Team

November 10, 2023

1 Develop Linear and Non-Linear (polynomial with degree n) regression models for predicting cases and deaths in US

[1]:
```
import pandas as pd
import numpy as np
from IPython.display import Image
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
from sklearn.metrics import mean_squared_error
import plotly.express as px
import plotly.graph_objects as go
```

1.0.1 Start from 2020.06.01 (Monday) to 2021.01.03 (Sunday) of infections in US. X-Axis - number of days, Y-Axis - number of new cases and deaths

[2]:
```
cases = pd.read_csv('../covid_confirmed_usafacts.csv')
cases.head()
```

[2]:

	countyFIPS	County Name	State	StateFIPS	2020-01-22	2020-01-23	\
0	0	Statewide Unallocated	AL	1	0	0	
1	1001	Autauga County	AL	1	0	0	
2	1003	Baldwin County	AL	1	0	0	
3	1005	Barbour County	AL	1	0	0	
4	1007	Bibb County	AL	1	0	0	

	2020-01-24	2020-01-25	2020-01-26	2020-01-27	…	2023-07-14	\
0	0	0	0	0	…	0	
1	0	0	0	0	…	19913	
2	0	0	0	0	…	70521	
3	0	0	0	0	…	7582	
4	0	0	0	0	…	8149	

	2023-07-15	2023-07-16	2023-07-17	2023-07-18	2023-07-19	2023-07-20	\
0	0	0	0	0	0	0	
1	19913	19913	19913	19913	19913	19913	
2	70521	70521	70521	70521	70521	70521	
3	7582	7582	7582	7582	7582	7582	

```
4        8149        8149        8149        8149        8149        8149

   2023-07-21  2023-07-22  2023-07-23
0           0           0           0
1       19913       19913       19913
2       70521       70521       70521
3        7582        7582        7582
4        8149        8149        8149

[5 rows x 1269 columns]
```

[3]:
```
selected_date_columns = [col for col in cases.columns if '2020-06-01' <= col <=␣
↪'2021-01-03']
cases = cases[selected_date_columns].diff(axis=1).dropna(axis=1)
cases.head()
```

[3]:
```
   2020-06-02  2020-06-03  2020-06-04  2020-06-05  2020-06-06  2020-06-07  \
0           0           0           0           0           0           0
1           5           1           2           7          11           6
2           0           0           1           3           8           9
3           3           2           0           6           7           3
4           0           0           0           0           1           0

   2020-06-08  2020-06-09  2020-06-10  2020-06-11  …  2020-12-25  \
0           0           0           0           0  …           0
1           7          10          13          17  …          48
2           7           5           6          12  …         145
3           4           2           9           6  …           6
4           2           6           4           4  …          14

   2020-12-26  2020-12-27  2020-12-28  2020-12-29  2020-12-30  2020-12-31  \
0           0           0           0           0           0           0
1           9          30          36          40          59          26
2          42         117         137         210         220         209
3           2           8          11          45          30          22
4          14           7          16          30          25          17

   2021-01-01  2021-01-02  2021-01-03
0           0           0           0
1          49          29          37
2         222         132         109
3           3          11           2
4          20           9          19

[5 rows x 216 columns]
```

[4]:
```
deaths = pd.read_csv('../covid_deaths_usafacts.csv')
deaths.head()
```

[4]:

	countyFIPS	County Name	State	StateFIPS	2020-01-22	2020-01-23	2020-01-24	2020-01-25	2020-01-26	2020-01-27	…	2023-07-14	2023-07-15	2023-07-16	2023-07-17	2023-07-18	2023-07-19	2023-07-20	2023-07-21	2023-07-22	2023-07-23
0	0	Statewide Unallocated	AL	1	0	0	0	0	0	0	…	0	0	0	0	0	0	0	0	0	0
1	1001	Autauga County	AL	1	0	0	0	0	0	0	…	235	235	235	235	235	235	235	235	235	235
2	1003	Baldwin County	AL	1	0	0	0	0	0	0	…	731	731	731	731	731	731	731	731	731	731
3	1005	Barbour County	AL	1	0	0	0	0	0	0	…	104	104	104	104	104	104	104	104	104	104
4	1007	Bibb County	AL	1	0	0	0	0	0	0	…	111	111	111	111	111	111	111	111	111	111

[5 rows x 1269 columns]

