\*

In [1]:

**import pandas as pd import numpy as np import seaborn as sns**

**import matplotlib.pyplot as plt**

%**matplotlib** inline

**import folium**

**from folium import** plugins

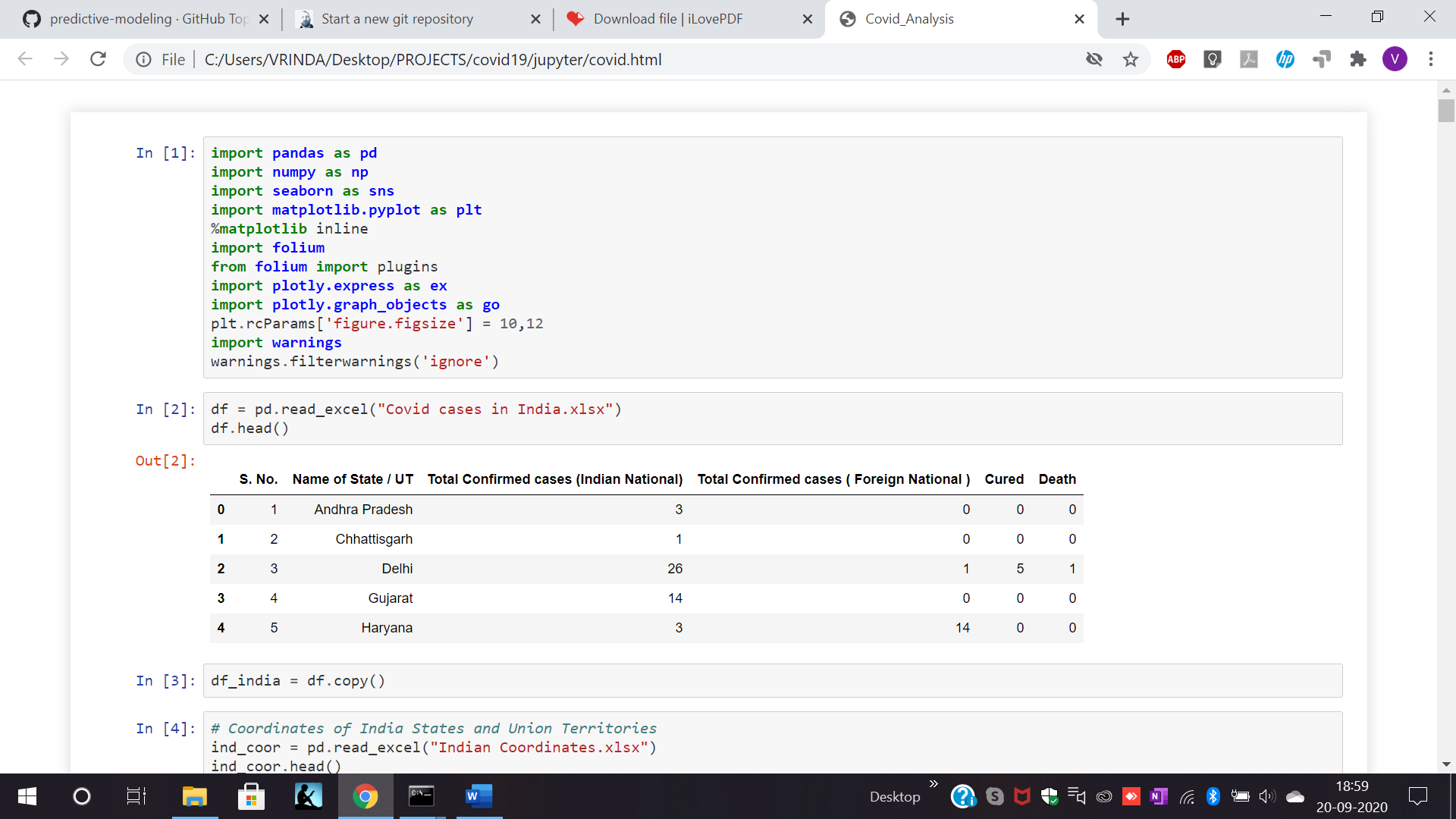
**import plotly.express as ex import plotly.graph\_objects as go**

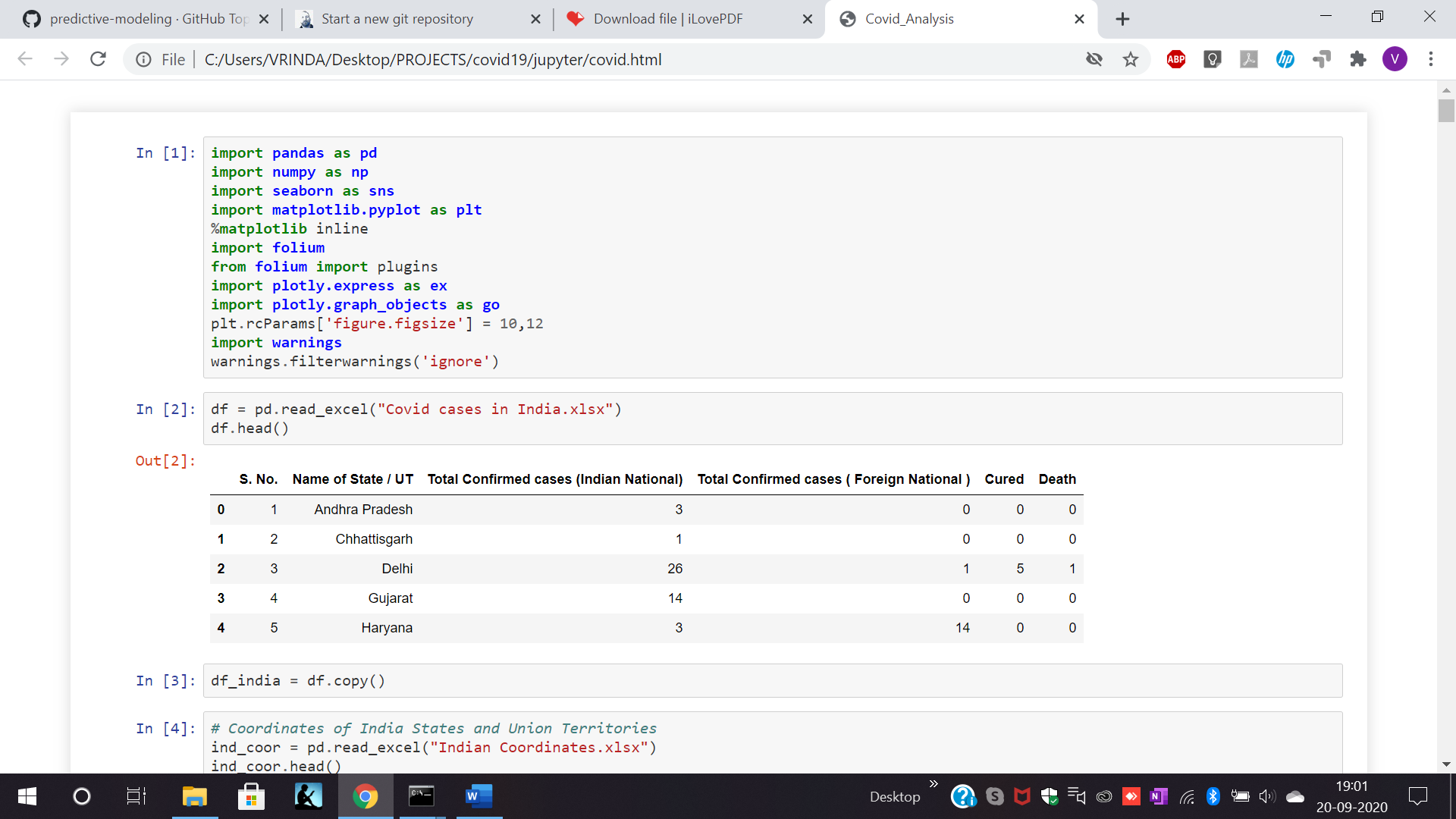
plt.rcParams['figure.figsize'] = 10,12 **import warnings** warnings.filterwarnings('ignore')

In [2]:

df = pd.read\_excel("Covid cases in India.xlsx") df.head()

Out[2]:

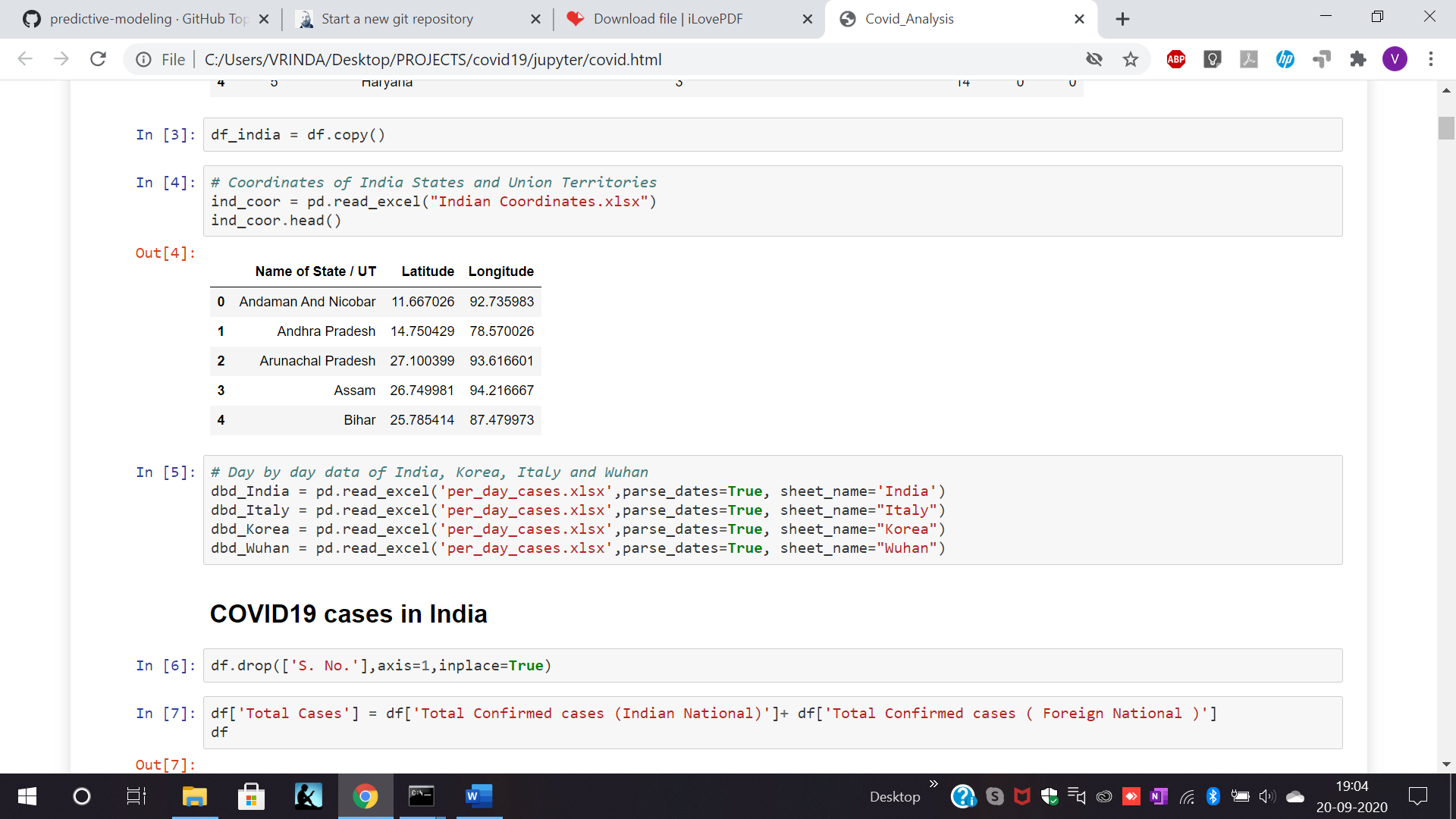


In [3]:

