```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import plotly.express as px
%matplotlib inline
import seaborn as sns
sns.set()
```

```
from google.colab import drive
drive.mount('/content/drive')
path = '/content/drive/MyDrive/Colab Notebooks/Latest Covid-19 India Status.csv'
data = pd.read_csv(path)
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
# To display first n rows of data
n = int(input("Enter number of top rows to be viewed:--"))
data.head(n)
```

```
Enter number of top rows to be viewed:--10
```

	State/UTs	Total Cases	Active	Discharged	Deaths	Active Ratio	Discharge Ratio	Death Ratio	Population
0	Andaman and Nicobar	10747	0	10618	129	0.00	98.80	1.20	100896618
1	Andhra Pradesh	2339078	7	2324338	14733	0.00	99.37	0.63	128500364
2	Arunachal Pradesh	66891	0	66595	296	0.00	99.56	0.44	658019
3	Assam	746100	0	738065	8035	0.00	98.92	1.08	290492
4	Bihar	851404	1	839100	12303	0.00	98.55	1.45	40100376
5	Chandigarh	99358	3	98174	1181	0.00	98.81	1.19	30501026
6	Chhattisgarh	1177768	8	1163614	14146	0.00	98.80	1.20	28900667

```
#To display the last n rows of dataset
l = int(input("Enter number of top rows to be viewed:--"))
data.tail(l)
```

```
Enter number of top rows to be viewed:--10
```

	State/UTs	Total Cases	Active	Discharged	Deaths	Active Ratio	Discharge Ratio	Death Ratio	Populatio
26	Puducherry	175636	73	173588	1975	0.04	98.83	1.12	207307
27	Punjab	784282	29	764964	19289	0.00	97.54	2.46	3469887
28	Rajasthan	1315564	5	1305906	9653	0.00	99.27	0.73	152199
29	Sikkim	44321	2	43820	499	0.00	98.87	1.13	8369777
30	Tamil Nadu	3594573	58	3556466	38049	0.00	98.94	1.06	3599875
31	Telengana	841453	27	837315	4111	0.00	99.51	0.49	6959976
32	Tripura	108034	0	107094	940	0.00	99.13	0.87	164605
33	Uttar Pradesh	2128154	18	2104502	23634	0.00	98.89	1.11	115804
34	Uttarakhand	449429	11	441665	7753	0.00	98.27	1.73	8500241

```
#Get overall statisics about the Dataframe
data.describe(include='all')
```

	State/UTs	Total Cases	Active	Discharged	Deaths	Active Ratio	Disc
count	36	3.600000e+01	36.000000	3.600000e+01	36.000000	36.000000	36.0
unique	36	NaN	NaN	NaN	NaN	NaN	
top	Andaman and Nicobar	NaN	NaN	NaN	NaN	NaN	
freq	1	NaN	NaN	NaN	NaN	NaN	
mean	NaN	1.241145e+06	56.527778	1.226346e+06	14742.388889	0.002222	98.8
25%	NaN	1.058650e+05	0.000000	1.048640e+05	1120.750000	0.000000	98.6

Shape of the Dataset

```
data.shape
```

```
(36, 9)
```

Checking the Columns/Features

```
data.columns
```

```
Index(['State/UTs', 'Total Cases', 'Active', 'Discharged', 'Deaths',
       'Active Ratio', 'Discharge Ratio', 'Death Ratio', 'Population'],
      dtype='object')
```

Information about the Dataset

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 36 entries, 0 to 35
Data columns (total 9 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   State/UTs        36 non-null     object 
 1   Total Cases      36 non-null     int64  
 2   Active           36 non-null     int64  
 3   Discharged       36 non-null     int64  
 4   Deaths           36 non-null     int64  
 5   Active Ratio     36 non-null     float64
 6   Discharge Ratio  36 non-null     float64
 7   Death Ratio      36 non-null     float64
 8   Population       36 non-null     int64  
dtypes: float64(3), int64(5), object(1)
memory usage: 2.7+ KB
```

Checking if there is any Duplicate values present in this Dataset

