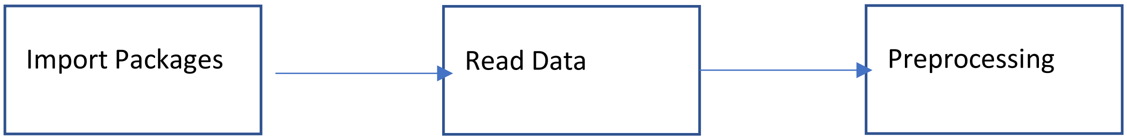
CORE FUNCTIONALITIES OF PHASE 1

Module – 1

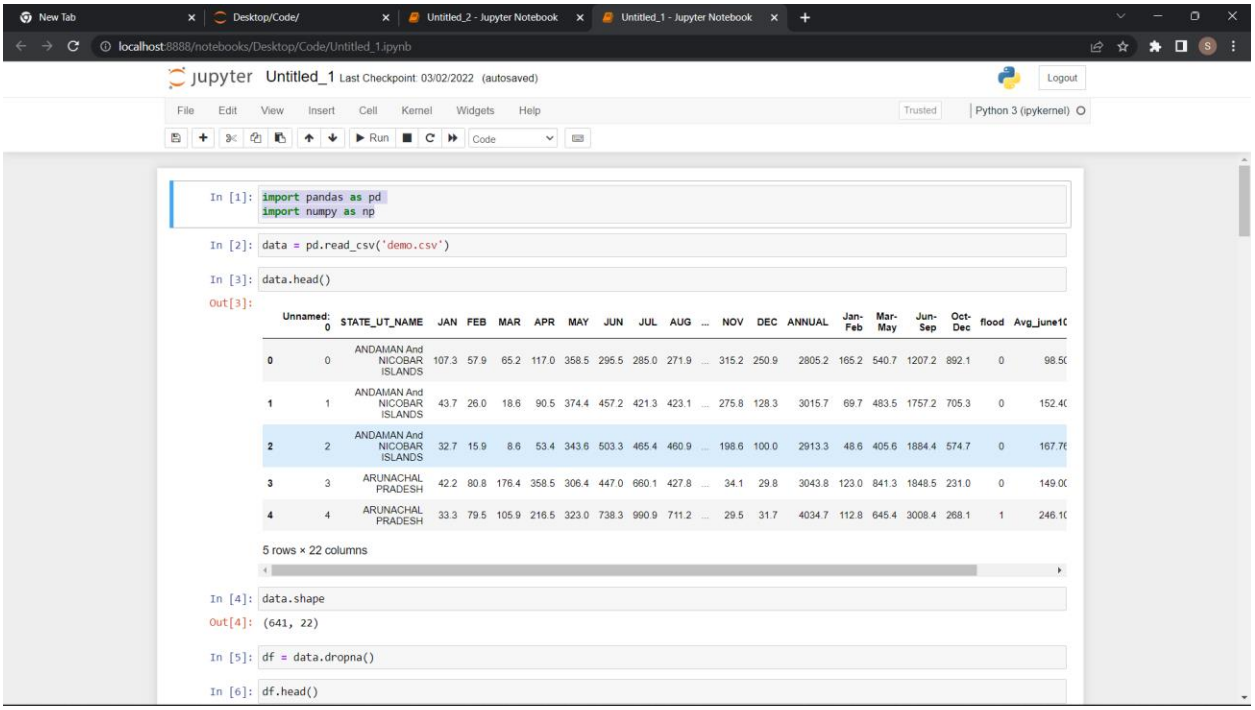
- In the module one the data is loaded, and we get removed noisy data as the output. The variable identification is done with Uni, Bi, Multi Variate analysis by importing the libraries to read the data set and to analyze the dataset.
- The dataset is displayed as data frame it shows the columns, describe the data frames.
- The missing values, outliers unique values of data frame will be identified.



```
import pandas as pd
import numpy as np


data = pd.read_csv('demo.csv')
data.head()
data.shape
df = data.dropna()
df.head()
df.shape
df.columns
df.info()
df.describe()
df.isnull()
df.isnull().sum()

df.STATE_UT_NAME.unique()

df.columns

df.flood.unique()

df.JAN.unique().mean()

df.FEB.unique().mean()

df.ANNUAL.unique().min()

df.ANNUAL.unique().max()

df.ANNUAL.unique().mean()

df.Avg_june10days.unique()

df.corr()
```

