

FINAL GROUP PROJECT REPORT

**Partial fulfilment for the award of the
Post Graduate Certification
in Data Analytics for Engineers
September 2021**

SUBMITTED BY:

Amal Kannan

Hari Sankar

Himali Patkar

Sumayya PP

Yashvardhan Rana

Under the Guidance of

Miss. Shalini Kumari

Technical Trainer

Edubridge



Post Graduate Certification in Data Analytics for Engineers

Edubridge India

September 2021

AKNOWLEDGMENT

Acknowledgement is not a mere obligation but the epitome of humility and ineptness to all those who have helped in the completion of this project. We are thankful to Miss Shalini Kumari, Edubridge India ASET for their constant guidance and encouragement provided in this endeavour. We also thank all our friends who helped us out in completing this project and helping us to solve various problems encountered during the progress of this project.

ABSTRACT

In this report, we completed the analysis of a single dataset of vaccination data using 5 tools . We got dataset from Kaggle .This vaccination data contains covid vaccination details of India in 202.

Tools used

- Python
- R
- Excel
- Tableau
- SAS

CONTENTS

1. Introduction

2. Analysis

2.1 PYTHON.....

2.2 R.....

2.3 EXCEL.....

2.4 TABLEAU.....

2.5 SAS.....

2. Conclusion

1.INTRODUCTION

A COVID-19 vaccine is a vaccine intended to provide acquired immunity against COVID-19. Prior to the COVID-19 pandemic, work to develop a vaccine against the coronavirus diseases. This dataset contains latest Covid-19 India state-wise vaccination data as on August 7, 2021. This dataset can be used to analyze covid condition in India. This dataset is great for Exploratory Data Analysis.

Dataset Columns :

- State
- Total cases
- active cases
- Discharged cases
- Deaths
- Total vaccination doses
- Dose 1
- Dose 2

2. ANALYSIS

2. 1. PYTHON

Jupyter notebook is used for analysis of data set. Firstly, import all libraries. Then read the data set and done EDA analysis. Then done Machine learning. We used linear regression for ML approach.

import libraries

```
In [99]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

read file

```
In [100]: d=pd.read_csv("C:/Users/AJAS_A37/Desktop/Sumayya\Kaggle_Datsets/vaccination_data_latest.csv")
```

view data

```
In [101]: d
```

```
Out[101]:
```

	State	Total Cases	Active	Discharged	Deaths	Total vaccination doses	Dose 1	Dose 2
0	Andaman And Nicobar	7541	4	7408	129	308179	214334	93845
1	Andhra Pradesh	1978350	20593	1944267	13490	23390769	17319963	6070806
2	Arunachal Pradesh	49668	3032	46399	237	868372	685988	182384
3	Assam	572546	11719	555470	5357	12089630	9977914	2111716
4	Bihar	725122	357	715119	9646	26796305	22501851	4294454
5	Chandigarh	61970	27	61132	811	892796	679894	212902
6	Chhattisgarh	1002958	1780	987642	13536	12448461	9900079	2548382
7	Dadra And Nagar Haveli And Daman And Diu	10652	12	10636	4	653551	576133	77418
8	Delhi	1436623	516	1411042	25065	10559677	7684071	2875606
9	Goa	171705	992	167556	3157	1370190	1081299	288891

Set the index

```
In [102]: #d.set_index("name")
```

give index a name

```
In [103]: d.index.name = "SNo"
d
```

```
Out[103]:
```

SNo	State	Total Cases	Active	Discharged	Deaths	Total vaccination doses	Dose 1	Dose 2
0	Andaman And Nicobar	7541	4	7408	129	308179	214334	93845
1	Andhra Pradesh	1978350	20593	1944267	13490	23390769	17319963	6070806
2	Arunachal Pradesh	49668	3032	46399	237	868372	685988	182384
3	Assam	572546	11719	555470	5357	12089630	9977914	2111716
4	Bihar	725122	357	715119	9646	26796305	22501851	4294454
5	Chandigarh	61970	27	61132	811	892796	679894	212902
6	Chhattisgarh	1002958	1780	987642	13536	12448461	9900079	2548382
7	Dadra And Nagar Haveli And Daman And Diu	10652	12	10636	4	653551	576133	77418
8	Delhi	1436623	516	1411042	25065	10559677	7684071	2875606
9	Goa	171705	992	167556	3157	1370190	1081299	288891
10	Gujarat	825001	204	814720	10077	35631476	26979294	8652182
11	Haryana	770042	690	759705	9647	12654504	9866525	2787979
12	Himachal Pradesh	207344	1727	202084	3533	5502831	4155025	1347806
13	Jammu And Kashmir	322286	1404	316496	4386	6603777	5175592	1428185
14	Jharkhand	347336	226	341980	5130	10094104	8159801	1934303
15	Karnataka	2915317	24354	2854222	36741	32350283	25190326	7159957

Desktop/Sumi Files/Final Group | EDA of Vaccination Data - Jupyter (autosaved)

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3

get columns names

```
In [104]: d.columns
```

```
Out[104]: Index(['State', 'Total Cases', 'Active', 'Discharged', 'Deaths',  
              'Total vaccination doses', 'Dose 1', 'Dose 2'],  
              dtype='object')
```

rename the columns names

```
In [105]: d.rename(columns = {"State":"State","Active":"Active"},inplace=True)
```

```
Out[105]:
```

SNo	State	Total Cases	Active	Discharged	Deaths	Total vaccination doses	Dose 1	Dose 2
0	Andaman And Nicobar	7541	4	7408	129	308179	214334	93845
1	Andhra Pradesh	1978350	20593	1944267	13490	23390769	17319963	6070806
2	Arunachal Pradesh	49668	3032	46399	237	868372	685988	182384
3	Assam	572546	11719	555470	5357	12089630	9977914	2111716
4	Bihar	725122	357	715119	9646	26796305	22501851	4294454
5	Chandigarh	61970	27	61132	811	892796	679894	212902
6	Chhattisgarh	1002958	1780	987642	13536	12448461	9900079	2548382
7	Dadra And Nagar Haveli And Daman And Diu	10652	12	10636	4	653551	576133	77418
8	Delhi	1436623	516	1411042	25065	10559677	7684071	2875606
9	Goa	171705	992	167558	3157	1370190	1081299	288891
10	Gujarat	825001	204	814720	10077	35631476	26979294	8652182
11	Haryana	770042	690	759705	9647	12654504	9866525	2787979
12	Himachal Pradesh	207344	1727	202084	3533	5502831	4155025	1347806

Desktop/Sumi Files/Final Group | EDA of Vaccination Data - Jupyter (autosaved)

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3

to print top 5 records

```
In [106]: d.head()
```

```
Out[106]:
```

SNo	State	Total Cases	Active	Discharged	Deaths	Total vaccination doses	Dose 1	Dose 2
0	Andaman And Nicobar	7541	4	7408	129	308179	214334	93845
1	Andhra Pradesh	1978350	20593	1944267	13490	23390769	17319963	6070806
2	Arunachal Pradesh	49668	3032	46399	237	868372	685988	182384
3	Assam	572546	11719	555470	5357	12089630	9977914	2111716
4	Bihar	725122	357	715119	9646	26796305	22501851	4294454

to print top 10 records

