stock' ].value_counts()
Out[12]:	AA      25 AXP     25 WMT     25 VZ      25 UTX     25 TRV     25 T       25 PG      25 PFE     25 MSFT    25 MRK     25 MMM     25 MCD     25 KO      25 KRFT    25 JPM     25 JNJ     25 INTC    25 IBM     25 HPQ     25 HD      25 GE      25 DIS     25 DD      25 CVX     25 CSCO    25 CAT     25 BAC     25 BA      25 XOM     25
	Name: stock, dtype: int64

```
plt.title('Stock Companies Count')
```

```
sns.countplot(data['stock'])
```

**Output:**

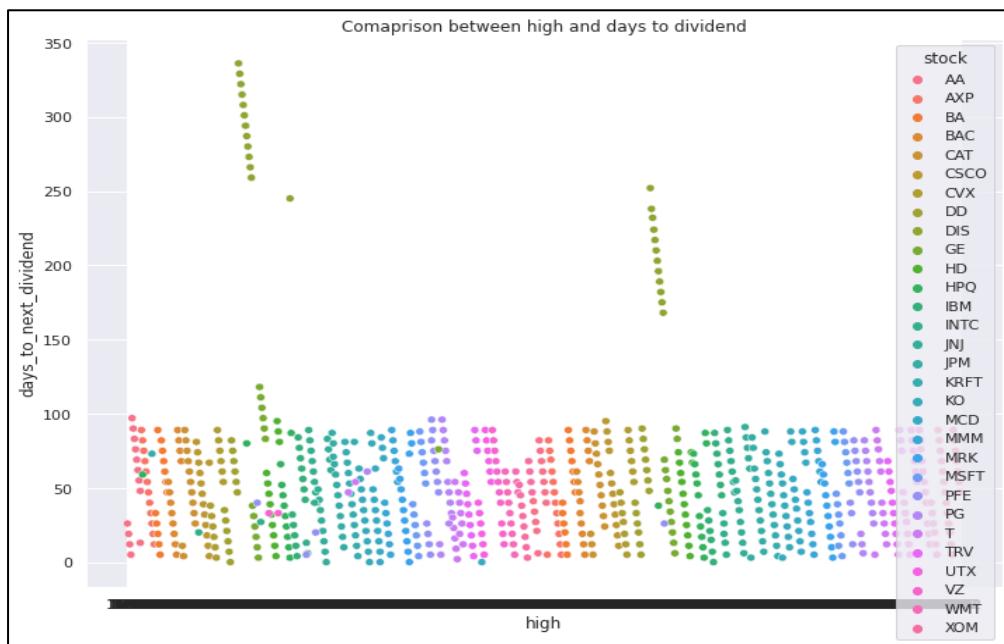


```

plt.title("Comaprison between high and days to dividend")
sns.scatterplot(data['high'],data['days_to_next_dividend'],hue=data['stock'])

```

**Output:**

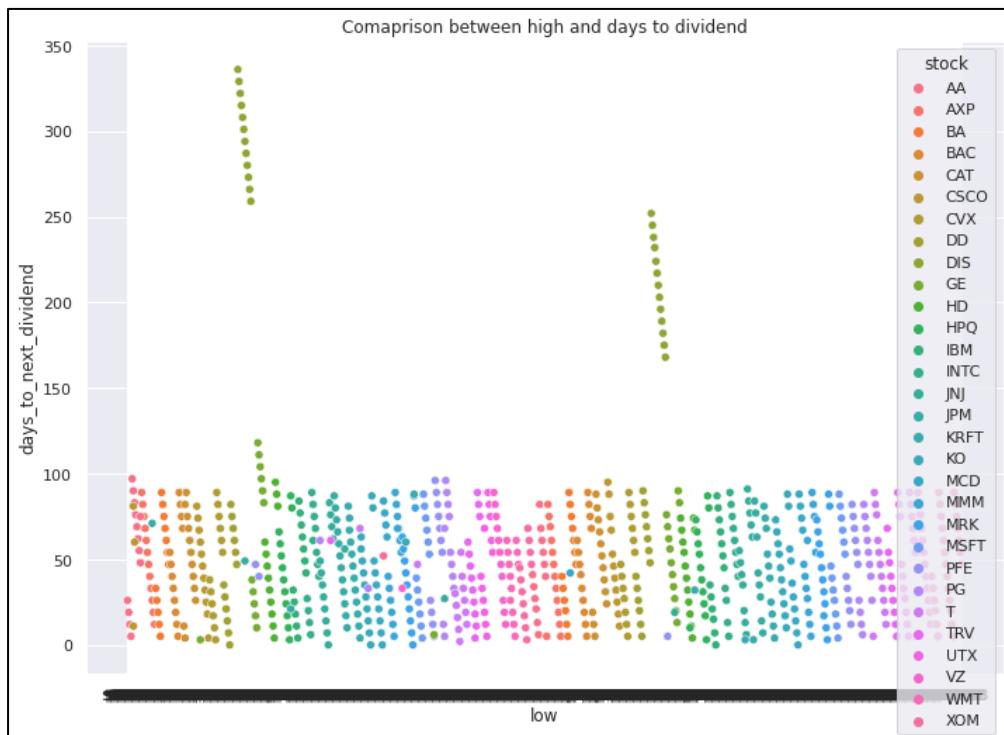


```

plt.title("Comaprison between high and days to dividend")
sns.scatterplot(data['low'],data['days_to_next_dividend'],hue=data['stock'])

```

**Output:**



```
sns.pairplot(data,hue="stock")
```

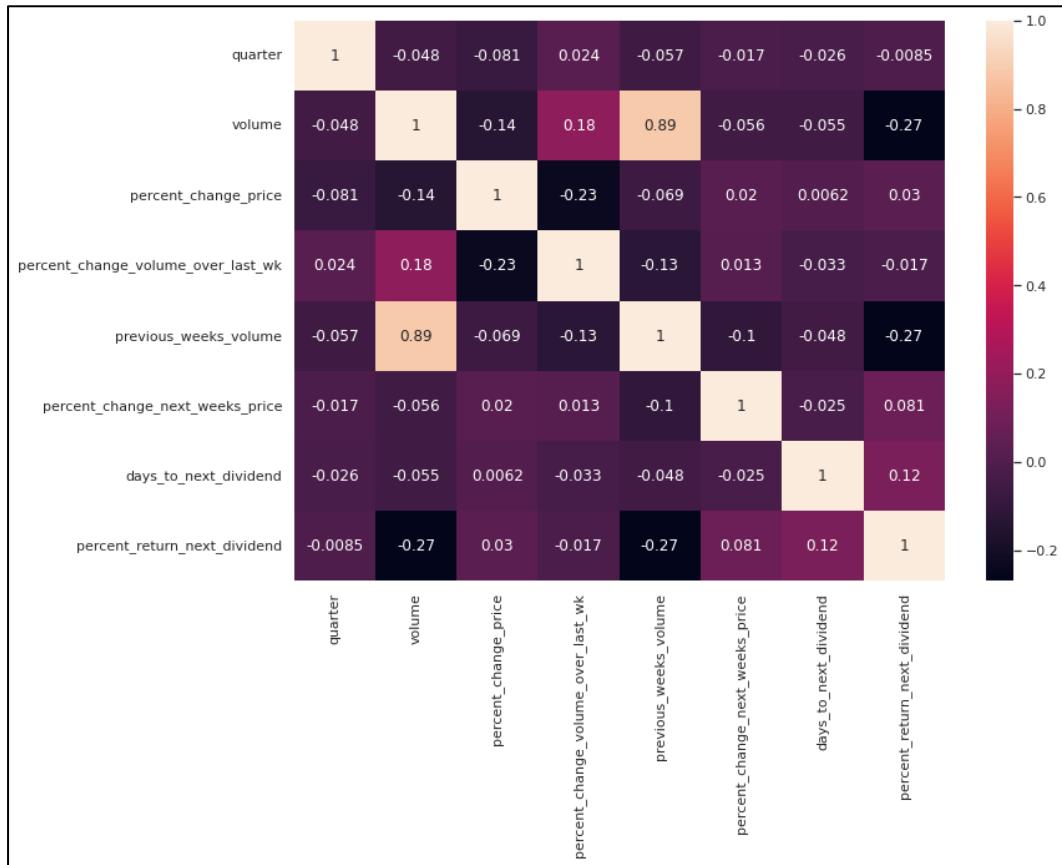
### Output:



```
sns.heatmap(data.corr(), annot=True)
```

```
plt.plot()
```

### Output:



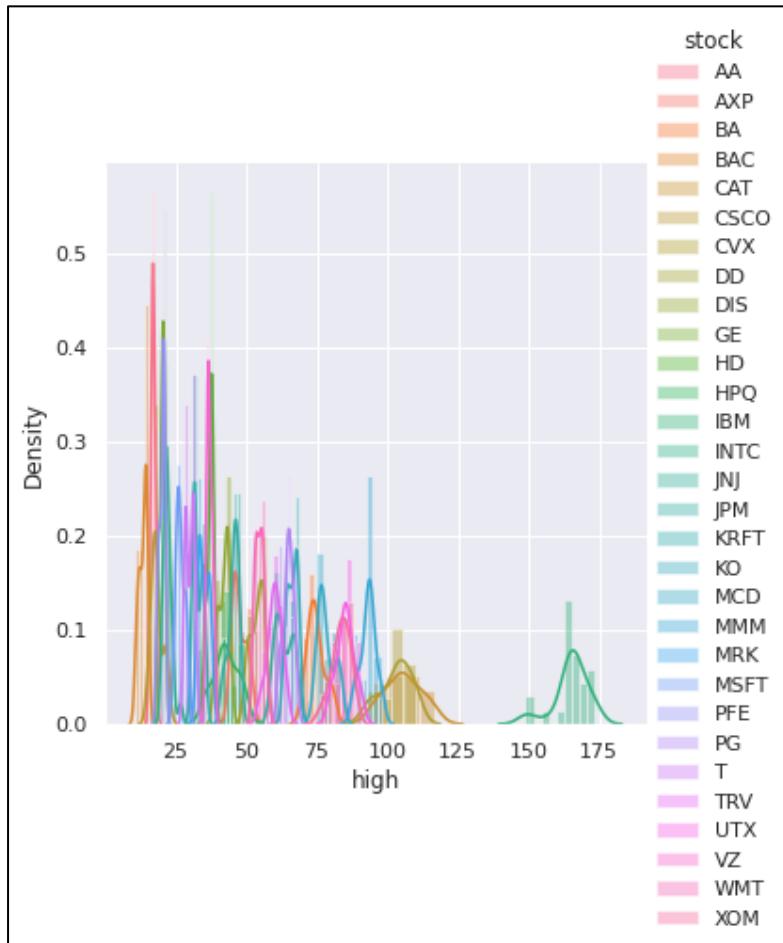
```
data.groupby('stock').agg(['mean','median'])
```

### Output:

stock	quarter		volume		percent_change_price		percent_change_volume_over_last_wk		previous_weeks_volume		percent_change_next	
	mean	median	mean	median	mean	median	mean	median	mean	median	mean	median
AA	1.52	2.0	1.296388e+08	114332562.0	-0.302084	-0.349650	4.032988	3.203676	1.308978e+08	114511920.5	-0.167325	
AXP	1.52	2.0	3.520848e+07	35427164.0	0.521029	0.706794	2.149548	-2.602828	3.522226e+07	35996388.0	0.740698	
BA	1.52	2.0	2.378142e+07	22770062.0	0.253427	0.122649	3.571908	-0.580803	2.349417e+07	22289736.5	0.216568	
BAC	1.52	2.0	7.229991e+08	699671790.0	-0.976965	-1.051160	-0.143670	-1.505000	7.279950e+08	706064273.5	-0.875759	
CAT	1.52	2.0	3.373112e+07	31169285.0	0.152660	0.311526	6.578793	6.373469	3.331479e+07	30933587.0	0.526747	
CSCO	1.52	2.0	3.586616e+08	308644641.0	-1.178297	-0.480513	16.967192	4.468062	3.557004e+08	306095259.5	-0.1033689	
CVX	1.52	2.0	3.857983e+07	39606692.0	0.340060	0.903809	6.117655	-4.519508	3.841169e+07	39222643.5	0.611745	
DD	1.52	2.0	2.911666e+07	27511651.0	0.362452	-0.180180	4.989947	-0.076329	2.887994e+07	27505269.0	0.587751	
DIS	1.52	2.0	4.744367e+07	42131642.0	0.175203	0.280177	6.859223	7.773932	4.732010e+07	41680515.0	0.217249	
GE	1.52	2.0	2.639381e+08	246591055.0	-0.240471	-0.287356	6.513043	-1.148899	2.626030e+08	242604845.5	0.039176	
HD	1.52	2.0	4.820505e+07	47021433.0	0.108416	0.216626	6.970308	-8.902819	4.748789e+07	46314048.0	0.395717	
HPQ	1.52	2.0	9.485843e+07	84066350.0	-0.472149	0.024207	18.604938	-10.509219	9.452860e+07	83344901.5	-0.515813	
IBM	1.52	2.0	2.473918e+07	23505533.0	0.587680	0.489097	5.512527	1.683035	2.481229e+07	23584077.0	0.780497	
INTC	1.52	2.0	2.990224e+08	289874871.0	0.188712	-0.410023	2.555824	5.484880	2.986054e+08	288109410.5	0.512314	
JNJ	1.52	2.0	5.600611e+07	53844149.0	0.270470	0.333991	3.465652	-2.778951	5.598406e+07	53614623.0	0.401220	
JPM	1.52	2.0	1.511090e+08	140414285.0	-0.070105	0.372353	2.027203	0.314943	1.504223e+08	139977205.0	0.091680	
KO	1.52	2.0	3.855617e+07	34863226.0	0.081142	0.293686	6.647356	4.548834	3.808740e+07	34405394.0	0.460708	

```
sns.FacetGrid(data,hue='stock',height=5).map(sns.distplot,'high').add_legend()  
plt.show()
```

### **Output:**



### **Result:**

Dow Jones Index Dataset has been successfully used for exploring various commands to do descriptive analytics and the results are verified.

### **3)Aim:**

To use personal (Dow Jones Index) dataset and do the following :

- a)Find correlation between features and visualize it
- b)Fill missing data
- c) Data Partitioning
- d) Perform statistical tests for the dataset
  - Normality test
  - Corelation test
  - Parametric statistical Hypothesis test
  - Nonparametric statistical Hypothesis test

### **Code & Output:**

```
import pandas as pd
```

```
import seaborn as sns
```

```
import numpy as np
```

```
import matplotlib.pyplot as plt
```

```
data=pd.read_csv("dow_jones_index.csv")
```

#### **#data partitioning**

```
data['high']=data['high'].str.replace('$', '')
```

```
data['open']=data['open'].str.replace('$', '')
```

```
data['low']=data['low'].str.replace('$', '')
```

```
data['close']=data['close'].str.replace('$', '')
```

```
data['next_weeks_open']=data['next_weeks_open'].str.replace('$', '')
```

```
data['next_weeks_close']=data['next_weeks_close'].str.replace('$', '')
```

```
data['high']=pd.to_numeric(data['high'])
```

```
data['open']=pd.to_numeric(data['open'])
```

```
data['low']=pd.to_numeric(data['low'])
```

```
data['close']=pd.to_numeric(data['close'])
```

```
data['next_weeks_open']=pd.to_numeric(data['next_weeks_open'])
```

```
data['next_weeks_close']=pd.to_numeric(data['next_weeks_close'])
```

## #fill missing data

```
data['percent_change_volume_over_last_wk'].fillna(value=0,inplace=True)
```

```
data['previous_weeks_volume'].fillna(value=0,inplace=True)
```

```
data
```

## Output:

In [6]:	data																																																																																																																																																												
Out[6]:	<table border="1"><thead><tr><th></th><th>quarter</th><th>stock</th><th>date</th><th>open</th><th>high</th><th>low</th><th>close</th><th>volume</th><th>percent_change_price</th><th>percent_change_volume_over_last_wk</th><th>previous_weeks_volume</th><th>ne</th></tr></thead><tbody><tr><td>0</td><td>1</td><td>AA</td><td>1/7/2011</td><td>15.82</td><td>16.72</td><td>15.78</td><td>16.42</td><td>239655616</td><td>3.79267</td><td>0.000000</td><td>0.0</td><td></td></tr><tr><td>1</td><td>1</td><td>AA</td><td>1/14/2011</td><td>16.71</td><td>16.71</td><td>15.64</td><td>15.97</td><td>242963398</td><td>-4.42849</td><td>1.380223</td><td>239655616.0</td><td></td></tr><tr><td>2</td><td>1</td><td>AA</td><td>1/21/2011</td><td>16.19</td><td>16.38</td><td>15.60</td><td>15.79</td><td>138428495</td><td>-2.47066</td><td>-43.024959</td><td>242963398.0</td><td></td></tr><tr><td>3</td><td>1</td><td>AA</td><td>1/28/2011</td><td>15.87</td><td>16.63</td><td>15.82</td><td>16.13</td><td>151379173</td><td>1.63831</td><td>9.355500</td><td>138428495.0</td><td></td></tr><tr><td>4</td><td>1</td><td>AA</td><td>2/4/2011</td><td>16.18</td><td>17.39</td><td>16.18</td><td>17.14</td><td>154387761</td><td>5.93325</td><td>1.987452</td><td>151379173.0</td><td></td></tr><tr><td>...</td><td>...</td><td>...</td><td>...</td><td>...</td><td>...</td><td>...</td><td>...</td><td>...</td><td>...</td><td>...</td><td>...</td><td></td></tr><tr><td>745</td><td>2</td><td>XOM</td><td>5/27/2011</td><td>80.22</td><td>82.63</td><td>80.07</td><td>82.63</td><td>68230855</td><td>3.00424</td><td>-21.355713</td><td>86758820.0</td><td></td></tr><tr><td>746</td><td>2</td><td>XOM</td><td>6/3/2011</td><td>83.28</td><td>83.75</td><td>80.18</td><td>81.18</td><td>78616295</td><td>-2.52161</td><td>15.221032</td><td>68230855.0</td><td></td></tr><tr><td>747</td><td>2</td><td>XOM</td><td>6/10/2011</td><td>80.93</td><td>81.87</td><td>79.72</td><td>79.78</td><td>92380844</td><td>-1.42098</td><td>17.508519</td><td>78616295.0</td><td></td></tr><tr><td>748</td><td>2</td><td>XOM</td><td>6/17/2011</td><td>80.00</td><td>80.82</td><td>78.33</td><td>79.02</td><td>100521400</td><td>-1.22500</td><td>8.811952</td><td>92380844.0</td><td></td></tr><tr><td>749</td><td>2</td><td>XOM</td><td>6/24/2011</td><td>78.65</td><td>81.12</td><td>76.78</td><td>76.78</td><td>118679791</td><td>-2.37762</td><td>18.064204</td><td>100521400.0</td><td></td></tr></tbody></table>		quarter	stock	date	open	high	low	close	volume	percent_change_price	percent_change_volume_over_last_wk	previous_weeks_volume	ne	0	1	AA	1/7/2011	15.82	16.72	15.78	16.42	239655616	3.79267	0.000000	0.0		1	1	AA	1/14/2011	16.71	16.71	15.64	15.97	242963398	-4.42849	1.380223	239655616.0		2	1	AA	1/21/2011	16.19	16.38	15.60	15.79	138428495	-2.47066	-43.024959	242963398.0		3	1	AA	1/28/2011	15.87	16.63	15.82	16.13	151379173	1.63831	9.355500	138428495.0		4	1	AA	2/4/2011	16.18	17.39	16.18	17.14	154387761	5.93325	1.987452	151379173.0		...	...	...	...	...	...	...	...	...	...	...	...		745	2	XOM	5/27/2011	80.22	82.63	80.07	82.63	68230855	3.00424	-21.355713	86758820.0		746	2	XOM	6/3/2011	83.28	83.75	80.18	81.18	78616295	-2.52161	15.221032	68230855.0		747	2	XOM	6/10/2011	80.93	81.87	79.72	79.78	92380844	-1.42098	17.508519	78616295.0		748	2	XOM	6/17/2011	80.00	80.82	78.33	79.02	100521400	-1.22500	8.811952	92380844.0		749	2	XOM	6/24/2011	78.65	81.12	76.78	76.78	118679791	-2.37762	18.064204	100521400.0	
	quarter	stock	date	open	high	low	close	volume	percent_change_price	percent_change_volume_over_last_wk	previous_weeks_volume	ne																																																																																																																																																	
0	1	AA	1/7/2011	15.82	16.72	15.78	16.42	239655616	3.79267	0.000000	0.0																																																																																																																																																		
1	1	AA	1/14/2011	16.71	16.71	15.64	15.97	242963398	-4.42849	1.380223	239655616.0																																																																																																																																																		
2	1	AA	1/21/2011	16.19	16.38	15.60	15.79	138428495	-2.47066	-43.024959	242963398.0																																																																																																																																																		
3	1	AA	1/28/2011	15.87	16.63	15.82	16.13	151379173	1.63831	9.355500	138428495.0																																																																																																																																																		
4	1	AA	2/4/2011	16.18	17.39	16.18	17.14	154387761	5.93325	1.987452	151379173.0																																																																																																																																																		
...	...	...	...	...	...	...	...	...	...	...	...																																																																																																																																																		
745	2	XOM	5/27/2011	80.22	82.63	80.07	82.63	68230855	3.00424	-21.355713	86758820.0																																																																																																																																																		
746	2	XOM	6/3/2011	83.28	83.75	80.18	81.18	78616295	-2.52161	15.221032	68230855.0																																																																																																																																																		
747	2	XOM	6/10/2011	80.93	81.87	79.72	79.78	92380844	-1.42098	17.508519	78616295.0																																																																																																																																																		
748	2	XOM	6/17/2011	80.00	80.82	78.33	79.02	100521400	-1.22500	8.811952	92380844.0																																																																																																																																																		
749	2	XOM	6/24/2011	78.65	81.12	76.78	76.78	118679791	-2.37762	18.064204	100521400.0																																																																																																																																																		
	750 rows × 16 columns																																																																																																																																																												

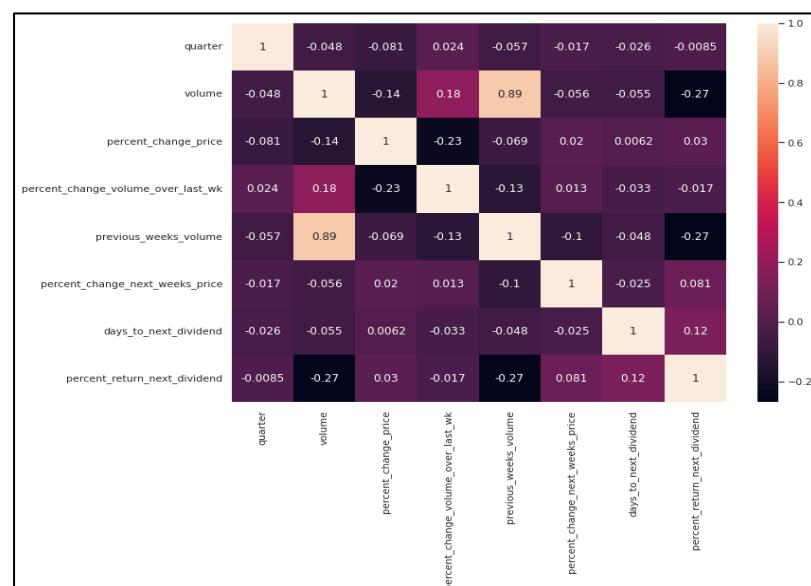
previous_weeks_volume	next_weeks_open	next_weeks_close	percent_change_next_weeks_price	days_to_next_dividend	percent_return_next_dividend
0.0	16.71	15.97	-4.42849	26	0.182704
239655616.0	16.19	15.79	-2.47066	19	0.187852
242963398.0	15.87	16.13	1.638310	12	0.189994
138428495.0	16.18	17.14	5.933250	5	0.185989
151379173.0	17.33	17.37	0.230814	97	0.175029
...	...	...	...	...	...
86758820.0	83.28	81.18	-2.521610	75	0.568801
68230855.0	80.93	79.78	-1.420980	68	0.578960
78616295.0	80.00	79.02	-1.225000	61	0.589120
92380844.0	78.65	76.78	-2.377620	54	0.594786
100521400.0	76.88	82.01	6.672740	47	0.612139

## #corelation

```
sns.heatmap(data.corr(),annot=True)
```

```
plt.plot()
```

## Output:



```

#normality test

from scipy import stats

z=[data['high'],data['low']]

for i in z:

    print([i])

    a,b=stats.normaltest(data['high'])

    print(a,b)

    alpha = 0.05

    if b<alpha :

        print('null hypothesis rejected')

    else:

        print('null hypothesis cannot be rejected')

```

**Output:**

```

[0      16.72
 1      16.71
 2      16.38
 3      16.63
 4      17.39
 ...
 745     82.63
 746     83.75
 747     81.87
 748     80.82
 749     81.12
Name: high, Length: 750, dtype: float64]
165.8155807197986 9.85380899939066e-37
null hypothesis rejected
[0      15.78
 1      15.64
 2      15.60
 3      15.82
 4      16.18
 ...
 745     80.07
 746     80.18
 747     79.72
 748     78.33
 749     76.78
Name: low, Length: 750, dtype: float64]
165.8155807197986 9.85380899939066e-37
null hypothesis rejected

```

**#correlation test**

```

from scipy.stats import pearsonr

data['high']= pd.to_numeric(data['high'])

data['low']= pd.to_numeric(data['low'])

list1 = data['high']

```

```
list2 = data['low']
corr,_= pearsonr(list1, list2)
print('Pearsons correlation: %.3f' % corr)
```

**Output:**

```
Pearsons correlation: 0.999
```

**#Parametric statistical Hypothesis test**

```
from scipy.stats import ttest_ind
z=[data['high'],data['low']]
for i in z:
    print([i])
    a,b=ttest_ind(data['high'],data['low'])
    print(a,b)
    alpha = 0.05
    if b>alpha :
        print('Same distributions')
    else:
        print('Different distributions')
```

**Output:**

```
[0      16.72
 1      16.71
 2      16.38
 3      16.63
 4      17.39
 ...
 745     82.63
 746     83.75
 747     81.87
 748     80.82
 749     81.12
Name: high, Length: 750, dtype: float64]
1.2030833315757794 0.22913417910648973
Same distributions
[0      15.78
 1      15.64
 2      15.60
 3      15.82
 4      16.18
 ...
 745     80.07
 746     80.18
 747     79.72
 748     78.33
 749     76.78
Name: low, Length: 750, dtype: float64]
1.2030833315757794 0.22913417910648973
Same distributions
```

## #Non Parametric statistical Hypothesis test

```
from scipy.stats import mannwhitneyu  
z=[data['high'],data['low']]  
for i in z:  
    print([i])  
    a,b=mannwhitneyu(data['high'],data['low'])  
    print(a,b)  
    alpha = 0.05  
    if b>alpha :  
        print('Same distributions')  
    else:  
        print('Different distributions')
```

### Output:

```
[0      16.72  
1      16.71  
2      16.38  
3      16.63  
4      17.39  
     ...  
745     82.63  
746     83.75  
747     81.87  
748     80.82  
749     81.12  
Name: high, Length: 750, dtype: float64]  
292745.5 0.17056138472040383  
Same distributions  
[0      15.78  
1      15.64  
2      15.60  
3      15.82  
4      16.18  
     ...  
745     80.07  
746     80.18  
747     79.72  
748     78.33  
749     76.78  
Name: low, Length: 750, dtype: float64]  
292745.5 0.17056138472040383  
Same distributions
```

### **Result:**

Dow Jones Index Dataset has been successfully used for exploring various commands to do descriptive analytics and the results are verified.

## UNIVARIATE ANALYSIS

### 1)Aim:

To use the diabetes data set from UCI and Pima Indians Diabetes data set and perform the following: Univariate analysis, Frequency, Mean, Median, Mode, Variance, Standard Deviation, Skewness and Kurtosis.

### Code & Output:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
data = pd.read_csv("diabetes.csv")
data
```

### Output:

In [3]:	data = pd.read_csv("diabetes.csv") data																																																																																																												
Out[3]:	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Pregnancies</th><th>Glucose</th><th>BloodPressure</th><th>SkinThickness</th><th>Insulin</th><th>BMI</th><th>DiabetesPedigreeFunction</th><th>Age</th><th>Outcome</th></tr> </thead> <tbody> <tr><td>0</td><td>6</td><td>148</td><td>72</td><td>35</td><td>0</td><td>33.6</td><td>0.627</td><td>50</td></tr> <tr><td>1</td><td>1</td><td>85</td><td>66</td><td>29</td><td>0</td><td>26.6</td><td>0.351</td><td>31</td></tr> <tr><td>2</td><td>8</td><td>183</td><td>64</td><td>0</td><td>0</td><td>23.3</td><td>0.672</td><td>32</td></tr> <tr><td>3</td><td>1</td><td>89</td><td>66</td><td>23</td><td>94</td><td>28.1</td><td>0.167</td><td>21</td></tr> <tr><td>4</td><td>0</td><td>137</td><td>40</td><td>35</td><td>168</td><td>43.1</td><td>2.288</td><td>33</td></tr> <tr><td>...</td><td>...</td><td>...</td><td>...</td><td>...</td><td>...</td><td>...</td><td>...</td><td>...</td></tr> <tr><td>763</td><td>10</td><td>101</td><td>76</td><td>48</td><td>180</td><td>32.9</td><td>0.171</td><td>63</td></tr> <tr><td>764</td><td>2</td><td>122</td><td>70</td><td>27</td><td>0</td><td>36.8</td><td>0.340</td><td>27</td></tr> <tr><td>765</td><td>5</td><td>121</td><td>72</td><td>23</td><td>112</td><td>26.2</td><td>0.245</td><td>30</td></tr> <tr><td>766</td><td>1</td><td>126</td><td>60</td><td>0</td><td>0</td><td>30.1</td><td>0.349</td><td>47</td></tr> <tr><td>767</td><td>1</td><td>93</td><td>70</td><td>31</td><td>0</td><td>30.4</td><td>0.315</td><td>23</td></tr> </tbody> </table> <p>768 rows × 9 columns</p>	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome	0	6	148	72	35	0	33.6	0.627	50	1	1	85	66	29	0	26.6	0.351	31	2	8	183	64	0	0	23.3	0.672	32	3	1	89	66	23	94	28.1	0.167	21	4	0	137	40	35	168	43.1	2.288	33	...	...	...	...	...	...	...	...	...	763	10	101	76	48	180	32.9	0.171	63	764	2	122	70	27	0	36.8	0.340	27	765	5	121	72	23	112	26.2	0.245	30	766	1	126	60	0	0	30.1	0.349	47	767	1	93	70	31	0	30.4	0.315	23
Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome																																																																																																					
0	6	148	72	35	0	33.6	0.627	50																																																																																																					
1	1	85	66	29	0	26.6	0.351	31																																																																																																					
2	8	183	64	0	0	23.3	0.672	32																																																																																																					
3	1	89	66	23	94	28.1	0.167	21																																																																																																					
4	0	137	40	35	168	43.1	2.288	33																																																																																																					
...	...	...	...	...	...	...	...	...																																																																																																					
763	10	101	76	48	180	32.9	0.171	63																																																																																																					
764	2	122	70	27	0	36.8	0.340	27																																																																																																					
765	5	121	72	23	112	26.2	0.245	30																																																																																																					
766	1	126	60	0	0	30.1	0.349	47																																																																																																					
767	1	93	70	31	0	30.4	0.315	23																																																																																																					

data.dtypes

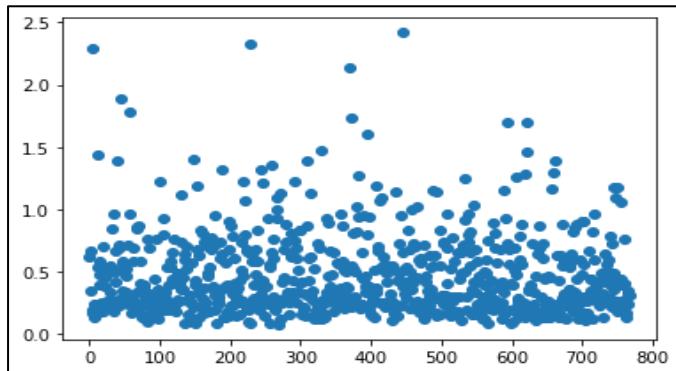
### Output:

In [5]:	data.dtypes
Out[5]:	Pregnancies int64
	Glucose int64
	BloodPressure int64
	SkinThickness int64
	Insulin int64
	BMI float64
	DiabetesPedigreeFunction float64
	Age int64
	Outcome int64
	dtype: object

```
plt.scatter(data.index,data['DiabetesPedigreeFunction'])
```

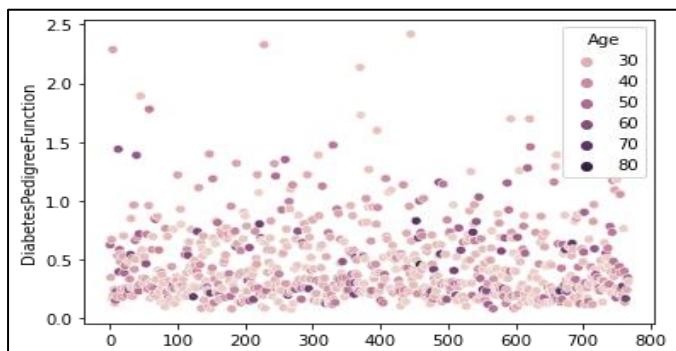
```
plt.show()
```

**Output:**



```
sns.scatterplot(x=data.index,y=data['DiabetesPedigreeFunction'],hue=data['Age'])
```

**Output:**



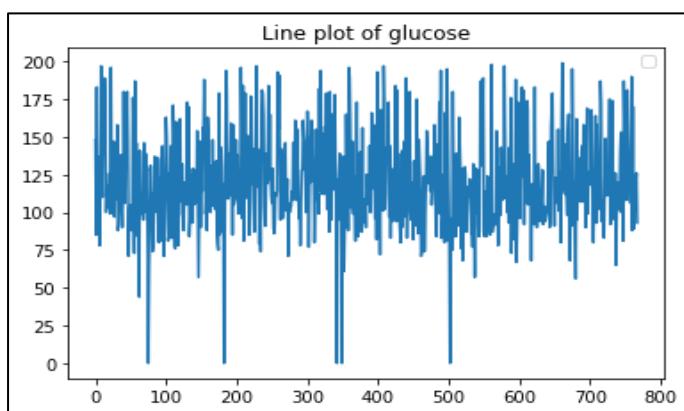
```
plt.title('Line plot of glucose')
```