Input: Data

Output: Removal of Noisy Data

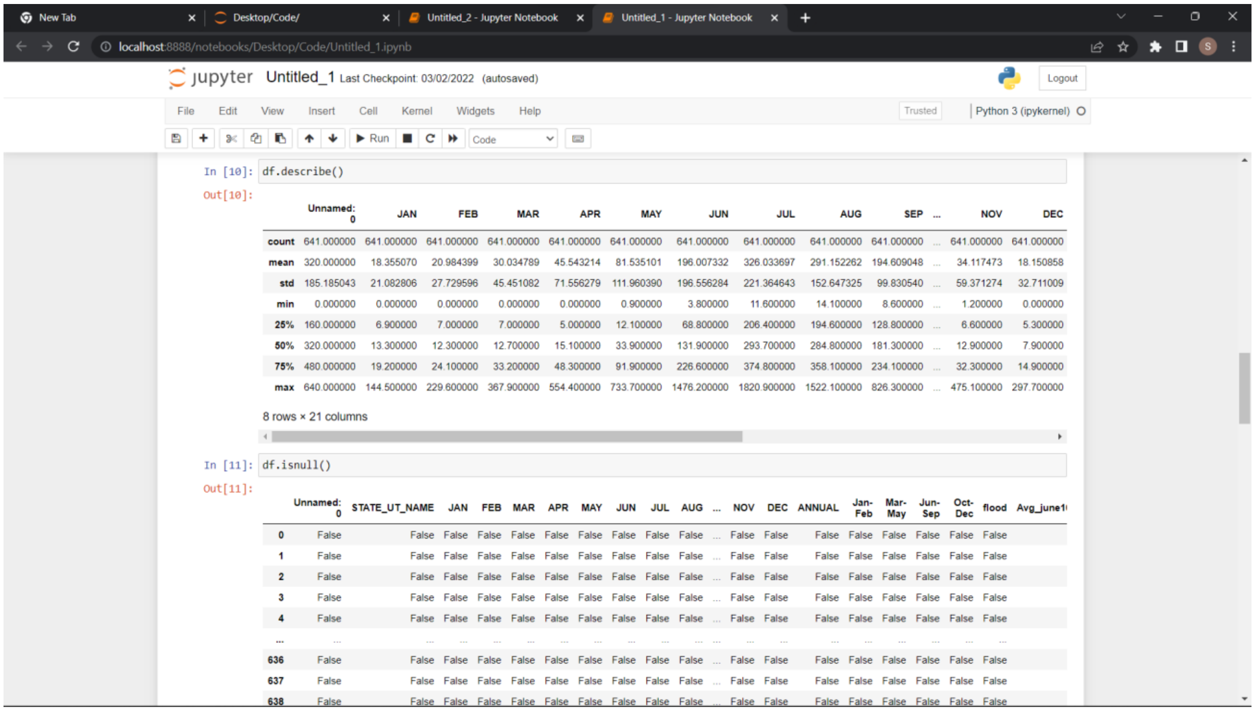

New Tab
Desktop/Code/
Untitled_2 - Jupyter Notebook
Untitled_1 - Jupyter Notebook
localhost:8888/notebooks/Desktop/Code/Untitled_1.ipynb
jupyter Untitled_1 Last Checkpoint: 03/02/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code

```
In [1]: import pandas as pd
        import numpy as np

In [2]: data = pd.read_csv('demo.csv')

In [3]: data.head()
Out[3]:
```

	Unnamed: 0	STATE_UT_NAME	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	...	NOV	DEC	ANNUAL	Jan-Feb	Mar-May	Jun-Sep	Oct-Dec	flood	Avg_june1(
0	0	ANDAMAN And NICOBAR ISLANDS	107.3	57.9	65.2	117.0	358.5	295.5	285.0	271.9	...	315.2	250.9	2805.2	165.2	540.7	1207.2	892.1	0	98.5(
1	1	ANDAMAN And NICOBAR ISLANDS	43.7	26.0	18.6	90.5	374.4	457.2	421.3	423.1	...	275.8	128.3	3015.7	69.7	483.5	1757.2	705.3	0	152.4(
2	2	ANDAMAN And NICOBAR ISLANDS	32.7	15.9	8.6	53.4	343.6	503.3	465.4	460.9	...	198.6	100.0	2913.3	48.6	405.6	1884.4	574.7	0	167.7(
3	3	ARUNACHAL PRADESH	42.2	80.8	176.4	358.5	306.4	447.0	660.1	427.8	...	34.1	29.8	3043.8	123.0	841.3	1848.5	231.0	0	149.0(
4	4	ARUNACHAL PRADESH	33.3	79.5	105.9	216.5	323.0	738.3	990.9	711.2	...	29.5	31.7	4034.7	112.8	645.4	3008.4	268.1	1	246.1(