```
In [107]: d.head(10)
```

```
Out[107]:
```

SNo	State	Total Cases	Active	Discharged	Deaths	Total vaccination doses	Dose 1	Dose 2
0	Andaman And Nicobar	7541	4	7408	129	308179	214334	93845
1	Andhra Pradesh	1978350	20593	1944267	13490	23390769	17319963	6070806
2	Arunachal Pradesh	49668	3032	46399	237	868372	685988	182384
3	Assam	572546	11719	555470	5357	12089630	9977914	2111716
4	Bihar	725122	357	715119	9646	26796305	22501851	4294454
5	Chandigarh	61970	27	61132	811	892796	679894	212902
6	Chhattisgarh	1002958	1780	987642	13536	12448461	9900079	2548382
7	Dadra And Nagar Haveli And Daman And Diu	10652	12	10636	4	653551	576133	77418

Desktop/Sumi Files/Final Group | x EDA of Vaccination Data - Jupyter | x +

localhost:8888/notebooks/Desktop/Sumi%20Files/Final%20Group%20Project/My-python/EDA%20of%20Vaccination%20Data.ipynb

jupyter EDA of Vaccination Data (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3

to check columns and rows

```
In [109]: d.shape
Out[109]: (36, 8)
```

to check the type of attributes

```
In [110]: d.dtypes
Out[110]: State                object
Total Cases              int64
Active                  int64
Discharged               int64
Deaths                  int64
Total vaccination doses  int64
Dose 1                   int64
Dose 2                   int64
dtype: object
```

to check the number of dimension of the dataframe

```
In [111]: d.ndim
Out[111]: 2
```

to check the size of the data

```
In [112]: d.size
Out[112]: 288
```

to get the index

```
In [113]: d.index
Out[113]: RangeIndex(start=0, stop=36, step=1, name='SNo')
```

to check datatype is empty or not

```
In [114]: d.empty
Out[114]: False
```

print info of dataset

```
In [115]: d.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 36 entries, 0 to 35
Data columns (total 8 columns):
#   Column                Non-Null Count  Dtype
---  -
0   State                  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   Total vaccination doses 36 non-null    int64
6   Dose 1                  36 non-null    int64
7   Dose 2                  36 non-null    int64
dtypes: int64(7), object(1)
memory usage: 2.4+ KB
```


to print statistical info

```
In [116]: d.describe()
Out[116]:
```

	Total Cases	Active	Discharged	Deaths	Total vaccination doses	Dose 1	Dose 2
count	3.600000e+01	36.000000	3.600000e+01	36.000000	3.600000e+01	3.600000e+01	3.600000e+01
mean	8.859829e+05	11448.694444	8.626628e+05	11871.416667	1.382116e+07	1.075671e+07	3.064444e+06
std	1.282199e+06	31839.162320	1.239847e+06	22952.890569	1.475343e+07	1.167673e+07	3.254998e+06
min	7.541000e+03	4.000000	7.408000e+03	4.000000	6.749100e+04	5.077200e+04	1.671900e+04
25%	6.657275e+04	326.000000	6.136675e+04	799.750000	8.866900e+05	6.844645e+05	2.106880e+05
50%	4.599410e+05	1387.000000	4.487250e+05	5243.500000	1.007573e+07	7.365391e+06	2.023010e+06
75%	9.892878e+05	9166.000000	9.721065e+05	13501.500000	2.374833e+07	1.805431e+07	4.802555e+06
max	6.341759e+06	178722.000000	6.130137e+06	133717.000000	5.329553e+07	4.467465e+07	1.184617e+07

```
In [117]: d.describe(include="all")
Out[117]:
```

	State	Total Cases	Active	Discharged	Deaths	Total vaccination doses	Dose 1	Dose 2
count	36	3.600000e+01	36.000000	3.600000e+01	36.000000	3.600000e+01	3.600000e+01	3.600000e+01
unique	36	NaN	NaN	NaN	NaN	NaN	NaN	NaN
top	Uttarakhand	NaN	NaN	NaN	NaN	NaN	NaN	NaN
freq	1	NaN	NaN	NaN	NaN	NaN	NaN	NaN
mean	NaN	8.859829e+05	11448.694444	8.626628e+05	11871.416667	1.382116e+07	1.075671e+07	3.064444e+06
std	NaN	1.282199e+06	31839.162320	1.239847e+06	22952.890569	1.475343e+07	1.167673e+07	3.254998e+06
min	NaN	7.541000e+03	4.000000	7.408000e+03	4.000000	6.749100e+04	5.077200e+04	1.671900e+04
25%	NaN	6.657275e+04	326.000000	6.136675e+04	799.750000	8.866900e+05	6.844645e+05	2.106880e+05
50%	NaN	4.599410e+05	1387.000000	4.487250e+05	5243.500000	1.007573e+07	7.365391e+06	2.023010e+06
75%	NaN	9.892878e+05	9166.000000	9.721065e+05	13501.500000	2.374833e+07	1.805431e+07	4.802555e+06

```
ju raise raise raise raise raise raise raise
31 False False False False False False False False
32 False False False False False False False False
33 False False False False False False False False
34 False False False False False False False False
35 False False False False False False False False
```

sum of missing value

```
In [121]: d.isna().sum()
Out[121]:
```

State	0
Total Cases	0
Active	0
Discharged	0
Deaths	0
Total vaccination doses	0
Dose 1	0
Dose 2	0
dtype: int64	

```
In [122]: d.isnull().sum()
Out[122]:
```

State	0
Total Cases	0
Active	0
Discharged	0
Deaths	0
Total vaccination doses	0
Dose 1	0
Dose 2	0
dtype: int64	

drop the column

```
In [123]: d.drop("Total vaccination doses",axis=1,inplace=True)
```

```
Out[123]:
```

	State	Total Cases	Active	Discharged	Deaths	Dose 1	Dose 2
SNo							
0	Andaman And Nicobar	7541	4	7408	129	214334	93845
1	Andhra Pradesh	1978350	20593	1944267	13490	17319963	6070806
2	Arunachal Pradesh	49668	3032	46399	237	685988	182384
3	Assam	572546	11719	555470	5357	9977914	2111716
4	Bihar	725122	357	715119	9646	22501851	4294454
5	Chandigarh	61970	27	61132	811	679894	212902
6	Chhattisgarh	1002958	1780	987642	13536	9900079	2548382
7	Dadra And Nagar Haveli And Daman And Diu	10652	12	10636	4	576133	77418
8	Delhi	1436623	516	1411042	25065	7684071	2875606
9	Goa	171705	992	167556	3157	1081299	288891
10	Gujarat	825001	204	814720	10077	26979294	8652182
11	Haryana	770042	690	759705	9647	9866525	2787979

to check duplicate value