```
data.duplicated().sum()
```

```
0
```

Checking if there is any missing values present in the Dataset

```
data.isnull().sum()
```

```
State/UTs          0
Total Cases        0
Active             0
Discharged         0
Deaths             0
Active Ratio       0
Discharge Ratio    0
Death Ratio        0
```

```
Population         0
dtype: int64
```

Check the Number of Numerical features in dadaset

```
numerical_features = data.select_dtypes(include = ['int', 'float']).columns.tolist()
print("Number of Numerical features:", len(numerical_features))
print(numerical_features)
```

```
Number of Numerical features: 8
['Total Cases', 'Active', 'Discharged', 'Deaths', 'Active Ratio', 'Discharge Ratio', 'Death Ratio', 'Population']
```

Check the Number of Categorical features in dadaset

```
categorical_features = data.select_dtypes(include = ['object']).columns.tolist()
print("Number of Categorical features:", len(categorical_features))
print(categorical_features)
```

```
Number of Categorical features: 1
['State/UTs']
```

Checking the Mean, Median , Max, Min in Number of Dataset

```
print("mean Total Cases:", round(data["Total Cases"].mean()))
```

```
mean Total Cases: 1241145
```

```
print("median Total Cases:", round(data["Total Cases"].median()))
```

```
median Total Cases: 612772
```

```
print("max Total Cases :", round(data["Total Cases"].max()))
```

```
max Total Cases : 8136945
```

```
print("min Total Cases:", round(data["Total Cases"].min()))
```

```
min Total Cases: 10747
```

Checking the Death Ratio in Dataset

```
data["Death Ratio"]. value_counts()
```

```
1.20    3
1.02    2
1.11    1
0.87    1
0.49    1
1.06    1
1.13    1
0.73    1
2.46    1
1.12    1
0.69    1
2.17    1
0.30    1
1.68    1
1.54    1
1.82    1
0.46    1
0.63    1
0.79    1
1.05    1
0.99    1
1.00    1
1.35    1
1.01    1
0.86    1
```

```
1.55    1
1.32    1
0.03    1
1.19    1
1.45    1
1.08    1
0.44    1
1.73    1
Name: Death Ratio, dtype: int64
```

Checking the Active cases in Dataset

```
data["Active"]. value_counts()
```

```
0       10
1        3
10       2
11       2
2        2
134      1
18       1
27       1
58       1
5        1
29       1
73       1
84       1
1300     1
7        1
123      1
14       1
38       1
15       1
8        1
3        1
50       1
Name: Active, dtype: int64
```

Checking the State/UTs in Dataset

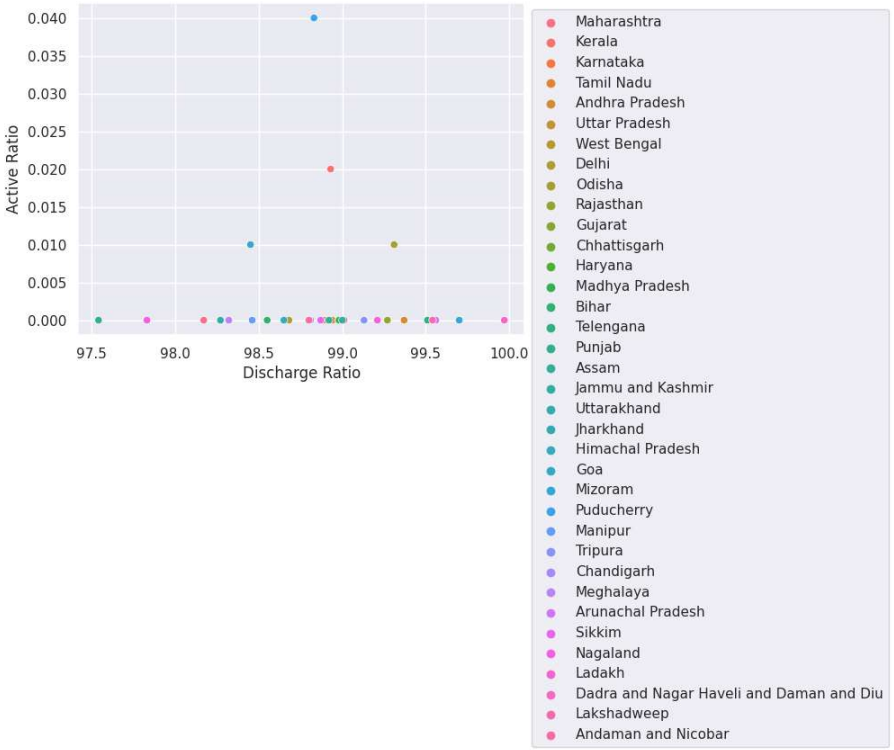
```
data["State/UTs"]. value_counts()
```