```
plt.plot(data.index,data['Glucose'])
```

```
plt.legend()
```

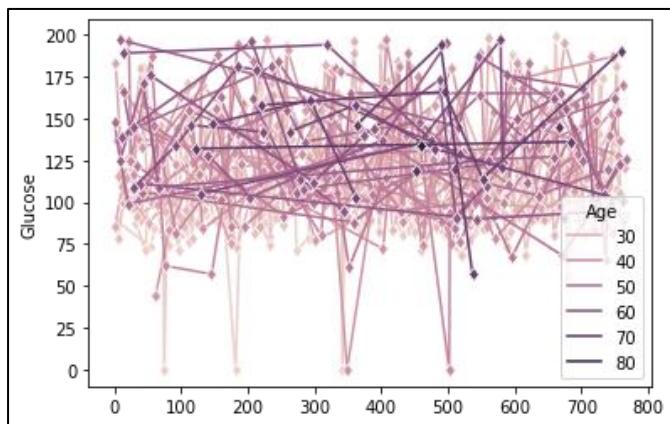
```
plt.show()
```

**Output:**



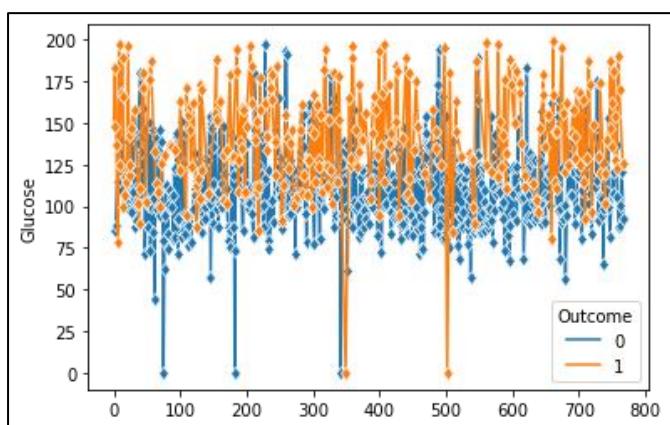
```
fig=sns.lineplot(x=data.index,y=data['Glucose'],markevery=1,marker='d',data=data,hue=data['Age'])
```

**Output:**



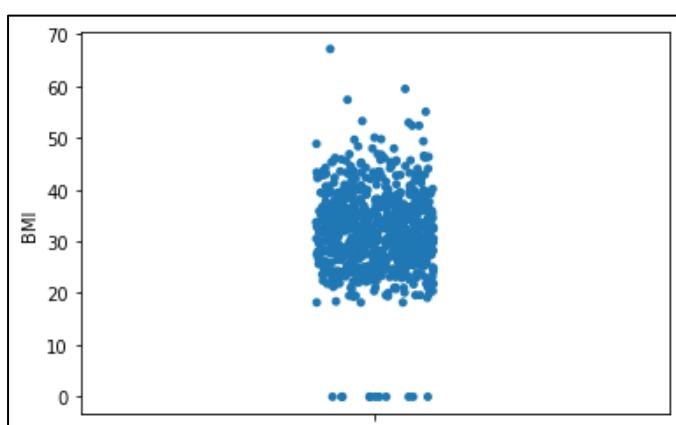
```
fig=sns.lineplot(x=data.index,y=data['Glucose'],markevery=1,marker='d',data=data,hue=data['Outcome'])
```

**Output:**



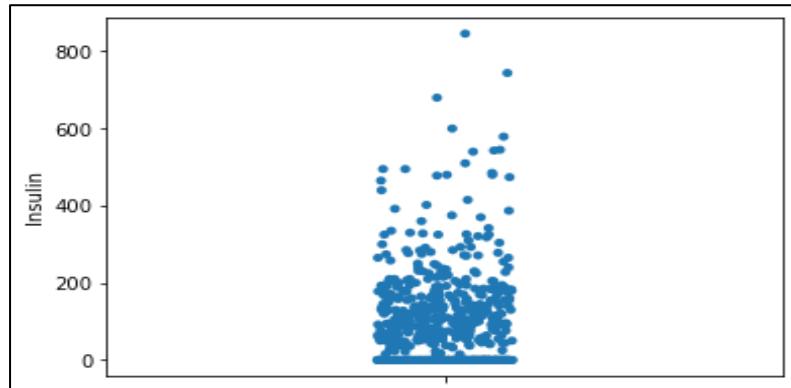
```
sns.stripplot(y=data['BMI'])
```

**Output:**



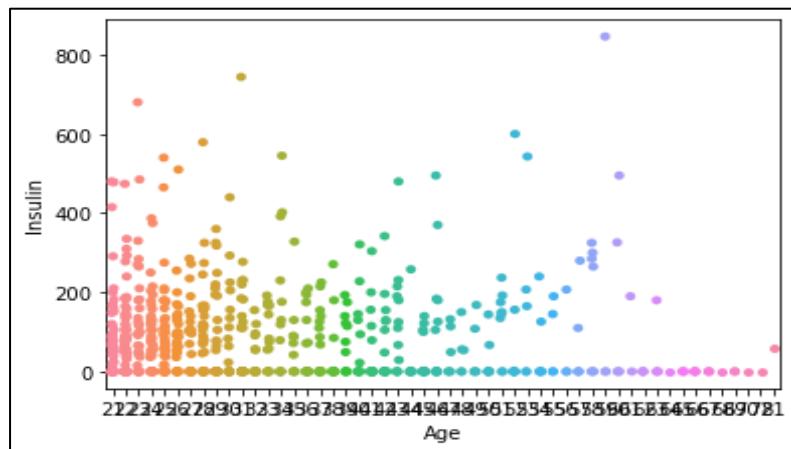
```
sns.stripplot(y=data['Insulin'])
```

**Output:**



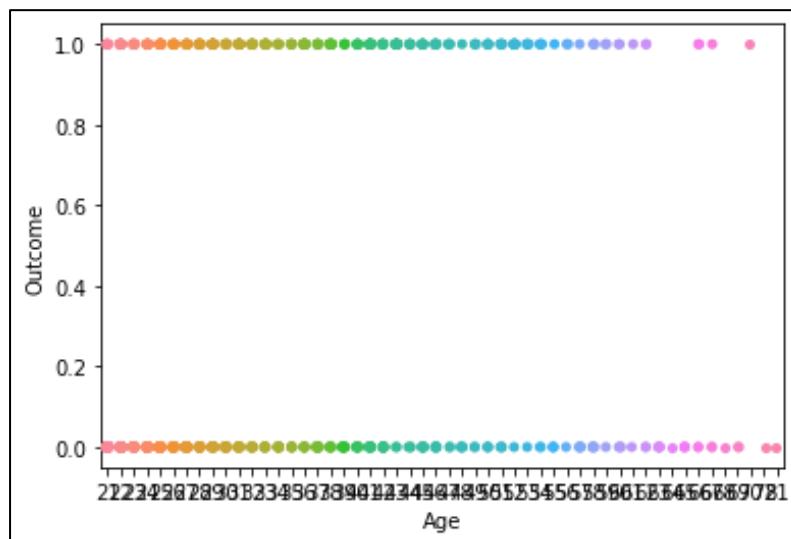
```
sns.stripplot(x=data['Age'],y=data['Insulin'])
```

**Output:**



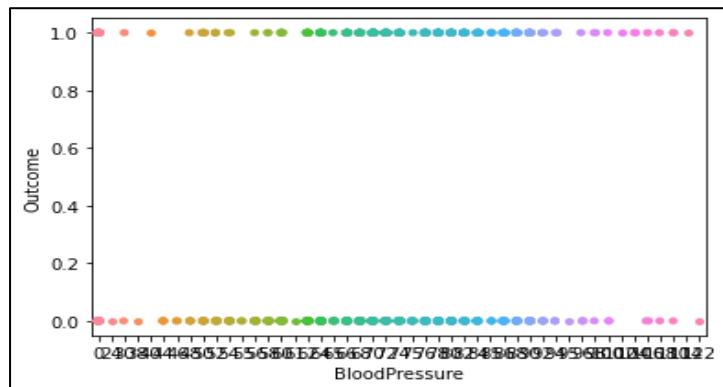
```
sns.stripplot(x=data['Age'],y=data['Outcome'])
```

**Output:**



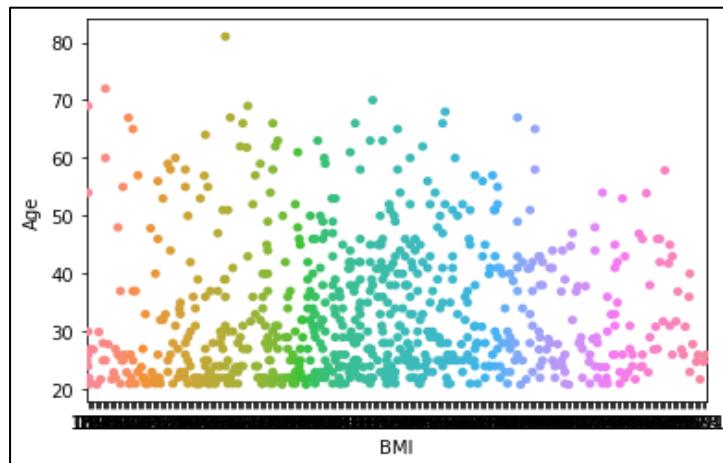
```
sns.stripplot(x=data['BloodPressure'],y=data['Outcome'])
```

**Output:**



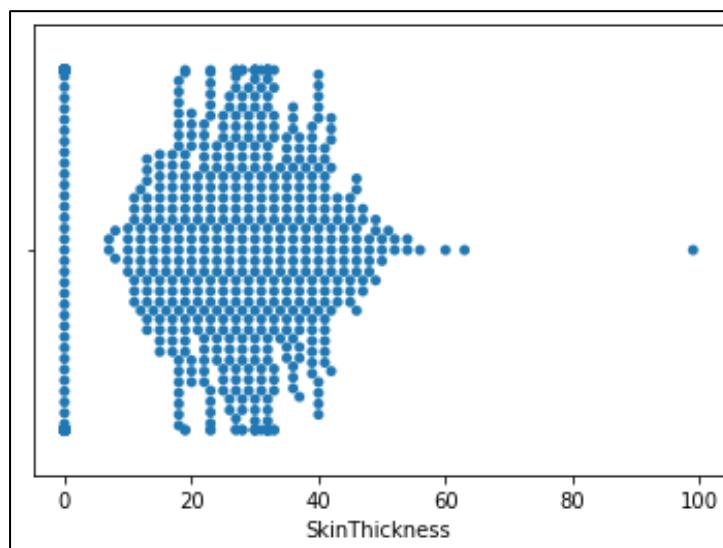
```
sns.swarmplot(x=data['BMI'],y=data['Age'])
```

**Output:**



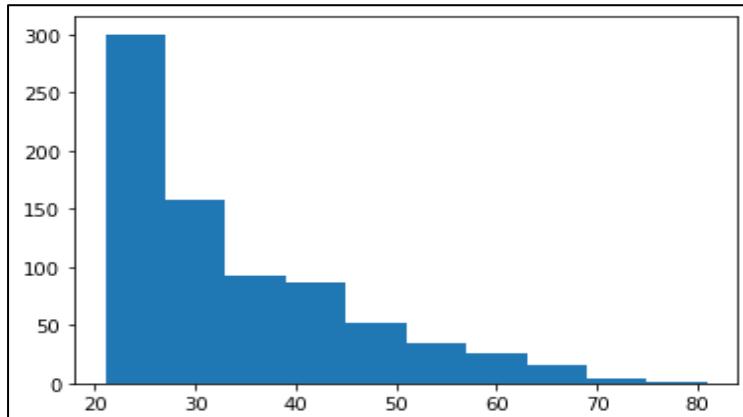
```
sns.swarmplot(x=data['SkinThickness'])
```

**Output:**



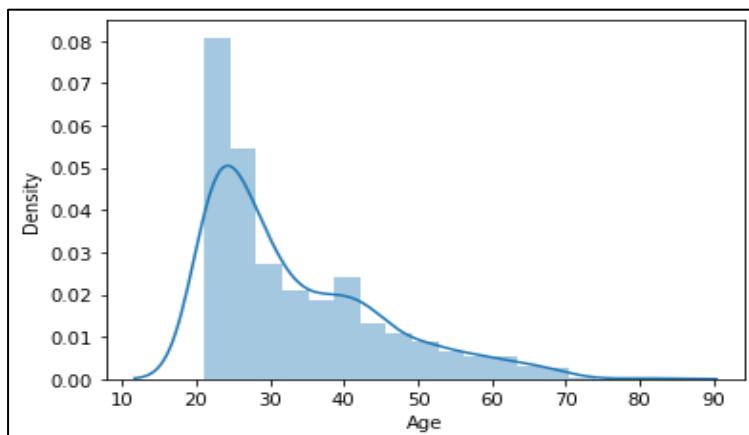
```
plt.hist(data['Age'])
```

**Output:**



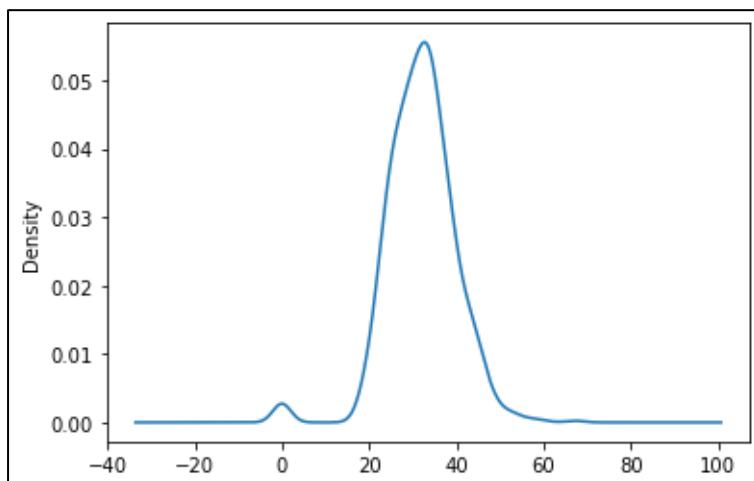
```
sns.distplot(data['Age'])
```

**Output:**



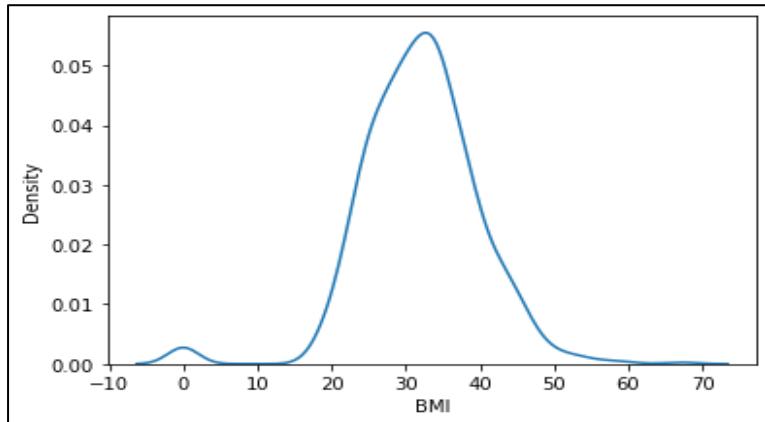
```
data['BMI'].plot(kind='density')
```

**Output:**



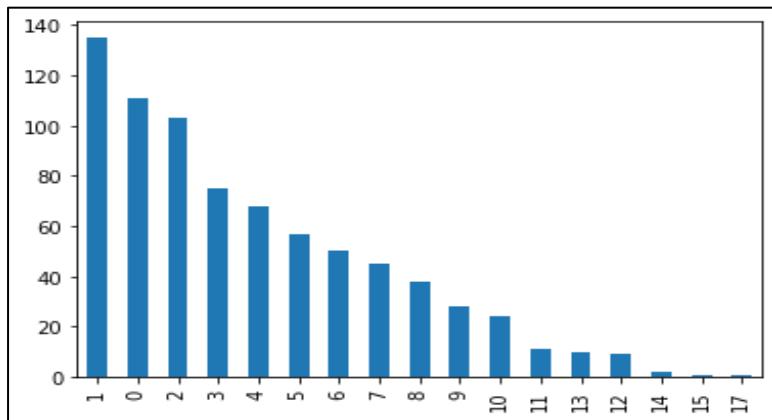
```
sns.kdeplot(data['BMI'])
```

**Output:**



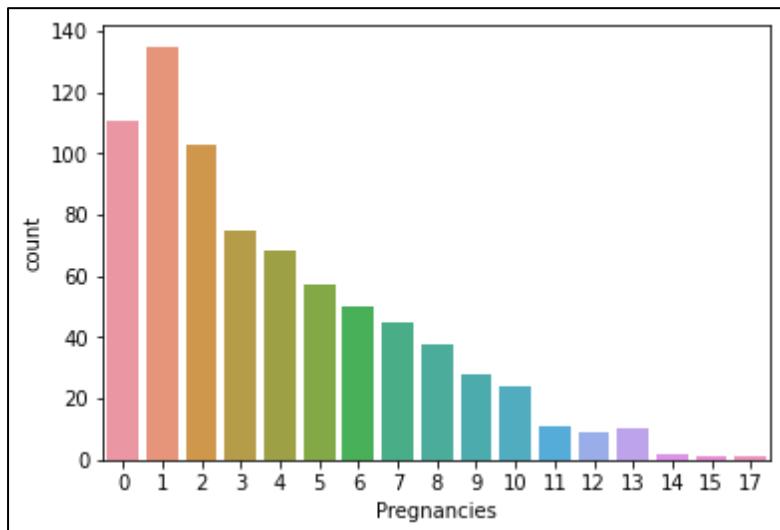
```
data['Pregnancies'].value_counts().plot.bar()
```

**Output:**



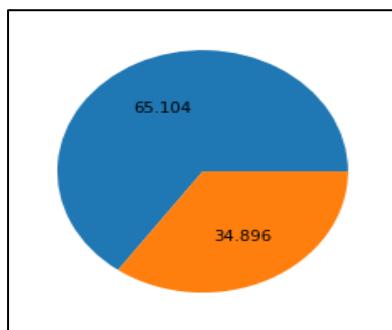
```
sns.countplot(data['Pregnancies'])
```

**Output:**



```
plt.pie(data['Outcome'].value_counts(), autopct='%.3f')
```

**Output:**



```
data['BMI'].mean()
```

**Output:**

```
In [32]: data['BMI'].mean()  
Out[32]: 31.992578124999977
```

```
data['BMI'].mode()
```

**Output:**

```
In [33]: data['BMI'].mode()  
Out[33]: 0    32.0  
          Name: BMI, dtype: float64
```

```
data['BMI'].median()
```

**Output:**

```
In [34]: data['BMI'].median()  
Out[34]: 32.0
```

```
data.mean()
```

**Output:**

```
In [35]: data.mean()  
Out[35]: Pregnancies      3.845052  
          Glucose         120.894531  
          BloodPressure   69.105469  
          SkinThickness  20.536458  
          Insulin        79.799479  
          BMI            31.992578  
          DiabetesPedigreeFunction 0.471876  
          Age             33.240885  
          Outcome        0.348958  
          dtype: float64
```

```
data.mode()
```

**Output:**

In [36]:	data.mode()									
Out[36]:	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome	
0	1.0	99	70.0	0.0	0.0	32.0		0.254	22.0	0.0
1	NaN	100	NaN	NaN	NaN	NaN		0.258	NaN	NaN

```
data.median()
```

**Output:**

In [37]:	data.median()	
Out[37]:		
	Pregnancies	3.0000
	Glucose	117.0000
	BloodPressure	72.0000
	SkinThickness	23.0000
	Insulin	30.5000
	BMI	32.0000
	DiabetesPedigreeFunction	0.3725
	Age	29.0000
	Outcome	0.0000
	dtype: float64	

```
data.var()
```

**Output:**

In [38]:	data.var()	
Out[38]:		
	Pregnancies	11.354056
	Glucose	1022.248314
	BloodPressure	374.647271
	SkinThickness	254.473245
	Insulin	13281.180078
	BMI	62.159984
	DiabetesPedigreeFunction	0.109779
	Age	138.303046
	Outcome	0.227483
	dtype: float64	

```
data.std()
```

**Output:**

In [39]:	data.std()	
Out[39]:		
	Pregnancies	3.369578
	Glucose	31.972618
	BloodPressure	19.355807
	SkinThickness	15.952218
	Insulin	115.244002
	BMI	7.884160
	DiabetesPedigreeFunction	0.331329
	Age	11.760232
	Outcome	0.476951
	dtype: float64	

```
data.skew()
```

**Output:**

```
In [40]: data.skew()

Out[40]: Pregnancies      0.901674
          Glucose        0.173754
          BloodPressure   -1.843608
          SkinThickness    0.109372
          Insulin         2.272251
          BMI            -0.428982
          DiabetesPedigreeFunction 1.919911
          Age             1.129597
          Outcome         0.635017
          dtype: float64
```

```
data.kurt()
```

**Output:**

```
In [41]: data.kurt()

Out[41]: Pregnancies      0.159220
          Glucose        0.640780
          BloodPressure   5.180157
          SkinThickness   -0.520072
          Insulin         7.214260
          BMI            3.290443
          DiabetesPedigreeFunction 5.594954
          Age             0.643159
          Outcome         -1.600930
          dtype: float64
```

**Result:**

Univariate analysis has been successfully done for Diabetes dataset and the results are verified.

## 2)Aim:

To use dow jones index data set and perform the following: Univariate analysis, Frequency, Mean, Median, Mode, Variance, Standard Deviation, Skewness and Kurtosis.

## Code & Output:

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

```
data = pd.read_csv('dow_jones_index.csv')  
data['high'] = data['high'].str.replace('$', '')  
data['open'] = data['open'].str.replace('$', '')  
data['low'] = data['low'].str.replace('$', '')  
data['close'] = data['close'].str.replace('$', '')  
data['next_weeks_open'] = data['next_weeks_open'].str.replace('$', '')  
data['next_weeks_close'] = data['next_weeks_close'].str.replace('$', '')  
data
```

## Output:

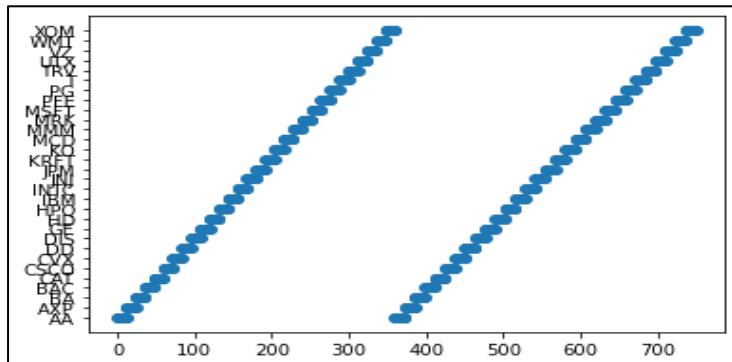
	quarter	stock	date	open	high	low	close	volume	percent_change_price	percent_change_volume_over_last_wk	previous_weeks_volume
0	1	AA	1/7/2011	15.82	16.72	15.78	16.42	239655616	3.79267	NaN	NaN
1	1	AA	1/14/2011	16.71	16.71	15.64	15.97	242963398	-4.42849	1.380223	239655616.0
2	1	AA	1/21/2011	16.19	16.38	15.60	15.79	138428495	-2.47066	-43.024959	242963398.0
3	1	AA	1/28/2011	15.87	16.63	15.82	16.13	151379173	1.63831	9.355500	138428495.0
4	1	AA	2/4/2011	16.18	17.39	16.18	17.14	154387761	5.93325	1.987452	151379173.0
...	...	...	...	...	...	...	...	...	...	...	...
745	2	XOM	5/27/2011	80.22	82.63	80.07	82.63	68230855	3.00424	-21.355713	86758820.0
746	2	XOM	6/3/2011	83.28	83.75	80.18	81.18	78616295	-2.52161	15.221032	68230855.0
747	2	XOM	6/10/2011	80.93	81.87	79.72	79.78	92380844	-1.42098	17.508519	78616295.0
748	2	XOM	6/17/2011	80.00	80.82	78.33	79.02	100521400	-1.22500	8.811952	92380844.0
749	2	XOM	6/24/2011	78.65	81.12	76.78	76.78	118679791	-2.37762	18.064204	100521400.0

750 rows × 16 columns

```
plt.scatter(data.index,data['stock'])
```

```
plt.show()
```

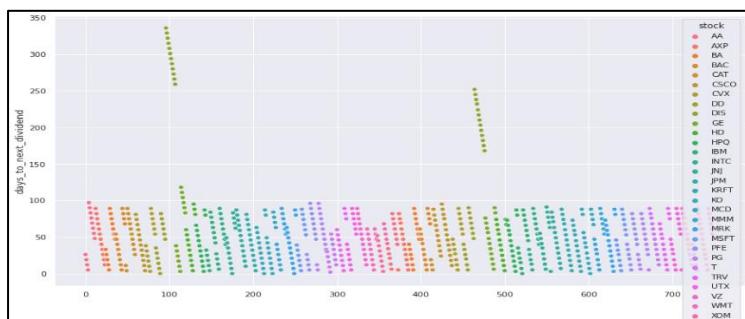
**Output:**



```
sns.set(rc={'figure.figsize':(15,8)})
```

```
sns.scatterplot(x=data.index,y=data['days_to_next_dividend'],hue=data['stock'])
```

**Output:**



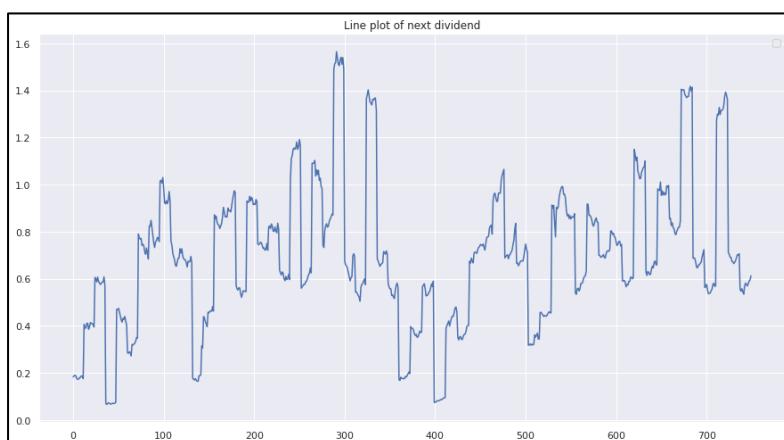
```
plt.title("Line plot of next dividend")
```

```
plt.plot(data.index,data['percent_return_next_dividend'])
```

```
plt.legend()
```

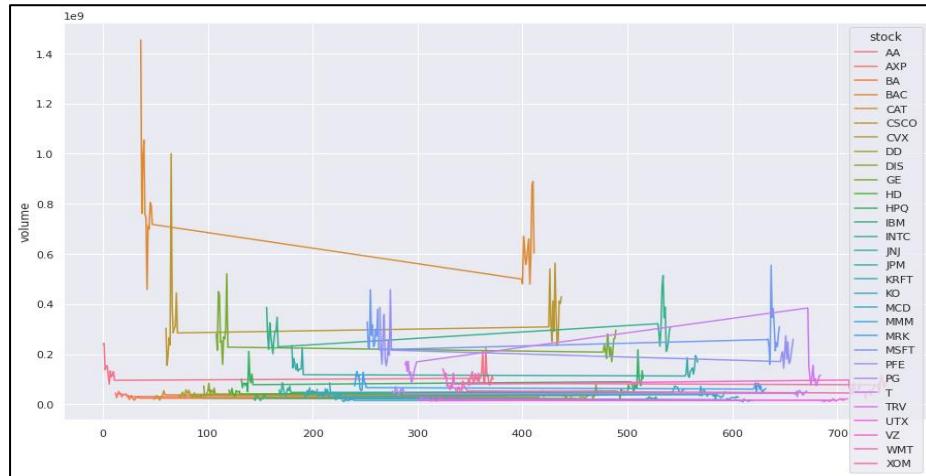
```
plt.show()
```

**Output:**



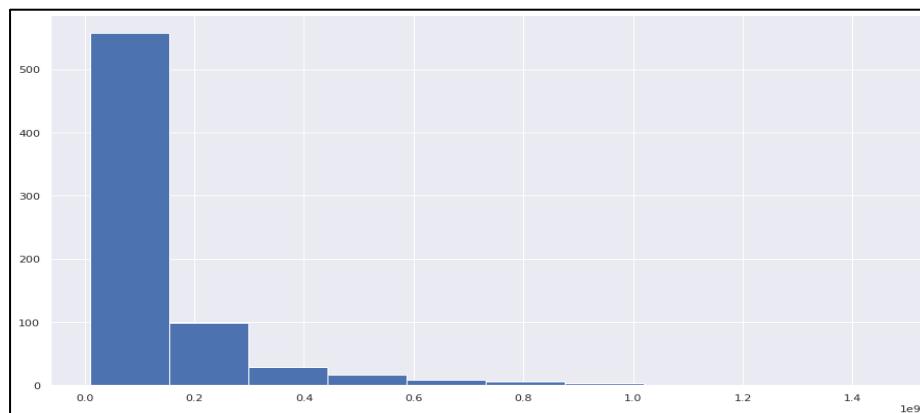
```
fig=sns.lineplot(x=data.index,y=data['volume'],hue=data['stock'])
```

**Output:**



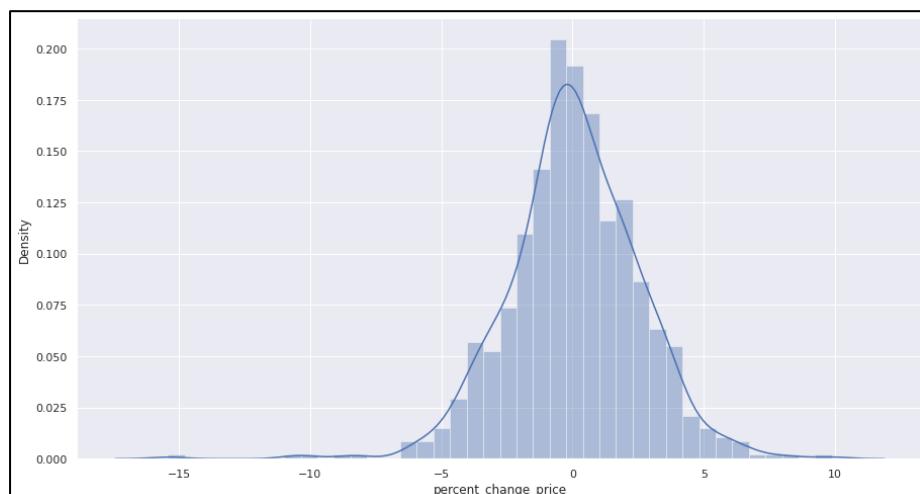
```
plt.hist(data['previous_weeks_volume'])
```

**Output:**



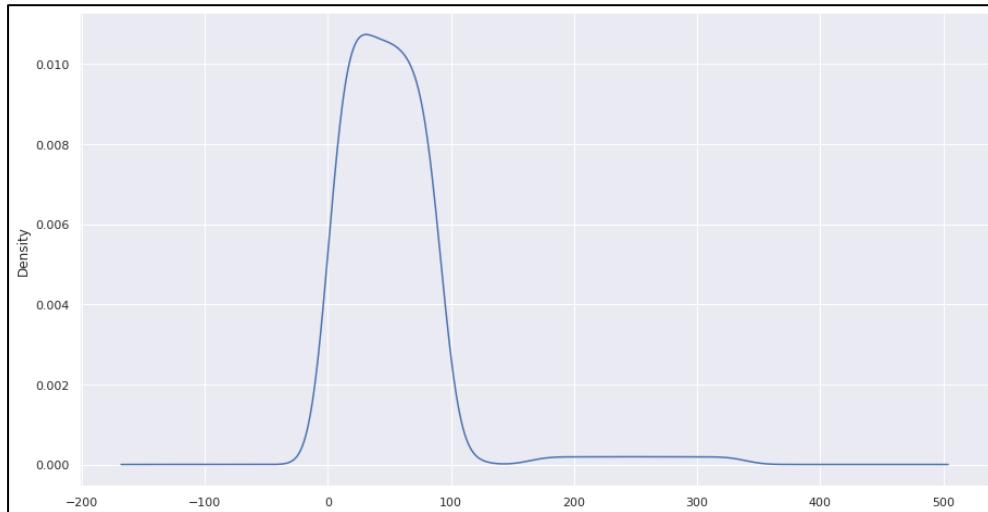
```
sns.distplot(data['percent_change_price'])
```

**Output:**



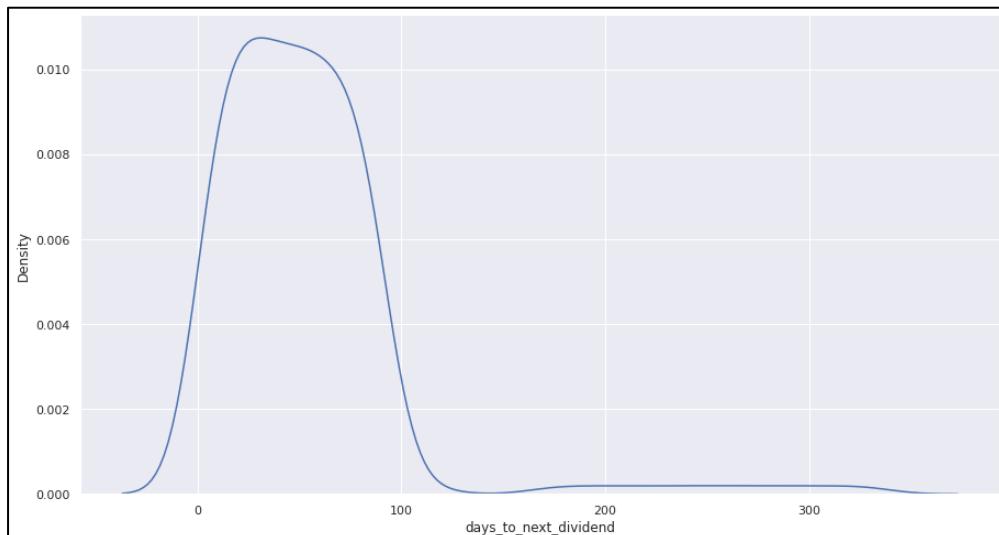
```
data['days_to_next_dividend'].plot(kind='density')
```

**Output:**



```
sns.kdeplot(data['days_to_next_dividend'])
```

**Output:**



```
data.var()
```

**Output:**

Out[28]:	quarter	2.499332e-01
	volume	2.510263e+16
	percent_change_price	6.339363e+00
	percent_change_volume_over_last_wk	1.643774e+03
	previous_weeks_volume	2.535490e+16
	percent_change_next_weeks_price	7.179926e+00
	days_to_next_dividend	2.146941e+03
	percent_return_next_dividend	9.331917e-02
	dtype:	float64

data.mean()

**Output:**

```
Out[29]: quarter          1.520000e+00
          volume           1.175478e+08
          percent_change_price 5.026241e-02
          percent_change_volume_over_last_wk 5.593627e+00
          previous_weeks_volume   1.173876e+08
          percent_change_next_weeks_price 2.384681e-01
          days_to_next_dividend    5.252533e+01
          percent_return_next_dividend 6.918256e-01
          dtype: float64
```

data.mode()

**Output:**

```
quarter stock      date  open  high   low close  volume  percent_change_price  percent_change_volume_over_last_wk  previous_weeks_volume
0       2.0     AA  1/14/2011 37.26 12.60 16.97 33.07 9718851          0.0          -61.433175          9718851.0
1      NaN     AXP 1/21/2011 44.75 17.24 63.40 36.00 10135730          NaN          -60.988311          10135730.0
2      NaN     BA 1/28/2011 NaN 17.64 NaN 41.52 10705548          NaN          -59.994858          10705548.0
3      NaN     BAC 1/7/2011 NaN 19.39 NaN 46.25 11585909          NaN          -59.668839          11585909.0
4      NaN     CAT 2/11/2011 NaN 20.20 NaN NaN 11690792          NaN          -57.488522          11690792.0
...
745     NaN     NaN  NaN  NaN  NaN  NaN 889460755          NaN          NaN          NaN
746     NaN     NaN  NaN  NaN  NaN  NaN 982445809          NaN          NaN          NaN
747     NaN     NaN  NaN  NaN  NaN  NaN 1000362015          NaN          NaN          NaN
748     NaN     NaN  NaN  NaN  NaN  NaN 1054415375          NaN          NaN          NaN
749     NaN     NaN  NaN  NaN  NaN  NaN 1453438639          NaN          NaN          NaN
750 rows × 16 columns
```

data.median()

**Output:**

```
Out[31]: quarter          2.000000e+00
          open            4.597000e+01
          high            4.688500e+01
          low             4.480000e+01
          close            4.593000e+01
          volume           5.306088e+07
          percent_change_price 0.000000e+00
          percent_change_volume_over_last_wk 5.125859e-01
          previous_weeks_volume   5.294556e+07
          next_weeks_open        4.601500e+01
          next_weeks_close       4.612500e+01
          percent_change_next_weeks_price 1.011926e-01
          days_to_next_dividend 4.700000e+01
          percent_return_next_dividend 6.810665e-01
          dtype: float64
```

data.skew()

**Output:**

```
Out[32]: quarter          -0.080225
          open            1.269375
          high            1.269406
          low             1.279839
          close            1.274375
          volume           3.223405
          percent_change_price -0.380849
          percent_change_volume_over_last_wk 2.544001
          previous_weeks_volume   3.256532
          next_weeks_open        1.273419
          next_weeks_close       1.278622
          percent_change_next_weeks_price -0.172372
          days_to_next_dividend 3.055018
          percent_return_next_dividend 0.394347
          dtype: float64
```

```
data.kurt()
```

**Output:**

Out[33]:	quarter	-1.998902
	open	2.007631
	high	1.991205
	low	2.079614
	close	2.028985
	volume	14.197635
	percent_change_price	2.706643
	percent_change_volume_over_last_wk	13.900139
	previous_weeks_volume	14.426624
	next_weeks_open	2.018298
	next_weeks_close	2.037552
	percent_change_next_weeks_price	2.268632
	days_to_next_dividend	13.312999
	percent_return_next_dividend	0.389911
	dtype:	float64

```
data.std()
```

**Output:**

Out[34]:	quarter	4.999332e-01
	volume	1.584381e+08
	percent_change_price	2.517809e+00
	percent_change_volume_over_last_wk	4.054348e+01
	previous_weeks_volume	1.592322e+08
	percent_change_next_weeks_price	2.679538e+00
	days_to_next_dividend	4.633510e+01
	percent_return_next_dividend	3.054819e-01
	dtype:	float64

```
df=data['stock'].value_counts()
```

```
df
```

**Output:**

Out[7]:	AA	25	JNJ	25
	AXP	25	INTC	25
	WMT	25	IBM	25
	VZ	25	HPQ	25
	UTX	25	HD	25
	TRV	25	GE	25
	T	25	DIS	25
	PG	25	DD	25
	PFE	25	CVX	25
	MSFT	25	CSCO	25
	MRK	25	CAT	25
	MMM	25	BAC	25
	MCD	25	BA	25
	KO	25	XOM	25
	KRFT	25		Name: stock, dtype: int64
	JPM	25		

**Result:**

Univariate analysis has been successfully done for dow jones index dataset and the results are verified.

### 3)Aim:

To perform the following using dow jones index dataset a)Univariate Linear Regression b)Summary statistics c)Frequency table d)Charts e)univariate analysis for numerical and categorical attributes f)Univariate Function Optimization g) Perform Feature Extraction with Univariate Statistical Tests and summarize the selected features

### Code & Output:

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

```
data = pd.read_csv('dow_jones_index.csv')
```

#### **#summary statistics**

```
data.describe()
```

### Output:

In [10]:	data.describe()
Out[10]:	quarter      volume      percent_change_price      percent_change_volume_over_last_wk      previous_weeks_volume      percent_change_next_weeks_price count    750.000000 7.500000e+02    750.000000    720.000000 7.200000e+02    750.000000 mean    1.520000 1.175478e+08    0.050262    5.593627 1.173876e+08    0.238468 std    0.499933 1.584381e+08    2.517809    40.543478 1.592322e+08    2.679538 min    1.000000 9.718851e+06    -15.422900    -61.433175 9.718851e+06    -15.422900 25%    1.000000 3.086624e+07    -1.288053    -19.804284 3.067832e+07    -1.222068 50%    2.000000 5.306088e+07    0.000000    0.512586 5.294556e+07    0.101193 75%    2.000000 1.327218e+08    1.650888    21.800622 1.333230e+08    1.845562 max    2.000000 1.453439e+09    9.882230    327.408924 1.453439e+09    9.882230

```
data['volume'].value_counts()
```

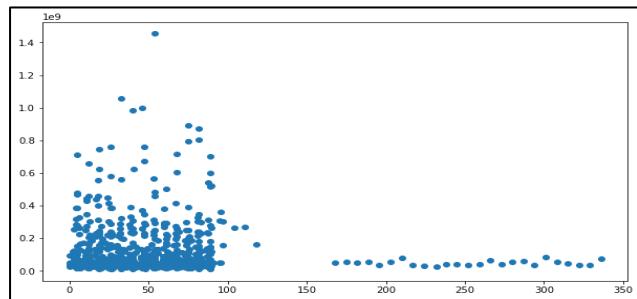
### Output:

In [11]:	data['volume'].value_counts()
Out[11]:	239655616    1 90069975    1 61019095    1 52439439    1 41391929    1 .. 328646154    1 227601331    1 220040646    1 457318851    1 118679791    1 Name: volume, Length: 750, dtype: int64

**#charts**

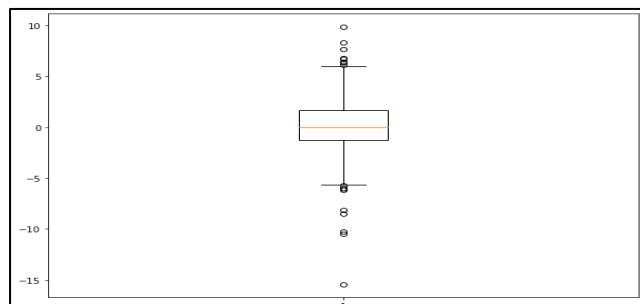
```
y=data['volume']
x=data['days_to_next_dividend']
plt.rcParams['figure.figsize']=(10,6)
plt.scatter(x,y)
```

**Output:**



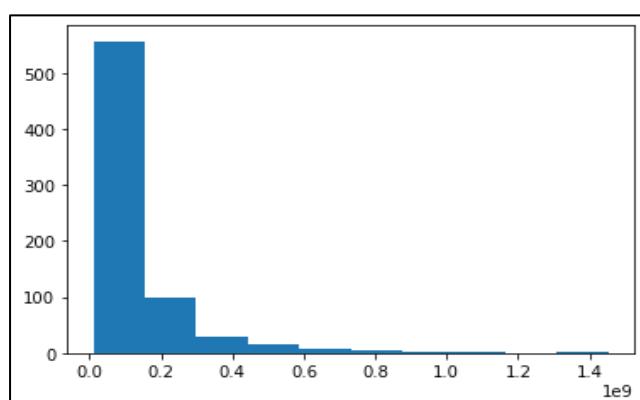
```
fig=plt.figure(figsize=(10,7))
plt.boxplot(data['percent_change_price'])
plt.show()
```

**Output:**



```
plt.hist(data['previous_weeks_volume'])
```

**Output:**

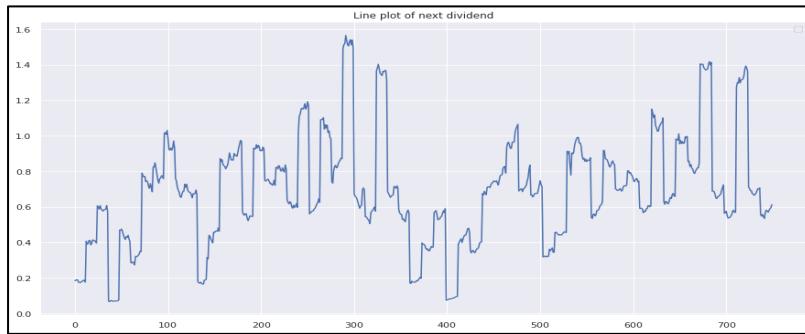


```

plt.title("Line plot of next dividend")
plt.plot(data.index,data['percent_return_next_dividend'])
plt.legend()
plt.show()

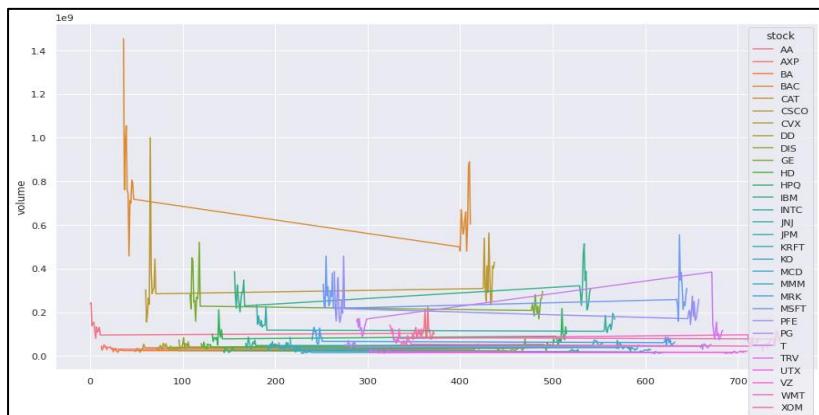
```

**Output:**



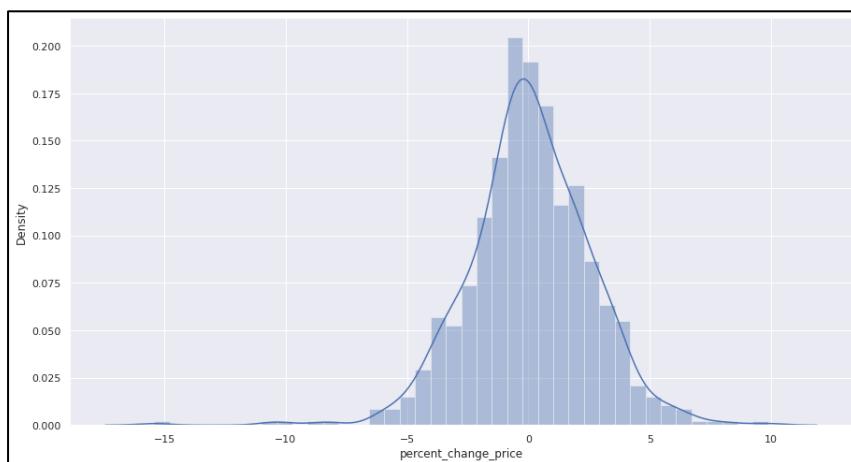
```
fig=sns.lineplot(x=data.index,y=data['volume'],hue=data['stock'])
```

**Output:**



```
sns.distplot(data['percent_change_price'])
```

**Output:**



### #univariate linear regression

```
from sklearn.linear_model import LinearRegression  
model = LinearRegression().fit(x,y)  
r_sq=model.score(x,y)  
print("R square value :",r_sq)
```

### Output:

```
In [25]: from sklearn.linear_model import LinearRegression  
model = LinearRegression().fit(x,y)  
r_sq=model.score(x,y)  
print("R square value :",r_sq)
```

R square value : 0.2623896707626756

### #frequency table

```
data['stock'].value_counts()
```

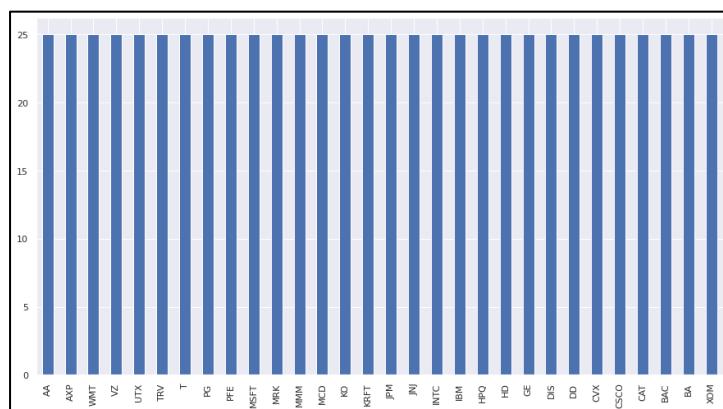
### Output:

Out[12]:	AA	25	INTC	25
	AXP	25	IBM	25
	WMT	25	HPQ	25
	VZ	25	HD	25
	UTX	25	GE	25
	TRV	25	DIS	25
	T	25	DD	25
	PG	25	CVX	25
	PFE	25	CSCO	25
	MSFT	25	CAT	25
	MRK	25	BAC	25
	MMM	25	BA	25
	MCD	25	XOM	25
	KO	25	Name: stock, dtype: int64	
	KRFT	25		
	JPM	25		
	JNJ	25		
	INTC	25		

### #univariate analysis numerical attributes

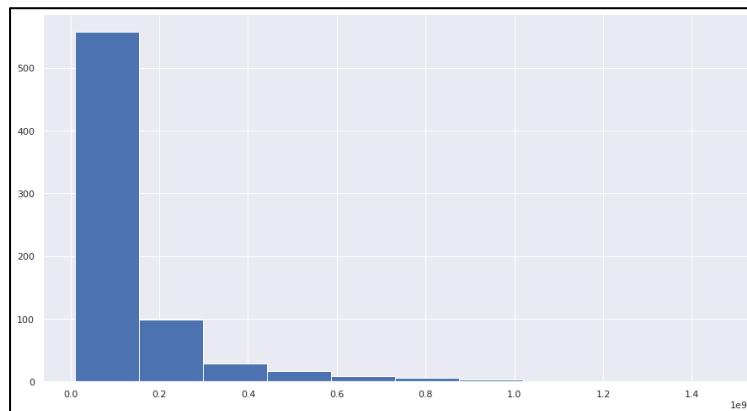
```
data['stock'].value_counts().plot.bar()
```

### Output:



```
plt.hist(data['previous_weeks_volume'])
```

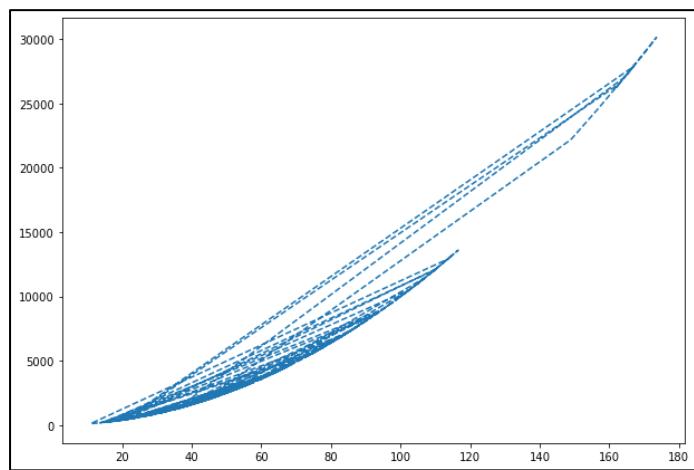
**Output:**



**#function optimization**

```
def objective(x):
    return (0.005 + x)**2
inputs = data['high']
targets = [objective(x) for x in inputs]
fig=plt.figure(figsize=(10,7))
plt.plot(inputs, targets,'--')
plt.show()
```

**Output:**



**#univariate statistical tests**

**#correlation test**

```
from scipy.stats import pearsonr
data['high']= pd.to_numeric(data['high'])
```

```

data['low']= pd.to_numeric(data['low'])

list1 = data['high']

list2 = data['low']

corr,_= pearsonr(list1, list2)

print('Pearsons correlation: %.3f'% corr)

```

**Output:**

Pearsons correlation: 0.999

**#normality test**

```

from scipy import stats

z=[data['high'],data['low']]

for i in z:

    print([i])

    a,b=stats.normaltest(data['high'])

    print(a,b)

    alpha = 0.05

    if b<alpha :

        print('null hypothesis rejected')

    else:

        print('null hypothesis cannot be rejected')

```

**Output:**

```

[0      16.72
 1      16.71
 2      16.38
 3      16.63
 4      17.39
 ...
745     82.63
746     83.75
747     81.87
748     80.82
749     81.12
Name: high, Length: 750, dtype: float64]
165.8155807197986 9.85380899939066e-37
null hypothesis rejected
[0      15.78
 1      15.64
 2      15.60
 3      15.82
 4      16.18
 ...
745     80.07
746     80.18
747     79.72
748     78.33
749     76.78
Name: low, Length: 750, dtype: float64]
165.8155807197986 9.85380899939066e-37
null hypothesis rejected

```