```
In [124]: d.duplicated()
```

```
Out[124]: SNo
0      False
1      False
2      False
3      False
4      False
5      False
6      False
7      False
8      False
9      False
10     False
11     False
12     False
13     False
14     False
15     False
16     False
17     False
18     False
19     False
20     False
21     False
22     False
23     False
24     False
25     False
26     False
27     False
28     False
29     False
30     False
31     False
```

to check sum of duplicate values

```
In [125]: d.duplicated().sum()
```

```
Out[125]: 0
```

to check number of unique values in a dataset

```
In [126]: d.nunique()
```

```
Out[126]: State      36
Total Cases  36
Active       36
Discharged   36
Deaths       36
Dose 1       36
Dose 2       36
dtype: int64
```

to get the unique values of a column

```
In [127]: d["State"].unique()
```

```
Out[127]: array(['Andaman And Nicobar', 'Andhra Pradesh', 'Arunachal Pradesh',
                'Assam', 'Bihar', 'Chandigarh', 'Chhattisgarh',
                'Dadra And Nagar Haveli And Daman And Diu', 'Delhi', 'Goa',
                'Gujarat', 'Haryana', 'Himachal Pradesh', 'Jammu And Kashmir',
                'Jharkhand', 'Karnataka', 'Kerala', 'Ladakh', 'Lakshadweep',
                'Madhya Pradesh', 'Maharashtra', 'Manipur', 'Meghalaya', 'Mizoram',
                'Nagaland', 'Odisha', 'Puducherry', 'Punjab', 'Rajasthan',
                'Sikkim', 'Tamil Nadu', 'Telangana', 'Tripura', 'Uttar Pradesh',
```

count the the values of a column

```
In [129]: d.State.value_counts()
Out[129]:
```

Uttarakhand	1
Telangana	1
Sikkim	1
Mizoram	1
Andaman And Nicobar	1
Assam	1
Goa	1
Jammu And Kashmir	1
Maharashtra	1
Bihar	1
Manipur	1
Himachal Pradesh	1
Punjab	1
Madhya Pradesh	1
Lakshadweep	1
Karnataka	1
Chandigarh	1
Andhra Pradesh	1
Ladakh	1
Uttar Pradesh	1
Jharkhand	1
Tamil Nadu	1
Gujarat	1
Arunachal Pradesh	1
Rajasthan	1
Meghalaya	1
Delhi	1
Dadra And Nagar Haveli And Daman And Diu	1
Kerala	1
Odisha	1
West Bengal	1

maximum values using sort function

to show highest top 5 records with highest Dose 1 Vaccinated State

```
In [131]: d.sort_values("Dose 1",ascending=False).head(5)
Out[131]:
```

	State	Total Cases	Active	Discharged	Deaths	Dose 1	Dose 2
33	Uttar Pradesh	1708689	619	1685299	22771	44974649	8320884
20	Maharashtra	6341759	77905	6130137	133717	34530719	11846167
19	Madhya Pradesh	791937	158	781265	10514	28719654	5519239
10	Gujarat	825001	204	814720	10077	26979294	8652182
28	Rajasthan	953793	233	944606	8954	26474741	8145985

```
In [132]: d.sort_values("Dose 1",ascending=False).head(1)
Out[132]:
```

	State	Total Cases	Active	Discharged	Deaths	Dose 1	Dose 2
33	Uttar Pradesh	1708689	619	1685299	22771	44974649	8320884

mean

```
In [149]: d.mean()
Out[149]:
```

Total Cases	8.859829e+05
Active	1.144869e+04
Discharged	8.626628e+05
Deaths	1.187142e+04
Dose 1	1.075671e+07
Dose 2	3.064444e+06

dtype: float64

max

```
In [150]: d.max()
Out[150]:
```

State	West Bengal
Total Cases	6341759
Active	178722
Discharged	6130137
Deaths	133717
Dose 1	44974649
Dose 2	11846167

dtype: object

Check if there is exists 0 or negative values in columns

```
In [151]: print((d==0).sum()[(d==0).sum() > 0])
Series([], dtype: int64)
```

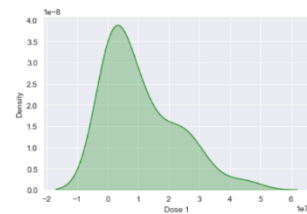
replace null value with zero

```
In [152]: d["Dose 1"].fillna(0,inplace=True)
d
```

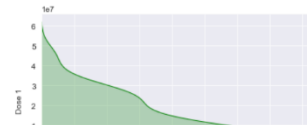
```
Out[152]:
```

SNo	State	Total Cases	Active	Discharged	Deaths	Dose 1	Dose 2
0	Andaman And Nicobar	7541	4	7408	129	214334	93845
1	Andhra Pradesh	1978350	20593	194287	13490	17319963	6070806
2	Arunachal Pradesh	49968	3032	46399	237	685988	182384
3	Assam	572546	11719	555470	5357	9977914	2111716
4	Bihar	725122	357	715119	9646	22501851	4294454
5	Chandigarh	61970	27	61132	811	679894	212902
6	Chhattisgarh	1002958	1780	987642	13536	9900079	2548382
7	Dadra And Nagar Haveli And Daman And Diu	10652	12	10636	4	576133	77418
8	Delhi	1436623	516	1411042	25065	7684071	2875606
9	Goa	171705	992	167556	3157	1081299	288891
10	Gujarat	825001	204	814720	10077	28979294	8652182
11	Haryana	770042	690	759705	9647	9896525	2787979
12	Himachal Pradesh	207344	1727	202084	3533	4155025	1347806

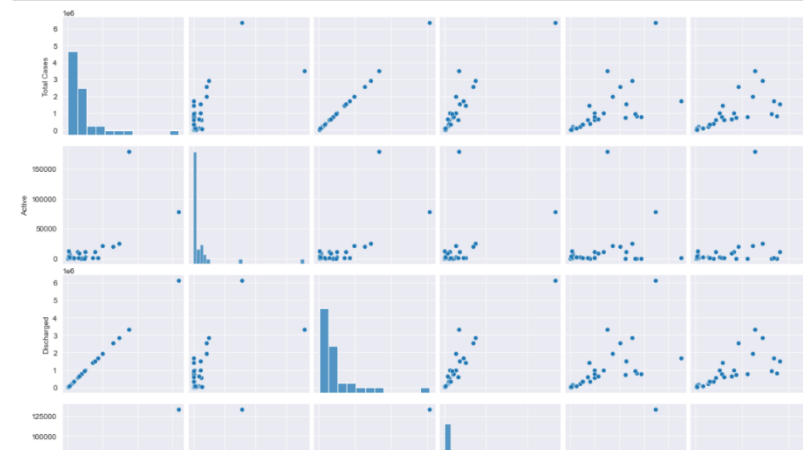
```
In [161]: sns.kdeplot(d["Dose 1"],color="green",shade=True)
plt.show()
```

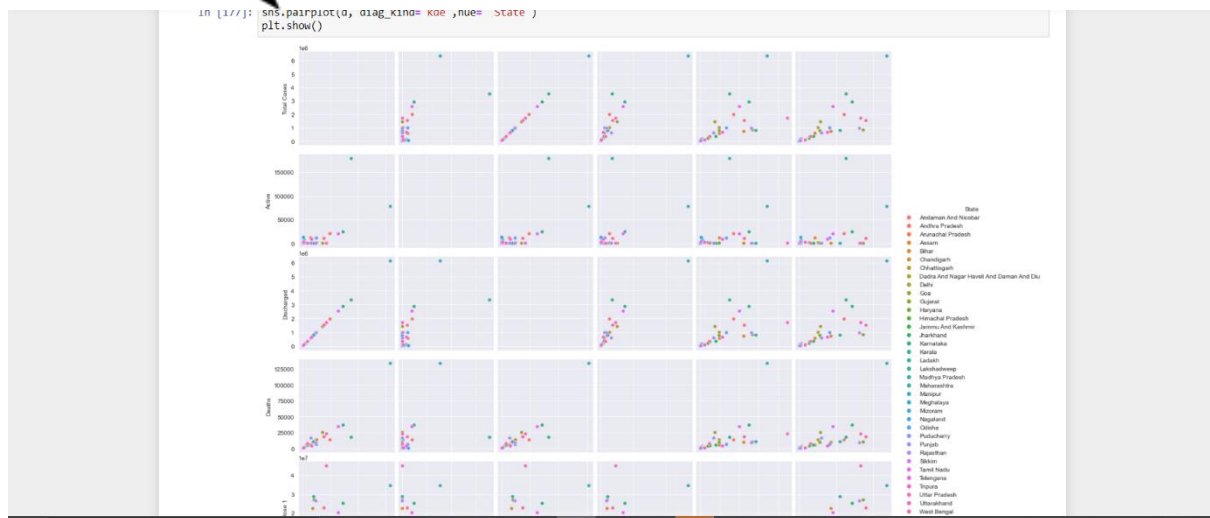


