This notebook follows the tutorial found here https://www.youtube.com/watch?v=sHWKN5dakPw

```
### Import Dependancies

%matplotlib inline
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import matplotlib.colors as mcolors

import random
import math
import time

from sklearn.model_selection import RandomizedSearchCV, train_test_split, GridSearchCV
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error, mean_absolute_error
import datetime
import operator
plt.style.use('seaborn')

from google.colab import files
from google.colab import drive
drive.mount('/Drive')

from scipy.fft import fft, ifft
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318989

Enter your authorization code:
··········
Mounted at /Drive
```

Import Data Set

```
whoURL = 'https://covid19.who.int/WHO-COVID-19-global-data.csv'
who_data = pd.read_csv(whoURL)
```

```
who_data.tail(5)
```

	Date_reported	Country_code	Country	WHO_region	New_cases	Cumulative_cases	New_d
28953	2020-07-13	ZW	Zimbabwe	AFRO	3	985	
28954	2020-07-14	ZW	Zimbabwe	AFRO	49	1034	
28955	2020-07-15	ZW	Zimbabwe	AFRO	30	1064	
28956	2020-07-16	ZW	Zimbabwe	AFRO	25	1089	
28957	2020-07-17	ZW	Zimbabwe	AFRO	273	1362	

```
# Drop rows that have missing values
```

```
print(who_data.shape)
who_data.dropna(inplace=True)
print(who_data.shape)
```

```
(28958, 8)
(28832, 8)
```

```
# reconfigure the data so we have just deaths
cumulative_deaths = who_data.filter(['Date_reported', ' Cumulative_deaths'], axis = 1)
cumulative_deaths = cumulative_deaths.groupby('Date_reported', as_index=False).sum()
cumulative_deaths.head(3)
```

	Date_reported	Cumulative_deaths
0	2020-01-11	1
1	2020-01-12	1
2	2020-01-13	1

```
# reconfigure the data so we have just cases
cumulative_cases = who_data.filter(['Date_reported', ' Cumulative_cases'], axis = 1)
cumulative_cases = cumulative_cases.groupby('Date_reported', as_index=False).sum()
cumulative_cases.tail(3)
```

	Date_reported	Cumulative_cases
187	2020-07-16	13377890
188	2020-07-17	13615561
189	2020-07-18	10272645

```
earliest_reported = who_data['Date_reported'].min()
earliest_reported
last_updated = who_data['Date_reported'].max()
last_updated
```

```
'2020-07-18'
```

```
dates = cumulative_cases['Date_reported']
dates
```

```
0      2020-01-11
1      2020-01-12
2      2020-01-13
3      2020-01-14
4      2020-01-15
          ...
185    2020-07-14
186    2020-07-15
187    2020-07-16
188    2020-07-17
189    2020-07-18
Name: Date_reported, Length: 190, dtype: object
```

```
world_cases = np.array(cumulative_cases[' Cumulative_cases'])
world_cases
```

```
array([      41,       41,       42,       43,       43,       43,
             48,       65,      125,      203,      296,      450,
            583,      854,     1323,     2021,     2806,     4595,
           6076,     7836,     9846,    11961,    14559,    17391,
          20649,    24563,    28284,    31486,    34899,    37568,
          40623,    43109,    45174,    60387,    64455,    67187,
          69277,    71438,    73431,    75287,    75773,    76817,
          77926,    78942,    79566,    80392,    81316,    82685,
          84133,    85959,    87850,    90131,    92463,    94663,
          97440,   101240,   105166,   108904,   112951,   117439,
         123964,   130483,   140126,   151636,   163489,   175280,
         190104,   208533,   232509,   265401,   288849,   298185,
         331907,   391864,   422395,   478118,   519722,   582513,
         644250,   701942,   760029,   833723,   907004,   982209,
        1062266,  1142094,  1219079,  1288420,  1362610,  1445881,
        1530937,  1622011,  1706056,  1782078,  1853451,  1927771,
        2003490,  2085767,  2170163,  2250526,  2319924,  2402593,
        2475093,  2547672,  2627928,  2716534,  2802456,  2889092,
        2963083,  3029372,  3102314,  3187182,  3277452,  3361190,
        3446871,  3530077,  3600988,  3685128,  3773146,  3867815,
        3956599,  4044548,  4123625,  4199559,  4282307,  4370816,
        4463415,  4558691,  4661732,  4736991,  4827345,  4929803,
        5020926,  5126957,  5230948,  5333441,  5422670,  5517058,
        5613624,  5724545,  5840981,  5956480,  6080535,  6194042,
        6287670,  6417145,  6536045,  6663921,  6800018,  6931315,
        7040539,  7146391,  7274820,  7411365,  7553632,  7690773,
        7823195,  7941966,  8062091,  8243348,  8386323,  8524956,
        8708405,  8861547,  8996264,  9132459,  9300402,  9478450,
        9659824,  9849987, 10028957, 10193022, 10357062, 10533664,
       10709501, 10920728, 11124729, 11327529, 11499395, 11668657,
       11873991, 12102666, 12322513, 12553372, 12768313, 12963946,
       13151805, 13377890, 13615561, 10272645])
```

```
total_deaths = np.array(cumulative_deaths[' Cumulative_deaths'])
total_deaths
```

```
array([     1,      1,      1,      1,      1,      1,      2,      2,
            3,      4,      6,      9,     17,     25,     41,     56,
```

```
days_since_start = np.array([i for i in range(len(dates))]).reshape(-1, 1)
world_cases      = np.array(world_cases).reshape(-1, 1)
total_deaths     = np.array(total_deaths).reshape(-1, 1)
```

```
         2874,   2942,   2987,   3074,   3154,   3245,   3346,   3459,
```

Future Forecasting

```
days_in_future  = 30  # WE WILL GO 30 DAYS INTO THE FUTURE
future_forecast = np.array([i for i in range(len(dates)+days_in_future)]).reshape(-1, 1)
adjusted_dates  = future_forecast[:-30]
```

```
        276677  281326  284829  289055  293954  299077  304238  308820
```

```
start = earliest_reported
start_date = datetime.datetime.strptime(start, '%Y-%m-%d')
future_forecast_dates = []
for i in range(len(future_forecast)):
    future_forecast_dates.append((start_date + datetime.timedelta(days=i)).strftime( '%m/%d/%Y'
```

```
        528216, 532352, 535770, 539915, 545486, 551048, 556329, 561614,
```

Machine Learning

```
# Have a function to go through each country one by one and create a new .csvfile.

def one_country(country_name, feature):
    one_country = who_data[who_data[' Country']==country_name]
    print("Processing: ", country_name.upper())
    #print(one_country)

    # reconfigure the data so we have just deaths
    X = one_country.filter(['Date_reported', feature], axis = 1)
    X = X.groupby('Date_reported', as_index=False).sum()
    #print(X.tail(3))

    y = np.array(X[feature])
    # y = fft(y)
    one_cases  = np.array(y).reshape(-1, 1)

    dates = X['Date_reported']

    days_since_start= np.array([i for i in range(len(dates))]).reshape(-1, 1)

    days_in_future  = 30  # WE WILL GO 30 DAYS INTO THE FUTURE
    future_forecast = np.array([i for i in range(len(dates)+days_in_future)]).reshape(-1, 1)

    adjusted_dates  = future_forecast[:-30]



    ### print(len(future_forecast))
    ### Split the data
    X_train_confirmed, X_test_confirmed,y_train_confirmed, y_test_confirmed = train_test_split(

    #building SVM Model
```

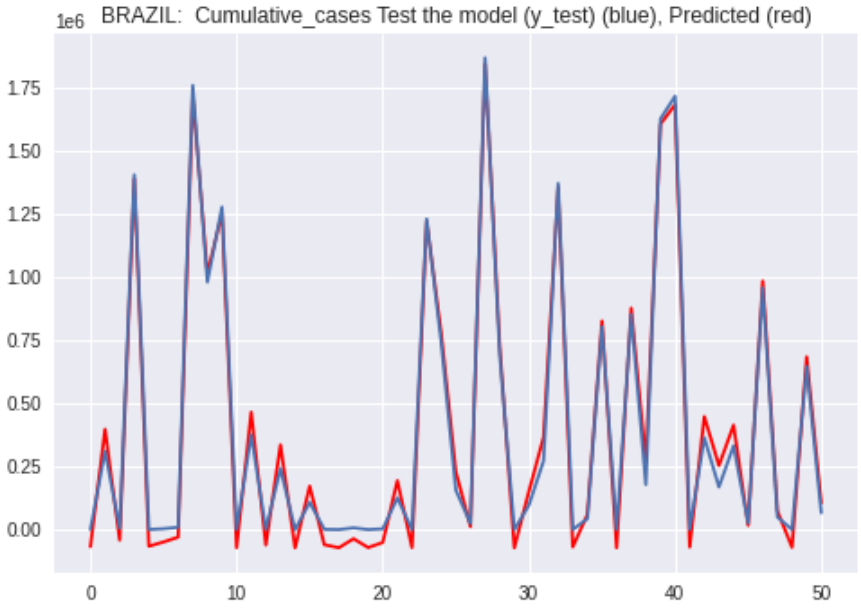
```
    kernel = ['poly', 'sigmoid', 'rbf']
    c = [0.01, 0.1, 1 ]
    gamma = [0.01, 0.1]
    epsilon = [0.01, 0.1]
    shrinking = [True, False]
    svm_grid = {'kernel':kernel, 'C':c, 'gamma':gamma, 'epsilon':epsilon, 'shrinking':shrinking

    svm        = SVR()
    svm_search = RandomizedSearchCV(svm, svm_grid, scoring='neg_mean_squared_error', cv = 3, n_

    svm_search.fit(X_train_confirmed, y_train_confirmed.ravel())


    svm_search.best_params_
    svm_confirmed = svm_search.best_estimator_
    svm_pred      = svm_confirmed.predict(future_forecast)

    # Check against testing data

    svm_test_pred = svm_confirmed.predict(X_test_confirmed)
    plt.figure()
    plt.title(country_name.upper() + ': ' + feature + ' Test the model (y_test) (blue), Predict
    plt.plot(svm_test_pred, color='RED') #Predicted.
    plt.plot(y_test_confirmed)   #Actual

    images_dir = '/Drive/My Drive/Colab Notebooks/COVID19/'
    plt.savefig(f"{images_dir}/{country}_test.png")

    # print ('Mean Absolute Error', mean_absolute_error(svm_test_pred, y_test_confirmed))
    # print ('Mean Squared Error ',  mean_squared_error(svm_test_pred, y_test_confirmed))

    plt.figure(figsize=(20, 12))
    plt.title(country_name.upper() + ": " + feature + ' (blue), Predicted (red)')
    plt.plot(adjusted_dates, one_cases)
    _ = plt.plot(future_forecast, svm_pred, linestyle='dashed', color='red')

    score = svm_confirmed.score(X_train_confirmed, y_train_confirmed)
    print('Country: ', country, " training score: ", score)

    score = svm_confirmed.score(X_test_confirmed, y_test_confirmed)
    print('Country: ',country, " test score: ", score)


    images_dir = '/Drive/My Drive/Colab Notebooks/COVID19/'
    plt.savefig(f"{images_dir}/{country}_pred.png")


    return(svm_pred)


India','Iran (Islamic Republic of)', 'Italy', 'Russian Federation', 'Spain', 'The United Kingdor


rst_affected_countries = ['Brazil', 'Chile', 'India', 'Iran (Islamic Republic of)', 'Mexico', 'I


worst_affected_countries = ['South Africa']
```

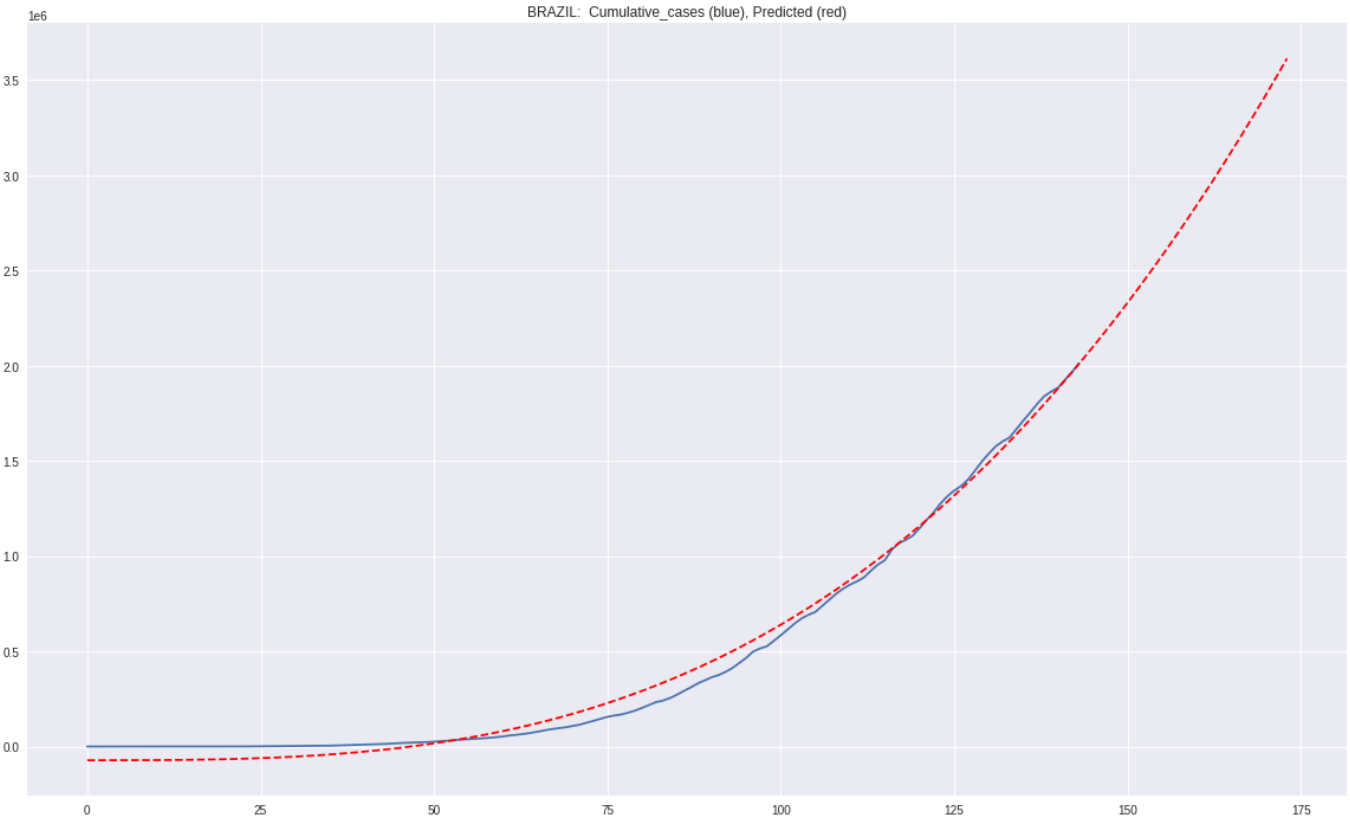
```
## LOOP THROUGH EACH COUNTRY
for country in worst_affected_countries:
    print(country)
    df =who_data[who_data[' Country']==country].reset_index()
    df = df.drop(['index'], axis=1)
    Country = df[' Country'][0]
    Country_code =  df[' Country_code'][0]
    WHO_region = df[' WHO_region'][0]
    #add the 30 days
    df = df.append(pd.DataFrame({'Date_reported': pd.date_range(start=df['Date_reported'].iloc[

    df = df.reset_index()
    df = df.drop(['index'], axis=1)

    df[' Country']      = Country
    df[' Country_code'] = Country_code
    df[' WHO_region']   = WHO_region
    pred_cumulative_cases   = one_country(country, ' Cumulative_cases')
    pred_cumulative_deaths  = one_country(country, ' Cumulative_deaths')
    # pred_new_cases          = one_country(country, ' New_cases')
    # pred_new_deaths         = one_country(country, ' New_deaths')

    df['PRED_CUMULATIVE_CASES'] = pred_cumulative_cases
    df['PRED_CUMULATIVE_DEATHS'] = pred_cumulative_deaths
    # df['PRED_NEW_CASES'] = pred_new_cases
    # df['PRED_NEW_DEATHS'] = pred_new_deaths


    with open('/Drive/My Drive/Colab Notebooks/COVID19/'+country+'.csv', 'w') as f:
      df.to_csv(f)
```