```
#parametric statistical hypothesis test
```

```
from scipy.stats import ttest_ind  
  
z=[data['high'],data['low']]  
  
for i in z:  
  
    print([i])  
  
    a,b=ttest_ind(data['high'],data['low'])  
  
    print(a,b)  
  
    alpha = 0.05  
  
    if b>alpha :  
  
        print('Same distributions')  
  
    else:  
  
        print('Different distributions')
```

**Output:**

```
[0      16.72  
1      16.71  
2      16.38  
3      16.63  
4      17.39  
     ...  
745    82.63  
746    83.75  
747    81.87  
748    80.82  
749    81.12  
Name: high, Length: 750, dtype: float64]  
1.2030833315757794 0.22913417910648973  
Same distributions  
[0      15.78  
1      15.64  
2      15.60  
3      15.82  
4      16.18  
     ...  
745    80.07  
746    80.18  
747    79.72  
748    78.33  
749    76.78  
Name: low, Length: 750, dtype: float64]  
1.2030833315757794 0.22913417910648973  
Same distributions
```

```
#non parametric statistical hypothesis test
```

```
from scipy.stats import mannwhitneyu  
z=[data['high'],data['low']]  
for i in z:  
    print([i])  
    a,b=mannwhitneyu(data['high'],data['low'])  
    print(a,b)  
    alpha = 0.05  
    if b>alpha :  
        print('Same distributions')  
    else:  
        print('Different distributions')
```

### **Output:**

```
[0      16.72  
1      16.71  
2      16.38  
3      16.63  
4      17.39  
...  
745     82.63  
746     83.75  
747     81.87  
748     80.82  
749     81.12  
Name: high, Length: 750, dtype: float64]  
292745.5 0.17056138472040383  
Same distributions  
[0      15.78  
1      15.64  
2      15.60  
3      15.82  
4      16.18  
...  
745     80.07  
746     80.18  
747     79.72  
748     78.33  
749     76.78  
Name: low, Length: 750, dtype: float64]  
292745.5 0.17056138472040383  
Same distributions
```

### **Result:**

Univariate analysis like linear regression , charts , analysis for numerical and categorical attributes , feature extraction has been done successfully and the results are verified.

## UNIVARIATE TIME SERIES ANALYSIS

### 1)Aim:

To get a time series data set and perform the following: a)create time series features  
b)plot feature importance c)forecast and evaluate the data

### Code & Output:

```
import pandas as pd
from pandas.plotting import autocorrelation_plot
from pandas import DataFrame
from pandas import concat
import numpy as np
from math import sqrt

from sklearn.metrics import mean_squared_error
from statsmodels.tsa.seasonal import seasonal_decompose
from statsmodels.tsa.stattools import adfuller
from statsmodels.tsa.stattools import acf
from statsmodels.tsa.stattools import pacf
from statsmodels.tsa.arima.model import ARIMA
from scipy.stats import boxcox
import seaborn as sns
sns.set_style("whitegrid")
import matplotlib.pyplot as plt
from matplotlib.pylab import rcParams
from matplotlib import colors
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')

col_names = ["date", "value"]
```

```

df = pd.read_csv("Electric_Production.csv",
                  names = col_names, header = 0, parse_dates = [0])
df['date'] = pd.to_datetime(df['date'],infer_datetime_format=True)
df = df.set_index(['date'])
df.head()

```

### **Output:**

Out[2]:	value
date	
1985-01-01	72.5052
1985-02-01	70.6720
1985-03-01	62.4502
1985-04-01	57.4714
1985-05-01	55.3151

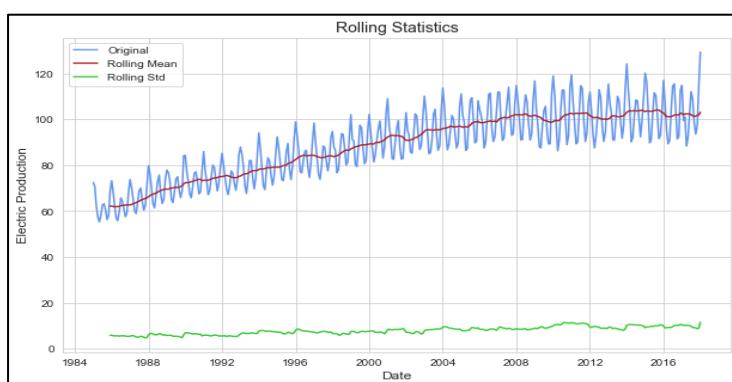
### **#time series features**

```

rolling_mean = df.rolling(window=12).mean()
rolling_std = df.rolling(window=12).std()
plt.figure(figsize = (10,6))
plt.plot(df, color='cornflowerblue', label='Original')
plt.plot(rolling_mean, color='firebrick', label='Rolling Mean')
plt.plot(rolling_std, color='limegreen', label='Rolling Std')
plt.xlabel('Date', size = 12)
plt.ylabel('Electric Production', size = 12)
plt.legend(loc = 'upper left')
plt.title('Rolling Statistics', size = 14)
plt.show()

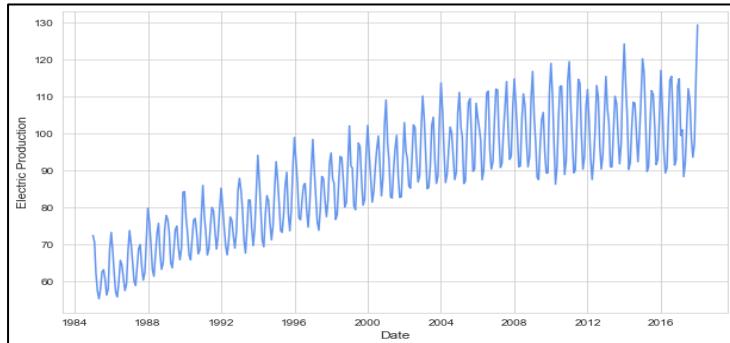
```

### **Output:**



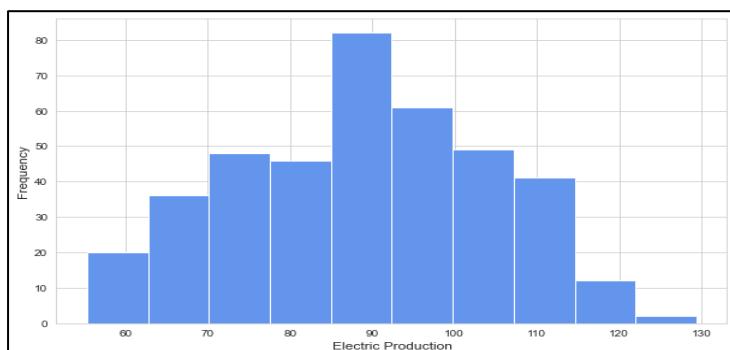
```
plt.figure(figsize = (10,6))
plt.plot(df['value'], color = 'cornflowerblue')
plt.xlabel('Date', size = 12)
plt.ylabel('Electric Production', size = 12)
plt.show()
```

**Output:**



```
plt.figure(figsize = (10,6))
plt.hist(df['value'], color = 'cornflowerblue')
plt.xlabel('Electric Production', size = 12)
plt.ylabel('Frequency', size = 12)
plt.show()
```

**Output:**



```
print("Data Shape: {}" .format(df.shape))
value_1 = df[0:199]
value_2 = df[200:397]
```

**Output:**

```
Data Shape: (397, 1)
```

```
print("Mean of value_1: {}".format(round(value_1.mean()[0],3)))  
print("Mean of value_2: {}".format(round(value_2.mean()[0],3)))
```

**Output:**

```
Mean of value_1: 77.497  
Mean of value_2: 100.258
```

```
print("Variance of value_1: {}".format(round(value_1.var()[0],3)))  
print("Variance of value_2: {}".format(round(value_2.var()[0],3)))
```

**Output:**

```
Variance of value_1: 123.226  
Variance of value_2: 91.677
```

```
def adfuller_test(ts, window = 12):  
  
    movingAverage = ts.rolling(window).mean()  
  
    movingSTD = ts.rolling(window).std()  
  
    plt.figure(figsize = (10,6))  
  
    orig = plt.plot(ts, color='cornflowerblue',label='Original')  
  
    mean = plt.plot(movingAverage, color='firebrick',label='Rolling Mean')  
  
    std = plt.plot(movingSTD, color='limegreen',label='Rolling Std')  
  
    plt.legend(loc = 'upper left')  
  
    plt.title('Rolling Statistics', size = 14)  
  
    plt.show(block=False)  
  
    adf = adfuller(ts, autolag='AIC')  
  
    print('ADF Statistic: {}'.format(round(adf[0],3)))  
  
    print('p-value: {}'.format(round(adf[1],3)))  
  
    print("#####")  
  
    print('Critical Values:')  
  
  
    for key, ts in adf[4].items():  
  
        print('{}: {}'.format(key, round(ts,3)))  
  
    print("#####")
```

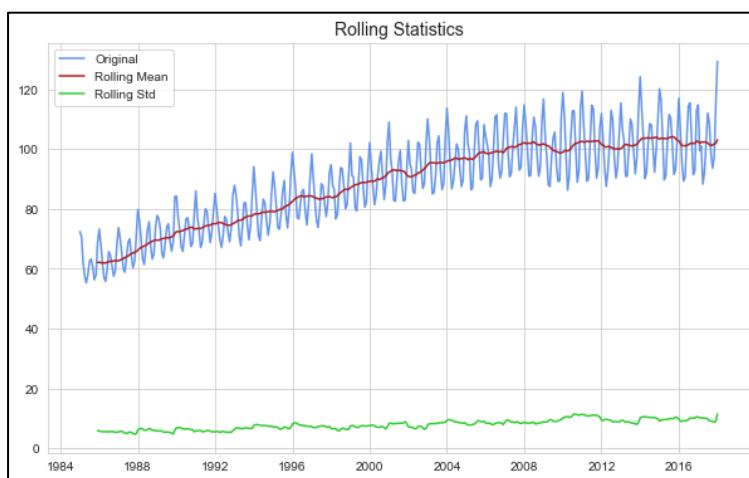
```

if adf[0] > adf[4]["5%"]:
    print("ADF > Critical Values")
    print ("Failed to reject null hypothesis, time series is non-stationary.")
else:
    print("ADF < Critical Values")
    print ("Reject null hypothesis, time series is stationary.")

adfuller_test(df, window = 12)

```

### **Output:**



```

ADF Statistic: -2.257
p-value: 0.186
#####
Critical Values:
1%: -3.448
5%: -2.869
10%: -2.571
#####
ADF > Critical Values
Failed to reject null hypothesis, time series is non-stationary.

```

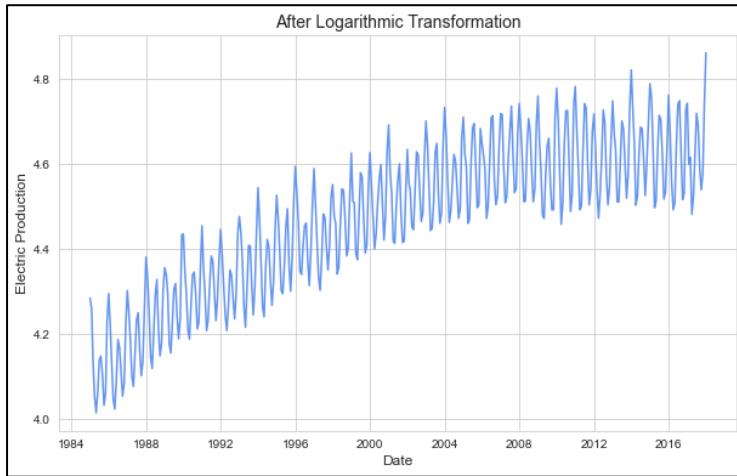
```

df_log_scaled = df
df_log_scaled['value'] = boxcox(df_log_scaled['value'], lmbda=0.0)
plt.figure(figsize = (10,6))
plt.plot(df_log_scaled, color = 'cornflowerblue')
plt.xlabel('Date', size = 12)
plt.ylabel('Electric Production', size = 12)
plt.title("After Logarithmic Transformation", size = 14)

```

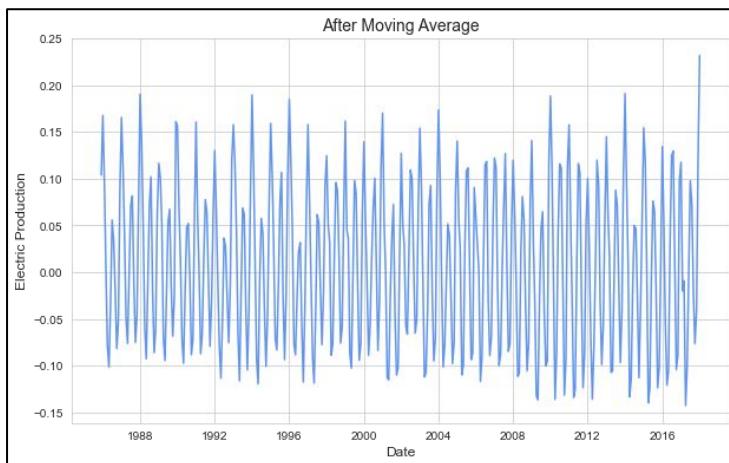
```
plt.show()
```

### Output:



```
moving_avg = df_log_scaled.rolling(window=12).mean()  
df_log_scaled_ma = df_log_scaled - moving_avg  
df_log_scaled_ma.dropna(inplace=True)  
plt.figure(figsize = (10,6))  
plt.plot(df_log_scaled_ma, color = 'cornflowerblue')  
plt.xlabel('Date', size = 12)  
plt.ylabel('Electric Production', size = 12)  
plt.title("After Moving Average", size = 14)  
plt.show()
```

### Output:



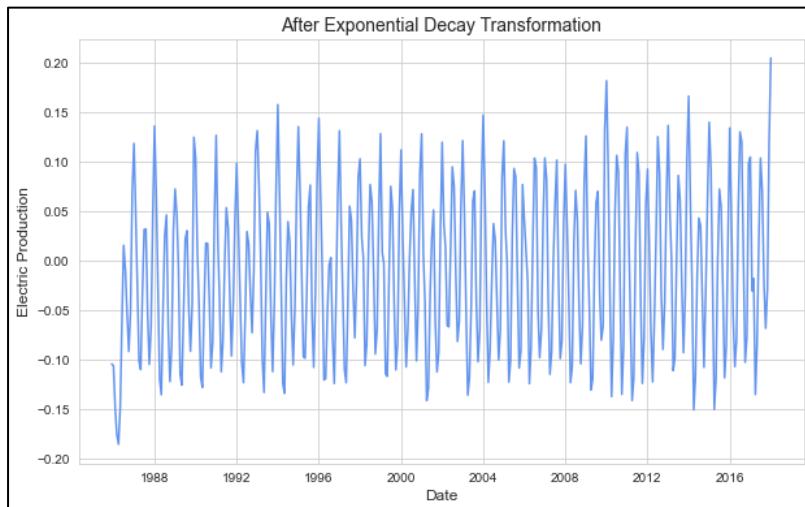
```
df_log_scaled_ma_ed = df_log_scaled_ma.ewm(halflife=12, min_periods=0,  
adjust=True).mean()
```

```

df_lsma_sub_df_lsma_ed = df_log_scaled_ma - df_log_scaled_ma_ed
plt.figure(figsize = (10,6))
plt.plot(df_lsma_sub_df_lsma_ed - df_log_scaled_ma_ed, color='cornflowerblue')
plt.xlabel('Date', size = 12)
plt.ylabel('Electric Production', size = 12)
plt.title("After Exponential Decay Transformation", size = 14)
plt.show()

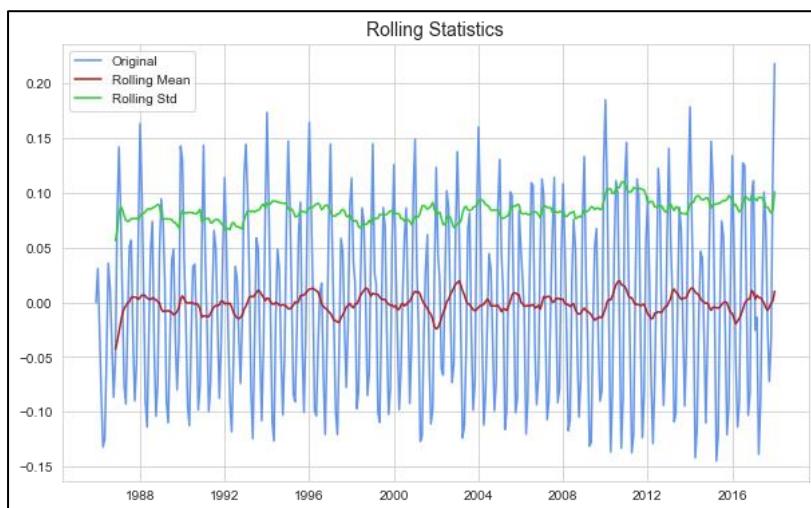
```

**Output:**



```
adfuller_test(df_lsma_sub_df_lsma_ed, window = 12)
```

**Output:**



```

ADF Statistic: -7.213
p-value: 0.0
#####
Critical Values:
1%: -3.448
5%: -2.869
10%: -2.571
#####
ADF < Critical Values
Reject null hypothesis, time series is stationary.

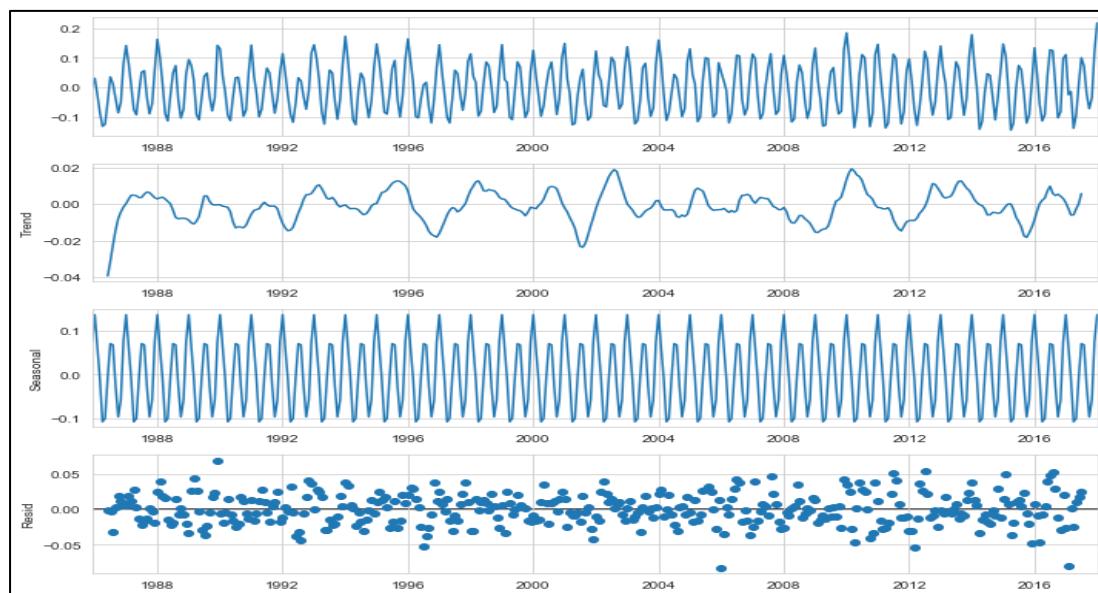
```

```

rcParams['figure.figsize']=10,8
df_seasonal_decompose = seasonal_decompose(df_lsma_sub_df_lsma_ed,
                                             model='duplicative')
df_seasonal_decompose.plot()
plt.show()

```

### Output:



```

auto_c_f = acf(df_lsma_sub_df_lsma_ed, nlags=20)
partial_auto_c_f = pacf(df_lsma_sub_df_lsma_ed, nlags=20, method='ols')
fig, axs = plt.subplots(1, 2, figsize =(12,5))
plt.subplot(121)
plt.plot(auto_c_f)
plt.axhline(y=0, linestyle='--', color='limegreen')
plt.axhline(y=-1.96/np.sqrt(len(df_lsma_sub_df_lsma_ed)),linestyle='--', color='firebrick')
plt.axhline(y=1.96/np.sqrt(len(df_lsma_sub_df_lsma_ed)),linestyle='--', color='firebrick')

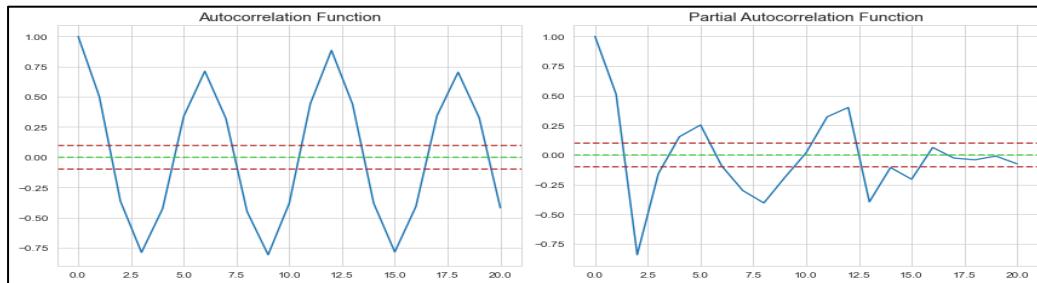
```

```

plt.title('Autocorrelation Function', size = 14)
plt.subplot(122)
plt.plot(partial_auto_c_f)
plt.axhline(y=0, linestyle='--', color='limegreen')
plt.axhline(y=-1.96/np.sqrt(len(df_lsma_sub_df_lsma_ed)), linestyle='--', color='firebrick')
plt.axhline(y=1.96/np.sqrt(len(df_lsma_sub_df_lsma_ed)),linestyle='--', color='firebrick')
plt.title('Partial Autocorrelation Function', size = 14)
plt.tight_layout()

```

### **Output:**



```

values = DataFrame(df_lsma_sub_df_lsma_ed.values)
persistence_df = concat([values.shift(1), values], axis=1)
persistence_df.columns = ['t-1', 't+1']
per_values = persistence_df.values
train = per_values[1:len(per_values)-10]
test = per_values[len(per_values)-10:]
X_train, y_train = train[:,0], train[:,1]
X_test, y_test = test[:,0], test[:,1]
def persistence(x):
    return x
predictions = []
for i in X_test:
    y_pred = persistence(i)
    predictions.append(y_pred)
persistence_score = mean_squared_error(y_test, predictions)
print('Persistence MSE: {}'.format(round(persistence_score,4)))

```

**Output:**

```
Persistence MSE: 0.0084
```

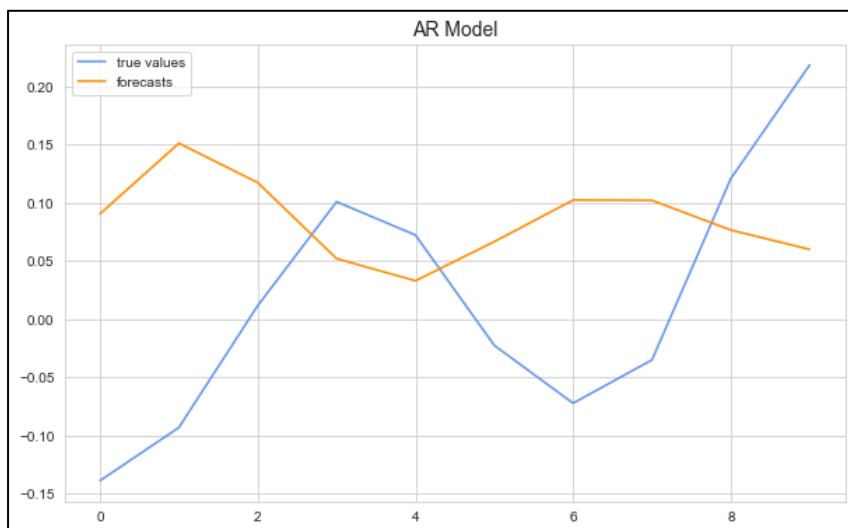
```
ar_values = df_lsma_sub_df_lsma_ed.values  
train = ar_values[1:len(ar_values)-10]  
test = ar_values[len(ar_values)-10:]  
model = ARIMA(train, order=(2,1,0))  
AR_model = model.fit()  
predictions = AR_model.predict(start=len(train), end=len(train)+len(test)-1, dynamic=False)  
ar_score = mean_squared_error(test, predictions)  
print('AR MSE: {}'.format(round(ar_score,4)))
```

**Output:**

```
AR MSE: 0.0211
```

```
plt.figure(figsize = (10,6))  
plt.plot(test, label = "true values", color = "cornflowerblue")  
plt.plot(predictions,label = "forecasts", color='darkorange')  
plt.title("AR Model", size = 14)  
plt.legend(loc = 'upper left')  
plt.show()
```

**Output:**



```

model = ARIMA(train, order=(0,1,2))
MA_model = model.fit()
predictions = MA_model.predict(start=len(train), end=len(train)+len(test)-1, dynamic=False)
ma_score = mean_squared_error(test, predictions)
print('MA MSE: {}'.format(round(ma_score,4)))

```

**Output:**

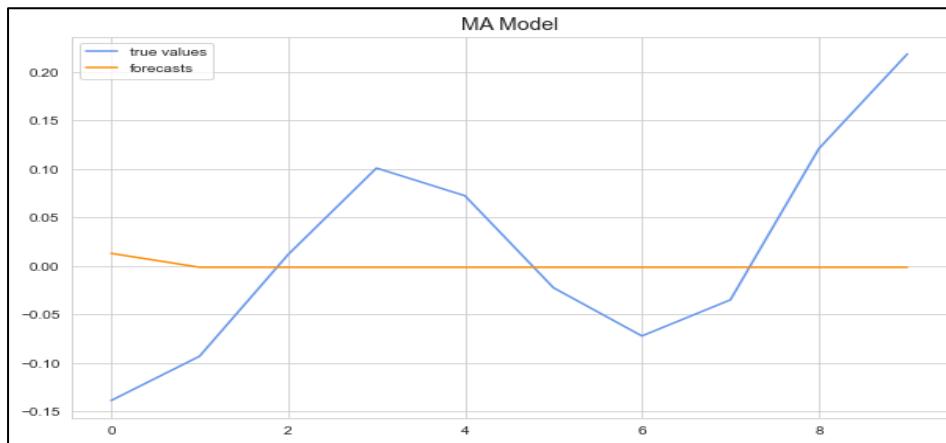
MA MSE: 0.0117

```

plt.figure(figsize = (10,6))
plt.plot(test, label = "true values", color = "cornflowerblue")
plt.plot(predictions,label = "forecasts", color='darkorange')
plt.title("MA Model", size = 14)
plt.legend(loc = 'upper left')
plt.show()

```

**Output:**



```

model = ARIMA(train, order=(2,1,2))
ARIMA_model = model.fit()
predictions = ARIMA_model.predict(start=len(train), end=len(train)+len(test)-1,
dynamic=False)
arima_score = mean_squared_error(test, predictions)
print('ARIMA MSE: {}'.format(round(arima_score,4)))

```

**Output:**

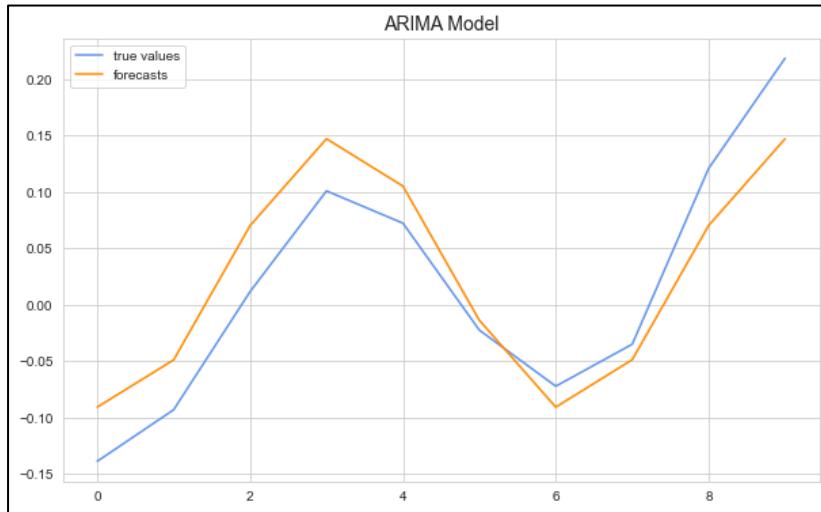
ARIMA MSE: 0.0019

```

plt.figure(figsize = (10,6))
plt.plot(test, label = "true values", color = "cornflowerblue")
plt.plot(predictions,label = "forecasts", color='darkorange')
plt.title("ARIMA Model", size = 14)
plt.legend(loc = 'upper left')
plt.show()

```

**Output:**



```

errors = pd.DataFrame()
errors["Model"] = ["Persistence", "Autoregression", "Moving Average", "ARIMA"]
errors["MSE"] = [persistence_score, ar_score, ma_score, arima_score]
errors = errors.sort_values("MSE", ascending = True, ignore_index = True)
errors.index = errors.Model
del errors["Model"]

```

```

def coloring_bg(s, min_-, max_, cmap='Reds', low=0, high=0):
    color_range = max_- min_-
    norm = colors.Normalize(min_- (color_range * low), max_- + (color_range * high))
    normed = norm(s.values)
    c = [colors.rgb2hex(x) for x in plt.cm.get_cmap(cmap)(normed)]
    return ['background-color: %s' % color for color in c]

```

```
errors.style.apply(coloring_bg,min_=errors.min().min(),  
max_= errors.max().max(), low = 0.1, high = 0.85)
```

**Output:**

Model	MSE
ARIMA	0.001913
Persistence	0.008401
Moving Average	0.011716
Autoregression	0.021126

**Result:**

Univariate time analysis has been done successfully and the results are verified.

## **OPENSTACK**

### **1)Aim:**

To understand openstack deployment, its implementation and its application.

### **Openstack:**

Open stack is a free, open standard cloud computing platform. It is mostly deployed as Infrastructure as Service (Iaas) in both public and private clouds where virtual servers and other resource are made available to users. The software platform consists of inter related components that control diverse, multi-vendor hardware pools of processing, storage and networking resources throughout a data center.

### **Deployment Models:**

#### **1)Open stack based public cloud**

A vendor provides a public cloud computing system based on the OpenStack project

#### **2) On-premises distribution**

In this model, a customer downloads and installs an OpenStack distribution in their internal network.

#### **3) Hosted OpenStack Private Cloud**

A vendor hosts an OpenStack-based private cloud: including the underlying hardware and the OpenStack software

#### **4) OpenStack-as-a-Service**

A vendor hosts OpenStack management software (without any hardware) as a service. Customers sign up for the service and pair it with their internal servers, storage and networks to get a fully operational private cloud

#### **5) Appliance based OpenStack**

Nebula was a vendor that sold appliances that could be plugged into a network which spawned an OpenStack deployment

### **Implementation:**

#### **1)We use Devstack to shell scrip the complete openstack development environment.**

#### **2)Install all components without clustering**

#### **3)Install Logical Volume Management (LVM). This process includes allocating disks , striping, mirroring and resizing logical volumes.**

#### **4)Install MaaS (Metal as a Service).**

5)Install Trove (Redstack)

**Applications:**

- a) Nova – It is the primary computing engine
- b) Swift – It is the storage system. It is used to give unique ID to every users.
- c) Cinder – It is a block storage component, is used to access data in a very high speed.
- d) Newton – It is the networking control and helps the components of OpenStack to communicate with one another.
- e) Glance – It allows images to be used as templates.
- f) Heat – It allows developers to store all requirements of each cloud application in a single file.
- g) Keystone – It provides with identity services and helps to map user's access methods

**Result:**

Openstack deployment, implementation and application's study has been done successfully.

## OPENSTACK INSTALLATION

### 1)Aim:

To implement infrastructure as service using Open Stack.

### Installation:

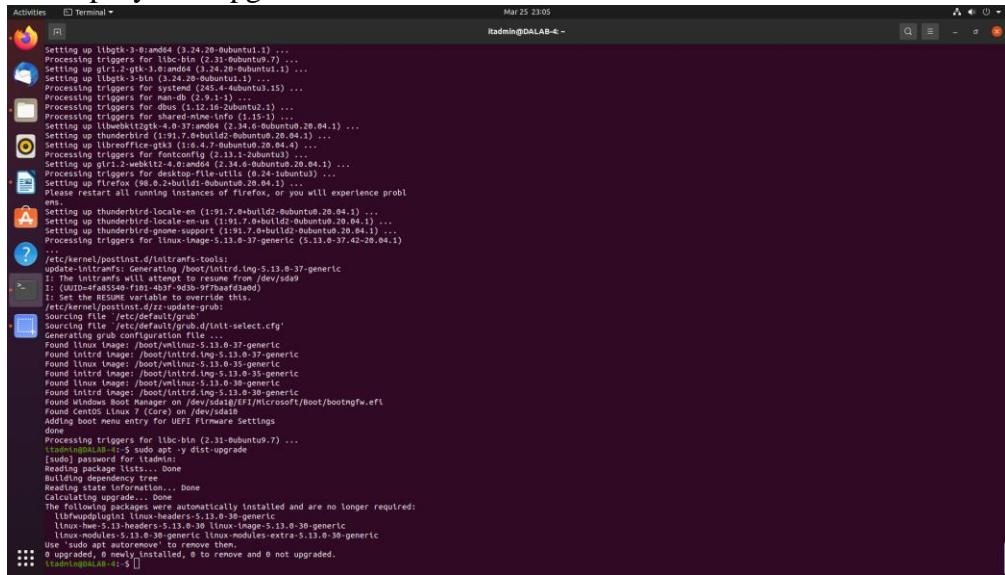
1. Login to sudo ("superuser do") user
2. Open Terminal
3. In Terminal type : sudo apt – update

```
Administrator@DALAB-41:~$ sudo apt update
Get:1 http://in.archive.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:2 http://in.archive.ubuntu.com/ubuntu focal InRelease [114 kB]
Get:3 http://in.archive.ubuntu.com/ubuntu focal-updates InRelease [108 kB]
Get:4 http://in.archive.ubuntu.com/ubuntu focal-updates/restricted Packages [1,074 kB]
Get:5 http://in.archive.ubuntu.com/ubuntu focal-updates/main i386 Packages [621 kB]
Get:6 http://in.archive.ubuntu.com/ubuntu focal-security/main amd64 Packages [1,347 kB]
Get:7 http://in.archive.ubuntu.com/ubuntu focal-updates/main i386 Packages [1,347 kB]
Get:8 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [1,347 kB]
Get:9 http://in.archive.ubuntu.com/ubuntu focal-updates/main i386 Translation-en [135 kB]
Get:10 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 Translation-en [135 kB]
Get:11 http://in.archive.ubuntu.com/ubuntu focal-updates/main DEP-11 48x48 Icons [278 kB]
Get:12 http://in.archive.ubuntu.com/ubuntu focal-updates/main DEP-11 48x48 Icons [278 kB]
Get:13 http://in.archive.ubuntu.com/ubuntu focal-updates/main DEP-11 48x48 Icons [278 kB]
Get:14 http://in.archive.ubuntu.com/ubuntu focal-updates/main DEP-11 48x48 Icons [278 kB]
Get:15 http://in.archive.ubuntu.com/ubuntu focal-updates/restricted DEP-11 48x48 Icons [278 kB]
Get:16 http://in.archive.ubuntu.com/ubuntu focal-updates/restricted DEP-11 48x48 Icons [278 kB]
Get:17 http://in.archive.ubuntu.com/ubuntu focal-updates/universe DEP-11 48x48 Icons [278 kB]
Get:18 http://in.archive.ubuntu.com/ubuntu focal-updates/universe DEP-11 48x48 Icons [278 kB]
Get:19 http://in.archive.ubuntu.com/ubuntu focal-updates/universe DEP-11 48x48 Icons [278 kB]
Get:20 http://in.archive.ubuntu.com/ubuntu focal-updates/universe DEP-11 48x48 Icons [278 kB]
Get:21 http://in.archive.ubuntu.com/ubuntu focal-updates/universe DEP-11 48x48 Icons [278 kB]
Get:22 http://in.archive.ubuntu.com/ubuntu focal-updates/universe DEP-11 48x48 Icons [278 kB]
Get:23 http://in.archive.ubuntu.com/ubuntu focal-updates/universe DEP-11 48x48 Icons [278 kB]
Get:24 http://in.archive.ubuntu.com/ubuntu focal-updates/universe DEP-11 48x48 Icons [278 kB]
Get:25 http://in.archive.ubuntu.com/ubuntu focal-updates/multiverse DEP-11 48x48 Icons [278 kB]
Get:26 http://in.archive.ubuntu.com/ubuntu focal-updates/multiverse DEP-11 48x48 Icons [278 kB]
Get:27 http://in.archive.ubuntu.com/ubuntu focal-updates/multiverse DEP-11 48x48 Icons [278 kB]
Get:28 http://in.archive.ubuntu.com/ubuntu focal-updates/multiverse DEP-11 48x48 Icons [278 kB]
Get:29 http://in.archive.ubuntu.com/ubuntu focal-updates/multiverse DEP-11 48x48 Icons [278 kB]
Get:30 http://in.archive.ubuntu.com/ubuntu focal-updates/multiverse DEP-11 48x48 Icons [278 kB]
Get:31 http://in.archive.ubuntu.com/ubuntu focal-security/main i386 Packages [497 kB]
Get:32 http://in.archive.ubuntu.com/ubuntu focal-security/main amd64 DEP-11 Metadata [497 kB]
Get:33 http://in.archive.ubuntu.com/ubuntu focal-security/main amd64 DEP-11 Metadata [497 kB]
Get:34 http://in.archive.ubuntu.com/ubuntu focal-security/main DEP-11 DEP-11 Metadata [497 kB]
Get:35 http://in.archive.ubuntu.com/ubuntu focal-security/main DEP-11 DEP-11 Metadata [497 kB]
Get:36 http://in.archive.ubuntu.com/ubuntu focal-security/main DEP-11 DEP-11 Metadata [497 kB]
Get:37 http://in.archive.ubuntu.com/ubuntu focal-security/restricted DEP-11 DEP-11 Metadata [497 kB]
Get:38 http://in.archive.ubuntu.com/ubuntu focal-security/restricted DEP-11 DEP-11 Metadata [497 kB]
Get:39 http://in.archive.ubuntu.com/ubuntu focal-security/restricted DEP-11 DEP-11 Metadata [497 kB]
Get:40 http://in.archive.ubuntu.com/ubuntu focal-security/restricted DEP-11 DEP-11 Metadata [497 kB]
Get:41 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:42 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:43 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:44 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:45 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:46 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:47 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:48 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:49 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:50 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:51 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:52 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:53 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:54 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:55 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:56 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:57 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:58 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:59 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:60 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:61 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:62 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:63 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:64 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:65 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:66 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:67 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:68 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:69 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:70 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:71 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:72 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:73 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:74 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:75 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:76 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:77 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:78 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:79 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:80 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:81 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:82 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:83 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:84 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:85 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:86 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:87 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:88 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:89 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:90 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:91 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:92 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:93 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:94 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:95 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:96 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:97 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:98 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Get:99 http://in.archive.ubuntu.com/ubuntu focal-security/universe DEP-11 DEP-11 Metadata [497 kB]
Reading package lists... Done
Building dependency tree
Reading state information... Done
99 packages can be upgraded. Run 'apt list --upgradable' to see them.
Administrator@DALAB-41:~$
```

4. sudo apt -y upgrade

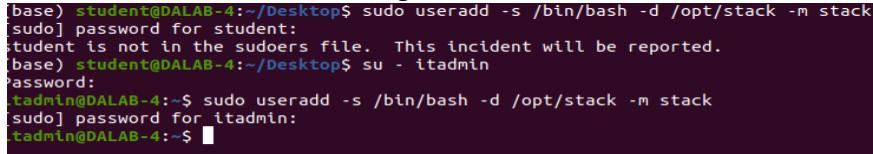
```
Administrator@DALAB-41:~$ sudo apt -y upgrade
Setting up libibreoffice-gnome (1:6.4.7-0ubuntu0.20.04.4) ...
Setting up libibreoffice-impress (1:6.4.7-0ubuntu0.20.04.4) ...
Setting up libibreoffice-base-core (1:6.4.7-0ubuntu0.20.04.4) ...
Setting up libpython3.6-stdlib (3.6.9-1~18.04.1) ...
Setting up libreoffice-ogltrans (1:6.4.7-0ubuntu0.20.04.4) ...
Setting up libreoffice-draw (1:6.4.7-0ubuntu0.20.04.4) ...
Setting up libreoffice-writer (1:6.4.7-0ubuntu0.20.04.4) ...
Processing triggers for mime-support (3.64ubuntu0.1) ...
Processing triggers for gnome-menus (3.36.0-1ubuntu1) ...
Processing triggers for libglade2.0 (2.31-0ubuntu0.7) ...
Processing triggers for liblbc-bin (3.21-0ubuntu1) ...
Setting up liblbc-bin (3.21-0ubuntu1) ...
Processing triggers for system (245.4-0ubuntu15) ...
Processing triggers for dbus (1.12.16-0ubuntu2.1) ...
Processing triggers for shared-mime-info (1:15.3-1) ...
Setting up thunderbird (1:91.7.8+build2-0ubuntu0.20.04.1) ...
Setting up thunderbird-locale-en (1:91.7.8+build2-0ubuntu0.20.04.1) ...
Processing triggers for man-db (2.8.5-1ubuntu0.1) ...
Processing triggers for fontconfig (2.13.1-2ubuntu0.1) ...
Setting up gir1.2-webkit2-4.0 (4.0.4-0ubuntu1) ...
Processing triggers for libglib2.0-0 (2.40.5-0ubuntu1) ...
Setting up glib2.0-0 (2.40.5-0ubuntu1) ...
Setting up firefox (98.0.2+build1-0ubuntu0.20.04.1) ...
Please restart all running instances of Firefox, or you will experience problems.
Setting up thunderbird-locale-en (1:91.7.8+build2-0ubuntu0.20.04.1) ...
Setting up thunderbird-locale-en (1:91.7.8+build2-0ubuntu0.20.04.1) ...
Processing triggers for man-db (2.8.5-1ubuntu0.1) ...
Processing triggers for libglib2.0-0 (2.40.5-0ubuntu1) ...
Setting up libglib2.0-0 (2.40.5-0ubuntu1) ...
Setting up liblbc-bin (3.21-0ubuntu1) ...
Processing triggers for liblbc-bin (3.21-0ubuntu1) ...
Sourcing file '/etc/default/grub'
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-5.13.0-35-generic
Found initrd image: /boot/initrd.img-5.13.0-35-generic
Found linux image: /boot/vmlinuz-5.13.0-36-generic
Found initrd image: /boot/initrd.img-5.13.0-36-generic
Found linux image: /boot/vmlinuz-5.13.0-37-generic
Found initrd image: /boot/initrd.img-5.13.0-37-generic
Found linux image: /boot/vmlinuz-5.13.0-38-generic
Found initrd image: /boot/initrd.img-5.13.0-38-generic
Adding boot menu entry for UEFI Firmware Settings
done
Processing triggers for liblbc-bin (3.21-0ubuntu1) ...
Administrator@DALAB-41:~$
```

## 5. sudo apt -y dist-upgrade



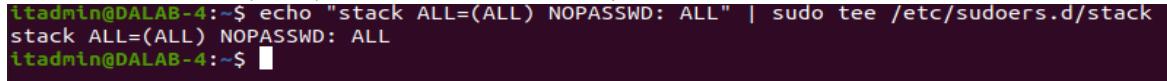
```
Setting up libgtk-3-0:amd64 (3.24.20-0ubuntu1.1) ...
Processing triggers for libc-bin (2.31-0ubuntu0.7) ...
Setting up libglib-2.0-0:amd64 (2.64.6-0ubuntu1.1) ...
Setting up libglib-2.0-0:amd64 (2.64.6-0ubuntu1.1) ...
Processing triggers for systemd (245.4-0ubuntu3.15) ...
Processing triggers for libgcc1 (1.1.0-0ubuntu1.1) ...
Processing triggers for libstdc++6 (9.3.0-1ubuntu2.1) ...
Processing triggers for shared-mime-info (1.15-1) ...
Setting up thunderbird (1:91.7.0+build2~Ubuntu0.20.04.1) ...
Setting up thunderbird (1:91.7.0+build2~Ubuntu0.20.04.1) ...
Setting up libreoffice-gtk3 (1:6.4.7~Ubuntu0.20.04.4) ...
Setting up libreoffice-gtk3 (1:6.4.7~Ubuntu0.20.04.4) ...
Setting up gcr1.2-wmkit2.4.0-standard (2.34.6~Ubuntu0.20.04.1) ...
Processing triggers for desktop-file-util (0.24~Ubuntu3) ...
Setting up gcr1.2-wmkit2.4.0-standard (2.34.6~Ubuntu0.20.04.1) ...
Please restart all running instances of Firefox, otherwise you will experience problems.
Setting up thunderbird-locale-enus (1:91.7.0+build2~Ubuntu0.20.04.1) ...
Setting up thunderbird-gnome-support (1:91.7.0+build2~Ubuntu0.20.04.1) ...
Processing triggers for linux-image-5.13.0-37-generic (5.13.0-37.42~Ubuntu0.20.04.1)
/etc/kernel/postinst.d/lntrnfts-tools:
update-intrnfts: Generating /boot/intrnd.Img 5.13.0-37-generic
1: (UUID=efab540-f01-4b1f-9d3b-977baaf3a3ed)
1: Set the RESUME variable to override this.
1: /etc/default/grub: 1: /etc/default/grub: /bin/sh: not found
Sourcing file '/etc/default/grub.d/101-select.cfg'
GRUB_DEFAULT='0'
Found linux image: '/boot/vmlinuz-5.13.0-37-generic'
Found intrnd image: '/boot/intrnd.Img-5.13.0-37-generic'
Found initrd image: '/boot/initrd.Img-5.13.0-37-generic'
Found linux image: '/boot/vmlinuz-5.13.0-38-generic'
Found intrnd image: '/boot/intrnd.Img-5.13.0-38-generic'
Found initrd image: '/boot/initrd.Img-5.13.0-38-generic'
Found Windows Boot Manager on /dev/sda1(F11/Microsoft/Boot/bootmgfw.efl)
Found Control Linux 7 (Core) on /dev/sda1
Adding boot menu entry for UEFI Firmware Settings
done
Processing triggers for liblxc-bin (2.31-0ubuntu0.7) ...
Processing triggers for man-db (2.9.1-1) ...
[sudo] password for itadmin:
Reading package lists... done
Building dependency tree
Reading state information... done
Calculating upgrade... Done
The following packages were automatically installed and are no longer required:
  linux-hwe-5.13-headers-5.13.0-30 generic linux-modules-extra-5.13.0-30-generic
  linux-hwe-5.13-headers-5.13.0-30 generic linux-modules-extra-5.13.0-30-generic
Use 'sudo apt autoremove' to remove them.
 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
[itadmin@DALAB-4:~]$
```

## 6. sudo useradd -s /bin/bash -d /opt/stack -m stack