[5]:
```
selected_date_columns = [col for col in deaths.columns if '2020-06-01' <= col
↪<= '2021-01-03']
deaths= deaths[selected_date_columns].diff(axis=1).dropna(axis=1)
deaths.head()
```

[5]:

	2020-06-02	2020-06-03	2020-06-04	2020-06-05	2020-06-06	2020-06-07	2020-06-08	2020-06-09	2020-06-10	2020-06-11	…	2020-12-25
0	0	0	0	0	0	0	0	0	0	0	…	0
1	0	0	0	0	0	0						
2	0	0	0	0	0	0						
3	0	0	0	0	0	0						
4	0	0	0	0	0	0						

```
1           0           0           1           0  …           0
2           0           0           0           0  …           0
3           0           0           0           0  …           0
4           0           0           0           0  …           0

   2020-12-26  2020-12-27  2020-12-28  2020-12-29  2020-12-30  2020-12-31  \
0           0           0           0           0           0           0
1           0           1           0           0           1           0
2           0           1           0           4           4           1
3           0           0           0           0           0           0
4           0           0           0           0           4           0

   2021-01-01  2021-01-02  2021-01-03
0           0           0           0
1           2           0           0
2           8           0           0
3           1           0           0
4           0           0           0

[5 rows x 216 columns]
```

[6]:
```
daily_cases = []
for col in cases.columns:
    daily_cases.append(cases[col].sum())
daily_cases_data = pd.Series(daily_cases)
daily_cases_data
```

[6]:
```
0       21795
1       21372
2       21923
3       28884
4       23787
        …
211    282351
212    236987
213    165207
214    232897
215    226866
Length: 216, dtype: int64
```

[7]:
```
daily_deaths = []
for col in deaths.columns:
    daily_deaths.append(deaths[col].sum())
daily_deaths_data = pd.Series(daily_deaths)
daily_deaths_data
```

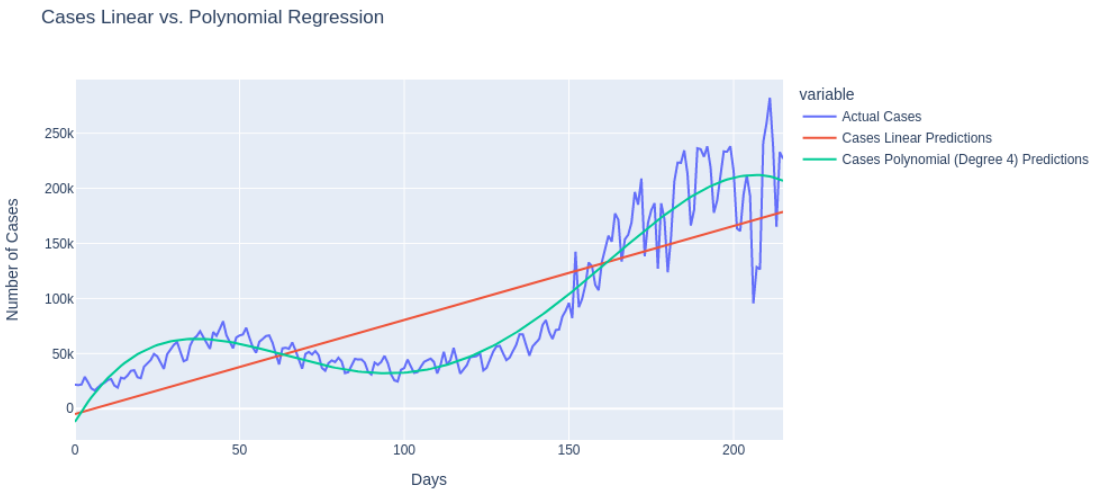
[7]:
```
0      1222
1       988
2       972
3      1095
4       781
       …
211    3547
212    3600
213    2684
214    3452
215    2345
Length: 216, dtype: int64
```