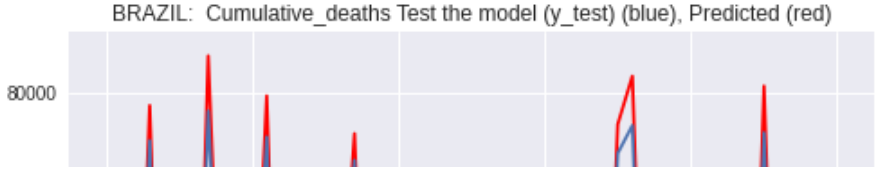
```
Brazil
Processing:  BRAZIL
Fitting 3 folds for each of 10 candidates, totalling 30 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  27 out of  30 | elapsed:   10.2s remaining:    1.1s
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed:   17.0s finished
Country:  Brazil  training score:  0.9925035287568548
Country:  Brazil  test score:  0.9893067930523401
Processing:  BRAZIL
Fitting 3 folds for each of 10 candidates, totalling 30 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  27 out of  30 | elapsed:   23.2s remaining:    2.6s
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed:  2.4min finished
Country:  Brazil  training score:  0.9908745691188993
Country:  Brazil  test score:  0.9848395386338618
Chile
Processing:  CHILE
Fitting 3 folds for each of 10 candidates, totalling 30 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed:    0.7s finished
Country:  Chile  training score:  0.9766477692257893
Country:  Chile  test score:  0.9757416344107424
Processing:  CHILE
Fitting 3 folds for each of 10 candidates, totalling 30 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed:  2.5min finished
Country:  Chile  training score:  0.9611708634583709
Country:  Chile  test score:  0.9631629076162626
India
Processing:  INDIA
Fitting 3 folds for each of 10 candidates, totalling 30 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed:    0.1s finished
Country:  India  training score:  0.9112439700242585
Country:  India  test score:  0.882231517612234
Processing:  INDIA
Fitting 3 folds for each of 10 candidates, totalling 30 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed:    9.0s finished
Country:  India  training score:  0.9530143101536356
Country:  India  test score:  0.9260427901357443
Iran (Islamic Republic of)
Processing:  IRAN (ISLAMIC REPUBLIC OF)
Fitting 3 folds for each of 10 candidates, totalling 30 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed:    8.3s finished
Country:  Iran (Islamic Republic of)  training score:  0.8786958488530864
Country:  Iran (Islamic Republic of)  test score:  0.8925621056608027
Processing:  IRAN (ISLAMIC REPUBLIC OF)
Fitting 3 folds for each of 10 candidates, totalling 30 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  27 out of  30 | elapsed:   51.1s remaining:    5.7s
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed:  1.4min finished
Country:  Iran (Islamic Republic of)  training score:  0.8484771219138945
Country:  Iran (Islamic Republic of)  test score:  0.7942896811591665
Mexico
Processing:  MEXICO
Fitting 3 folds for each of 10 candidates, totalling 30 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed: 11.6min finished
Country:  Mexico  training score:  0.9979105517606817
Country:  Mexico  test score:  0.9968147400054161
Processing:  MEXICO
Fitting 3 folds for each of 10 candidates, totalling 30 fits
```

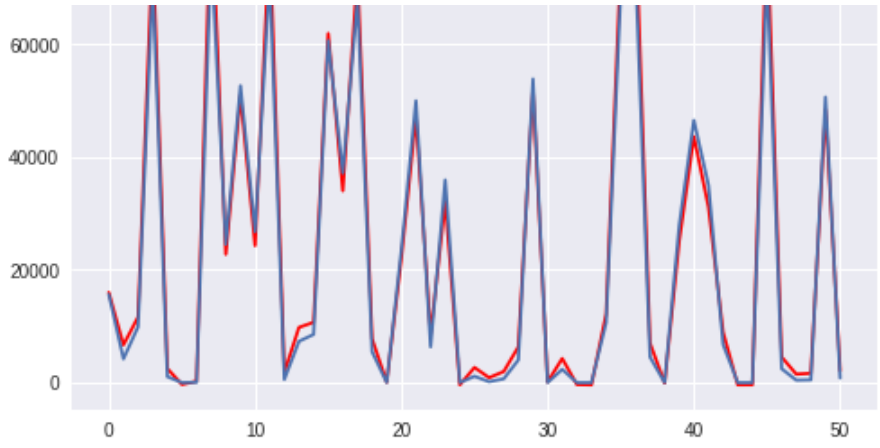
```
Fitting 3 folds for each of 10 candidates, totalling 30 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed:  1.4min finished
Country:  Mexico  training score:  0.993749043251022
Country:  Mexico  test score:  0.9943525002903069
Peru
Processing:  PERU
Fitting 3 folds for each of 10 candidates, totalling 30 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  27 out of  30 | elapsed:    0.7s remaining:    0.1s
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed:    1.0s finished
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:54: RuntimeWarning: More than
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:65: RuntimeWarning: More than
Country:  Peru  training score:  0.948151731438868
Country:  Peru  test score:  0.9378530209994066
Processing:  PERU
Fitting 3 folds for each of 10 candidates, totalling 30 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  27 out of  30 | elapsed:    3.2s remaining:    0.4s
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed:   17.3s finished
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:54: RuntimeWarning: More than
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:65: RuntimeWarning: More than
Country:  Peru  training score:  0.9894354730240016
Country:  Peru  test score:  0.9879348621031396
Russian Federation
Processing:  RUSSIAN FEDERATION
Fitting 3 folds for each of 10 candidates, totalling 30 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed:    0.1s finished
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:54: RuntimeWarning: More than
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:65: RuntimeWarning: More than
Country:  Russian Federation  training score:  0.9692732067528602
Country:  Russian Federation  test score:  0.957962635687427
Processing:  RUSSIAN FEDERATION
Fitting 3 folds for each of 10 candidates, totalling 30 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed: 35.3min finished
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:54: RuntimeWarning: More than
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:65: RuntimeWarning: More than
Country:  Russian Federation  training score:  0.9866832569084781
Country:  Russian Federation  test score:  0.9912223279095053
The United Kingdom
Processing:  THE UNITED KINGDOM
Fitting 3 folds for each of 10 candidates, totalling 30 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed:    2.4s finished
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:54: RuntimeWarning: More than
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:65: RuntimeWarning: More than
Country:  The United Kingdom  training score:  0.692752235116248
Country:  The United Kingdom  test score:  0.7272482782375589
Processing:  THE UNITED KINGDOM
Fitting 3 folds for each of 10 candidates, totalling 30 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed:    3.3s finished
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:54: RuntimeWarning: More than
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:65: RuntimeWarning: More than
Country:  The United Kingdom  training score:  0.7283291084974526
Country:  The United Kingdom  test score:  0.7761643659775367
United States of America
Processing:  UNITED STATES OF AMERICA
Fitting 3 folds for each of 10 candidates, totalling 30 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed:    0.8s finished
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:54: RuntimeWarning: More than
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:65: RuntimeWarning: More than
```

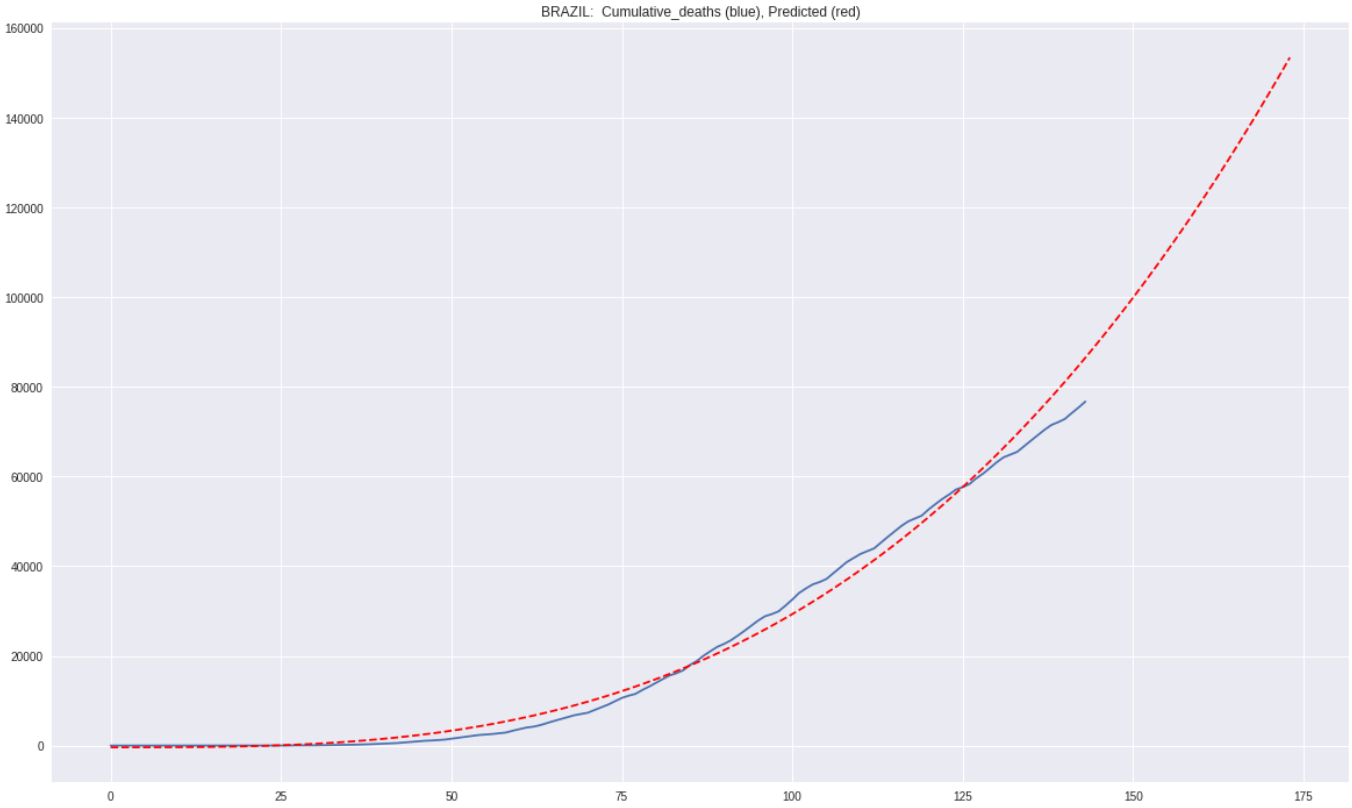
```
Country:  United States of America  training score:  0.9583216601189959
Country:  United States of America  test score:  0.9699866428096187
Processing:  UNITED STATES OF AMERICA
Fitting 3 folds for each of 10 candidates, totalling 30 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  27 out of  30 | elapsed:    6.3s remaining:    0.7s
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed:    8.2s finished
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:54: RuntimeWarning: More than
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:65: RuntimeWarning: More than
Country:  United States of America  training score:  0.8708859590591964
Country:  United States of America  test score:  0.8585405155252228
```