In [4]:

*# Coordinates of India States and Union Territories* ind\_coor = pd.read\_excel("Indian Coordinates.xlsx") ind\_coor.head()

Out[4]:



In [5]:

*# Day by day data of India, Korea, Italy and Wuhan*

dbd\_India = pd.read\_excel('per\_day\_cases.xlsx',parse\_dates=**True**, sheet\_name='India') dbd\_Italy = pd.read\_excel('per\_day\_cases.xlsx',parse\_dates=**True**, sheet\_name="Italy") dbd\_Korea = pd.read\_excel('per\_day\_cases.xlsx',parse\_dates=**True**, sheet\_name="Korea") dbd\_Wuhan = pd.read\_excel('per\_day\_cases.xlsx',parse\_dates=**True**, sheet\_name="Wuhan")

# COVID19 cases in India

In [6]:

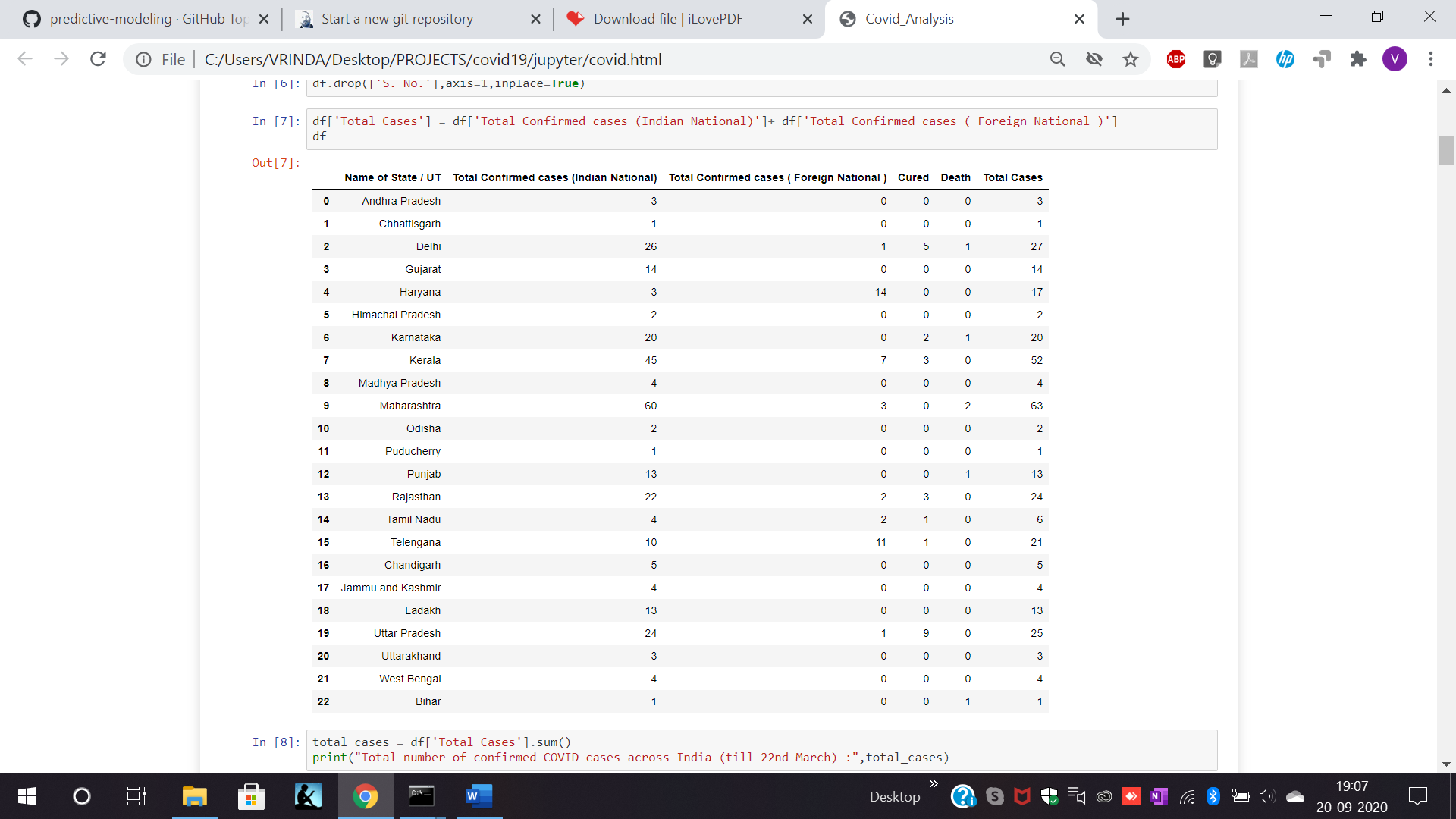
df.drop(['S. No.'],axis=1,inplace=**True**)

In [7]:

df['Total Cases'] = df['Total Confirmed cases (Indian National)']+ df['Total Confirmed cases ( Foreign National )']

df

Out[7]:



In [8]:

total\_cases = df['Total Cases'].sum()

print("Total number of confirmed COVID cases across India (till 22nd March) :",total\_cases)

Total number of confirmed COVID cases across India (till 22nd March) : 325

In [9]:

df.style.background\_gradient(cmap="Reds")

Out[9]:



## Visualization Inference

Maharashtra has the highest number of confirmed cases of Indian National followed by Kerala. Haryana has the highest number of confirmed cases of Foreign National.

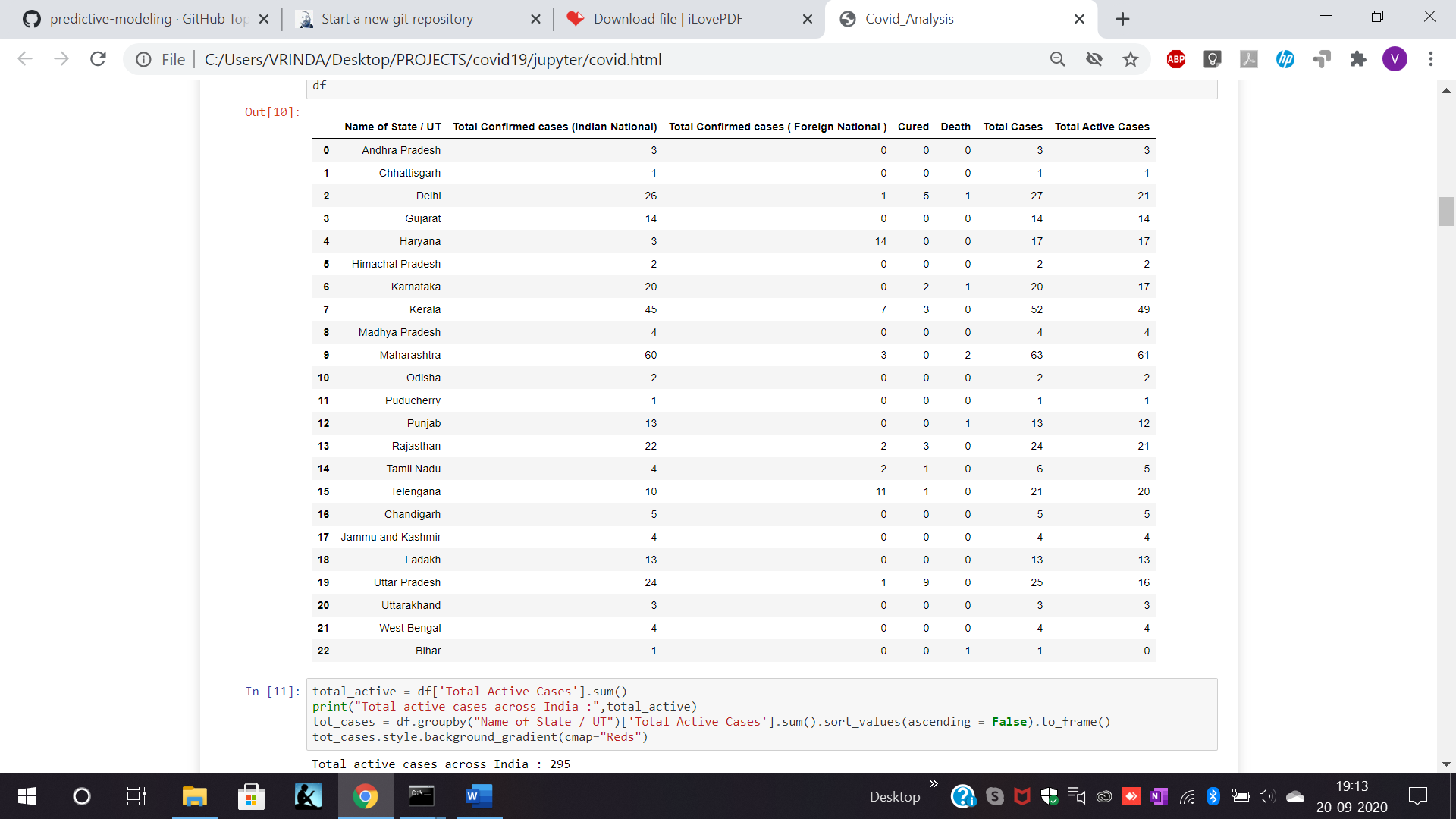
Maharashtra, Kerala, Delhi and U.P. had reported with maximum number of cases.