```
Andaman and Nicobar                         1
Andhra Pradesh                              1
Maharashtra                                 1
Manipur                                     1
Meghalaya                                   1
Mizoram                                     1
Nagaland                                    1
Odisha                                      1
Puducherry                                  1
Punjab                                      1
Rajasthan                                   1
Sikkim                                      1
Tamil Nadu                                  1
Telengana                                   1
Tripura                                     1
Uttar Pradesh                               1
Uttarakhand                                 1
Madhya Pradesh                              1
Lakshadweep                                 1
Ladakh                                      1
Delhi                                       1
Arunachal Pradesh                           1
Assam                                       1
Bihar                                       1
Chandigarh                                  1
Chhattisgarh                                1
Dadra and Nagar Haveli and Daman and Diu    1
Goa                                         1
Kerala                                      1
Gujarat                                     1
Haryana                                     1
Himachal Pradesh                            1
Jammu and Kashmir                           1
Jharkhand                                   1
Karnataka                                   1
```

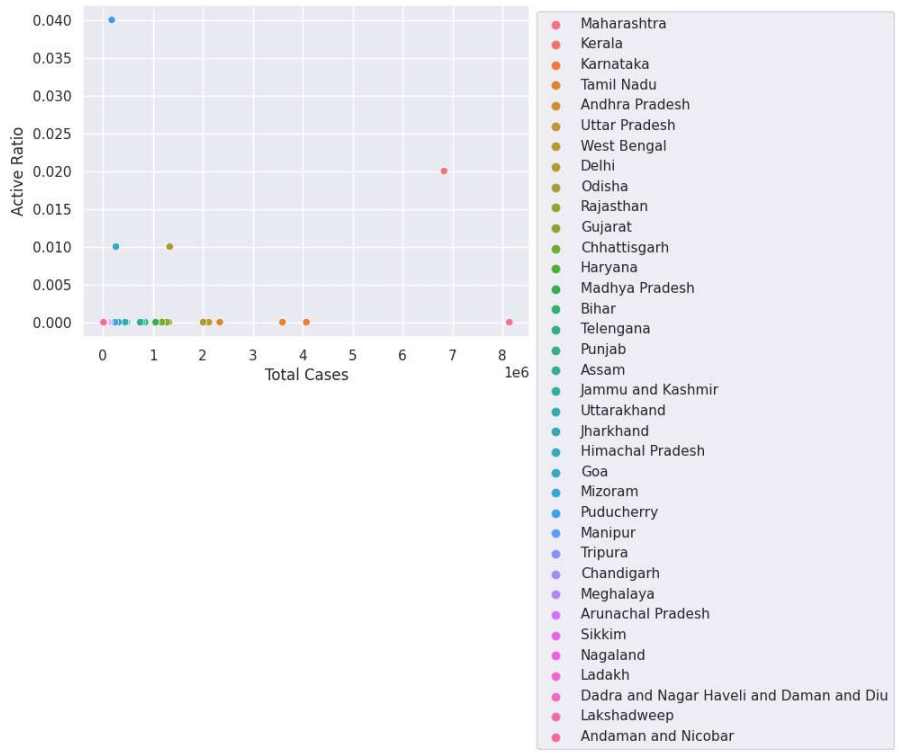
```
West Bengal                                1
Name: State/UTs, dtype: int64
```

Scatter plot

```
sns.scatterplot(y='Active Ratio', x='Discharge Ratio',hue='State/UTs', data=data, )