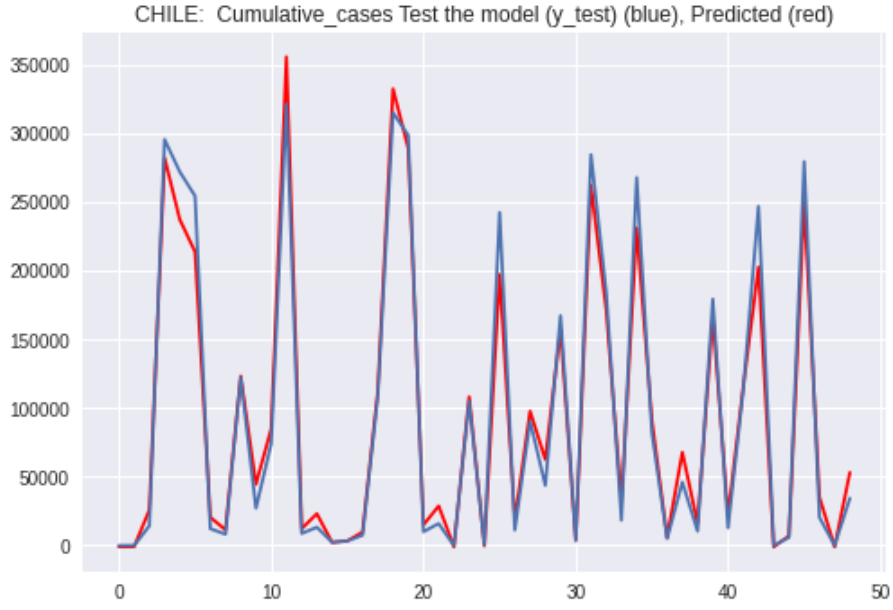


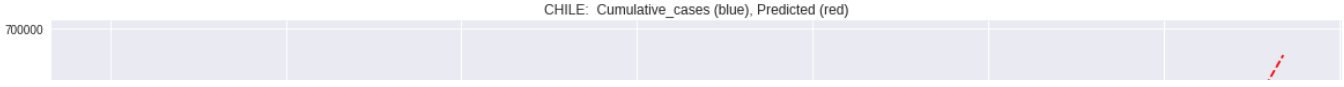


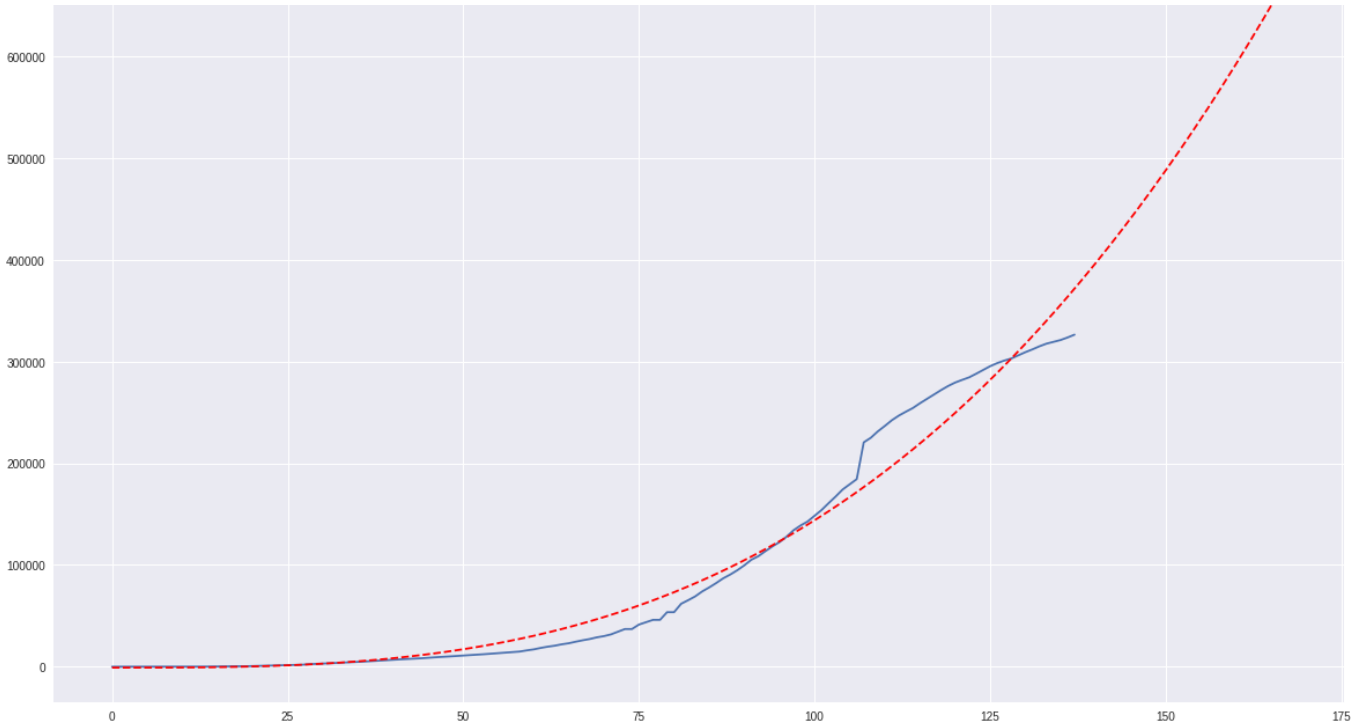



60000
40000
20000
0
0
10
20
30
40
50

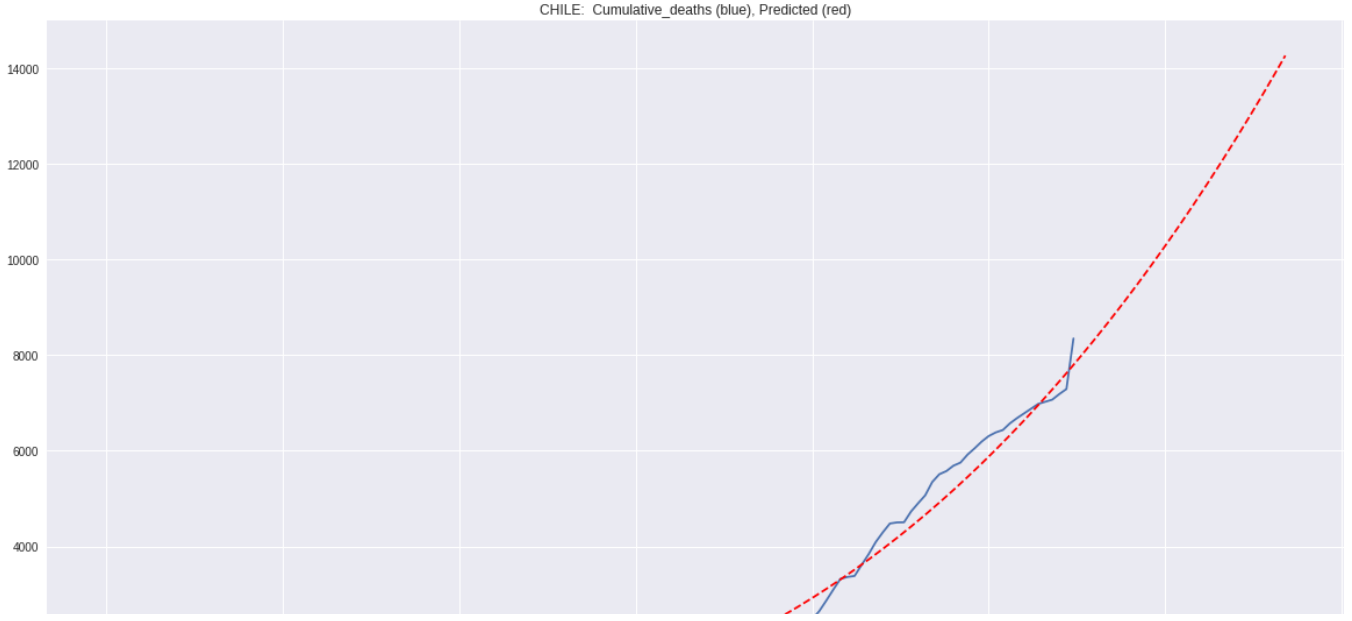

BRAZIL: Cumulative_deaths (blue), Predicted (red)
160000
140000
120000
100000
80000
60000
40000
20000
0
0
25
50
75
100
125
150
175

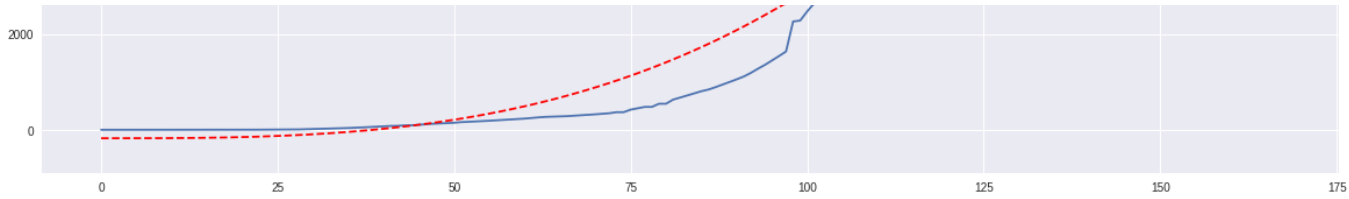

CHILE: Cumulative_cases Test the model (y_test) (blue), Predicted (red)
350000
300000
250000
200000
150000
100000
50000
0
0
10
20
30
40
50

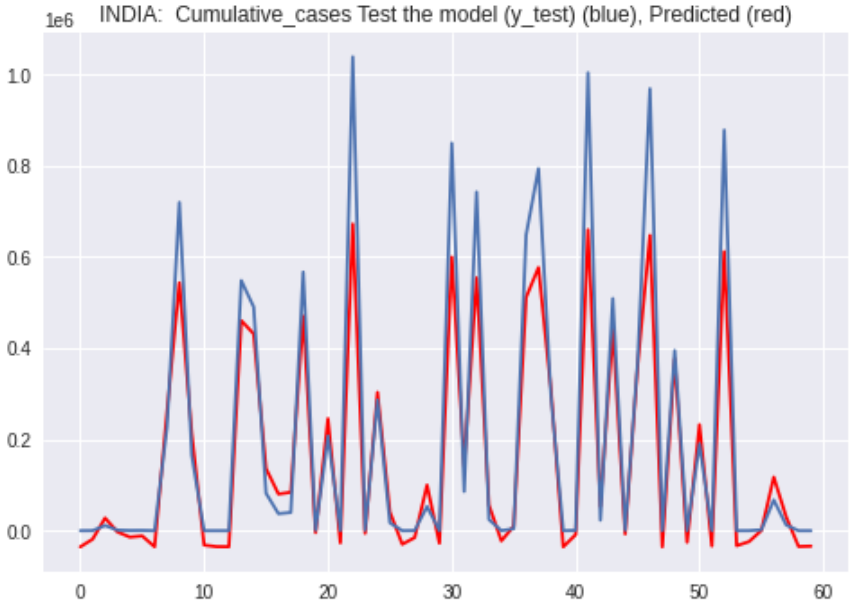

CHILE: Cumulative_cases (blue), Predicted (red)
700000

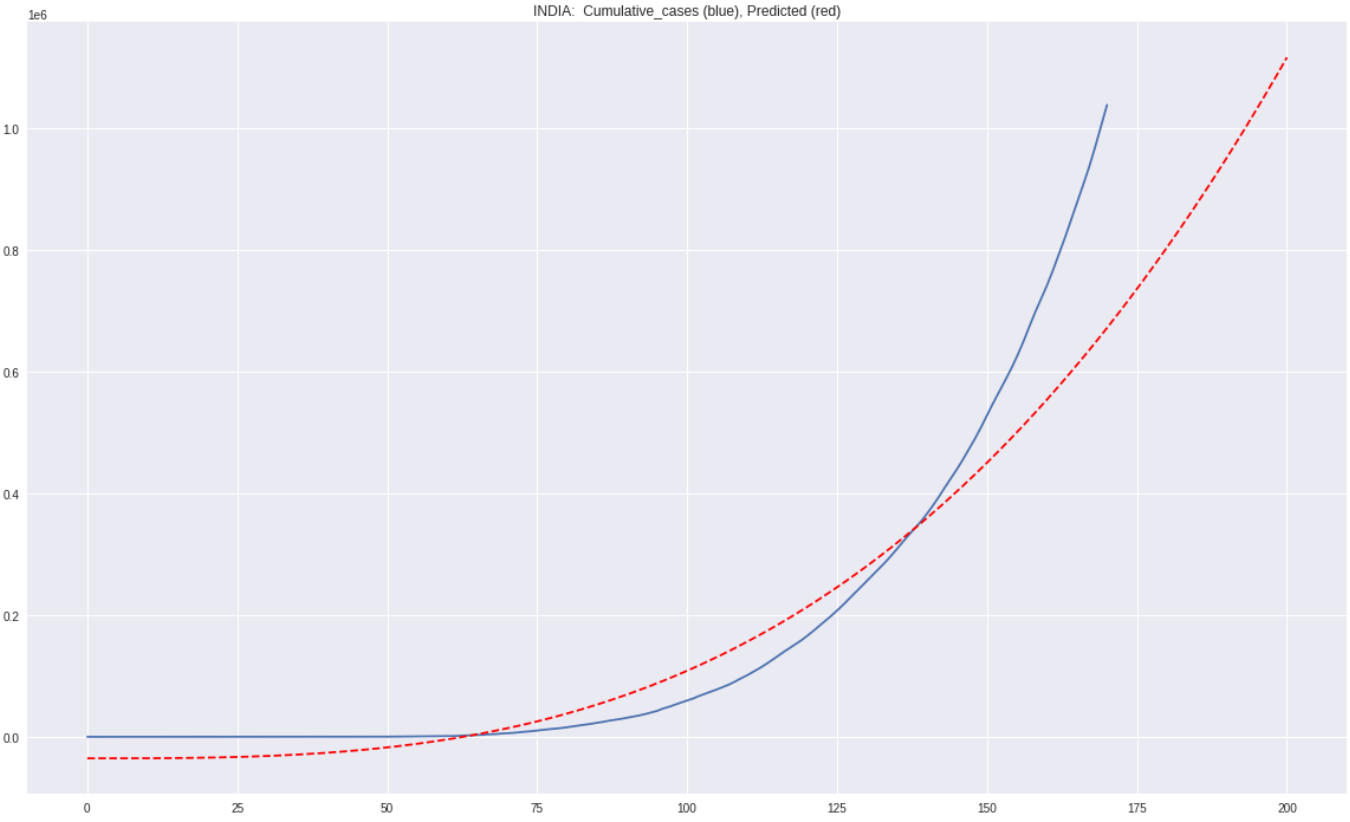

600000
500000
400000
300000
200000
100000
0
0
25
50
75
100
125
150
175

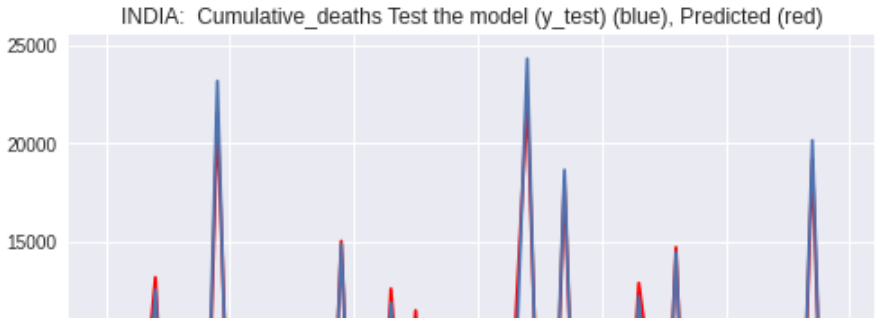

CHILE: Cumulative_deaths Test the model (y_test) (blue), Predicted (red)
7000
6000
5000
4000
3000
2000
1000
0
0
10
20
30
40
50