```
[base] student@DALAB-4:~/Desktop$ sudo useradd -s /bin/bash -d /opt/stack -m stack
[sudo] password for student:
student is not in the sudoers file. This incident will be reported.
[base] student@DALAB-4:~/Desktop$ su - itadmin
Password:
[itadmin@DALAB-4:~$ sudo useradd -s /bin/bash -d /opt/stack -m stack
[sudo] password for itadmin:
[itadmin@DALAB-4:~$ ]
```

## 7. echo "stack ALL=(ALL) NOPASSWD: ALL" | sudo tee /etc/sudoers.d/stack

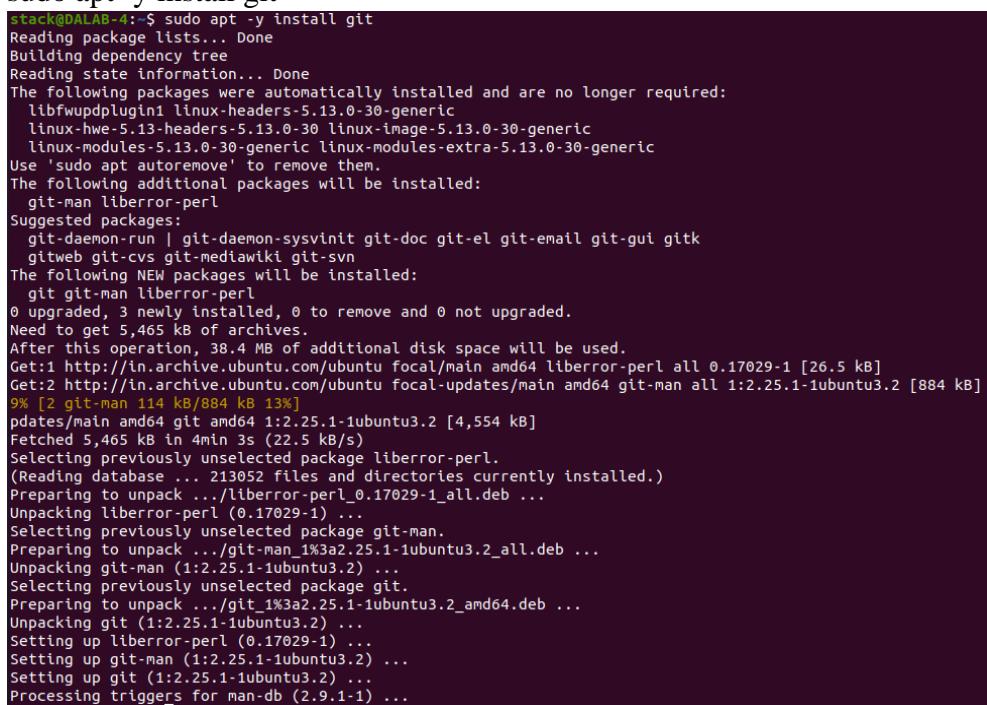


```
[itadmin@DALAB-4:~$ echo "stack ALL=(ALL) NOPASSWD: ALL" | sudo tee /etc/sudoers.d/stack
stack ALL=(ALL) NOPASSWD: ALL
[itadmin@DALAB-4:~$ ]
```

## 8. sudo su - stack

```
itadmin@DALAB-4:~$ sudo su - stack
```

## 9. sudo apt -y install git



```
stack@DALAB-4:~$ sudo apt -y install git
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libfwupdplugin1 linux-headers-5.13.0-30-generic
  linux-hwe-5.13-headers-5.13.0-30 linux-image-5.13.0-30-generic
  linux-modules-5.13.0-30-generic linux-modules-extra-5.13.0-30-generic
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  git-man liberror-perl
Suggested packages:
  git-daemon-run | git-daemon-sysvinit git-doc git-el git-email git-gui gitk
  gitweb git-cvs git-mediawiki git-svn
The following NEW packages will be installed:
  git git-man liberror-perl
0 upgraded, 3 newly installed, 0 to remove and 0 not upgraded.
Need to get 5,465 kB of archives.
After this operation, 38.4 MB of additional disk space will be used.
Get:1 http://in.archive.ubuntu.com/ubuntu focal/main amd64 liberror-perl all 0.17029-1 [26.5 kB]
Get:2 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 git-man all 1:2.25.1-1ubuntu3.2 [884 kB]
9% [2 git-man 114 kB/884 kB 13%]
Fetched 5,465 kB in 4min 3s (22.5 kB/s)
Selecting previously unselected package liberror-perl.
(Reading database ... 213052 files and directories currently installed.)
Preparing to unpack .../liberror-perl_0.17029-1_all.deb ...
Unpacking liberror-perl (0.17029-1) ...
Selecting previously unselected package git-man.
Preparing to unpack .../git-man_1%3a2.25.1-1ubuntu3.2_all.deb ...
Unpacking git-man (1:2.25.1-1ubuntu3.2) ...
Selecting previously unselected package git.
Preparing to unpack .../git_1%3a2.25.1-1ubuntu3.2_amd64.deb ...
Unpacking git (1:2.25.1-1ubuntu3.2) ...
Setting up liberror-perl (0.17029-1) ...
Setting up git-man (1:2.25.1-1ubuntu3.2) ...
Setting up git (1:2.25.1-1ubuntu3.2) ...
Processing triggers for man-db (2.9.1-1) ...
```

10. git clone <https://git.openstack.org/openstack-dev/devstack>

```
stack@DALAB-4:~$ git clone https://github.com/openstack-dev/devstack.git
Cloning into 'devstack'...
remote: Enumerating objects: 48499, done.
remote: Counting objects: 100% (1725/1725), done.
remote: Compressing objects: 100% (686/686), done.
remote: Total 48499 (delta 1177), reused 1444 (delta 1033), pack-reused 46774
Receiving objects: 100% (48499/48499), 15.48 MiB | 4.31 MiB/s, done.
Resolving deltas: 100% (33808/33808), done.
```

11. cd devstack

```
stack@DALAB-4:~$ cd devstack
stack@DALAB-4:~/devstack$
```

12. vi local.conf / nano local.conf

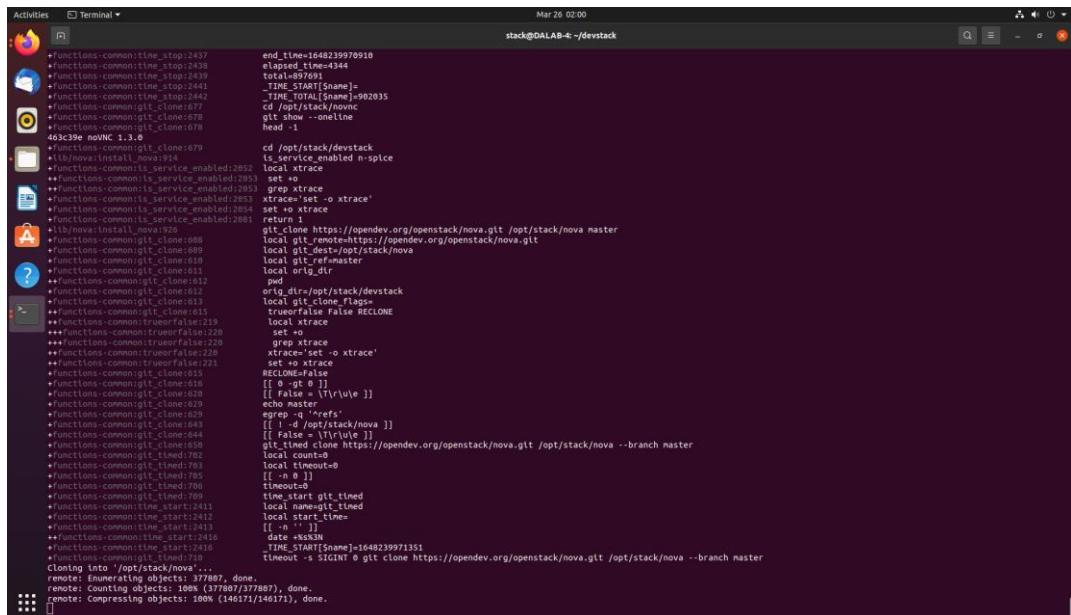
```
stack@DALAB-4:~$ cd devstack
stack@DALAB-4:~/devstack$ nano local.conf
stack@DALAB-4:~/devstack$ vi local.conf
```

13. We need to add the following in local.conf

```
[[local|localrc]]
ADMIN_PASSWORD=StrongAdminSecret
DATABASE_PASSWORD=$ADMIN_PASSWORD
RABBIT_PASSWORD=$ADMIN_PASSWORD
SERVICE_PASSWORD=$ADMIN_PASSWORD
```

14. Run this script to setup OpenStack

```
./stack.sh
```



```
Activities Terminal Mar 26 02:00 stack@DALAB-4:~/devstack
+functions-common:time_stop:2437      end_time=1648239970910
+functions-common:time_stop:2438      elapsed_time=4344
+functions-common:time_stop:2439      total_time=769
+functions-common:time_stop:2440      TIME_START[$name]=1648239970910
+functions-common:time_stop:2441      TIME_TOTAL[$name]=002035
+functions-common:git_clone:677      cd /opt/stack/novnc
+functions-common:git_clone:678      git show --oneline
+functions-common:git_clone:678      head :1
463c39e novNC 1.3.0
+functions-common:git_clone:679      cd /opt/stack/devstack
+functions-common:service_enabled:2852      local_xtrace
+functions-common:service_enabled:2853      set +o
+functions-common:xtrace
+functions-common:xtrace
+functions-common:service_enabled:2854      set +o xtrace
+functions-common:service_enabled:2854      return 1
+http:nova/install:nova:926      git clone https://opendev.org/openstack/nova.git /opt/stack/nova master
+functions-common:git_clone:688      local git_dest=/opt/stack/nova
+functions-common:git_clone:689      local git_dest=$git_dest
+functions-common:git_clone:690      local git_ref=master
+functions-common:git_clone:691      local orig_dir=$git_dest
+functions-common:git_clone:692      pushd $orig_dir>/dev/null
+functions-common:git_clone:692      orig_dir=/opt/stack/devstack
+functions-common:git_clone:693      local git_clone_flags=
+functions-common:git_clone:693      local xtrace
+functions-common:git_clone:693      local xtrace
+functions-common:git_clone:693      set +o
+functions-common:git_clone:693      grep xtrace
+functions-common:git_clone:693      xtrace_set -o xtrace
+functions-common:git_clone:693      set +o xtrace
+functions-common:git_clone:693      RECLONE=False
+functions-common:git_clone:695      [[ $git == '' ]]
+functions-common:git_clone:695      [[ $False == '' || $True == '' ]]
+functions-common:git_clone:696      echo master
+functions-common:git_clone:697      egrep -q '^ref='
+functions-common:git_clone:697      [[ $git == 'refs/heads/master' ]]
+functions-common:git_clone:697      [[ $False == '' || $True == '' ]]
+functions-common:git_clone:698      git_timed_clone https://opendev.org/openstack/nova.git /opt/stack/nova --branch master
+functions-common:git_timed:702      local count=0
+functions-common:git_timed:703      local timeout=0
+functions-common:git_timed:703      [[ -n $0 ]]
+functions-common:git_timed:706      timeout=0
+functions-common:git_timed:706      time_start=$(date)
+functions-common:git_timed:706      local git_timed
+functions-common:git_timed:706      local start_time=
+functions-common:git_start:2413      [[ -z $git ]]
+functions-common:git_start:2416      date +%SSN
+functions-common:git_start:2416      TIME_START[$name]=1648239971151
+functions-common:git_start:2416      TIMEOUT=-s SIGINT 0 git clone https://opendev.org/openstack/nova.git /opt/stack/nova --branch master
+functions-common:git_start:2417      Cloning into '/opt/stack/nova'...
+functions-common:git_start:2417      remote: Counting objects: 100% (377807/377807), done.
+functions-common:git_start:2417      remote: Compressing objects: 100% (146171/146171), done.
```

## After Installation:

```
=====
DevStack Component Timing
(times are in seconds)
=====
wait_for_service      18
pip_install          280
apt-get              942
run_process           68
dbsync                66
git_timed             1034
apt-get-update        1
test_with_retry       51
async_wait            1564
osc                   283
-----
Unaccounted time     816
=====
Total runtime         5123

=====
Async summary
=====
Time spent in the background minus waits: 2015 sec
Elapsed time: 5123 sec
Time if we did everything serially: 7138 sec
Speedup:  1.39332

This is your host IP address: 192.168.112.197
This is your host IPv6 address: ::1
Horizon is now available at http://192.168.112.197/dashboard
Keystone is serving at http://192.168.112.197/identity/
The default users are: admin and demo
The password: StrongAdminSecret

Services are running under systemd unit files.
For more information see:
https://docs.openstack.org/devstack/latest/systemd.html

DevStack Version: zed
Change: 0ed70e3f7687ffa62a8a4a38cdad14abdc8c7fa7 Merge "Update DEVSTACK_SERIES to zed" 2022-03-28 13:02:03 +0000
OS Version: Ubuntu 20.04 focal

2022-03-29 15:26:41.917 | stack.sh completed in 5123 seconds.
```

## Result:

Openstack has been successfully installed and it has been verified.

## CREATION OF VMs IN OPENSTACK

### 1)Aim:

To create a VM in openstack and execution of simple application in openstack.

### Steps to Deploy VM:

1. We need to launch openstack

The screenshot shows two stacked browser windows. The top window displays the 'Overview' section of the OpenStack dashboard. It features a 'Limit Summary' for Compute resources: Instances (Used 0 of 10), vCPUs (Used 0 of 20), RAM (Used 0 of 50GB), and Volume Storage (Used 0B of 10000GB). Below this is a 'Usage Summary' section with a date range selector from 2022-04-08 to 2022-04-09, showing Active Instances (0), Active RAM (0B), and This Period's vCPU Hours (0.00). The bottom window shows the 'Images' section, listing a single image entry: 'cirros-0.5.2-x86\_64-disk' (Type: Image, Status: Active, Visibility: Public, Protected: No, Disk Format: QCOW2, Size: 15.55 MB). There is a 'Launch' button next to the image entry.

2. After clicking launch option we will get a popup window

The screenshot shows the 'Launch Instance' dialog box. On the left, there is a sidebar with tabs: Details, Source, Flavor \*, Networks \*, Network Ports, Security Groups, Key Pair, Configuration, Server Groups, Scheduler Hints, and Metadata. The 'Details' tab is selected. The main area contains fields for Project Name (admin), Instance Name \* (Project1), Description (empty), Availability Zone (nova), and Count \* (1). To the right, there is a circular progress bar labeled 'Total Instances (10 Max)' with '10%' filled. A legend indicates '0 Current Usage', '1 Added', and '9 Remaining'. At the bottom right are 'Back', 'Next >', and a large blue 'Launch Instance' button.

3. We need to give an instance name and leave all fields as default and click next
4. We will get a source tab in launch instance window where storage is created and click next

**Launch Instance**

**Details**

**Source**

**Flavor \***

**Networks \***

**Network Ports**

**Security Groups**

**Key Pair**

**Configuration**

**Server Groups**

**Scheduler Hints**

**Metadata**

**Select Boot Source**

Image

**Create New Volume**

Yes No

**Delete Volume on Instance Delete**

Yes No

**Allocated**

Displaying 1 item

Name	Updated	Size	Format	Visibility
cirros-0.5.2-x86_64-disk	4/5/22 10:37 PM	15.55 MB	QCOW2	Public

**Available (0)**

Click here for filters or full text search.

**Displaying 0 items**

Name	Updated	Size	Format	Visibility
No items to display.				

**Displaying 0 items**

**x Cancel** **< Back** **Next >** **Launch Instance**

5. We need to allocate virtual machine resources by adding a flavor suited to our needs and click next
6. From available instance we need to add instance which we already launched by clicking upward arrow mark

**Launch Instance**

**Details**

**Source**

**Flavor \***

**Networks \***

**Network Ports**

**Security Groups**

**Key Pair**

**Configuration**

**Server Groups**

**Scheduler Hints**

**Metadata**

**Allocated**

Name VCPUS RAM Total Disk Root Disk Ephemeral Disk Public

Select an item from Available items below

**Available (12)**

Click here for filters or full text search.

Name	VCPUS	RAM	Total Disk	Root Disk	Ephemeral Disk	Public
m1.nano	1	128 MB	1 GB	1 GB	0 GB	Yes
m1.micro	1	192 MB	1 GB	1 GB	0 GB	Yes
cirros256	1	256 MB	1 GB	1 GB	0 GB	Yes
m1.tiny	1	512 MB	1 GB	1 GB	0 GB	Yes
ds512M	1	512 MB	5 GB	5 GB	0 GB	Yes
ds1G	1	1 GB	10 GB	10 GB	0 GB	Yes
m1.small	1	2 GB	20 GB	20 GB	0 GB	Yes
ds2G	2	2 GB	10 GB	10 GB	0 GB	Yes
m1.medium	2	4 GB	40 GB	40 GB	0 GB	Yes
ds4G	4	4 GB	20 GB	20 GB	0 GB	Yes
m1.large	4	8 GB	80 GB	80 GB	0 GB	Yes
m1.xlarge	8	16 GB	160 GB	160 GB	0 GB	Yes

**x Cancel** **< Back** **Next >** **Launch Instance**

**Launch Instance**

Flavors manage the sizing for the compute, memory and storage capacity of the instance.

**Allocated**

Name	VCPUS	RAM	Total Disk	Root Disk	Ephemeral Disk	Public
cirros256	1	256 MB	1 GB	1 GB	0 GB	Yes

**Available (11)**

Name	VCPUS	RAM	Total Disk	Root Disk	Ephemeral Disk	Public
m1.nano	1	128 MB	1 GB	1 GB	0 GB	Yes
m1.micro	1	192 MB	1 GB	1 GB	0 GB	Yes
m1.tiny	1	512 MB	1 GB	1 GB	0 GB	Yes
ds12M	1	512 MB	5 GB	5 GB	0 GB	Yes
ds1G	1	1 GB	10 GB	10 GB	0 GB	Yes
m1.small	1	2 GB	20 GB	20 GB	0 GB	Yes
ds2G	2	2 GB	10 GB	10 GB	0 GB	Yes
m1.medium	2	4 GB	40 GB	40 GB	0 GB	Yes
ds4G	4	4 GB	20 GB	20 GB	0 GB	Yes
m1.large	4	8 GB	80 GB	80 GB	0 GB	Yes
m1.xlarge	8	16 GB	160 GB	160 GB	0 GB	Yes

< Back | Next > | **Launch Instance**

- Finally, we add the shared network for available networks in openstack to our instance using the upward arrow mark button and hit on Launch Instance to start the virtual machine

**Launch Instance**

Networks provide the communication channels for instances in the cloud.

**Allocated (1)**

Network	Subnets Associated	Shared	Admin State	Status
shared	shared-subnet	Yes	Up	Active

**Available (1)**

Network	Subnets Associated	Shared	Admin State	Status
public	public-subnet ipv6-public-subnet	No	Up	Active

< Back | Next > | **Launch Instance**

- We click the console

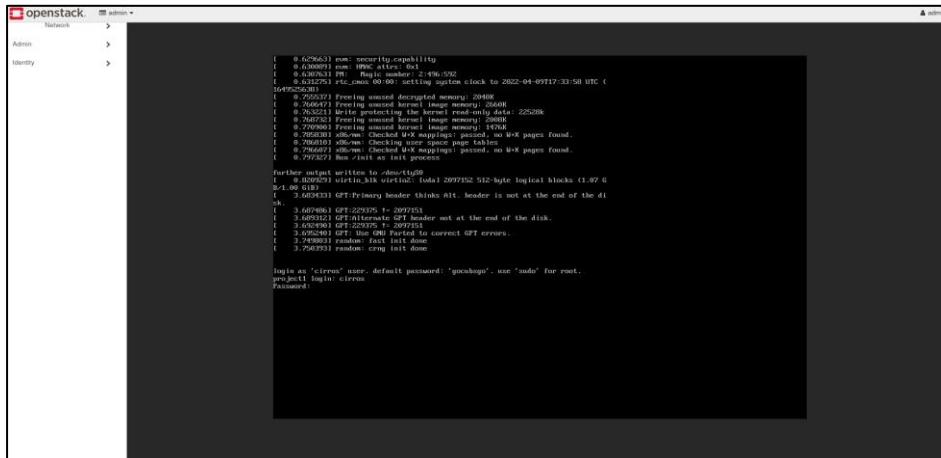
**Instances**

Instance ID	Filter	Launch Instance	Delete Instance	More Actions
orm028	-			

**Actions**

- Associate Floating IP
- Attach Interface
- Detach Interface
- Edit Name
- Attach Volume
- Detach Volume
- Update Metadata
- Edit Security Groups
- Delete Security Groups
- Console
- View Log
- Power Instance
- Power Off
- Suspend Instance
- Shelve Instance
- Resave Instance
- Delete Instance
- Soft Reboot Instance
- Hard Reboot Instance
- Shut Off Instance
- Relaunch Instance

9. In console window , we create a text file. Eg. vi text.txt .



The screenshot shows a terminal window titled "openstack" with the user "admin". The terminal displays a detailed boot log from a kernel. Key entries include:

- [ 0.629653] e8p: security\_capability
- [ 0.630025] PM: Magic number: 2496/592
- [ 0.630251] rtc\_cmos 00:00: setting system clock to 2022-04-09T17:33:58 UTC (1676256337)
- [ 0.752337] Freeing unused descriptor arrays: 2048
- [ 0.752337] [mem=256M-260M] Freeing unneeded kernel memory: 2560K
- [ 0.765221] Write protecting the kernel read-only data: 22528K
- [ 0.770903] Freeing unused kernel image memory: 3748K
- [ 0.770903] Freeing unused kernel image memory: 3748K
- [ 0.786103] x86/mm: Checking over space page tables, found 4048 pages.
- [ 0.797371] Ram: 2147 as init process

Further output shows disk geometry and GPT headers:

- [ 1.08 418] 21.08 GiB
- [ 3.605933] GPT:Primary header thinks Alt. header is not at the end of the disk
- [ 3.605940] GPT:229375 + 299751
- [ 3.605940] GPT:Primary header is not at the end of the disk.
- [ 3.605940] GPT:229375 + 299751
- [ 3.605940] GPT:Primary header is not at the end of the disk. current GPT errors.
- [ 3.749883] random: fast init done
- [ 3.750593] random: crng init done

Login information follows:

```
login as 'cirrus' user. default password: 'yourlogo'. use 'sudo' for root.
project login: cirrus
Password:
```

We display the contents using cat text.txt

```
$ ls
test.txt
$ cat test.txt
Hello world!!
$ -
```

We can also type ifconfig and execute the command

```
$ ifconfig
eth0      Link encap:Ethernet HWaddr FA:16:3E:03:30:1A
          inet addr:192.168.233.104 Bcast:192.168.233.255 Mask:255.255.255.0
          inet6 addr: fe80::f816:3eff:fe03:301a/64 Scope:Link
             UP BROADCAST RUNNING MULTICAST MTU:1442 Metric:1
             RX packets:85 errors:0 dropped:0 overruns:0 frame:0
             TX packets:124 errors:0 dropped:0 overruns:0 carrier:0
             collisions:0 txqueuelen:1000
             RX bytes:11369 (11.1 KiB) TX bytes:10674 (10.4 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
             UP LOOPBACK RUNNING MTU:65536 Metric:1
             RX packets:28 errors:0 dropped:0 overruns:0 frame:0
             TX packets:28 errors:0 dropped:0 overruns:0 carrier:0
             collisions:0 txqueuelen:1000
             RX bytes:2532 (2.4 KiB) TX bytes:2532 (2.4 KiB)

$ -
```

## Result:

Creation of VM in openstack has been done successfully and execution of simple application using Open Stack has also been done successfully and the results are verified.

## **MONGODB BASICS**

### **1)Aim:**

To do a study on MongoDB.

### **MongoDB:**

MongoDB is an open-source document database. It is a NOSQL database. MongoDB is written in C++. It is a highly scalable and performance-oriented database.

### **Features:**

#### **A) Documents**

MongoDB stores data as JSON documents. The document data model maps naturally to objects in application code, making it simple for developers to learn and use. Documents can be nested to express hierarchical relationships and store structures as arrays.

#### **B) Collections**

In MongoDB, a collection is a group of documents. Each collection is associated with one MongoDB database.

#### **C) Replica Sets**

When we create a database in MongoDB, the system automatically creates at least two or more copies of data referred as a replica set. It ensures high availability of data, offers redundancy and protection against downtime in face of system failure / maintenance.

#### **D) Sharding**

MongoDB shards data at collection level, distributing documents in a collection across the shards in a cluster. It is useful to handle massive data growth scaling.

#### **E) Indexes**

Indexes support the efficient execution of queries. MongoDB offers variety of different indexing strategies. Indexes speed up queries because queries scan the index instead of reading every document in the collection.

### **Result:**

Study on MongoDB has been done successfully.

## **2)Aim:**

To create a database and corresponding tables, populate the tables with data and apply insert /search operation

### **Code:**

show dbs

### **Output:**

```
> show dbs
admin    0.000GB
config   0.000GB
local    0.000GB
```

use sample

### **Output:**

```
samlpe> use sample
switched to db sample
```

show tables

### **Output:**

```
sample> show tables
faculty
student
subject
```

### **Insertion of data:**

```
db.student.insertMany([
  {
    "name": "rozen",
    "reg": "1",
    "cgpa": "9.6"
  },
  {
    "name": "raj",
    "reg": "2",
  }
])
```

```
        "cgpa":"9.2"
    },
{
    "name" : "raju",
    "reg":"3",
    "cgpa":"9.2"
}
])
```

### Output:

```
{  
    "acknowledged" : true,  
    "insertedIds" : [  
        ObjectId("6255913b36edfee473189a04"),  
        ObjectId("6255913b36edfee473189a05"),  
        ObjectId("6255913b36edfee473189a06")  
    ]  
}
```

```
db.subject.insertMany(  
[  
    {  
        "name":"maths",  
        "code":"MA5501"  
    },  
    {  
        "name" : "chemistry",  
        "code":"CH5801"  
    },  
    {  
        "name" : "physics",  
        "code":"PH8801"  
    }  
])  
db.student.find().pretty()
```

## **Output:**

```
sample> db.student.find().pretty()
[
  {
    _id: ObjectId("6255d59d6f49f81bc99a1856"),
    name: 'rozen',
    reg: '1',
    cgpa: '9.6'
  },
  {
    _id: ObjectId("6255d59d6f49f81bc99a1857"),
    name: 'raj',
    reg: '2',
    cgpa: '9.2'
  },
  {
    _id: ObjectId("6255d59d6f49f81bc99a1858"),
    name: 'raju',
    reg: '3',
    cgpa: '9.2'
  }
]
sample> █
```

```
db.faculty.insertMany(
[
  {
    name: "Dr.A",
    subject: 'maths'
  },
  {
    name: "Dr.B",
    subject: 'chemistry'
  },
  {
    name: "Dr.C",
    subject: 'physics'
  }
])
```

```
db.faculty.find().pretty()
```

**Output:**

```
sample> db.faculty.find().pretty()
[
  {
    _id: ObjectId("6255d93b6f49f81bc99a185c"),
    name: 'Dr.D',
    subject: 'maths'
  },
  {
    _id: ObjectId("6255d93b6f49f81bc99a185d"),
    name: 'Dr.B',
    subject: 'chemistry'
  },
  {
    _id: ObjectId("6255d93b6f49f81bc99a185e"),
    name: 'Dr.C',
    subject: 'physics'
  }
]
sample> ■
```

```
db.subject.insertMany(
[
  {
    "name": "dsa",
    "code": "IT5603"
  },
  {
    "name": "cloud",
    "code": "IT5602"
  },
  {
    "name": "embedded",
    "code": "IT5601"
  }
])
```

```
db.subject.find().pretty()
```

**Output:**

```
sample> db.subject.find().pretty()
[
  {
    _id: ObjectId("6255d68f6f49f81bc99a1859"),
    name: 'maths',
    code: 'MA5501'
  },
  {
    _id: ObjectId("6255d68f6f49f81bc99a185a"),
    name: 'chemistry',
    code: 'CH5801'
  },
  {
    _id: ObjectId("6255d68f6f49f81bc99a185b"),
    name: 'physics',
    code: 'PH8801'
  },
  {
    _id: ObjectId("6255dd376f49f81bc99a185f"),
    name: 'dsa',
    code: 'IT5603'
  },
  {
    _id: ObjectId("6255dd376f49f81bc99a1860"),
    name: 'cloud',
    code: 'IT5602'
  },
  {
    _id: ObjectId("6255dd376f49f81bc99a1861"),
    name: 'embedded',
    code: 'IT5601'
  }
]
sample> █
```

use shopping

```
db.products.insertOne({"name":"abc","price":30,"color":"red"})
```

**Output:**

```
{
  "acknowledged" : true,
  "insertedId" : ObjectId("5c766be6d27187cc70d7666f")
}
```

```
db.products.find()
```

**Output:**

```
> db.products.find().pretty();
{
  "_id" : ObjectId("5c766be6d27187cc70d7666f"),
  "name" : "abc",
  "price" : 30,
  "color" : "red"
}
```

```
db.products.insertMany([{"name":"mno","price":50}, {"name":"pqr","price":100,"qty":1}]);
```

**Output:**

```
> db.products.insertMany([{"name": "mno", "price": 50}, {"name": "pqr", "price": 100, "qty": 1}]);  
{  
    "acknowledged" : true,  
    "insertedIds" : [  
        ObjectId("5c767a06136fb7ca8026002e"),  
        ObjectId("5c767a06136fb7ca8026002f")  
    ]  
}
```

```
db.products.find().pretty()
```

**Output:**

```
> db.products.find().pretty();  
{  
    "_id" : ObjectId("5c766be6d27187cc70d7666f"),  
    "name" : "abc",  
    "price" : 30,  
    "color" : "red"  
}  
{  
    "_id" : ObjectId("5c767a06136fb7ca8026002e"),  
    "name" : "mno",  
    "price" : 50  
}  
{  
    "_id" : ObjectId("5c767a06136fb7ca8026002f"),  
    "name" : "pqr",  
    "price" : 100,  
    "qty" : 1  
}
```

```
db.products.insertOne({"name":"abc",_id:"p01"})
```

```
db.products.updateOne({name:'mno'},{$set:{price:100,qty:10}})
```

**Output:**

```
> db.products.updateOne({name: "mno"}, {$set: {price: 100, qty: 10}});  
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
```

```
db.products.updateMany({name:'abc'},{$set:{ qty:150}})
```

**Output:**

```
> db.products.updateMany({name: "abc"}, {$set: {qty: 150}});  
{ "acknowledged" : true, "matchedCount" : 2, "modifiedCount" : 2 }
```

```
db.products.deleteOne({ _id:'p01'});
```

**Output:**

```
{ "acknowledged" : true, "deletedCount" : 1 }
```

**Result:**

Database has been created, populated with data , insert/search operations are performed successfully and the results are verified.

## BIVARIATE ANALYSIS

### 1)Aim:

To perform the following from the dow jones index dataset:

- a)Pearson Correlation Coefficients and Interpretation b)Simple Linear Regression
- c)Chi-square test d)T-test e)Analysis of Variance f)Scatterplots

### Code:

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

```
data['high'] = data['high'].str.replace('$', '')
data['open'] = data['open'].str.replace('$', '')
data['low'] = data['low'].str.replace('$', '')
data['close'] = data['close'].str.replace('$', '')
```

```
data['next_weeks_open'] = data['next_weeks_open'].str.replace('$', '')
data['next_weeks_close'] = data['next_weeks_close'].str.replace('$', '')
```

```
data['high']=pd.to_numeric(data['high'])
data['open']=pd.to_numeric(data['open'])
data['low']=pd.to_numeric(data['low'])
data['close']=pd.to_numeric(data['close'])
data['next_weeks_open']=pd.to_numeric(data['next_weeks_open'])
data['next_weeks_close']=pd.to_numeric(data['next_weeks_close'])
```

```
data['percent_change_volume_over_last_wk'].fillna(value=0,inplace=True)
data['previous_weeks_volume'].fillna(value=0,inplace=True)
```

data

**Output:**

quarter	stock	date	open	high	low	close	volume	percent_change_price	percent_change_volume_over_last_wk	previous_weeks_volume	rank
0	1	AA	1/7/2011	15.82	16.72	15.78	16.42	239655616	3.79267	0.000000	0.0
1	1	AA	1/14/2011	16.71	16.71	15.64	15.97	242963398	-4.42849	1.380223	239655616.0
2	1	AA	1/21/2011	16.19	16.38	15.60	15.79	138428495	-2.47066	-43.024959	242963398.0
3	1	AA	1/28/2011	15.87	16.63	15.82	16.13	151379173	1.63831	9.355500	138428495.0
4	1	AA	2/4/2011	16.18	17.39	16.18	17.14	154387761	5.93325	1.987452	151379173.0
...	...	...	...	...	...	...	...	...	...	...	...
745	2	XOM	5/27/2011	80.22	82.63	80.07	82.63	68230855	3.00424	-21.355713	86758820.0
746	2	XOM	6/3/2011	83.28	83.75	80.18	81.18	78616295	-2.52161	15.221032	68230855.0
747	2	XOM	6/10/2011	80.93	81.87	79.72	79.78	92380844	-1.42098	17.508519	78616295.0
748	2	XOM	6/17/2011	80.00	80.82	78.33	79.02	100521400	-1.22500	8.811952	92380844.0
749	2	XOM	6/24/2011	78.65	81.12	76.78	76.78	118679791	-2.37762	18.064204	100521400.0

750 rows × 16 columns

**#Pearson Correlation**

```
import scipy.stats  
  
corr,_=scipy.stats.pearsonr(data['high'],data['low'])  
  
print(corr)
```

**Output:**

```
0.9994191540117697
```

```
corr,_=scipy.stats.pearsonr(data['volume'],data['open'])  
  
print(corr)
```

**Output:**

```
-0.5134239596601641
```

```
corr,_=scipy.stats.pearsonr(data['days_to_next_dividend'],data['next_weeks_close'])  
  
print(corr)
```

**Output:**

```
-0.06772514422301334
```

```
corr,_=scipy.stats.pearsonr(data['open'],data['close'])  
  
print(corr)
```

**Output:**

```
0.999044034320146
```

```
corr,_=scipy.stats.pearsonr(data['percent_change_volume_over_last_wk'],data['percent_change_price'])
```

```
print(corr)
```

**Output:**

```
-0.22956755048417227
```

```
corr,_=scipy.stats.pearsonr(data['next_weeks_open'],data['next_weeks_close'])
```

```
print(corr)
```

**Output:**

```
0.9988798616255573
```

```
corr,_=scipy.stats.pearsonr(data['days_to_next_dividend'],data['percent_return_next_dividend'])
```

```
print(corr)
```

**Output:**

```
0.12174809745628305
```

```
from matplotlib.pyplot import figure
```

```
figure(figsize=(18, 16))
```

```
sns.heatmap(data.corr(),annot=True)
```

```
plt.plot()
```

**Output:**



```

#Simple Linear Regression

def estimate_coef(x, y):

    # number of observations/points
    n = np.size(x)

    # mean of x and y vector
    m_x = np.mean(x)
    m_y = np.mean(y)

    # calculating cross-deviation and deviation about x
    SS_xy = np.sum(y*x) - n*m_y*m_x
    SS_xx = np.sum(x*x) - n*m_x*m_x

    # calculating regression coefficients
    b_1 = SS_xy / SS_xx
    b_0 = m_y - b_1*m_x

    return (b_0, b_1)

def plot_regression_line(x, y, b):

    # plotting the actual points as scatter plot
    plt.scatter(x, y, color = "m",
                marker = "o", s = 30)

    # predicted response vector
    y_pred = b[0] + b[1]*x

    # plotting the regression line
    plt.plot(x, y_pred, color = "g")

    # putting labels
    plt.xlabel('x')
    plt.ylabel('y')

    # function to show plot
    plt.show()

def main():

    # observations / data

```

```

x = np.array(data['high'])
y = np.array(data['low'])

# estimating coefficients
b = estimate_coef(x, y)

print("Estimated coefficients:\nc = {} \nm = {}".format(b[0], b[1]))

# plotting regression line
plot_regression_line(x, y, b)

```

```
if __name__ == "__main__":
    main()
```

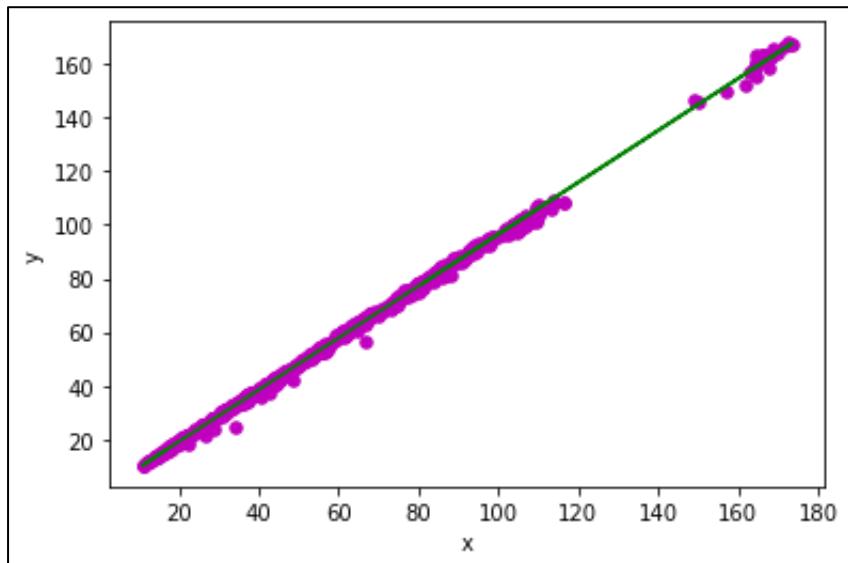
**Output:**

```

Estimated coefficients:
c = -0.19404242022824292
m = 0.9664206201909

```

#scatter plot



#chi square test

```

from scipy.stats import chi2_contingency

# defining the table
data_c = [data['low'], data['open'], data['high']]
stat, p, dof, expected = chi2_contingency(data_c)

```

```

# interpret p-value
alpha = 0.05
print("p value is " + str(p))
if p <= alpha:
    print('Dependent (reject H0)')
else:
    print('Independent (H0 holds true)')

```

**Output:**

```

p value is 1.0
Independent (H0 holds true)

```

**#t test**

```

from scipy.stats import ttest_ind
# defining the table
stat, p = ttest_ind(data['low'],data['open'])
# interpret p-value
alpha = 0.05
print("p value is " + str(p))
if p <= alpha:
    print('Dependent (reject H0)')
else:
    print('Independent (H0 holds true)')

```

**Output:**

```

p value is 0.5452473592620216
Independent (H0 holds true)

```

```

from scipy.stats import ttest_ind
# defining the table
stat, p = ttest_ind(data['high'],data['close'])

```

```
# interpret p-value  
alpha = 0.05  
print("p value is " + str(p))  
if p <= alpha:  
    print('Dependent (reject H0)')  
else:  
    print('Independent (H0 holds true)')
```

**Output:**

```
p value is 0.581045977109643  
Independent (H0 holds true)
```

**#Analysis of Variance**

```
import scipy.stats as st  
st.f_oneway(data['high'],data['low'],data['open'])
```

**Output:**