In [10]:

*# Total active cases in India*

df['Total Active Cases'] = df['Total Cases'] - (df['Cured'] + df['Death']) df

Out[10]:



In [11]:

total\_active = df['Total Active Cases'].sum() print("Total active cases across India :",total\_active)

tot\_cases = df.groupby("Name of State / UT")['Total Active Cases'].sum().sort\_values(ascending = **False**)

.to\_frame() tot\_cases.style.background\_gradient(cmap="Reds")

Out[11]:

Total active cases across India : 295

**Total Active Cases**

**Name of State / UT**

**Maharashtra**

61

**Kerala Rajasthan Delhi Telengana Haryana Karnataka**

**Uttar Pradesh**

49

21

21

20

17

17

16

**Gujarat Ladakh Punjab Tamil Nadu Chandigarh**

**Jammu and Kashmir**

**West Bengal**

14

13

12

5

5

4

4

**Madhya Pradesh**

4

**Uttarakhand Andhra Pradesh Himachal Pradesh Odisha Chhattisgarh**

**Puducherry**

3

3

2

2

1

1

**Bihar**

0

# Visualizing geographically spread

In [12]:

df\_spread = pd.merge(ind\_coor,df,on="Name of State / UT") df\_spread

Out[12]:

**Name of State**

**Total Confirmed cases**

**Total Confirmed cases (**

**Total**

**Total**

**/ UT Latitude Longitude**

**(Indian National)**

**Foreign National ) Cured Death Cases Active**

**Cases**

**0** Andhra 14.750429 78.570026

Pradesh

3

0 0 0 3 3

**1** Delhi 28.669993 77.230004

26

1 5 1 27 21

**2** Haryana 28.450006 77.019991

**3** Karnataka 12.570381 76.919997

**4** Kerala 8.900373 76.569993

**5** Maharashtra 19.250232 73.160175

**6** Punjab 31.519974 75.980003

**7** Rajasthan 26.449999 74.639981

**8** Telengana 18.112400 79.019300

3

20

45

60

13

22

10

14 0 0 17 17

0 2 1 20 17

7 3 0 52 49

3 0 2 63 61

0 0 1 13 12

2 3 0 24 21

11 1 0 21 20

**9** Tamil Nadu 12.920386 79.150042

4

2 1 0 6 5

**10** Uttar Pradesh 27.599981 78.050006 24 1 9 0 25 16

**11** Uttarakhand 30.320409 78.050006 3 0 0 0 3 3

In [13]:

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

**for** lat,lon,value,name **in** zip(df\_spread['Latitude'],df\_spread['Longitude'],df\_spread['Total Cases'], df\_spread['Name of State / UT']): folium.CircleMarker([lat,lon],radius=value\*0.8,color='red',fill\_c

olor='red',fill\_opacity=0.3 ).add\_to(map) map

Out[13]:



Make this Notebook Trusted to load map: File -> Trust Notebook

+

−

[Leaflet](https://leafletjs.com/) | Map tiles by [Stamen Design](http://stamen.com/), under [CC BY 3.0](http://creativecommons.org/licenses/by/3.0). Data by © [OpenStreetMap](http://openstreetmap.org/), under [CC BY SA](http://creativecommons.org/licenses/by-sa/3.0).

## Confirmed vs Recovered figures

In [14]:

data = df\_spread[['Name of State / UT','Total Cases','Cured','Death']] data

Out[14]:

**Name of State / UT Total Cases Cured Death**

1. Andhra Pradesh 3 0 0
2. Delhi 27 5 1

**2**

Haryana

17 0 0

**3**

**4**

Karnataka Kerala

1. Maharashtra
2. Punjab
3. Rajasthan
4. Telengana
5. Tamil Nadu

20 2 1

52 3 0

63 0 2

13 0 1

24 3 0

21 1 0

6 1 0

**10** Uttar Pradesh

25 9 0

**11** Uttarakhand 3 0 0

In [15]:

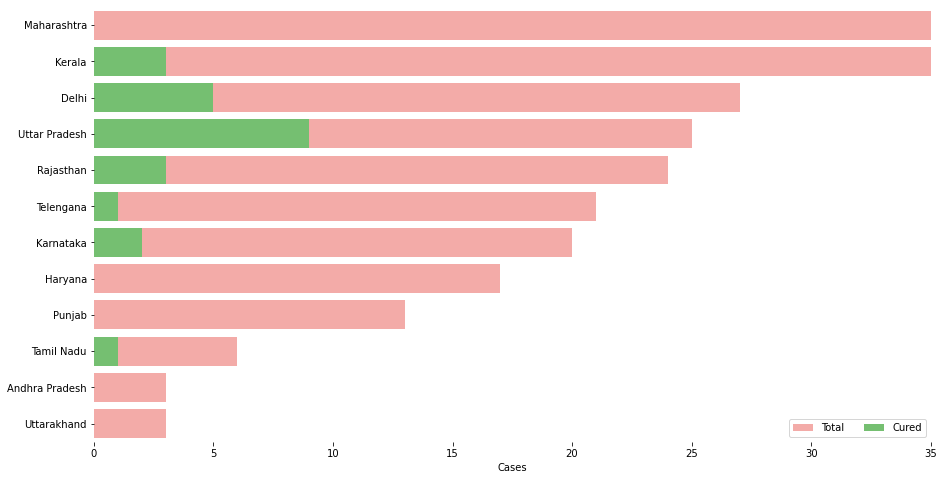
f,ax = plt.subplots(figsize = (15,8))

data.sort\_values('Total Cases',ascending=**False**,inplace = **True**) sns.set\_color\_codes("pastel")

sns.barplot(x = "Total Cases",y = "Name of State / UT",data = data,label = "Total",color = 'r') sns.set\_color\_codes("muted")

sns.barplot(x = "Cured",y = "Name of State / UT",data = data,label = "Cured",color = 'g') ax.legend(ncol = 2,loc = "lower right",frameon=**True**)

ax.set(xlim = (0,35),ylabel = "",xlabel = "Cases") sns.despine(left = **True**,bottom = **True**)



In [16]:

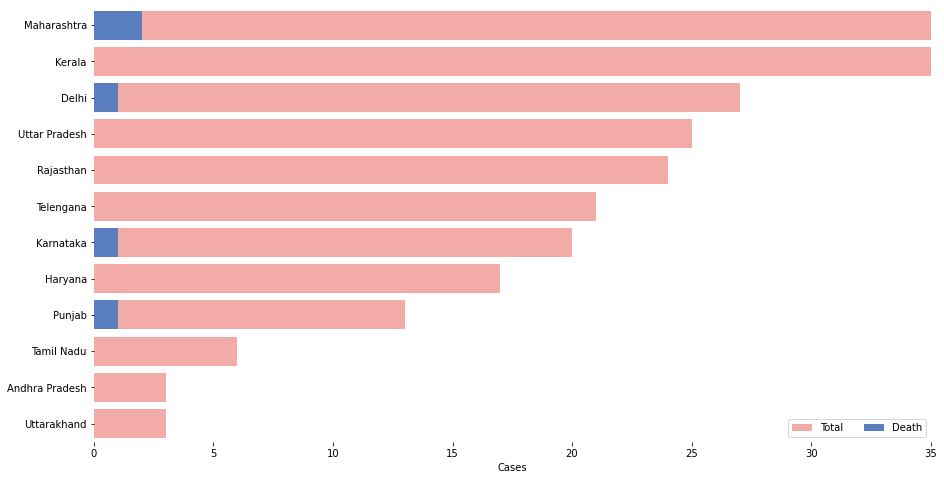
f,ax = plt.subplots(figsize = (15,8))

data.sort\_values('Total Cases',ascending=**False**,inplace = **True**) sns.set\_color\_codes("pastel")

sns.barplot(x = "Total Cases",y = "Name of State / UT",data = data,label = "Total",color = 'r') sns.set\_color\_codes("muted")

sns.barplot(x = "Death",y = "Name of State / UT",data = data,label = "Death",color = 'b') ax.legend(ncol = 2,loc = "lower right",frameon=**True**)

ax.set(xlim = (0,35),ylabel = "",xlabel = "Cases") sns.despine(left = **True**,bottom = **True**)



# How the Conoronavirus cases are rising?

In [17]:

**import plotly.io as pio**

pio.renderers.default = "iframe"

In [18]:

*# Rise of COVID-19 cases in India*

fig = go.Figure()

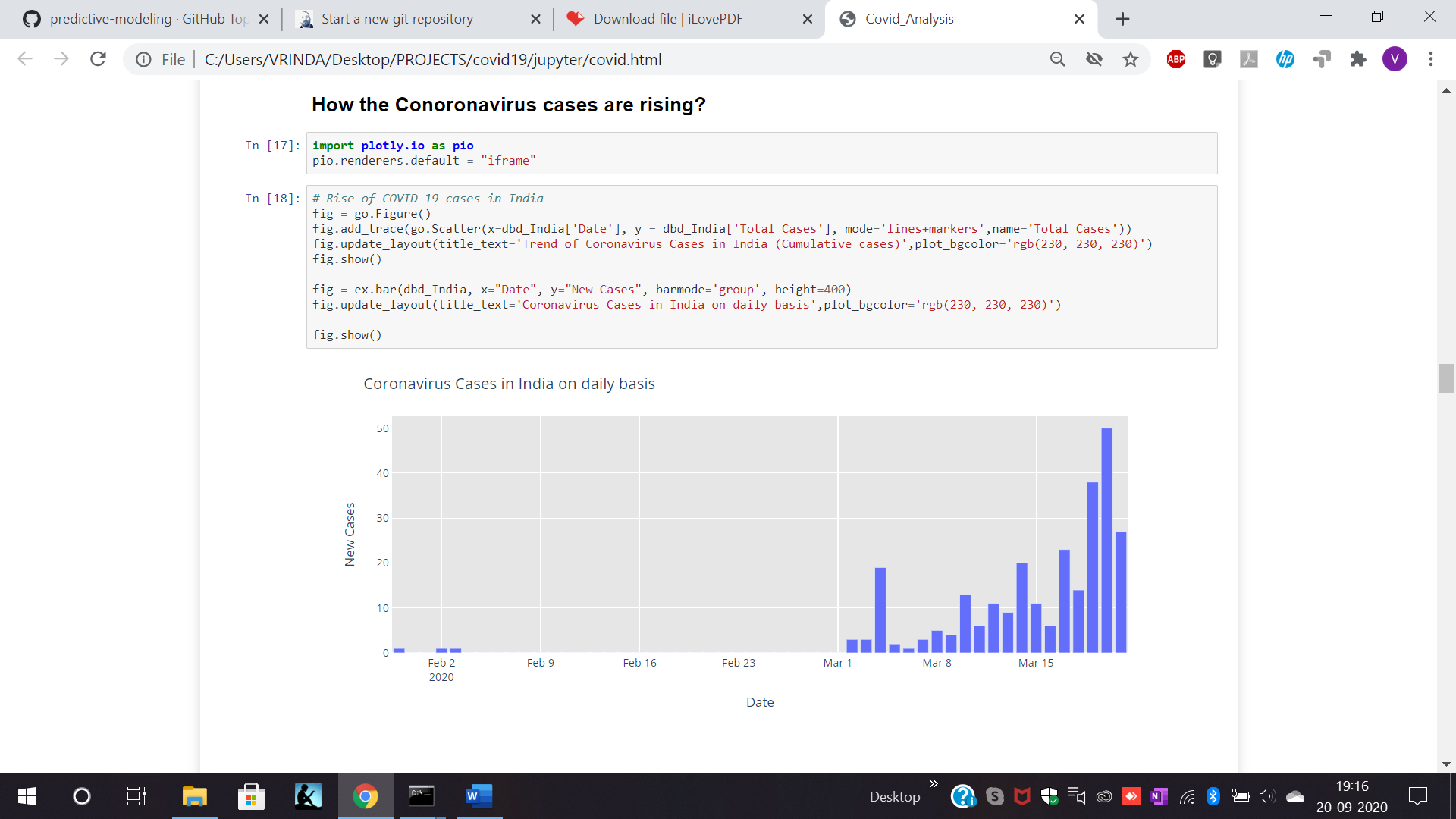
fig.add\_trace(go.Scatter(x=dbd\_India['Date'], y = dbd\_India['Total Cases'], mode='lines+markers',name= 'Total Cases'))