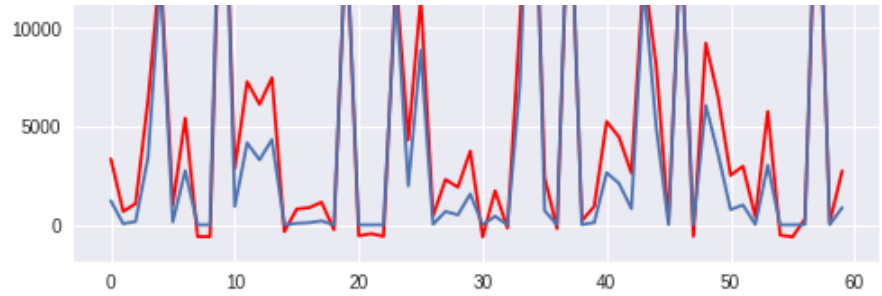

CHILE: Cumulative_deaths (blue), Predicted (red)
14000
12000
10000
8000
6000
4000

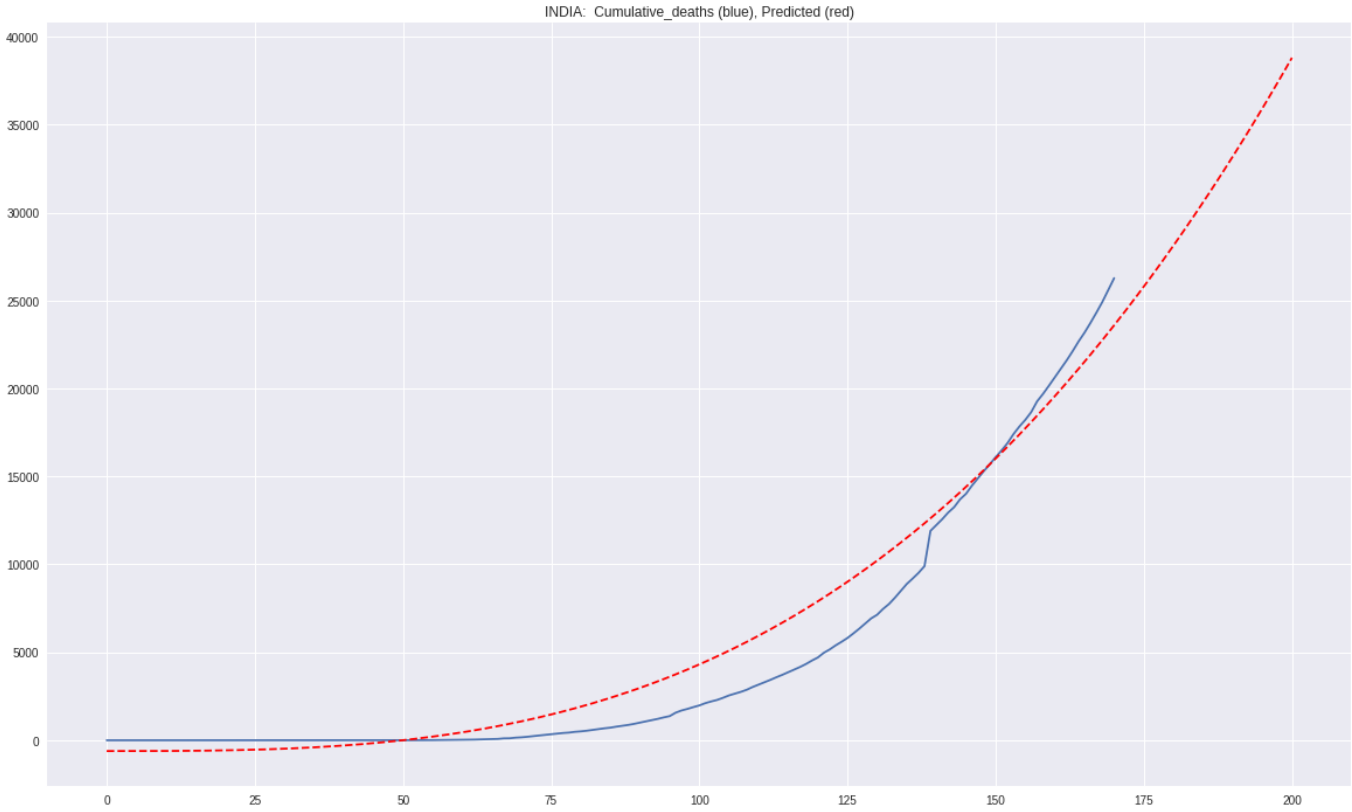

2000
0
0
25
50
75
100
125
150
175

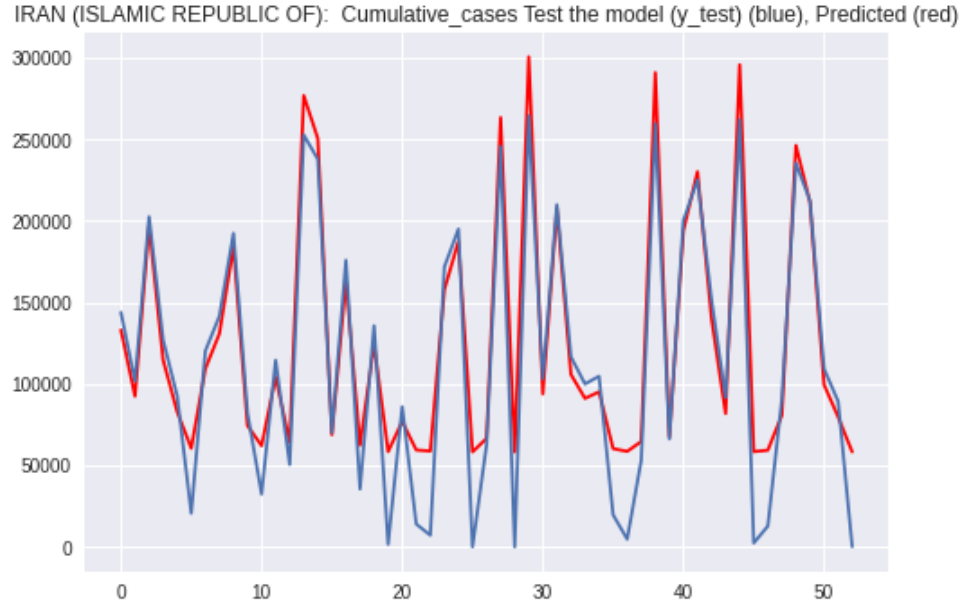

1e6
INDIA: Cumulative_cases Test the model (y_test) (blue), Predicted (red)
1.0
0.8
0.6
0.4
0.2
0.0
0
10
20
30
40
50
60

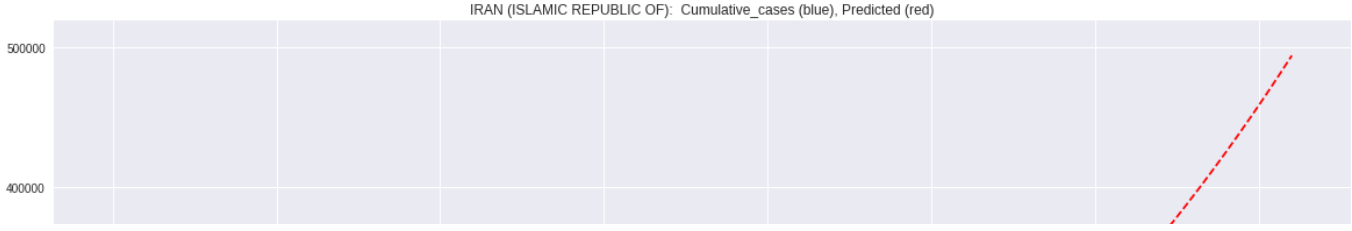

1e6
INDIA: Cumulative_cases (blue), Predicted (red)
1.0
0.8
0.6
0.4
0.2
0.0
0
25
50
75
100
125
150
175
200

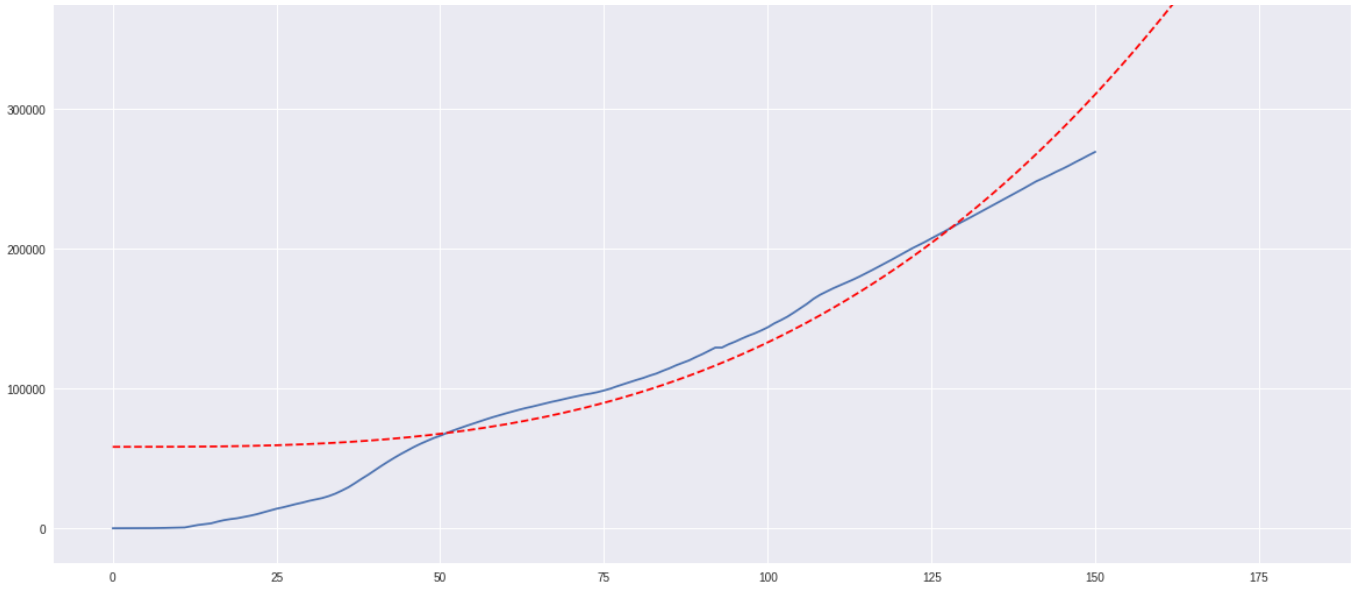

INDIA: Cumulative_deaths Test the model (y_test) (blue), Predicted (red)
25000
20000
15000


10000
5000
0
0
10
20
30
40
50
60

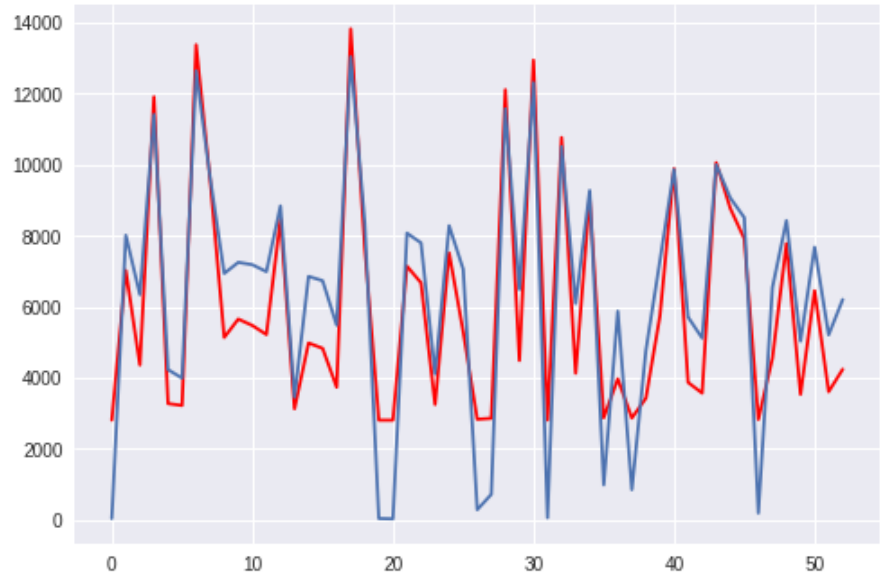

INDIA: Cumulative_deaths (blue), Predicted (red)
40000
35000
30000
25000
20000
15000
10000
5000
0
0
25
50
75
100
125
150
175
200


IRAN (ISLAMIC REPUBLIC OF): Cumulative_cases Test the model (y_test) (blue), Predicted (red)
300000
250000
200000
150000
100000
50000
0
0
10
20
30
40
50