# Placing Legend outside the Figure
plt.legend(bbox_to_anchor=(1, 1), loc=2)

plt.show()
```

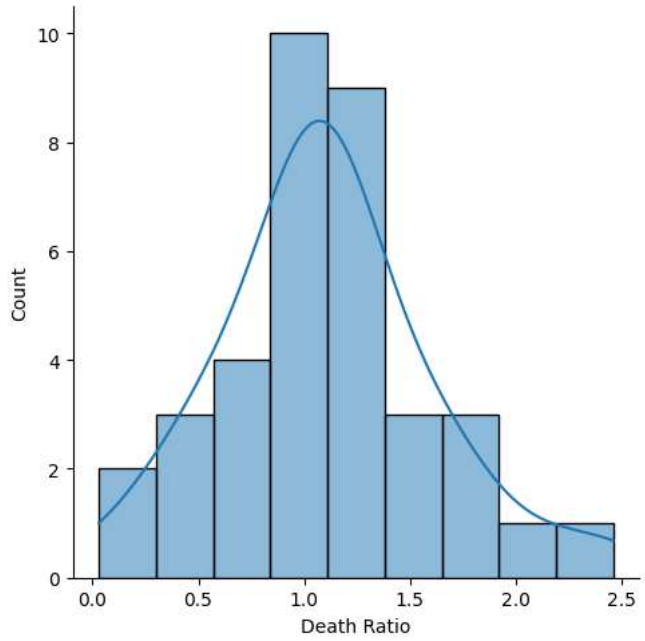



```
sns.scatterplot(y='Active Ratio', x='Total Cases',hue='State/UTs', data=data, )

# Placing Legend outside the Figure
plt.legend(bbox_to_anchor=(1, 1), loc=2)

plt.show()
```



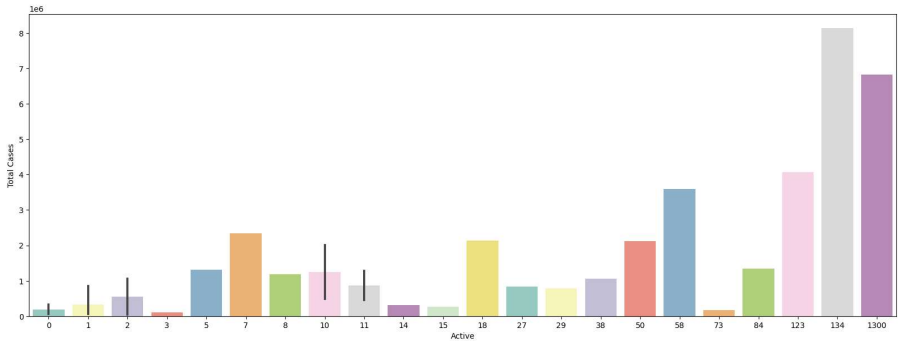
Show Death Ratio in dataset through displot function (with Kde)

```
sns.displot(data['Death Ratio'],kde=True)
plt.show()
```



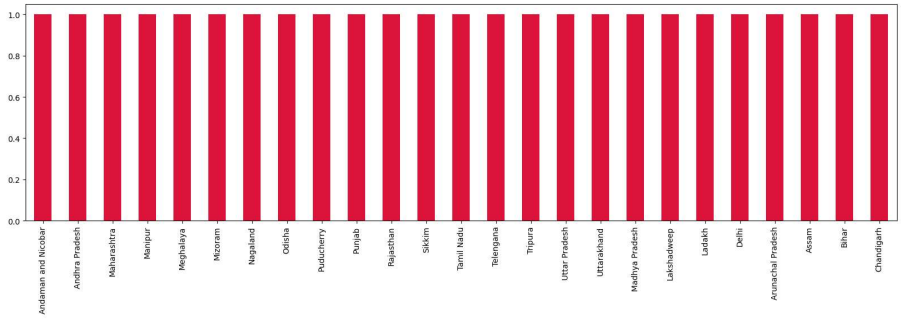
BARPLOT FOR Total Cases VS Active Cases

