import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import plotly

import plotly.express as px

import plotly.graph\_objects as go

import folium

from folium import plugins

plt.rcParams['figure.figsize'] = 10, 12

import warnings

from sklearn.metrics import mean\_squared\_error

warnings.filterwarnings('ignore')

%matplotlib inline

df\_India= pd.read\_csv('covid\_19\_India.csv')

# df\_daywise\_India = pd.read\_excel('per\_day\_cases.xlsx',parse\_dates=True, sheet\_name='India')

India\_coord = pd.read\_excel('Indian Coordinates.xlsx')

print(df\_India.info())

df\_India.head()

df\_India.tail()

df\_India.dtypes

­­­­­­­­­­­­­­­­­­­­­­­

print(India\_coord.info())

India\_coord.head()

def replace\_dash\_with\_zeros(inp):

return int(inp.replace("-","0"))

df\_India.drop(['Sno'],axis=1,inplace=True)

df\_India['Date'] = pd.to\_datetime(df\_India['Date'], format = "%d/%m/%y")

# https://www.stat.berkeley.edu/~s133/dates.html

df\_India['ConfirmedIndianNational'] = df\_India['ConfirmedIndianNational'].apply(replace\_dash\_with\_zeros)

df\_India['ConfirmedForeignNational'] = df\_India['ConfirmedForeignNational'].apply(replace\_dash\_with\_zeros)

df\_India.sort\_values("Confirmed", ascending = False, inplace = True)

df\_India

df\_India.loc[df\_India["ConfirmedForeignNational"] == "-",:]

list(zip(df\_India.columns,df\_India.dtypes,df\_India.isna().sum()))

print(f'We have data available from : {df\_India.Date.min()} to {df\_India.Date.max()}')

df\_India.groupby(["State/UnionTerritory", "Date"]).sum()

States = df\_India['State/UnionTerritory'].unique().tolist()

States

States.remove("Cases being reassigned to states")

States.remove("Unassigned")

States

len(States)

df\_final\_India = pd.DataFrame()

dates = pd.DataFrame({"Date": pd.date\_range(df\_India.Date.min(),df\_India.Date.max())})

for state in States:

all\_dates\_df = pd.merge(dates,

df\_India.loc[df\_India['State/UnionTerritory'] == state,:], on = "Date",

how = "left")

all\_dates\_df['State/UnionTerritory'] = state

all\_dates\_df = all\_dates\_df.fillna(0)

all\_dates\_df['New Cases'] = all\_dates\_df['Confirmed'] - all\_dates\_df['Confirmed'].shift(1)

# print(state)

# display(all\_dates\_df.loc[all\_dates\_df['New Cases'] < 0,:])

df\_final\_India = pd.concat([df\_final\_India, all\_dates\_df],axis = 0)

print("Finally we have a data of Size: ",df\_final\_India.shape)

df\_final\_India.head()

df\_final\_India.dropna(inplace = True)

df\_final\_India.shape

del df\_final\_India['Time']

del df\_final\_India['ConfirmedIndianNational']

del df\_final\_India['ConfirmedForeignNational']

df\_final\_India

df\_final\_India.groupby(["State/UnionTerritory", "Date"]).sum()

df\_final\_India = df\_final\_India.groupby(["State/UnionTerritory", "Date"]).sum().reset\_index()

df\_final\_India

def plot\_pie(active,cured,death,title):

labels = ['Active','Recovered','Died']

sizes = [active,cured,death]

color= ['#66b3ff','green','red']

explode = []

for i in labels:

explode.append(0.05)

plt.figure(figsize= (15,6))

plt.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=9, explode =explode,colors = color)

centre\_circle = plt.Circle((0,0),0.70,fc='white')

fig = plt.gcf()

fig.gca().add\_artist(centre\_circle)

plt.title(title + 'COVID-19 Cases',fontsize = 20)

plt.axis('equal')

plt.tight\_layout()

total\_cases\_india = 0

cured\_cases\_india = 0

death\_cases\_india = 0

active\_cases\_india = 0

state\_df = pd.DataFrame()

for state in States:

one\_state\_df = df\_final\_India.loc[df\_final\_India['State/UnionTerritory'] == state,:]

state\_df = pd.concat([state\_df,pd.DataFrame(one\_state\_df.iloc[-1,:]).T],axis = 0)

total\_cases = one\_state\_df['Confirmed'].values[-1]

cured = one\_state\_df['Cured'].values[-1]

deaths = one\_state\_df['Deaths'].values[-1]

active = total\_cases - cured - deaths

plot\_pie(active, cured, deaths,state)

total\_cases\_india += total\_cases

cured\_cases\_india += cured

death\_cases\_india += deaths

active\_cases\_india += active

state\_df.reset\_index(inplace = True,drop = True)

state\_df

f, ax = plt.subplots(figsize=(12, 28))

data = state\_df[['State/UnionTerritory','Confirmed','Cured','Deaths']]

data.sort\_values('Confirmed',ascending=False,inplace=True)

sns.set\_color\_codes("pastel")

sns.barplot(x="Confirmed", y="State/UnionTerritory", data=data,label="Total", color="red")

sns.set\_color\_codes("muted")

sns.barplot(x="Cured", y="State/UnionTerritory", data=data, label="Cured", color="green")

ax.legend(ncol=5, loc="lower right", frameon=True)

ax.set(ylabel="",xlabel="Cases")

i = 0

for p in ax.patches:

x = p.get\_x() + p.get\_width() + 3

y = p.get\_y() + p.get\_height()/2

if i <= len(States):

ax.annotate(" "\*10 + str(int(p.get\_width())), (x, y))

else:

ax.annotate(int(p.get\_width()), (x, y))

i += 1

f, ax = plt.subplots(figsize=(12, 28))