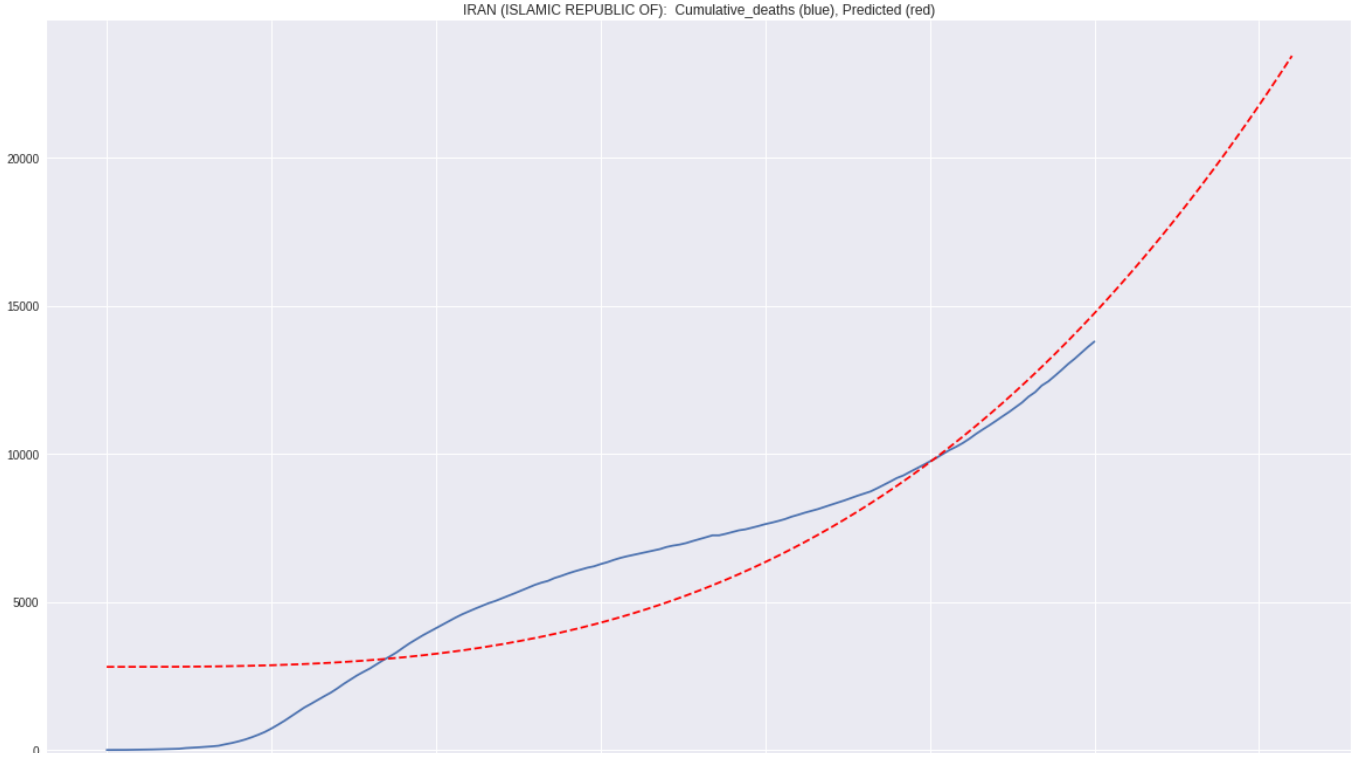

IRAN (ISLAMIC REPUBLIC OF): Cumulative_cases (blue), Predicted (red)
500000
400000



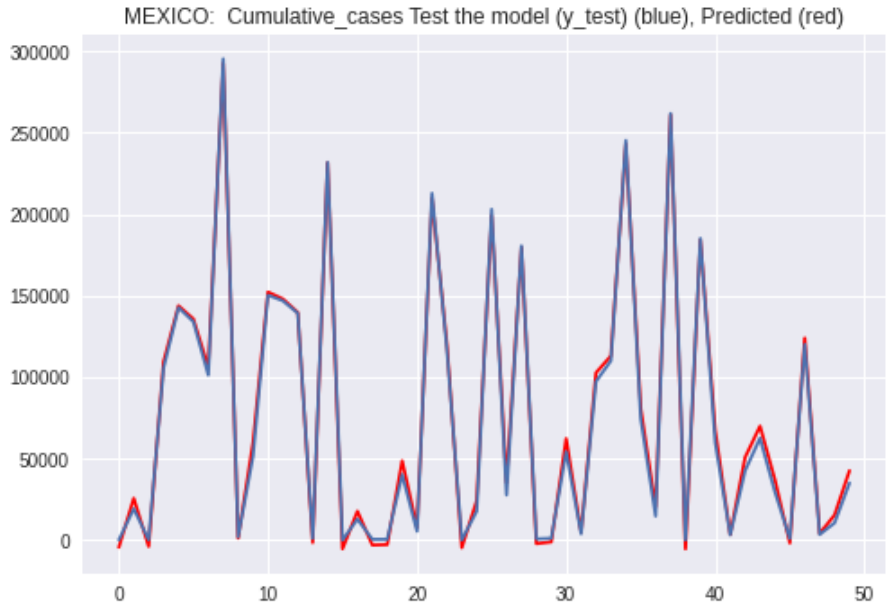
IRAN (ISLAMIC REPUBLIC OF): Cumulative_deaths Test the model (y_test) (blue), Predicted (red)

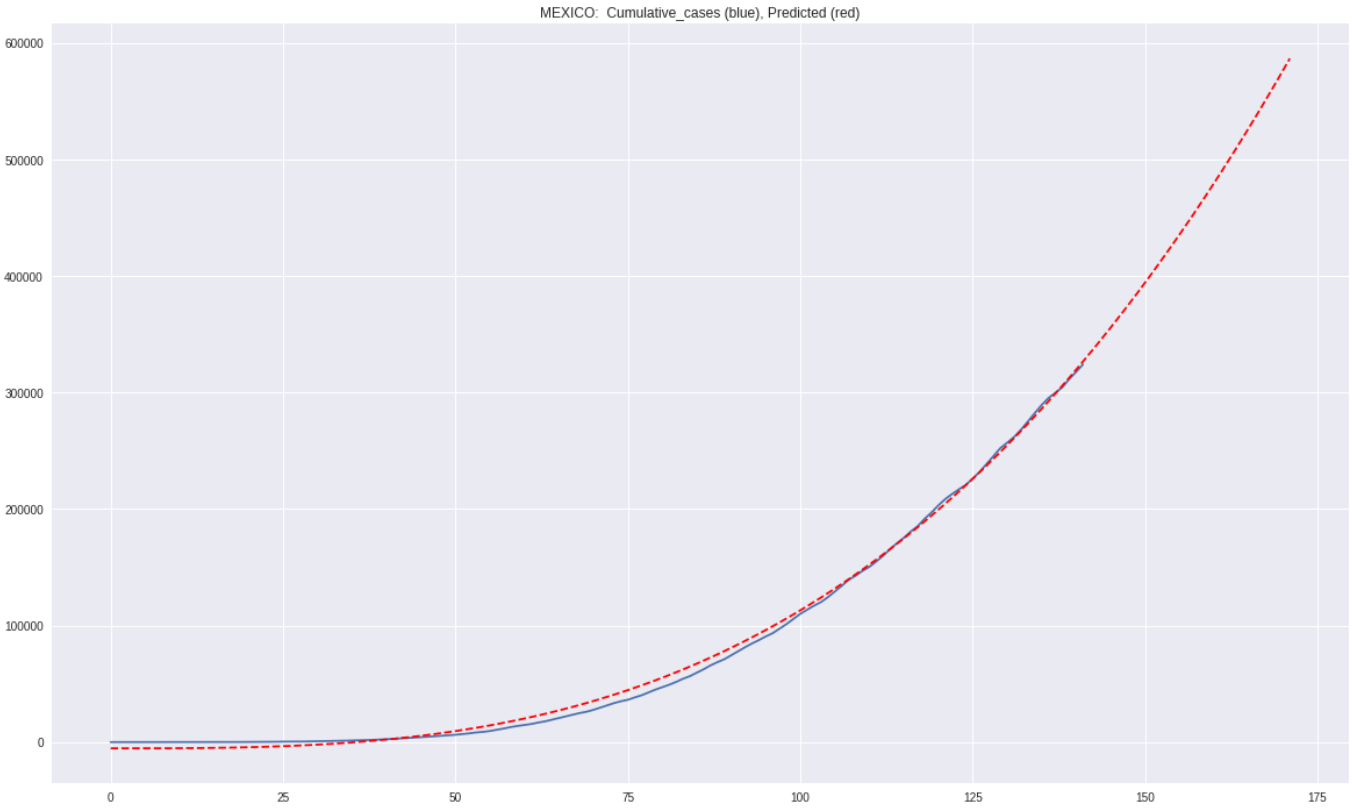


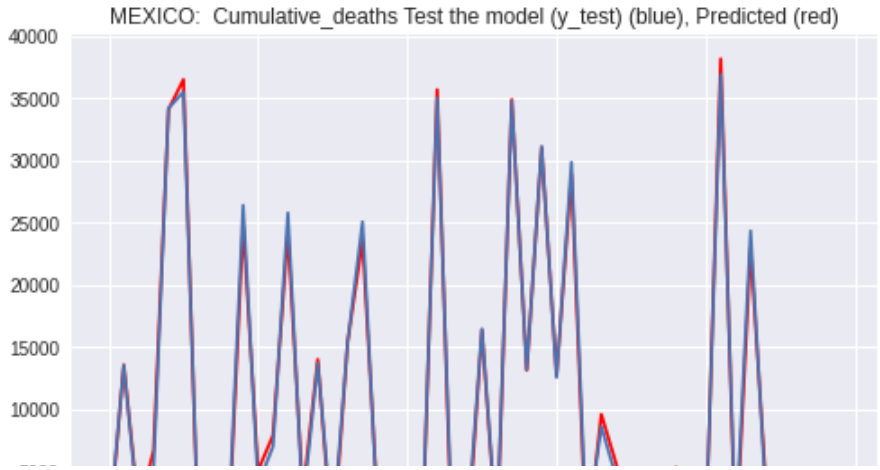
IRAN (ISLAMIC REPUBLIC OF): Cumulative_deaths (blue), Predicted (red)

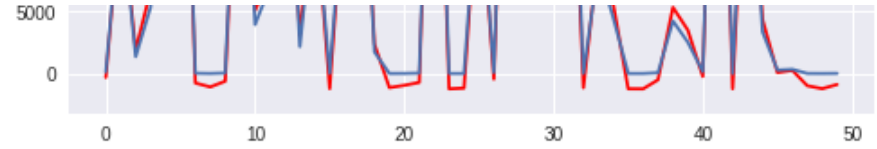


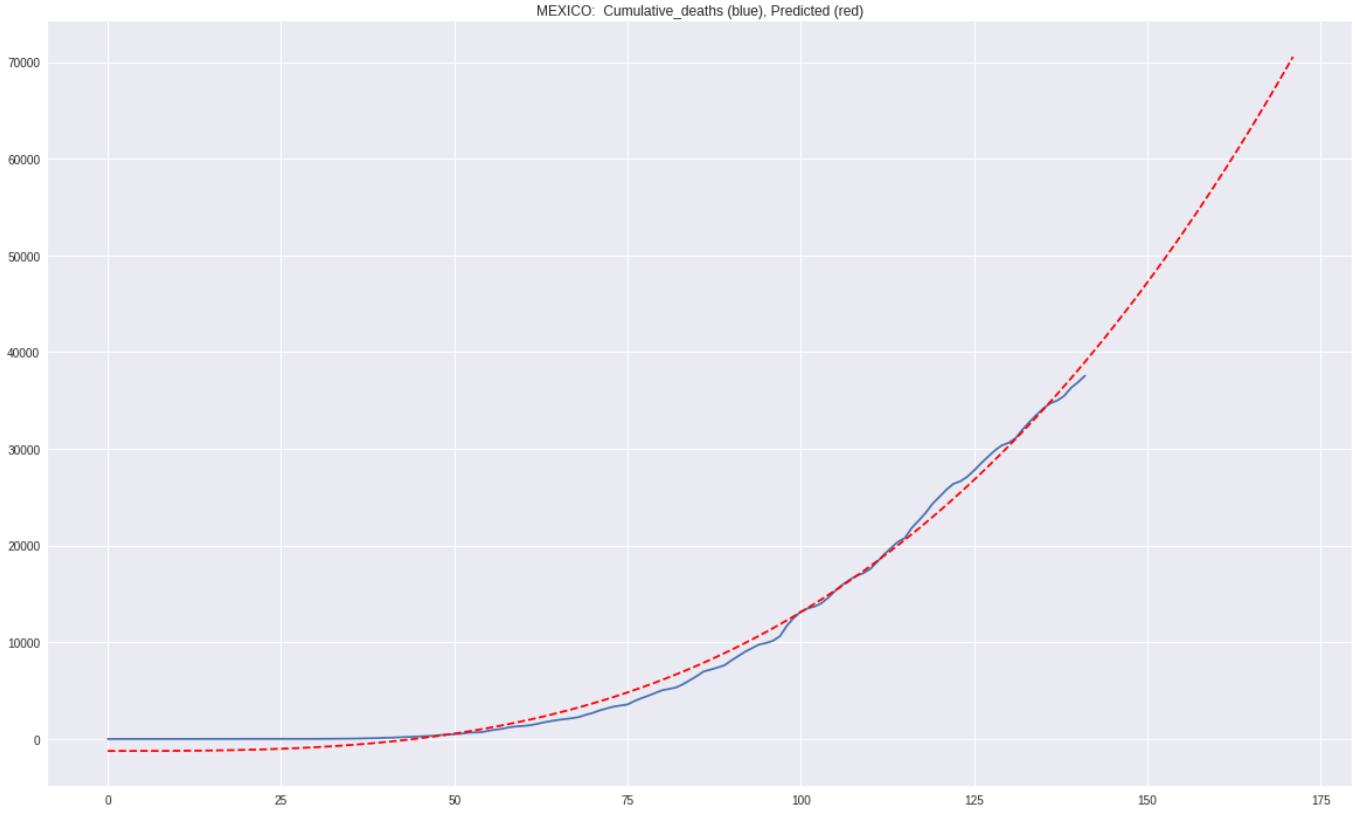

0
25
50
75
100
125
150
175

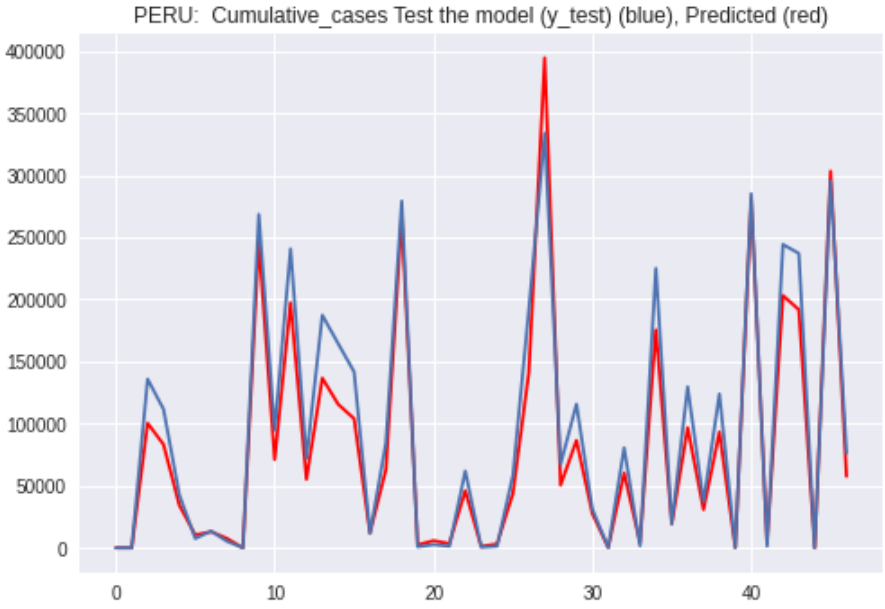

MEXICO: Cumulative_cases Test the model (y_test) (blue), Predicted (red)
300000
250000
200000
150000
100000
50000
0
0
10
20
30
40
50