```
F_onewayResult(statistic=0.7242002902172524, pvalue=0.4848251428613093)
```

**Result:**

Operations related to bivariate analysis has been successfully executed and the results are verified.

## LOGISTIC REGRESSION

### **1)Aim:**

To perform the following from the dataset i)Binary ii)Multi-class or Multinominal  
iii)Ordinal

### **Code:**

#### **i)Binary**

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

```
data=pd.read_csv("diabetes.csv")
data.info()
```

### **Output:**

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Pregnancies      768 non-null    int64  
 1   Glucose          768 non-null    int64  
 2   BloodPressure    768 non-null    int64  
 3   SkinThickness    768 non-null    int64  
 4   Insulin          768 non-null    int64  
 5   BMI              768 non-null    float64 
 6   DiabetesPedigreeFunction 768 non-null    float64 
 7   Age              768 non-null    int64  
 8   Outcome          768 non-null    int64  
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

```
from sklearn.model_selection import train_test_split
train,test=train_test_split(data,test_size=0.20,random_state=42)
```

```
import statsmodels.api as sm
from statsmodels.formula.api import logit
formula=('Outcome ~ Pregnancies + Glucose + BloodPressure + SkinThickness + Insulin +
BMI + DiabetesPedigreeFunction + Age')
model=logit(formula=formula,data=train).fit()
```

**Output:**

```
Optimization terminated successfully.  
Current function value: 0.467835  
Iterations 6
```

model.summary()

**Output:**

Logit Regression Results						
<b>Dep. Variable:</b>	Outcome	<b>No. Observations:</b>	614			
<b>Model:</b>	Logit	<b>Df Residuals:</b>	605			
<b>Method:</b>	MLE	<b>Df Model:</b>	8			
<b>Date:</b>	Sun, 15 May 2022	<b>Pseudo R-squ.:</b>	0.2752			
<b>Time:</b>	21:05:52	<b>Log-Likelihood:</b>	-287.25			
<b>converged:</b>	True	<b>LL-Null:</b>	-396.34			
<b>Covariance Type:</b>	nonrobust	<b>LLR p-value:</b>	9.311e-43			

	coef	std err	z	P> z	[0.025	0.975]
<b>Intercept</b>	-9.0359	0.837	-10.802	0.000	-10.675	-7.396
<b>Pregnancies</b>	0.0645	0.036	1.791	0.073	-0.006	0.135
<b>Glucose</b>	0.0341	0.004	8.055	0.000	0.026	0.042
<b>BloodPressure</b>	-0.0139	0.006	-2.260	0.024	-0.026	-0.002
<b>SkinThickness</b>	0.0031	0.008	0.397	0.691	-0.012	0.019
<b>Insulin</b>	-0.0018	0.001	-1.782	0.075	-0.004	0.000
<b>BMI</b>	0.1026	0.017	5.948	0.000	0.069	0.136
<b>DiabetesPedigreeFunction</b>	0.6945	0.330	2.107	0.035	0.049	1.341
<b>Age</b>	0.0371	0.011	3.400	0.001	0.016	0.058

AME=model.get\_margeff(at='overall',method='dydx')

AME.summary()

## Output:

Logit Marginal Effects

Dep. Variable: Outcome

Method: dydx

At: overall

	dy/dx	std err	z	P> z	[0.025]	0.975]
Pregnancies	0.0098	0.005	1.807	0.071	-0.001	0.021
Glucose	0.0052	0.001	10.107	0.000	0.004	0.006
BloodPressure	-0.0021	0.001	-2.291	0.022	-0.004	-0.000
SkinThickness	0.0005	0.001	0.398	0.691	-0.002	0.003
Insulin	-0.0003	0.000	-1.797	0.072	-0.001	2.51e-05
BMI	0.0156	0.002	6.584	0.000	0.011	0.020
DiabetesPedigreeFunction	0.1059	0.050	2.132	0.033	0.009	0.203
Age	0.0057	0.002	3.514	0.000	0.003	0.009

```
import numpy as np
from sklearn.metrics import classification_report,accuracy_score,confusion_matrix
prediction = model.predict(exog=test)
cutoff=0.5
y_pred=np.where(prediction>cutoff ,1,0)
y_actual=test['Outcome']
conf_matrix=pd.crosstab(y_actual,y_pred,rownames=['Actual'],colnames=['Predicted'],margins=True)
conf_matrix
```

## Output:

Predicted	0	1	All
Actual			
0	79	20	99
1	18	37	55
All	97	57	154

```
ax = sns.heatmap(confusion_matrix(y_actual,y_pred), annot=True, cmap='Blues')
```

```

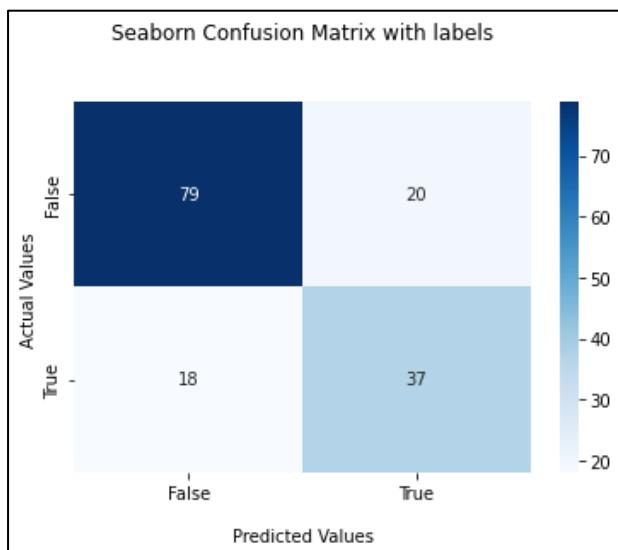
ax.set_title('Seaborn Confusion Matrix with labels\n\n');
ax.set_xlabel("\nPredicted Values")
ax.set_ylabel('Actual Values ');

## Ticket labels - List must be in alphabetical order
ax.xaxis.set_ticklabels(['False','True'])
ax.yaxis.set_ticklabels(['False','True'])

## Display the visualization of the Confusion Matrix.
plt.show()

```

**Output:**



```
acc=accuracy_score(y_actual,y_pred)
```

```
acc
```

**Output:**

acc
0.7532467532467533

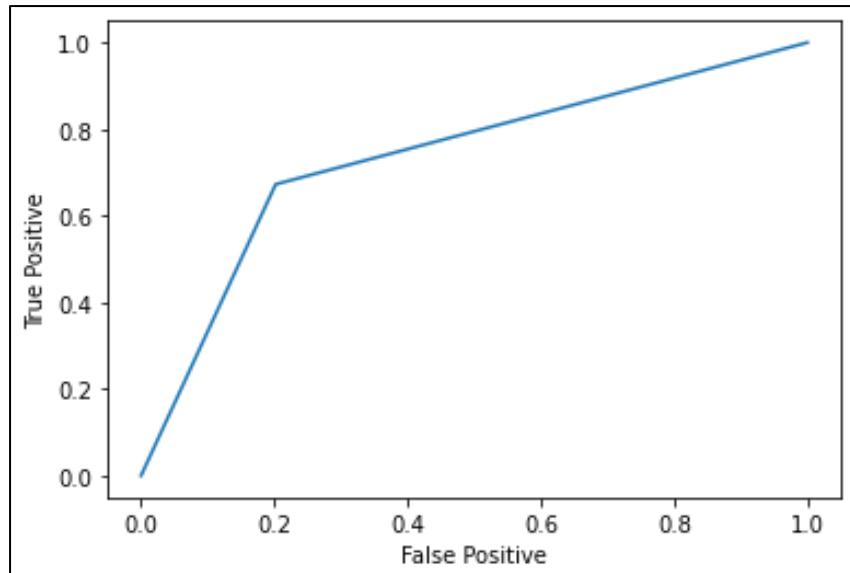
```
print(classification_report(y_actual,y_pred))
```

**Output:**

	precision	recall	f1-score	support
0	0.81	0.80	0.81	99
1	0.65	0.67	0.66	55
accuracy			0.75	154
macro avg	0.73	0.74	0.73	154
weighted avg	0.76	0.75	0.75	154

```
from sklearn.metrics import roc_curve  
fpr,tpr,_=roc_curve(y_actual,y_pred)  
plt.plot(fpr,tpr)  
plt.xlabel('False Positive')  
plt.ylabel('True Positive')  
plt.show()
```

**Output:**



**ii) Multiclass or Multinomial**

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

```
data = pd.read_csv("dow_jones_index.csv")
```

```
data.corr()
```

### Output:

	quarter	open	high	low	close	volume	percent_change_price
quarter	1.000000	0.025883	0.024727	0.024730	0.022102	-0.047599	-0.080959
open	0.025883	1.000000	0.999607	0.999344	0.999044	-0.513424	0.061485
high	0.024727	0.999607	1.000000	0.999419	0.999558	-0.512240	0.078656
low	0.024730	0.999344	0.999419	1.000000	0.999549	-0.514850	0.083170
close	0.022102	0.999044	0.999558	0.999549	1.000000	-0.514545	0.097980
volume	-0.047599	-0.513424	-0.512240	-0.514850	-0.514545	1.000000	-0.138924
percent_change_price	-0.080959	0.061485	0.078656	0.083170	0.097980	-0.138924	1.000000
percent_change_volume_over_last_wk	0.028977	-0.007966	-0.006020	-0.016874	-0.015809	0.169133	-0.229568
previous_weeks_volume	-0.024383	-0.495972	-0.495702	-0.495703	-0.495598	0.817078	-0.071942
next_weeks_open	0.021018	0.998942	0.999480	0.999430	0.999916	-0.514324	0.098270
next_weeks_close	0.019103	0.997787	0.998500	0.998326	0.998854	-0.513456	0.099243
percent_change_next_weeks_price	-0.017460	0.066603	0.069474	0.067320	0.067664	-0.056389	0.019955
days_to_next_dividend	-0.025641	-0.065228	-0.065516	-0.066270	-0.065975	-0.055486	0.006217
percent_return_next_dividend	-0.008505	-0.146001	-0.148343	-0.143838	-0.146417	-0.269735	0.029925

```
cols=['open','high','low','close','next_weeks_open','next_weeks_close']
```

```
X = data[cols]
```

```
Y = data['stock']
```

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 0.2, random_state=9)
```

```
from sklearn.linear_model import LogisticRegression
```

```
model=LogisticRegression(multi_class='multinomial', solver='lbfgs', penalty='none')
```

```
model.fit(X_train,y_train)
```

### Output:

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
```

```
pred=model.predict(X_test)
```

```
from sklearn.metrics import classification_report , accuracy_score
```

```
acc=accuracy_score(y_test,pred)
```

```
print(acc)
```

### Output:

```
acc=accuracy_score(y_test,pred)  
print(acc)
```

```
0.1
```

```
print(classification_report(y_test,pred))
```

**Output:**

	precision	recall	f1-score	support
AA	0.00	0.00	0.00	5
AXP	0.17	0.33	0.22	3
BA	0.00	0.00	0.00	5
BAC	0.33	0.20	0.25	10
CAT	0.33	0.17	0.22	6
CSCO	0.00	0.00	0.00	8
CVX	0.00	0.00	0.00	4
DD	0.50	0.12	0.20	8
DIS	0.00	0.00	0.00	5
GE	0.18	0.67	0.29	3
HD	0.00	0.00	0.00	7
HPQ	0.00	0.00	0.00	2
IBM	0.50	1.00	0.67	4
INTC	0.00	0.00	0.00	6
JNJ	0.00	0.00	0.00	8
JPM	0.00	0.00	0.00	4
KO	0.00	0.00	0.00	5

**iii)Ordinal**

```
import pandas as pd  
data=pd.read_csv('diamonds.csv')  
data
```

**Output:**

	Unnamed: 0	carat	cut	color	clarity	depth	table	price	x	y	z
0	1	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43
1	2	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31
2	3	0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31
3	4	0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63
4	5	0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75
...	...	...	...	...	...	...	...	...	...	...	...
53935	53936	0.72	Ideal	D	SI1	60.8	57.0	2757	5.75	5.76	3.50
53936	53937	0.72	Good	D	SI1	63.1	55.0	2757	5.69	5.75	3.61
53937	53938	0.70	Very Good	D	SI1	62.8	60.0	2757	5.66	5.68	3.56
53938	53939	0.86	Premium	H	SI2	61.0	58.0	2757	6.15	6.12	3.74
53939	53940	0.75	Ideal	D	SI2	62.2	55.0	2757	5.83	5.87	3.64

```
data.dtypes
```

**Output:**

```
    Unnamed: 0      int64
  carat           float64
    cut            object
  color            object
clarity           object
  depth           float64
  table           float64
  price          int64
    x             float64
    y             float64
    z             float64
  dtype: object
```

```
data.info()
```

**Output:**

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 53940 entries, 0 to 53939
Data columns (total 11 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   Unnamed: 0   53940 non-null   int64  
 1   carat       53940 non-null   float64 
 2   cut          53940 non-null   object  
 3   color        53940 non-null   object  
 4   clarity      53940 non-null   object  
 5   depth        53940 non-null   float64 
 6   table        53940 non-null   float64 
 7   price        53940 non-null   int64  
 8   x            53940 non-null   float64 
 9   y            53940 non-null   float64 
 10  z            53940 non-null   float64 
 dtypes: float64(6), int64(2), object(3)
 memory usage: 4.5+ MB
```

```
from pandas.api.types import CategoricalDtype
```

```
cat_type = CategoricalDtype(categories=['Fair', 'Good', 'Ideal', 'Very Good', 'Premium'],
ordered=True)
```

```
data["cut"] = data["cut"].astype(cat_type)
```

```
data['cut'].dtype
```

## **Output:**

```
data['cut'].dtype
CategoricalDtype(categories=['Fair', 'Good', 'Ideal', 'Very Good', 'Premium'], ordered=True)
```

```
data['volume'] = data['x']*data['y']*data['z']
```

```
data.drop(['x','y','z'],axis=1,inplace=True)
```

```
import matplotlib.pyplot as plt
```

```
plt.figure(figsize=[24,24])
```

```
plt.subplot(221)
```

```
plt.hist(data['carat'],bins=20,color='b')
```

```
plt.xlabel('Weight')
```

```
plt.title('Distribution by Weight')
```

```
plt.subplot(222)
```

```
plt.hist(data['depth'],bins=20,color='r')
```

```
plt.xlabel('Diamond Depth')
```

```
plt.title('Distribution by Depth')
```

```
plt.subplot(223)
```

```
plt.hist(data['price'],bins=20,color='g')
```

```
plt.xlabel('Price')
```

```
plt.title('Distribution by Price')
```

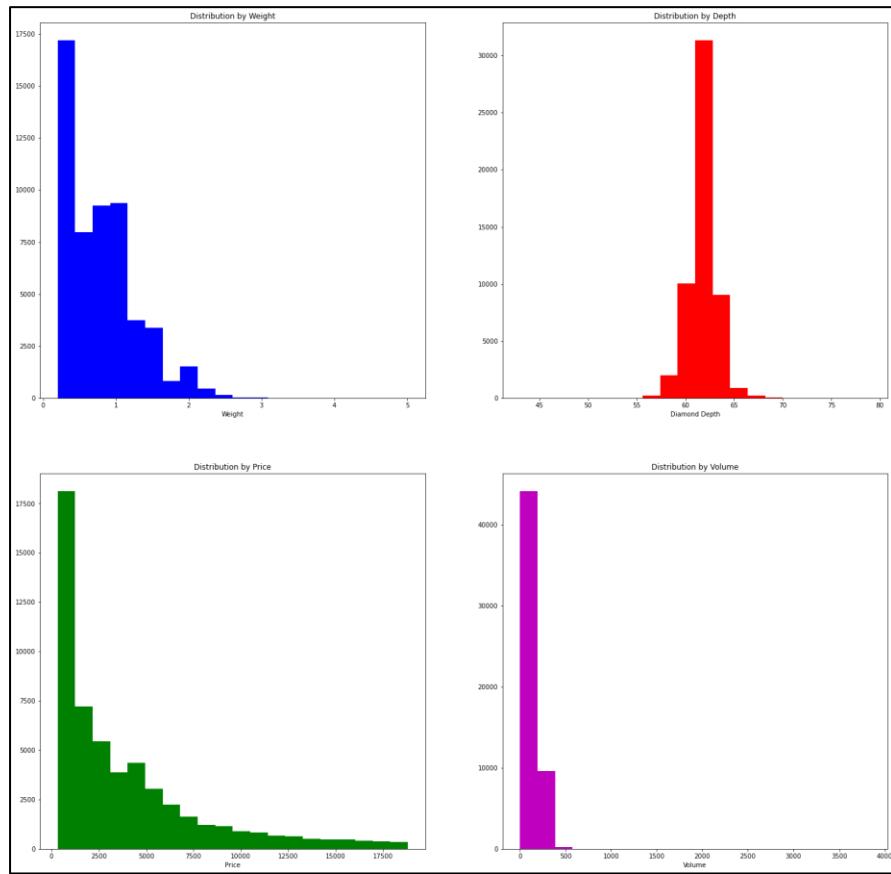
```
plt.subplot(224)
```

```
plt.hist(data['volume'],bins=20,color='m')
```

```
plt.xlabel('Volume')
```

```
plt.title('Distribution by Volume')
```

## Output:



```
from statsmodels.miscmodels.ordinal_model import OrderedModel  
mod_prob = OrderedModel(data['cut'], data[['volume', 'price', 'carat']], distr='probit')  
  
res_prob = mod_prob.fit(method='bfgs')  
res_prob.summary()
```

## Output:

```
Optimization terminated successfully.  
Current function value: 1.369122  
Iterations: 29  
Function evaluations: 35  
Gradient evaluations: 35
```

OrderedModel Results

Dep. Variable:	cut	Log-Likelihood:	-73850.
Model:	OrderedModel	AIC:	1.477e+05
Method:	Maximum Likelihood	BIC:	1.478e+05
Date:	Sun, 15 May 2022		
Time:	23:59:16		
No. Observations:	53940		
Df Residuals:	53933		
Df Model:	7		

### #ordered logit

```
mod_prob = OrderedModel(data['cut'],data[['volume', 'price', 'carat']],distr='logit')
res_log = mod_prob.fit(method='bfgs')
res_log.summary()
```

#### Output:

```
Optimization terminated successfully.
    Current function value: 1.369215
    Iterations: 36
    Function evaluations: 42
    Gradient evaluations: 42
```

OrderedModel Results

Dep. Variable:	cut	Log-Likelihood:	-73855.
Model:	OrderedModel	AIC:	1.477e+05
Method:	Maximum Likelihood	BIC:	1.478e+05
Date:	Mon, 16 May 2022		
Time:	00:01:08		
No. Observations:	53940		
Df Residuals:	53933		
Df Model:	7		

#### Result:

Various types of logistic regression models have been tried successfully and the results are verified.

## **2)Aim:**

To build a logistic regression model from the unique dataset and perform the following:

a.Data Understanding, Data cleaning, Exploratory Data Analysis-Relationship between target variable and other variables, Feature Engineering, Feature Normalization, Build and Train the model, Prediction, Perform Evaluation using standard metrics.

b.Build Model by Optimizing the Hyperparameters, Evaluate the model.

## **Code:**

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt # data visualization
import seaborn as sns # statistical data visualization

import warnings
warnings.filterwarnings('ignore')
df=pd.read_csv('weatherAUS.csv')
col_names = df.columns
col_names
```

## **Output:**

```
Index(['Date', 'Location', 'MinTemp', 'MaxTemp', 'Rainfall', 'Evaporation',
       'Sunshine', 'WindGustDir', 'WindGustSpeed', 'WindDir9am', 'WindDir3pm',
       'WindSpeed9am', 'WindSpeed3pm', 'Humidity9am', 'Humidity3pm',
       'Pressure9am', 'Pressure3pm', 'Cloud9am', 'Cloud3pm', 'Temp9am',
       'Temp3pm', 'RainToday', 'RISK_MM', 'RainTomorrow'],
      dtype='object')
```

```
df.drop('RISK_MM',axis=1,inplace=True)
categorical = [var for var in df.columns if df[var].dtype=='O']
print('There are {} categorical variables\n'.format(len(categorical)))
print('The categorical variables are :', categorical)
```

## **Output:**

```
There are 7 categorical variables
The categorical variables are : ['Date', 'Location', 'WindGustDir', 'WindDir9am', 'WindDir3pm', 'Rain
Today', 'RainTomorrow']
```

```
# parse the dates, currently coded as strings, into datetime format
```

```
df['Date'] = pd.to_datetime(df['Date'])
```

```
# drop the original Date variable
```

```
df.drop('Date', axis=1, inplace = True)
```

```
# get k-1 dummy variables after One Hot Encoding
```

```
# preview the dataset with head() method
```

```
pd.get_dummies(df.Location, drop_first=True).head()
```

```
# preview the dataset with head() method
```

### **Output:**

	Albany	Albury	AliceSprings	BadgerysCreek	Ballarat	Bendigo	Brisbane	Cairns	Canberra	Cobar	
0	0	0	1	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0	0	0
3	0	0	1	0	0	0	0	0	0	0	0
4	0	0	1	0	0	0	0	0	0	0	0

5 rows × 48 columns

```
pd.get_dummies(df.WindGustDir, drop_first=True, dummy_na=True).head()
```

### **Output:**

	ENE	ESE	N	NE	NNE	NNW	NW	S	SE	SSE	SSW	SW	W	WNW	WSW	NaN
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
3	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

```
df[numerical].isnull().sum()
```

```
plt.figure(figsize=(15,10))
```

```
plt.subplot(2, 2, 1)
```

```
fig = df.boxplot(column='Rainfall')
```

```
fig.set_title("")
```

```
fig.set_ylabel('Rainfall')
```

```
plt.subplot(2, 2, 2)
```

```
fig = df.boxplot(column='Evaporation')
```

```

fig.set_title("")
fig.set_ylabel('Evaporation')

plt.subplot(2, 2, 3)
fig = df.boxplot(column='WindSpeed9am')
fig.set_title("")
fig.set_ylabel('WindSpeed9am')

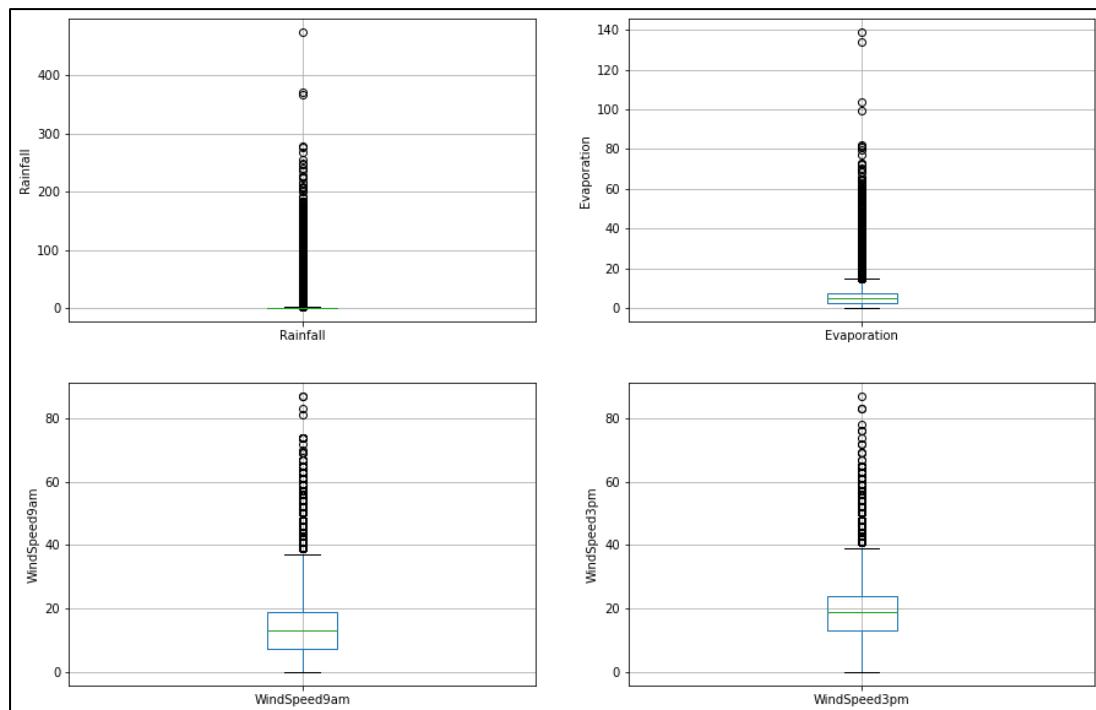
```

```

plt.subplot(2, 2, 4)
fig = df.boxplot(column='WindSpeed3pm')
fig.set_title("")
fig.set_ylabel('WindSpeed3pm')

```

### **Output:**



```
# plot histogram to check distribution
plt.figure(figsize=(15,10))

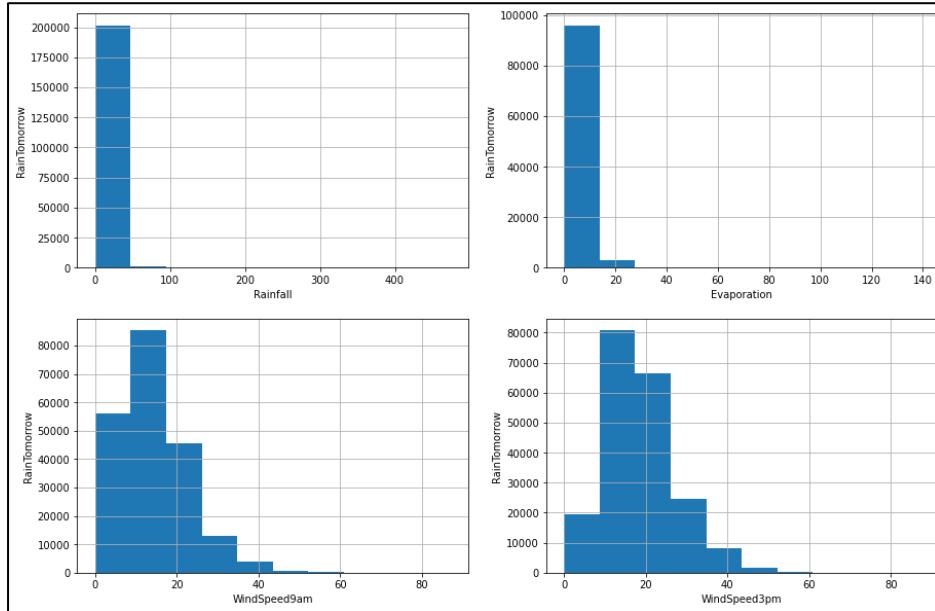
plt.subplot(2, 2, 1)
fig = df.Rainfall.hist(bins=10)
fig.set_xlabel('Rainfall')
fig.set_ylabel('RainTomorrow')

plt.subplot(2, 2, 2)
fig = df.Evaporation.hist(bins=10)
fig.set_xlabel('Evaporation')
fig.set_ylabel('RainTomorrow')

plt.subplot(2, 2, 3)
fig = df.WindSpeed9am.hist(bins=10)
fig.set_xlabel('WindSpeed9am')
fig.set_ylabel('RainTomorrow')

plt.subplot(2, 2, 4)
fig = df.WindSpeed3pm.hist(bins=10)
fig.set_xlabel('WindSpeed3pm')
fig.set_ylabel('RainTomorrow')
```

## Output:



# find outliers for Rainfall variable

```
IQR = df.Rainfall.quantile(0.75) - df.Rainfall.quantile(0.25)
```

```
Lower_fence = df.Rainfall.quantile(0.25) - (IQR * 3)
```

```
Upper_fence = df.Rainfall.quantile(0.75) + (IQR * 3)
```

```
print('Rainfall outliers are values < {lowerboundary} or > {upperboundary}'.format(lowerboundary=Lower_fence, upperboundary=Upper_fence))
```

## Output:

```
Rainfall outliers are values < -1.7999999999999998 or > 2.4
```

```
X = df.drop(['RainTomorrow'], axis=1)
```

```
y = df['RainTomorrow']
```

# split X and y into training and testing sets

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0)
```

```
X_train.shape, X_test.shape
```

## Output:

```
((166796, 24), (41699, 24))
```

```
categorical = [col for col in X_train.columns if X_train[col].dtypes == 'O']
```

```
categorical
```

**Output:**

```
['Location', 'WindGustDir', 'WindDir9am', 'WindDir3pm', 'RainToday']
```

```
X_test = pd.concat([X_test[numerical], X_test[['RainToday_0', 'RainToday_1']],
```

```
    pd.get_dummies(X_test.Location),
```

```
    pd.get_dummies(X_test.WindGustDir),
```

```
    pd.get_dummies(X_test.WindDir9am),
```

```
    pd.get_dummies(X_test.WindDir3pm)], axis=1)
```

```
from sklearn.preprocessing import MinMaxScaler
```

```
scaler = MinMaxScaler()
```

```
X_train = scaler.fit_transform(X_train)
```

```
X_test = scaler.transform(X_test)
```

```
X_train = pd.DataFrame(X_train, columns=[cols])
```

```
X_test = pd.DataFrame(X_test, columns=[cols])
```

```
X_train.describe()
```

**Output:**

	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustSpeed	WindSpeed9am
count	166796.000000	166796.000000	166796.000000	166796.000000	166796.000000	166796.000000	166796.000000
mean	0.487908	0.513154	0.202868	0.234397	0.560815	0.285994	0.255518
std	0.149906	0.134568	0.364995	0.118438	0.173719	0.098076	0.160017
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.382629	0.413662	0.000000	0.220183	0.586207	0.218045	0.127273
50%	0.483568	0.502846	0.000000	0.220183	0.586207	0.278195	0.236364
75%	0.596244	0.607211	0.187500	0.220183	0.586207	0.330827	0.345455
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

```
y_train.fillna('Yes', inplace=True)
```

```
y_test.fillna('Yes', inplace=True)
```

```
# train a logistic regression model on the training set
```

```
from sklearn.linear_model import LogisticRegression
```

```
# instantiate the model  
logreg = LogisticRegression(solver='liblinear', random_state=0)  
  
# fit the model  
logreg.fit(X_train, y_train)
```

**Output:**

```
LogisticRegression(random_state=0, solver='liblinear')
```

```
y_pred_test = logreg.predict(X_test)  
y_pred_test
```

**Output:**

```
array(['Yes', 'No', 'No', ..., 'No', 'Yes', 'No'], dtype=object)
```

```
# probability of getting output as 0 - no rain  
logreg.predict_proba(X_test)[:,0]
```

**Output:**

```
array([0.40401211, 0.94783797, 0.83423034, ..., 0.9788821 , 0.32928284,  
0.88280924])
```

```
# probability of getting output as 1 - rain  
logreg.predict_proba(X_test)[:,1]
```

**Output:**

```
array([0.59598789, 0.05216203, 0.16576966, ..., 0.0211179 , 0.67071716,  
0.11719076])
```

```
from sklearn.metrics import accuracy_score  
print('Model accuracy score: {:.4f}'.format(accuracy_score(y_test, y_pred_test)))
```

**Output:**

```
Model accuracy score: 0.8329
```

```
y_pred_train = logreg.predict(X_train)
```

```
y_pred_train
```

**Output:**

```
array(['No', 'No', 'No', ..., 'Yes', 'No', 'No'], dtype=object)
```

```
print('Training-set accuracy score: {:.4f}'.format(accuracy_score(y_train, y_pred_train)))
```

**Output:**

```
Training-set accuracy score: 0.8349
```

```
# print the scores on training and test set
```

```
print('Training set score: {:.4f}'.format(logreg.score(X_train, y_train)))
```

```
print('Test set score: {:.4f}'.format(logreg.score(X_test, y_test)))
```

**Output:**

```
Training set score: 0.8349  
Test set score: 0.8329
```

```
# check null accuracy score
```

```
null_accuracy = (31538/(31538+10161))
```

```
print('Null accuracy score: {:.4f}'.format(null_accuracy))
```

**Output:**

```
Null accuracy score: 0.7563
```

```
from sklearn.metrics import confusion_matrix
```

```
cm = confusion_matrix(y_test, y_pred_test)
```

```
print('Confusion matrix\n\n', cm)
```

```
print('\nTrue Positives(TP) = ', cm[0,0])
```

```
print('\nTrue Negatives(TN) = ', cm[1,1])
```

```
print('\nFalse Positives(FP) = ', cm[0,1])
```

```
print('\nFalse Negatives(FN) = ', cm[1,0])
```

### Output:

```
Confusion matrix

[[29652 1886]
 [ 5081 5080]]

True Positives(TP) = 29652

True Negatives(TN) = 5080

False Positives(FP) = 1886

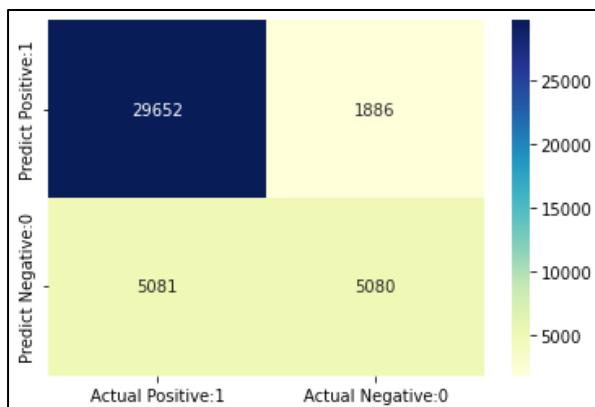
False Negatives(FN) = 5081
```

### # visualize confusion matrix with seaborn heatmap

```
cm_matrix = pd.DataFrame(data=cm, columns=['Actual Positive:1', 'Actual Negative:0'],
                           index=['Predict Positive:1', 'Predict Negative:0'])

sns.heatmap(cm_matrix, annot=True, fmt='d', cmap='YlGnBu')
```

### Output:



```
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred_test))
```

### Output:

	precision	recall	f1-score	support
No	0.85	0.94	0.89	31538
Yes	0.73	0.50	0.59	10161
accuracy			0.83	41699
macro avg	0.79	0.72	0.74	41699
weighted avg	0.82	0.83	0.82	41699

```

TP = cm[0,0]
TN = cm[1,1]
FP = cm[0,1]
FN = cm[1,0]

# print classification accuracy
classification_accuracy = (TP + TN) / float(TP + TN + FP + FN)
print('Classification accuracy : {0:0.4f}'.format(classification_accuracy))

```

**Output:**

Classification accuracy : 0.8329

```

# print classification error
classification_error = (FP + FN) / float(TP + TN + FP + FN)
print('Classification error : {0:0.4f}'.format(classification_error))

```

**Output:**

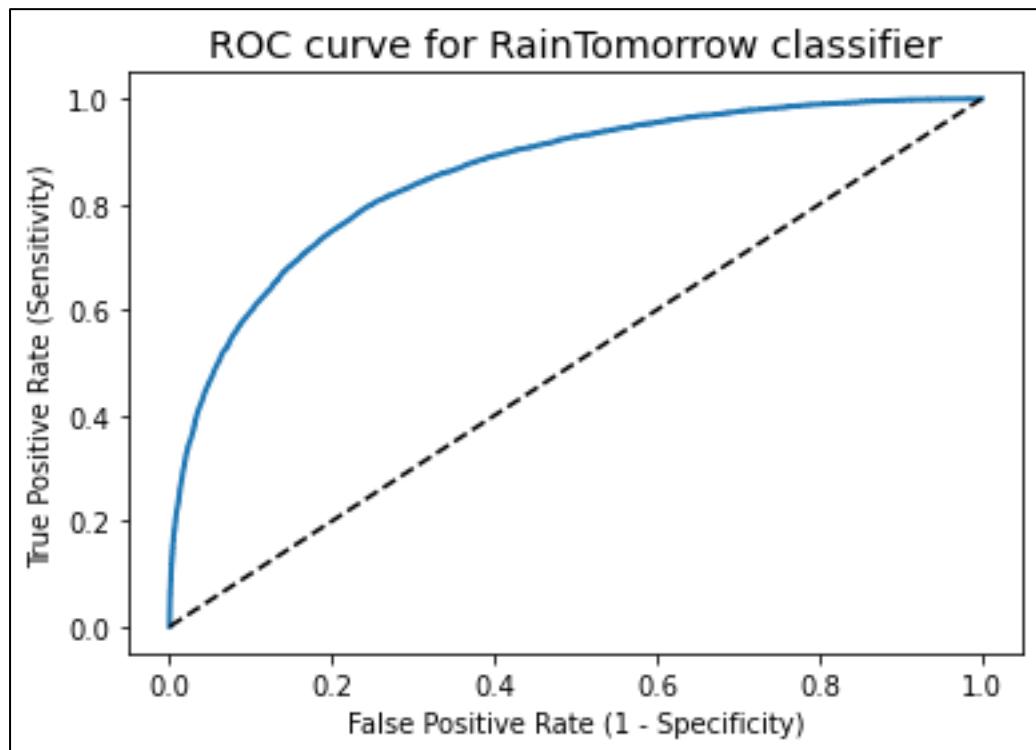
Classification error : 0.1671

```

# plot ROC Curve
y_pred1 = logreg.predict_proba(X_test)[:,1]
y_pred1 = y_pred1.reshape(-1,1)
from sklearn.metrics import roc_curve
fpr, tpr, thresholds = roc_curve(y_test, y_pred1, pos_label = 'Yes')
plt.figure(figsize=(6,4))
plt.plot(fpr, tpr, linewidth=2)
plt.plot([0,1], [0,1], 'k--' )
plt.rcParams['font.size'] = 12
plt.title('ROC curve for RainTomorrow classifier')
plt.xlabel('False Positive Rate (1 - Specificity)')
plt.ylabel('True Positive Rate (Sensitivity)')
plt.show()

```

**Output:**



```
from sklearn.metrics import roc_auc_score  
ROC_AUC = roc_auc_score(y_test, y_pred1)  
print('ROC AUC : {:.4f}'.format(ROC_AUC))
```

**Output:**

```
ROC AUC : 0.8571
```

```
from sklearn.model_selection import cross_val_score  
scores = cross_val_score(logreg, X_train, y_train, cv = 5, scoring='accuracy')  
print('Cross-validation scores:{}'.format(scores))
```

**Output:**

```
Cross-validation scores:[0.83228417 0.83326838 0.83326838 0.83776492 0.83599628]
```

```
# compute Average cross-validation score  
print('Average cross-validation score: {:.4f}'.format(scores.mean()))
```

**Output:**

```
Average cross-validation score: 0.8345
```

```
from sklearn.model_selection import GridSearchCV  
parameters = [ {'penalty':['l1','l2']}, {'C':[1, 10, 100, 1000]}]  
grid_search = GridSearchCV(estimator = logreg, param_grid = parameters,scoring =  
'accuracy', cv = 5,verbose=0)  
grid_search.fit(X_train, y_train)
```

**Output:**

```
GridSearchCV(cv=5,  
            estimator=LogisticRegression(random_state=0, solver='liblinear'),  
            param_grid=[{'penalty': ['l1', 'l2']}, {'C': [1, 10, 100, 1000]}],  
            scoring='accuracy')
```

```
# examine the best model  
  
# best score achieved during the GridSearchCV  
  
print('GridSearch CV best score : {:.4f}\n'.format(grid_search.best_score_))  
  
# print parameters that give the best results  
  
print('Parameters that give the best results :', '\n', (grid_search.best_params_))  
  
# print estimator that was chosen by the GridSearch  
  
print('\n\nEstimator that was chosen by the search :', '\n', (grid_search.best_estimator_))
```

**Output:**

```
GridSearch CV best score : 0.8347  
  
Parameters that give the best results :  
  
{'C': 10}  
  
Estimator that was chosen by the search :  
  
LogisticRegression(C=10, random_state=0, solver='liblinear')
```

```
# calculate GridSearch CV score on test set  
print('GridSearch CV score on test set: {:.4f}'.format(grid_search.score(X_test, y_test)))
```

**Output:**

**GridSearch CV score on test set: 0.8328**

**Result:**

Logistic Regression model has been successfully built and the results are verified.