```
In [162]: sns.kdeplot(d["Dose 1"],color="green",vertical=True,shade=True)
plt.show()
```



```
In [164]: sns.pairplot(d)
plt.show()
```



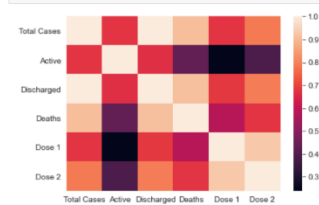


Heat map

correlation

d.corr()

```
In [178]: sns.heatmap(d.corr())
plt.show()
```



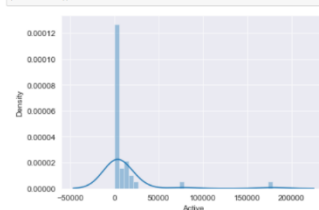
```
In [179]: sns.heatmap(d.corr(), cmap="RdBu")
plt.show()
```



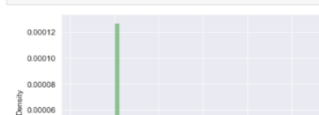
Dist plot

Seaborn distplot lets you show a histogram with a line on it. This can be shown in all kinds of variations. A distplot plot a univariate distribution of observations.

```
In [186]: sns.distplot(d.Active)
plt.show()
```



```
In [187]: sns.distplot(d.Active, color="green")
plt.show()
```

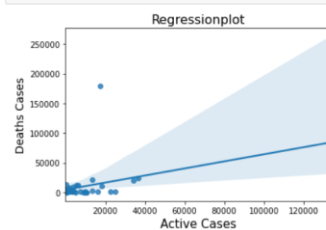


LINEAR REGRESSION:

Linear Regression in Python Linear Regression is a machine learning algorithm based on supervised learning. Linear Regression is a predictive model that is used for finding the linear relationship between a dependent variable and one or more independent variables.

LINEAR REGRESSION

```
In [116]: sns.regplot(y=data["Active"],x=data["Deaths"])
plt.title("Regressionplot",size=16)
plt.xlabel("Active Cases",size=15)
plt.ylabel("Deaths Cases",size=14)
plt.show()
```



```
In [117]: data.corr()
```

	Total Cases	Active	Discharged	Deaths	Total vaccination doses	Dose 1	Dose 2
Total Cases	1.000000	0.678344	0.999882	0.910559	0.709719	0.676837	0.788803
Active	0.678344	1.000000	0.667871	0.430221	0.275594	0.239117	0.391355
Discharged	0.999882	0.667871	1.000000	0.912101	0.715604	0.683056	0.793169
Deaths	0.910559	0.430221	0.912101	1.000000	0.609315	0.581222	0.676718
Total vaccination doses	0.709719	0.275594	0.715604	0.609315	1.000000	0.996723	0.956979

display the shape

```
In [119]: X=data.iloc[:,2:]
Y=data.iloc[:,1]
```

```
In [120]: X.head()
```

	Active	Discharged	Deaths	Total vaccination doses	Dose 1	Dose 2
0	4	7408	129	308179	214334	93845
1	20593	1944267	13490	23390769	17319963	6070806
2	3032	46399	237	868372	685988	182384
3	11719	555470	5357	12089630	9977914	2111716
4	357	715119	9646	26796305	22501851	4294454

```
In [121]: Y
```

```
Out[121]: 0    7541
1   1978350
2    49668
3   572546
4   725122
5    61970
6   1002958
7    10652
8   1436623
9    171705
10  825901
11  770042
12  207344
13  322286
```

import the library to train ,test and split

```
In [122]: from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.3)

In [123]: print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)

(27, 6)
(9, 6)
(27,)
(9,)

In [124]: from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, Y_train)

Out[124]: LinearRegression()

In [125]: Y_pred = regressor.predict(X_test)

In [126]: Y_pred

Out[126]: array([ 68107., 1978350., 725122., 791937., 1002958., 770042.,
121421., 984731., 825001., 648388., 43530.])

In [127]: accuracy = regressor.score(X_test,Y_pred)
"Accuracy: {}".format(int(round(accuracy*100)))

Out[127]: 'Accuracy: 100%'
```

Comparing actual vs predicted

```
In [128]: df = pd.DataFrame({"Actual": Y_test,"Predicted": Y_pred})
df

Out[128]:
```

	Actual	Predicted
22	68107	68107.0
1	1978350	1978350.0
4	725122	725122.0
19	791937	791937.0
6	1002958	1002958.0
11	770042	770042.0
26	121421	121421.0
25	984731	984731.0
10	825001	825001.0
31	648388	648388.0
23	43530	43530.0

Visualising the comparison

```
In [130]: df.plot(kind="bar",figsize=(16,10))
plt.grid(which="major",linestyle="-",linewidth="0.5",color="green")
plt.grid(which="minor",linestyle=":",linewidth="0.5",color="black")
plt.show()
```



2.2. R

Exploratory Data Analysis (EDA) in R is the process of analyzing and visualizing the data to get a better understanding of the data and glean insight from it. There are various steps involved when doing EDA but the following are the common steps that a data analyst can take when performing EDA:

1. Import the data
2. Clean the data
3. Process the data
4. Visualize the data

Here I added the contents of EDA & Linear Regression in R of vaccination data

Import libraries first

```
library(ggplot2)
library(dplyr)
library(choroplethr)
library(choroplethrMaps)
library(openintro)
library(tidyverse)
library(scales)
```