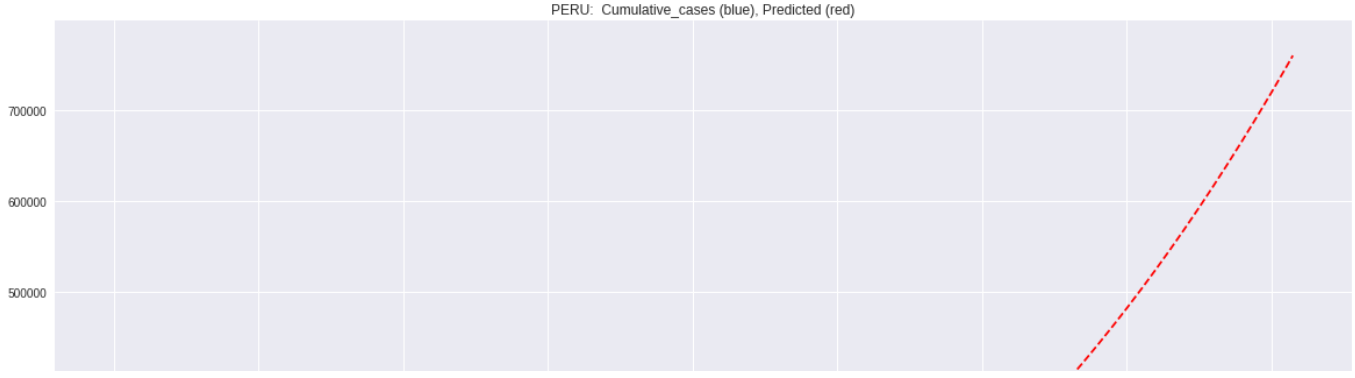

MEXICO: Cumulative_cases (blue), Predicted (red)
600000
500000
400000
300000
200000
100000
0
0
25
50
75
100
125
150
175

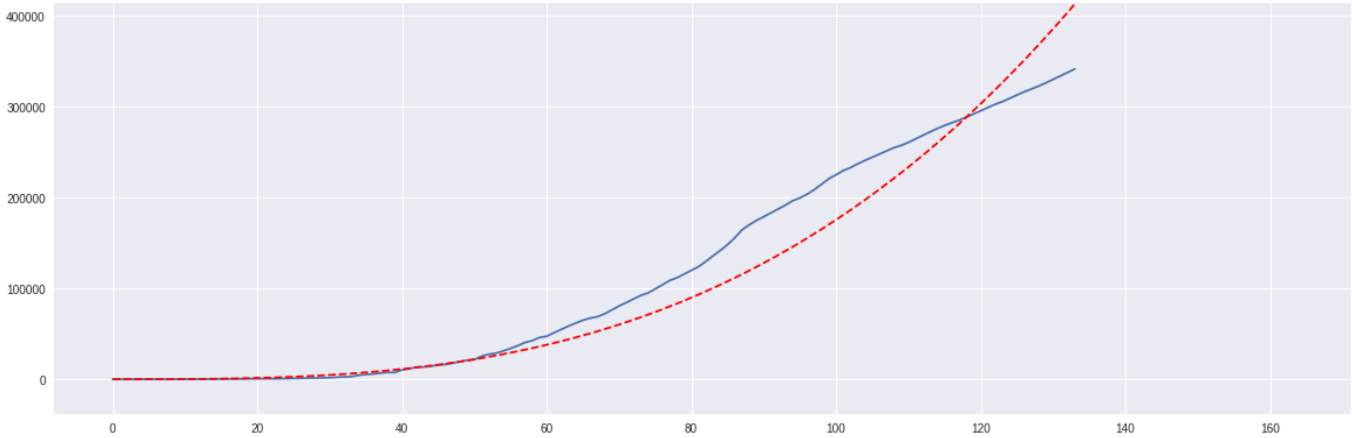

MEXICO: Cumulative_deaths Test the model (y_test) (blue), Predicted (red)
40000
35000
30000
25000
20000
15000
10000

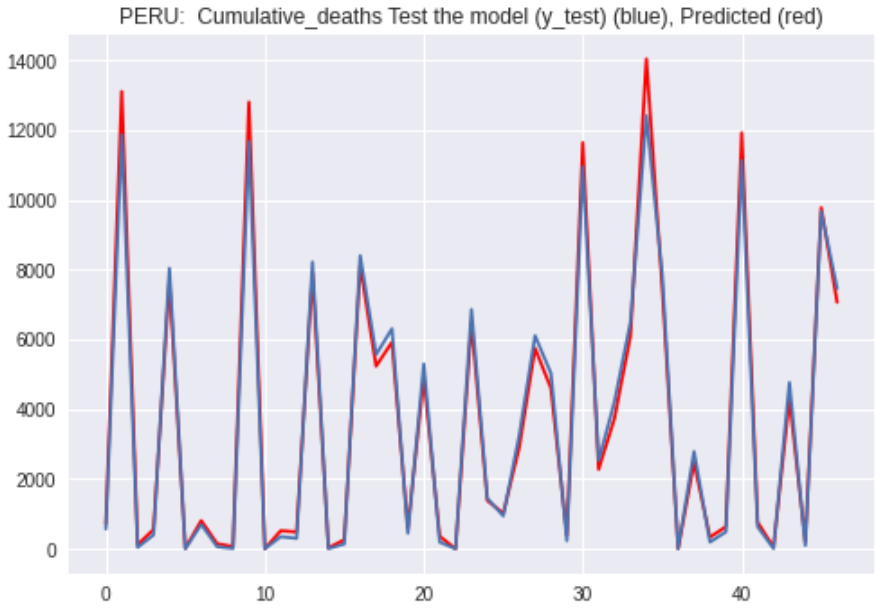

5000
0
0
10
20
30
40
50

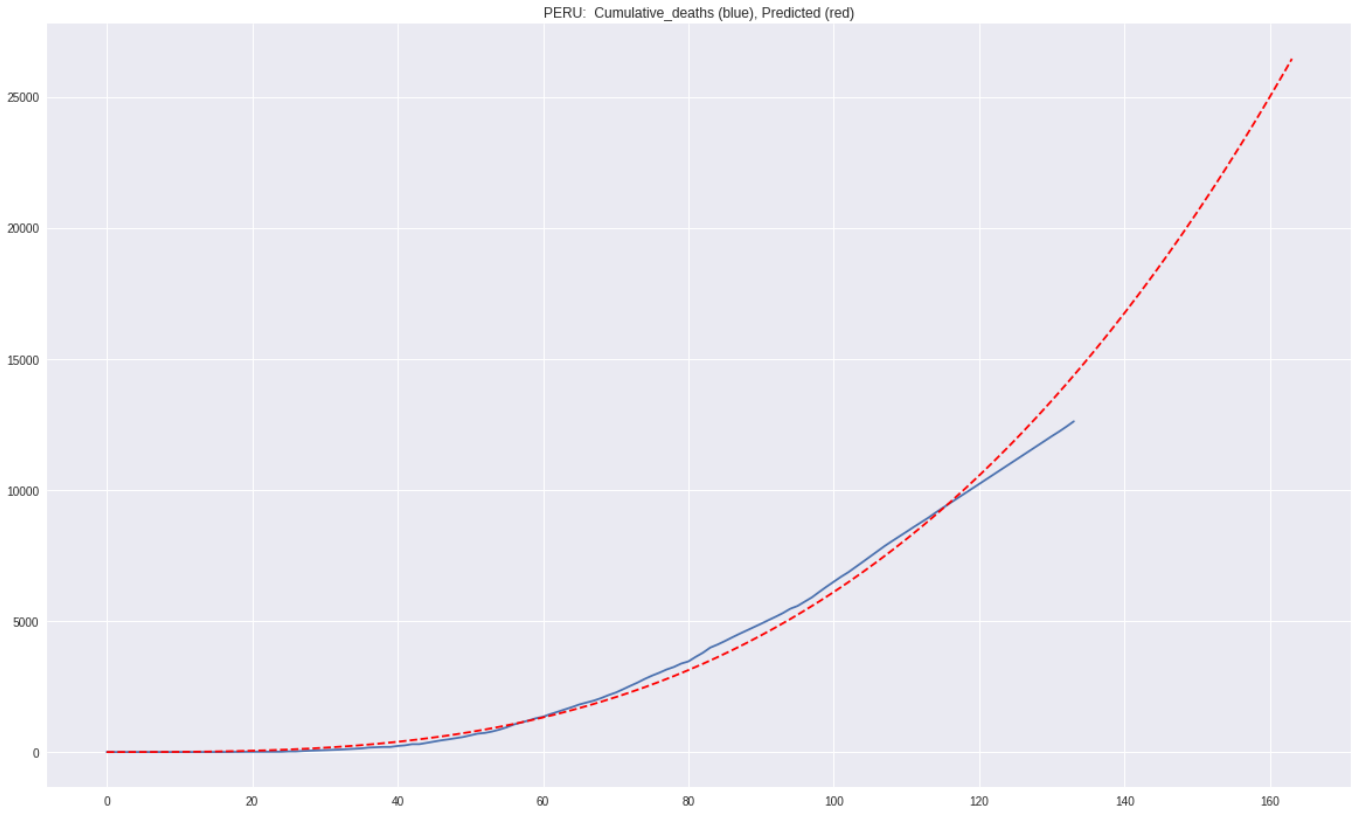

MEXICO: Cumulative_deaths (blue), Predicted (red)
70000
60000
50000
40000
30000
20000
10000
0
0
25
50
75
100
125
150
175


PERU: Cumulative_cases Test the model (y_test) (blue), Predicted (red)
400000
350000
300000
250000
200000
150000
100000
50000
0
0
10
20
30
40

