

Covid-19 visualization, prediction and forecasting

```
In [1]: 1 # Importing all the important libraries.
2
3 import numpy as np
4 import pandas as pd
5 import matplotlib.pyplot as plt
6 import matplotlib.colors as mcolors
7 import random
8 import math
9 import time
10 import datetime
11 import operator
12 from sklearn.model_selection import RandomizedSearchCV, train_test_split
13 from sklearn.svm import SVR
14 from sklearn.metrics import mean_squared_error, mean_absolute_error
15 from sklearn.linear_model import LinearRegression
16
17 plt.style.use('seaborn')
18 %matplotlib inline
```

```
In [2]: 1 # Loading all the three datasets
2
3 confirmed_cases = pd.read_csv("time_series_covid_19_confirmed.csv")
```

```
In [3]: 1 deaths_reported = pd.read_csv("time_series_covid_19_deaths.csv")
```

```
In [4]: 1 recovered_cases = pd.read_csv("time_series_covid_19_recovered.csv")
```

```
In [5]: 1 # display the head of the Dataset
2
3 confirmed_cases.head()
```

Out[5]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	...	6/13/20	6/14/20	6/15/20	6/16/20	6/17/20	6/18/20	6/19/20
0	NaN	Afghanistan	33.0000	65.0000	0	0	0	0	0	0	...	24102	24766	25527	26310	26874	27532	28296
1	NaN	Albania	41.1533	20.1683	0	0	0	0	0	0	...	1464	1521	1590	1672	1722	1788	1855
2	NaN	Algeria	28.0339	1.6596	0	0	0	0	0	0	...	10810	10919	11031	11147	11268	11385	11502
3	NaN	Andorra	42.5063	1.5218	0	0	0	0	0	0	...	853	853	853	854	854	855	855
4	NaN	Angola	-11.2027	17.8739	0	0	0	0	0	0	...	138	140	142	148	155	166	177

5 rows × 157 columns

```
In [6]: 1 deaths_reported.head()
```

Out[6]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	...	6/13/20	6/14/20	6/15/20	6/16/20	6/17/20	6/18/20	6/19/20
0	NaN	Afghanistan	33.0000	65.0000	0	0	0	0	0	0	...	451	471	478	491	504	546	566
1	NaN	Albania	41.1533	20.1683	0	0	0	0	0	0	...	36	36	36	37	38	39	40
2	NaN	Algeria	28.0339	1.6596	0	0	0	0	0	0	...	760	767	777	788	799	811	823
3	NaN	Andorra	42.5063	1.5218	0	0	0	0	0	0	...	51	51	51	52	52	52	52
4	NaN	Angola	-11.2027	17.8739	0	0	0	0	0	0	...	6	6	6	6	7	8	9

5 rows × 157 columns

```
In [7]: 1 recovered_cases.head()
```

Out[7]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	...	6/13/20	6/14/20	6/15/20	6/16/20	6/17/20	6/18/20	6/19/20
0	NaN	Afghanistan	33.0000	65.0000	0	0	0	0	0	0	...	4201	4725	5164	5508	6158	7660	8162
1	NaN	Albania	41.1533	20.1683	0	0	0	0	0	0	...	1039	1044	1055	1064	1077	1086	1095
2	NaN	Algeria	28.0339	1.6596	0	0	0	0	0	0	...	7420	7606	7735	7842	7943	8078	8213
3	NaN	Andorra	42.5063	1.5218	0	0	0	0	0	0	...	781	781	789	789	791	792	792
4	NaN	Angola	-11.2027	17.8739	0	0	0	0	0	0	...	61	61	64	64	64	64	64

5 rows × 157 columns

```
In [8]: 1 # Extracting all the columns using the .keys() function.
2 cols = confirmed_cases.keys()
3 cols
```

Out[8]: Index(['Province/State', 'Country/Region', 'Lat', 'Long', '1/22/20', '1/23/20', '1/24/20', '1/25/20', '1/26/20', '1/27/20', ..., '6/13/20', '6/14/20', '6/15/20', '6/16/20', '6/17/20', '6/18/20', '6/19/20', '6/20/20', '6/21/20', '6/22/20'], dtype='object', length=157)

```
In [9]: 1 # Extracting only the dates columns that have information of confirmed,deaths and recovered cases.
2 confirmed = confirmed_cases.loc[:, cols[4]:cols[-1]]
```

```
In [10]: 1 deaths = deaths_reported.loc[:, cols[4]:cols[-1]]
```

```
In [11]: 1 recoveries = recovered_cases.loc[:, cols[4]:cols[-1]]
```

```
In [12]: 1 # Check the head of the outbreak cases.
2 confirmed.head()
```

Out[12]:

	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	1/28/20	1/29/20	1/30/20	1/31/20	...	6/13/20	6/14/20	6/15/20	6/16/20	6/17/20	6/18/20	6/19/20	6/20/20
0	0	0	0	0	0	0	0	0	0	0	...	24102	24766	25527	26310	26874	27532	27878	28424
1	0	0	0	0	0	0	0	0	0	0	...	1464	1521	1590	1672	1722	1788	1838	1891
2	0	0	0	0	0	0	0	0	0	0	...	10810	10919	11031	11147	11268	11385	11504	11631
3	0	0	0	0	0	0	0	0	0	0	...	853	853	853	854	854	855	855	855
4	0	0	0	0	0	0	0	0	0	0	...	138	140	142	148	155	166	172	176

5 rows × 153 columns



```
In [13]: 1 # Finding the total confirmed cases,death cases and the recovered cases and append them to an 4 empty Lists.
2 # Also, calculate the total mortality rate which is the death sum/confirmed cases.
3
4 dates = confirmed.keys()
5 world_cases = []
6 total_deaths = []
7 mortality_rate = []
8 total_recovered = []
9
10 for i in dates:
11     confirmed_sum = confirmed[i].sum()
12     death_sum = deaths[i].sum()
13     recovered_sum = recoveries[i].sum()
14     world_cases.append(confirmed_sum)
15     total_deaths.append(death_sum)
16     mortality_rate.append(death_sum/confirmed_sum)
17     total_recovered.append(recovered_sum)
```

```
In [14]: 1 # Lets display each of the newly created variables
2 confirmed_sum
```