5 rows × 22 columns

```
In [4]: data.shape
Out[4]: (641, 22)

In [5]: df = data.dropna()

In [6]: df.head()
```


New Tab
Desktop/Code/
Untitled_2 - Jupyter Notebook
Untitled_1 - Jupyter Notebook
localhost:8888/notebooks/Desktop/Code/Untitled_1.ipynb
jupyter Untitled_1 Last Checkpoint: 03/02/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code

```
In [10]: df.describe()
Out[10]:
```

	Unnamed: 0	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	...	NOV	DEC
count	641.000000	641.000000	641.000000	641.000000	641.000000	641.000000	641.000000	641.000000	641.000000	641.000000	...	641.000000	641.000000
mean	320.000000	18.355070	20.984399	30.034789	45.543214	81.535101	196.007332	326.033697	291.152262	194.609048	...	34.117473	18.150858
std	185.185043	21.082806	27.729596	45.451082	71.556279	111.960390	196.556284	221.364643	152.647325	99.830540	...	59.371274	32.711009
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.900000	3.800000	11.600000	14.100000	8.600000	...	1.200000	0.000000
25%	160.000000	6.900000	7.000000	7.000000	5.000000	12.100000	68.800000	206.400000	194.600000	128.800000	...	6.600000	5.300000
50%	320.000000	13.300000	12.300000	12.700000	15.100000	33.900000	131.900000	293.700000	284.800000	181.300000	...	12.900000	7.900000
75%	480.000000	19.200000	24.100000	33.200000	48.300000	91.900000	226.600000	374.800000	358.100000	234.100000	...	32.300000	14.900000
max	640.000000	144.500000	229.600000	367.900000	554.400000	733.700000	1476.200000	1820.900000	1522.100000	826.300000	...	475.100000	297.700000

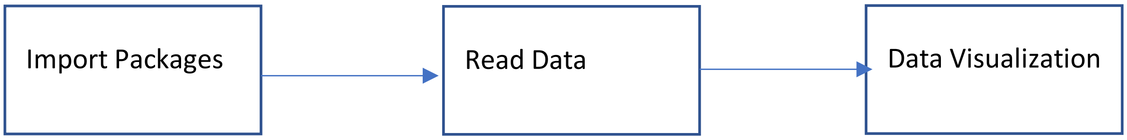
8 rows × 21 columns

```
In [11]: df.isnull()
Out[11]:
```

	Unnamed: 0	STATE_UT_NAME	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	...	NOV	DEC	ANNUAL	Jan-Feb	Mar-May	Jun-Sep	Oct-Dec	flood	Avg_june1(
0	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	
1	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	
2	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	
3	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	
4	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
636	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	
637	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	
638	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	