[8]:
```
days = np.arange(len(daily_cases_data))


# Linear Regression for Cases
lr_cases = LinearRegression()
lr_cases.fit(days.reshape(-1, 1), daily_cases_data)
cases_linear_predictions = lr_cases.predict(days.reshape(-1, 1))

# Linear Regression for Deaths
lr_deaths = LinearRegression()
lr_deaths.fit(days.reshape(-1, 1), daily_deaths_data)
deaths_linear_predictions = lr_deaths.predict(days.reshape(-1, 1))

degree = 4
poly = PolynomialFeatures(degree=degree)
X_poly = poly.fit_transform(days.reshape(-1, 1))

# Polynomial Regression for Cases
pr_cases = LinearRegression()
pr_cases.fit(X_poly, daily_cases_data)
cases_poly_predictions = pr_cases.predict(X_poly)

# Polynomial Regression for Deaths
pr_deaths = LinearRegression()
pr_deaths.fit(X_poly, daily_deaths_data)
deaths_poly_predictions = pr_deaths.predict(X_poly)

df = pd.DataFrame({'Days': days,
                   'Actual Cases': daily_cases_data,
                   'Cases Linear Predictions': cases_linear_predictions,
                   f'Cases Polynomial (Degree {degree}) Predictions':␣
 ↪cases_poly_predictions,
                   'Actual Deaths': daily_deaths_data,
                   'Deaths Linear Predictions': deaths_linear_predictions,
```

```
                    f'Deaths Polynomial (Degree {degree}) Predictions':␣
 ↪deaths_poly_predictions})

fig_cases = px.line(df, x='Days', y=['Actual Cases', 'Cases Linear Predictions',
                                     f'Cases Polynomial (Degree {degree})␣
 ↪Predictions'],
                    labels={'value': 'Number of Cases'}, title='Cases Linear vs.
 ↪ Polynomial Regression')
fig_cases.update_layout(width=1000, height=500)
fig_deaths = px.line(df, x='Days', y=['Actual Deaths', 'Deaths Linear␣
 ↪Predictions',
                                      f'Deaths Polynomial (Degree {degree})␣
 ↪Predictions'],
                     labels={'value': 'Number of Deaths'}, title='Deaths Linear␣
 ↪vs. Polynomial Regression')
fig_deaths.update_layout(width=1000, height=500)

fig_cases.write_image("fig_cases.png")
fig_deaths.write_image("fig_deaths.png")
```

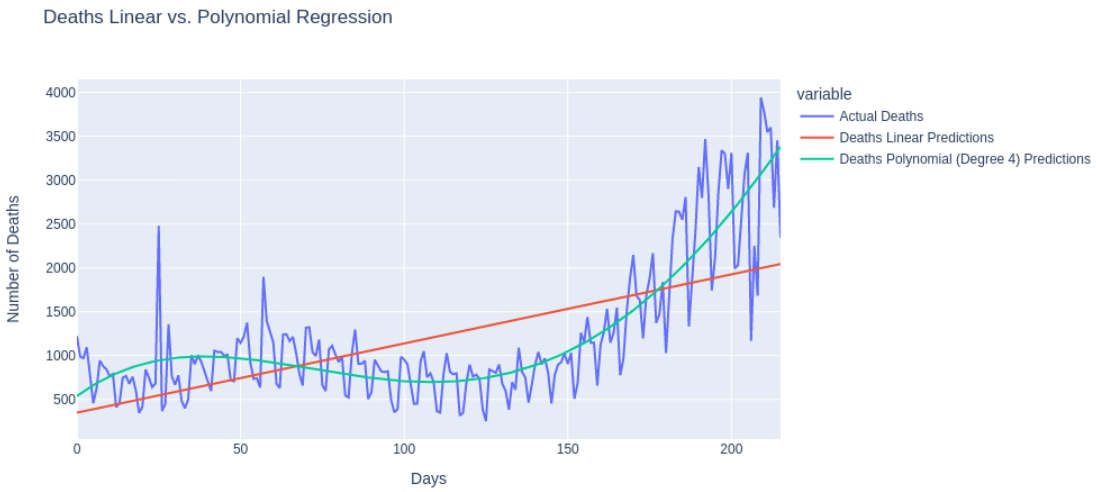
[9]:
```
Image(filename="fig_cases.png")
```

[9]:



[10]:
```
Image(filename="fig_deaths.png")
```

[10]:



[11]:
```
# Calculate Root Mean Square Error (RMSE) for each model
rmse_linear_cases = np.sqrt(mean_squared_error(daily_cases_data,␣
 ↪cases_linear_predictions))
rmse_linear_deaths = np.sqrt(mean_squared_error(daily_deaths_data,␣
 ↪deaths_linear_predictions))
rmse_poly_cases = np.sqrt(mean_squared_error(daily_cases_data,␣
 ↪cases_poly_predictions))
rmse_poly_deaths = np.sqrt(mean_squared_error(daily_deaths_data,␣
 ↪deaths_poly_predictions))

print(f'RMSE for Cases (Linear): {rmse_linear_cases:.2f}')
print(f'RMSE for Deaths (Linear): {rmse_linear_deaths:.2f}\n')
print(f'RMSE for Cases (Polynomial Degree {degree}): {rmse_poly_cases:.2f}')
print(f'RMSE for Deaths (Polynomial Degree {degree}): {rmse_poly_deaths:.2f}')
```

```
RMSE for Cases (Linear): 40066.91
RMSE for Deaths (Linear): 611.96

RMSE for Cases (Polynomial Degree 4): 21624.12
RMSE for Deaths (Polynomial Degree 4): 401.60
```

[12]:
```
deaths = pd.read_csv('../covid_deaths_usafacts.csv')
cases = pd.read_csv('../covid_confirmed_usafacts.csv')
selected_date_columns = [col for col in cases.columns if '2020-06-01' <= col <=␣
 ↪'2021-01-10']
cases = cases[selected_date_columns].diff(axis=1).dropna(axis=1)
deaths= deaths[selected_date_columns].diff(axis=1).dropna(axis=1)
```

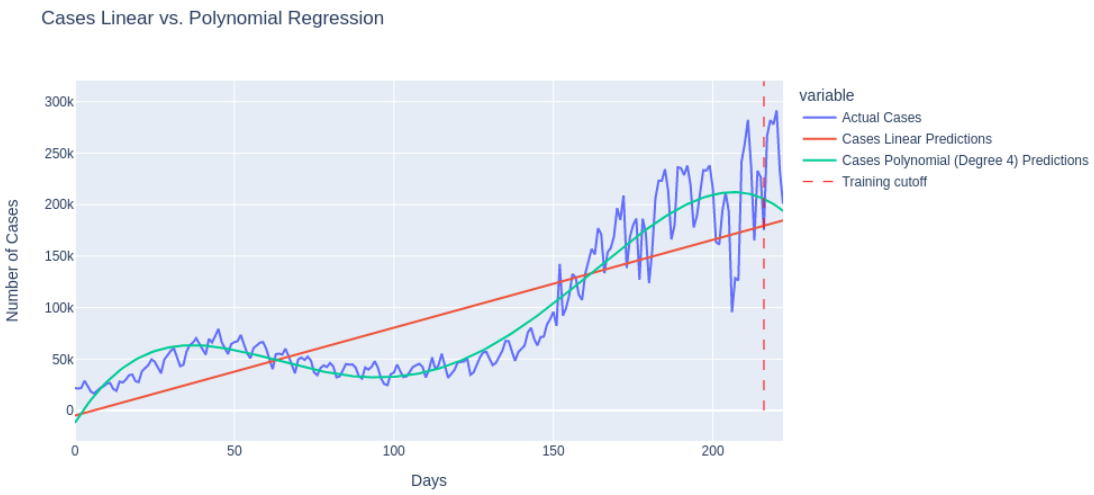
```
daily_deaths = []
for col in deaths.columns:
    daily_deaths.append(deaths[col].sum())
daily_deaths_data = pd.Series(daily_deaths)
daily_deaths_data

daily_cases = []
for col in cases.columns:
    daily_cases.append(cases[col].sum())
daily_cases_data = pd.Series(daily_cases)
daily_cases_data


days = np.arange(len(daily_cases_data))
print(len(days))

# Linear Regression
cases_linear_predictions = lr_cases.predict(days.reshape(-1, 1))
deaths_linear_predictions = lr_deaths.predict(days.reshape(-1, 1))

degree = 4
poly = PolynomialFeatures(degree=degree)
X_poly = poly.fit_transform(days.reshape(-1, 1))

# Polynomial Regression for Cases
cases_poly_predictions = pr_cases.predict(X_poly)

# Polynomial Regression for Deaths
deaths_poly_predictions = pr_deaths.predict(X_poly)

df = pd.DataFrame({'Days': days,
                   'Actual Cases': daily_cases_data,
                   'Cases Linear Predictions': cases_linear_predictions,
                   f'Cases Polynomial (Degree {degree}) Predictions':␣
 ↪cases_poly_predictions,
                   'Actual Deaths': daily_deaths_data,
                   'Deaths Linear Predictions': deaths_linear_predictions,
                   f'Deaths Polynomial (Degree {degree}) Predictions':␣
 ↪deaths_poly_predictions})

fig_cases = px.line(df, x='Days', y=['Actual Cases', 'Cases Linear Predictions',
                                     f'Cases Polynomial (Degree {degree})␣
 ↪Predictions'],
                    labels={'value': 'Number of Cases'}, title='Cases Linear vs.
 ↪ Polynomial Regression')
fig_cases.update_layout(width=1000, height=500)
```

```
fig_deaths = px.line(df, x='Days', y=['Actual Deaths', 'Deaths Linear␣
↪Predictions',
                                      f'Deaths Polynomial (Degree {degree})␣
↪Predictions'],
                     labels={'value': 'Number of Deaths'}, title='Deaths Linear␣
↪vs. Polynomial Regression')
fig_deaths.update_layout(width=1000, height=500)

fig_deaths.add_shape(dict(type='line', x0=216, x1=216, y0=0, y1=4500,␣
↪line=dict(color='red', width=1, dash='dash')))
fig_cases.add_shape(dict(type='line', x0=216, x1=216, y0=0, y1=320000,␣
↪line=dict(color='red', width=1, dash='dash')))
fig_deaths.add_trace(go.Scatter(x=[216, 216], y=[0, 10], mode='lines',␣
↪name='Training cutoff', line=dict(color='red', width=1, dash='dash')))
fig_cases.add_trace(go.Scatter(x=[216, 216], y=[0, 10], mode='lines',␣
↪name='Training cutoff', line=dict(color='red', width=1, dash='dash')))

fig_cases.write_image("fig_cases_prediction.png")
fig_deaths.write_image("fig_deaths_prediction.png")
```

