Regression for Exponential Growth - Applied to the Corona Virus

See https://towardsdatascience.com/modeling-exponential-growth-49a2b6f22e1f (https://towardsdatascience.com/modeling-exponential-growth-49a2b6f22e1f)

In [2]:
```
import statsmodels.api as sm
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [3]:
```
confirmed_df = pd.read_csv('https://raw.githubusercontent.com/CSSEGIS
andData/COVID-19/master/csse_covid_19_data/csse_covid_19_time_series/
time_series_covid19_confirmed_global.csv')
```

In [4]:
```
confirmed_df
```

Out[4]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/
0	NaN	Afghanistan	33.0000	65.0000	0	0	0	0	
1	NaN	Albania	41.1533	20.1683	0	0	0	0	
2	NaN	Algeria	28.0339	1.6596	0	0	0	0	
3	NaN	Andorra	42.5063	1.5218	0	0	0	0	
4	NaN	Angola	-11.2027	17.8739	0	0	0	0	
...	...	...	...	...	...	...	...	...	
249	Anguilla	United Kingdom	18.2206	-63.0686	0	0	0	0	
250	British Virgin Islands	United Kingdom	18.4207	-64.6400	0	0	0	0	
251	Turks and Caicos Islands	United Kingdom	21.6940	-71.7979	0	0	0	0	
252	NaN	MS Zaandam	0.0000	0.0000	0	0	0	0	
253	NaN	Botswana	-22.3285	24.6849	0	0	0	0	

254 rows × 73 columns

In [5]:
```
# US data only
us_data = confirmed_df.loc[confirmed_df['Country/Region'] == 'US']
us_data
```

Out[5]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/
225	NaN	US	37.0902	-95.7129	1	1	2	2	

1 rows × 73 columns

In [7]:
```
us_time_series_only = us_data.drop(["Province/State", "Country/Region", "Lat", "Long"], axis=1)
us_time_series_only
```

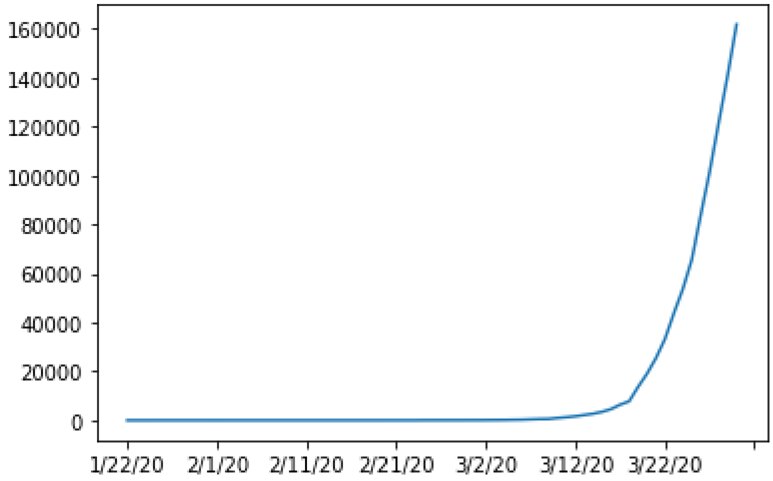
Out[7]:

	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	...
225	1	1	2	2	5	5	5	5	5	7	...

1 rows × 69 columns

In [9]:
```
us_time_series_only.transpose().plot(legend=False)
```

Out[9]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd8dda0c898>
```



In [11]:
```
transposed = us_time_series_only.transpose()
transposed
```

Out[11]:

	225
1/22/20	1
1/23/20	1
1/24/20	2
1/25/20	2
1/26/20	5
...	...
3/26/20	83836
3/27/20	101657
3/28/20	121478
3/29/20	140886
3/30/20	161807