data = state\_df[['State/UnionTerritory','Confirmed','Cured','Deaths']]

data.sort\_values('Confirmed',ascending=False,inplace=True)

sns.set\_color\_codes("pastel")

sns.barplot(x="Confirmed", y="State/UnionTerritory", data=data,label="Total", color="red")

sns.set\_color\_codes("muted")

sns.barplot(x="Cured", y="State/UnionTerritory", data=data, label="Cured", color="green")

ax.legend(ncol=5, loc="lower right", frameon=True)

ax.set(ylabel="",xlabel="Cases")

total = total\_cases\_india

i = 0

for p in ax.patches:

percentage = '{:.1f}%'.format(100 \* p.get\_width()/total)

x = p.get\_x() + p.get\_width() + 3

y = p.get\_y() + p.get\_height()/2

if i <= len(States):

ax.annotate(" "\*10 + str(percentage), (x, y))

else:

ax.annotate(percentage, (x, y))

i += 1

print("Total infected cases in India: ", total\_cases\_india)

print("Total cured cases in India: ", cured\_cases\_india)

print("Total active cases in India: ", active\_cases\_india)

print("Total death cases in India: ", death\_cases\_india)

plot\_pie(active\_cases\_india, cured\_cases\_india, death\_cases\_india, "India")

India\_coord.rename(columns = {"Name of State / UT" : "State/UnionTerritory"},inplace = True)

set(India\_coord['State/UnionTerritory'].values).symmetric\_difference(set(state\_df['State/UnionTerritory'].values))

India\_coord['State/UnionTerritory'] = India\_coord['State/UnionTerritory'].str.strip()

state\_df['State/UnionTerritory'] = state\_df['State/UnionTerritory'].str.strip()

set(India\_coord['State/UnionTerritory'].values).symmetric\_difference(set(state\_df['State/UnionTerritory'].values))

India\_coord.loc[India\_coord.shape[0]] = ['Gujarat','22.2587','71.1924']

India\_coord

set(India\_coord['State/UnionTerritory'].values).symmetric\_difference(set(state\_df['State/UnionTerritory'].values))

India\_coord['State/UnionTerritory'] = np.where(India\_coord['State/UnionTerritory'] == "Andaman And Nicobar",

"Andaman and Nicobar Islands",India\_coord['State/UnionTerritory'])

India\_coord['State/UnionTerritory'] = np.where(India\_coord['State/UnionTerritory'] == "Union Territory of Jammu and Kashmir",

"Jammu and Kashmir",India\_coord['State/UnionTerritory'])

India\_coord['State/UnionTerritory'] = np.where(India\_coord['State/UnionTerritory'] == "Union Territory of Ladakh",

"Ladakh",India\_coord['State/UnionTerritory'])

India\_coord['State/UnionTerritory'] = np.where(India\_coord['State/UnionTerritory'] == "Orissa",

"Odisha",India\_coord['State/UnionTerritory'])

India\_coord['State/UnionTerritory'] = np.where(India\_coord['State/UnionTerritory'] == "Dadra And Nagar Haveli",

"Dadar Nagar Haveli",India\_coord['State/UnionTerritory'])

set(India\_coord['State/UnionTerritory'].values).symmetric\_difference(set(state\_df['State/UnionTerritory'].values))

df\_full = pd.merge(India\_coord,state\_df,on='State/UnionTerritory').reset\_index(drop = True)

df\_full

map = folium.Map(location=[20, 70], zoom\_start=4,tiles='Stamenterrain')

for lat, lon, value, name in zip(df\_full['Latitude'], df\_full['Longitude'], df\_full['Confirmed'], df\_full['State/UnionTerritory']):

folium.CircleMarker([lat, lon], radius=value\*0.0015, popup = ('<strong>State</strong>: ' + str(name).capitalize() + '<br>''<strong>Total Cases</strong>: ' + str(value) + '<br>'),color='red',fill\_color='red',fill\_opacity=0.3 ).add\_to(map)

map

df\_daywise\_India = df\_final\_India.groupby("Date")['Confirmed','Cured','Deaths',"New Cases"].sum().reset\_index()

df\_daywise\_India

fig = go.Figure()

fig.add\_trace(go.Scatter(x=df\_daywise\_India['Date'], y = df\_daywise\_India['Confirmed'], mode='lines+markers',name='Total Cases'))

fig.update\_layout(title\_text='Trend of Coronavirus Cases in India (Cumulative cases)',plot\_bgcolor='rgb(230, 230, 230)')

fig.show()

fig = px.bar(df\_daywise\_India, x="Date", y="New Cases", barmode='group', height=400)

fig.update\_layout(title\_text='Coronavirus Cases in India on daily basis',plot\_bgcolor='rgb(230, 230, 230)')

fig.show()

fig = px.bar(df\_daywise\_India, x="Date", y="Confirmed", color='Confirmed', orientation='v', height=600,

title='Confirmed Cases in India', color\_discrete\_sequence = px.colors.cyclical.IceFire)

'''Colour Scale for plotly

https://plot.ly/python/builtin-colorscales/

'''

fig.update\_layout(plot\_bgcolor='rgb(230, 230, 230)')

fig.show()

fig = px.line(x=df\_daywise\_India['Date'], y=df\_daywise\_India['New Cases'], labels = {'x': "Dates",'y': "Counts"})

fig.update\_layout( showlegend=False,title\_text="Trend of Coronavirus cases")

fig.update\_layout(plot\_bgcolor='rgb(250, 242, 242)')

fig.show()

forecasting

from fbprophet import Prophet

df = df\_daywise\_India.iloc[:-1,]

df\_train = df.loc[df['Date']<= "2020-05-23",:]

df\_test = df.loc[df['Date'] > "2020-05-23",:]