Module – 2

- In this module The data will be visualized after cleaning the noisy data

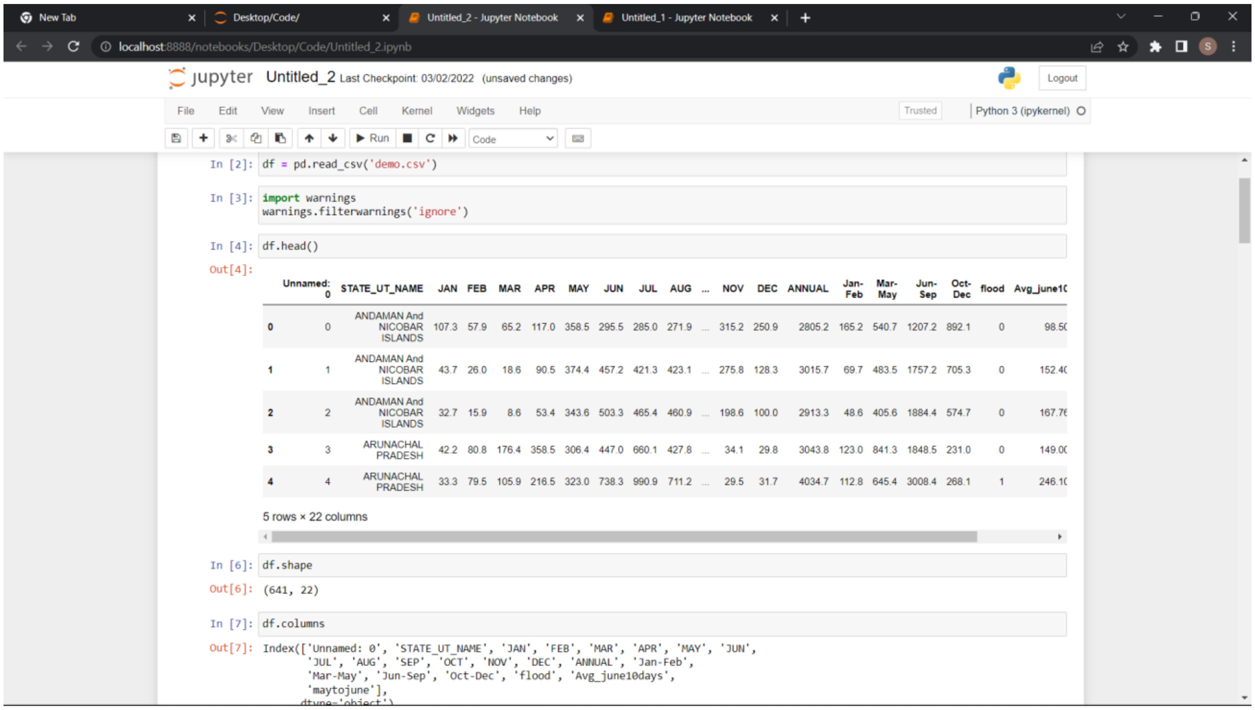


```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns


df = pd.read_csv('demo.csv')
import warnings
warnings.filterwarnings('ignore')


df.head()
df.shape
df.columns
plt.hist(df['STATE_UT_NAME'],color="red")
plt.xlabel("STATE_UT_NAME")
plt.show()
plt.hist(df['ANNUAL'],color="blue")
plt.xlabel("ANNUAL")
plt.show()
plt.figure(figsize=(10,6))
plt.plot(df['ANNUAL'])
plt.show()
```

```
fig, ax = plt.subplots(figsize=(18,13))
sns.heatmap(df.corr(), ax=ax, annot=True)
boxplot = df.boxplot(column=['ANNUAL'], by="flood",figsize=(8,6),  fontsize=15)
plt.show()


def PropByVar(df, variable):
    dataframe_pie = df[variable].value_counts()
    ax = dataframe_pie.plot.pie(figsize=(8,12), autopct='%1.2f%%', fontsize = 12)
    ax.set_title(variable + ' \n', fontsize = 15)
    plt.show()
    return np.round(dataframe_pie/df.shape[0]*100,2)
PropByVar(df,"flood")
plt.show()


plt.figure(figsize=(12,8))
plt.scatter(df["ANNUAL"],df["flood"],color="black")
data1=df.iloc[:,:9]
data2=df.iloc[:,9:-1]
a=pd.plotting.scatter_matrix(data1, alpha=0.05, figsize=(18,10), diagonal='hist')
axis = 'off'

plt.show()
```

Input: Data

Output: Data Visualization


New Tab
Desktop/Code/
Untitled_2 - Jupyter Notebook
Untitled_1 - Jupyter Notebook
localhost:8888/notebooks/Desktop/Code/Untitled_2.ipynb
jupyter
Untitled_2 Last Checkpoint: 03/02/2022 (unsaved changes)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3 (ipykernel)
Run
Code

```
In [2]: df = pd.read_csv('demo.csv')
```

```
In [3]: import warnings
        warnings.filterwarnings('ignore')
```

```
In [4]: df.head()
```