```
library(corrgram)  
print(getwd)
```

```
# read the dataset named vaccination data  
d<-read.csv("C:/Users/Sumi/vaccination_data_latest.csv")  
print(d)
```

```
# print head and tail rows  
print(head(d))  
print(tail(d))
```

```
# summary of the dataset  
print(summary(d))  
print(summary(d$Total.Cases))  
plot(d$Total.Cases)
```

```
# dimation of dataset  
print(dim(d))
```

```
# column names of the dataset  
print(names(d))
```

```
# details of death  
print(d$Deaths)
```

```
# length of the dataset  
print(length(d$Active))
```

```
# structure of the dataset
```

```
print(str(d))
```

```
# glimpse of the dataset
```

```
print(glimpse(d))
```

```
# check unique values
```

```
print(unique(d))
```

```
# statistical values
```

```
print(is.na(d))
```

```
print(is.data.frame(d))
```

```
print(is.name(d))
```

```
print(ncol(d))
```

```
print(nrow(d))
```

```
print(max(d$Active))
```

```
print(min(d$Active))
```

```
print(sort(d$Active))
```

```
print(which.max(d$Active))
```

```
print(which.min(d$Active))
```

```
print(mean(d$Active))
```

```
print(mean(d$Active,trim=0.10))
```

```
print(var(d$Active))
```

```
print(median(d$Active))
print(mad(d$Active))# mean absolute deviation
print(sd(d$Active))
print(mode(d$Active))
print(range(d$Active))
print(scale(d$Active))
print(sd(d$Total.Cases/sqrt(length(d$Active))))
print(max(d$Total.Cases-min(d$Active)))
print(quantile(d$Active))
print(quantile(d$Active,c(0.75)))
print(IQR(d$Active))
print(t.test(d$Active))

# data visualisation

# plotting of total cases
plot(d$Total.Cases,col="red",xlab="X-axis",ylab="Y-axis",main="total
cases")

# plotting of total vaccination doses
plot(d$Total.vaccination.doses,col="red",xlab="X-axis",ylab="Y-
axis",main="total vaccination doses taken")

# first dose and second dose vaccination
```

```
plot(x=d$Dose.1,y=d$Dose.2,main="first and second dose  
vaccines",xlab="dose1",ylab="dose2",col="blue")
```

```
# geographical plot of states related to total cases
```

```
statewise_totalcase=d %>% group_by(State) %>%  
summarise(Total.Cases)
```

```
View(statewise_totalcase)
```

```
# geographical plot of states related to active cases
```

```
statewise_activecases=d %>% group_by(State) %>% summarise(Active)
```

```
View(statewise_activecases)
```

```
# statewise total vaccination
```

```
statewise_vaccination=d %>% group_by(State) %>%  
summarise(Total.vaccination.doses) %>% arrange((desc  
                                                    (Total.vaccination.doses)))
```

```
View(statewise_vaccination)
```

```
# statewise vaccination details(dose1 and dose2)
```

```
statewise_vaccination_doses=d %>% group_by(State) %>%  
summarise(Dose.1,Dose.2)
```

```
View(statewise_vaccination_doses)
```

```
# statewise covid details(total,active and discharged cases)
```

```
statewise_details=d %>% group_by(State) %>%  
summarise(Total.Cases,Active,Discharged)
```

```
View(statewise_details)
```

```
# statewise dose 1 vaccination using ggplot
```

```
statewisedose1=ggplot(d,aes(x=Dose.1,y=State,fill=Dose.1))+geom_col()  
print(statewisedose1)
```

```
# statewise dose 2 vaccination using bargraph
```

```
statewisedose2=ggplot(d,aes(x=Dose.2,y=State,fill=Dose.2))+geom_col(  
)  
print(statewisedose2)
```

```
# statewise total vaccination doses using scatter plot
```

```
statewisetotalvaccination=ggplot(d,aes(x=Total.vaccination.doses,y=State,  
fill=Total.vaccination.doses))+geom_point()  
print(statewisetotalvaccination)
```

```
# statewise total cases using scatter plot
```

```
statewisetotalcases=ggplot(d,aes(x=Total.Cases,y=State,fill=Total.Cases)  
) +geom_point()
```

```
print(statewisetotalcases)
```

```
# statewise active cases in dotplot
```

```
statewiseactivecases=ggplot(d,aes(x=Active,y=State,fill=Active))+geom_  
jitter()
```

```
print(statewiseactivecases)
```

```
# active cases compared to total cases using lineplot
```

```
totalcases=ggplot(d,aes(x=Active,y=Total.Cases,fill=Active))+geom_line  
()
```

```
print(totalcases)
```

```
# statewise active cases using boxplot
```

```
totalcases=ggplot(d,aes(x=Active,y=State,fill=Active),size=3.0)+geom_b  
oxplot()
```

```
print(totalcases)
```

```
# active cases compared to total cases using barplot
```

```
vaccination<-table(d$Active,d$Total.Cases)
```

```
barplot(vaccination,main='active cases compared to total  
cases',xlab='totalcases',ylab='active')
```

```
# discharged static using histogram  
hist(d$Discharged,col='steelblue',main='discharged  
patients',xlab='discharged')
```

```
# plotting active cases vs the total cases:  
print(plot(Active~Total.Cases,data=d))
```