Out[14]: 9098643

```
In [15]: 1 death_sum
```

Out[15]: 472171

```
In [16]: 1 recovered_sum
```

Out[16]: 4526333

```
In [17]: 1 world_cases
```

Out[17]: [555,
654,
941,
1434,
2118,
2927,
5578,
6166,
8234,
9927,
12038,
16787,
19881,
23892,
27635,
30794,
34391,
37120,
40150,

```
In [18]: 1 # Convert all the dates and the cases in the form of a numpy array
2
3 days_since_1_22 = np.array([i for i in range(len(dates))]).reshape(-1,1)
4 world_cases = np.array(world_cases).reshape(-1,1)
5 total_deaths = np.array(total_deaths).reshape(-1,1)
6 total_recovered = np.array(total_recovered).reshape(-1,1)
```

```
In [19]: 1 days_since_1_22
```

```
Out[19]: array([[ 0],
 [ 1],
 [ 2],
 [ 3],
 [ 4],
 [ 5],
 [ 6],
 [ 7],
 [ 8],
 [ 9],
[10],
[11],
[12],
[13],
[14],
[15],
[16],
[17],
[18],
[19]])
```

```
In [20]: 1 world_cases
```

```
Out[20]: array([[ 555],
 [ 654],
 [ 941],
 [1434],
 [2118],
 [2927],
 [5578],
 [6166],
 [8234],
 [9927],
 [12038],
 [16787],
 [19881],
 [23892],
 [27635],
 [30794],
 [34391],
 [37120],
 [40150],
 [42762],
 [45355],
 [47938],
 [50511],
 [53074],
 [55627],
 [58170],
 [60703],
 [63226],
 [65739],
 [68242],
 [70735],
 [73218],
 [75691],
 [78154],
 [80607],
 [83050],
 [85483],
 [87906],
 [90319],
 [92722],
 [95115],
 [97498],
 [100000],
 [102500],
 [105000],
 [107500],
 [110000],
 [112500],
 [115000],
 [117500],
 [120000],
 [122500],
 [125000],
 [127500],
 [130000],
 [132500],
 [135000],
 [137500],
 [140000],
 [142500],
 [145000],
 [147500],
 [150000],
 [152500],
 [155000],
 [157500],
 [160000],
 [162500],
 [165000],
 [167500],
 [170000],
 [172500],
 [175000],
 [177500],
 [180000],
 [182500],
 [185000],
 [187500],
 [190000],
 [192500],
 [195000],
 [197500],
 [200000],
 [202500],
 [205000],
 [207500],
 [210000],
 [212500],
 [215000],
 [217500],
 [220000],
 [222500],
 [225000],
 [227500],
 [230000],
 [232500],
 [235000],
 [237500],
 [240000],
 [242500],
 [245000],
 [247500],
 [250000],
 [252500],
 [255000],
 [257500],
 [260000],
 [262500],
 [265000],
 [267500],
 [270000],
 [272500],
 [275000],
 [277500],
 [280000],
 [282500],
 [285000],
 [287500],
 [290000],
 [292500],
 [295000],
 [297500],
 [300000],
 [302500],
 [305000],
 [307500],
 [310000],
 [312500],
 [315000],
 [317500],
 [320000],
 [322500],
 [325000],
 [327500],
 [330000],
 [332500],
 [335000],
 [337500],
 [340000],
 [342500],
 [345000],
 [347500],
 [350000],
 [352500],
 [355000],
 [357500],
 [360000],
 [362500],
 [365000],
 [367500],
 [370000],
 [372500],
 [375000],
 [377500],
 [380000],
 [382500],
 [385000],
 [387500],
 [390000],
 [392500],
 [395000],
 [397500],
 [400000],
 [402500],
 [405000],
 [407500],
 [410000],
 [412500],
 [415000],
 [417500],
 [420000],
 [422500],
 [425000],
 [427500],
 [430000],
 [432500],
 [435000],
 [437500],
 [440000],
 [442500],
 [445000],
 [447500],
 [450000],
 [452500],
 [455000],
 [457500],
 [460000],
 [462500],
 [465000],
 [467500],
 [470000],
 [472500],
 [475000],
 [477500],
 [480000],
 [482500],
 [485000],
 [487500],
 [490000],
 [492500],
 [495000],
 [497500],
 [500000],
 [502500],
 [505000],
 [507500],
 [510000],
 [512500],
 [515000],
 [517500],
 [520000],
 [522500],
 [525000],
 [527500],
 [530000],
 [532500],
 [535000],
 [537500],
 [540000],
 [542500],
 [545000],
 [547500],
 [550000],
 [552500],
 [555000],
 [557500],
 [560000],
 [562500],
 [565000],
 [567500],
 [570000],
 [572500],
 [575000],
 [577500],
 [580000],
 [582500],
 [585000],
 [587500],
 [590000],
 [592500],
 [595000],
 [597500],
 [600000],
 [602500],
 [605000],
 [607500],
 [610000],
 [612500],
 [615000],
 [617500],
 [620000],
 [622500],
 [625000],
 [627500],
 [630000],
 [632500],
 [635000],
 [637500],
 [640000],
 [642500],
 [645000],
 [647500],
 [650000],
 [652500],
 [655000],
 [657500],
 [660000],
 [662500],
 [665000],
 [667500],
 [670000],
 [672500],
 [675000],
 [677500],
 [680000],
 [682500],
 [685000],
 [687500],
 [690000],
 [692500],
 [695000],
 [697500],
 [700000],
 [702500],
 [705000],
 [707500],
 [710000],
 [712500],
 [715000],
 [717500],
 [720000],
 [722500],
 [725000],
 [727500],
 [730000],
 [732500],
 [735000],
 [737500],
 [740000],
 [742500],
 [745000],
 [747500],
 [750000],
 [752500],
 [755000],
 [757500],
 [760000],
 [762500],
 [765000],
 [767500],
 [770000],
 [772500],
 [775000],
 [777500],
 [780000],
 [782500],
 [785000],
 [787500],
 [790000],
 [792500],
 [795000],
 [797500],
 [800000],
 [802500],
 [805000],
 [807500],
 [810000],
 [812500],
 [815000],
 [817500],
 [820000],
 [822500],
 [825000],
 [827500],
 [830000],
 [832500],
 [835000],
 [837500],
 [840000],
 [842500],
 [845000],
 [847500],
 [850000],
 [852500],
 [855000],
 [857500],
 [860000],
 [862500],
 [865000],
 [867500],
 [870000],
 [872500],
 [875000],
 [877500],
 [880000],
 [882500],
 [885000],
 [887500],
 [890000],
 [892500],
 [895000],
 [897500],
 [900000],
 [902500],
 [905000],
 [907500],
 [910000],
 [912500],
 [915000],
 [917500],
 [920000],
 [922500],
 [925000],
 [927500],
 [930000],
 [932500],
 [935000],
 [937500],
 [940000],
 [942500],
 [945000],
 [947500],
 [950000],
 [952500],
 [955000],
 [957500],
 [960000],
 [962500],
 [965000],
 [967500],
 [970000],
 [972500],
 [975000],
 [977500],
 [980000],
 [982500],
 [985000],
 [987500],
 [990000],
 [992500],
 [995000],
 [997500],
 [1000000],
 [1002500],
 [1005000],
 [1007500],
 [1010000],
 [1012500],
 [1015000],
 [1017500],
 [1020000],
 [1022500],
 [1025000],
 [1027500],
 [1030000],
 [1032500],
 [1035000],
 [1037500],
 [1040000],
 [1042500],
 [1045000],
 [1047500],
 [1050000],
 [1052500],
 [1055000],
 [1057500],
 [1060000],
 [1062500],
 [1065000],
 [1067500],
 [1070000],
 [1072500],
 [1075000],
 [1077500],
 [1080000],
 [1082500],
 [1085000],
 [1087500],
 [1090000],
 [1092500],
 [1095000],
 [1097500],
 [1100000],
 [1102500],
 [1105000],
 [1107500],
 [1110000],
 [1112500],
 [1115000],
 [1117500],
 [1120000],
 [1122500],
 [1125000],
 [1127500],
 [1
```

```
In [21]: 1 total_deaths
```

```
Out[21]: array([[ 17],
 [ 18],
 [ 26],
 [ 42],
 [ 56],
 [ 82],
 [131],
 [133],
 [171],
 [213],
 [259],
 [362],
 [426],
 [492],
 [564],
 [634],
 [719],
 [806],
 [906],
 [1012],
 [1122],
 [1236],
 [1354],
 [1476],
 [1602],
 [1732],
 [1866],
 [2004],
 [2146],
 [2292],
 [2442],
 [2596],
 [2754],
 [2916],
 [3082],
 [3252],
 [3426],
 [3604],
 [3786],
 [3972],
 [4162],
 [4356],
 [4554],
 [4756],
 [4962],
 [5172],
 [5386],
 [5604],
 [5826],
 [6052],
 [6282],
 [6516],
 [6754],
 [6996],
 [7242],
 [7492],
 [7746],
 [8004],
 [8266],
 [8532],
 [8802],
 [9076],
 [9354],
 [9636],
 [9922],
[10212],
[10506],
[10804],
[11106],
[11412],
[11722],
[12036],
[12354],
[12676],
[13002],
[13332],
[13666],
[14004],
[14346],
[14692],
[15042],
[15396],
[15754],
[16116],
[16482],
[16852],
[17226],
[17604],
[17986],
[18372],
[18762],
[19156],
[19554],
[19956],
[20362],
[20772],
[21186],
[21604],
[22026],
[22452],
[22882],
[23316],
[23754],
[24196],
[24642],
[25092],
[25546],
[26004],
[26466],
[26932],
[27402],
[27876],
[28354],
[28836],
[29322],
[29812],
[30306],
[30804],
[31306],
[31812],
[32322],
[32836],
[33354],
[33876],
[34402],
[34932],
[35466],
[36004],
[36546],
[37092],
[37642],
[38196],
[38754],
[39316],
[39882],
[40452],
[41026],
[41604],
[42186],
[42772],
[43362],
[43956],
[44554],
[45156],
[45762],
[46372],
[46986],
[47604],
[48226],
[48852],
[49482],
[50116],
[50754],
[51396],
[52042],
[52692],
[53346],
[54004],
[54666],
[55332],
[56002],
[56676],
[57354],
[58036],
[58722],
[59412],
[60106],
[60804],
[61506],
[62212],
[62922],
[63636],
[64354],
[65076],
[65802],
[66532],
[67266],
[68004],
[68746],
[69492],
[70242],
[71006],
[71774],
[72546],
[73322],
[74102],
[74886],
[75674],
[76466],
[77262],
[78062],
[78866],
[79674],
[80486],
[81302],
[82122],
[82946],
[83774],
[84606],
[85442],
[86282],
[87126],
[87974],
[88826],
[89682],
[90542],
[91406],
[92274],
[93146],
[94022],
[94902],
[95786],
[96674],
[97566],
[98462],
[99362],
[100266],
[101174],
[102086],
[102992],
[103902],
[104816],
[105734],
[106656],
[107582],
[108512],
[109446],
[110384],
[111326],
[112272],
[113222],
[114176],
[115134],
[116096],
[117062],
[118032],
[119006],
[120004],
[121006],
[122012],
[123022],
[124036],
[125054],
[126076],
[127102],
[128132],
[129166],
[130204],
[131246],
[132292],
[133342],
[134396],
[135454],
[136516],
[137582],
[138652],
[139726],
[140804],
[141886],
[142972],
[144062],
[145156],
[146254],
[147356],
[148462],
[149572],
[150686],
[151804],
[152926],
[154052],
[155182],
[156316],
[157454],
[158596],
[159742],
[160892],
[162046],
[163204],
[164366],
[165532],
[166702],
[167876],
[169054],
[170236],
[171422],
[172612],
[173806],
[175004],
[176206],
[177412],
[178622],
[179836],
[181054],
[182276],
[183502],
[184732],
[185966],
[187204],
[188446],
[189692],
[190942],
[192196],
[193454],
[194716],
[195982],
[197252],
[198526],
[199804],
[201086],