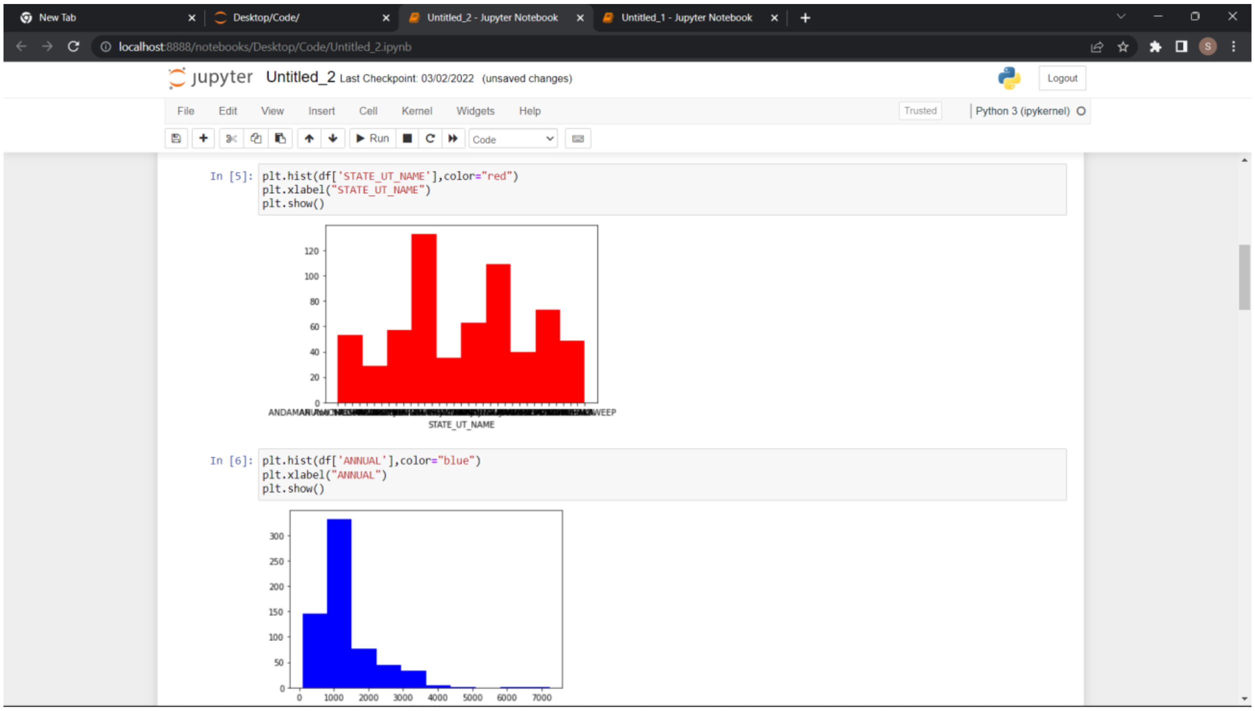
Out[4]:

	Unnamed: 0	STATE_UT_NAME	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	...	NOV	DEC	ANNUAL	Jan-Feb	Mar-May	Jun-Sep	Oct-Dec	flood	Avg_june1(
0	0	ANDAMAN And NICOBAR ISLANDS	107.3	57.9	65.2	117.0	358.5	295.5	285.0	271.9	...	315.2	250.9	2805.2	165.2	540.7	1207.2	892.1	0	98.5(
1	1	ANDAMAN And NICOBAR ISLANDS	43.7	26.0	18.6	90.5	374.4	457.2	421.3	423.1	...	275.8	128.3	3015.7	69.7	483.5	1757.2	705.3	0	152.4(
2	2	ANDAMAN And NICOBAR ISLANDS	32.7	15.9	8.6	53.4	343.6	503.3	465.4	460.9	...	198.6	100.0	2913.3	48.6	405.6	1884.4	574.7	0	167.7(
3	3	ARUNACHAL PRADESH	42.2	80.8	176.4	358.5	306.4	447.0	660.1	427.8	...	34.1	29.8	3043.8	123.0	841.3	1848.5	231.0	0	149.0(
4	4	ARUNACHAL PRADESH	33.3	79.5	105.9	216.5	323.0	738.3	990.9	711.2	...	29.5	31.7	4034.7	112.8	645.4	3008.4	268.1	1	246.1(

5 rows × 22 columns

```
In [6]: df.shape
Out[6]: (641, 22)
```

```
In [7]: df.columns
Out[7]: Index(['Unnamed: 0', 'STATE_UT_NAME', 'JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN',
               'JUL', 'AUG', 'SEP', 'OCT', 'NOV', 'DEC', 'ANNUAL', 'Jan-Feb',
               'Mar-May', 'Jun-Sep', 'Oct-Dec', 'flood', 'Avg_june10days',
               'maytojune'],
              dtype='object')
```


New Tab
Desktop/Code/
Untitled_2 - Jupyter Notebook
Untitled_1 - Jupyter Notebook
localhost:8888/notebooks/Desktop/Code/Untitled_2.ipynb
jupyter
Untitled_2 Last Checkpoint: 03/02/2022 (unsaved changes)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3 (ipykernel)
Run
Code

```
In [5]: plt.hist(df['STATE_UT_NAME'],color="red")
        plt.xlabel("STATE_UT_NAME")
        plt.show()
```

```
In [6]: plt.hist(df['ANNUAL'],color="blue")
        plt.xlabel("ANNUAL")
        plt.show()
```