```
# linear regression:  
lr=lm(Active~Total.Cases,data=d)  
print(lr)
```

```
summary(lr)
```

```
par(mfrow=c(1,1))  
plot(lr)
```

```
i<-ggplot(d,aes(x=Active,y=Total.Cases))+geom_point()  
print(i)
```

```
i<-i+geom_smooth(method="lm",col="blue")  
print(i)
```

2.3 Excel

A dashboard is a visual representation of key metrics that allow you to quickly view and analyze your data in one place. Dashboards not only provide consolidated data views, but a self-service business intelligence opportunity, where users are able to filter the data to display just what's important to them.

We have created this excel dashboard on India Vaccination Data to show the contrast between active, discharged and total cases. The dashboard also provides information about the total vaccination doses provided to the residents of the country.

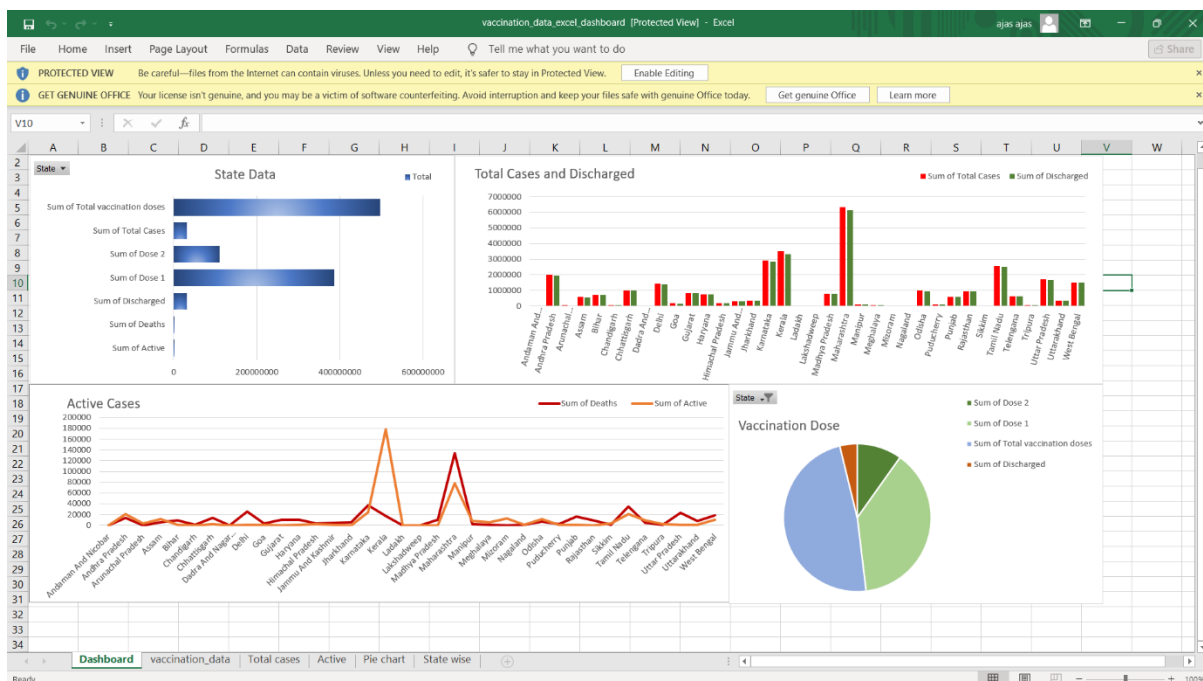


Figure: India Vaccination Dashboard

State Data

This is a bar graph which provides overall analysis state wise. We can choose any Indian state from top left corner and the bars will change accordingly. The chart represents total vaccination doses provided, total cases, sum of dose 2 and dose 1 provided, discharged and active cases and total deaths occurred in the particular state.

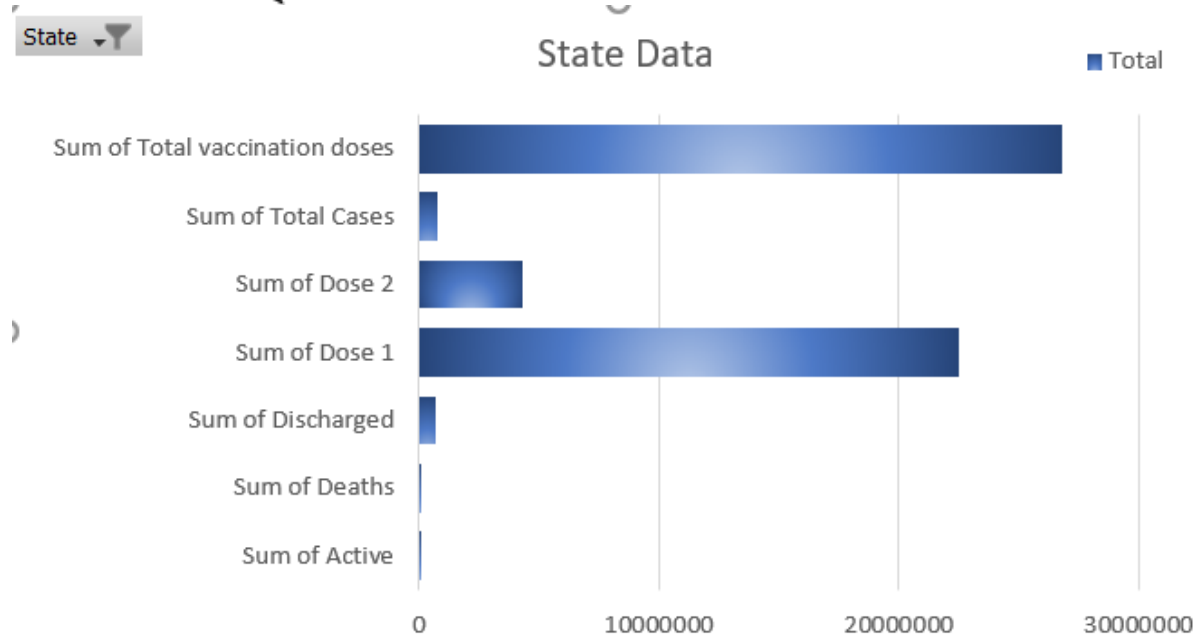


Figure: Bar chart on Delhi state data

We can also select more than 1 state and the bar chart will show the data of those particular states. Through this we can compare the north state data with other south, east, west side data. We can also analyse data with aspects.

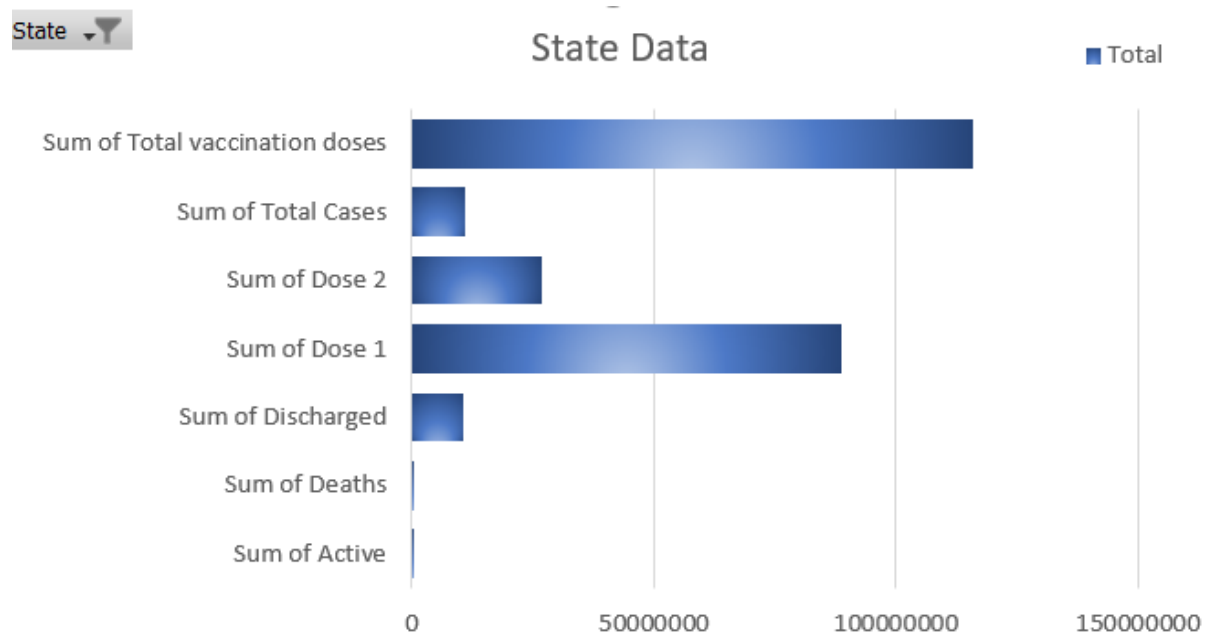


Figure: Bar chart on South states