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

fig.show()

fig = ex.bar(dbd\_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, 23 0)')

fig.show()



# Part 2: Is the trend similar to Italy/ S.Korea/ Wuhan?

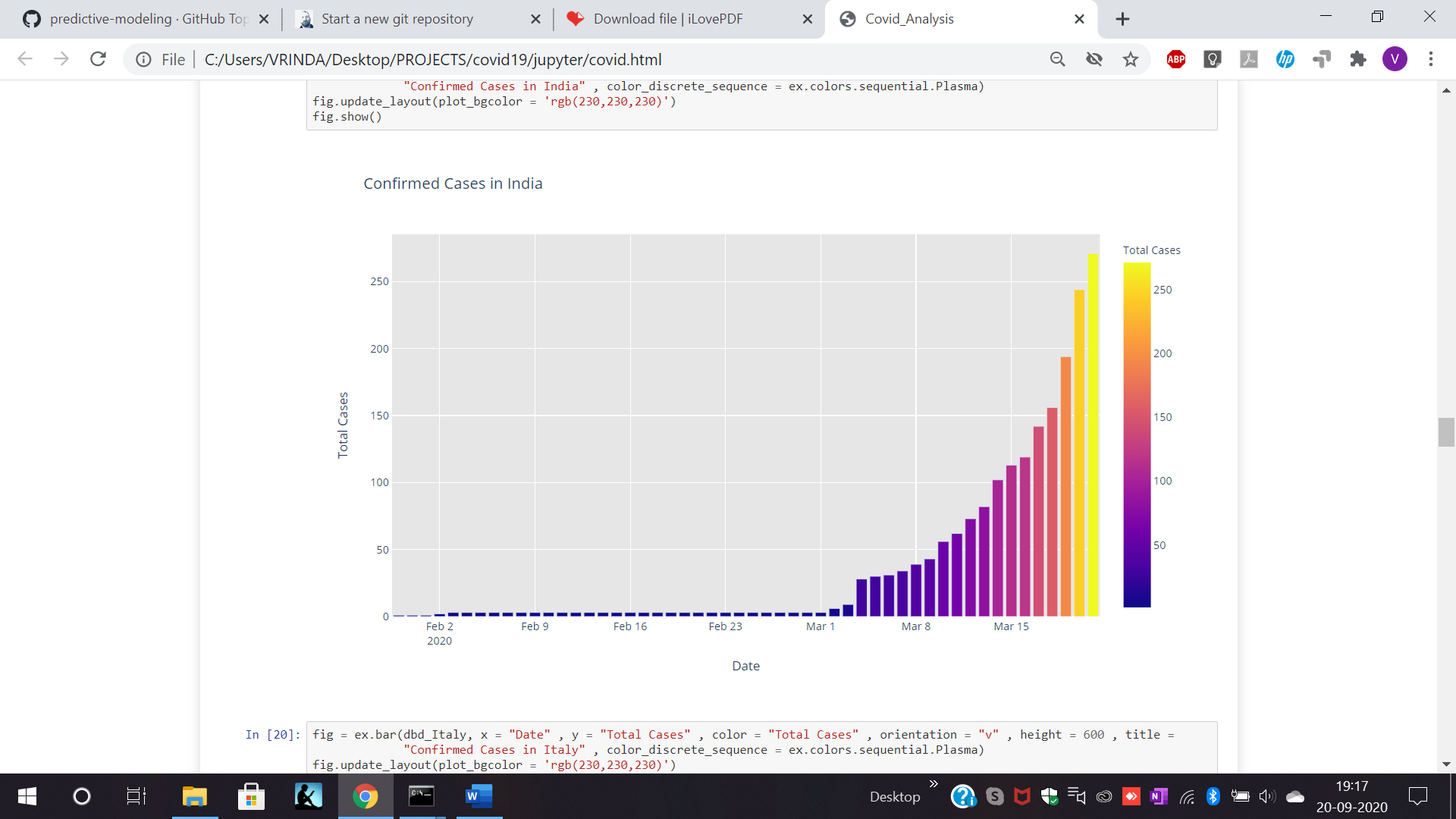
## Culmulative cases in India, Itlay ,S.Korea and Wuhan

In [19]:

fig = ex.bar(dbd\_India, x = "Date" , y = "Total Cases" , color = "Total Cases" , orientation = "v" , he ight = 600 , title =

"Confirmed Cases in India" , color\_discrete\_sequence = ex.colors.sequential.Plasma) fig.update\_layout(plot\_bgcolor = 'rgb(230,230,230)')

fig.show()

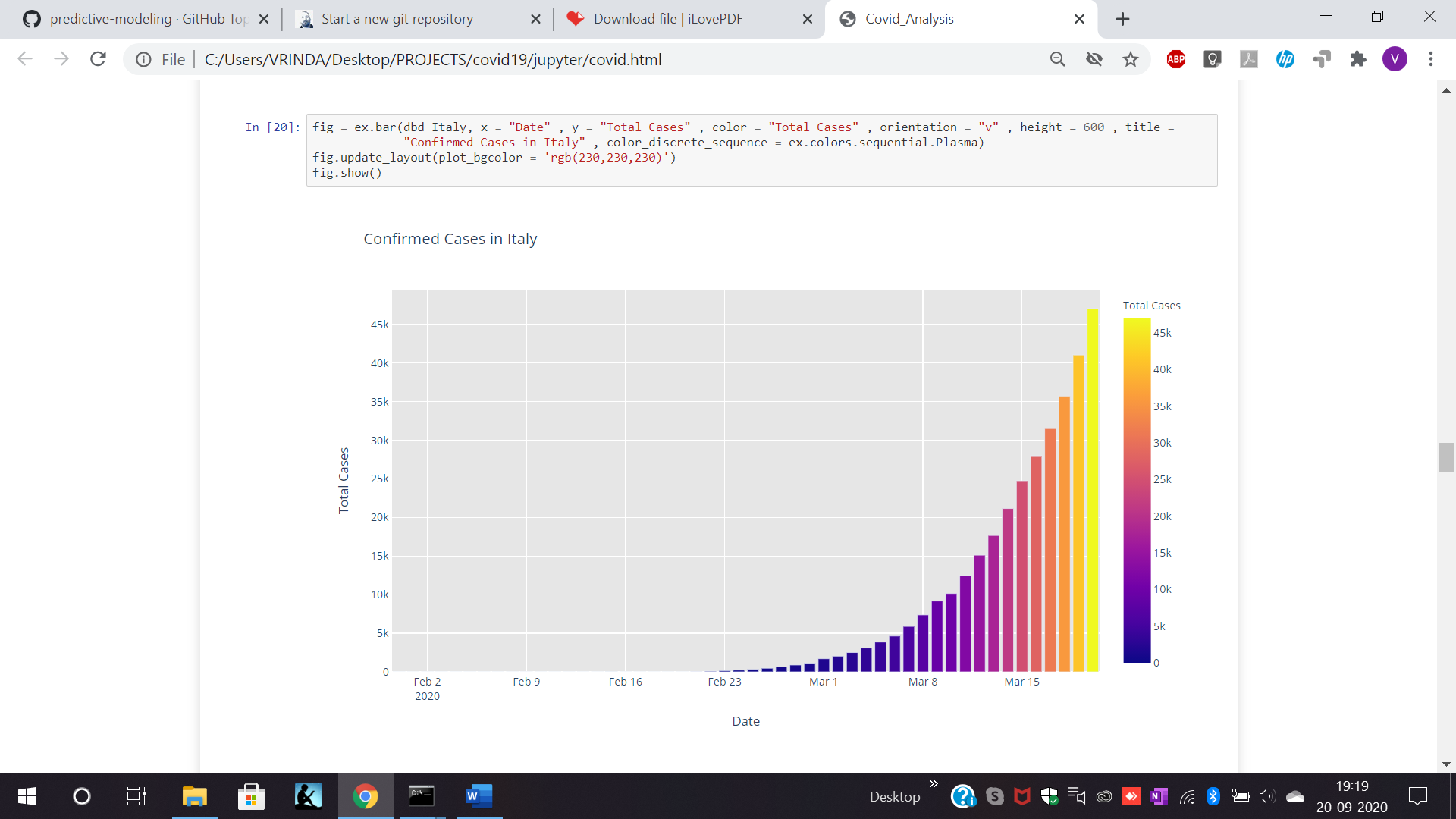


In [20]:

fig = ex.bar(dbd\_Italy, x = "Date" , y = "Total Cases" , color = "Total Cases" , orientation = "v" , he ight = 600 , title =

"Confirmed Cases in Italy" , color\_discrete\_sequence = ex.colors.sequential.Plasma) fig.update\_layout(plot\_bgcolor = 'rgb(230,230,230)')

fig.show()