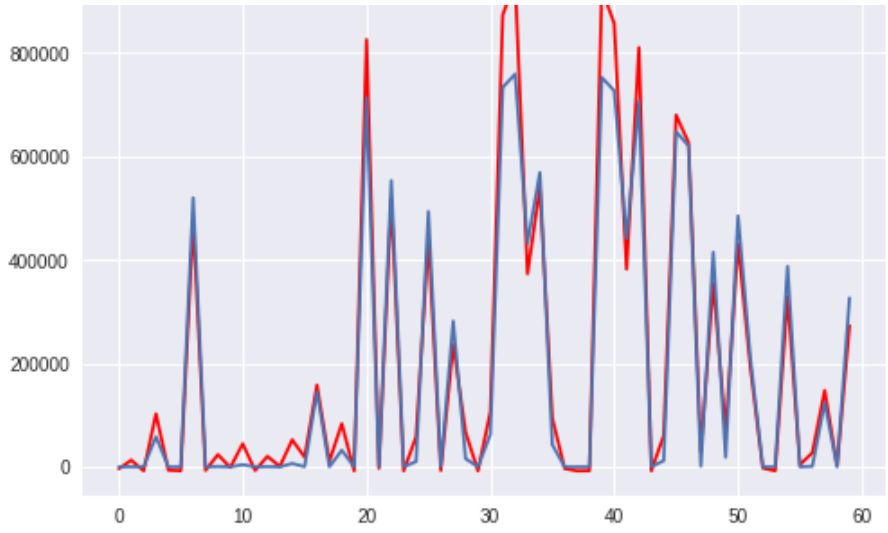

PERU: Cumulative_cases (blue), Predicted (red)
700000
600000
500000

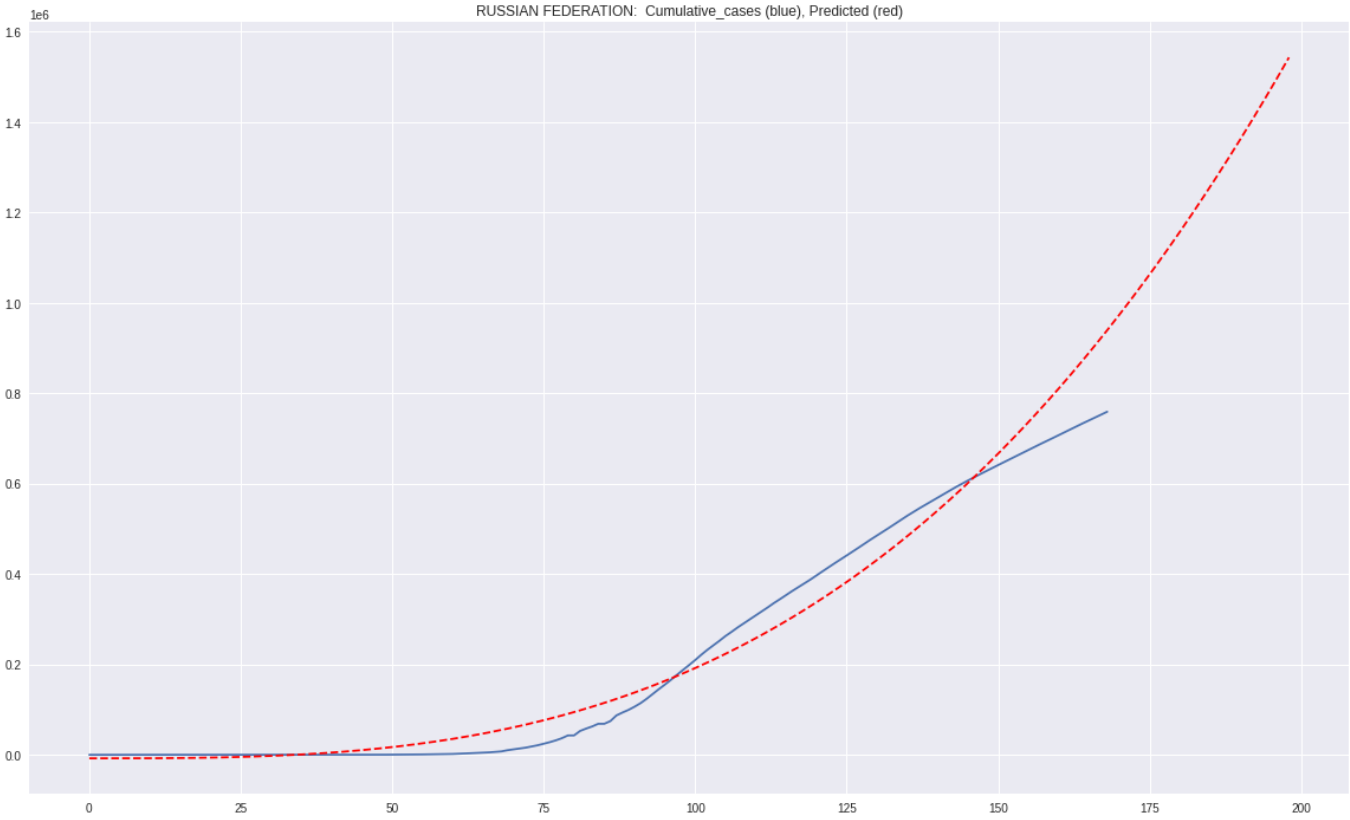

400000
300000
200000
100000
0
0
20
40
60
80
100
120
140
160

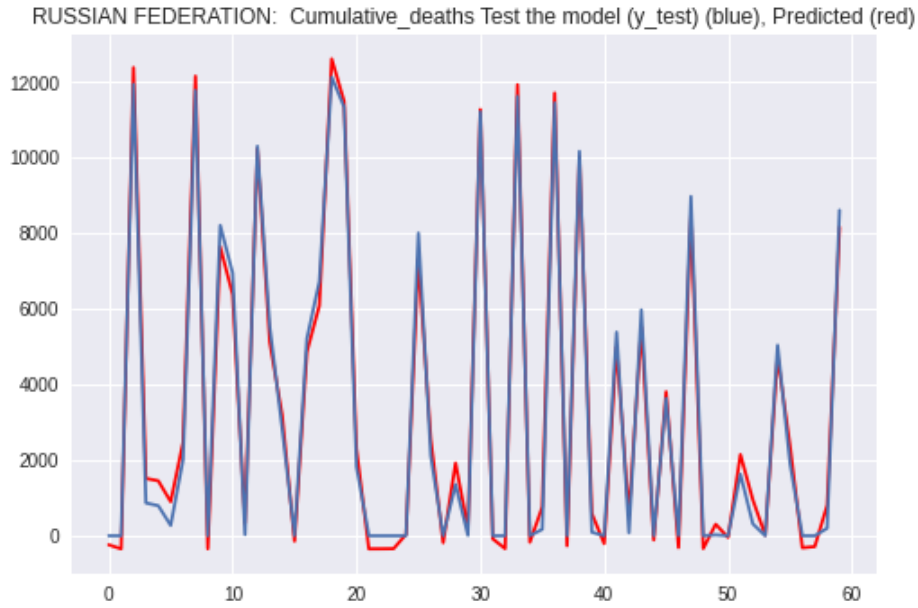

PERU: Cumulative_deaths Test the model (y_test) (blue), Predicted (red)
14000
12000
10000
8000
6000
4000
2000
0
0
10
20
30
40

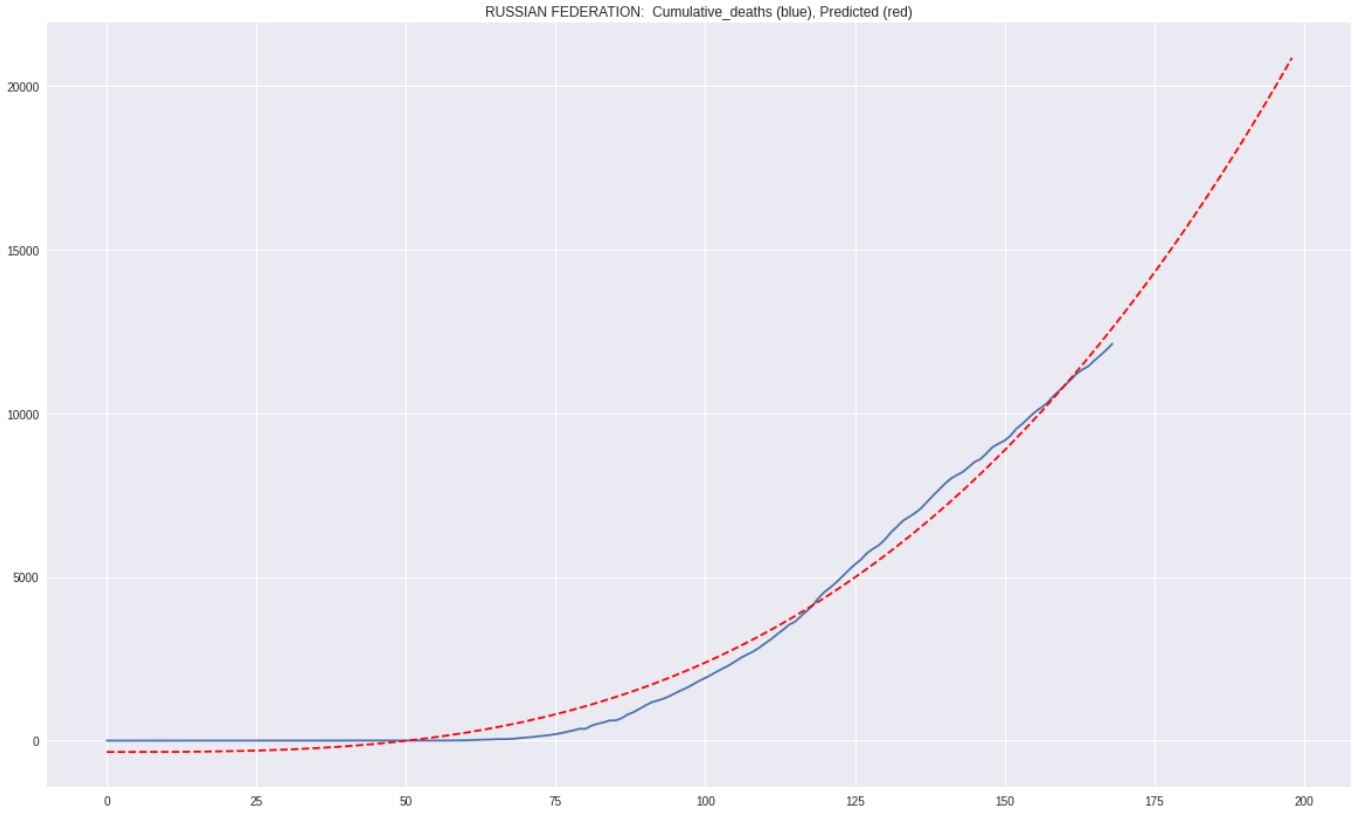

PERU: Cumulative_deaths (blue), Predicted (red)
25000
20000
15000
10000
5000
0
0
20
40
60
80
100
120
140
160

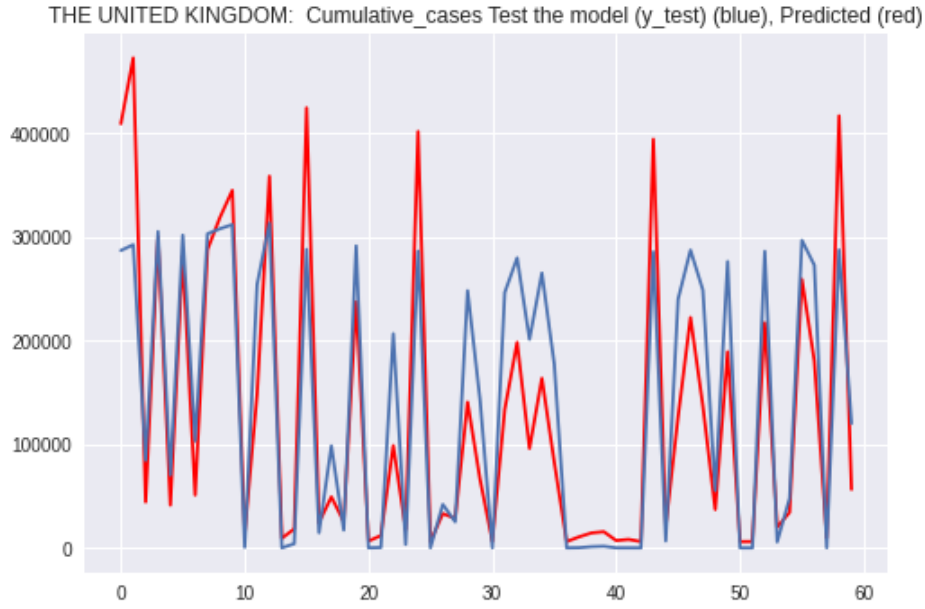

RUSSIAN FEDERATION: Cumulative_cases Test the model (y_test) (blue), Predicted (red)

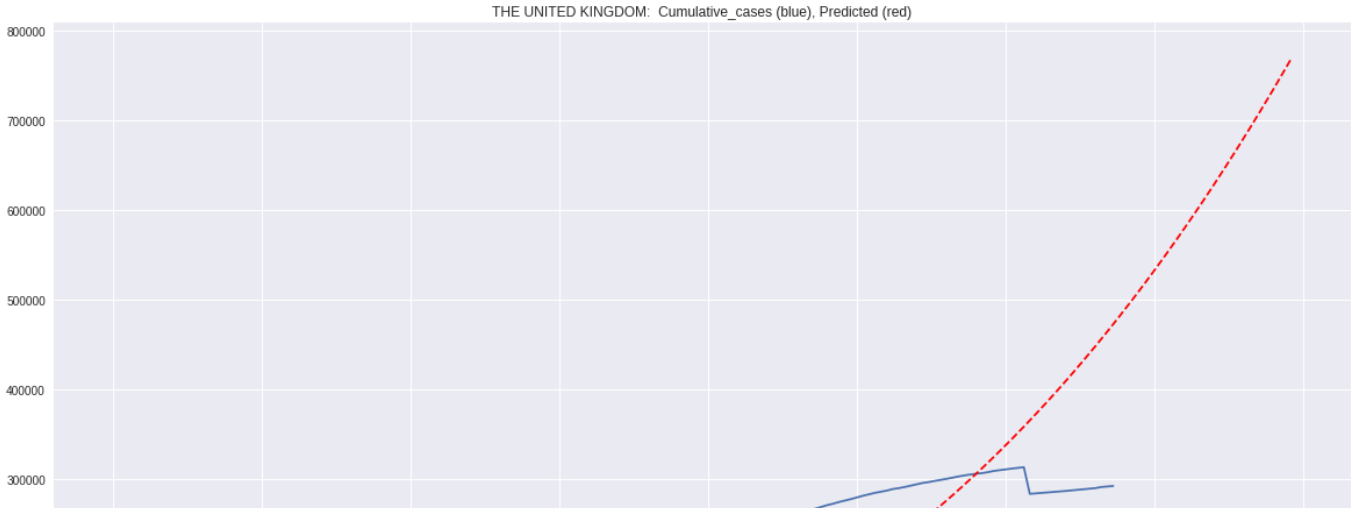

800000
600000
400000
200000
0
0
10
20
30
40
50
60

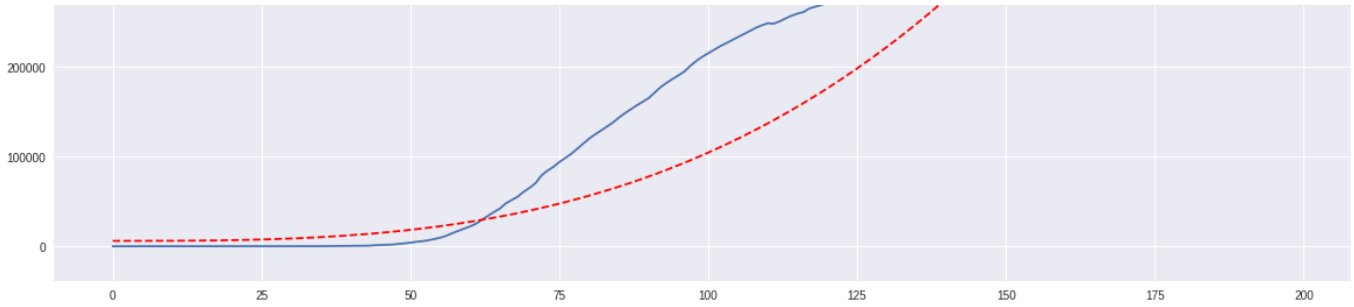

RUSSIAN FEDERATION: Cumulative_cases (blue), Predicted (red)
1e6
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
0
25
50
75
100
125
150
175
200