[202372],
[203662],
[204956],
[206254],
[207556],
[208862],
[210172],
[211486],
[212804],
[214126],
[215452],
[216782],
[218116],
[219454],
[220796],
[222142],
[223492],
[224846],
[226204],
[227566],
[228932],
[230302],
[231676],
[233054],
[234436],
[235816],
[237202],
[238592],
[239986],
[241384],
[242786],
[244192],
[245602],
[247016],
[248434],
[249856],
[251282],
[252712],
[254146],
[255584],
[257026],
[258472],
[259922],
[261376],
[262834],
[264296],
[265762],
[267232],
[268706],
[270184],
[271666],
[273152],
[274642],
[276136],
[277634],
[279136],
[280642],
[282152],
[283666],
[285184],
[286706],
[288232],
[289762],
[291296],
[292834],
[294376],
[295922],
[297472],
[299026],
[300584],
[302146],
[303712],
[305282],
[306856],
[308434],
[310016],
[311602],
[313192],
[314786],
[316384],
[317986],
[319592],
[321202],
[322816],
[324434],
[326056],
[327682],
[329312],
[330946],
[332584],
[334226],
[335872],
[337522],
[339176],
[340834],
[342496],
[344162],
[345832],
[347506],
[349184],
[350866],
[352552],
[354242],
[355936],
[357634],
[359336],
[361042],
[362752],
[364466],
[366184],
[367906],
[369632],
[371362],
[373096],
[374834],
[376576],
[378322],
[380072],
[381826],
[383584],
[385346],
[387112],
[388882],
[390656],
[392434],
[394216],
[396002],
[397792],
[399586],
[401384],
[403186],
[404992],
[406802],
[408616],
[410434],
[412256],
[414082],
[415912],
[417746],
[419584],
[421426],
[423272],
[425122],
[426976],
[428834],
[430696],
[432562],
[434432],
[436306],
[438184],
[440066],
[441952],
[443842],
[445736],
[447634],
[449536],
[451442],
[453352],
[455266],
[457184],
[459106],
[461032],
[462962],
[464896],
[466834],
[468776],
[470722],
[472672],
[474626],
[476584],
[478546],
[480512],
[482482],
[484456],
[486434],
[488416],
[490402],
[492392],
[4943
```

```
In [22]: 1 total_recovered
```

```
Out[22]: array([[ 28],
 [ 30],
 [ 36],
 [ 39],
 [ 52],
 [ 61],
 [ 107],
 [ 126],
 [ 143],
 [ 222],
 [ 284],
 [ 472],
 [ 623],
 [ 852],
 [ 1124],
 [ 1487],
 [ 2011],
 [ 2616],
 [ 3244],
 [ 3855],
 [ 4531],
 [ 5262],
 [ 6048],
 [ 6889],
 [ 7796],
 [ 8769],
 [ 9808],
 [10913],
 [12084],
 [13321],
 [14624],
 [15993],
 [17428],
 [18929],
 [20496],
 [22129],
 [23838],
 [25613],
 [27454],
 [29361],
 [31334],
 [33373],
 [35478],
 [37649],
 [39886],
 [42189],
 [44558],
 [46993],
 [49494],
 [52061],
 [54694],
 [57393],
 [60158],
 [62989],
 [65886],
 [68849],
 [71878],
 [74973],
 [78134],
 [81361],
 [84654],
 [88013],
 [91438],
 [94929],
 [98486],
 [102109],
 [105798],
 [109553],
 [113374],
 [117261],
 [121214],
 [125233],
 [129318],
 [133469],
 [137686],
 [141969],
 [146318],
 [150733],
 [155214],
 [159761],
 [164374],
 [169053],
 [173798],
 [178609],
 [183486],
 [188429],
 [193438],
 [198503],
 [203634],
 [208831],
 [214094],
 [219423],
 [224818],
 [230279],
 [235806],
 [241399],
 [247058],
 [252783],
 [258574],
 [264431],
 [270354],
 [276343],
 [282398],
 [288519],
 [294706],
 [300959],
 [307278],
 [313663],
 [320114],
 [326631],
 [333214],
 [339863],
 [346578],
 [353359],
 [360206],
 [367119],
 [374098],
 [381143],
 [388254],
 [395431],
 [402674],
 [409983],
 [417358],
 [424799],
 [432306],
 [439879],
 [447518],
 [455223],
 [462994],
 [470831],
 [478734],
 [486703],
 [494738],
 [502839],
 [510906],
 [519039],
 [527238],
 [535503],
 [543834],
 [552231],
 [560694],
 [569223],
 [577818],
 [586479],
 [595206],
 [603999],
 [612858],
 [621783],
 [630774],
 [639831],
 [648954],
 [658143],
 [667398],
 [676719],
 [686106],
 [695559],
 [705078],
 [714663],
 [724314],
 [734031],
 [743814],
 [753663],
 [763578],
 [773559],
 [783606],
 [793719],
 [803898],
 [814143],
 [824454],
 [834831],
 [845274],
 [855783],
 [866358],
 [876999],
 [887706],
 [898479],
 [909318],
 [920223],
 [931194],
 [942231],
 [953334],
 [964493],
 [975718],
 [987009],
 [998366],
 [1009789],
 [1021278],
 [1032833],
 [1044454],
 [1056141],
 [1067894],
 [1079713],
 [1091598],
 [1103549],
 [1115566],
 [1127649],
 [1139798],
 [1151913],
 [1164094],
 [1176341],
 [1188654],
 [1201033],
 [1213478],
 [1225989],
 [1238566],
 [1251209],
 [1263918],
 [1276693],
 [1289534],
 [1302441],
 [1315414],
 [1328453],
 [1341558],
 [1354729],
 [1367966],
 [1381269],
 [1394638],
 [1408073],
 [1421574],
 [1435141],
 [1448774],
 [1462473],
 [1476238],
 [1490069],
 [1503966],
 [1517929],
 [1531958],
 [1546053],
 [1560214],
 [1574441],
 [1588734],
 [1603093],
 [1617518],
 [1631999],
 [1646536],
 [1661139],
 [1675808],
 [1690543],
 [1705344],
 [1720201],
 [1735124],
 [1750113],
 [1765168],
 [1780289],
 [1795476],
 [1810729],
 [1826048],
 [1841433],
 [1856884],
 [1872391],
 [1887954],
 [1903573],
 [1919248],
 [1934979],
 [1950766],
 [1966609],
 [1982508],
 [1998463],
 [2014474],
 [2030541],
 [2046664],
 [2062843],
 [2079078],
 [2095369],
 [2111716],
 [2128119],
 [2144578],
 [2161098],
 [2177679],
 [2194321],
 [2210924],
 [2227589],
 [2244316],
 [2261005],
 [2277756],
 [2294569],
 [2311444],
 [2328381],
 [2345380],
 [2362441],
 [2379564],
 [2396749],
 [2413996],
 [2431305],
 [2448676],
 [2466109],
 [2483604],
 [2501161],
 [2518780],
 [2536461],
 [2554204],
 [2572009],
 [2589876],
 [2607805],
 [2625796],
 [2643849],
 [2661964],
 [2680141],
 [2698380],
 [2716681],
 [2735044],
 [2753469],
 [2771956],
 [2790505],
 [2809116],
 [2827789],
 [2846524],
 [2865321],
 [2884180],
 [2903101],
 [2922084],
 [2941129],
 [2960236],
 [2979405],
 [2998636],
 [3017929],
 [3037284],
 [3056701],
 [3076180],
 [3095721],
 [3115324],
 [3134989],
 [3154716],
 [3174505],
 [3194356],
 [3214269],
 [3234244],
 [3254281],
 [3274380],
 [3294541],
 [3314764],
 [3335049],
 [3355396],
 [3375805],
 [3396276],
 [3416809],
 [3437404],
 [3458061],
 [3478780],
 [3499561],
 [3520404],
 [3541309],
 [3562276],
 [3583305],
 [3604396],
 [3625549],
 [3646764],
 [3668041],
 [3689380],
 [3710781],
 [3732244],
 [3753769],
 [3775356],
 [3796005],
 [3816716],
 [3837489],
 [3858324],
 [3879221],
 [3900180],
 [3921201],
 [3942284],
 [3963429],
 [3984636],
 [4005905],
 [4027236],
 [4048629],
 [4069984],
 [4091401],
 [4112880],
 [4134421],
 [4155924],
 [4177489],
 [4199016],
 [4220605],
 [4242256],
 [4263969],
 [4285744],
 [4307581],
 [4329480],
 [4351441],
 [4373464],
 [4395549],
 [4417696],
 [4439905],
 [4462176],
 [4484509],
 [4506904],
 [4529361],
 [4551880],
 [4574461],
 [4597104],
 [4619809],
 [4642576],
 [4665405],
 [4688296],
 [4711249],
 [4734264],
 [4757341],
 [4780480],
 [4803681],
 [4826944],
 [4850269],
 [4873646],
 [4897085],
 [4920586],
 [4944149],
 [4967774],
 [4991461],
 [5015212],
 [5039027],
 [5062904],
 [5086843],
 [5110844],
 [5134907],
 [5159032],
 [5183219],
 [5207468],
 [5231779],
 [5256152],
 [5280587],
 [5305084],
 [5329643],
 [5354264],
 [5378947],
 [5403692],
 [5428499],
 [5453368],
 [5478299],
 [5503292],
 [5528347],
 [5553464],
 [5578643],
 [5603884],
 [5629187],
 [5654552],
 [5679979],
 [5705468],
 [5731019],
 [5756632],
 [5782307],
 [5808044],
 [5833843],
 [5859704],
 [5885627],
 [5911612],
 [5937659],
 [5963768],
 [5989939],
 [6016172],
 [6042467],
 [6068824],
 [6095243],
 [6121724],
 [6148267],
 [6174872],
 [6201539],
 [6228268],
 [6255059],
 [6281912],
 [6308827],
 [6335804],
 [6362843],
 [6389944],
 [6417107],
 [6444332],
 [6471619],
 [6498968],
 [6526379],
 [6553852],
 [6581387],
 [6608984],
 [6636643],
 [6664364],
 [6692147],
 [6719992],
 [6747909],
 [6775888],
 [6803929],
 [6832032],
 [6860197],
 [6888424],
 [6916713],
 [6945064],
 [6973477],
 [7001952],
 [7030489],
 [7059088],
 [7087749],
 [7116472],
 [7145257],
 [7174104],
 [7203013],
 [7231984],
 [7261017],
 [7290112],
 [7319269],
 [7348488],
 [7377769],
 [7407112],
 [7436517],
 [7465984],
 [7495513],
 [7525104],
 [7554757],
 [7584472],
 [7614249],
 [7644088],
 [7673989],
 [7703952],
 [7733977],
 [7764064],
 [7794213],
 [7824424],
 [7854697],
 [7885032],
 [7915429],
 [7945888],
 [7976409],
 [8006992],
 [8037637],
 [8068344],
 [8099113],
 [8129944],
 [8160837],
 [8191792],
 [8222809],
 [8253888],
 [8285029],
 [8316232],
 [8347497],
 [8378824],
 [8410213],
 [8441664],
 [8473177],
 [8504752],
 [8536389],
 [8568088],
 [8599849],
 [8631672],
 [8663557],
 [8695504],
 [8727513],
 [8759584],
 [8791717],
 [8823912],
 [8856169],
 [8888488],
 [8920869],
 [8953312],
 [8985817],
 [9018384],
 [9050913],
 [9083504],
 [9116157],
 [9148872],
 [9181649],
 [9214488],
 [9247389],
 [9280352],
 [9313377],
 [9346454],
 [9379593],
 [9412794],
 [9446057],
 [9479382],
 [9512769],
 [9546218],
 [9579729],
 [9613292],
 [9646917],
 [9680594],
 [9714333],
 [9748134],
 [9781997],
 [9815922],
 [9849909],
 [9883958],
 [9918069],
 [9952242],
 [9986477],
 [10020774],
 [10055133],
 [10089554],
 [10124037],
 [10158582],
 [10193189],
 [10227858],
 [10262589],
 [10297382],
 [10332237],
 [10367154],
 [10402133],
 [10437174],
 [10472277],
 [10507442],
 [10542669],
 [10577958],