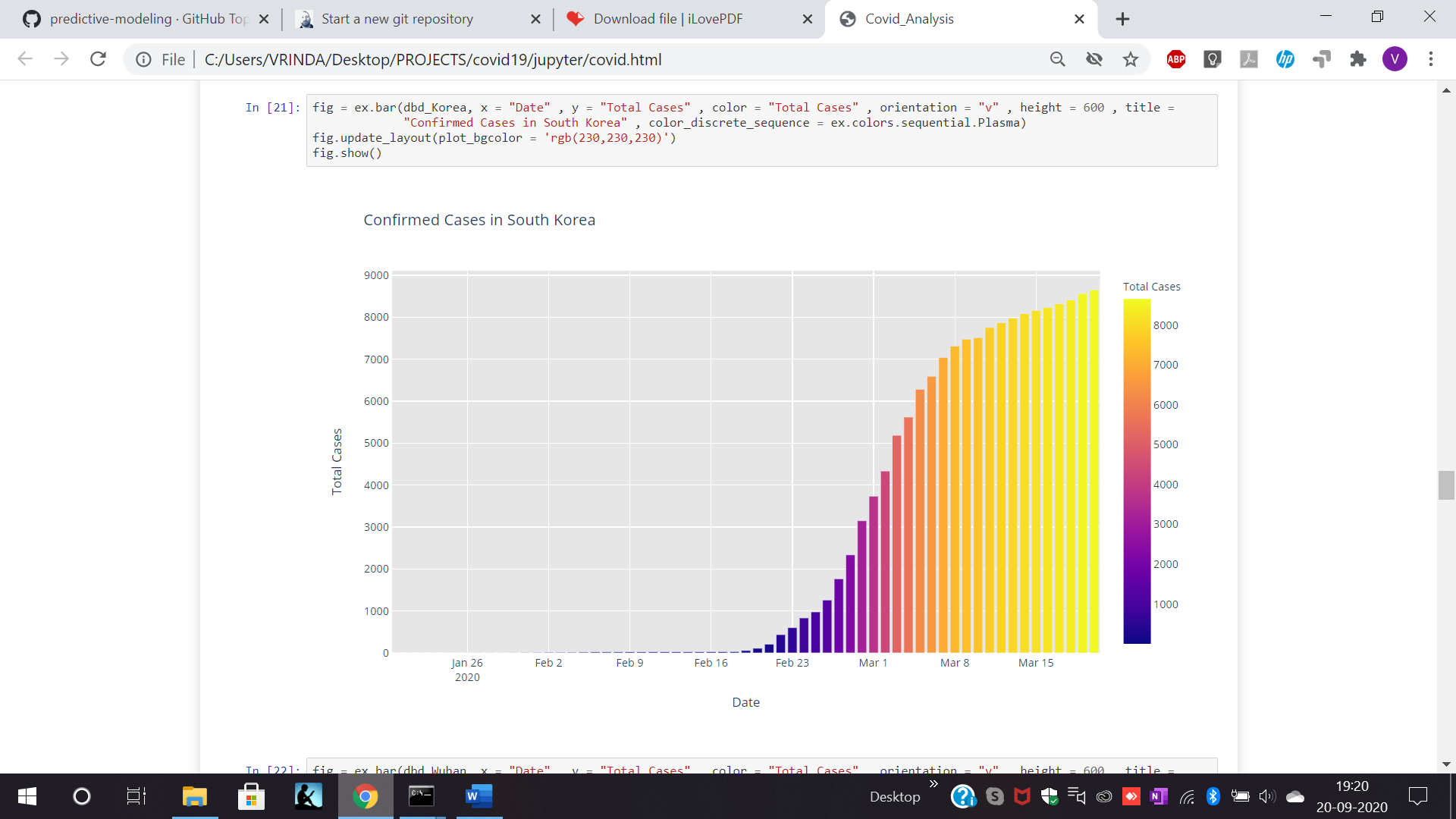


In [21]:

fig = ex.bar(dbd\_Korea, x = "Date" , y = "Total Cases" , color = "Total Cases" , orientation = "v" , he ight = 600 , title =

"Confirmed Cases in South Korea" , color\_discrete\_sequence = ex.colors.sequential.Plasma) fig.update\_layout(plot\_bgcolor = 'rgb(230,230,230)')

fig.show()

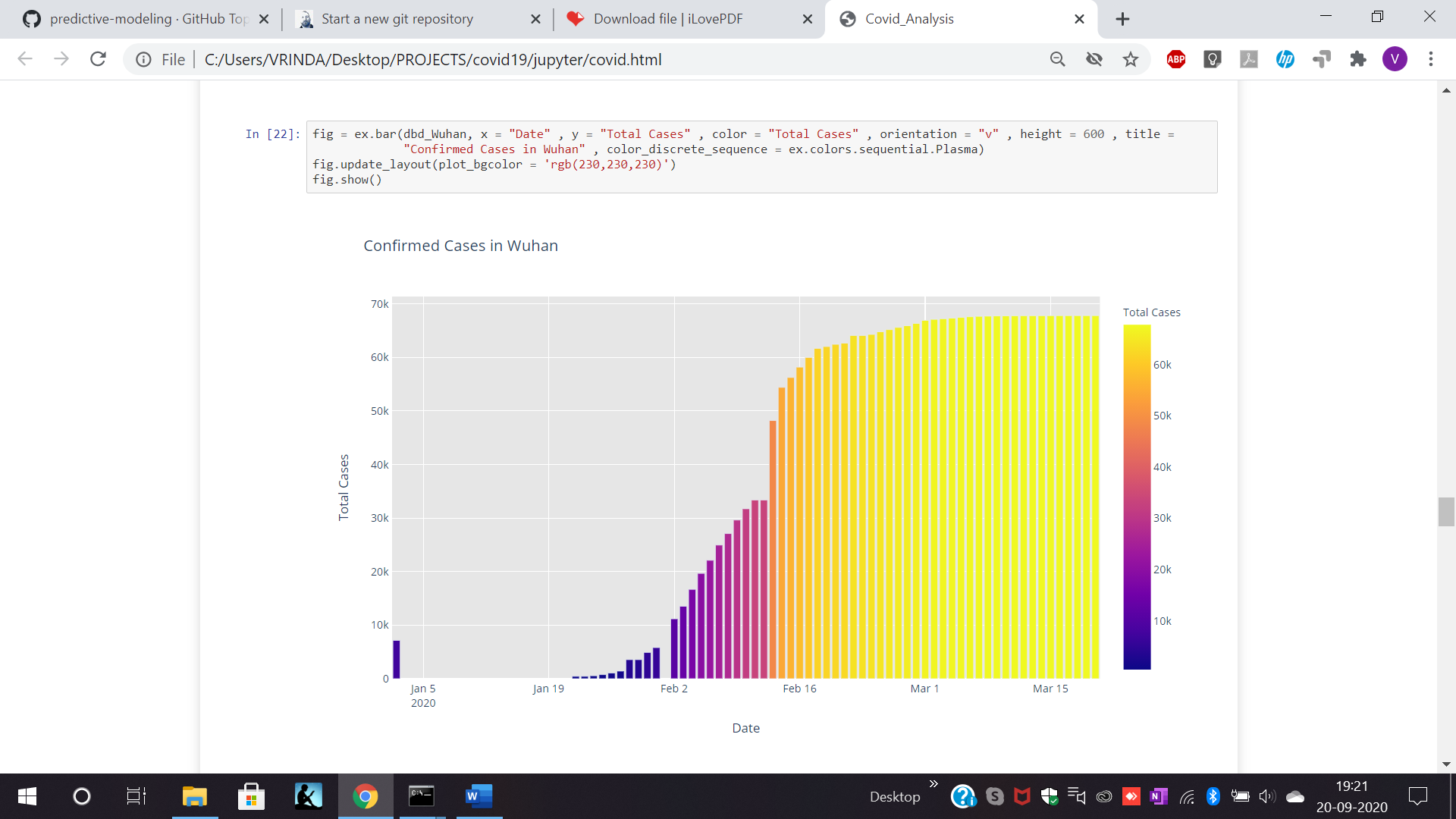


In [22]:

fig = ex.bar(dbd\_Wuhan, x = "Date" , y = "Total Cases" , color = "Total Cases" , orientation = "v" , he ight = 600 , title =

"Confirmed Cases in Wuhan" , color\_discrete\_sequence = ex.colors.sequential.Plasma) fig.update\_layout(plot\_bgcolor = 'rgb(230,230,230)')

fig.show()



## Comparison between the rise of cases in Wuhan, S.Korea, Italy and India

In [23]:

*# making sub-plots*

**from plotly.subplots import** make\_subplots

fig = make\_subplots(rows=2,cols=2,specs =[[{},{}],[{"colspan": 2},**None**]],subplot\_titles=("S.Korea","Ita ly","India","Wuhan"))

fig.add\_trace(go.Bar(x=dbd\_Korea['Date'], y=dbd\_Korea['Total Cases'], marker=dict(color=dbd\_Korea['Total Cases'], coloraxis="coloraxis")),1, 1)

fig.add\_trace(go.Bar(x=dbd\_Italy['Date'], y=dbd\_Italy['Total Cases'], marker=dict(color=dbd\_Italy['Total Cases'], coloraxis="coloraxis")),1, 2)

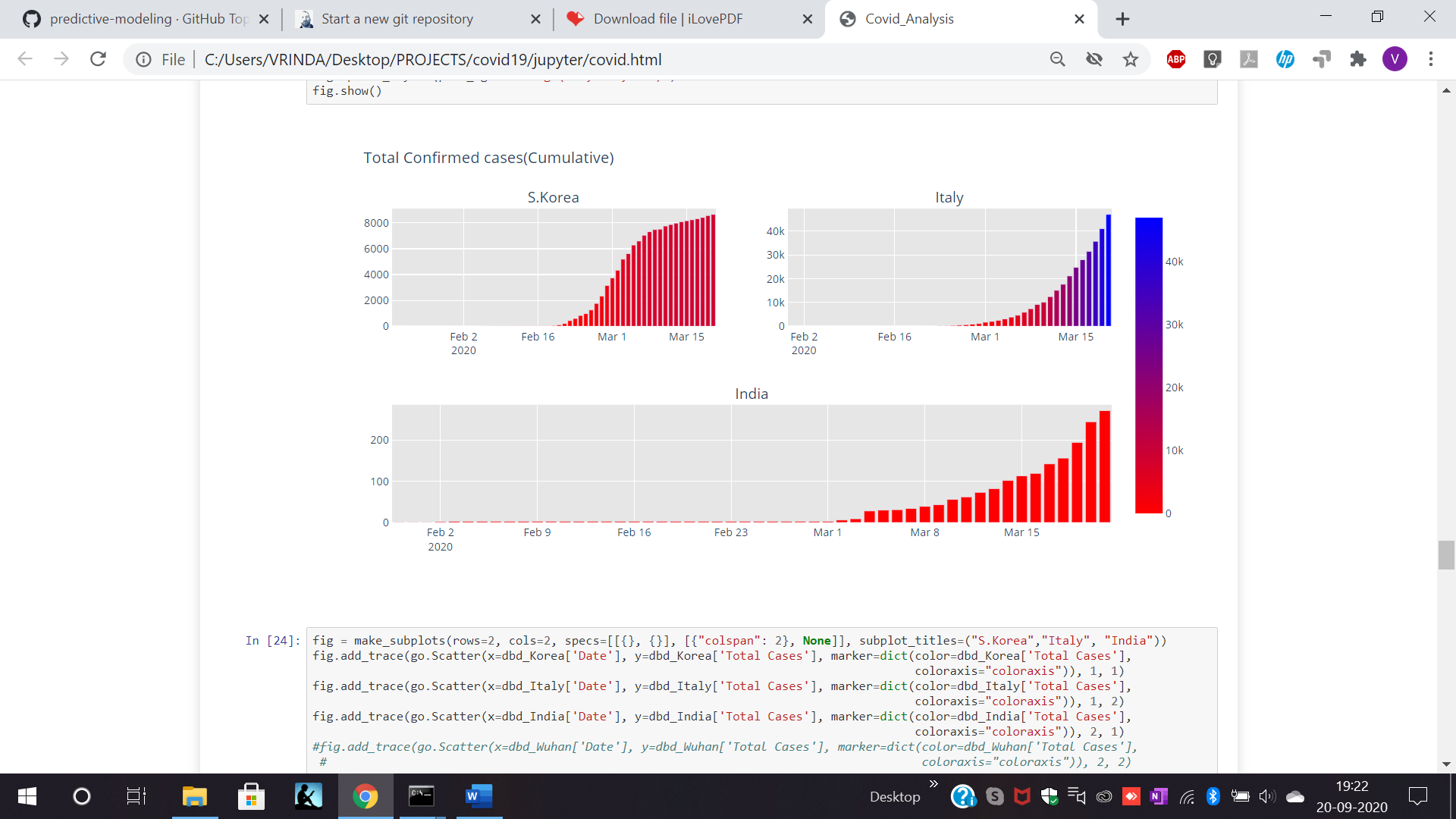
fig.add\_trace(go.Bar(x=dbd\_India['Date'], y=dbd\_India['Total Cases'], marker=dict(color=dbd\_India['Total Cases'], coloraxis="coloraxis")),2, 1)