69 rows × 1 columns

In [17]:

```
transposed.index
```

Out[17]:

```
Index(['1/22/20', '1/23/20', '1/24/20', '1/25/20', '1/26/20', '1/27/2
0',
       '1/28/20', '1/29/20', '1/30/20', '1/31/20', '2/1/20', '2/2/2
0',
       '2/3/20', '2/4/20', '2/5/20', '2/6/20', '2/7/20', '2/8/20',
'2/9/20',
       '2/10/20', '2/11/20', '2/12/20', '2/13/20', '2/14/20', '2/15/2
0',
       '2/16/20', '2/17/20', '2/18/20', '2/19/20', '2/20/20', '2/21/2
0',
       '2/22/20', '2/23/20', '2/24/20', '2/25/20', '2/26/20', '2/27/2
0',
       '2/28/20', '2/29/20', '3/1/20', '3/2/20', '3/3/20', '3/4/20',
'3/5/20',
       '3/6/20', '3/7/20', '3/8/20', '3/9/20', '3/10/20', '3/11/20',
'3/12/20',
       '3/13/20', '3/14/20', '3/15/20', '3/16/20', '3/17/20', '3/18/2
0',
       '3/19/20', '3/20/20', '3/21/20', '3/22/20', '3/23/20', '3/24/2
0',
       '3/25/20', '3/26/20', '3/27/20', '3/28/20', '3/29/20', '3/30/2
0'],
      dtype='object')
```

In [32]:

```
transposed = transposed.rename(columns={225:"Confirmed"})
transposed
```

Out[32]:

	Confirmed
1/22/20	1
1/23/20	1
1/24/20	2
1/25/20	2
1/26/20	5
...	...
3/26/20	83836
3/27/20	101657
3/28/20	121478
3/29/20	140886
3/30/20	161807

69 rows × 1 columns

In [33]:

```
transposed.columns
```

Out[33]:

```
Index(['Confirmed'], dtype='object')
```

In [46]:

```
for index in transposed.index:
    print(index, ": ", transposed.Confirmed.get(index))
```

```
1/22/20 :  1
1/23/20 :  1
1/24/20 :  2
1/25/20 :  2
1/26/20 :  5
1/27/20 :  5
1/28/20 :  5
1/29/20 :  5
1/30/20 :  5
1/31/20 :  7
2/1/20 :  8
2/2/20 :  8
2/3/20 :  11
2/4/20 :  11
2/5/20 :  11
2/6/20 :  11
2/7/20 :  11
2/8/20 :  11
2/9/20 :  11
2/10/20 :  11
2/11/20 :  12
2/12/20 :  12
2/13/20 :  13
2/14/20 :  13
2/15/20 :  13
2/16/20 :  13
2/17/20 :  13
2/18/20 :  13
2/19/20 :  13
2/20/20 :  13
2/21/20 :  15
2/22/20 :  15
2/23/20 :  15
2/24/20 :  51
2/25/20 :  51
2/26/20 :  57
2/27/20 :  58
2/28/20 :  60
2/29/20 :  68
3/1/20 :  74
3/2/20 :  98
3/3/20 :  118
3/4/20 :  149
3/5/20 :  217
3/6/20 :  262
3/7/20 :  402
3/8/20 :  518
3/9/20 :  583
3/10/20 :  959
3/11/20 :  1281
3/12/20 :  1663
3/13/20 :  2179
3/14/20 :  2727
3/15/20 :  3499
3/16/20 :  4632
3/17/20 :  6421
3/18/20 :  7783
```

```
3/19/20 :  13677
3/20/20 :  19100
3/21/20 :  25489
3/22/20 :  33276
3/23/20 :  43847
3/24/20 :  53740
3/25/20 :  65778
3/26/20 :  83836
3/27/20 :  101657
3/28/20 :  121478
3/29/20 :  140886
3/30/20 :  161807
```

In [48]:

```
transposed['logInfections'] = np.log(transposed.Confirmed)
transposed
```

Out[48]:

	Confirmed	logInfections
1/22/20	1	0.000000
1/23/20	1	0.000000
1/24/20	2	0.693147
1/25/20	2	0.693147
1/26/20	5	1.609438
...	...	...
3/26/20	83836	11.336618
3/27/20	101657	11.529360
3/28/20	121478	11.707488
3/29/20	140886	11.855706
3/30/20	161807	11.994160

69 rows × 2 columns

In [52]:

```
transposed['Day'] = range(len(transposed))
transposed
```

Out[52]:

	Confirmed	logInfections	Day
1/22/20	1	0.000000	0
1/23/20	1	0.000000	1
1/24/20	2	0.693147	2
1/25/20	2	0.693147	3
1/26/20	5	1.609438	4
...	...	...	...
3/26/20	83836	11.336618	64
3/27/20	101657	11.529360	65
3/28/20	121478	11.707488	66
3/29/20	140886	11.855706	67
3/30/20	161807	11.994160	68