 [10613309],
 [10648722],
 [10684197],
 [10719734],
 [10755333],
 [10790994],
 [10826717],
 [10862502],
 [10898349],
 [10934258],
 [10970229],
 [11006262],
 [11042357],
 [11078504],
 [11114713],
 [11150984],
 [11187317],
 [11223712],
 [11260169],
 [11296688],
 [11333269],
 [11369912],
 [11406617],
 [11443384],
 [11480213],
 [11517104],
 [11554057],
 [11591072],
 [11628149],
 [11665288],
 [11702489],
 [11739752],
 [11777077],
 [11814464],
 [11851913],
 [11889424],
 [11926997],
 [11964632],
 [12002329],
 [12040088],
 [12077909],
 [12115792],
 [12153737],
 [12191744],
 [12229813],
 [12267944],
 [12306137],
 [12344392],
 [12382709],
 [12421088],
 [12459529],
 [12498032],
 [12536597],
 [12575224],
 [12613913],
 [12652664],
 [12691477],
 [12730352],
 [12769289],
 [12808288],
 [12847349],
 [12886472],
 [12925657],
 [12964904],
 [13004213],
 [13043584],
 [13083017],
 [13122512],
 [13162069],
 [13201688],
 [13241369],
 [13281112],
 [13320917],
 [13360784],
 [13400713],
 [13440704],
 [13480757],
 [13520872],
 [13561049],
 [13601288],
 [13641589],
 [13681952],
 [13722377],
 [13762864],
 [13803413],
 [13844024],
 [13884697],
 [13925432],
 [13966229],
 [14007088],
 [14048009],
 [14088992],
 [14129037],
 [14169144],
 [14209313],
 [14249544],
 [14289837],
 [14330192],
 [14370609],
 [14411088],
 [14451629],
 [14492232],
 [14532897],
 [14573624],
 [14614413],
 [14655264],
 [14696177],
 [14737152],
 [14778189],
 [14819288],
 [14860449],
 [14901672],
 [14942957],
 [14984304],
 [15025713],
 [15067184],
 [15108717],
 [15150312],
 [15191969],
 [15233688],
 [15275469],
 [15317312],
 [15359217],
 [15401184],
 [15443213],
 [15485304],
 [15527457],
 [15569672],
 [15611949],
 [15654288],
 [15696689],
 [15739152],
 [15781677],
 [15824264],
 [15866913],
 [15909624],
 [15952397],
 [15995232],
 [16038129],
 [16081088],
 [16124109],
 [16167192],
 [16210337],
 [16253544],
 [16296813],
 [16340144],
 [16383537],
 [16426992],
 [16470509],
 [16514088],
 [16557729],
 [16601432],
 [16645197],
 [16689024],
 [16732913],
 [16776864],
 [16820877],
 [16864952],
 [16909089],
 [16953288],
 [16997549],
 [17041872],
 [17086257],
 [17130704],
 [17175213],
 [17219784],
 [17264417],
 [17309112],
 [17353869],
 [17398688],
 [17443569],
 [17488512],
 [17533517],
 [17578584],
 [17623713],
 [17668904],
 [17714157],
 [17759472],
 [17804849],
 [17850288],
 [17895789],
 [17941352],
 [17986977],
 [18032664],
 [18078413],
 [18124224],
 [18170097],
 [18216032],
 [18262029],
 [18308088],
 [18354209],
 [18400392],
 [18446637],
 [18492944],
 [18539313],
 [18585744],
 [18632237],
 [18678792],
 [18725409],
 [18772088],
 [18818829],
 [18865632],
 [18912497],
 [18959424],
 [19006413],
 [19053464],
 [19100577],
 [19147752],
 [19194989],
 [19242288],
 [19289649],
 [19337072],
 [19384557],
 [19432094],
 [19479693],
 [19527354],
 [19575077],
 [19622862],
 [19670709],
 [19718618],
 [19766589],
 [19814624],
 [19862723],
 [19910884],
 [19959109],
 [20007398],
 [20055751],
 [20104168],
 [20152649],
 [20201194],
 [20249803],
 [20298476],
 [20347213],
 [20396014],
 [20444879],
 [20493808],
 [20542801],
 [20591858],
 [20640979],
 [20690164],
 [20739413],
 [20788726],
 [20838103],
 [20887544],
 [20937049],
 [20986618],
 [21036251],
 [21085948],
 [21135709],
 [21185534],
 [21235423],
 [21285376],
 [21335393],
 [21385474],
 [21435619],
 [21485828],
 [21536091],
 [21586418],
 [21636809],
 [21687264],
 [21737783],
 [21788366],
 [21838913],
 [21889524],
 [21940199],
 [21990938],
 [22041741],
 [22092608],
 [22143539],
 [22194534],
 [22245593],
 [22296716],
 [22347903],
 [22399154],
 [22450469],
 [22501848],
 [22553291],
 [22604808],
 [22656389],
 [22708034],
 [22759743],
 [22811516],
 [22863353],
 [22915254],
 [22967219],
 [23019248],
 [23071341],
 [23123498],
 [23175719],
 [23227994],
 [23280333],
 [23332736],
 [23385203],
 [23437734],
 [23490329],
 [23542988],
 [23595711],
 [23648498],
 [23701349],
 [23754264],
 [23807243],
 [23860286],
 [23913393],
 [23966564],
 [24019799],
 [24073098],
 [24126461],
 [24179888],
 [24233379],
 [24286934],
 [24340553],
 [24394236],
 [24447983],
 [24501794],
 [24555669],
 [24609608],
 [24663611],
 [24717678],
 [24771809],
 [24825994],
 [24880243],
 [24934556],
 [24988933],
 [25043374],
 [25097879],
 [25152448],
 [25207081],
 [25261778],
 [25316539],
 [25371364],
 [25426253],
 [25481206],
 [25536223],
 [25591304],
 [25646449],
 [25701658],
 [25756931],
 [25812268],
 [25867669],
 [25923134],
 [25978663],
 [26034256],
 [26089913],
 [26145634],
 [26201419],
 [26257268],
 [26313181],
 [26369158],
 [26425199],
 [26481304],
 [26537473],
 [26593706],
 [26649993],
 [26706344],
 [26762759],
 [26819238],
 [26875781],
 [26932388],
 [26989059],
 [27045794],
 [27102593],
 [27159456],
 [27216383],
 [27273374],
 [27330429],
 [27387548],
 [27444731],
 [27501978],
 [27559289],
 [27616664],
 [27674103],
 [27731606],
 [27789173],
 [27846804],
 [27904499],
 [27962258],
 [28020081],
 [28077968],
 [28135919],
 [28193934],
 [28252013],
 [28310156],
 [28368363],
 [28426634],
 [28484969],
 [28543368],
 [28601831],
 [28660358],
 [28718949],
 [28777604],
 [28836323],
 [28895106],
 [28953953],
 [29012864],
 [29071839],
 [29130878],
 [29189981],
 [29249148],
 [29308379],
 [29367674],
 [29427033],
 [29486456],
 [29545943],
 [29605494],
 [29665109],
 [29724788],
 [29784531],
 [29844338],
 [29904209],
 [29964144],
 [30024143],
 [30084206],
 [30144333],
 [30204524],
 [30264779],
 [30325098],
 [30385481],
 [30445928],
 [30506439],
 [30566914],
 [30627453],
 [30688056],
 [30748723],
 [30809454],
 [30870249],
 [30931108],
 [30992031],
 [31053018],
 [31114069],
 [31175184],
 [31236363],
 [31297606],
 [31358913],
 [31420284],
 [31481719],
 [31543218],
 [31604781],
 [31666408],
 [31728099],
 [31789854],
 [31851673],
 [31913556],
 [31975503],
 [32037514],
 [32099589],
 [32161728],
 [32223931],
 [32286198],
 [32348529],
 [32410924],
 [32473383],
 [32535906],
 [32598493],
 [32661144],
 [32723859],
 [32786638],
 [32849481],
 [32912388],
 [32975359],
 [33038394],
 [33101493],
 [33164656],
 [33227881],
 [33291168],
 [33354519],
 [33417934],
 [33481413],
 [33544956],
 [33608569],
 [33672244],
 [33735981],
 [33799781],
 [33863644],
 [33927569],
 [33991556],
 [34055607],
 [34119722],
 [34183901],
 [34248144],
 [34312451],
 [34376822],
 [34441257],
 [34505756],
 [34570319],
 [34634946],
 [34699637],
 [34764392],
 [34829211],
 [34894094],
 [34959041],
 [35024052],
 [35089127],
 [35154266],
 [35219469],
 [35284736],
 [35350067],
 [35415462],
 [35480921],
 [35546444],
 [35612031],
 [35677682],
 [35743397],
 [35809176],
 [35875019],
 [35940926],
 [36006897],
 [36072932],
 [36139031],
 [36205194],
 [36271421],
 [36337712],
 [36404067],
 [36470486],
 [36536969],
 [36603516],
 [36670127],
 [36736802],
 [36803541],
 [36870344],
 [36937211],
 [37004142],
 [37071137],
 [37138196],
 [37205319],
 [37272506],
 [37339757],
 [37407072],
 [37474451],
 [37541894],
 [37609401],
 [37676972],
 [37744607],
 [37812306],
 [37880069],
 [37947896],
 [38015787],
 [38083742],
 [38151761],
 [38219844],
 [38287991],
 [38356202],
 [38424477],
 [38492816],
 [38561219],
 [38629686],
 [38698217],
 [38766812],
 [38835471],
 [38904194],
 [38973081],
 [39041932],
 [39110847],
 [39179826],
 [39248869],
 [39317981],
 [39387152],
 [39456383],
 [39525674],
 [39595025],
 [39664436],
 [39733907],
 [39803438],
 [39873029],
 [39942680],
 [40012391],
 [40082162],
 [40152003],
 [40221914],
 [40291885],
 [40361916],
 [40432007],
 [40502158],
 [40572369],
 [40642640],
 [40712971],
 [40783362],
 [40853813],
 [40924324],
 [40994895],
 [41065526],
 [41136217],
 [41206968],
 [41277779],
 [41348650],
 [41419581],
 [41490572],
 [41561623],
 [41632734],
 [41703905],
 [41775136],
 [41846427],
 [41917778],
 [41989189],
 [42060660],
 [42132191],
 [42203882],
 [42275633],
 [42347444],
 [42419315],
 [4249124
```

```
In [28]: 1 # The next line of code will basically calculate the total number of confirmed cases by each country.
2
3 country_confirmed_cases = []
4 no_cases = []
5 for i in unique_countries:
6     cases = latest_confirmed[confirmed_cases['Country/Region']==i].sum()
7     if cases > 0:
8         country_confirmed_cases.append(cases)
9     else:
10        no_cases.append(i)
11
12 for i in no_cases:
13     unique_countries.remove(i)
14
15 unique_countries = [k for k, v in sorted(zip(unique_countries, country_confirmed_cases), key=operator.itemgetter(1), reverse=True)]
16 for i in range(len(unique_countries)):
17     country_confirmed_cases[i] = latest_confirmed[confirmed_cases['Country/Region']==unique_countries[i]].sum()
```

```
In [29]: 1 # Number of cases per Country/Region
2 print('Confirmed Cases by Countries/Regions:')
3 for i in range(len(unique_countries)):
4     print(f'{unique_countries[i]}: {country_confirmed_cases[i]} cases')
```