Total cases and Discharged cases

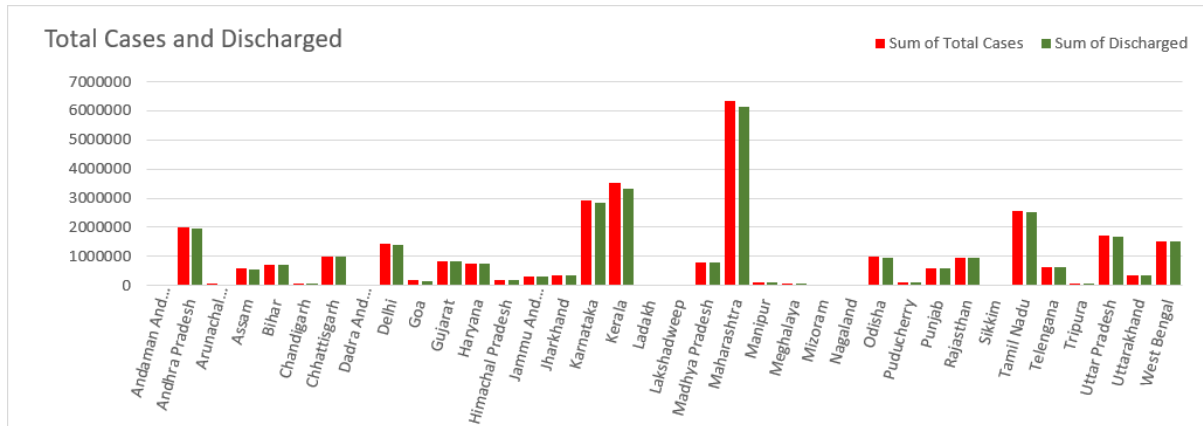


Figure: Total cases of all states

This Clustered column chart which represents total number of cases and total discharged cases in all the states. The red candles represent total number of cases and green candles represent total discharged cases. Through this chart we can tell that Maharashtra has faced the greatest number of cases.

Active Cases and Death

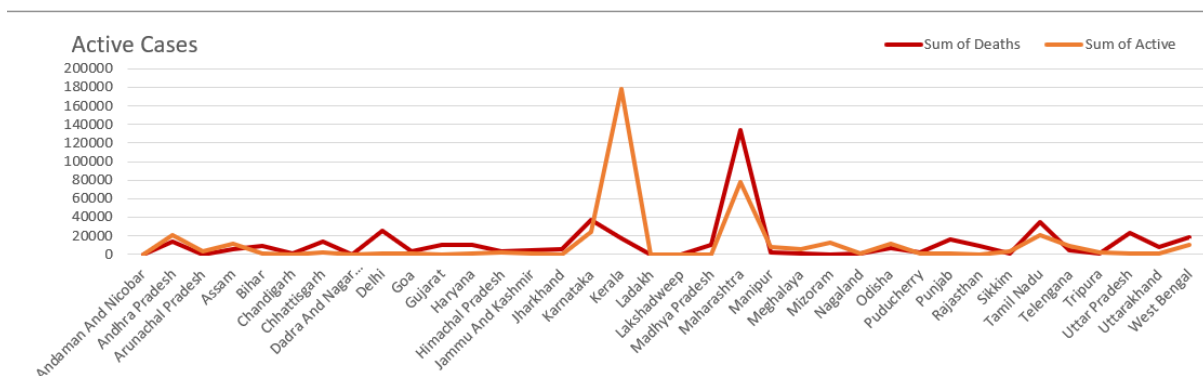


Figure: Active cases and Deaths

The above line chart shows the sum of active cases and the total number of deaths occurred in all the states. The orange line represents the active cases and the red line represents the total deaths.

Vaccination Dose

We have created a pie chart which shows total number of vaccinations provided done in every state. This chart also shows the people who has received dose 1 and dose 2 separately. It shows data state wise and will have option to select multiple states.

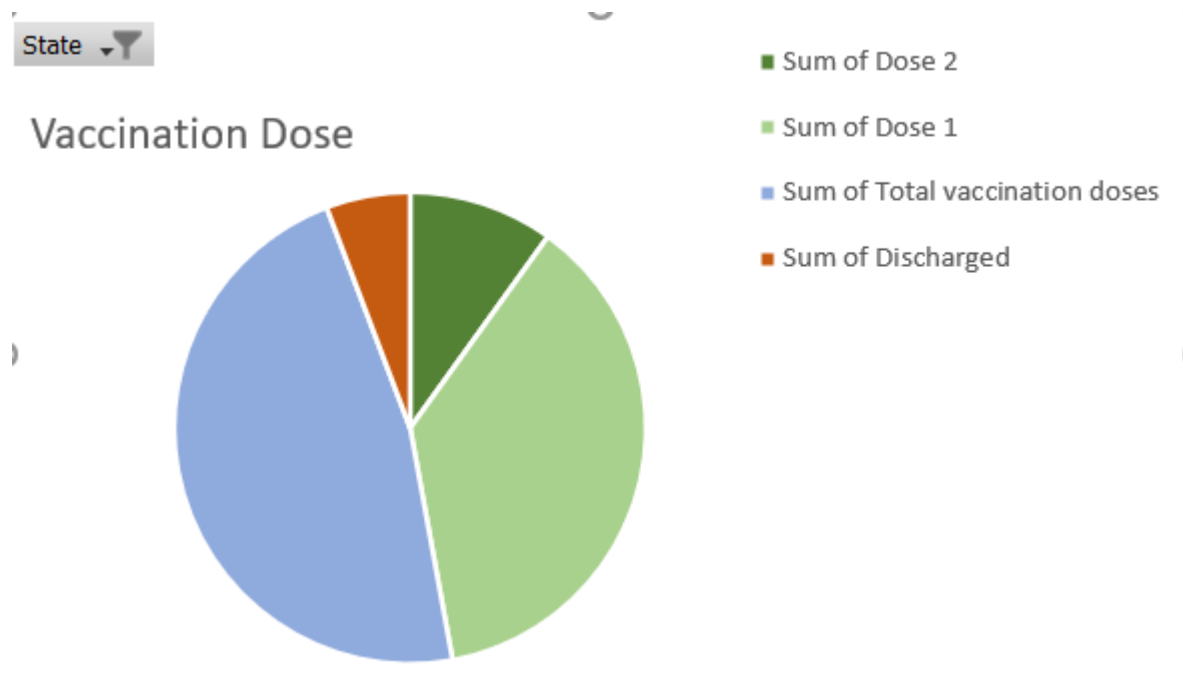
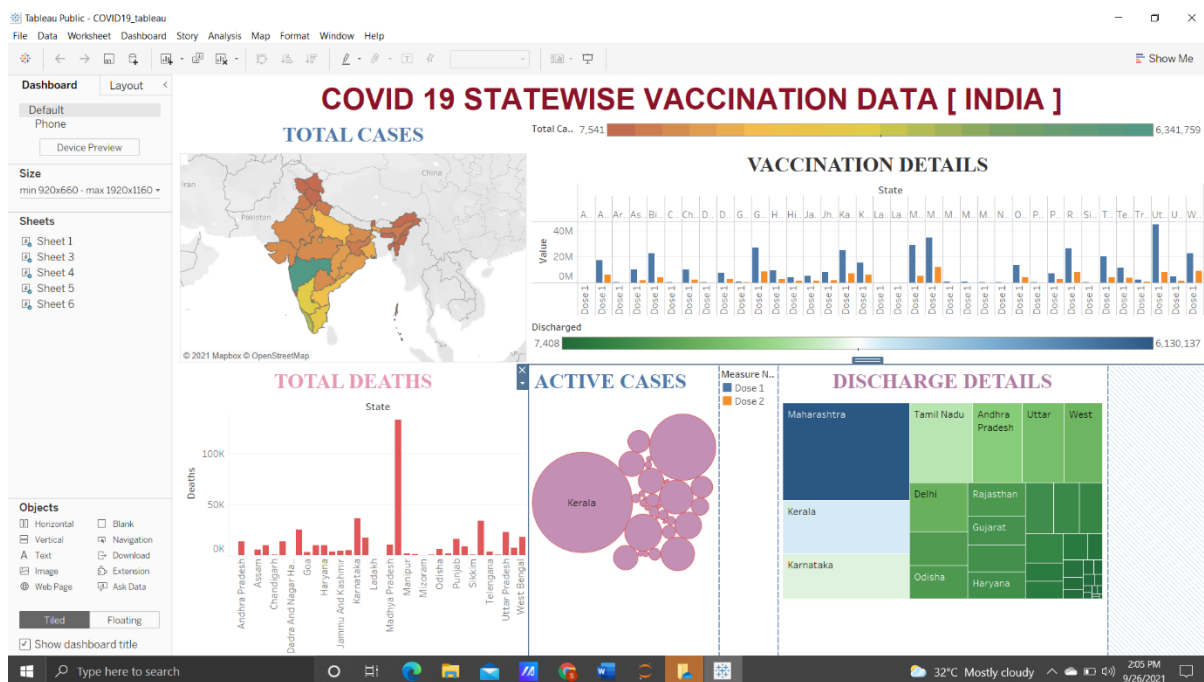


Figure: Goa vaccination data

2.4 Tableau

Tableau is a powerful and fastest growing data visualization tool used in the Business Intelligence Industry. It helps in simplifying raw data in a very easily understandable format. Tableau helps create the data that can be understood by professionals at any level in an organization. It also allows non-technical users to create customized dashboards. Data analysis is very fast with Tableau tool and the visualizations created are in the form of dashboards and worksheets.



2.5. SAS

Exploratory Data Analysis in SAS