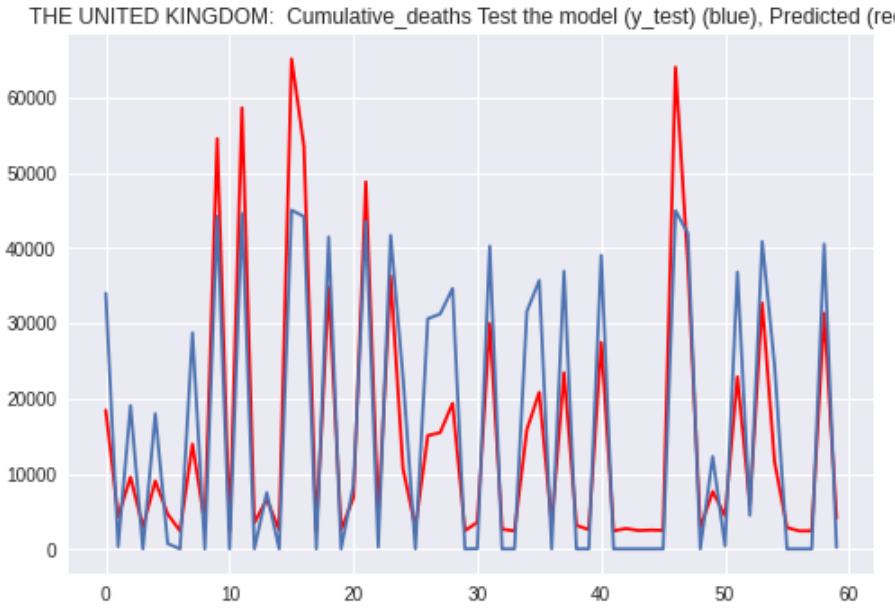

RUSSIAN FEDERATION: Cumulative_deaths Test the model (y_test) (blue), Predicted (red)
12000
10000
8000
6000
4000
2000
0
0
10
20
30
40
50
60

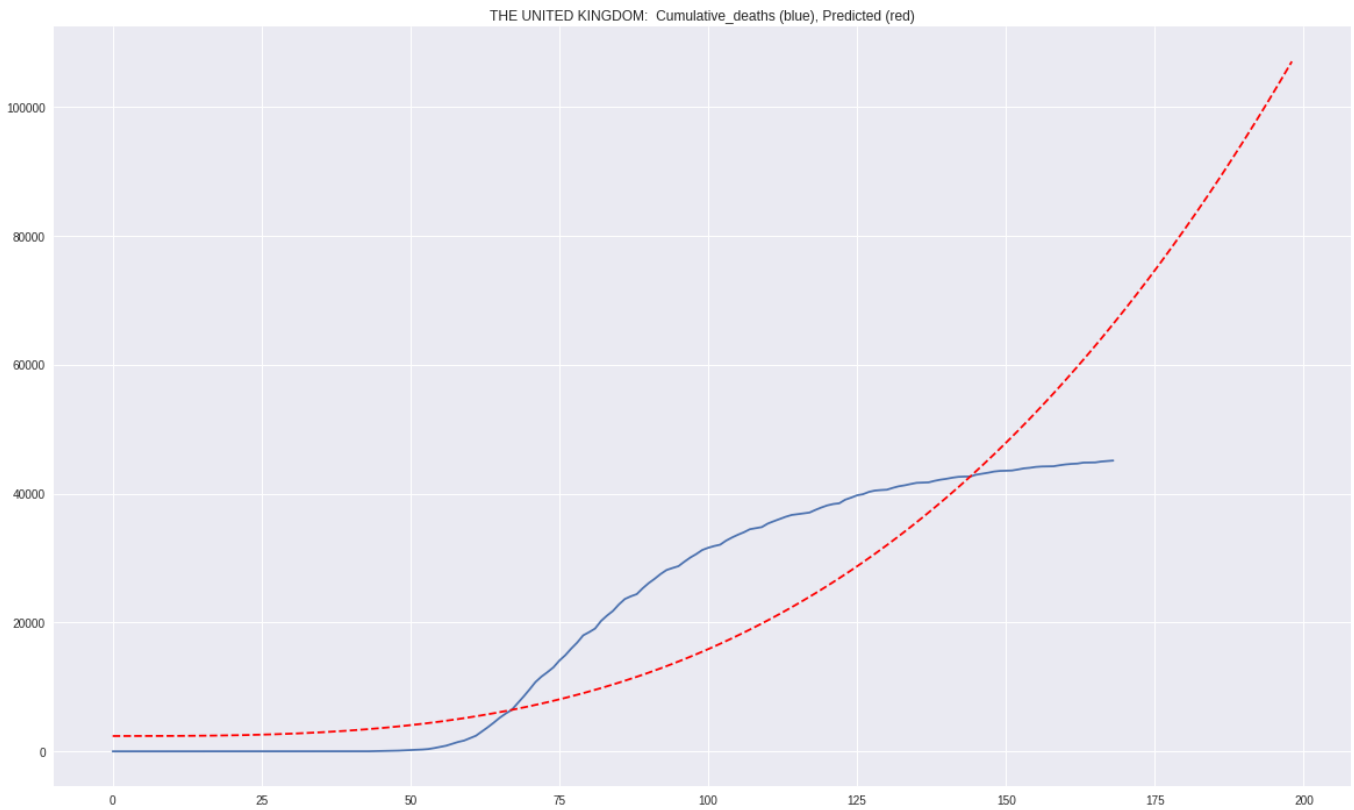

RUSSIAN FEDERATION: Cumulative_deaths (blue), Predicted (red)
20000
15000
10000
5000
0
0
25
50
75
100
125
150
175
200

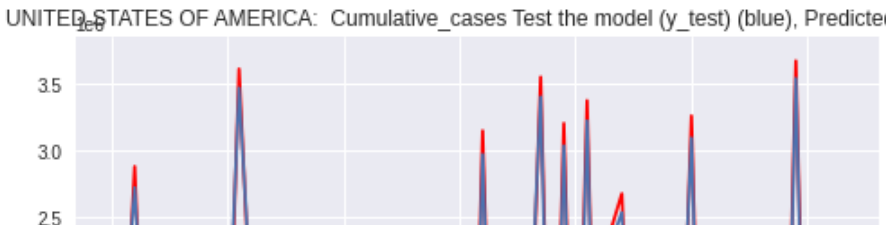

THE UNITED KINGDOM: Cumulative_cases Test the model (y_test) (blue), Predicted (red)
400000
300000
200000
100000
0
0
10
20
30
40
50
60

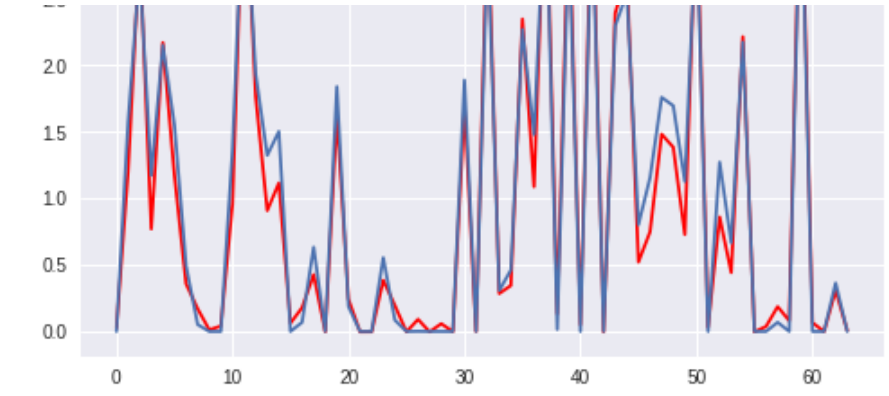

THE UNITED KINGDOM: Cumulative_cases (blue), Predicted (red)
800000
700000
600000
500000
400000
300000

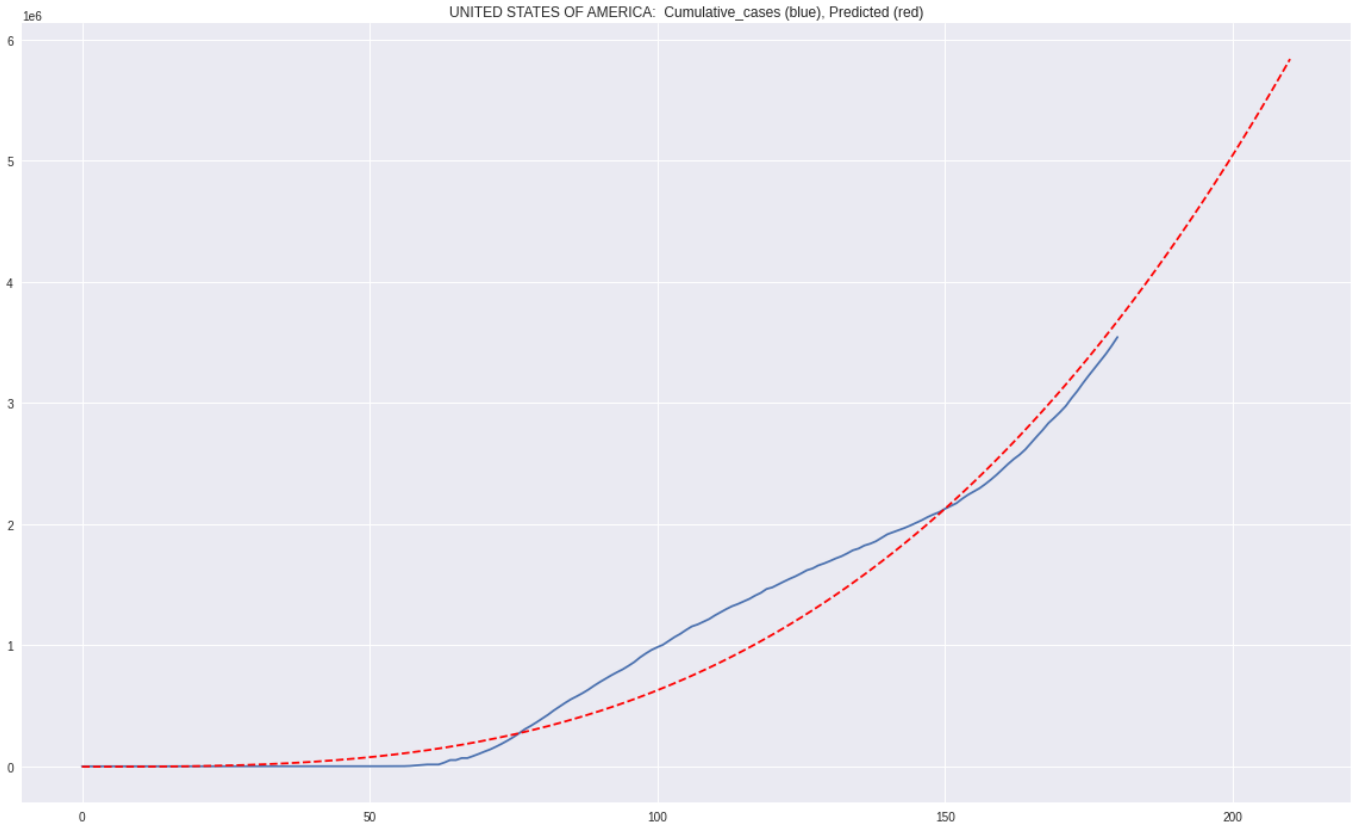

200000
100000
0
0
25
50
75
100
125
150
175
200

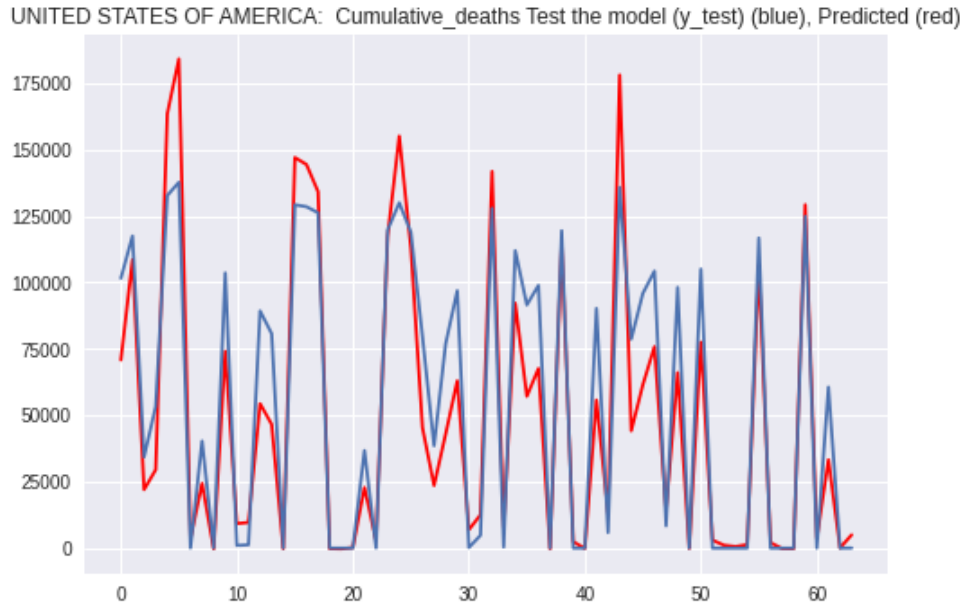

THE UNITED KINGDOM: Cumulative_deaths Test the model (y_test) (blue), Predicted (red)
60000
50000
40000
30000
20000
10000
0
0
10
20
30
40
50
60

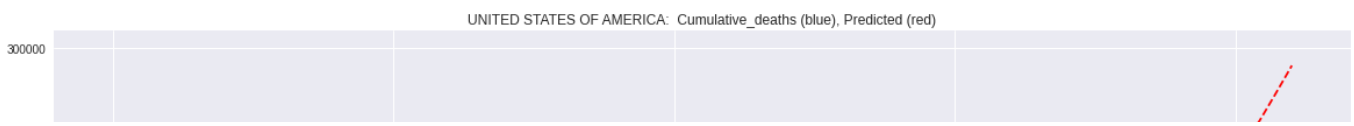

THE UNITED KINGDOM: Cumulative_deaths (blue), Predicted (red)
100000
80000
60000
40000
20000
0
0
25
50
75
100
125
150
175
200


UNITED STATES OF AMERICA: Cumulative_cases Test the model (y_test) (blue), Predicted (red)
1e6
3.5
3.0
2.5


2.0
1.5
1.0
0.5
0.0
0
10
20
30
40
50
60


UNITED STATES OF AMERICA: Cumulative_cases (blue), Predicted (red)
1e6
6
5
4
3
2
1
0
0
50
100
150
200


UNITED STATES OF AMERICA: Cumulative_deaths Test the model (y_test) (blue), Predicted (red)
175000
150000
125000
100000
75000
50000
25000
0
0
10
20
30
40
50
60


UNITED STATES OF AMERICA: Cumulative_deaths (blue), Predicted (red)
300000