223

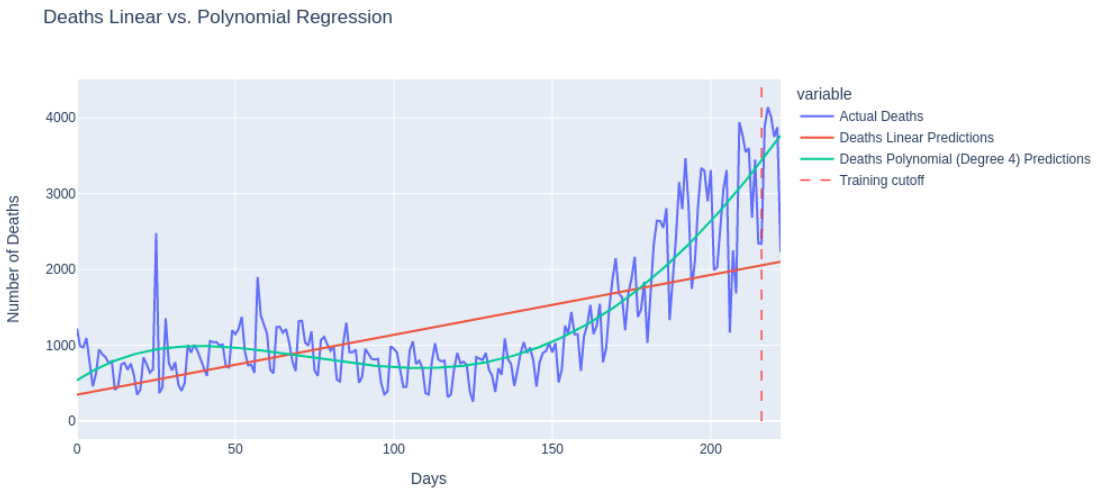
[13]:
```
Image(filename="fig_cases_prediction.png")
```

[13]:



[14]:
```
Image(filename="fig_deaths_prediction.png")
```

[14]:



We can see that in both cases the linear regression is worse than the polynomial which is to be expected. However, both prediction are difficult to judge since there is a lot of varience in the data, espcially over one week. For the deaths, the polynomail will over shoot if we continue the prediction. For the cases, the polynomail does predicting the fall in daily numbers correctly, if we expand the prediction.

Other countries expireenced similar trends a at the beginning of Janaury, as numbers where starting to decrease worldwide around that time. as seen here:

https://www.who.int/docs/default-source/coronaviruse/situation-reports/20210202_Weekly_Epi_Update_25.pdf

[0]: