Mike Rocchio

MSDS 422

Assignment 1

Data preparation, exploration, visualization – This assignment was quite easy, and I was actually able to utilize some code I had written at work to describe telematics data with a few tweaks.

Review research design and modeling methods – This is a very standard statistical dive and was very easy to code. I wanted to just do a surface level analysis and show a few standard statistical measures.

Review results, evaluate models – I believe that this method is simple and shows promise but due to time series differences in the data caused by government & population responses further and more advanced statistical methods would need to be employed to develop a reliable model for this data. It would also need to be deployed in the short term at this point which can prove difficult.

Implementation and programming – This was very easy to program and I had little difficulty, Pandas, Numpy, and Matplotlib were made to chew through this stuff and I love to code through datasets in Python.

Exposition, problem description and management recommendations – My recommendation is that this was a good surface level analysis of the Covid data but that further analysis would need to be done to accurately predict future infection rates. As you can see my model assumed

linear growth despite nonlinear data and will therefor be inaccurate in the short term as this pandemic is ending and long-term analysis will not be needed.

In [94]:
```
test=df_agg.sort_values(by='index', ascending=False)
test
```

Out[94]:

	index	year	month	day	cases	deaths	popData2019	death_rate
349	349	2020	12	14	540659	7085	7.623307e+09	0.013104
348	348	2020	12	13	626421	10254	7.670244e+09	0.016369
347	347	2020	12	12	657140	12355	7.670244e+09	0.018801
346	346	2020	12	11	693352	12327	7.670244e+09	0.017779
345	345	2020	12	10	676114	12376	7.670244e+09	0.018305
...	...	...	...	...	...	...	...	...
4	4	2020	1	4	0	0	5.800630e+09	0.000000
3	3	2020	1	3	17	0	5.800630e+09	0.000000
2	2	2020	1	2	0	0	5.800630e+09	0.000000
1	1	2020	1	1	0	0	5.800630e+09	0.000000
0	0	2019	12	31	27	0	5.800630e+09	0.000000

350 rows × 8 columns

In [123]:

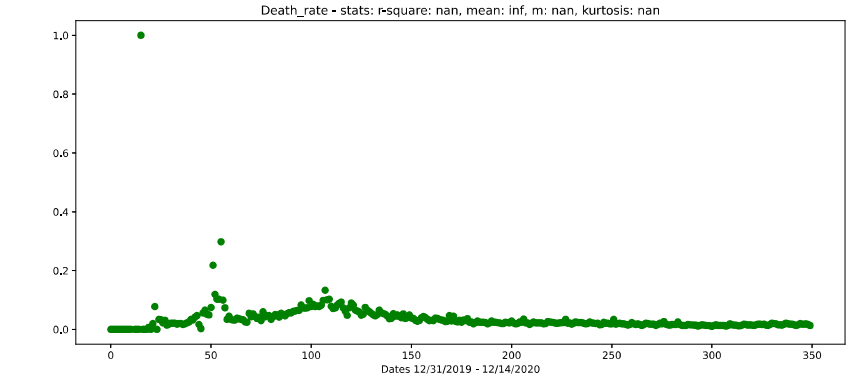
```
import pandas as pd
import numpy as np
import scipy
from scipy import stats
from scipy.stats import norm, kurtosis
import matplotlib.gridspec as gridspec
import matplotlib.pyplot as plt
import matplotlib.patheffects as pe


def r_square(x, y, degree):
    import numpy as np
    results = {}
    coefficents = np.polyfit(x, y, degree)
    polynomial = coefficents.tolist()
    p = np.poly1d(coefficents)
    yhat = p(x)
    ybar = np.sum(y)/len(y)
    r_square = round(np.sum((yhat-ybar)**2) / np.sum((y - ybar)**2),4)
    return r_square

def labels1(i):
  slope, dummy = np.polyfit(cov_x, np.array(df_agg[i]), 1)
  name=i.replace('_100','').capitalize()
  kurt = round(kurtosis(np.array(df_agg[i])),4)
  i=('{} - stats: r-square: {}, mean: {}, m: {}, kurtosis: {}'.format(name, r_square(cov_x, np.a
rray(df_agg[i]), 1), round(np.mean(np.array(df_agg[i])),4), round(slope,4), kurt))
  return i




df=pd.read_csv("C:/users/rocchm1/Downloads/data.csv")
df['cases']=abs(df['cases'])
df['deaths']=abs(df['deaths'])
numcol=['cases', 'deaths', 'popData2019', 'Cumulative_number_for_14_days_of_COVID-19_cases_per_1
00000']
for i in numcol:
    df[i]=pd.to_numeric(df[i]).fillna(0)
```

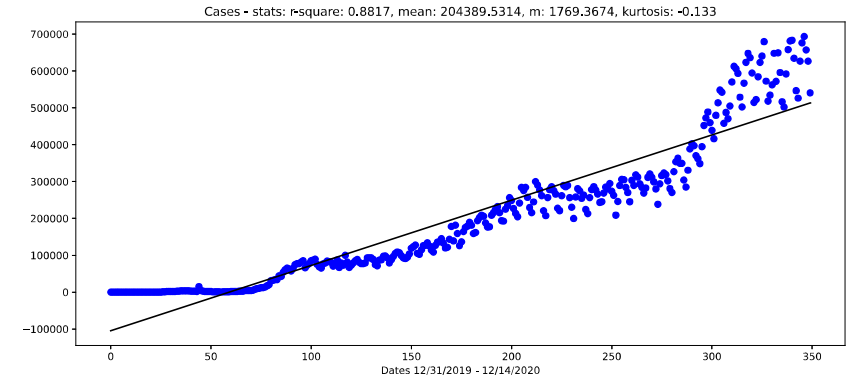
```
# df['Cumulative_number_for_14_days_of_COVID-19_cases_per_100000']=pd.to_numeric(df['Cumulative_
number_for_14_days_of_COVID-19_cases_per_100000']).fillna(0)
df['popData2019']=pd.to_numeric(df['popData2019']).fillna(0)
df=df.sort_values(by=['year', 'month', 'day']).reset_index(drop=True)
df_agg=df.groupby(['year', 'month', 'day'], as_index=False)[['cases', 'deaths', 'popData2019']].
sum().fillna(0)
df_agg=df_agg.reset_index()
df_agg['death_rate']=(df_agg['deaths']/df_agg['cases']).fillna(0)
# df_scplot=pd.melt(df_agg, id_vars=['index'], value_vars=['cases', 'deaths'])
# df_scplot['varibale']=np.where(df_scplot['variable']=='cases', 'blue', 'red')

cov_x=np.array(df_agg['index'])
cov_cases=np.array(df_agg['cases'])
cov_deaths=np.array(df_agg['deaths'])
death_rate=np.array(df_agg['death_rate'])

case_m, case_b=np.polyfit(cov_x, cov_cases, 1)
death_m, death_b=np.polyfit(cov_x, cov_deaths, 1)
deathr_m, deathr_b=np.polyfit(cov_x, death_rate, 1)

gs = gridspec.GridSpec(3, 2)
plt.figure(figsize=(30,20))

ax3 = plt.subplot(gs[0, 0])
ax3.scatter(df_agg['index'], df_agg['death_rate'], color='green')
ax3.plot(cov_x, cov_x*deathr_m + deathr_b, color='black')
plt.title("{}".format(labels1(i='death_rate')))
plt.xlabel('Dates 12/31/2019 - 12/14/2020')

ax0 = plt.subplot(gs[1, 0])
ax0.scatter(df_agg['index'], df_agg['cases'], color='b')
ax0.plot(cov_x, cov_x*case_m + case_b, color='black')
plt.title("{}".format(labels1(i='cases')))
plt.xlabel('Dates 12/31/2019 - 12/14/2020')

ax1 = plt.subplot(gs[1, 1])
ax1.scatter(df_agg['index'], df_agg['deaths'], color='r')
ax1.plot(cov_x, cov_x*death_m + death_b, color='black')
plt.title("{}".format(labels1(i='deaths')))
```

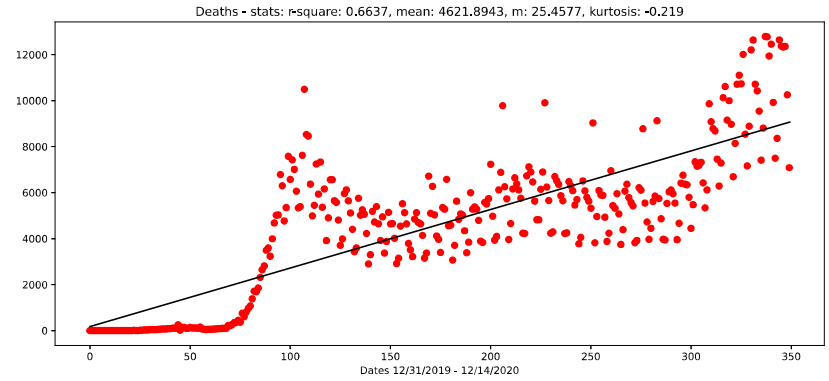
```
plt.xlabel('Dates 12/31/2019 - 12/14/2020')

ax00 = plt.subplot(gs[2, 0])
res1 = stats.probplot(cov_cases, plot=ax00, dist="norm")
ax00.get_lines()[0].set_markerfacecolor('b')
ax00.get_lines()[0].set_markeredgecolor('black')
plt.title('Cases QQ Norm')

ax01 = plt.subplot(gs[2, 1])
res2 = stats.probplot(cov_deaths, plot=ax01, dist="norm")
ax01.get_lines()[0].set_markerfacecolor('r')
ax01.get_lines()[0].set_markeredgecolor('black')
plt.title('Deaths QQ Norm')

plt.plot()
```

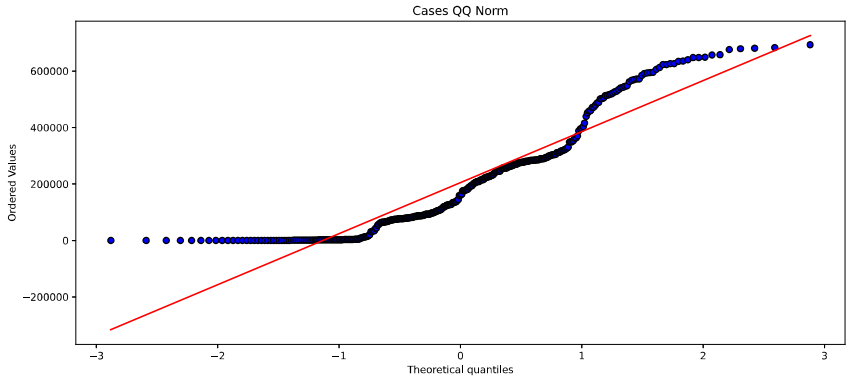
```
C:\Users\rocchm1\AppData\Roaming\Python\Python37\site-packages\scipy\stats\stats.py:1082: Runtim
eWarning: invalid value encountered in subtract
  a_zero_mean = a - np.expand_dims(np.mean(a, axis), axis)
ipykernel_launcher:19: RuntimeWarning: invalid value encountered in subtract
```

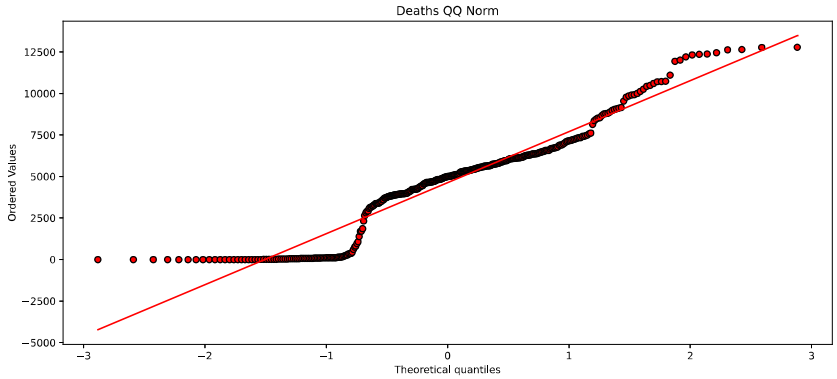
Out[123]: []











In [116]:

```
import pandas as pd
from scipy import stats

###MY HYPOTHESIS IS THAT AS US CASES ROSE SO DID MEXICOS

df['death_rate']=(df['deaths']/df['cases']).fillna(0)
united_states=df[df['countriesAndTerritories']=='United_States_of_America'].reset_index(drop=Tru
e)
mexico=df[df['countriesAndTerritories']=='Mexico'].reset_index(drop=True)
united_states=united_states.reset_index()
mexico=mexico.reset_index()

del united_states['countriesAndTerritories']
del united_states['geoId']
del united_states['countryterritoryCode']
del mexico['countriesAndTerritories']
del mexico['geoId']
del mexico['countryterritoryCode']
changevar=['cases',     'deaths',       'popData2019', 'death_rate',    'continentExp', 'Cumulat
ive_number_for_14_days_of_COVID-19_cases_per_100000']
for i in changevar:
    united_states['us_{}'.format(i)]=united_states[i]
    del united_states[i]
for i in changevar:
    mexico['mex_{}'.format(i)]=mexico[i]
    del mexico[i]

final=pd.merge(united_states, mexico, how='inner')
final

final[['us_death_rate','mex_death_rate']].describe()


ttest,pval = stats.ttest_rel(final['us_death_rate'], final['mex_death_rate'])
print(pval)
if pval<0.05:
    print("reject null hypothesis, there is not a statistical correlation between us and mexican
covid cases")
else:
```

```
    print("accept null hypothesis, there is a statistical correlation between us and mexican cov
id cases")
```

```
3.4084816470721526e-20
reject null hypothesis, there is not a statistical correlation between us and mexican covid case
s
```

In [119]:
```
df
```

Out[119]:

	dateRep	day	month	year	cases	deaths	countriesAndTerritories	geoId	countryterritoryCode	popData2019
0	31/12/2019	31	12	2019	0	0	Afghanistan	AF	AFG	38041757.0
1	31/12/2019	31	12	2019	0	0	Algeria	DZ	DZA	43053054.0
2	31/12/2019	31	12	2019	0	0	Armenia	AM	ARM	2957728.0
3	31/12/2019	31	12	2019	0	0	Australia	AU	AUS	25203200.0
4	31/12/2019	31	12	2019	0	0	Austria	AT	AUT	8858775.0
...	...	...	...	...	...	...	...	...	...	...
61895	14/12/2020	14	12	2020	0	0	Wallis_and_Futuna	WF	NaN	0.0
61896	14/12/2020	14	12	2020	0	0	Western_Sahara	EH	ESH	582458.0
61897	14/12/2020	14	12	2020	0	0	Yemen	YE	YEM	29161922.0
61898	14/12/2020	14	12	2020	57	1	Zambia	ZM	ZMB	17861034.0
61899	14/12/2020	14	12	2020	27	0	Zimbabwe	ZW	ZWE	14645473.0

61900 rows × 13 columns

In [122]:
```
print('correlation coefficent for cases vs deaths:')
np.corrcoef(df['deaths'], df['cases'])[0,1]
```

```
correlation coefficent for cases vs deaths:
```