```
data vaccination_data_latest;
infile 'vaccination_data_latest.csv' dlm=',' firstobs=2 dsd;
input State$ TotalCase Active Discharged Deaths Totalvaccinationdoses
Dose1 Dose2;
run;

proc print data=vaccination_data_latest;
run;

/* To check is there any missing values present in table*/
proc means data=vaccination_data_latest nmiss;
run;

/* To check the summary of the data*/
proc summary data=vaccination_data_latest print n mean median
mode stddev min max;
var TotalCase Active Discharged Deaths Totalvaccinationdoses Dose1
Dose2 ;
run;

/* To check the correlation between columns */
proc corr data=vaccination_data_latest;
run;

/* To find information of data */
proc contents data=vaccination_data_latest ;
run;
```

```
/* To compare the Median and Maximum values of both dose 1 and dose
2 */
```

```
proc means data=vaccination_data_latest(where=(Outcome=1)) print
median max;
```

```
var TotalCase Active Discharged Deaths Totalvaccinationdoses Dose1
Dose2 ;
```

```
title "vaccination_data_latest";
```

```
proc means data=vaccination_data_latest(where=(Outcome=0)) print
median max;
```

```
var TotalCase Active Discharged Deaths Totalvaccinationdoses Dose1
Dose2 ;
```

```
title "vaccination_data_latest";
```

```
run;
```

```
/* ..... */
```

```
/* Create a sql for storing vaccination details */
```

```
proc sql;
```

```
create table vaccinationdetails as
```

```
select * from vaccination_data_latest;
```

```
quit;
```

```
proc print data= vaccinationdetails;
```

```
run;
```

```
/*Select total vaccination details of State */
```

```
proc sql;
```

```
select State,Totalvaccinationdoses
```

```
from vaccination_data_latest
```

```
;
```

```
quit;
```

```
/* Maximum number of Totalvaccinationdoses display */
```

```
proc sql;
```

```
select State,Totalvaccinationdoses from vaccination_data_latest order  
by Totalvaccinationdoses desc
```

```
;
```

```
quit;
```

```
/* Maximum number of Totalvaccinationdoses display */
```

```
proc sql;
```

```
select State,Totalvaccinationdoses from vaccination_data_latest order  
by Totalvaccinationdoses desc
```

```
;
```

```
quit;
```

```
/* Maximum number of Totalcases display */
```

```
proc sql;
```

```
select State,TotalCase from vaccination_data_latest order by TotalCase  
desc
```

```
;
```

```
quit;
```

```
/* Visualization */
```

```
/* Histogram */  
title "Histogram of vaccination data";  
proc sgplot data=vaccination_data_latest;  
histogram Totalvaccinationdoses/group=TotalCase transparency=0.5  
scale=count;  
density Totalvaccinationdoses /type=normal group=TotalCase;  
keylegend /location=inside position =topright across=1;  
run;
```

```
/* Histogram */  
title "Histogram of vaccination data";  
proc sgplot data=vaccination_data_latest;  
histogram Dose1/group=TotalCase transparency=0.5 scale=count;  
density Dose1/type=normal group=TotalCase;  
keylegend /location=inside position =topright across=1;  
run;
```

```
/* Histogram */  
title "Histogram of vaccination data";  
proc sgplot data=vaccination_data_latest;  
histogram Dose2/group=TotalCase transparency=0.5 scale=count;  
density Dose2/type=normal group=TotalCase;  
keylegend /location=inside position =topright across=1;  
run;
```

```
proc sgplot data=vaccination_data_latest;  
scatter x=Totalvaccinationdoses y=TotalCase;
```



```
ellipse x = Active y = Deaths;  
    title 'Scatter plot';
```

```
/* Hbar */  
proc sgplot data=vaccination_data_latest;  
    hbar TotalCase/response=Dose1 stat=mean  
    datalabel datalabelattrs=(weight=bold) fillattrs=(color=cadetblue);  
    title 'Dose 1 vaccination';  
run;
```

- Highest Dose 1 vaccinated state in India in 202 is Uttar Pradesh
- Lowest Dose 1 vaccinated state in India in 202 is Lakshadweep
- Highest total cases in 2020 in India is in Maharashtra State
- Highest discharged persons in 2020 in India is in Maharashtra State
- Highest active cases is found in 2020 in India is in Kerala
- Highest Deaths in India in 2020 is in Madhya Pradesh