## MULTIPLE REGRESSION ANALYSIS

### 1)Aim:

To build a multiple linear regression model

### Code:

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

```
data = pd.read_csv("dow_jones_index.csv")
```

```
data
```

### Output:

quarter	stock	date	open	high	low	close	volume	percent_change_price	percent_change_volume_over_last_wk	p
0	1	AA 1/7/2011	\$15.82	\$16.72	\$15.78	\$16.42	239655616	3.79267		NaN
1	1	AA 1/14/2011	\$16.71	\$16.71	\$15.64	\$15.97	242963398	-4.42849		1.380223
2	1	AA 1/21/2011	\$16.19	\$16.38	\$15.60	\$15.79	138428495	-2.47066		-43.024959
3	1	AA 1/28/2011	\$15.87	\$16.63	\$15.82	\$16.13	151379173	1.63831		9.355500
4	1	AA 2/4/2011	\$16.18	\$17.39	\$16.18	\$17.14	154387761	5.93325		1.987452
...	...	...	...	...	...	...	...	...		...
745	2	XOM 5/27/2011	\$80.22	\$82.63	\$80.07	\$82.63	68230855	3.00424		-21.355713
746	2	XOM 6/3/2011	\$83.28	\$83.75	\$80.18	\$81.18	78616295	-2.52161		15.221032
747	2	XOM 6/10/2011	\$80.93	\$81.87	\$79.72	\$79.78	92380844	-1.42098		17.508519
748	2	XOM 6/17/2011	\$80.00	\$80.82	\$78.33	\$79.02	100521400	-1.22500		8.811952
749	2	XOM 6/24/2011	\$78.65	\$81.12	\$76.78	\$76.78	118679791	-2.37762		18.064204

```
data['high'] = data['high'].str.replace('$', '')
data['open'] = data['open'].str.replace('$', '')
data['low'] = data['low'].str.replace('$', '')
data['close'] = data['close'].str.replace('$', '')
data['next_weeks_open'] = data['next_weeks_open'].str.replace('$', '')
data['next_weeks_close'] = data['next_weeks_close'].str.replace('$', '')
```

```
data['high']=pd.to_numeric(data['high'])
data['open']=pd.to_numeric(data['open'])
```

```

data['low']=pd.to_numeric(data['low'])

data['close']=pd.to_numeric(data['close'])

data['next_weeks_open']=pd.to_numeric(data['next_weeks_open'])

data['next_weeks_close']=pd.to_numeric(data['next_weeks_close'])

data['percent_change_volume_over_last_wk'].fillna(value=0,inplace=True)

data['previous_weeks_volume'].fillna(value=0,inplace=True)

data

```

**Output:**

quarter	stock	date	open	high	low	close	volume	percent_change_price	percent_change_volume_over_last_wk	pre
0	1	AA 1/7/2011	15.82	16.72	15.78	16.42	239655616	3.79267	0.000000	
1	1	AA 1/14/2011	16.71	16.71	15.64	15.97	242963398	-4.42849	1.380223	
2	1	AA 1/21/2011	16.19	16.38	15.60	15.79	138428495	-2.47066	-43.024959	
3	1	AA 1/28/2011	15.87	16.63	15.82	16.13	151379173	1.63831	9.355500	
4	1	AA 2/4/2011	16.18	17.39	16.18	17.14	154387761	5.93325	1.987452	
...	...	...	...	...	...	...	...	...	...	...
745	2	XOM 5/27/2011	80.22	82.63	80.07	82.63	68230855	3.00424	-21.355713	
746	2	XOM 6/3/2011	83.28	83.75	80.18	81.18	78616295	-2.52161	15.221032	
747	2	XOM 6/10/2011	80.93	81.87	79.72	79.78	92380844	-1.42098	17.508519	
748	2	XOM 6/17/2011	80.00	80.82	78.33	79.02	100521400	-1.22500	8.811952	

data.info()

**Output:**

```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 750 entries, 0 to 749
Data columns (total 16 columns):
 #   Column           Non-Null Count  Dtype  
 ---  -- 
 0   quarter          750 non-null    int64  
 1   stock             750 non-null    object 
 2   date              750 non-null    object 
 3   open               750 non-null    float64
 4   high              750 non-null    float64
 5   low               750 non-null    float64
 6   close              750 non-null    float64
 7   volume             750 non-null    int64  
 8   percent_change_price  750 non-null    float64
 9   percent_change_volume_over_last_wk 750 non-null    float64
 10  previous_weeks_volume      750 non-null    float64
 11  next_weeks_open         750 non-null    float64
 12  next_weeks_close        750 non-null    float64
 13  percent_change_next_weeks_price 750 non-null    float64
 14  days_to_next_dividend   750 non-null    int64  
 15  percent_return_next_dividend 750 non-null    float64
dtypes: float64(11), int64(3), object(2)
memory usage: 93.9+ KB

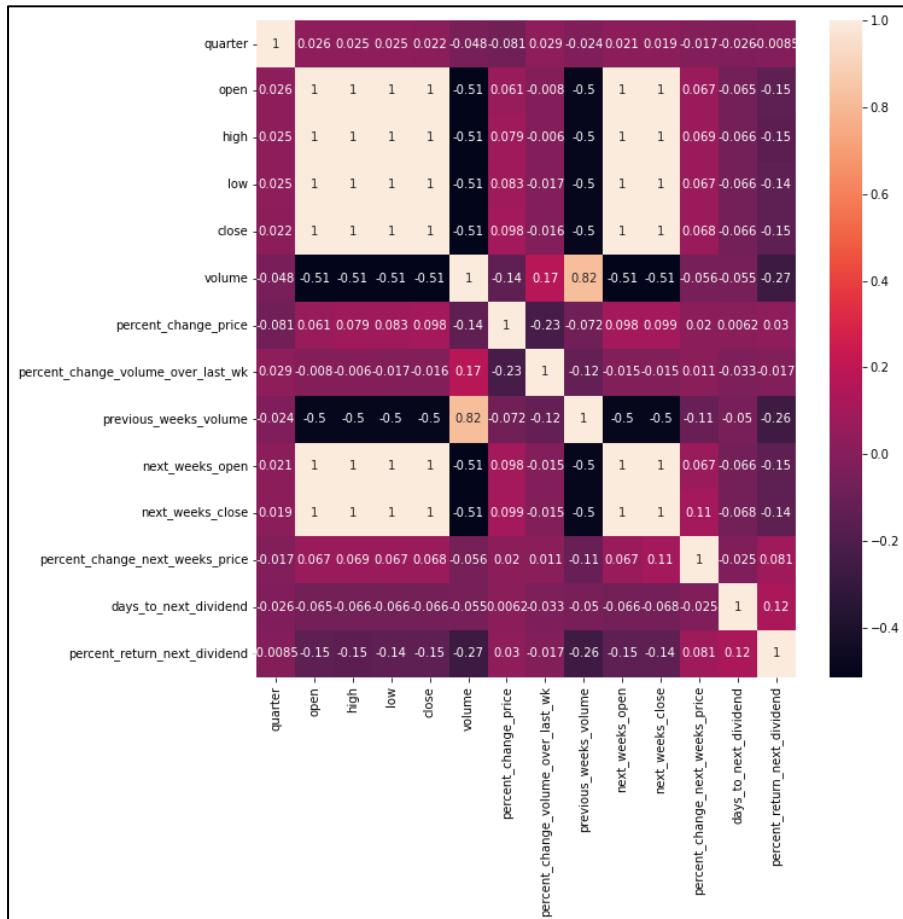
```

```

plt.figure(figsize=(10,10))
sns.heatmap(data.corr(),annot=True)

```

**Output:**



```

plt.scatter(data.index,data['stock'])

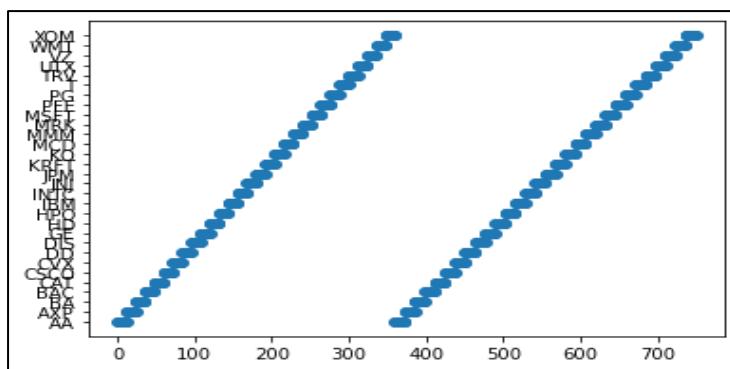
```

```

plt.show()

```

**Output:**



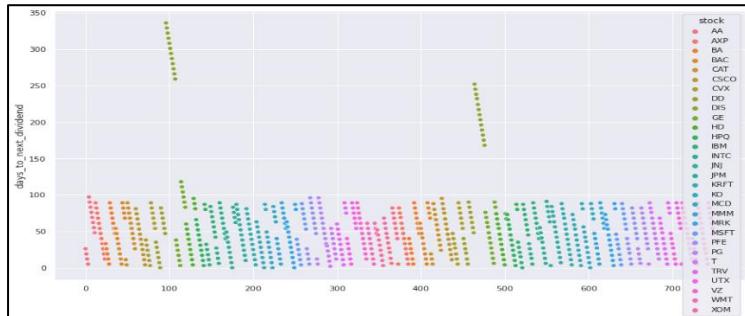
```

sns.set(rc={'figure.figsize':(15,8)})

```

```
sns.scatterplot(x=data.index,y=data['days_to_next_dividend'],hue=data['stock'])
```

**Output:**



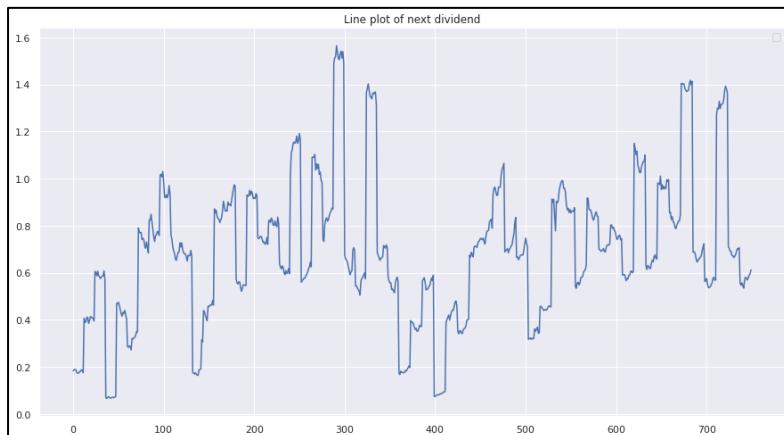
```
plt.title("Line plot of next dividend")
```

```
plt.plot(data.index,data['percent_return_next_dividend'])
```

```
plt.legend()
```

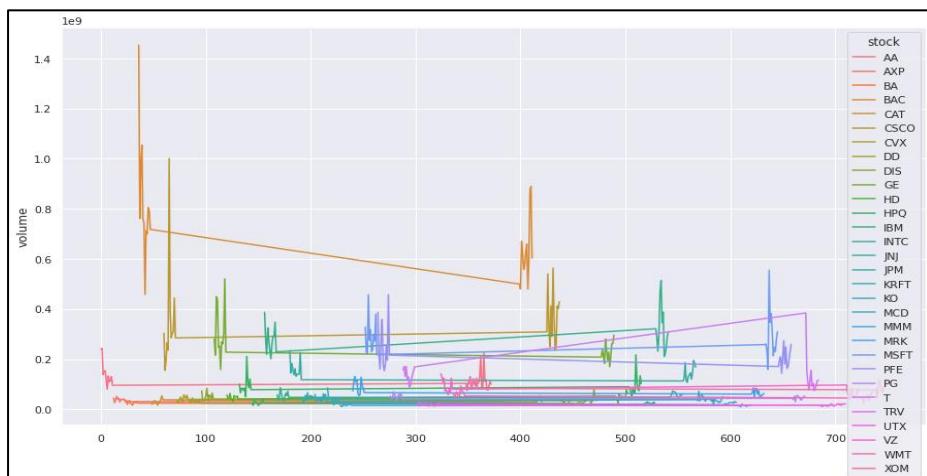
```
plt.show()
```

**Output:**



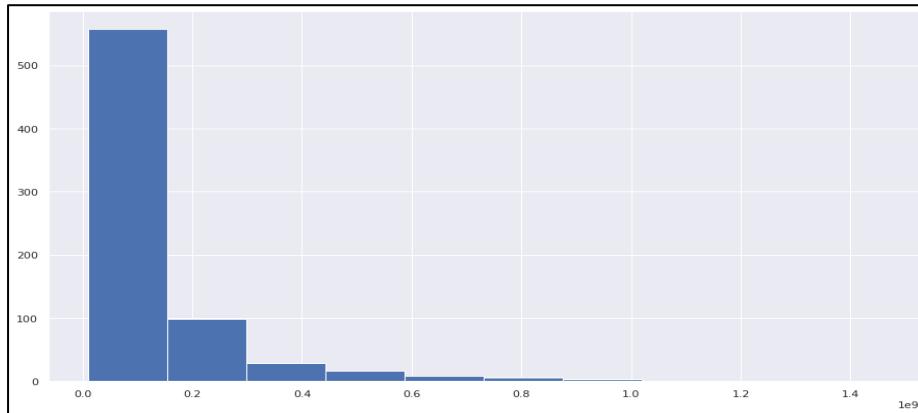
```
fig=sns.lineplot(x=data.index,y=data['volume'],hue=data['stock'])
```

**Output:**



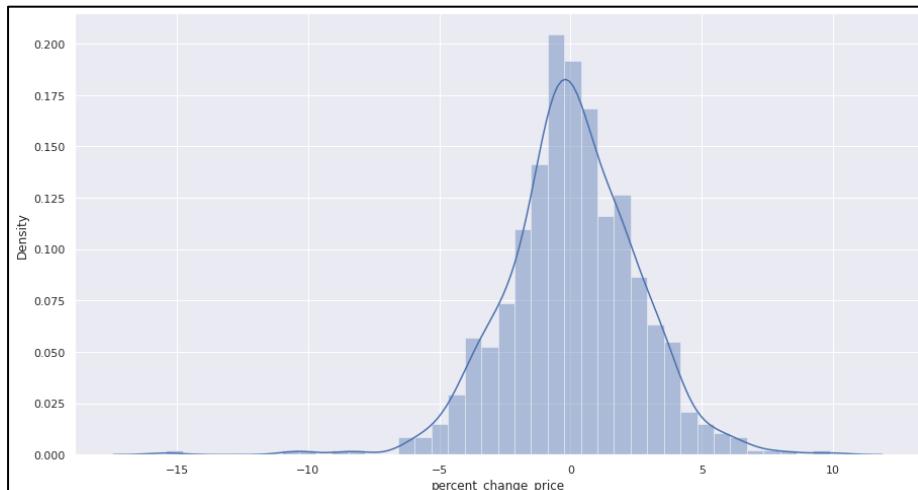
```
plt.hist(data['previous_weeks_volume'])
```

**Output:**



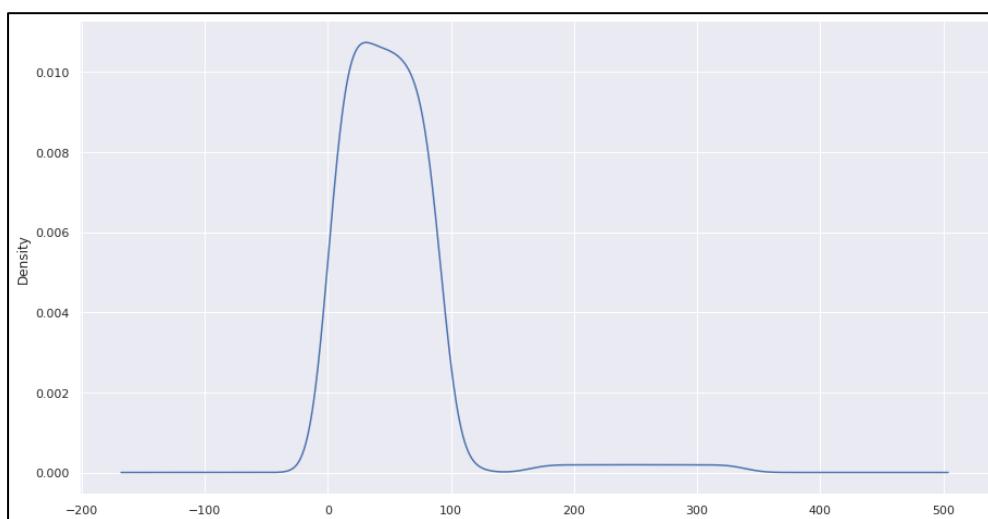
```
sns.distplot(data['percent_change_price'])
```

**Output:**



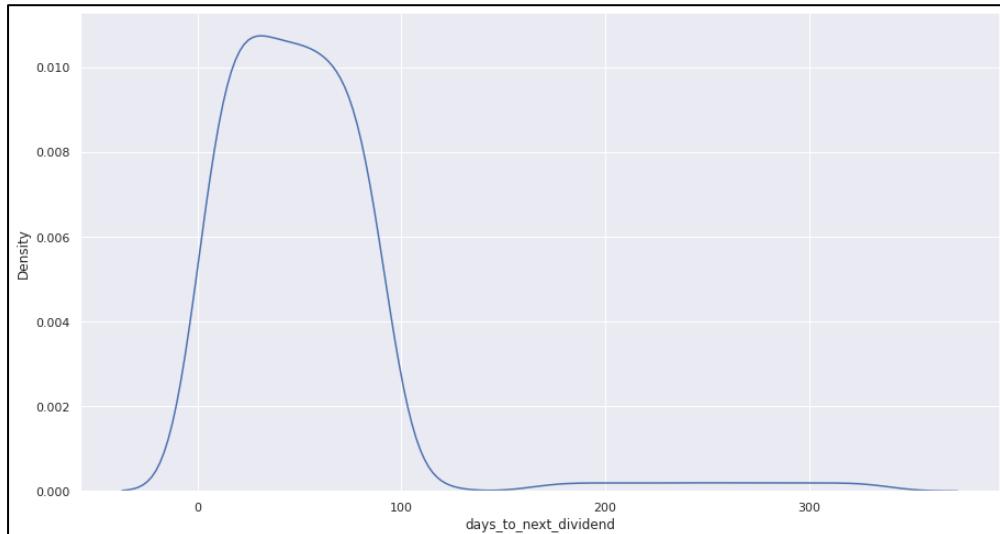
```
data['days_to_next_dividend'].plot(kind='density')
```

**Output:**



```
sns.kdeplot(data['days_to_next_dividend'])
```

**Output:**



```
data.var()
```

**Output:**

<b>Out[28]:</b>	quarter	2.499332e-01
	volume	2.510263e+16
	percent_change_price	6.339363e+00
	percent_change_volume_over_last_wk	1.643774e+03
	previous_weeks_volume	2.535490e+16
	percent_change_next_weeks_price	7.179926e+00
	days_to_next_dividend	2.146941e+03
	percent_return_next_dividend	9.331917e-02
	dtype: float64	

```
cols=['open','high','low','close','volume','percent_change_price','next_weeks_open','next_weeks_close','days_to_next_dividend','percent_return_next_dividend']
```

```
X=data[cols]
```

```
y=data['quarter']
```

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state=9)
```

```
from sklearn.linear_model import LinearRegression
```

```
model=LinearRegression()
```

```
model.fit(X_train,y_train)
```

**Output:**

```
model.fit(X_train,y_train)
```

```
LinearRegression()
```

```
pred=model.predict(X_test)
```

```
pred
```

**Output:**

```
array([1.54913074, 1.54813425, 1.49749033, 1.54832875, 1.57205104,
       1.55717716, 1.52709849, 1.52365576, 1.49634746, 1.46801631,
       1.55007912, 1.50031549, 1.52780842, 1.48809572, 1.49257709,
       1.42785859, 1.4869727 , 1.49517245, 1.5455281 , 1.6338926 ,
       1.49020232, 1.42692687, 1.52580381, 1.52978727, 1.52162128,
       1.48612595, 1.50104161, 1.55947532, 1.58657317, 1.45367538,
       1.4444433 , 1.44392882, 1.54155301, 1.56212753, 1.59859272,
       1.53719739, 1.45900454, 1.6713445 , 1.46808814, 1.33376232,
       1.48355797, 1.48292294, 1.5028681 , 1.54954379, 1.55624776,
       1.59012436, 1.75127226, 1.56747024, 1.46323332, 1.43074352,
       1.4686594 , 1.45738999, 1.44713124, 1.61041957, 1.52232834,
       1.46044442, 1.50823382, 1.51220186, 1.52237218, 1.51761204,
       1.58537606, 1.53079649, 1.63827155, 1.63449266, 1.59079475,
       1.65187618, 1.47724958, 1.4886941 , 1.59469714, 1.47597925,
       1.64065202, 1.50897175, 1.63079161, 1.5126231 , 1.51115985,
       1.63870211, 1.47376849, 1.49901546, 1.54872444, 1.47210942,
       1.56179107, 1.53450276, 1.57096706, 1.50328843, 1.64137974,
       1.56695183, 1.53711204, 1.51617007, 1.49040867, 1.52103114,
       1.38263396, 1.58400685, 1.53274877, 1.60790847, 1.51426924,
       1.47758578 1.53083506 1.49721352 1.6319571 1.51986792])
```

```
print('Coefficients: ', model.coef_)
```

**Output:**

```
Coefficients: [ 1.21559843e-03  1.68935129e-02  1.32063229e-02  9.71230820e-02
 -6.90544664e-11 -7.60770675e-03 -1.14063908e-01 -1.42938759e-02
 -1.34534545e-04 -4.45672353e-02]
```

```
print('Variance score: {:.2f}'.format(model.score(X_test, y_test)))
```

**Output:**

```
Variance score: 0.012848113274527173
```

```
plt.scatter(model.predict(X_train), model.predict(X_train) - y_train,
```

```
color = "green", s = 10, label = 'Train data')
```

```
## plotting residual errors in test data
```

```
plt.scatter(model.predict(X_test), model.predict(X_test) - y_test,
```

```

color = "blue", s = 10, label = 'Test data')

## plotting line for zero residual error

plt.hlines(y = 0, xmin = 0, xmax = 50, linewidth = 2)

## plotting legend

plt.legend(loc = 'upper right')

## plot title

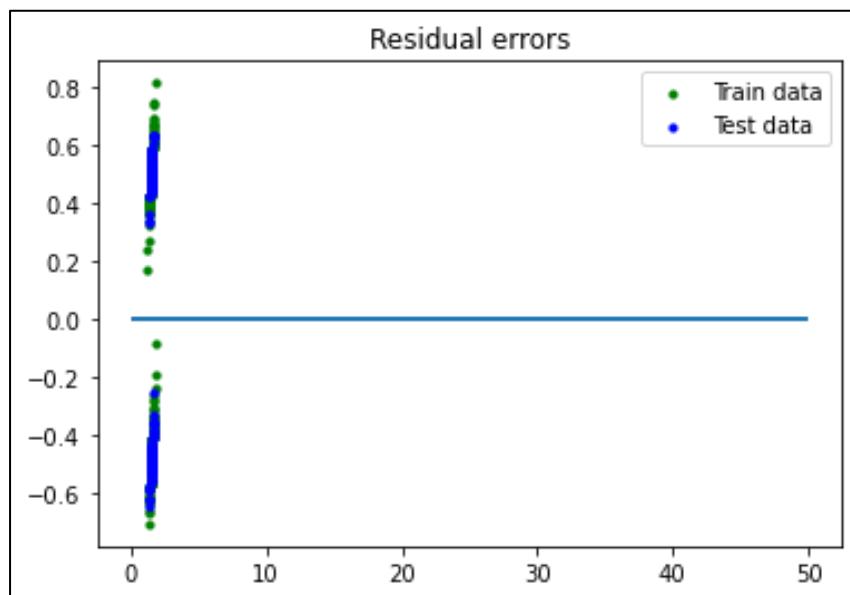
plt.title("Residual errors")

## method call for showing the plot

plt.show()

```

**Output:**



**Result:**

Multivariate Linear Regression model has been successfully built and the results are verified.

## Classification –NaïveBayes

### 1)Aim:

To build a model to perform Naïve Bayes classifier with following specifications :

- a. Exploratory data analysis
- b. Identify Numerical and Categorical variables
- c. Identify the missing values
- d. Apply feature engineering and apply feature scaling
- e. Train the model
- d. Test the model and display the results
- e. Check accuracy score
- f. Check for overfitting and underfitting
- g. Summarize the performance of model using confusion matrix
- h. Evaluate the model performance with classification report
- i. Analyze the model performance visually.

### Code:

#### i) Iris Dataset

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
dataset = pd.read_csv('iris.csv')
from sklearn import preprocessing
label_encoder = preprocessing.LabelEncoder()
dataset['species']= label_encoder.fit_transform(dataset['species'])
dataset['species'].unique()
```

### Output:

```
array([0, 1, 2])
```

```
X = dataset.iloc[:, :4].values
y = dataset['species'].values
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2)
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
from sklearn.naive_bayes import GaussianNB
classifier = GaussianNB()
```

```
classifier.fit(X_train, y_train)
```

**Output:**

```
: from sklearn.naive_bayes import GaussianNB  
classifier = GaussianNB()  
classifier.fit(X_train, y_train)  
  
: GaussianNB()
```

```
y_pred = classifier.predict(X_test)
```

```
y_pred
```

**Output:**

```
array([2, 2, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 2, 2, 1, 0, 1, 0, 2, 0, 0, 2,  
     0, 1, 1, 0, 2, 0, 1, 2])
```

```
from sklearn.metrics import confusion_matrix
```

```
cm = confusion_matrix(y_test, y_pred)
```

```
from sklearn.metrics import accuracy_score
```

```
print ("Accuracy : ", accuracy_score(y_test, y_pred))
```

```
cm
```

**Output:**

```
Accuracy : 0.9333333333333333
```

```
array([[11,  0,  0],  
       [ 0,  9,  0],  
       [ 0,  2,  8]])
```

```
from sklearn.metrics import classification_report
```

```
print(classification_report(y_test,y_pred))
```

**Output:**

	precision	recall	f1-score	support
0	1.00	1.00	1.00	11
1	0.82	1.00	0.90	9
2	1.00	0.80	0.89	10
accuracy			0.93	30
macro avg	0.94	0.93	0.93	30
weighted avg	0.95	0.93	0.93	30

```

from sklearn.model_selection import KFold
from sklearn.metrics import mean_absolute_error
kf = KFold(n_splits=20)
list_training_error = []
list_testing_error = []
for train_index, test_index in kf.split(X):
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]
    model = GaussianNB()
    model.fit(X_train, y_train)
    y_train_data_pred = model.predict(X_train)
    y_test_data_pred = model.predict(X_test)
    fold_training_error = mean_absolute_error(y_train, y_train_data_pred)
    fold_testing_error = mean_absolute_error(y_test, y_test_data_pred)
    list_training_error.append(fold_training_error)
    list_testing_error.append(fold_testing_error)

plt.subplot(1,2,1)
plt.plot(range(1, kf.get_n_splits() + 1), np.array(list_training_error).ravel(), 'o-')
plt.xlabel('number of fold')
plt.ylabel('training error')
plt.title('Training error across folds')
plt.tight_layout()

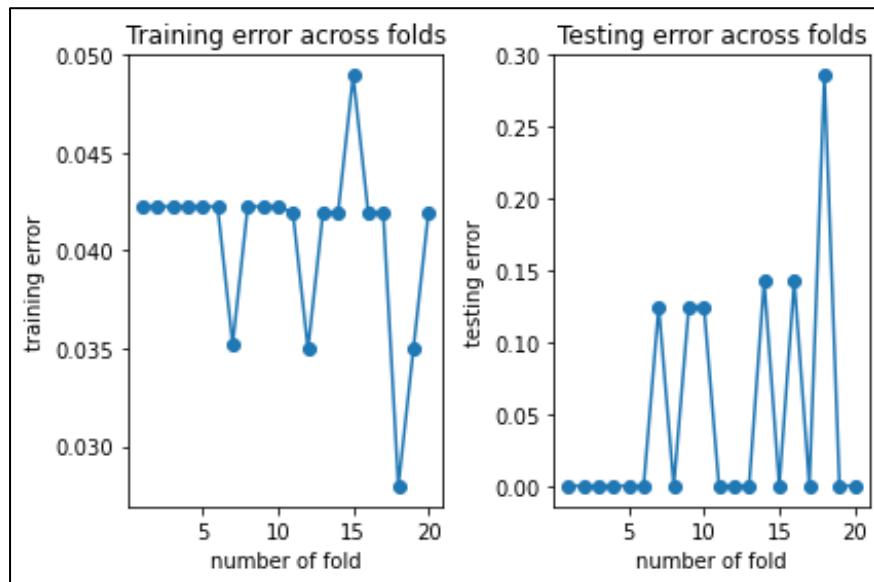
plt.subplot(1,2,2)

plt.plot(range(1, kf.get_n_splits() + 1), np.array(list_testing_error).ravel(), 'o-')
plt.xlabel('number of fold')
plt.ylabel('testing error')
plt.title('Testing error across folds')
plt.tight_layout()

```

```
plt.show()
```

**Output:**



**ii)Diabetes.csv**

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import confusion_matrix
from sklearn.metrics import f1_score
from sklearn.metrics import accuracy_score
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2
from sklearn.feature_selection import VarianceThreshold
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
dataset = pd.read_csv('diabetes.csv')

print(dataset.describe())
```

```
X = dataset.iloc[:, 0:8]
```

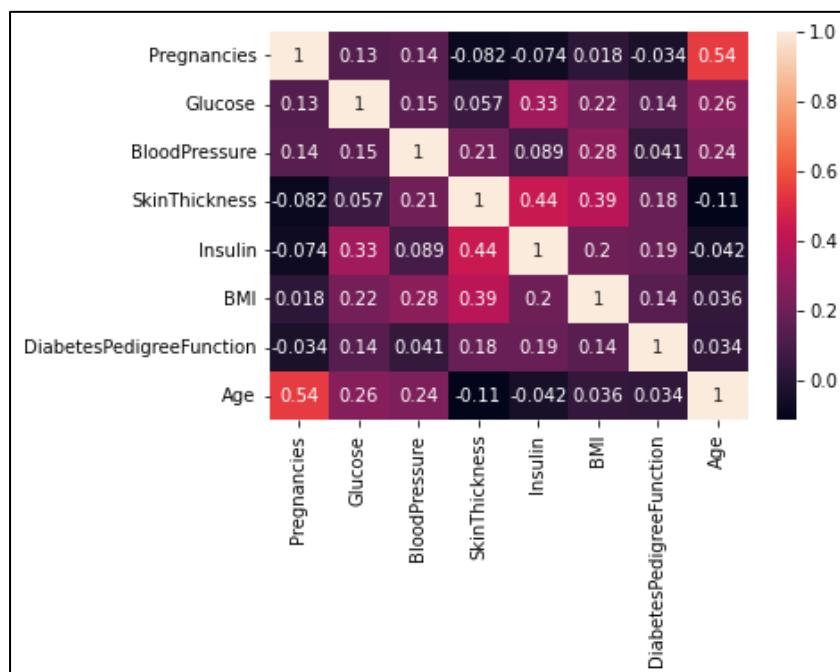
```
y = dataset.iloc[:, 8]
```

### Output:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	\
count	768.000000	768.000000	768.000000	768.000000	768.000000	
mean	3.845052	120.894531	69.105469	20.536458	79.799479	
std	3.369578	31.972618	19.355807	15.952218	115.244002	
min	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	1.000000	99.000000	62.000000	0.000000	0.000000	
50%	3.000000	117.000000	72.000000	23.000000	30.500000	
75%	6.000000	140.250000	80.000000	32.000000	127.250000	
max	17.000000	199.000000	122.000000	99.000000	846.000000	
	BMI	DiabetesPedigreeFunction	Age	Outcome		
count	768.000000	768.000000	768.000000	768.000000		
mean	31.992578	0.471876	33.240885	0.348958		
std	7.884160	0.331329	11.760232	0.476951		
min	0.000000	0.078000	21.000000	0.000000		
25%	27.300000	0.243750	24.000000	0.000000		
50%	32.000000	0.372500	29.000000	0.000000		
75%	36.600000	0.626250	41.000000	1.000000		
max	67.100000	2.420000	81.000000	1.000000		

```
sns.heatmap(X.corr(), annot = True)
```

### Output:



```
zero_not_accepted = ['Glucose', 'BloodPressure', 'SkinThickness', 'BMI', 'Insulin']
```

```
for column in zero_not_accepted:
```

```
    X[column] = X[column].replace(0,np.NaN)
```

```
    mean = int(X[column].mean(skipna=True))
```

```
    X[column] = X[column].replace(np.NaN, mean)
```

```

sel = VarianceThreshold(threshold=(.8 * (1 - .8)))

X_filtered = sel.fit_transform(X)

print(X.head(1))

print(X_filtered[0])

#DiabetesPedigreeFunction was dropped

X = X.drop('DiabetesPedigreeFunction', axis=1)

top_4_features = SelectKBest(score_func=chi2, k=4)

X_top_4_features = top_4_features.fit_transform(X, y)

print(X.head())

print(X_top_4_features)

X = X.drop(['Pregnancies', 'BloodPressure', 'SkinThickness'], axis=1)

```

**Output:**

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	\
0	6	148	72	35	79	33.6	
							DiabetesPedigreeFunction
0							Age
							0.627
	[	6.	148.	72.	35.	79.	33.6
							50.
							]
							Pregnancies
0							Glucose
1							BloodPressure
2							SkinThickness
3							Insulin
4							BMI
							Age
0	6	148	72	35	79	33.6	50
1	1	85	66	29	79	26.6	31
2	8	183	64	20	79	23.3	32
3	1	89	66	23	94	28.1	21
4	0	137	40	35	168	43.1	33
	[[	6.	148.	79.	50.		]
		1.	85.	79.	31.		]
		8.	183.	79.	32.		]
	...						
	[[	5.	121.	112.	30.		]
		1.	126.	79.	47.		]
		1.	93.	79.	23.		]]

```

X_train, X_test, y_train, y_test = train_test_split(X.values, y.values, random_state=0,
test_size=0.20)

sc_X = StandardScaler()

X_train = sc_X.fit_transform(X_train)

X_test = sc_X.transform(X_test)

classifier = GaussianNB()

classifier.fit(X_train, y_train)

```

**Output:**

GaussianNB()
--------------

```
y_pred = classifier.predict(X_test)
```

```
cm = confusion_matrix(y_test, y_pred)
print(cm)
print(f1_score(y_test, y_pred))
print(accuracy_score(y_test, y_pred))
```

**Output:**

```
[[96 11]
 [20 27]]
0.6352941176470589
0.7987012987012987
```

```
from sklearn.metrics import classification_report
print(classification_report(y_test,y_pred))
```

**Output:**

	precision	recall	f1-score	support
0	0.83	0.90	0.86	107
1	0.71	0.57	0.64	47
accuracy			0.80	154
macro avg	0.77	0.74	0.75	154
weighted avg	0.79	0.80	0.79	154

```
from sklearn.model_selection import KFold
from sklearn.metrics import mean_absolute_error
kf = KFold(n_splits=20)
list_training_error = []
list_testing_error = []
X=X.values
y=y.values
for train_index, test_index in kf.split(X):
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]
    model = GaussianNB()
    model.fit(X_train, y_train)
```

```

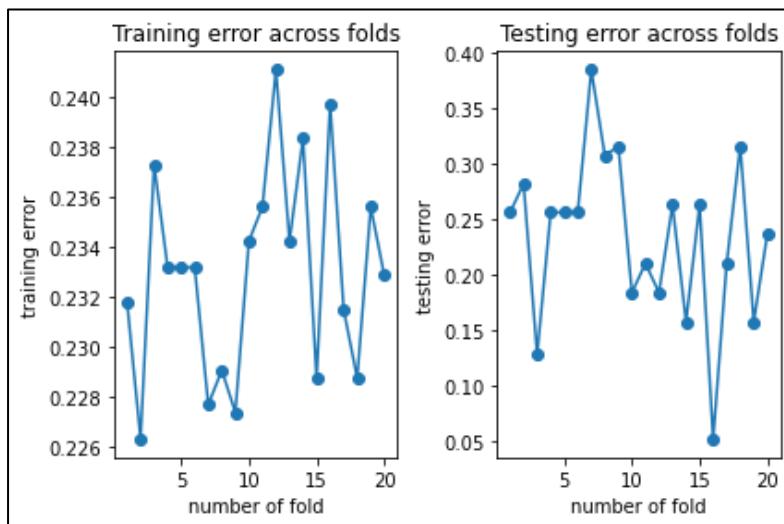
y_train_data_pred = model.predict(X_train)
y_test_data_pred = model.predict(X_test)
fold_training_error = mean_absolute_error(y_train, y_train_data_pred)
fold_testing_error = mean_absolute_error(y_test, y_test_data_pred)
list_training_error.append(fold_training_error)
list_testing_error.append(fold_testing_error)

plt.subplot(1,2,1)
plt.plot(range(1, kf.get_n_splits() + 1), np.array(list_training_error).ravel(), 'o-')
plt.xlabel('number of fold')
plt.ylabel('training error')
plt.title('Training error across folds')
plt.tight_layout()

plt.subplot(1,2,2)
plt.plot(range(1, kf.get_n_splits() + 1), np.array(list_testing_error).ravel(), 'o-')
plt.xlabel('number of fold')
plt.ylabel('testing error')
plt.title('Testing error across folds')
plt.tight_layout()
plt.show()

```

**Output:**



### iii) dow\_jones\_index

```
import numpy as np  
import matplotlib.pyplot as plt  
import pandas as pd  
data = pd.read_csv('dow_jones_index.csv')  
from sklearn.preprocessing import LabelEncoder  
encoder=LabelEncoder()  
data['stock']=encoder.fit_transform(data['stock'])  
data['stock'].unique()
```

#### Output:

```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 17,  
       16, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29])
```

```
categorical = [cols for cols in data.columns if data[cols].dtype=='O']
```

```
categorical
```

#### Output:

```
['date']
```

```
numerical = [cols for cols in data.columns if data[cols].dtype!='O']
```

```
numerical
```

#### Output:

```
['quarter',  
 'stock',  
 'open',  
 'high',  
 'low',  
 'close',  
 'volume',  
 'percent_change_price',  
 'percent_change_volume_over_last_wk',  
 'previous_weeks_volume',  
 'next_weeks_open',  
 'next_weeks_close',  
 'percent_change_next_weeks_price',  
 'days_to_next_dividend',  
 'percent_return_next_dividend']
```

```

cols=['open','high','low','volume','close','percent_change_price','percent_change_volume_over
_last_wk','previous_weeks_volume','next_weeks_open','next_weeks_close','percent_change_
next_weeks_price','days_to_next_dividend','percent_return_next_dividend']

X=data[cols]

y=data['stock']

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2)

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()

X_train = sc.fit_transform(X_train)

X_test = sc.transform(X_test)

from sklearn.naive_bayes import GaussianNB

classifier = GaussianNB()

classifier.fit(X_train, y_train)

```

**Output:**

**GaussianNB()**

```
y_pred = classifier.predict(X_test)
```

```
y_pred
```

**Output:**

```
array([19, 19, 2, 22, 23, 10, 5, 14, 21, 2, 16, 1, 24, 12, 11, 19, 19,
       17, 23, 19, 9, 18, 19, 18, 12, 7, 18, 25, 7, 0, 0, 27, 25, 6,
       27, 20, 23, 7, 7, 9, 13, 4, 9, 24, 0, 17, 24, 19, 2, 20, 23,
       16, 12, 15, 2, 11, 17, 24, 17, 7, 28, 6, 10, 29, 18, 9, 4, 27,
       23, 6, 21, 7, 16, 5, 6, 2, 27, 27, 3, 19, 20, 11, 18, 4, 15,
       28, 12, 2, 29, 5, 26, 20, 19, 26, 13, 17, 13, 14, 15, 3, 19, 28,
       21, 2, 13, 17, 24, 5, 24, 9, 6, 0, 26, 18, 28, 18, 8, 4, 9,
       6, 27, 28, 7, 24, 13, 25, 5, 24, 26, 6, 7, 3, 6, 18, 13, 22,
       16, 12, 9, 28, 29, 23, 6, 13, 3, 3, 7, 10, 21, 27])
```

```
from sklearn.metrics import accuracy_score
```

```
print ("Accuracy : ", accuracy_score(y_test, y_pred))
```

**Output:**

**Accuracy : 0.96**

```
from sklearn.metrics import classification_report  
print(classification_report(y_test,y_pred))
```

**Output:**

	precision	recall	f1-score	support
0	1.00	1.00	1.00	4
1	1.00	1.00	1.00	1
2	1.00	1.00	1.00	7
3	1.00	1.00	1.00	5
4	1.00	1.00	1.00	4
5	1.00	1.00	1.00	5
6	1.00	1.00	1.00	9
7	0.89	1.00	0.94	8
8	1.00	1.00	1.00	1
9	1.00	1.00	1.00	7
10	1.00	1.00	1.00	3

accuracy			0.96	150
macro avg	0.97	0.95	0.95	150
weighted avg	0.97	0.96	0.96	150

```
from sklearn.model_selection import KFold  
  
from sklearn.metrics import mean_absolute_error  
  
kf = KFold(n_splits=20)  
  
list_training_error = []  
list_testing_error = []  
  
for train_index, test_index in kf.split(X):  
    X_train, X_test = X[train_index], X[test_index]  
    y_train, y_test = y[train_index], y[test_index]  
    model = GaussianNB()  
    model.fit(X_train, y_train)  
    y_train_data_pred = model.predict(X_train)  
    y_test_data_pred = model.predict(X_test)  
    fold_training_error = mean_absolute_error(y_train, y_train_data_pred)  
    fold_testing_error = mean_absolute_error(y_test, y_test_data_pred)
```

```

list_training_error.append(fold_training_error)
list_testing_error.append(fold_testing_error)

plt.subplot(1,2,1)
plt.plot(range(1, kf.get_n_splits() + 1), np.array(list_training_error).ravel(), 'o-')
plt.xlabel('number of fold')
plt.ylabel('training error')
plt.title('Training error across folds')
plt.tight_layout()

plt.subplot(1,2,2)

plt.plot(range(1, kf.get_n_splits() + 1), np.array(list_testing_error).ravel(), 'o-')
plt.xlabel('number of fold')
plt.ylabel('testing error')
plt.title('Testing error across folds')
plt.tight_layout()
plt.show()

```

### **Output:**



### **iv) Multinomial dow\_jones\_index**

```

import numpy as np
import matplotlib.pyplot as plt

```

```

import pandas as pd

data = pd.read_csv('dow_jones_index.csv')

cols=['open','high','low','close','volume','percent_change_price','percent_change_volume_over
      _last_wk','previous_weeks_volume','next_weeks_open','next_weeks_close','percent_change_
      next_weeks_price','days_to_next_dividend','percent_return_next_dividend']

X=data[cols]

y=data['quarter']

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2)

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()

X_train = sc.fit_transform(X_train)

X_test = sc.transform(X_test)

from sklearn.naive_bayes import MultinomialNB

from sklearn.preprocessing import MinMaxScaler

from sklearn.pipeline import Pipeline

p = Pipeline([('Normalizing',MinMaxScaler()),('MultinomialNB',MultinomialNB())])

p.fit(X_train,y_train)

```

**Output:**

```

Pipeline(steps=[('Normalizing', MinMaxScaler()),
                ('MultinomialNB', MultinomialNB())])

```

```
y_pred = p.predict(X_test)
```

```
y_pred
```

**Output:**

```

array([2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2])

```

```
from sklearn.metrics import accuracy_score
```

```
print ("Accuracy : ", accuracy_score(y_test, y_pred))
```

**Output:**

```
Accuracy : 0.5266666666666666
```

```
from sklearn.metrics import classification_report  
print(classification_report(y_test,y_pred))
```

**Output:**

	precision	recall	f1-score	support
1	0.00	0.00	0.00	71
2	0.53	1.00	0.69	79
accuracy			0.53	150
macro avg	0.26	0.50	0.34	150
weighted avg	0.28	0.53	0.36	150

**Result:**

The Naïve Bayes model has been successfully executed and the results are verified.

## **Visualization Tools in Python**

### **1)Aim:**

To explore the following plotting/visualization tools in python:

- Matplotlib
- Diagram
- Mayai
- Seaborn
- VisPy
- PyQtGraph
- Chartify

### **Matplotlib:**

- 1)pyplot() – It is a state based interface to a matplotlib module which provides a MATLAB like interface
- 2)figure() – creates a new figure
- 3)hist() – Plot a histogram
- 4)pie() – Display a figure
- 5)show() – Display a figure
- 6)spy() – Plot the sparsity pattern of a 2D array
- 7)bar() – Plot bar graphs

### **Diagram:**

- 1)aws() – node classes list of aws provider
- 2)azure() – node classes list of azure provider
- 3)gcp() – node classes list of gcp provider
- 4)programming – node classes list of programming provider
- 5)saaS() – node classes list of saas provider
- 6)ibm() – node classes list of ibm provider
- 7)kbs() – node classes list of kbs provider

### **Maya:**

- 1)now() – present datetime
- 2)when() – required datetime
- 3)datetime() – show database info

- 4)slang\_date() – print date
- 5)slang\_time() – print time
- 6)from\_iso8601() – print time from iso 8601
- 7)from\_rfc2822 – print time from rfc 2822

### **Seaborn:**

- 1)set\_theme() – Apply the default theme
- 2)load\_dataset() – Load an dataset
- 3)replot() – visualizes relation b/w variables
- 4)lmplot() – plot linear regression model
- 5)displot() – plot distributed function of data
- 6)catplot() – plot scatter plot
- 7)jointplot() – plot a joint plot

### **VisPy:**

- 1)app() – integrates event system and offers united interface
- 2)gloo() – object oriented interface to openGL
- 3)plot() – high level plotting interfaces
- 4)visual() – graphical abstractions
- 5)transform() – implement 2D/3D transforms
- 6)shaden() – implements shader composition
- 7)scene() – system for underlying high level visualization interfaces

### **PyQtGraph:**

- 1)QtQApp() – build a Q application
- 2)setConfiguration() – set up a configuration
- 3)plot() – plot with colours
- 4)GLGridItem() – create grids
- 5)rotate() – rotate x and y grids
- 6)scale() – scale each grid
- 7)export() – export the file

### **Chartify:**

- 1)\_core\_plot.PlotCategoricalXY() – Plot functions for categorical X & Y

- 2)PlotNumericXY() – plot functions for numerical X,Y
- 3)PlotNumericDensityXY() – plot single density
- 4)PlotDensityXY() – plot density X&Y
- 5)PlotMixedTypeXY() – plot mixed type
- 6)lollipop() – Lollipop chart
- 7)parallel() – Parallel coordinate plot

### **Result:**

Visualization tools have been explored successfully in Python

## **Hadoop Installation**

### **1)Aim:**

To study on Hadoop architecture and its components

### **Hadoop:**

Hadoop is a framework written in Java that utilizes a large cluster of commodity hardware to maintain and store big size data. Hadoop works on MapReduce Programming algorithm that was introduced by Google. The Hadoop architecture mainly consists of four components:

- MapReduce
- HDFS
- YARN
- Common Utilities

### **MapReduce**

It is an algorithm based on the YARN framework. It performs the distributed processing in parallel in a Hadoop cluster. This makes Hadoop work faster.

### **HDFS:**

Hadoop Distributed File System is utilized for storage permission in a Hadoop cluster. It is mainly designed for working on commodity Hardware. HDFS provides fault-tolerance and high availability to the storage layer.

### **YARN:**

Yet Another Resource Navigator (YARN) is a framework on which MapReduce works. YARN performs two operations that are Job Scheduling and Resource Management. Features of YARN:

- Multi-tenancy
- Scalability
- Cluster-utilization
- Compatibility

### **Common Utilities:**

Common Utilities are JAVA files we need for all other components present in a Hadoop cluster. These utilities are used by HDFS, YARN and MapReduce for running the Hadoop cluster.

### **Result:**

Study on Hadoop architecture and its components have been done successfully.

## **2)Aim:**

To install and configure Hadoop

### **Steps for Installation:**

#### **Step-1 sudo apt update**