*# fig.add\_trace(go.Bar(x=dbd\_Wuhan['Date'], y=dbd\_Wuhan['Total Cases'],*

*# marker=dict(color=dbd\_Wuhan['Total Cases'], coloraxis="coloraxis")),2, 2)*

fig.update\_layout(coloraxis=dict(colorscale='Bluered\_r'), showlegend=**False**,title\_text="Total Confirmed cases(Cumulative)")

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



In [24]:

fig = make\_subplots(rows=2, cols=2, specs=[[{}, {}], [{"colspan": 2}, **None**]], subplot\_titles=("S.Korea"

,"Italy", "India"))

fig.add\_trace(go.Scatter(x=dbd\_Korea['Date'], y=dbd\_Korea['Total Cases'], marker=dict(color=dbd\_Korea[ 'Total Cases'],

coloraxis="colora

xis")), 1, 1)

fig.add\_trace(go.Scatter(x=dbd\_Italy['Date'], y=dbd\_Italy['Total Cases'], marker=dict(color=dbd\_Italy[ 'Total Cases'],

coloraxis="colora

xis")), 1, 2)

fig.add\_trace(go.Scatter(x=dbd\_India['Date'], y=dbd\_India['Total Cases'], marker=dict(color=dbd\_India[ 'Total Cases'],

coloraxis="colora

xis")), 2, 1)

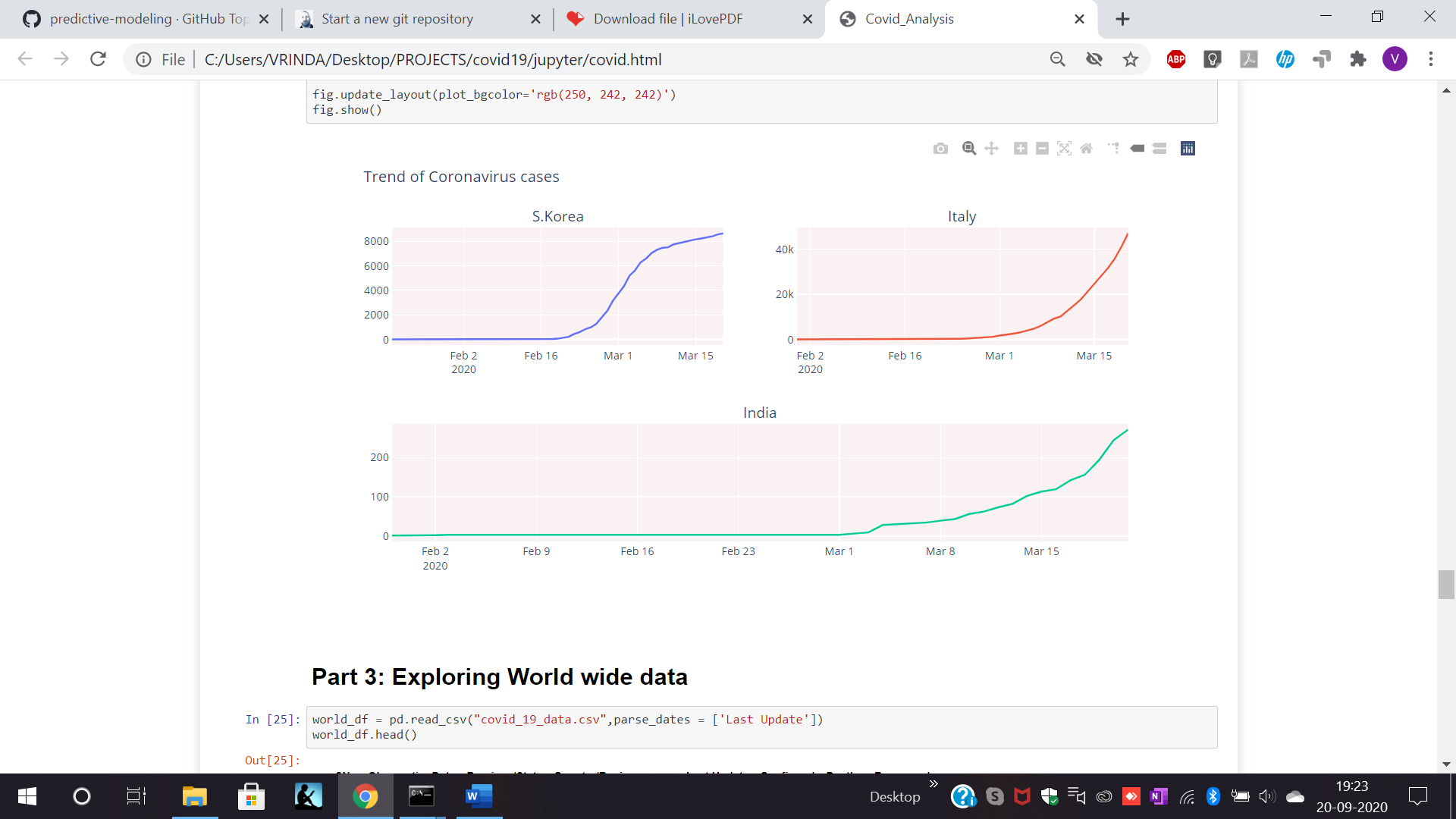
*#fig.add\_trace(go.Scatter(x=dbd\_Wuhan['Date'], y=dbd\_Wuhan['Total Cases'], marker=dict(color=dbd\_Wuhan ['Total Cases'],*

*# coloraxis="color*

*axis")), 2, 2)*

fig.update\_layout(coloraxis=dict(colorscale='Bluered\_r'), showlegend=**False**,title\_text="Trend of Coronav irus cases")

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

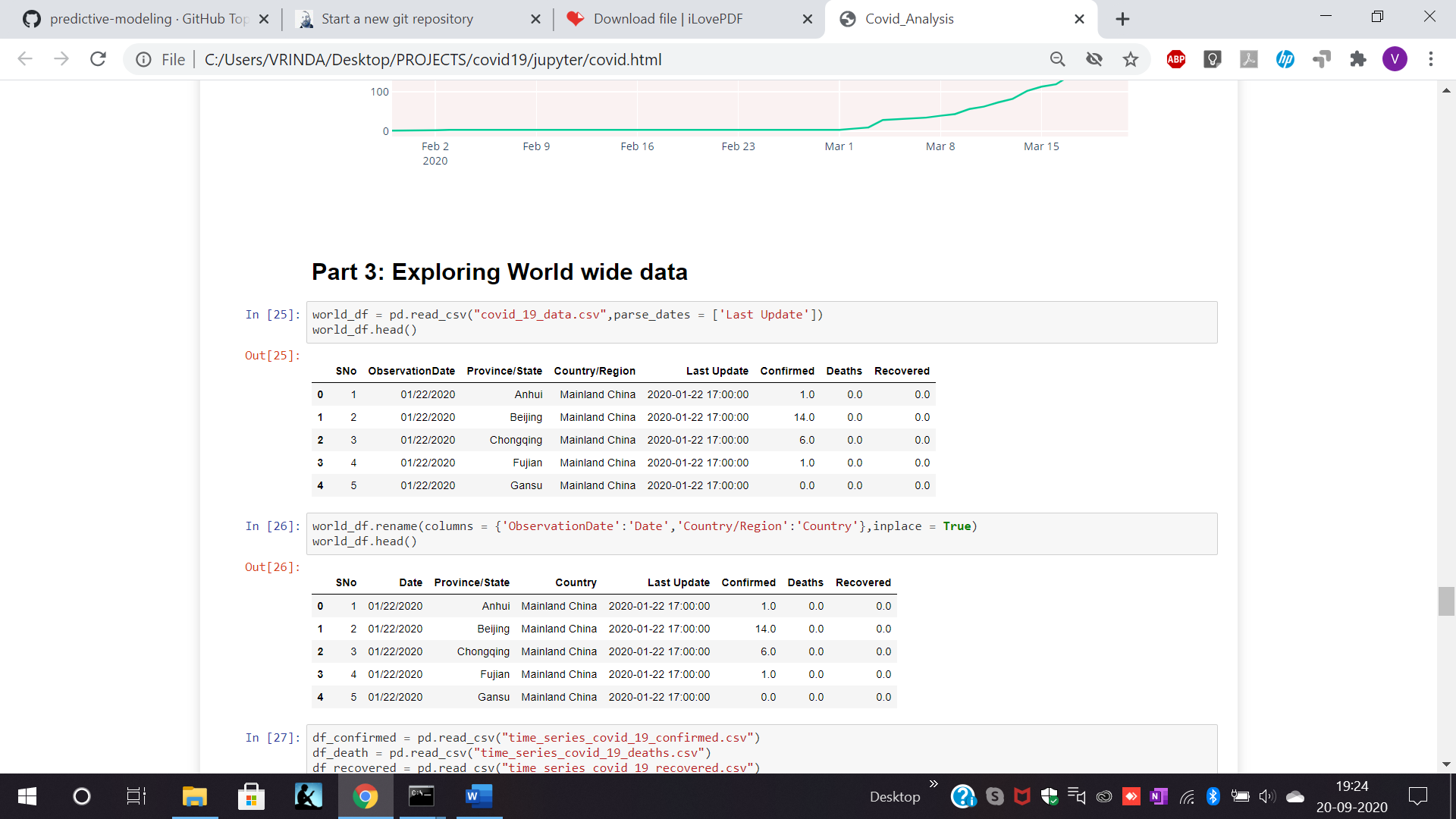


# Part 3: Exploring Worldwide data

In [25]:

world\_df = pd.read\_csv("covid\_19\_data.csv",parse\_dates = ['Last Update']) world\_df.head()