Confirmed Cases by Countries/Regions:

US: 2312302 cases
Brazil: 1106470 cases
Russia: 591465 cases
India: 440215 cases
United Kingdom: 306761 cases
Peru: 257447 cases
Chile: 246963 cases
Spain: 246504 cases
Italy: 238720 cases
Iran: 207525 cases
France: 197381 cases
Germany: 191768 cases
Turkey: 188897 cases
Mexico: 185122 cases
Pakistan: 185034 cases
Saudi Arabia: 161005 cases
Bangladesh: 115786 cases
Canada: 103418 cases
South Africa: 101500 cases

```
In [30]: 1 # Find the list of unique provinces
2 unique_provinces = list(confirmed_cases['Province/State'].unique())
```

```
In [31]: 1 # Finding the number of confirmed cases per provinces,state or city.
2 province_confirmed_cases = []
3 no_cases = []
4 for i in unique_provinces:
5     cases = latest_confirmed[confirmed_cases['Province/State']==i].sum()
6     if cases > 0:
7         province_confirmed_cases.append(cases)
8     else:
9         no_cases.append(i)
10 # remove areas with no confirmed cases
11 for i in no_cases:
12     unique_provinces.remove(i)
```

```
In [32]: 1 # Number of cases per province/state/city
2 for i in range(len(unique_provinces)):
3     print(f'{unique_provinces[i]}: {province_confirmed_cases[i]} cases')
```

Australian Capital Territory: 108 cases
New South Wales: 3150 cases
Northern Territory: 29 cases
Queensland: 1066 cases
South Australia: 440 cases
Tasmania: 228 cases
Victoria: 1864 cases
Western Australia: 607 cases
Alberta: 7736 cases
British Columbia: 2822 cases
Grand Princess: 13 cases
Manitoba: 314 cases
New Brunswick: 164 cases
Newfoundland and Labrador: 261 cases
Nova Scotia: 1061 cases
Ontario: 35418 cases
Prince Edward Island: 27 cases
Quebec: 54835 cases
Saskatchewan: 751 cases
Yukon: 22 cases

In [33]:

```
1 # Handling nan values if there is any,it is usually a float: float('nan')
2
3 nan_indices = []
4
5 for i in range(len(unique_provinces)):
6     if type(unique_provinces[i]) == float:
7         nan_indices.append(i)
8
9 unique_provinces = list(unique_provinces)
10 province_confirmed_cases = list(province_confirmed_cases)
11
12 for i in nan_indices:
13     unique_provinces.pop(i)
14     province_confirmed_cases.pop(i)
```



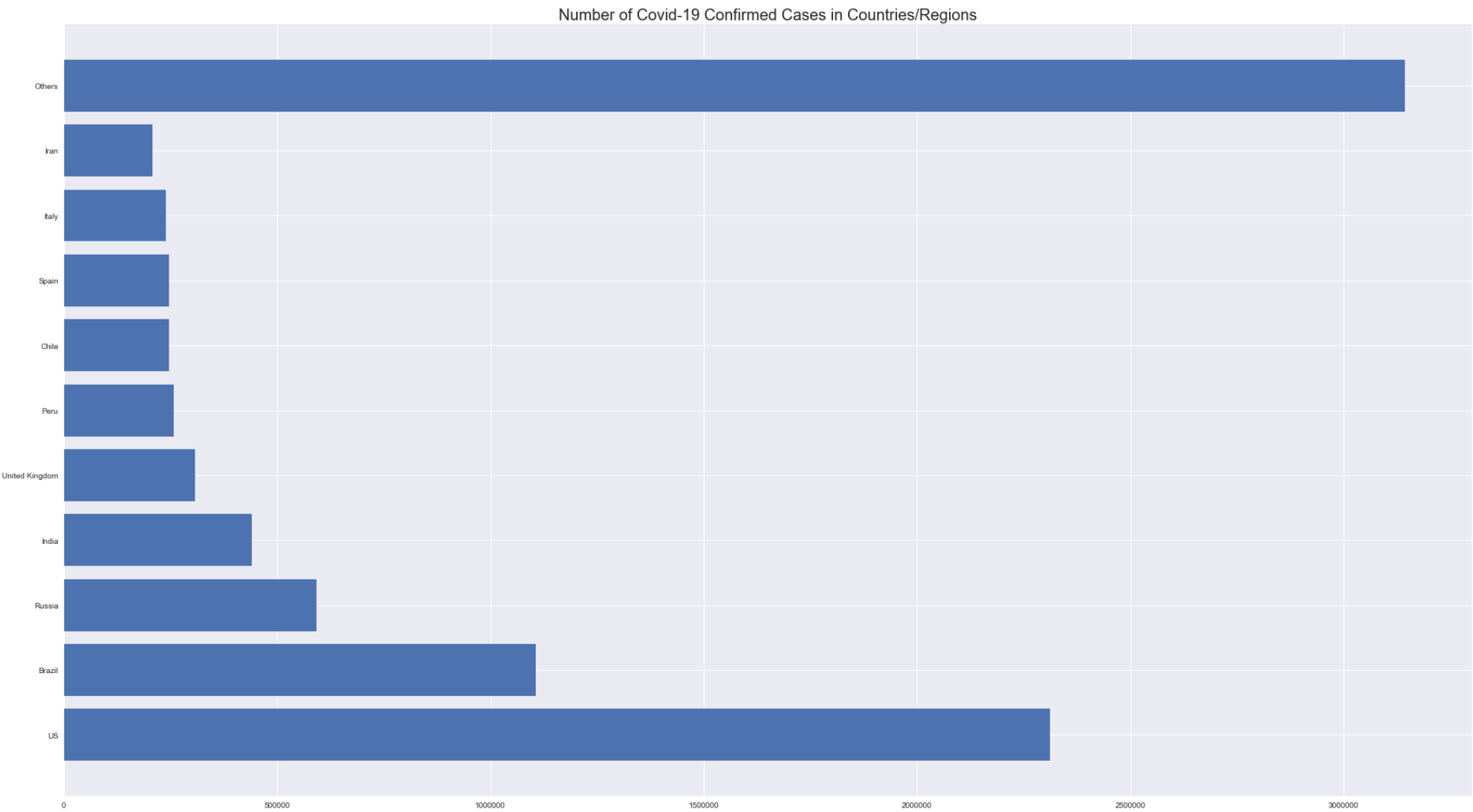
In [35]:

```
1 # Only show 10 countries with the most confirmed cases, the rest are grouped into the  category named others
2 visual_unique_countries = []
3 visual_confirmed_cases = []
4 others = np.sum(country_confirmed_cases[10:])
5 for i in range(len(country_confirmed_cases[:10])):
6     visual_unique_countries.append(unique_countries[i])
7     visual_confirmed_cases.append(country_confirmed_cases[i])
8
9 visual_unique_countries.append('Others')
10 visual_confirmed_cases.append(others)
```

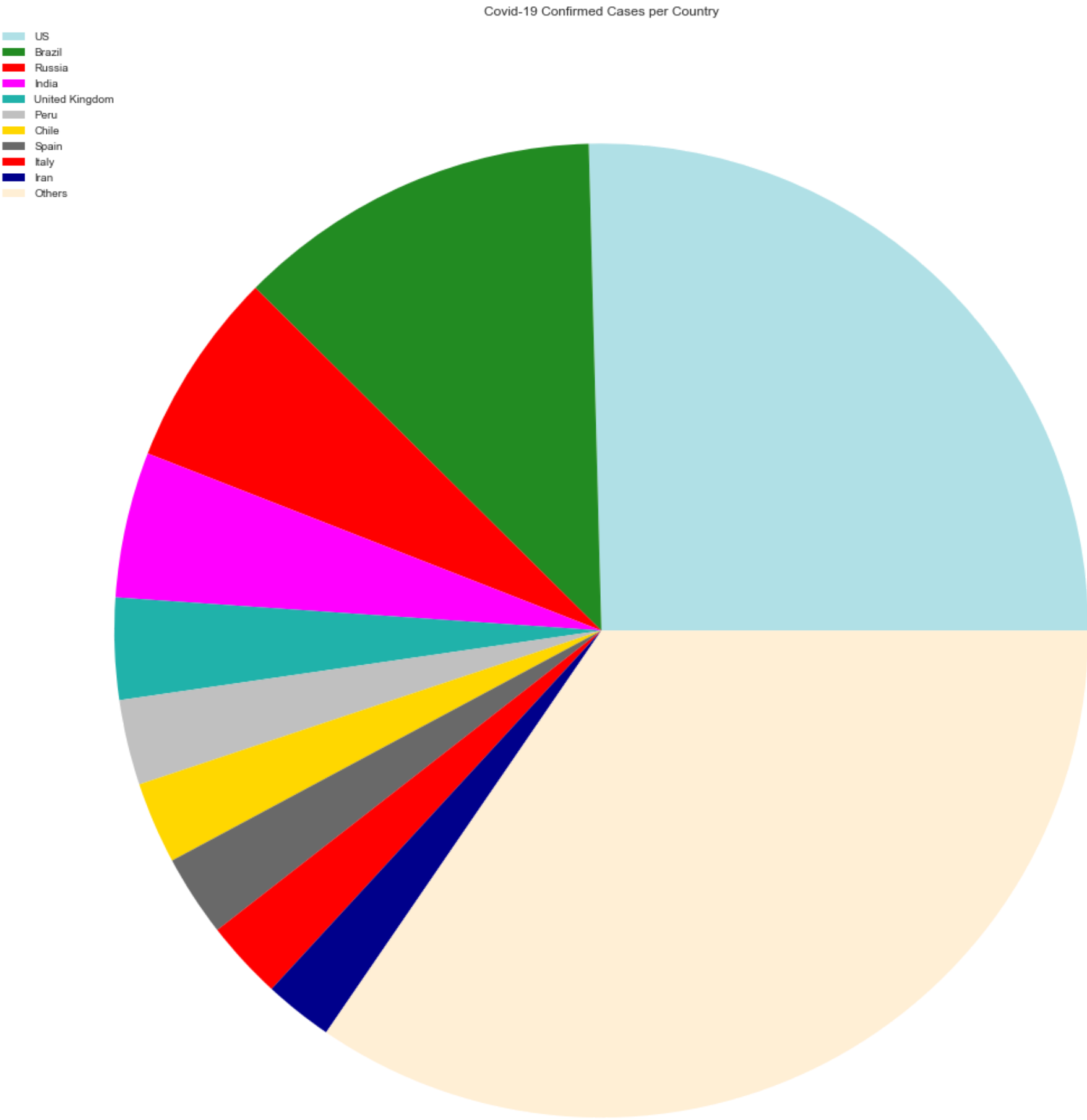
Visual Representations (bar charts and pie charts)

In [36]:

```
1 # Visualize the 10 Countries
2 plt.figure(figsize=(32, 18))
3 plt.barh(visual_unique_countries, visual_confirmed_cases)
4 plt.title('Number of Covid-19 Confirmed Cases in Countries/Regions', size=20)
5 plt.show()
```



```
In [37]: 1 # Create a Pie chart to see the total confirmed cases in 10 different countries.
2
3 c = random.choices(list(mcolors.CSS4_COLORS.values()),k = len(unique_countries))
4 plt.figure(figsize=(20,20))
5 plt.title('Covid-19 Confirmed Cases per Country')
6 plt.pie(visual_confirmed_cases, colors=c)
7 plt.legend(visual_unique_countries, loc='best')
8 plt.show()
```




```
In [38]: 1 X_train_confirmed, X_test_confirmed, y_train_confirmed, y_test_confirmed = train_test_split(days_since_1_22, world_cases, test
```

```
In [39]: 1 # Building the SVM Model
2 kernel = ['poly', 'sigmoid', 'rbf']
3 c = [0.01, 0.1, 1, 10]
4 gamma = [0.01, 0.1, 1]
5 epsilon = [0.01, 0.1, 1]
6 shrinking= [True, False]
7 svm_grid = {'kernel': kernel, 'C': c, 'gamma': gamma, 'epsilon': epsilon, 'shrinking': shrinking}
8 svm = SVR(kernel='poly')
9 svm_search = RandomizedSearchCV(svm, svm_grid, scoring='neg_mean_squared_error', cv=3, return_train_score=True, n_jobs=-1, n_i
10 svm_search.fit(X_train_confirmed, y_train_confirmed)
```

Fitting 3 folds for each of 40 candidates, totalling 120 fits

[Parallel(n_jobs=-1)]: Using backend LokyBackend with 8 concurrent workers.

[Parallel(n_jobs=-1)]: Done 34 tasks | elapsed: 7.7s

[Parallel(n_jobs=-1)]: Done 105 out of 120 | elapsed: 25.0s remaining: 3.5s

[Parallel(n_jobs=-1)]: Done 120 out of 120 | elapsed: 9.6min finished

K:\conda\lib\site-packages\sklearn\model_selection_search.py:813: DeprecationWarning: The default of the `iid` parameter will change from True to False in version 0.22 and will be removed in 0.24. This will change numeric results when test-set sizes are unequal.

DeprecationWarning)

K:\conda\lib\site-packages\sklearn\utils\validation.py:724: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().

y = column_or_1d(y, warn=True)

```
Out[39]: RandomizedSearchCV(cv=3, error_score='raise-deprecating',
    estimator=SVR(C=1.0, cache_size=200, coef0=0.0, degree=3,
    epsilon=0.1, gamma='auto_deprecated',
    kernel='poly', max_iter=-1, shrinking=True,
    tol=0.001, verbose=False),
    iid='warn', n_iter=40, n_jobs=-1,
    param_distributions={'C': [0.01, 0.1, 1, 10],
    'epsilon': [0.01, 0.1, 1],
    'gamma': [0.01, 0.1, 1],
    'kernel': ['poly', 'sigmoid', 'rbf'],
    'shrinking': [True, False]},
    pre_dispatch='2*n_jobs', random_state=None, refit=True,
    return_train_score=True, scoring='neg_mean_squared_error',
    verbose=1)
```

```
In [40]: 1 # Finding the best parameters for the model
2 svm_search.best_params_
```

```
Out[40]: {'shrinking': True, 'kernel': 'poly', 'gamma': 1, 'epsilon': 0.01, 'C': 0.1}
```

```
In [41]: 1 # Finding the best estimator and predict the future forecast
2 svm_confirmed = svm_search.best_estimator_
3 svm_pred = svm_confirmed.predict(future_forecast)
```

```
In [42]: 1 # The values of best estimator
2 svm_confirmed
```

```
Out[42]: SVR(C=0.1, cache_size=200, coef0=0.0, degree=3, epsilon=0.01, gamma=1,
    kernel='poly', max_iter=-1, shrinking=True, tol=0.001, verbose=False)
```

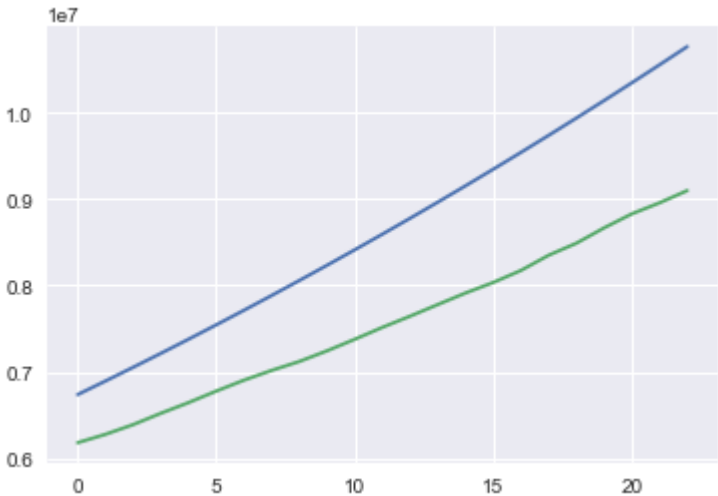
In [43]:

```
1 # Predicted values for all the dates
2 svm_pred
```

Out[43]:

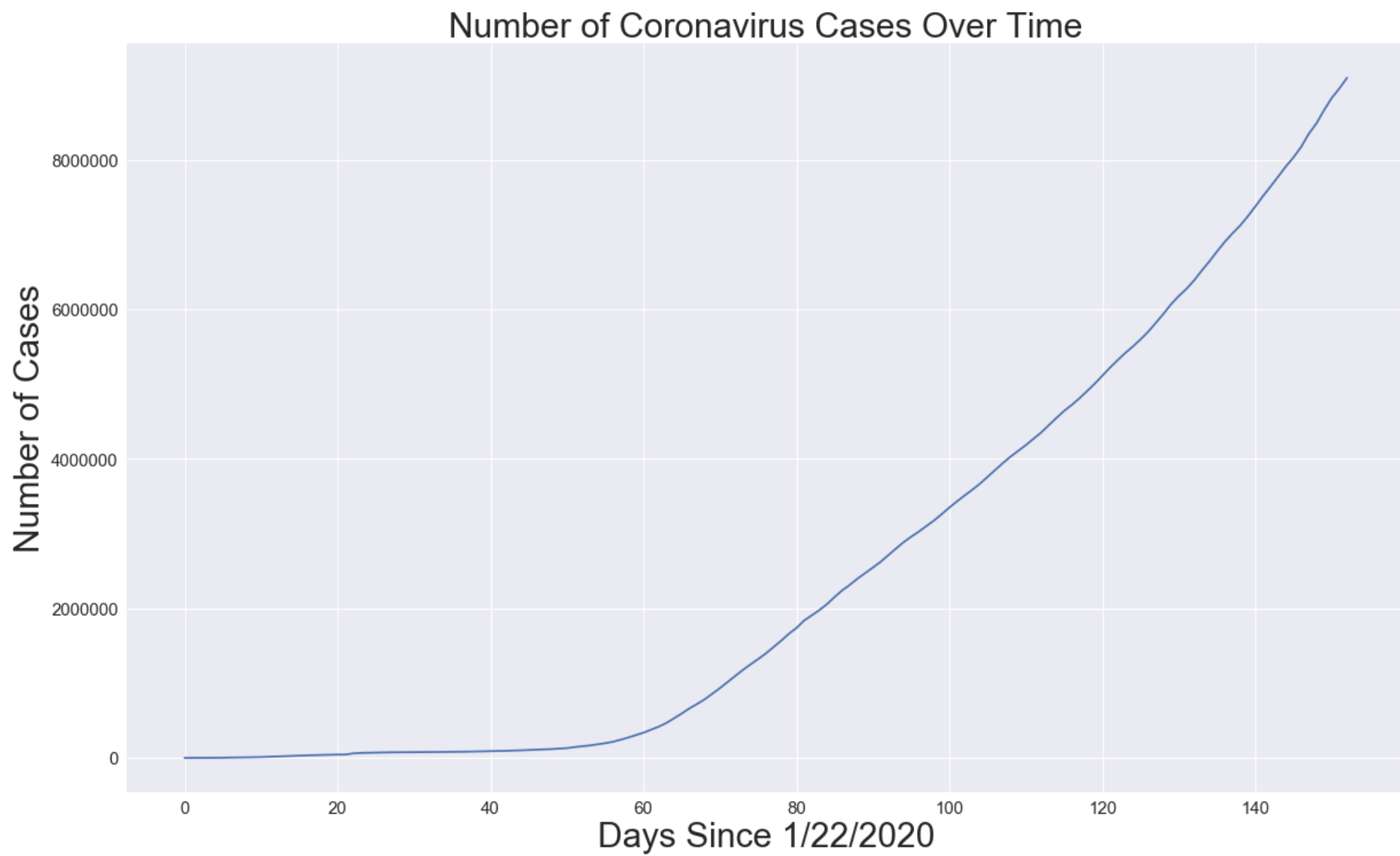
In [44]:

MAE: 1090973.2364436
MSE: 1300741123700.0754



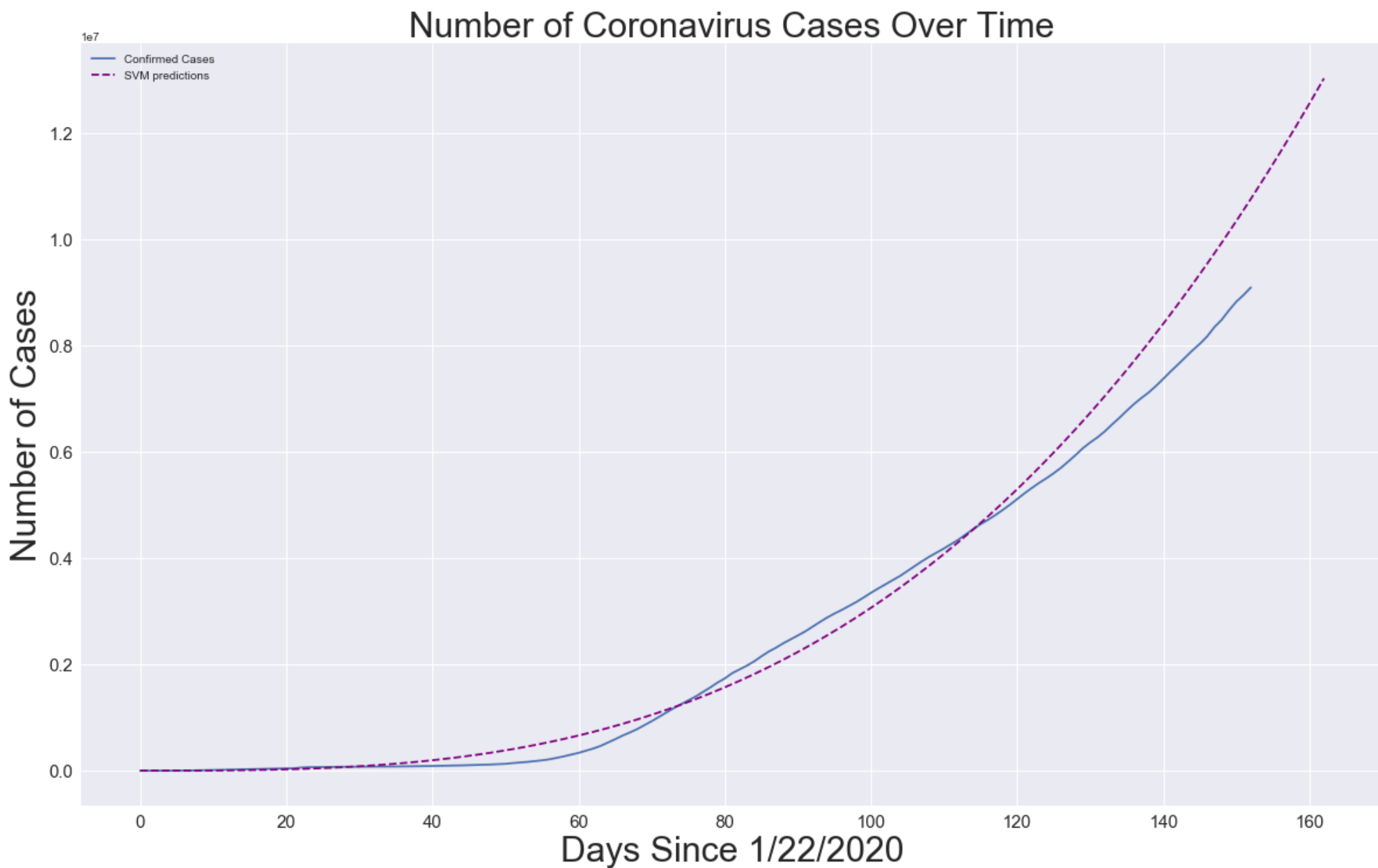
In [45]:

```
1 # Total number of coronavirus cases over time
2 plt.figure(figsize=(20, 12))
3 plt.plot(adjusted_dates, world_cases)
4 plt.title('Number of Coronavirus Cases Over Time', size=30)
5 plt.xlabel('Days Since 1/22/2020', size=30)
6 plt.ylabel('Number of Cases', size=30)
7 plt.xticks(size=15)
8 plt.yticks(size=15)
9 plt.show()
```



In [46]:

```
1 # Confirmed vs Predicted Cases
2 plt.figure(figsize=(20, 12))
3 plt.plot(adjusted_dates, world_cases)
4 plt.plot(future_forecast, svm_pred, linestyle='dashed', color='purple')
5 plt.title('Number of Coronavirus Cases Over Time', size=30)
6 plt.xlabel('Days Since 1/22/2020', size=30)
7 plt.ylabel('Number of Cases', size=30)
8 plt.legend(['Confirmed Cases', 'SVM predictions'])
9 plt.xticks(size=15)
10 plt.yticks(size=15)
11 plt.show()
```



In [47]:

```
1 # Prediction for the next 10 days using SVM
2 print('SVM future predictions:')
3 set(zip(future_forecast_dates[-10:], svm_pred[-10:]))
```

SVM future predictions:

Out[47]:

```
{('06/23/2020', 10979931.628414027),
 ('06/24/2020', 11196596.682735316),
 ('06/25/2020', 11416093.921748988),
 ('06/26/2020', 11638441.734493129),
 ('06/27/2020', 11863658.510738246),
 ('06/28/2020', 12091762.639156215),
 ('06/29/2020', 12322772.516719691),
 ('06/30/2020', 12556706.531123988),
 ('07/01/2020', 12793583.072383754),
 ('07/02/2020', 13033420.534175746)}
```

In [48]:

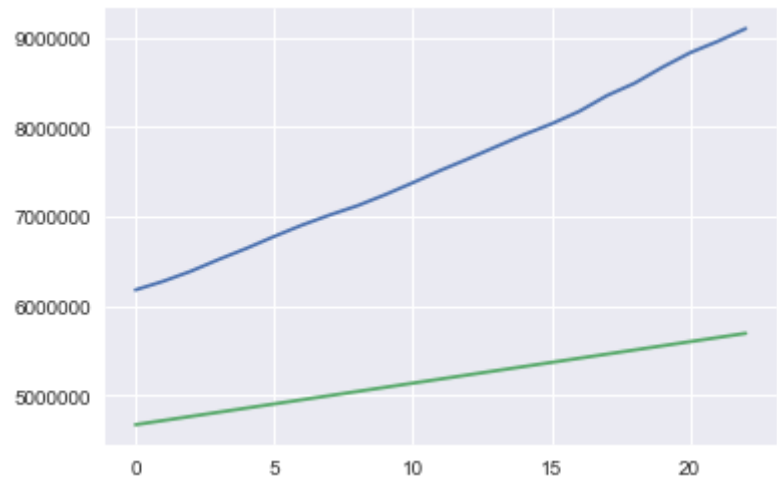
```
1 # Using Linear Regression Model to make Predictions
2 linear_model = LinearRegression(normalize = True, fit_intercept = True)
3 linear_model.fit(X_train_confirmed, y_train_confirmed)
4 test_linear_pred = linear_model.predict(X_test_confirmed)
5 linear_pred = linear_model.predict(future_forecast)
6 print('MAE:', mean_absolute_error(test_linear_pred, y_test_confirmed))
7 print('MSE:', mean_squared_error(test_linear_pred, y_test_confirmed))
```

MAE: 2376440.8069835715

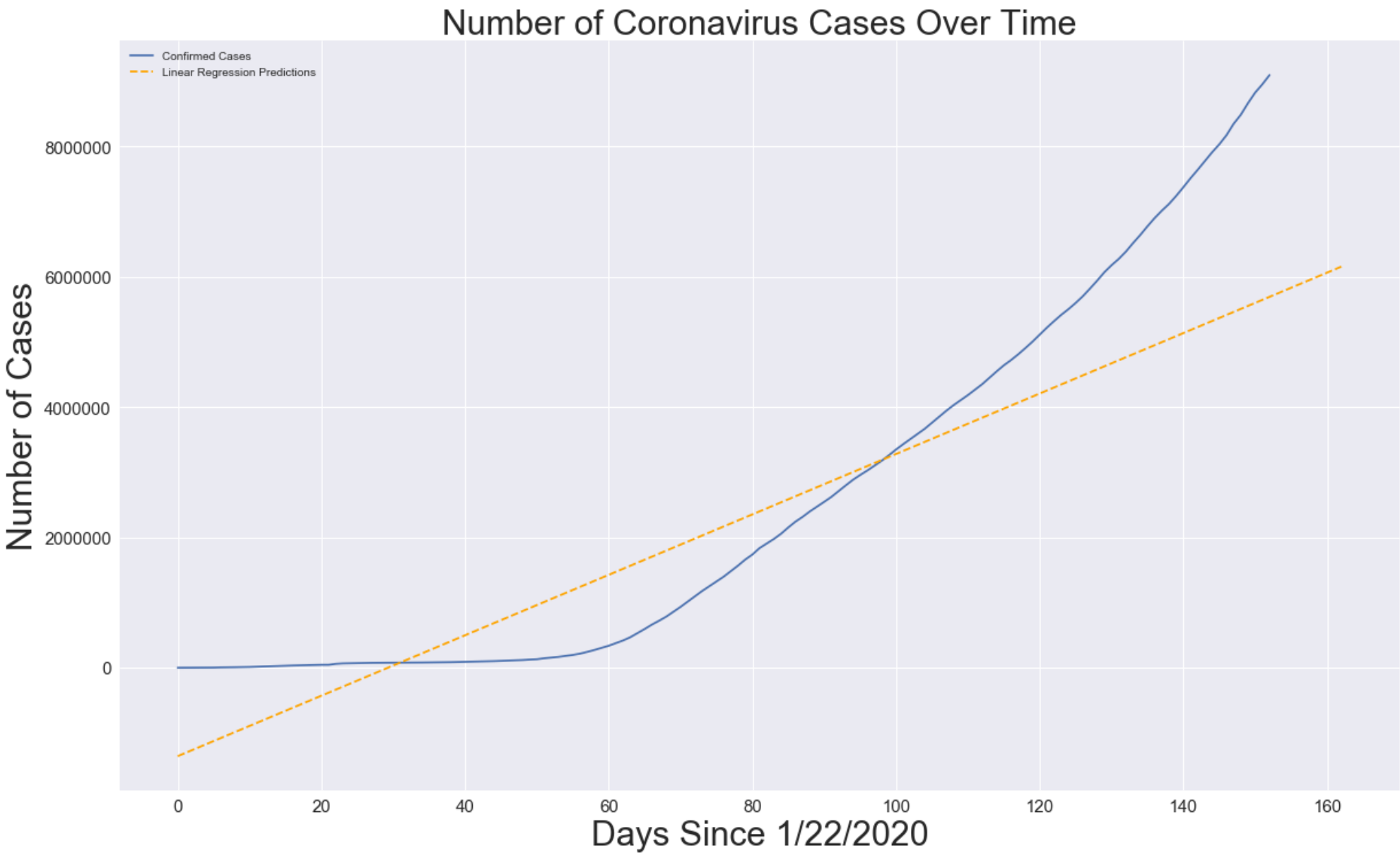
MSE: 5979711806506.45

```
In [49]: 1 plt.plot(y_test_confirmed)
2         plt.plot(test_linear_pred)
```