```
itadmin@PRLAB-22:~$ sudo apt update
[sudo] password for itadmin:
Get:1 http://in.archive.ubuntu.com/ubuntu focal InRelease [114 kB]
Ign:2 https://repo.mongodb.org/apt/ubuntu/focal/mongodb-org/5.0 InRelease
Get:4 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:5 http://in.archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Get:6 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [1,790 kB]
Get:7 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/5.0 Release [4,406 kB]
Get:8 http://in.archive.ubuntu.com/ubuntu focal-updates/main i386 Packages [650 kB]
Get:9 http://in.archive.ubuntu.com/ubuntu focal-updates/main Translation-en [330 kB]
Get:10 http://repo.mongodb.org/ubuntu/focal/mongodb-org/5.0 Release.gpg [801 B]
Get:11 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/5.0 Release.gpg [801 B]
Get:12 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 DEP-11 Metadata [278 kB]
Get:13 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 c-n-f Metadata [15.1 kB]
Get:14 http://in.archive.ubuntu.com/ubuntu focal-updates/restricted amd64 Packages [975 kB]
Get:15 http://in.archive.ubuntu.com/ubuntu focal-updates/restricted Translation-en [139 kB]
Get:16 http://in.archive.ubuntu.com/ubuntu focal-updates/universe amd64 Packages [432 kB]
Get:17 http://in.archive.ubuntu.com/ubuntu focal-updates/universe i386 Packages [679 kB]
Get:18 http://in.archive.ubuntu.com/ubuntu focal-updates/universe amd64 DEP-11 Metadata [390 kB]
Get:19 http://in.archive.ubuntu.com/ubuntu focal-updates/universe amd64 c-n-f Metadata [20.6 kB]
Get:20 http://in.archive.ubuntu.com/ubuntu focal-updates/multiverse amd64 DEP-11 Metadata [944 B]
Get:21 http://in.archive.ubuntu.com/ubuntu focal-backports/main amd64 DEP-11 Metadata [9,588 B]
Get:22 http://security.ubuntu.com/ubuntu focal-security/main amd64 Packages [1,451 kB]
Get:23 http://in.archive.ubuntu.com/ubuntu focal-backports/universe amd64 DEP-11 Metadata [30.8 kB]
Get:24 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/5.0/multiverse arm64 Packages [13.4 kB]
Get:25 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/5.0/multiverse amd64 Packages [15.5 kB]
Get:26 http://security.ubuntu.com/ubuntu focal-security/main Translation-en [249 kB]
Get:27 http://security.ubuntu.com/ubuntu focal-security/main amd64 DEP-11 Metadata [40.6 kB]
Get:28 http://security.ubuntu.com/ubuntu focal-security/main amd64 c-n-f Metadata [10.1 kB]
Get:29 http://security.ubuntu.com/ubuntu focal-security/restricted amd64 Packages [911 kB]
Get:30 http://security.ubuntu.com/ubuntu focal-security/restricted Translation-en [130 kB]
Get:31 http://security.ubuntu.com/ubuntu focal-security/universe amd64 Packages [702 kB]
Get:32 http://security.ubuntu.com/ubuntu focal-security/universe i386 Packages [550 kB]
Get:33 http://security.ubuntu.com/ubuntu focal-security/universe amd64 DEP-11 Metadata [66.3 kB]
Get:34 http://security.ubuntu.com/ubuntu focal-security/universe amd64 c-n-f Metadata [14.4 kB]
Get:35 http://security.ubuntu.com/ubuntu focal-security/multiverse amd64 DEP-11 Metadata [2,464 B]
Fetched 11.2 MB in 4s (2,700 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
123 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

#### **Step-2 : sudo apt install openjdk-8-jdk -y**

```
itadmin@PRLAB-22:~$ sudo apt install openjdk-8-jdk -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
libfwupdplugin1
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
ca-certificates-java fonts-dejavu-extra java-common libatk-wrapper-java libatk-wrapper-java-jni libice-dev libpthread-stubs0-dev libsm-dev libx11-dev libxau-dev libxcb1-dev libxdmcp-dev libxt-dev
openjdk-8-jdk-headless openjdk-8-jre openjdk-8-jre-headless x11proto-core-dev x11proto-dev xorg-sgml-doctools xtrans-dev
Suggested packages:
default-jre libICE1-doc libSM1-doc libX11-3.0.0 libXau1-doc libxcb1-3.0.0 libxdmcp1-3.0.0 libxt1-3.0.0
fonts-wqy-zenhei
The following NEW packages will be installed:
ca-certificates-java fonts-dejavu-extra java-common libatk-wrapper-java libatk-wrapper-java-jni libice-dev libpthread-stubs0-dev libsm-dev libx11-dev libxau-dev libxcb1-dev libxdmcp-dev libxt-dev
openjdk-8-jdk openjdk-8-jdk-headless openjdk-8-jre openjdk-8-jre-headless x11proto-core-dev x11proto-dev xorg-sgml-doctools xtrans-dev
0 upgraded, 21 newly installed, 0 to remove and 123 not upgraded.
Need to get 1,02 MB of additional disk space will be used.
After this operation, 102 MB of additional disk space will be used.
Get:1 http://in.archive.ubuntu.com/ubuntu focal/main amd64 java-common all 0.72 [6,816 B]
Get:2 http://in.archive.ubuntu.com/ubuntu focal-updates/universe amd64 openjdk-8-jre-headless amd64 8u312-b07-0ubuntu1~20.04 [28.2 kB]
Get:3 http://in.archive.ubuntu.com/ubuntu focal/main amd64 ca-certificates-java all 20190405ubuntu1 [12.2 kB]
Get:4 http://in.archive.ubuntu.com/ubuntu focal/main amd64 fonts-dejavu-extra all 2.37-1 [1,953 kB]
Get:5 http://in.archive.ubuntu.com/ubuntu focal/main amd64 libatk-wrapper-java all 0.37.1-1 [53.0 kB]
Get:6 http://in.archive.ubuntu.com/ubuntu focal/main amd64 libatk-wrapper-java-jni all 0.37.1-1 [45.1 kB]
Get:7 http://in.archive.ubuntu.com/ubuntu focal/main amd64 xorg-sgml-doctools all 1:1.11-1 [12.9 kB]
Get:8 http://in.archive.ubuntu.com/ubuntu focal/main amd64 x11proto-dev all 2019.2-1ubuntu1 [594 kB]
Get:9 http://in.archive.ubuntu.com/ubuntu focal/main amd64 x11proto-core-dev all 2019.2-1ubuntu1 [2,620 B]
Get:10 http://in.archive.ubuntu.com/ubuntu focal/main amd64 libice-dev amd64 2:1.0.10-0ubuntu1 [47.8 kB]
Get:11 http://in.archive.ubuntu.com/ubuntu focal/main amd64 libpthread-stubs0-dev amd64 0.4-1 [5,384 B]
Get:12 http://in.archive.ubuntu.com/ubuntu focal/main amd64 libsm-dev amd64 2:1.2.3-1 [17.8 kB]
```

#### **Step-3 java -version; javac -version**

```
itadmin@PRLAB-22:~$ java -version; javac -version
openjdk version "1.8.0_312"
OpenJDK Runtime Environment (build 1.8.0_312-8u312-b07-0ubuntu1~20.04-b07)
OpenJDK 64-Bit Server VM (build 25.312-b07, mixed mode)
javac 1.8.0_312
```

#### **Step-4 sudo apt install openssh-server openssh-client -y**

```
itadmin@PRLAB-22:~$ sudo apt install openssh-server openssh-client -y
Reading package lists... Done
Building dependency trees...
Reading state information... Done
The following package was automatically installed and is no longer required:
  libfwupdplugin1
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  openssh-sftp-server
Suggested packages:
  keychain libpam-ssh monkeysphere ssh-askpass molly-guard
The following packages will be upgraded:
  openssh-client openssh-server openssh-sftp-server
3 upgraded, 0 newly installed, 0 to remove and 120 not upgraded.
Need to get 1,099 kB of archives.
After this operation, 0 B of additional disk space will be used.
Get:1 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 openssh-sftp-server amd64 1:8.2p1-4ubuntu0.5 [51.5 kB]
Get:2 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 openssh-server amd64 1:8.2p1-4ubuntu0.5 [377 kB]
Get:3 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 openssh-client amd64 1:8.2p1-4ubuntu0.5 [671 kB]
Fetched 1,099 kB in 0s (2,271 kB/s)
Preconfiguring packages...
(Reading database ... 213340 files and directories currently installed.)
Preparing to unpack .../openssh-sftp-server_1%3a8.2p1-4ubuntu0.5_amd64.deb ...
Unpacking openssh-sftp-server (1:8.2p1-4ubuntu0.5) over (1:8.2p1-4ubuntu0.4) ...
Preparing to unpack .../openssh-server_1%3a8.2p1-4ubuntu0.5_amd64.deb ...
Unpacking openssh-server (1:8.2p1-4ubuntu0.5) over (1:8.2p1-4ubuntu0.4) ...
Preparing to unpack .../openssh-client_1%3a8.2p1-4ubuntu0.5_amd64.deb ...
Unpacking openssh-client (1:8.2p1-4ubuntu0.5) over (1:8.2p1-4ubuntu0.4) ...
Setting up openssh-client (1:8.2p1-4ubuntu0.5) ...
Setting up openssh-sftp-server (1:8.2p1-4ubuntu0.5) ...
Setting up openssh-server (1:8.2p1-4ubuntu0.5) ...
rescue-ssh.target is a disabled or a static unit, not starting it.
Processing triggers for systemd (249.5-0ubuntu3.15) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for ufw (0.36-6ubuntu1) ...
```

### Step-5 sudo adduser hdoop

```
itadmin@PRLAB-22:~$ sudo adduser hdoop
Adding user `hdoop' ...
Adding new group `hdoop' (1002) ...
Adding new user `hdoop' (1002) with group `hdoop' ...
Creating home directory `/home/hdoop' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for hdoop
Enter the new value, or press ENTER for the default
      Full Name []:
      Room Number []:
      Work Phone []:
      Home Phone []:
      Other []:
Is the information correct? [Y/n] y
```

### Step-6 : su – hdoop

```
itadmin@PRLAB-22:~$ su - hdoop
Password:
hdoop@PRLAB-22:~$
```

### Step-7 ssh-keygen -t rsa -P " -f ~/.ssh/id\_rsa

```
hdoop@PRLAB-22:~$ ssh-keygen -t rsa -P '' -f ~/.ssh/id_rsa
Generating public/private rsa key pair.
Created directory '/home/hdoop/.ssh'.
Your identification has been saved in /home/hdoop/.ssh/id_rsa
Your public key has been saved in /home/hdoop/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:ML0nTmQKjrrLU8f7Y+ogsbBvRvyK8hur4xvz6CTYUA4 hdoop@PRLAB-22
The key's randomart image is:
+--[RSA 3072]----+
| . . . . . . . . |
| . . . . . . . . |
| . . . . . . . . |
| . . . . . . . . |
| . . . . . . . . |
| . . . . . . . . |
| . . . . . . . . |
| . . . . . . . . |
+----[SHA256]----+
```

### Step-8: cat ~/.ssh/id\_rsa.pub >> ~/.ssh/authorized\_keys

### Step-9 chmod 0600 ~/.ssh/authorized\_keys

```
[SNIP]
hdoop@PRLAB-22:~$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
hdoop@PRLAB-22:~$ chmod 0600 ~/.ssh/authorized_keys
```

## Step-10 ssh localhost

```
hadoop@PRLAB-22:~$ ssh localhost
The authenticity of host 'localhost (127.0.0.1)' can't be established.
ECDSA key fingerprint is SHA256:SaTAp+60cPT8NCkg14kfRqJ8pzC8qlN5mV2T5rgLfLE.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.13.0-39-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

118 updates can be applied immediately.
67 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Your Hardware Enablement Stack (HWE) is supported until April 2025.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
```

## Step-11

- Visit the official Apache Hadoop project page, and select the version of Hadoop you want to implement

<https://hadoop.apache.org/releases.html>

Download				
Hadoop is released as source code tarballs with corresponding binary tarballs for convenience. The downloads are distributed via mirror sites and should be checked for tampering using GPG or SHA-512.				
Version	Release date	Source download	Binary download	Release notes
3.2.3	2022 Mar 28	source (checksum signature)	binary (checksum signature)	<a href="#">Announcement</a>
3.3.2	2022 Mar 3	source (checksum signature)	binary (checksum signature) binary-aarch64 (checksum signature)	<a href="#">Announcement</a>
2.10.1	2020 Sep 21	source (checksum signature)	binary (checksum signature)	<a href="#">Announcement</a>

- Click recent version 3.2.3 binary

We suggest the following site for your download:  
<https://dlcdn.apache.org/hadoop/common/hadoop-3.2.3/hadoop-3.2.3.tar.gz>  
Alternate download locations are suggested below.  
It is essential that you verify the integrity of the downloaded file using the PGP signature ([.asc](#) file) or a hash ([.md5](#) or [.sha](#) file).

- Click the link to download locally in a machine  
Or type
- Wget <https://dlcdn.apache.org/hadoop/common/hadoop-3.2.3/hadoop-3.2.3.tar.gz>

Once the download is complete, extract the files to initiate the Hadoop installation:

-x : Extract the archive , -v : Displays Verbose Information, -z : zip, tells tar command that creates tar file using gzip , -f : creates archive with given filename

- tar xvzf hadoop-3.2.3.tar.gz

```
hadoop-3.2.3/share/doc/hadoop/hadoop-distcp/css/maven-base.css
hadoop-3.2.3/share/doc/hadoop/hadoop-distcp/css/maven-theme.css
hadoop-3.2.3/share/doc/hadoop/hadoop-distcp/css/site.css
hadoop-3.2.3/share/doc/hadoop/hadoop-distcp/css/print.css
hadoop-3.2.3/share/doc/hadoop/hadoop-distcp/dependency-analysis.html
hadoop-3.2.3/lib/
hadoop-3.2.3/lib/native/
hadoop-3.2.3/lib/native/libhdfspp.so.0.1.0
hadoop-3.2.3/lib/native/libhadooppipes.a
hadoop-3.2.3/lib/native/libhdfs.so.0.0.0
hadoop-3.2.3/lib/native/libhadooputils.a
hadoop-3.2.3/lib/native/libhadoop.so
hadoop-3.2.3/lib/native/libhdfspp.so
hadoop-3.2.3/lib/native/libhadoop.so.1.0.0
hadoop-3.2.3/lib/native/libnativetask.a
hadoop-3.2.3/lib/native/examples/
hadoop-3.2.3/lib/native/examples/wordcount-part
hadoop-3.2.3/lib/native/examples/wordcount-nopipe
hadoop-3.2.3/lib/native/examples/pipes-sort
hadoop-3.2.3/lib/native/examples/wordcount-simple
hadoop-3.2.3/lib/native/libhdfs.a
hadoop-3.2.3/lib/native/libhdfs.so
hadoop-3.2.3/lib/native/libhadoop.a
hadoop-3.2.3/lib/native/libnativetask.so.1.0.0
hadoop-3.2.3/lib/native/libnativetask.so
hadoop-3.2.3/lib/native/libhdfspp.a
hadoop-3.2.3/LICENSE.txt
```

Step-12 mv hadoop-3.2.3 hadoop

Step-13 dirname \$(dirname \$(readlink -f \$(which java)))

Step-14 Vi .bashrc

Define the Hadoop environment variables by adding the following content to the end of the file:

### #Hadoop Related Options

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export HADOOP_HOME=/home/hadoop/hadoop
export HADOOP_INSTALL=$HADOOP_HOME
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib/native"
```

Step-15 source ~/.bashrc

Step-16 Vi \$HADOOP\_HOME/etc/hadoop/hadoop-env.sh

Uncomment the \$JAVA\_HOME variable (i.e., remove the # sign) and add the full path to the OpenJDK installation on your system.

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
```

```
hadoop@PRLAB-22:~$ ls
hadoop  hadoop-3.2.3.tar.gz  test.txt
hadoop@PRLAB-22:~$ cd hadoop
hadoop@PRLAB-22:~/hadoop$ ls
bin  include  libexec  NOTICE.txt  sbin
etc  lib  LICENSE.txt  README.txt  share
hadoop@PRLAB-22:~/hadoop$ cd etc
hadoop@PRLAB-22:~/hadoop/etc$ ls
hadoop
hadoop@PRLAB-22:~/hadoop/etc$ cd hadoop
hadoop@PRLAB-22:~/hadoop/etc/hadoop$ ls
capacity-scheduler.xml          kms-log4j.properties
configuration.xsl                kms-site.xml
container-executor.cfg           log4j.properties
core-site.xml                   mapred-env.cmd
hadoop-env.cmd                 mapred-env.sh
hadoop-env.sh                  mapred-queues.xml.template
hadoop-metrics2.properties      mapred-site.xml
hadoop-policy.xml               shellprofile.d
hadoop-user-functions.sh.example ssl-client.xml.example
hdfs-site.xml                   ssl-server.xml.example
httpfs-env.sh                  user_ec_policies.xml.template
httpfs-log4j.properties         workers
httpfs-signature.secret        yarn-env.cmd
httpfs-site.xml                yarn-env.sh
kms-acls.xml                   yarnservice-log4j.properties
kms-env.sh                      yarn-site.xml
hadoop@PRLAB-22:~/hadoop/etc/hadoop$ vi hadoop-env.sh
hadoop@PRLAB-22:~/hadoop/etc/hadoop$
```

Step 17- configure Apache Hadoop on the Ubuntu system. For this, the step is to create two directories: datanode and namenode, inside the home directory of Hadoop:

```
mkdir -p ~/dfsdata/namenode
```

```
mkdir -p ~/dfsdata/datanode
```

Step-18 Vi \$HADOOP\_HOME/etc/hadoop/core-site.xml

Add the following configuration to override the default values for the temporary directory and add your HDFS URL to replace the default local file system setting:

```
<property>
  <name>hadoop.tmp.dir</name>
  <value>/home/hadoop/tmpdata</value>
</property>
<property>
  <name>fs.default.name</name>
  <value>hdfs://127.0.0.1:9000</value>
</property>
```

Step-19 Vi \$HADOOP\_HOME/etc/hadoop/hdfs-site.xml

Add the following configuration to the file and, if needed, adjust the NameNode and DataNode directories to your custom locations:

```
<configuration>
<property>
<name>dfs.data.dir</name>
```

```

<value>/home/hdoop/dfsdata/namenode</value>
</property>
<property>
<name>dfs.data.dir</name>
<value>/home/hdoop/dfsdata/datanode</value>
</property>
<property>
<name>dfs.replication</name>
<value>1</value>
</property>
</configuration>

```

```

<configuration>
<property>
<name>dfs.data.dir</name>
<value>/home/hdoop/dfsdata/namenode</value>
</property>
<property>
<name>dfs.data.dir</name>
<value>/home/hdoop/dfsdata/datanode</value>
</property>
<property>
<name>dfs.replication</name>
<value>1</value>
</property></configuration>

```

Step 20 Vi \$HADOOP\_HOME/etc/hadoop/mapred-site.xml

Add the following configuration to change the default MapReduce framework name value to yarn:

```

<configuration>
<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>
</configuration>

```

```

<configuration>
<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property> </configuration>

```

Step 21 Vi \$HADOOP\_HOME/etc/hadoop/yarn-site.xml

Append the following configuration to the file:

```

<configuration>
  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
  <property>
    <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
    <value>org.apache.hadoop.mapred.ShuffleHandler</value>
  </property>
  <property>
    <name>yarn.resourcemanager.hostname</name>
    <value>127.0.0.1</value>
  </property>
  <property>
    <name>yarn.acl.enable</name>
    <value>0</value>
  </property>
  <property>
    <name>yarn.nodemanager.env-whitelist</name>
    <value>JAVA_HOME,HADOOP_COMMON_HOME,HADOOP_HDFS_HOME,HADOOP_CONF_DIR,CLASSPATH_PERPEND_DISTCACHE,HADOOP_YARN_HOME,HADOOP_MAPRED_HOME</value>
  </property>
</configuration>

```

```

<property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
</property>
<property>
  <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
  <value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>
<property>
  <name>yarn.resourcemanager.hostname</name>
  <value>127.0.0.1</value>
</property>
<property>
  <name>yarn.acl.enable</name>
  <value>0</value>
</property>
<property>
  <name>yarn.nodemanager.env-whitelist</name>
  <value>JAVA_HOME,HADOOP_COMMON_HOME,HADOOP_HDFS_HOME,HADOOP_CONF_DIR,CLASSPATH_PERPEND_DISTCACHE,HADOOP_YARN_HOME,HADOOP_MAPRED_HOME</value>
</property></configuration>

```

Step 22 hdfs namenode -format

Step-23 ./start-dfs.sh

```
hadoop@PRLAB-22:~$ cd hadoop  
hadoop@PRLAB-22:~/hadoop$ ls  
bin etc include lib libexec LICENSE.txt logs NOTICE.txt README.txt sbin share  
hadoop@PRLAB-22:~/hadoop$ cd sbin  
hadoop@PRLAB-22:~/hadoop/sbin$ ./start-dfs.sh
```

```
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [PRLAB-22]
```

Step-24 ./start-yarn.sh

```
hadoop@PRLAB-22:~/hadoop/sbin$ ./start-yarn.sh
Starting resourcemanager
Starting nodemanagers
```

## Step 25 ips

```
hadoop@PRLAB-22:~/hadoop/sbin$ jps  
36208 NodeManager  
36627 Jps  
35547 DataNode  
36059 ResourceManager  
35372 NameNode  
35789 SecondaryNameNode
```

## Step 26 <http://localhost:9870>

The screenshot shows the 'Overview' section of the Namenode information page. It displays the following details:

Started:	Fri May 20 03:13:28 +0530 2022
Version:	3.2.3, rabe5358143720085498613d399be3bbf01e0f131
Compiled:	Sun Mar 20 06:48:00 +0530 2022 by ubuntu from branch-3.2.3
Cluster ID:	CID-2d693310-a1b5-41df-aa5e-6b2e0fa5e167
Block Pool ID:	BP-1597927012-127.0.1.1-1652996439092

**Summary**

Security is off.  
Safemode is off.  
1 files and directories, 0 blocks (0 replicated blocks, 0 erasure coded block groups) = 1 total filesystem object(s).  
Heap Memory used 127.48 MB of 357 MB Heap Memory. Max Heap Memory is 3.42 GB.  
Non Heap Memory used 49.3 MB of 50.58 MB Committed Non Heap Memory. Max Non Heap Memory is <unbounded>.

Configured Capacity:	186.74 GB
Configured Remote Capacity:	0 B
DFS Used:	24 KB (0%)
Non DFS Used:	9.54 GB
DFS Remaining:	167.65 GB (89.78%)
Block Pool Used:	24 KB (0%)

## Step-27 <http://localhost:9864>

The screenshot shows the 'DataNode on PRLAB-22:9866' section of the DataNode Information page. It displays the following details:

Cluster ID:	CID-2d693310-a1b5-41df-aa5e-6b2e0fa5e167
Version:	3.2.3, rabe5358143720085498613d399be3bbf01e0f131

**Block Pools**

Namenode Address	Block Pool ID	Actor State	Last Heartbeat	Last Block Report	Last Block Report Size (Max Size)
localhost:9000	BP-1597927012-127.0.1.1-1652996439092	RUNNING	1s	3 minutes	0 B (64 MB)

**Volume Information**

Directory	StorageType	Capacity Used	Capacity Left	Capacity Reserved	Reserved Space for Replicas	Blocks
/home/hadoop/dfsdata/datanode	DISK	24 KB	167.65 GB	0 B	0 B	0

Hadoop, 2022.

## Step 28 <http://localhost:8088>

A screenshot of a web browser window titled "All Applications" from the Hadoop cluster at localhost:8088. The page displays various metrics and tables related to the cluster's state. On the left, there is a sidebar with links like "Cluster Metrics", "Cluster Nodes Metrics", "Scheduler Metrics", and "Tools". The main content area shows "Cluster Metrics" with values: Apps Submitted: 0, Apps Pending: 0, Apps Running: 0, Apps Completed: 0, Containers Running: 0, Used Resources: &lt;memory:0 B, vCores:0&gt;, Total Resources: &lt;memory:8 GB, vCores:4&gt;. Below this is a table for "Scheduler Metrics" with columns: Scheduler Type, Scheduling Resource Type, Minimum Allocation, and Maximum Allocation. The table shows one entry for Capacity Scheduler with values: [memory\_mb (unit=Mi), vcores] &lt;memory:1024, vCores:1&gt;. A note below the table says "No data available in table". At the bottom, it says "Showing 0 to 0 of 0 entries".

## Step 29 To Stop namenode , datanode and yarn

```
hadoop@PRLAB-22:~/hadoop/sbin$ ./stop-yarn.sh
Stopping nodemanagers
Stopping resourcemanager
hadoop@PRLAB-22:~/hadoop/sbin$ ./stop-dfs.sh
Stopping namenodes on [localhost]
Stopping datanodes
Stopping secondary namenodes [PRLAB-22]
hadoop@PRLAB-22:~/hadoop/sbin$ jps
37827 Jps
```

## Result:

Hadoop has been successfully installed in the system.

## FILE MANAGEMENT IN HADOOP

### **1)Aim:**

To implement the following task in Hadoop: a.Adding Directories. b.Adding Files. c.Retrieving files and its content.d.Deleting files.

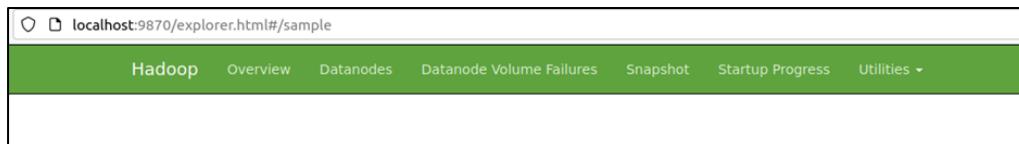
### **Adding Directory**

hadoop fs -mkdir /sample

```
hadoop@PRLAB-22:~$ hadoop fs -mkdir /sample
```

Now login in web browser:

<http://localhost:9870>



Under Utilities → click (Browse the file system option)



### **Adding file:**

In terminal create new file

vi eg.txt

```
hadoop@PRLAB-22:~$ vi eg.txt
```

Move the file to the newly created directory

hadoop fs -put eg.txt /sample

```
hadoop@PRLAB-22:~$ hadoop fs -put eg.txt /sample
```

File	Size	Last Modified	Replication	Block Size	Name
eg.txt	18 B	May 21 02:46	1	128 MB	eg.txt

## Retrieve a file:

hadoop fs -cat /sample/eg.txt

```
hadoop@PRLAB-22:~$ hadoop fs -cat /sample/eg.txt
welcome to hadoop
```

## Delete a file:

Existing files

File	Size	Last Modified	Replication	Block Size	Name
eg.txt	18 B	May 21 02:46	1	128 MB	eg.txt
hsample.txt	18 B	May 21 02:54	1	128 MB	hsample.txt

hadoop fs -rm /sample/hsample.txt

```
hadoop@PRLAB-22:~$ hadoop fs -rm /sample/hsample.txt
Deleted /sample/hsample.txt
```

After deletion

The screenshot shows a web-based Hadoop file explorer interface. At the top, there is a navigation bar with links for Overview, Datanodes, Datanode Volume Failures, Snapshot, Startup Progress, and Utilities. Below the navigation bar, the title "Browse Directory" is displayed. A search bar with the path "/sample" and a "Go!" button is present. There is also a search field labeled "Search:" with a magnifying glass icon. The main area displays a table of file entries:

	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
<input type="checkbox"/>	-rw-r--r--	hdoop	supergroup	18 B	May 21 02:46	1	128 MB	eg.txt

Below the table, it says "Showing 1 to 1 of 1 entries". At the bottom right, there are buttons for "Previous", a page number "1", and "Next".

## Result:

File Management in Hadoop has been done successfully and the results are verified.

## MongoDB-Data Replication

### 1)Aim:

To perform data replication using Mongo DB

Steps:

Step 1: Creating a node in port 27021

Step 2:

```
mkdir -p $HOME/mongo/data/db01
```

```
mongod --replSet dbrs --port 27021 --dbpath $HOME/mongo/data/db01
```

Step 3: Open another terminal

```
mkdir -p $HOME/mongo/data/db02
```

```
mongod --replSet dbrs --port 27022 --dbpath $HOME/mongo/data/db02
```

### Output:

```
(base) student@PRLAB-32:~$ mkdir -p $HOME/mongo/data/db02
(base) student@PRLAB-32:~$ mongod --replSet dbrs --port 27022 --dbpath $HOME/mongo/data/db02
{"t": {"$date": "2022-05-24T13:29:31.785+05:30"}, "s": "I", "c": "CONTROL", "id": 23285, "ctx": "thread1", "msg": "Automatically disabling TLS 1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols 'none'"}
{"t": {"$date": "2022-05-24T13:29:31.786+05:30"}, "s": "I", "c": "NETWORK", "id": 4915701, "ctx": "thread1", "msg": "Initialized wire specification", "attr": {"spec": {"incomingExternalClient": {"minWireVersion": 0, "maxWireVersion": 13}, "incomingInternalClient": {"minWireVersion": 0, "maxWireVersion": 13}, "outgoing": {"minWireVersion": 0, "maxWireVersion": 13}, "isInternalClient": true}}}
{"t": {"$date": "2022-05-24T13:29:31.787+05:30"}, "s": "W", "c": "ASIO", "id": 22
```

Step 4: Third Terminal

```
mkdir -p $HOME/mongo/data/db03
```

```
mongod --replSet dbrs --port 27023 --dbpath $HOME/mongo/data/db03
```

Step 5: Connect to any one of the mongo daemon

```
mongo --port 27021
```

## **Output:**

```
ceive and display
    metrics about your deployment (disk utilization, CPU, operation statisti
cs, etc).

    The monitoring data will be available on a MongoDB website with a unique
URL accessible to you
    and anyone you share the URL with. MongoDB may use this information to m
ake product
    improvements and to suggest MongoDB products and deployment options to y
ou.

    To enable free monitoring, run the following command: db.enableFreeMonit
oring()
    To permanently disable this reminder, run the following command: db.disa
bleFreeMonitoring()
...
db:PRIMARY> 
```

Step 6: rsconf = {

```
_id: "dbrs",
members: [
{
    _id: 0,
    host: "127.0.0.1:27021",
    priority: 3
},
{
    _id: 1,
    host: "127.0.0.1:27022",
    priority: 1,
},
{
    _id: 2,
    host: "127.0.0.1:27023",
    priority: 2
}
]
```

```
};

db.getMongo().setReadPref('secondary')
```

**Output:**

```
]
}
> rs.initiate(rsconf);
{ "ok" : 1 }
```

**Result:**

Data replication using MongoDB has been successfully done and the results are verified.

## Classification-SVM

### 1)Aim:

To build a SVM classification model and perform the following a. Exploratory data analysis b. Explore missing values in variables c. Handle outliers d .Check the distribution of variables e. Declare feature vector and target variable f. Split data into separate training and test set g. Perform Feature Scaling h. Run SVM with default Hyperparameters i. Run SVM with RBF kernel j. Run SVM with Linear kernel k. Run SVM with polynomial kernel l. Run SVM with sigmoid kernel m. Compare the train-set and test-set accuracy n. Check for overfitting and underfitting o. Summarize the performance of model using confusion matrix p. Evaluate the model performance with classification report q. Analyze the model performance visually.

### Code:

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

data = pd.read_csv('dow_jones_index.csv')
data['high'] = data['high'].str.replace('$', '')
data['open'] = data['open'].str.replace('$', '')
data['low'] = data['low'].str.replace('$', '')
data['close'] = data['close'].str.replace('$', '')
data['next_weeks_open'] = data['next_weeks_open'].str.replace('$', '')
data['next_weeks_close'] = data['next_weeks_close'].str.replace('$', '')

data['high']=pd.to_numeric(data['high'])
data['open']=pd.to_numeric(data['open'])
data['low']=pd.to_numeric(data['low'])
data['close']=pd.to_numeric(data['close'])
data['next_weeks_open']=pd.to_numeric(data['next_weeks_open'])
data['next_weeks_close']=pd.to_numeric(data['next_weeks_close'])
```

```
data['percent_change_volume_over_last_wk'].fillna(value=0,inplace=True)  
data['previous_weeks_volume'].fillna(value=0,inplace=True)  
data.shape
```

**Output:**

```
data.shape
```

```
(750, 16)
```

```
data.info()
```

**Output:**

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 750 entries, 0 to 749  
Data columns (total 16 columns):  
 #   Column           Non-Null Count  Dtype     
 ---  --  
 0   quarter          750 non-null    int64    
 1   stock            750 non-null    object    
 2   date             750 non-null    object    
 3   open              750 non-null    float64  
 4   high              750 non-null    float64  
 5   low               750 non-null    float64  
 6   close              750 non-null    float64  
 7   volume             750 non-null    int64    
 8   percent_change_price  750 non-null    float64  
 9   percent_change_volume_over_last_wk 720 non-null    float64  
 10  previous_weeks_volume      720 non-null    float64  
 11  next_weeks_open          750 non-null    float64  
 12  next_weeks_close         750 non-null    float64  
 13  percent_change_next_weeks_price 750 non-null    float64  
 14  days_to_next_dividend    750 non-null    int64    
 15  percent_return_next_dividend 750 non-null    float64  
dtypes: float64(11), int64(3), object(2)  
memory usage: 93.9+ KB
```

```
data.describe()
```

**Output:**

	quarter	open	high	low	close	volume	percent_change_price	percent_change_volume_over_last_wk	p
count	750.000000	750.000000	750.000000	750.000000	750.000000	7.500000e+02	750.000000	720.000000	
mean	1.520000	53.651840	54.669987	52.640160	53.729267	1.175478e+08	0.050262	5.593627	
std	0.499933	32.638852	33.215994	32.119277	32.788787	1.584381e+08	2.517809	40.543478	
min	1.000000	10.590000	10.940000	10.400000	10.520000	9.718851e+06	-15.422900	-61.433175	
25%	1.000000	29.830000	30.627500	28.720000	30.365000	3.086624e+07	-1.288053	-19.804284	
50%	2.000000	45.970000	46.885000	44.800000	45.930000	5.306088e+07	0.000000	0.512586	
75%	2.000000	72.715000	74.287500	71.037500	72.667500	1.327218e+08	1.650888	21.800622	
max	2.000000	172.110000	173.540000	167.820000	170.580000	1.453439e+09	9.882230	327.408924	

percent_change_volume_over_last_wk	previous_weeks_volume	next_weeks_open	next_weeks_close	percent_change_next_weeks_price	days_to_next_dividend
720.000000	7.200000e+02	750.000000	750.000000	750.000000	750.000000
5.593627	1.173876e+08	53.702440	53.889080	0.238468	52.525333
40.543478	1.592322e+08	32.778111	33.016677	2.679538	46.335098
-61.433175	9.718851e+06	10.520000	10.520000	-15.422900	0.000000
-19.804284	3.067832e+07	30.315000	30.462500	-1.222068	24.000000
0.512586	5.294556e+07	46.015000	46.125000	0.101193	47.000000
21.800622	1.333230e+08	72.715000	72.915000	1.845562	69.000000
327.408924	1.453439e+09	172.110000	174.540000	9.882230	336.000000

```
cols=['open','high','low','close','volume','next_weeks_open','next_weeks_close','percent_change_next_weeks_price']
```

```
X=data[cols]
```

```
y=data['quarter']
```

### # split X and y into training and testing sets

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0)
```

### # check the shape of X\_train and X\_test

```
X_train.shape, X_test.shape
```

### Output:

```
# check the shape of X_train and X_test
X_train.shape, X_test.shape
((600, 8), (150, 8))
```

```
cols = X_train.columns
```

```
from sklearn.preprocessing import StandardScaler
```

```
scaler = StandardScaler()
```

```
X_train = scaler.fit_transform(X_train)
```

```
X_test = scaler.transform(X_test)
```

```
X_train = pd.DataFrame(X_train, columns=[cols])
```

```
X_test = pd.DataFrame(X_test, columns=[cols])
```

```
X_train.describe()
```

## **Output:**

	open	high	low	close	volume	next_weeks_open	next_weeks_close	percent_change_next_weeks_price
<b>count</b>	6.000000e+02	6.000000e+02	6.000000e+02	6.000000e+02	6.000000e+02	6.000000e+02	6.000000e+02	6.000000e+02
<b>mean</b>	-2.035409e-18	-5.939693e-17	-2.096471e-16	8.252658e-17	-3.044873e-17	1.789309e-16	2.942091e-17	4.847974e-17
<b>std</b>	1.000834e+00	1.000834e+00	1.000834e+00	1.000834e+00	1.000834e+00	1.000834e+00	1.000834e+00	1.000834e+00
<b>min</b>	-1.327585e+00	-1.322818e+00	-1.323036e+00	-1.324817e+00	-6.656439e-01	-1.323918e+00	-1.321378e+00	-4.059979e+00
<b>25%</b>	-7.113305e-01	-7.140684e-01	-7.073728e-01	-7.108043e-01	-5.316059e-01	-7.101924e-01	-7.060590e-01	-5.750435e-01
<b>50%</b>	-2.460951e-01	-2.421160e-01	-2.535420e-01	-2.500281e-01	-3.982804e-01	-2.441872e-01	-2.396332e-01	-2.436433e-02
<b>75%</b>	5.751698e-01	5.639386e-01	5.608326e-01	5.671071e-01	9.044814e-02	5.701036e-01	5.695192e-01	6.376863e-01
<b>max</b>	3.620221e+00	3.565823e+00	3.566168e+00	3.551337e+00	8.305524e+00	3.597767e+00	3.642331e+00	3.616121e+00

## **# import SVC classifier**

```
from sklearn.svm import SVC  
from sklearn.metrics import accuracy_score  
# instantiate classifier with default hyperparameters  
svc=SVC()  
svc.fit(X_train,y_train)  
y_pred=svc.predict(X_test)  
print('Model accuracy score with default hyperparameters: {0:0.4f}'.format(accuracy_score(y_test, y_pred)))
```

## **Output:**

```
Model accuracy score with default hyperparameters: 0.4733
```

## **# instantiate classifier with rbf kernel and C=100**

```
svc=SVC(C=100.0)  
svc.fit(X_train,y_train)  
y_pred=svc.predict(X_test)  
print('Model accuracy score with rbf kernel and C=100.0 : {0:0.4f}'.format(accuracy_score(y_test, y_pred)))
```

## **Output:**

```
Model accuracy score with rbf kernel and C=100.0 : 0.5333
```

## **# instantiate classifier with rbf kernel and C=1000**

```
svc=SVC(C=1000.0)  
svc.fit(X_train,y_train)  
y_pred=svc.predict(X_test)
```

```
print('Model accuracy score with rbf kernel and C=1000.0 : {0:0.4f}'.format(accuracy_score(y_test, y_pred)))
```

**Output:**

```
Model accuracy score with rbf kernel and C=1000.0 : 0.5733
```

**# instantiate classifier with linear kernel and C=1.0**

```
linear_svc=SVC(kernel='linear', C=1.0)  
linear_svc.fit(X_train,y_train)  
y_pred_test=linear_svc.predict(X_test)  
print('Model accuracy score with linear kernel and C=1.0 : {0:0.4f}'.format(accuracy_score(y_test, y_pred_test)))
```

**Output:**

```
Model accuracy score with linear kernel and C=1.0 : 0.5333
```

**# instantiate classifier with linear kernel and C=100.0**

```
linear_svc100=SVC(kernel='linear', C=100.0)  
linear_svc100.fit(X_train, y_train)  
y_pred=linear_svc100.predict(X_test)  
print('Model accuracy score with linear kernel and C=100.0 : {0:0.4f}'.format(accuracy_score(y_test, y_pred)))
```

**Output:**

```
Model accuracy score with linear kernel and C=100.0 : 0.5200
```

**# instantiate classifier with linear kernel and C=1000.0**

```
linear_svc1000=SVC(kernel='linear', C=1000.0)  
linear_svc1000.fit(X_train, y_train)  
y_pred=linear_svc1000.predict(X_test)  
print('Model accuracy score with linear kernel and C=1000.0 : {0:0.4f}'.format(accuracy_score(y_test, y_pred)))
```

**Output:**

```
Model accuracy score with linear kernel and C=1000.0 : 0.5333
```

```
# instantiate classifier with polynomial kernel and C=1.0  
poly_svc=SVC(kernel='poly', C=1.0)  
poly_svc.fit(X_train,y_train)  
y_pred_test=poly_svc.predict(X_test)  
print('Model accuracy score with polynomial kernel and C=1.0 : {0:0.4f}'.  
format(accuracy_score(y_test, y_pred_test)))
```

**Output:**

```
Model accuracy score with polynomial kernel and C=1.0 : 0.5200
```

```
# instantiate classifier with polynomial kernel and C=100.0
```

```
poly_svc=SVC(kernel='poly', C=100.0)  
poly_svc.fit(X_train,y_train)  
y_pred_test=poly_svc.predict(X_test)  
print('Model accuracy score with polynomial kernel and C=100.0 : {0:0.4f}'.  
format(accuracy_score(y_test, y_pred_test)))
```

**Output:**

```
Model accuracy score with polynomial kernel and C=100.0 : 0.5333
```

```
# instantiate classifier with sigmoid kernel and C=1.0
```

```
sig_svc=SVC(kernel='sigmoid', C=1.0)  
sig_svc.fit(X_train,y_train)  
y_pred_test=sig_svc.predict(X_test)  
print('Model accuracy score with sigmoid kernel and C=1.0 : {0:0.4f}'.  
format(accuracy_score(y_test, y_pred_test)))
```

**Output:**

```
Model accuracy score with sigmoid kernel and C=1.0 : 0.4600
```

```
# instantiate classifier with sigmoid kernel and C=100.0
```

```
sig_svc=SVC(kernel='sigmoid', C=100.0)  
sig_svc.fit(X_train,y_train)  
y_pred_test=sig_svc.predict(X_test)  
print('Model accuracy score with sigmoid kernel and C=100.0 : {0:0.4f}'.  
format(accuracy_score(y_test, y_pred_test)))
```

**Output:**