```
plt.figure(figsize = (20, 7))
sns.barplot(x = data["Active"], y = data["Total Cases"], palette = "Set3");
```



BARPLOT FOR NUMBER OF DIFFERENT State/UTs

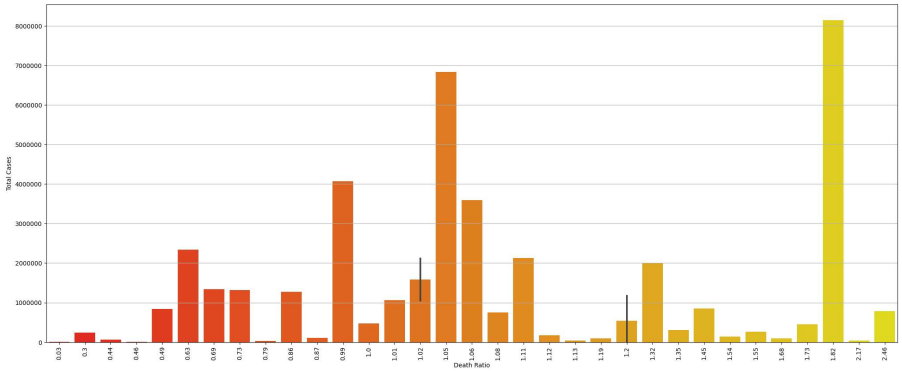
```
plt.figure(figsize = (20, 5))
ax = data["State/UTs"].value_counts()[:25].plot(kind = 'bar',
                                                color = "crimson")

for p in ax.patches:
    ax.annotate(int(p.get_height()), (p.get_x() + 0.25, p.get_height() + 1), ha = 'center', va = 'bottom', color = 'black')
```



BARPLOT FOR Total Cases VS Death Ratio

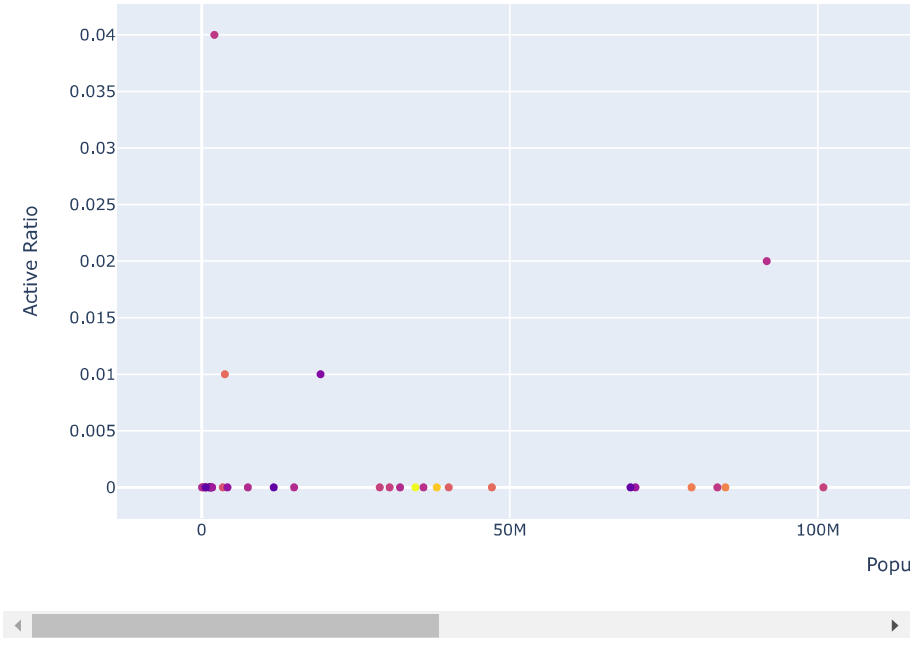
```
plt.figure(figsize = (25, 10))
plt.grid()
plt.xticks(rotation = 90)
plt.ticklabel_format(style = 'plain')
data.sort_values("Total Cases", axis = 0,
```

```
                        ascending = False,
                        inplace = True)
sns.barplot(x = data["Death Ratio"][:100],
            y = data["Total Cases"],
            palette = "autumn");
```



CONTRAST SCATTOR PLOT ON Population VS Active Ratio VS Death Ratio

```
fig = px.scatter(data, x = "Population", y = "Active Ratio", color = "Death Ratio")
fig.show()
```




```
sns.pairplot(data,hue="State/UTs", height=6)
```

```
<seaborn.axisgrid.PairGrid at 0x7f7759e9ed40>
```