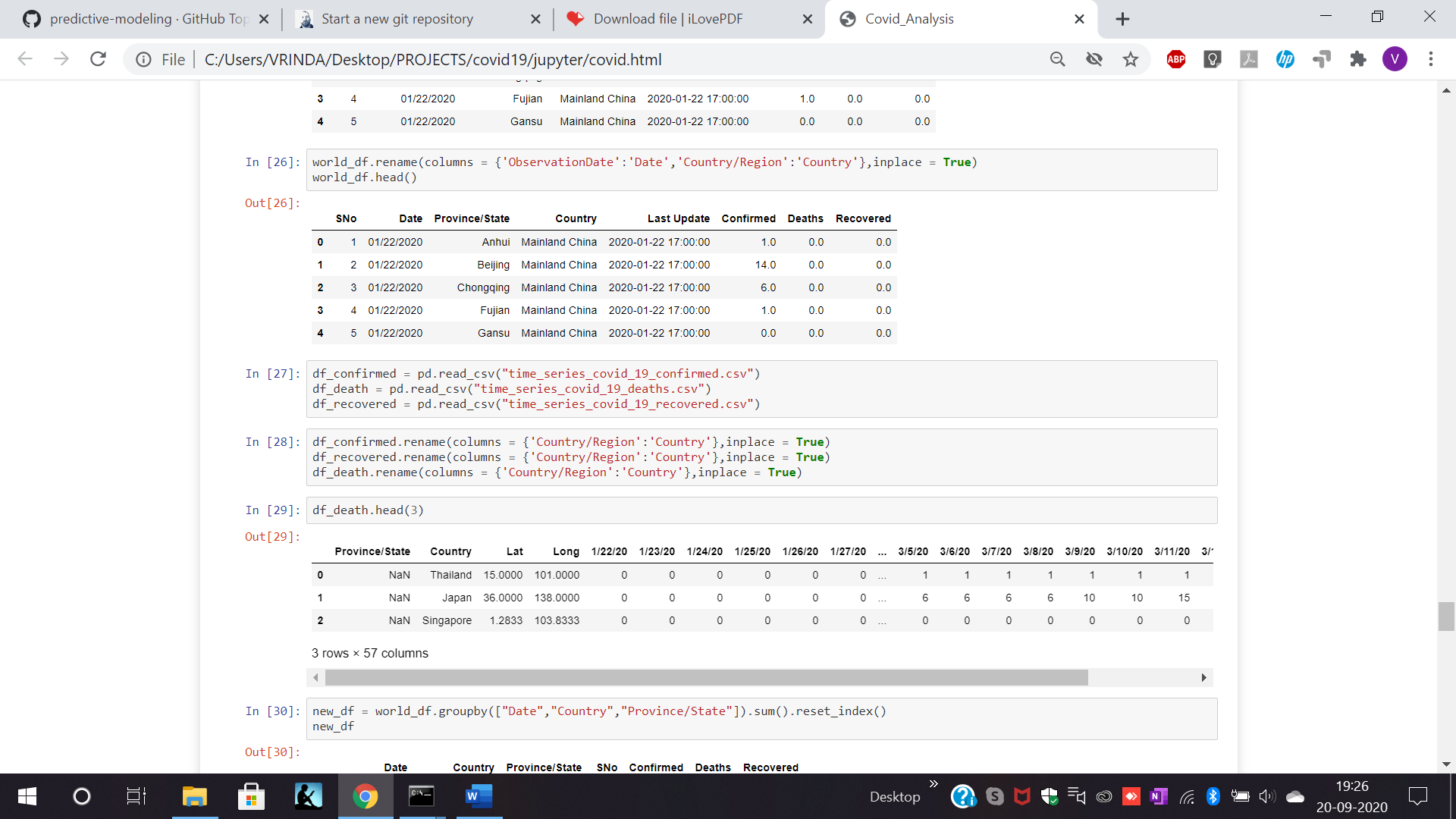
Out[25]:



In [26]:

world\_df.rename(columns = {'ObservationDate':'Date','Country/Region':'Country'},inplace = **True**) world\_df.head()

Out[26]:

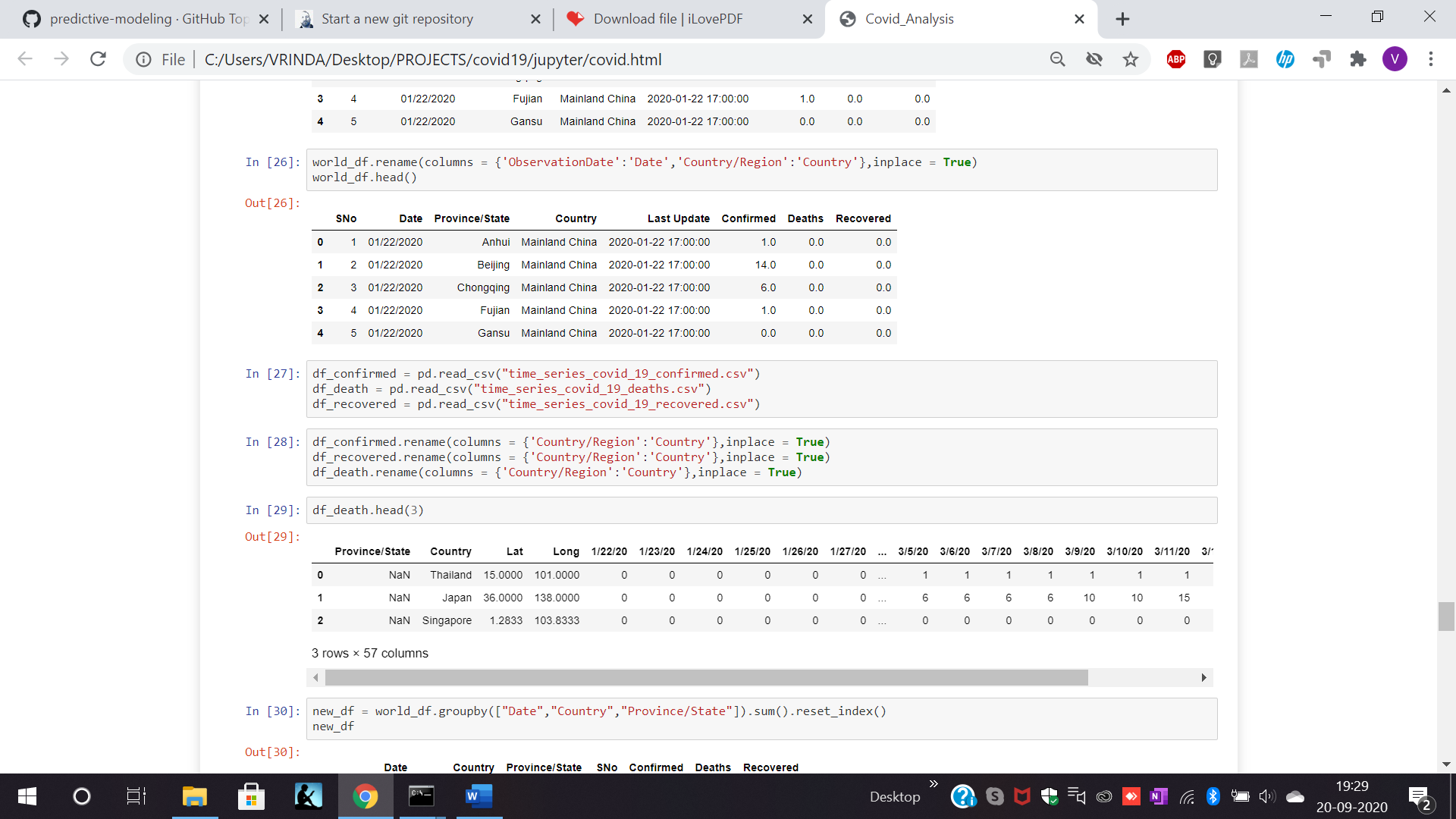


In [27]:

df\_confirmed = pd.read\_csv("time\_series\_covid\_19\_confirmed.csv") df\_death = pd.read\_csv("time\_series\_covid\_19\_deaths.csv") df\_recovered = pd.read\_csv("time\_series\_covid\_19\_recovered.csv")

In [28]:

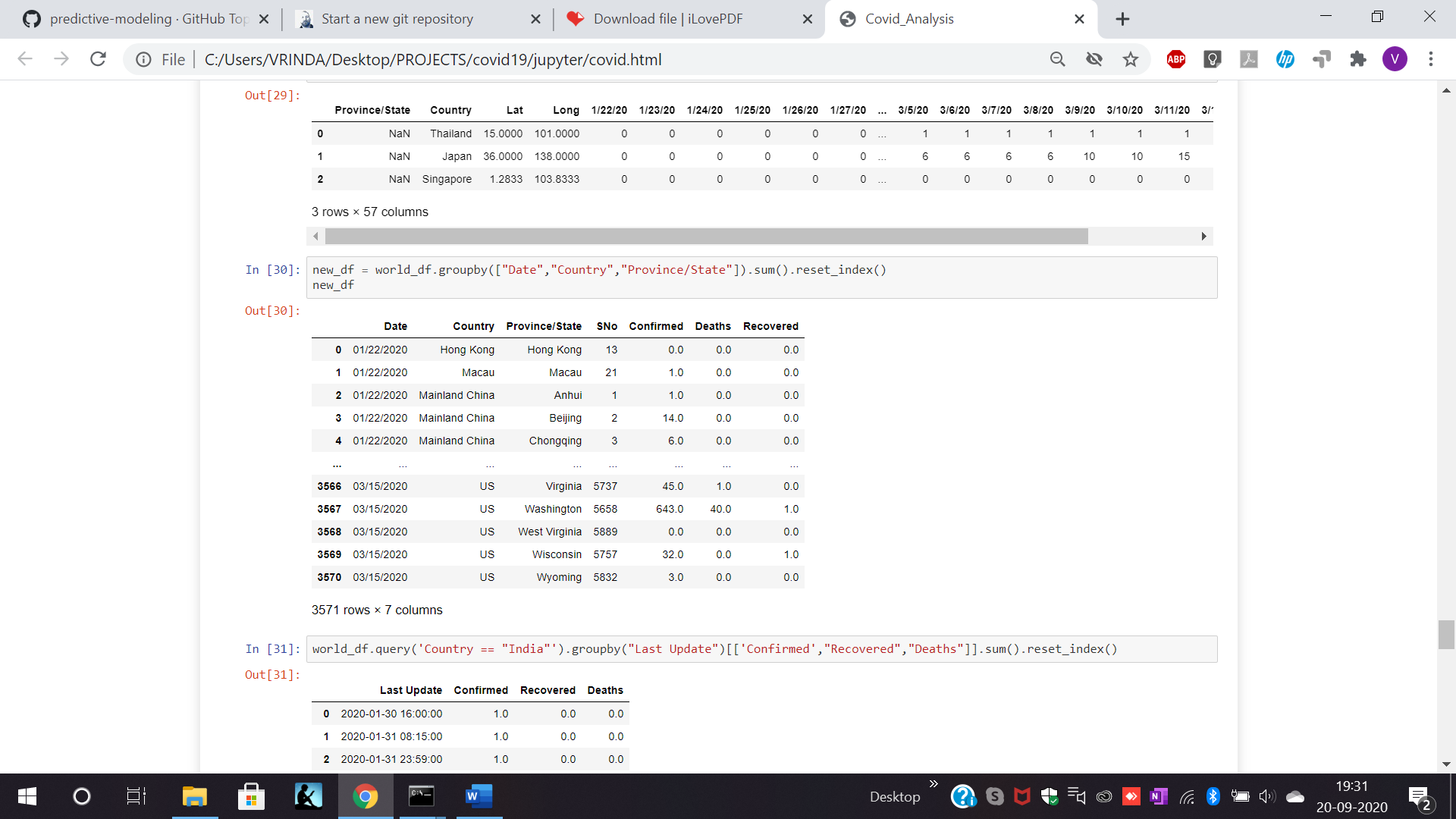
df\_confirmed.rename(columns = {'Country/Region':'Country'},inplace = **True**) df\_recovered.rename(columns = {'Country/Region':'Country'},inplace = **True**) df\_death.rename(columns = {'Country/Region':'Country'},inplace = **True**)