```
Model accuracy score with sigmoid kernel and C=100.0 : 0.4733
```

```
y_pred_train = linear_svc.predict(X_train)
```

```
print('Training-set accuracy score: {:.4f}'.format(accuracy_score(y_train, y_pred_train)))
```

**Output:**

```
Training-set accuracy score: 0.5283
```

```
# print the scores on training and test set
```

```
print('Training set score: {:.4f}'.format(linear_svc.score(X_train, y_train)))
```

```
print('Test set score: {:.4f}'.format(linear_svc.score(X_test, y_test)))
```

**Output:**

```
Training set score: 0.5283
Test set score: 0.5200
```

```
# check class distribution in test set
```

```
y_test.value_counts()
```

**Output:**

```
2    80
1    70
Name: quarter, dtype: int64
```

```
# check null accuracy score
```

```
null_accuracy = (80/(80+70))
```

```
print('Null accuracy score: {:.4f}'.format(null_accuracy))
```

**Output:**

```
Null accuracy score: 0.5333
```

```
from sklearn.metrics import confusion_matrix
```

```
cm = confusion_matrix(y_test, y_pred_test)
```

```
print('Confusion matrix\n\n', cm)
```

```
print("\nTrue Positives(TP) = ", cm[0,0])
print("\nTrue Negatives(TN) = ", cm[1,1])
print("\nFalse Positives(FP) = ", cm[0,1])
print("\nFalse Negatives(FN) = ", cm[1,0])
```

**Output:**

```
Confusion matrix

[[30 40]
 [39 41]]

True Positives(TP) = 30

True Negatives(TN) = 41

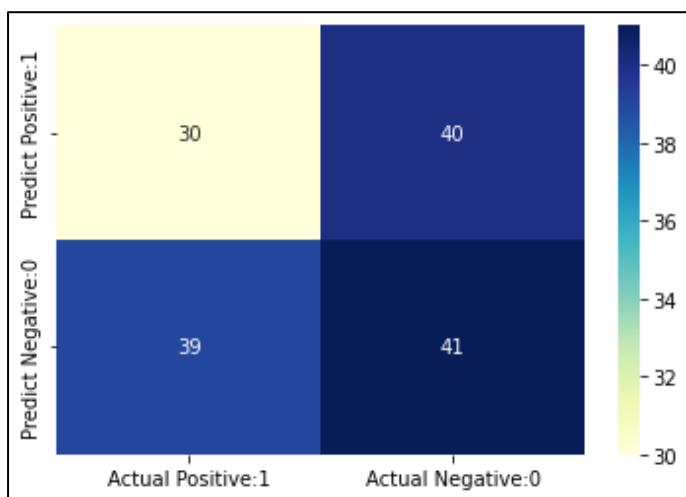
False Positives(FP) = 40

False Negatives(FN) = 39
```

```
cm_matrix = pd.DataFrame(data=cm, columns=['Actual Positive:1', 'Actual Negative:0'],
                           index=['Predict Positive:1', 'Predict Negative:0'])

sns.heatmap(cm_matrix, annot=True, fmt='d', cmap='YlGnBu')
```

**Output:**



```
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred_test))
```

**Output:**

	precision	recall	f1-score	support
1	0.43	0.43	0.43	70
2	0.51	0.51	0.51	80
accuracy			0.47	150
macro avg	0.47	0.47	0.47	150
weighted avg	0.47	0.47	0.47	150

TP = cm[0,0]

TN = cm[1,1]

FP = cm[0,1]

FN = cm[1,0]

classification\_accuracy = (TP + TN) / float(TP + TN + FP + FN)

print('Classification accuracy : {0:0.4f}'.format(classification\_accuracy))

**Output:**

**Classification accuracy : 0.4733**

classification\_error = (FP + FN) / float(TP + TN + FP + FN)

print('Classification error : {0:0.4f}'.format(classification\_error))

**Output:**

**Classification error : 0.5267**

precision = TP / float(TP + FP)

print('Precision : {0:0.4f}'.format(precision))

**Output:**

**Precision : 0.4286**

recall = TP / float(TP + FN)

print('Recall or Sensitivity : {0:0.4f}'.format(recall))

**Output:**

**Recall or Sensitivity : 0.4348**

```
true_positive_rate = TP / float(TP + FN)
print('True Positive Rate : {:.4f}'.format(true_positive_rate))
```

**Output:**

```
True Positive Rate : 0.4348
```

```
false_positive_rate = FP / float(FP + TN)
print('False Positive Rate : {:.4f}'.format(false_positive_rate))
```

**Output:**

```
False Positive Rate : 0.4938
```

```
specificity = TN / (TN + FP)
print('Specificity : {:.4f}'.format(specificity))
```

**Output:**

```
Specificity : 0.5062
```

```
# calculate cross-validated ROC AUC
from sklearn.model_selection import cross_val_score
Cross_validated_ROC_AUC = cross_val_score(linear_svc, X_train, y_train, cv=10,
scoring='roc_auc').mean()
print('Cross validated ROC AUC : {:.4f}'.format(Cross_validated_ROC_AUC))
```

**Output:**

```
Cross validated ROC AUC : 0.5186
```

```
import numpy as np
arr = np.array(y_test)
indices_one = arr == 1
indices_zero = arr == 2
arr[indices_one] = 0 # replacing 1s with 0s
arr[indices_zero] = 1 # replacing 2s with 1s
arr1=arr
```

```

import numpy as np
arr = np.array(y_pred_test)
indices_one = arr == 1
indices_zero = arr == 2
arr[indices_one] = 0 # replacing 1s with 0s
arr[indices_zero] = 1 # replacing 2s with 1s
arr2=arr

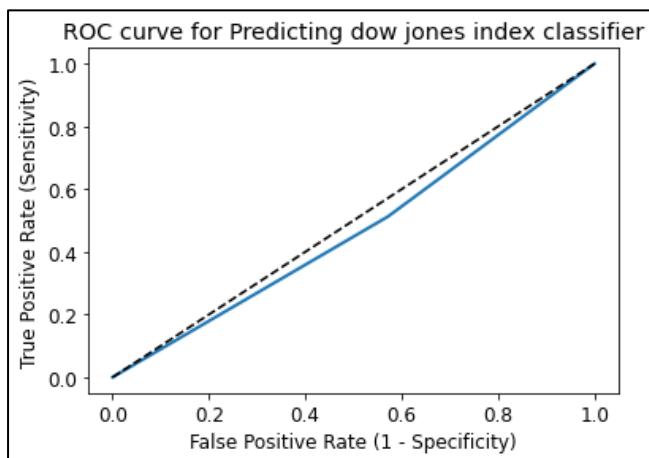
```

```

from sklearn.metrics import roc_curve
fpr, tpr, thresholds = roc_curve(arr1, arr2)
plt.figure(figsize=(6,4))
plt.plot(fpr, tpr, linewidth=2)
plt.plot([0,1], [0,1], 'k--' )
plt.rcParams['font.size'] = 12
plt.title('ROC curve for Predicting dow jones index classifier')
plt.xlabel('False Positive Rate (1 - Specificity)')
plt.ylabel('True Positive Rate (Sensitivity)')
plt.show()

```

### **Output:**



### **Result:**

SVM model has been successfully completed and the results are verified.

## **Hive – Installation**

### **1)Aim:**

A study on Hive architecture and its working

### **Hive:**

Hive is a Hadoop data warehouse infrastructure solution that allows you to process structured data. It sits atop Hadoop to summarise Big Data and facilitate searching and analysis. Hive is a declarative SQL-based programming language that is mostly used for data analysis and report generation. Hive is a server-side cluster management system.

Hive makes schema flexibility and evolution, as well as data summarization, data querying, and analysis, more easier. We may create two sorts of tables in Hive: partitioned and bucketed, which allow us to process data stored in HDFS while also improving performance.

### **HIVE Architecture:**

**METASTORE** – It is used to store metadata of tables schema, time of creation, location, etc. It also provides metadata partition to help the driver to keep the track of the progress of various datasets distributed over the cluster.

**DRIVER** – It acts as a controller. The driver starts the execution of the statement by creating sessions and monitors the life cycle and progress of execution.

**COMPILER** – It is used to compile a Hive query, which converts the query to an execution plan.

**OPTIMIZER** – It optimizes and performs transformations on an execution plan to get an optimized Directed Acyclic Graph abbreviated as DAG.

**EXECUTOR** – It executes tasks after compilation and optimization have been done. It interacts with the job tracker of Hadoop to schedule tasks to be run.

### **Result:**

Study on HIVE has been successfully executed and the results are verified.

## **2)Aim:**

To install HIVE above Hadoop in the system

## **Steps to installation:**

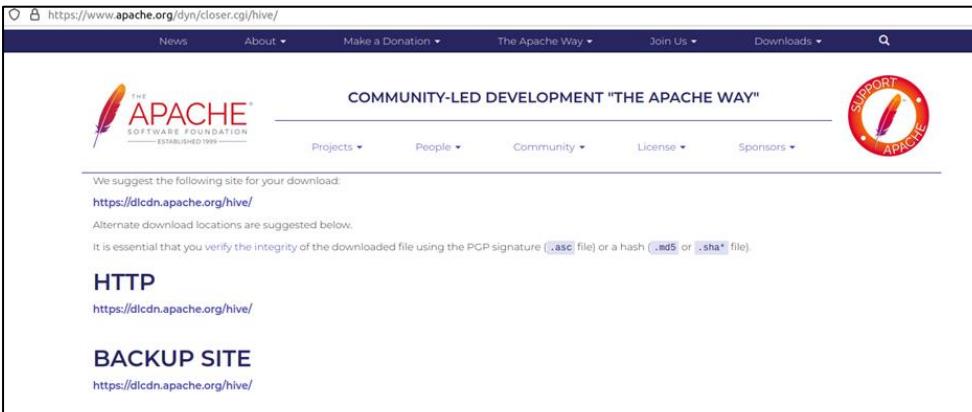
Step-1 Download Apache HIVE

Visit <https://hive.apache.org/downloads.html>



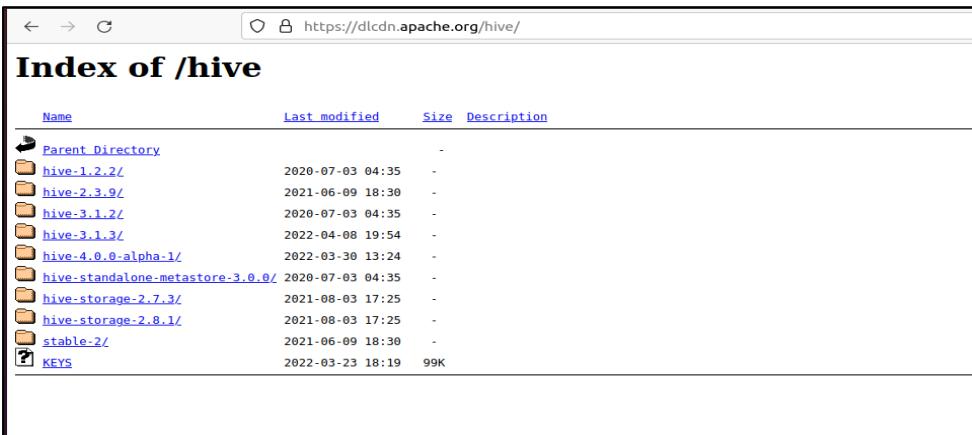
The screenshot shows the Apache Hive download page at https://hive.apache.org/downloads.html. The page features a logo of a yellow bee with wings spread wide, above the word "HIVE". On the left, there's a sidebar with links for "GENERAL" (Home, Downloads, License, Privacy Policy) and "DOCUMENTATION" (Language Manual, Javadoc). The main content area is titled "DOWNLOADS" and contains a sub-section "Releases may be downloaded from Apache mirrors:". It includes a link "Download a release now!" and a note: "On the mirror, all recent releases are available, but are not guaranteed to be stable. For stable releases, look in the stable directory." Below this, there's a "SUPPORT" button with a red feather icon.

Click on download a release now! Link



The screenshot shows the Apache Software Foundation homepage at https://www.apache.org/dyn/closer.cgi/hive/. The page has a dark header with links for News, About, Make a Donation, The Apache Way, Join Us, Downloads, and a search bar. The main content area features the Apache logo and the tagline "COMMUNITY-LED DEVELOPMENT 'THE APACHE WAY'". It lists download links under "HTTP" and "BACKUP SITE", both pointing to https://dlcdn.apache.org/hive/. There's also a "SUPPORT" button with a red feather icon.

Click on <https://dlcdn.apache.org/hive/>



The screenshot shows the Apache Hive download index at https://dlcdn.apache.org/hive/. The page title is "Index of /hive". It displays a table of contents with columns: Name, Last modified, Size, and Description. The table lists various versions of the Hive software, including "Parent Directory", "hive-1.2.2/", "hive-2.3.9/", "hive-3.1.2/", "hive-3.1.3/", "hive-4.0.0-alpha-1/", "hive-standalone-metastore-3.0.0/", "hive-storage-2.7.3/", "hive-storage-2.8.1/", "stable-2/", and "KEYS". Each entry shows the file name, last modified date, size (e.g., -), and a brief description.

Select the apache-hive-3.1.2-bin.tar.gz file to begin the download process.

Name	Last modified	Size	Description
<a href="#">Parent Directory</a>		-	
<a href="#">apache-hive-3.1.2-bin.tar.gz</a>	2020-07-03 04:35	266M	
<a href="#">apache-hive-3.1.2-bin.tar.gz.asc</a>	2020-07-03 04:34	833	
<a href="#">apache-hive-3.1.2-bin.tar.gz.sha256</a>	2020-07-03 04:34	95	
<a href="#">apache-hive-3.1.2-src.tar.gz</a>	2020-07-03 04:34	24M	
<a href="#">apache-hive-3.1.2-src.tar.gz.asc</a>	2020-07-03 04:35	833	
<a href="#">apache-hive-3.1.2-src.tar.gz.sha256</a>	2020-07-03 04:34	95	

Step-2 su - hdoop

```
wget https://dlcdn.apache.org/hive/hive-3.1.2/apache-hive-3.1.2-bin.tar.gz
```

Step 3 tar xvzf apache-hive-3.1.2-bin.tar.gz

```
hdoop@PRLAB-22:~$ tar xvzf apache-hive-3.1.2-bin.tar.gz
apache-hive-3.1.2-bin/LICENSE
apache-hive-3.1.2-bin/NOTICE
apache-hive-3.1.2-bin/RELEASE_NOTES.txt
apache-hive-3.1.2-bin/binary-package-licenses/asm-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/com.google.protobuf-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/com.ibm.icu.icu4j-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/com.sun.jersey-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/com.thoughtworks.paranamer-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/javax.transaction.transaction-api-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/javolution-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/jline-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/NOTICE
apache-hive-3.1.2-bin/binary-package-licenses/org.abego.treelayout.core-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/org.antlr.LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/org.antlr.antlr4-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/org.antlr.stringtemplate-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/org.codehaus.janino-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/org.jamon.jamon-runtime-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/org.jruby-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/org.mozilla.rhino-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/org.slf4j-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/sqlline-LICENSE
apache-hive-3.1.2-bin/examples/files/2000_cols_data.csv
apache-hive-3.1.2-bin/examples/files/3col_data.txt
apache-hive-3.1.2-bin/examples/files/4col_data.txt
```

Step 4 vi .bashrc

```
hdoop@PRLAB-22:~$ vi .bashrc
hdoop@PRLAB-22:~$
```

Append the following Hive environment variables to the .bashrc file:

```
export HIVE_HOME= "/home/hdoop/apache-hive-3.1.2-bin"
export PATH=$PATH:$ HIVE_HOME/bin
```

```

hadoop@PRLAB-22: ~
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc .
if ! shopt -oq posix; then
    if [ -f /usr/share/bash-completion/bash_completion ]; then
        . /usr/share/bash-completion/bash_completion
    elif [ -f /etc/bash_completion ]; then
        . /etc/bash_completion
    fi
fi
#Hadoop Related Options
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export HADOOP_HOME=/home/hadoop/hadoop
export HADOOP_INSTALL=$HADOOP_HOME
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib/native"
#Hive Related Options
export HIVE_HOME="/home/hadoop/apache-hive-3.1.2-bin"
export PATH=$PATH:$HIVE_HOME/bin

```

Step 5 source ~/.bashrc

```

hadoop@PRLAB-22: ~$ source ~/.bashrc

```

Step 6 vi \$HIVE\_HOME/bin/hive-config.sh

```

hadoop@PRLAB-22: ~/apache-hive-3.1.2-bin/bin
HIVE_CONF_DIR=$confdir
;;
--auxpath)
shift
HIVE_AUX_JARS_PATH=$1
shift
;;
*)
break;
;;
esac
done

# Allow alternate conf dir location.
HIVE_CONF_DIR="${HIVE_CONF_DIR:-$HIVE_HOME/conf}"

export HIVE_CONF_DIR=$HIVE_CONF_DIR
export HADOOP_HOME=/home/hadoop/hadoop
export HIVE_AUX_JARS_PATH=$HIVE_AUX_JARS_PATH

```

Start hadoop file system and yarn

```

itadmin@PRLAB-22:~$ su - hdoop
Password:
hdoop@PRLAB-22:~$ cd hadoop
hdoop@PRLAB-22:~/hadoop$ cd sbin
hdoop@PRLAB-22:~/hadoop/sbin$ ./start-dfs.sh
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [PRLAB-22]
hdoop@PRLAB-22:~/hadoop/sbin$ ./start-yarn.sh
Starting resourcemanager
Starting nodemanagers
hdoop@PRLAB-22:~/hadoop/sbin$ jps
37620 NodeManager
38374 Jps
36616 DataNode
36856 SecondaryNameNode
37467 ResourceManager
36060 NameNode

```

Step-7 Create two separate directories to store data in the HDFS layer:

Step 7.1 hadoop fs -mkdir /tmp

```

hdoop@PRLAB-22:~$ hadoop fs -mkdir /tmp
hdoop@PRLAB-22:~$ 

```

The screenshot shows the Hadoop File Explorer interface at [localhost:9870/explorer.html#](http://localhost:9870/explorer.html#/). The top navigation bar includes links for Hadoop, Overview, Datanodes, Datanode Volume Failures, Snapshot, Startup Progress, and Utilities. The main area is titled "Browse Directory" and displays the contents of the /tmp directory. The table shows the following entries:

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxr-xr-x	hdoop	supergroup	0 B	May 25 01:21	0	0 B	may24
drwxr-xr-x	hdoop	supergroup	0 B	May 21 02:57	0	0 B	sample
drwxr-xr-x	dr.who	supergroup	0 B	May 21 02:13	0	0 B	test
drwxr-xr-x	hdoop	supergroup	0 B	May 28 02:09	0	0 B	tmp

Below the table, it says "Showing 1 to 4 of 4 entries". At the bottom left, it says "Hadoop, 2022."

Step 7.2 hadoop fs -chmod g+w /tmp

Step 7.3 hadoop fs -ls /

The screenshot shows the Hadoop File Explorer interface at [localhost:9870/explorer.html#](http://localhost:9870/explorer.html#/). The top navigation bar includes links for Hadoop, Overview, Datanodes, Datanode Volume Failures, Snapshot, Startup Progress, and Utilities. The main area is titled "Browse Directory" and displays the contents of the / directory. The table shows the following entries:

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxr-xr-x	hdoop	supergroup	0 B	May 25 01:21	0	0 B	may24
drwxr-xr-x	hdoop	supergroup	0 B	May 21 02:57	0	0 B	sample
drwxr-xr-x	dr.who	supergroup	0 B	May 21 02:13	0	0 B	test
drwxrwxr-x	hdoop	supergroup	0 B	May 28 02:09	0	0 B	tmp

Below the table, it says "Showing 1 to 4 of 4 entries".

```

hadoop@PRLAB-22:~$ hadoop fs -ls /
Found 4 items
drwxr-xr-x  - hdoop  supergroup          0 2022-05-25 01:21 /may24
drwxr-xr-x  - hdoop  supergroup          0 2022-05-21 02:57 /sample
drwxr-xr-x  - dr.who supergroup          0 2022-05-21 02:13 /test
drwxrwxr-x  - hdoop  supergroup          0 2022-05-28 02:09 /tmp
hadoop@PRLAB-22:~$
```

Step 7.4 hadoop fs -mkdir -p /user/hive/warehouse

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxr-xr-x	hdoop	supergroup	0 B	May 28 02:18	0	0 B	warehouse

Step 7.5 hadoop fs -chmod g+w /user/hive/warehouse

```

hadoop@PRLAB-22:~$ hadoop fs -chmod g+w /user/hive/warehouse
hadoop@PRLAB-22:~$
```

```

hadoop@PRLAB-22:~$ hadoop fs -ls /user/hive
Found 1 items
drwxrwxr-x  - hdoop  supergroup          0 2022-05-28 02:18 /user/hive/warehouse
hadoop@PRLAB-22:~$
```

Step 8 cd \$HIVE\_HOME/conf

```

hadoop@PRLAB-22:~/apache-hive-3.1.2-bin/conf$ cd $HIVE_HOME/conf
hadoop@PRLAB-22:~/apache-hive-3.1.2-bin/conf$ ls
beeline-log4j2.properties.template
hive-default.xml.template
hive-env.sh.template
hive-exec-log4j2.properties.template
hive-log4j2.properties.template
ivysettings.xml
llap-cli-log4j2.properties.template
llap-daemon-log4j2.properties.template
parquet-logging.properties
hadoop@PRLAB-22:~/apache-hive-3.1.2-bin/conf$
```

Use the hive-default.xml.template to create the hive-site.xml file:

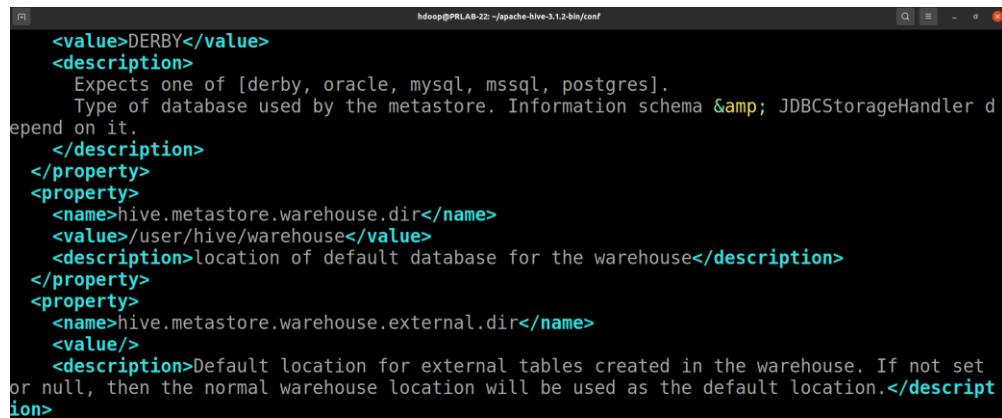
```

hadoop@PRLAB-22:~/apache-hive-3.1.2-bin/conf$ cp hive-default.xml.template hive-site.xml
hadoop@PRLAB-22:~/apache-hive-3.1.2-bin/conf$
```

Access the hive-site.xml file using the text editor of your choice

```
hadoop@PRLAB-22:~/apache-hive-3.1.2-bin/conf$ vi hive-site.xml  
hadoop@PRLAB-22:~/apache-hive-3.1.2-bin/conf$
```

Configure the system to use your local storage rather than the HDFS layer by setting the `hive.metastore.warehouse.dir` parameter value to the location of your Hive warehouse directory.



```
<value>DERBY</value>
<description>
  Expects one of [derby, oracle, mysql, mssql, postgres].
  Type of database used by the metastore. Information schema & JDBCStorageHandler depend on it.
</description>
</property>
<property>
  <name>hive.metastore.warehouse.dir</name>
  <value>/user/hive/warehouse</value>
  <description>location of default database for the warehouse</description>
</property>
<property>
  <name>hive.metastore.warehouse.external.dir</name>
  <value/>
  <description>Default location for external tables created in the warehouse. If not set or null, then the normal warehouse location will be used as the default location.</description>
```

### Step 9: Initiate Database

```
$HIVE_HOME/bin/schematool -dbType derby -initSchema
```

```
hadoop@PRLAB-22:~ $ $HIVE_HOME/bin/schematool -dbType derby -initSchema
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/hadoop/apache-hive-3.1.2-bin/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/hadoop/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Exception in thread "main" java.lang.NoSuchMethodError: com.google.common.base.Preconditions.checkNotNull(ZLjava/lang/String;Ljava/lang/Object;)V
        at org.apache.hadoop.conf.Configuration.set(Configuration.java:1357)
        at org.apache.hadoop.conf.Configuration.set(Configuration.java:1338)
        at org.apache.hadoop.mapred.JobConf.setJar(JobConf.java:536)
        at org.apache.hadoop.mapred.JobConf.setJarByClass(JobConf.java:554)
        at org.apache.hadoop.mapred.JobConf.<init>(JobConf.java:448)
        at org.apache.hadoop.hive.conf.HiveConf.initialize(HiveConf.java:5141)
        at org.apache.hadoop.hive.conf.HiveConf.<init>(HiveConf.java:5104)
        at org.apache.hive.beeline.HiveSchemaTool.<init>(HiveSchemaTool.java:96)
        at org.apache.hive.beeline.HiveSchemaTool.main(HiveSchemaTool.java:1473)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.lang.reflect.Method.invoke(Method.java:498)
        at org.apache.hadoop.util.RunJar.run(RunJar.java:323)
```

### Step 10.1 Locate the guava jar file in the Hive lib directory:

```
ls $HIVE_HOME/lib
```

```
hadoop@PRLAB-22: ~
esri-geometry-api-2.0.0.jar
findbugs-annotations-1.3.9-1.jar
flatbuffers-1.2.0-3f79e055.jar
groovy-all-2.4.11.jar
gson-2.2.4.jar
guava-19.0.jar
hbase-client-2.0.0-alpha4.jar
hbase-common-2.0.0-alpha4.jar
hbase-common-2.0.0-alpha4-tests.jar
hbase-hadoop2-compat-2.0.0-alpha4.jar
hbase-hadoop2-compat-2.0.0-alpha4-tests.jar
```

Step 10.2 Locate the guava jar file in the Hadoop lib directory as well:

```
ls $HADOOP_HOME/share/hadoop/hdfs/lib
```

```
hadoop@PRLAB-22: ~
error_prone_annotations-2.2.0.jar
failureaccess-1.0.jar
gson-2.2.4.jar
guava-27.0-jre.jar
hadoop-annotations-3.2.3.jar
hadoop-auth-3.2.3.jar
htrace-core4-4.1.0-incubating.jar
httpclient-4.5.13.jar
httpcore-4.4.13.jar
j2objc-annotations-1.1.jar
```

Step 10.3 The two listed versions are not compatible and are causing the error. Remove the existing guava file from the Hive lib directory:

```
rm $HIVE_HOME/lib/guava-19.0.jar
```

```
hadoop@PRLAB-22:~$ rm $HIVE_HOME/lib/guava-19.0.jar
hadoop@PRLAB-22:~$
```

Step 10.4 Copy the guava file from the Hadoop lib directory to the Hive lib directory:

```
cp $HADOOP_HOME/share/hadoop/hdfs/lib/guava-27.0-jre.jar $HIVE_HOME/lib/
```

Step 11-Use the schematool command once again to initiate the Derby database:

```
$HIVE_HOME/bin/schematool -dbType derby -initSchema
```

```

hadoop@PRLAB-22:~$ $HIVE_HOME/bin/schematool -dbType derby -initSchema
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/hadoop/apache-hive-3.1.2-bin/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/hadoop/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Metastore connection URL:      jdbc:derby:;databaseName=metastore_db;create=true
Metastore Connection Driver :  org.apache.derby.jdbc.EmbeddedDriver
Metastore connection User:     APP
Starting metastore schema initialization to 3.1.0
Initialization script hive-schema-3.1.0.derby.sql

```

Initialization script completed  
schemaTool completed

```

hadoop@PRLAB-22:~$ 
hadoop@PRLAB-22:~$ 
hadoop@PRLAB-22:~$ 
hadoop@PRLAB-22:~$ 

```

Step-12 Launch Hive Client Shell on Ubuntu

```
cd $HIVE_HOME/bin
```

```
hive
```

Step 12.1 While starting hive if these types or errors occur

```

hadoop@PRLAB-22:~/apache-hive-3.1.2-bin$ hive
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/hadoop/apache-hive-3.1.2-bin/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/hadoop/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Hive Session ID = 981ad83d-79b6-4856-89f7-9a1c51cef3f2

Logging initialized using configuration in jar:file:/home/hadoop/apache-hive-3.1.2-bin/lib/hive-common-3.1.2.jar!/hive-log4j2.properties.Async: true
Exception in thread "main" java.lang.IllegalArgumentException: java.net.URISyntaxException: Relative path in absolute URI: ${system:java.io.tmpdir%7D${%7Bsystem:user.name%7D
        at org.apache.hadoop.fs.Path.initialize(Path.java:263)
        at org.apache.hadoop.fs.Path.<init>(Path.java:221)
        at org.apache.hadoop.hive ql.session.SessionState.createSessionDirs(SessionState.java:710)
        at org.apache.hadoop.hive ql.session.SessionState.start(SessionState.java:627)
        at org.apache.hadoop.hive ql.session.SessionState.beginStart(SessionState.java:591)
        at org.apache.hadoop.hive cli.CliDriver.run(CliDriver.java:747)
        at org.apache.hadoop.hive cli.CliDriver.main(CliDriver.java:683)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccesso
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccesso
        at java.lang.reflect.Method.invoke(Method.java:498)
        at org.apache.hadoop.util.RunJar.run(RunJar.java:323)
        at org.apache.hadoop.util.RunJar.main(RunJar.java:236)

```

Step 12.2 If you are receiving the below exception then open the `hive-site.xml` file using nano editor and press **CTRL+W** and search for “`system:java.io.tmpdir`” and replace it with `/tmp/mydir`.

```

hadoop@PRLAB-22:~/apache-hive-3.1.2-bin$ cd conf
hadoop@PRLAB-22:~/apache-hive-3.1.2-bin/conf$ ls
beeline-log4j2.properties.template    hive-site.xml
hive-default.xml.template           ivysettings.xml
hive-env.sh.template                llap-cl
hive-exec-log4j2.properties.template llap-daemon-log4j2.properties.template
hive-log4j2.properties.template     parquet-logging.properties
hadoop@PRLAB-22:~/apache-hive-3.1.2-bin/conf$ vi hive-site.xml
hadoop@PRLAB-22:~/apache-hive-3.1.2-bin/conf$ nano hive-site.xml

```

Step 12.3 Now after that again Launch Hive Client Shell on Ubuntu

```
hadoop@PRLAB-22:~/apache-hive-3.1.2-bin$ cd bin
hadoop@PRLAB-22:~/apache-hive-3.1.2-bin/bin$ hive
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/hadoop/apache-hive-3.1.2-bin/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Hive Session ID = 8852e552-7e42-4c72-9012-09ad21f0c1d
Logging initialized using configuration in jar:file:/home/hadoop/apache-hive-3.1.2-bin/lib/hive-common-3.1.2.jar!/hive-log4j2.properties Async: true
Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
hive>
```

Step 13- error may occur when hive-shell started before metastore\_db service. To avoid this just delete or move your metastore\_db and try the below command.

```
$ mv metastore_db metastore_db.tmp
```

Step-14 : once again to initiate the Derby database

```
$ schematool -dbType derby -initSchema
```

## **Result:**

HIVE has been successfully installed above Hadoop in the system

## Hive – CRUD Operations

### **1)Aim:**

To implement any schema definition in HIVE and perform CRUD operations.

### **Code:**

#### **Table Creation:**

```
create table studentnew (id int, name string, location string) clustered by (location) into 3
buckets row format delimited fields terminated by ',' lines terminated by '\n' stored as orc
TBLPROPERTIES ('transactional'='true');
```

While table creation you might face some errors so do use these commands in terminal for as temporary solution

```
Set hive.support.concurrency = true;
Set hive.enforce.bucketing = true;
set hive.exec.dynamic.partition.mode = nonstrict;
set hive.txn.manager = org.apache.hadoop.hive.ql.lockmgr.DbTxnManager;
set hive.compactor.initiator.on = true;
set hive.compactor.worker.threads =1;
```

### **Output:**

```
hive> Set hive.support.concurrency = true;
hive> Set hive.enforce.bucketing = true;
hive> set hive.exec.dynamic.partition.mode = nonstrict;
hive> set hive.txn.manager = org.apache.hadoop.hive.ql.lockmgr.DbTxnManager;
hive> set hive.compactor.initiator.on = true;
hive> set hive.compactor.worker.threads =1;
hive> create table studentnew (id int, name string, location string) clustered by (location) into 3 bucket
s row format delimited fields terminated by ',' lines terminated by '\n' stored as orc TBLPROPERTIES ('tran
sactional'='true');
OK
Time taken: 0.807 seconds
```

### **Insertion of records:**

```
INSERT INTO TABLE studentnew VALUES (101,'abc','GST Road'),
(102,'A','Guindy'),
(103,'PRABU','henry road'),
(104,'KUMAR','gandhi road');
```

## **Output:**

```
hive> insert into table studentnew values(101,'abc','GST Road');
Query ID = hdoop_20220531030502_54f3eff2-c5fa-452d-85e0-c728e6b2a704
Total jobs = 2
Launching Job 1 out of 2
Number of reduce tasks determined at compile time: 3
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1653941890266_0001, Tracking URL = http://PRLAB-22:8088/proxy/application_1653941890266_0001/
Kill Command = /home/hdoop/hadoop/bin/mapred job -kill job_1653941890266_0001
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 3
2022-05-31 03:05:13,825 Stage-1 map = 0%, reduce = 0%
2022-05-31 03:05:16,916 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 1.7 sec
2022-05-31 03:05:22,021 Stage-1 map = 100%, reduce = 67%, Cumulative CPU 4.5 sec
2022-05-31 03:05:24,050 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 6.36 sec
MapReduce Total cumulative CPU time: 6 seconds 360 msec
Ended Job = job_1653941890266_0001
Loading data to table default.studentnew
Launching Job 2 out of 2
Number of reduce tasks determined at compile time: 1

Time taken: 42.007 seconds
hive> insert into table studentnew values(102,'prabhu','guindy');
Query ID = hdoop_20220531030612_c8da22c1-8645-4bf3-a5aa-a4fd8b25f306
Total jobs = 2
Launching Job 1 out of 2
Number of reduce tasks determined at compile time: 3
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1653941890266_0003, Tracking URL = http://PRLAB-22:8088/proxy/application_1653941890266_0003/
Kill Command = /home/hdoop/hadoop/bin/mapred job -kill job_1653941890266_0003
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 3
2022-05-31 03:06:17,426 Stage-1 map = 0%, reduce = 0%
2022-05-31 03:06:20,519 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 1.72 sec
2022-05-31 03:06:24,612 Stage-1 map = 100%, reduce = 33%, Cumulative CPU 3.08 sec
2022-05-31 03:06:26,677 Stage-1 map = 100%, reduce = 67%, Cumulative CPU 5.09 sec
2022-05-31 03:06:27,691 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 6.44 sec
MapReduce Total cumulative CPU time: 6 seconds 440 msec
Ended Job = job_1653941890266_0003
Loading data to table default.studentnew
Launching Job 2 out of 2
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1653941890266_0004, Tracking URL = http://PRLAB-22:8088/proxy/application_1653941890266_0004/
Kill Command = /home/hdoop/hadoop/bin/mapred job -kill job_1653941890266_0004
Hadoop job information for Stage-3: number of mappers: 1; number of reducers: 1
2022-05-31 03:06:37,734 Stage-3 map = 0%, reduce = 0%
2022-05-31 03:06:41,839 Stage-3 map = 100%, reduce = 0%, Cumulative CPU 0.98 sec
2022-05-31 03:06:45,934 Stage-3 map = 100%, reduce = 100%, Cumulative CPU 2.37 sec
MapReduce Total cumulative CPU time: 2 seconds 370 msec
Ended Job = job_1653941890266_0004
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 3  Cumulative CPU: 6.44 sec  HDFS Read: 25696 HDFS Write: 1503 SUCCESS
Stage-Stage-3: Map: 1  Reduce: 1  Cumulative CPU: 2.37 sec  HDFS Read: 12604 HDFS Write: 201 SUCCESS
Total MapReduce CPU Time Spent: 8 seconds 810 msec
OK
Time taken: 34.986 seconds
```

```

hive> insert into table studentnew values(103,'gandhi','radha nagar chrompet');
Query ID = hdoop_20220531030844_f8903933-7850-4c09-a4c3-2bae37a525d9
Total jobs = 2
Launching Job 1 out of 2
Number of reduce tasks determined at compile time: 3
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1653941890266_0005, Tracking URL = http://PRLAB-22:8088/proxy/application_1653941890266_0005/
Kill Command = /home/hdoop/hadoop/bin/mapred job -kill job_1653941890266_0005
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 3
2022-05-31 03:08:49,897 Stage-1 map = 0%, reduce = 0%
2022-05-31 03:08:52,959 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 1.99 sec
2022-05-31 03:08:57,012 Stage-1 map = 100%, reduce = 33%, Cumulative CPU 3.33 sec
2022-05-31 03:08:59,079 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 6.42 sec
MapReduce Total cumulative CPU time: 6 seconds 420 msec
Ended Job = job_1653941890266_0005
Loading data to table default.studentnew
Launching Job 2 out of 2
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1653941890266_0006, Tracking URL = http://PRLAB-22:8088/proxy/application_1653941890266_0006/
Kill Command = /home/hdoop/hadoop/bin/mapred job -kill job_1653941890266_0006
Hadoop job information for Stage-3: number of mappers: 1; number of reducers: 1
2022-05-31 03:09:09,700 Stage-3 map = 0%, reduce = 0%
2022-05-31 03:09:13,772 Stage-3 map = 100%, reduce = 0%, Cumulative CPU 0.89 sec
2022-05-31 03:09:17,832 Stage-3 map = 100%, reduce = 100%, Cumulative CPU 2.08 sec
MapReduce Total cumulative CPU time: 2 seconds 80 msec
Ended Job = job_1653941890266_0006
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Reduce: 3 Cumulative CPU: 6.42 sec HDFS Read: 25813 HDFS Write: 1574 SUCCESS
Stage-Stage-3: Map: 1 Reduce: 1 Cumulative CPU: 2.08 sec HDFS Read: 12604 HDFS Write: 203 SUCCESS
Total MapReduce CPU Time Spent: 8 seconds 500 msec
OK
Time taken: 34.621 seconds

```

```

hive> select * from studentnew;
OK
101      abc      GST Road
102      prabhu   guindy
103      gandhi   radha nagar chrompet
Time taken: 0.333 seconds, Fetched: 3 row(s)

```

## Updation of records:

Update studentnew SET id = 113 WHERE id = 103;

## Output:

```

hive> Update studentnew SET id = 113 WHERE id = 103;
Query ID = hdoop_20220531031127_9ec9ac2d-39f9-4bd5-b809-5c4ea6ceb5d2
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 3
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1653941890266_0007, Tracking URL = http://PRLAB-22:8088/proxy/application_1653941890266_0007/
Kill Command = /home/hdoop/hadoop/bin/mapred job -kill job_1653941890266_0007
Hadoop job information for Stage-1: number of mappers: 3; number of reducers: 3
2022-05-31 03:11:33,595 Stage-1 map = 0%, reduce = 0%
2022-05-31 03:11:38,685 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 8.12 sec
2022-05-31 03:11:42,751 Stage-1 map = 100%, reduce = 33%, Cumulative CPU 9.56 sec
2022-05-31 03:11:43,770 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 12.97 sec
MapReduce Total cumulative CPU time: 12 seconds 970 msec
Ended Job = job_1653941890266_0007
Loading data to table default.studentnew
MapReduce Jobs Launched:
Stage-Stage-1: Map: 3 Reduce: 3 Cumulative CPU: 12.97 sec HDFS Read: 43124 HDFS Write: 1801 SUCCESS
Total MapReduce CPU Time Spent: 12 seconds 970 msec
OK
Time taken: 18.984 seconds

```

```

hive> select * from studentnew;
OK
101      abc      GST Road
102      prabhu   guindy
113      gandhi   radha nagar chrompet
Time taken: 0.16 seconds, Fetched: 3 row(s)

```

### Deletion of records:

delete from studentnew where id=102;

### Output:

```

hive> delete from studentnew where id=102;
Query ID = hdoop_20220531031339_a531bcab-2da2-4826-81af-a8d4f44dc0f4
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 3
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1653941890266_0008, Tracking URL = http://PRLAB-22:8088/proxy/application_1653941890266_0008/
Kill Command = /home/hdoop/hadoop/bin/mapred job -kill job_1653941890266_0008
Hadoop job information for Stage-1: number of mappers: 4; number of reducers: 3
2022-05-31 03:13:45,465 Stage-1 map = 0%, reduce = 0%
2022-05-31 03:13:51,592 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 11.5 sec
2022-05-31 03:13:55,652 Stage-1 map = 100%, reduce = 33%, Cumulative CPU 13.28 sec
2022-05-31 03:13:57,690 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 16.71 sec
MapReduce Total cumulative CPU time: 16 seconds 710 msec
Ended Job = job_1653941890266_0008
Loading data to table default.studentnew
MapReduce Jobs Launched:
Stage-Stage-1: Map: 4 Reduce: 3   Cumulative CPU: 16.71 sec   HDFS Read: 51452 HDFS Write: 896 SUCCESS
Total MapReduce CPU Time Spent: 16 seconds 710 msec
OK
Time taken: 19.13 seconds
hive> select * from studentnew;
OK
101      abc      GST Road
113      gandhi   radha nagar chrompet
Time taken: 0.168 seconds, Fetched: 2 row(s)

```

View the table created in Browser-UI

File Type	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
Directory	drwxr-xr-x	hdoop	supergroup	0 B	May 31 02:33	0	0 B	student
Directory	drwxr-xr-x	hdoop	supergroup	0 B	May 31 03:13	0	0 B	studentnew

The screenshot shows a browser window for a Hadoop file system. The URL is `localhost:9870/explorer.html#/user/hive/warehouse/studentnew`. The page title is "Browse Directory". The address bar shows the path `/user/hive/warehouse/studentnew`. The top navigation bar includes links for Hadoop, Overview, Datanodes, Datanode Volume Failures, Snapshot, Startup Progress, and Utilities. Below the navigation is a search bar with placeholder "Search:" and a "Go!" button. A dropdown menu "Show 25 entries" is open. The main content is a table listing file entries:

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxr-xr-x	hdoop	supergroup	0 B	May 31 03:11	0	0 B	delete_delta_000004_000004_0000
drwxr-xr-x	hdoop	supergroup	0 B	May 31 03:13	0	0 B	delete_delta_000005_000005_0000
drwxr-xr-x	hdoop	supergroup	0 B	May 31 03:05	0	0 B	delta_0000001_0000001_0000
drwxr-xr-x	hdoop	supergroup	0 B	May 31 03:06	0	0 B	delta_0000002_0000002_0000
drwxr-xr-x	hdoop	supergroup	0 B	May 31 03:09	0	0 B	delta_0000003_0000003_0000
drwxr-xr-x	hdoop	supergroup	0 B	May 31 03:11	0	0 B	delta_0000004_0000004_0000

Below the table, it says "Showing 1 to 6 of 6 entries". There are "Previous" and "Next" buttons. At the bottom left, it says "Hadoop, 2022."

## Result:

CRUD operations have been successfully executed in HIVE and the results are verified.