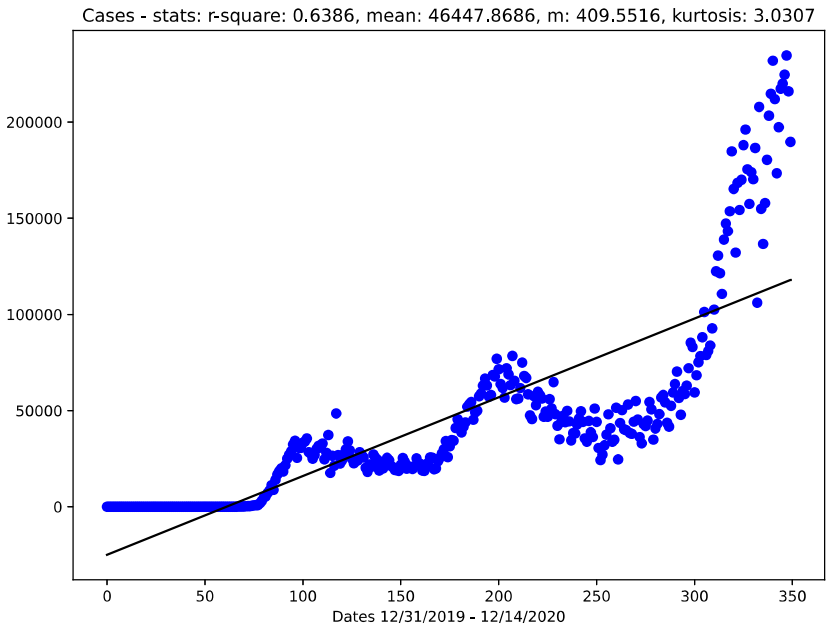
Out[122]:
```
0.744691444841185
```

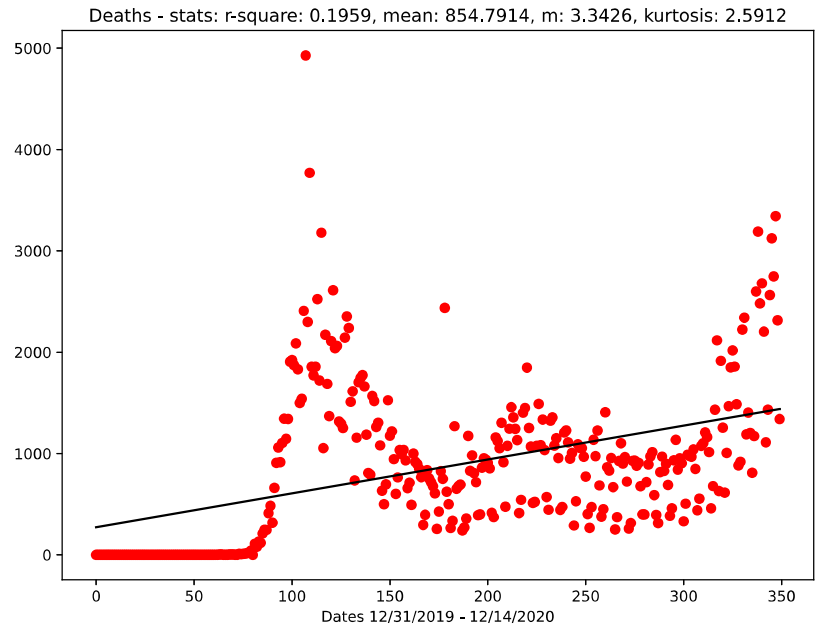
In [147]:

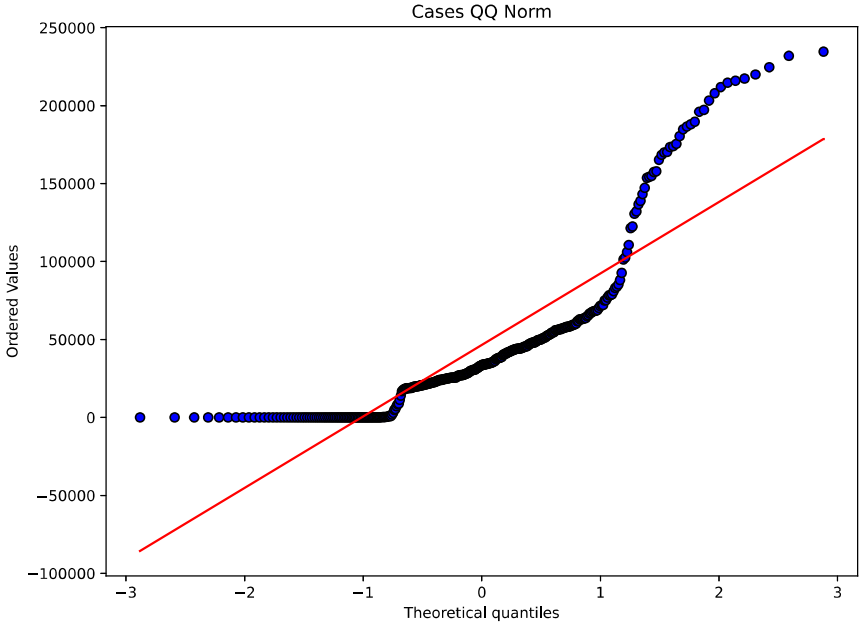
```
import pandas as pd
import numpy as np
import scipy
from scipy import stats
from scipy.stats import norm, kurtosis
import matplotlib.gridspec as gridspec
import matplotlib.pyplot as plt
import matplotlib.patheffects as pe


def r_square(x, y, degree):
    import numpy as np
    results = {}
    coefficents = np.polyfit(x, y, degree)
    polynomial = coefficents.tolist()
    p = np.poly1d(coefficents)
    yhat = p(x)
    ybar = np.sum(y)/len(y)
    r_square = round(np.sum((yhat-ybar)**2) / np.sum((y - ybar)**2),4)
    return r_square

def labels1(i):
  slope, dummy = np.polyfit(cov_x, np.array(united_states1[i]), 1)
  name=i.replace('_100','').capitalize()
  kurt = round(kurtosis(np.array(united_states1[i])),4)
  i=('{} - stats: r-square: {}, mean: {}, m: {}, kurtosis: {}'.format(name, r_square(cov_x, np.a
rray(united_states1[i]), 1), round(np.mean(np.array(united_states1[i])),4), round(slope,4), kurt
))
  return i

united_states1=df[df['countriesAndTerritories']=='United_States_of_America'].reset_index(drop=Tr
ue)
united_states1=united_states1.reset_index()


us_x=np.array(united_states1['index'])
us_cases=np.array(united_states1['cases'])
us_deaths=np.array(united_states1['deaths'])
```

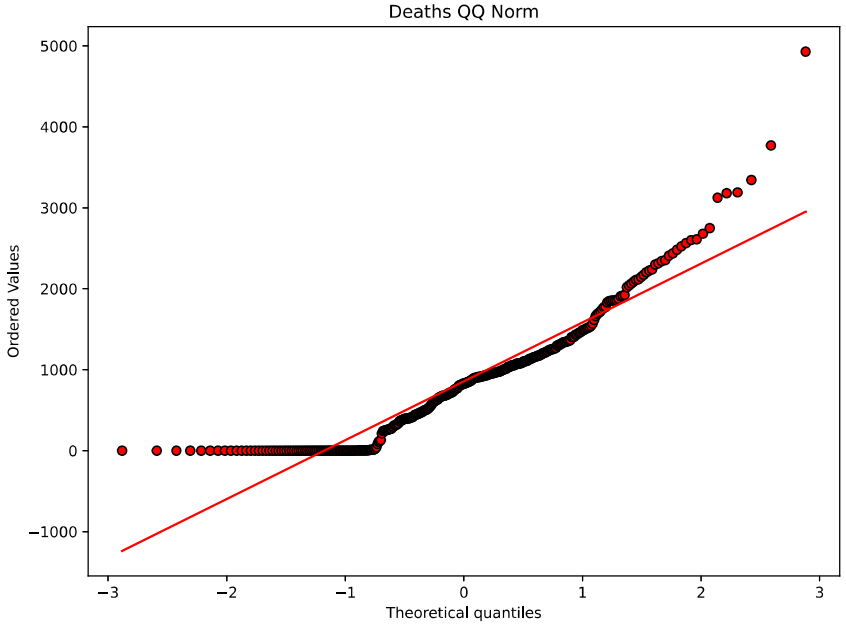
```
usc_m, usc_b=np.polyfit(us_x, us_cases, 1)
usd_m, usd_b=np.polyfit(us_x, us_deaths, 1)

gs = gridspec.GridSpec(2, 2)
plt.figure(figsize=(20,15))

us0 = plt.subplot(gs[0, 0])
us0.scatter(united_states1['index'], united_states1['cases'], color='b')
us0.plot(us_x, usc_m*us_x + usc_b, color='black')
plt.title("{}".format(labels1(i='cases')))
plt.xlabel('Dates 12/31/2019 - 12/14/2020')

us1 = plt.subplot(gs[0, 1])
us1.scatter(united_states1['index'], united_states1['deaths'], color='r')
us1.plot(us_x, usd_m*us_x + usd_b, color='black')
plt.title("{}".format(labels1(i='deaths')))
plt.xlabel('Dates 12/31/2019 - 12/14/2020')

us00 = plt.subplot(gs[1, 0])
resus1 = stats.probplot(us_cases, plot=us00, dist="norm")
us00.get_lines()[0].set_markerfacecolor('b')
us00.get_lines()[0].set_markeredgecolor('black')
plt.title('Cases QQ Norm')

us01 = plt.subplot(gs[1, 1])
resus2 = stats.probplot(us_deaths, plot=us01, dist="norm")
us01.get_lines()[0].set_markerfacecolor('r')
us01.get_lines()[0].set_markeredgecolor('black')
plt.title('Deaths QQ Norm')
```

-25018.877606837596









In [152]:
```
print("Model predicion of next 5 days after 12/14/2020")
print("Prediction for 12/15/2020 Cases: {} Deaths: {}".format(usc_m*350 + usc_b, usd_m*350 + usd
_b))
print("Prediction for 12/15/2020 Cases: {} Deaths: {}".format(usc_m*351 + usc_b, usd_m*351 + usd
_b))
print("Prediction for 12/15/2020 Cases: {} Deaths: {}".format(usc_m*352 + usc_b, usd_m*352 + usd
_b))
print("Prediction for 12/15/2020 Cases: {} Deaths: {}".format(usc_m*353 + usc_b, usd_m*353 + usd
_b))
print("Prediction for 12/15/2020 Cases: {} Deaths: {}".format(usc_m*354 + usc_b, usd_m*354 + usd
_b))
```

```
Model predicion of next 5 days after 12/14/2020
Prediction for 12/15/2020 Cases: 118324.16630372498 Deaths: 1441.4116250511677
Prediction for 12/15/2020 Cases: 118733.71785775517 Deaths: 1444.7541902732744
Prediction for 12/15/2020 Cases: 119143.26941178532 Deaths: 1448.096755495381
Prediction for 12/15/2020 Cases: 119552.82096581551 Deaths: 1451.4393207174878
Prediction for 12/15/2020 Cases: 119962.3725198457 Deaths: 1454.7818859395948
```