69 rows × 3 columns

In [53]:

```
X = transposed.Day
X = sm.add_constant(X)
```

In [54]:

```
y = transposed.logInfections
```

In [55]:

```python
mod = sm.OLS(y, X)
res = mod.fit()
print(res.summary())
```

```
                            OLS Regression Results                            
=====================================================================
=========
Dep. Variable:          logInfections   R-squared:
0.908
Model:                            OLS   Adj. R-squared:
0.907
Method:                 Least Squares   F-statistic:
664.1
Date:                Mon, 30 Mar 2020   Prob (F-statistic):
1.73e-36
Time:                        20:57:08   Log-Likelihood:
-100.90
No. Observations:                  69   AIC:
205.8
Df Residuals:                      67   BIC:
210.3
Df Model:                           1
Covariance Type:            nonrobust
=====================================================================
=========
                 coef    std err          t      P>|t|      [0.025
0.975]
---------------------------------------------------------------------
---------
const         -0.6383      0.252     -2.529      0.014      -1.142
-0.134
Day            0.1651      0.006     25.770      0.000       0.152
0.178
=====================================================================
=========
Omnibus:                       42.389   Durbin-Watson:
0.044
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
5.614
Skew:                          -0.126   Prob(JB):
0.0604
Kurtosis:                       1.625   Cond. No.
78.0
=====================================================================
=========

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors i
s correctly specified.
```

Reading the table to make predictions

In [57]:

```
# log initial = 0.4480
# initial =
np.exp(-0.6383)
```

Out[57]:

```
0.5281895835334227
```

In [58]:

```
# log 1 + r = 0.1128
# real 1 + r =
np.exp(0.1651)
```

Out[58]:

```
1.1795110639204238
```

In [59]:

```
def linear_predictions(t):
    return np.exp(-0.6383) * np.exp(0.1651) ** t
```

In [61]:

```
transposed['Predictions'] = transposed.Day.apply(linear_predictions)
transposed.head(10)
```

Out[61]:

	Confirmed	logInfections	Day	Predictions
1/22/20	1	0.000000	0	0.528190
1/23/20	1	0.000000	1	0.623005
1/24/20	2	0.693147	2	0.734842
1/25/20	2	0.693147	3	0.866754
1/26/20	5	1.609438	4	1.022346
1/27/20	5	1.609438	5	1.205868
1/28/20	5	1.609438	6	1.422335
1/29/20	5	1.609438	7	1.677660
1/30/20	5	1.609438	8	1.978819
1/31/20	7	1.945910	9	2.334038

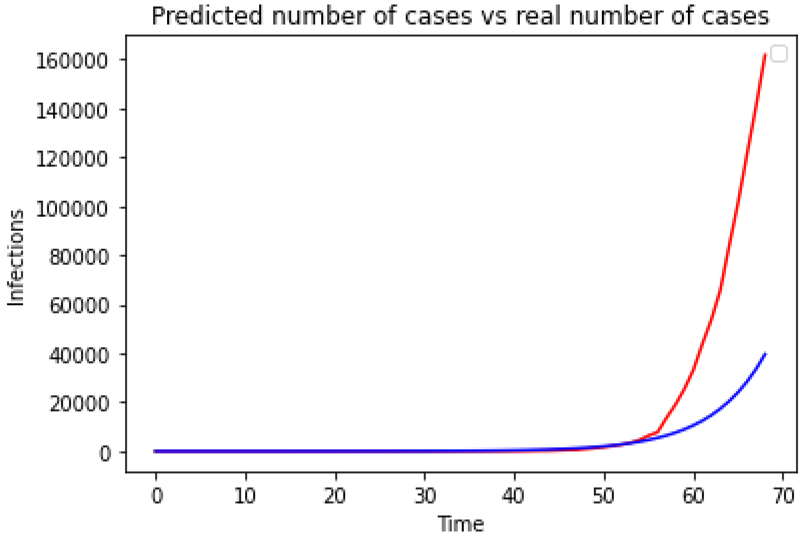
In [64]:

```
plt.plot(transposed.Day, transposed.Confirmed, 'red')
plt.plot(transposed.Day, transposed.Predictions, 'blue')
plt.title('Predicted number of cases vs real number of cases')
plt.xlabel('Time')
plt.ylabel('Infections')
plt.legend()
```

```
No handles with labels found to put in legend.
```

Out[64]:

```
<matplotlib.legend.Legend at 0x7fd8dc492198>
```



In [ ]:

```
```