In [30]:

new\_df = world\_df.groupby(["Date","Country","Province/State"]).sum().reset\_index() new\_df

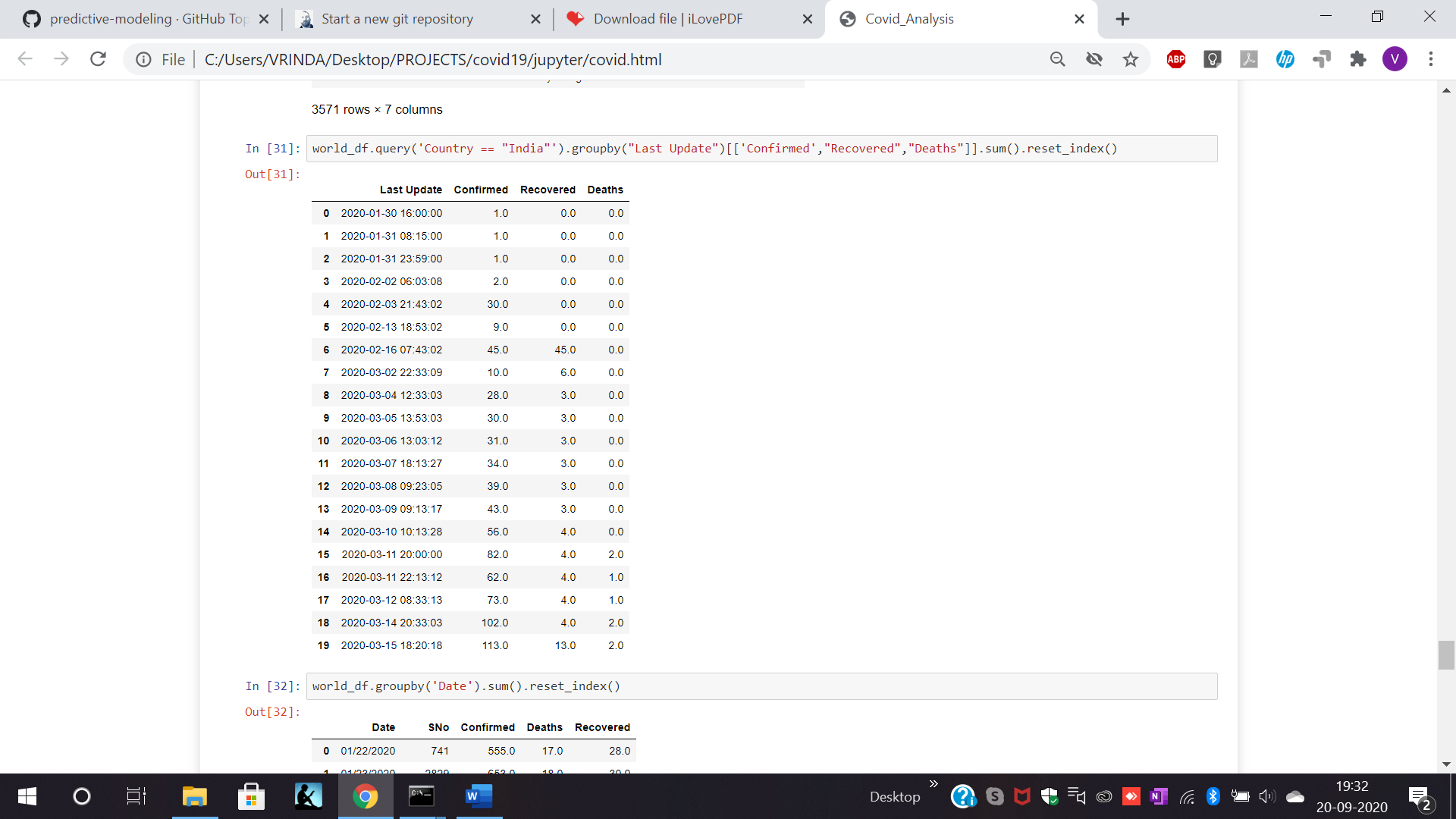
Out[30]:



In [31]:

world\_df.query('Country == "India"').groupby("Last Update")[['Confirmed',"Recovered","Deaths"]].sum().r eset\_index()

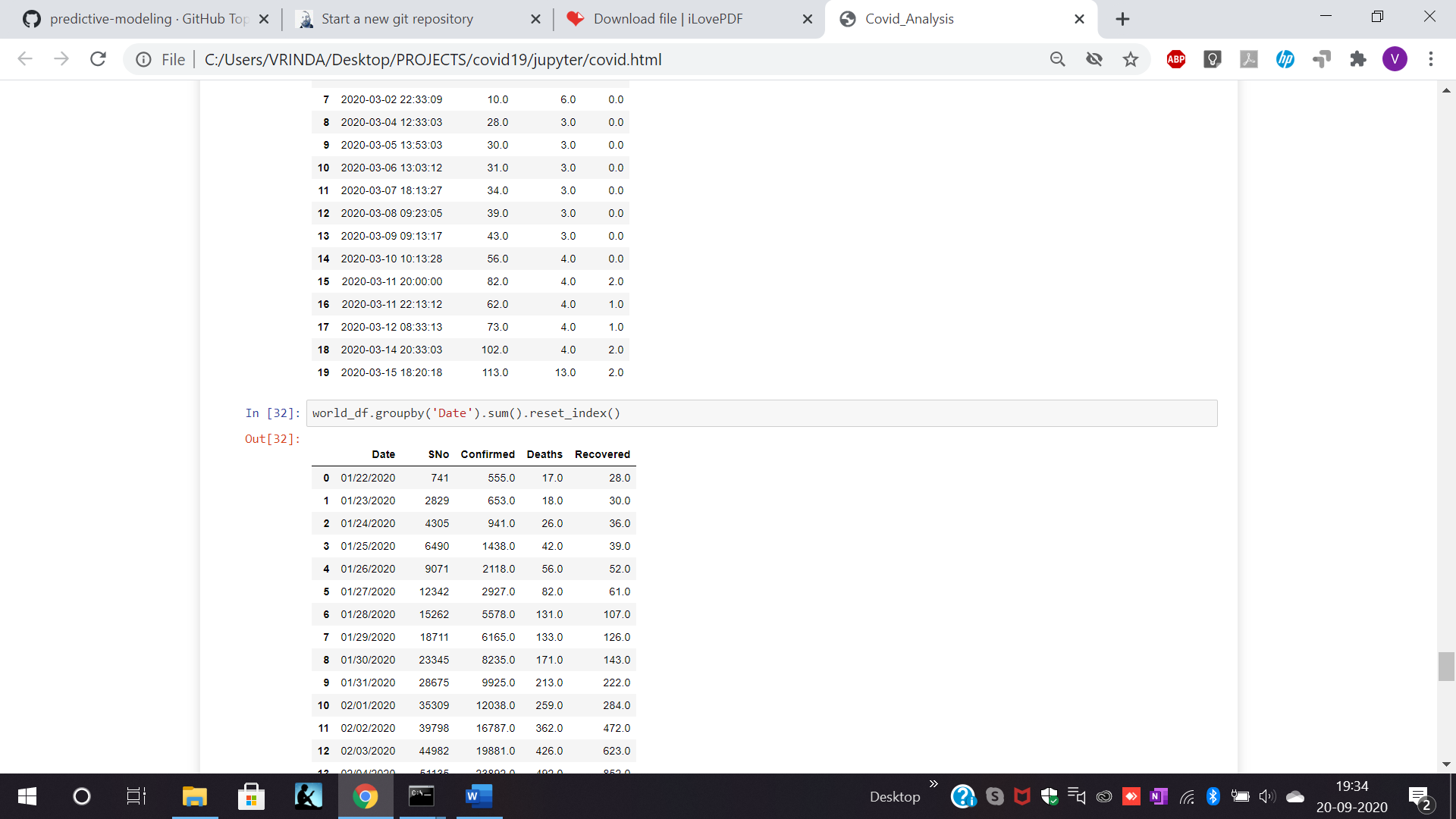
Out[31]:



In [32]:

world\_df.groupby('Date').sum().reset\_index()

Out[32]:



# Visualizing : Worldwide N COVID-19 cases

In [33]:

confirmed = world\_df.groupby('Date').sum()['Confirmed'].reset\_index() recovered = world\_df.groupby('Date').sum()['Recovered'].reset\_index() death = world\_df.groupby('Date').sum()['Deaths'].reset\_index()

In [34]:

fig = go.Figure()

fig.add\_trace(go.Scatter(x = confirmed['Date'] , y = confirmed['Confirmed'] , mode = "lines+markers" , name = "Confirmed" ,

line = dict(color = "Blue" , width = 2)))

fig.add\_trace(go.Scatter(x = recovered['Date'] , y = recovered['Recovered'] , mode = "lines+markers" , name = "Recovered" ,

line = dict(color = "Green" , width = 2)))

fig.add\_trace(go.Scatter(x = death['Date'] , y = death['Deaths'] , mode = "lines+markers" , name = "Dea ths" ,

line = dict(color = "Red" , width = 2)))

fig.update\_layout(title='Worldwide N COVID-19 Cases', xaxis\_tickfont\_size=14,xaxis=dict(title= 'Date'), yaxis=dict(title='Number of Cases'))

fig.show()