Out[49]: [<matplotlib.lines.Line2D at 0x817eb0>]



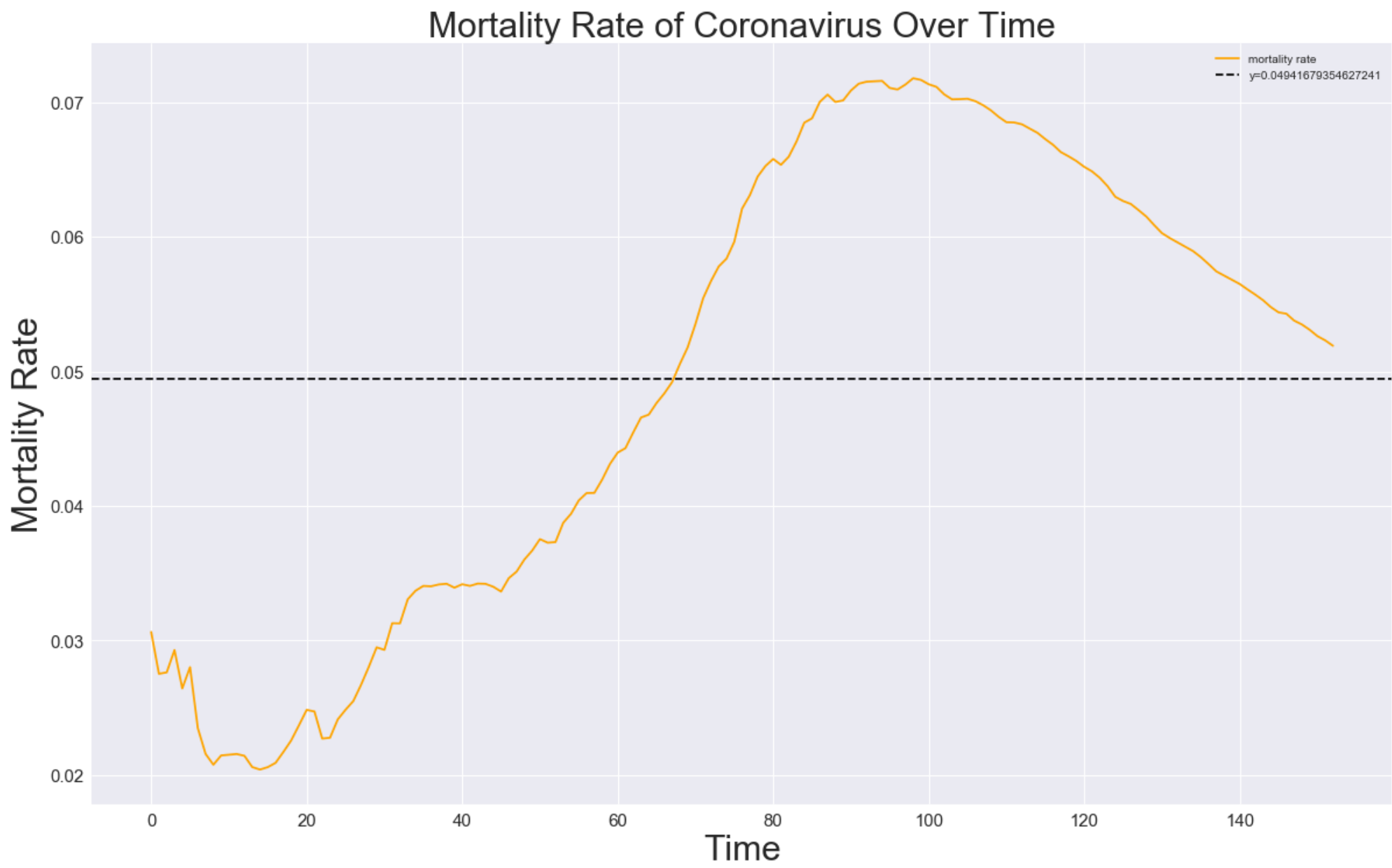
```
In [50]: 1 # Graphing the number of confirmed cases, deaths, active cases, and the mortality rate over time, as well as the number of rec
2         plt.figure(figsize=(20, 12))
3         plt.plot(adjusted_dates, world_cases)
4         plt.plot(future_forecast, linear_pred, linestyle='dashed', color='orange')
5         plt.title('Number of Coronavirus Cases Over Time', size=30)
6         plt.xlabel('Days Since 1/22/2020', size=30)
7         plt.ylabel('Number of Cases', size=30)
8         plt.legend(['Confirmed Cases', 'Linear Regression Predictions'])
9         plt.xticks(size=15)
10        plt.yticks(size=15)
11        plt.show()
```



```
In [51]: 1 # Prediction for the next 10 days using Linear Regression
2 print('Linear regression future predictions:')
3 print(future_forcast_dates[-10:],linear_pred[-10:])

Linear regression future predictions:
['06/23/2020', '06/24/2020', '06/25/2020', '06/26/2020', '06/27/2020', '06/28/2020', '06/29/2020', '06/30/2020', '07/01/2020', '07/02/2020']
[[5740468.66824073]
[5786857.62450942]
[5833246.58077811]
[5879635.5370468 ]
[5926024.49331549]
[5972413.44958418]
[6018802.40585287]
[6065191.36212156]
[6111580.31839026]
[6157969.27465895]]
```

```
In [52]: 1 mean_mortality_rate = np.mean(mortality_rate)
2 plt.figure(figsize=(20, 12))
3 plt.plot(adjusted_dates, mortality_rate, color='orange')
4 plt.axhline(y = mean_mortality_rate,linestyle='--', color='black')
5 plt.title('Mortality Rate of Coronavirus Over Time', size=30)
6 plt.legend(['mortality rate', 'y='+str(mean_mortality_rate)])
7 plt.xlabel('Time', size=30)
8 plt.ylabel('Mortality Rate', size=30)
9 plt.xticks(size=15)
10 plt.yticks(size=15)
11 plt.show()
```



In [53]:

1

Number of Coronavirus cases recovered vs the number of deaths

2

plt.figure(figsize=(20, 12))

3

plt.plot(adjusted_dates, total_deaths, color='r')

4

plt.plot(adjusted_dates, total_recovered, color='green')

5

plt.legend(['death', 'recoveries'], loc='best', fontsize=20)

6

plt.title('Number of Coronavirus Cases', size=30)

7

plt.xlabel('Time', size=30)

8

plt.ylabel('Number of Cases', size=30)

9

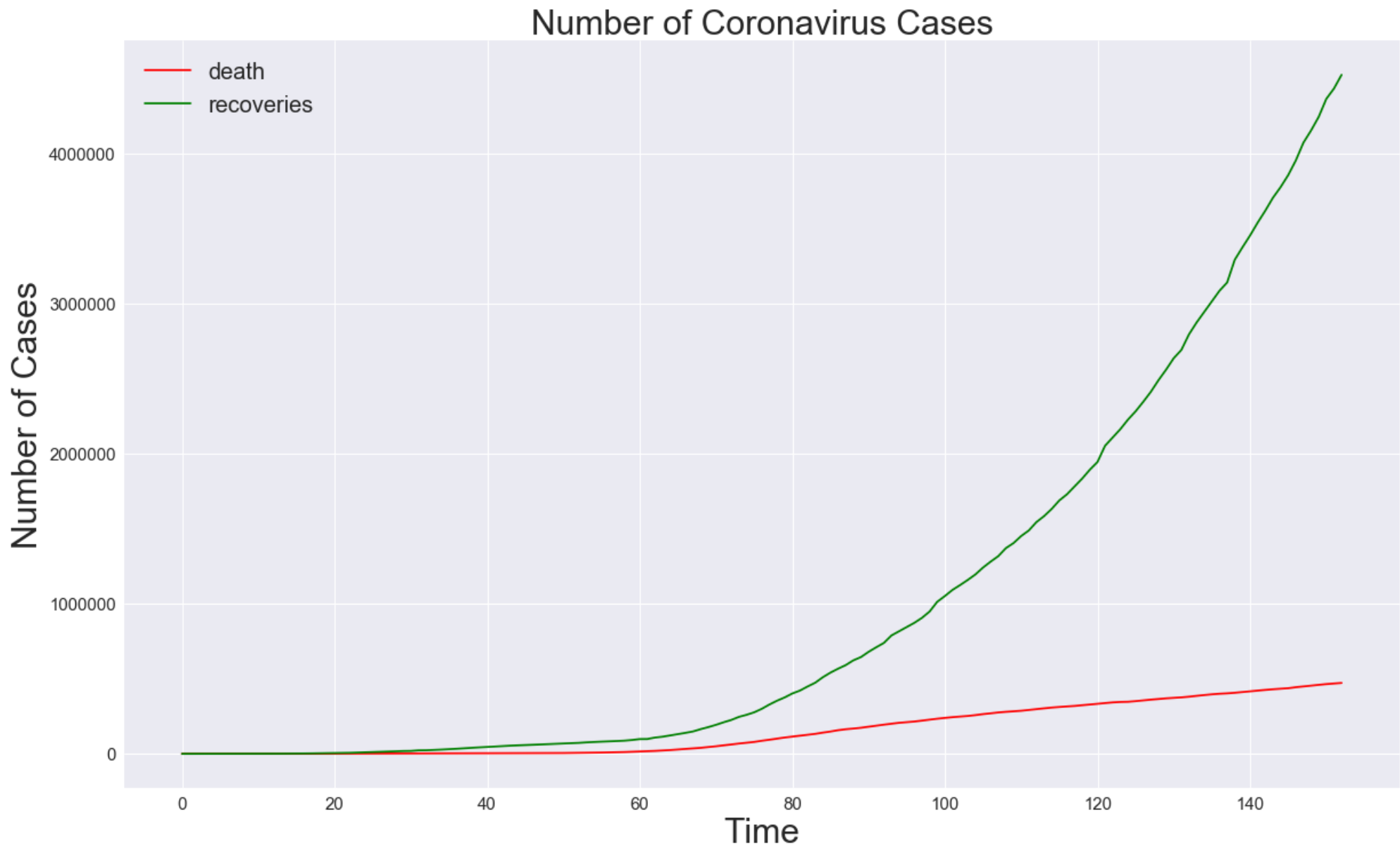
plt.xticks(size=15)

10

plt.yticks(size=15)

11

plt.show()



In []:

1