Tugas 1 (Regresi Non-Linear)

Herlina Anwar - D082222026

Penjelasan Kode

Import library

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

Langkah pertama yang dilakukan yaitu meng-import library yang dibutuhkan seperti gambar diatas.

Import pandas ad pd . Library Pandas merupakan library open source dalam bahasa pemograman python yang befungsi untuk memproses data, mulai dari pembersihan data, manipulasi data, hingga melakukan analisis data.

Import numpy as pd. Library Numpy (numerical Python) merupakan library python yang menyediakan fungsi yang siap dipakai untuk memudahkan kita melakukan perhitungan ssaintifik seperti matrix, aljabara, statistik dan sebagainya.

Import matplotlib.pyplot as plt. Matplotlib digunakan untuk melakukan visualisasi data secara 2D ataupun 3D yang dapat disimpan dengan format gambar seperti JPEG, JPG, dan PNG.

%matplotlib inline digunaan untuk meng-embed gambar plot statis didalam program.

Membaca Dataset

```
#membaca dataset pada direktori path
path = '/content/china_gdp.csv'
df = pd.read_csv(path)
```

Setelah itu, melakukan proses membaca file (*read*) berformat .csv yang berisi data. Yang disimpan pada direktori. Proses pembacaan file .csv dideklarasikan sebagai *df*.

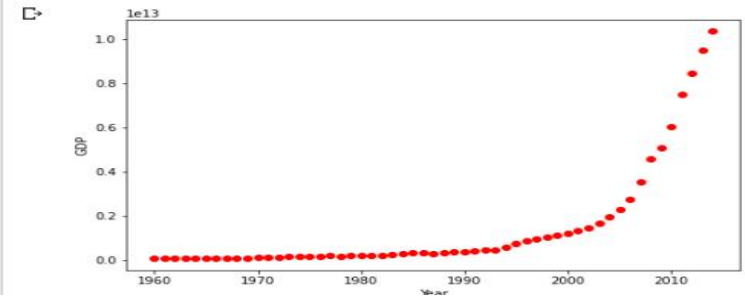
```
# Menampilkan 10 data teratas
df.head(10)
```

	Year	Value
0	1960	5.918412e+10
1	1961	4.955705e+10
2	1962	4.668518e+10
3	1963	5.009730e+10
4	1964	5.906225e+10
5	1965	6.970915e+10
6	1966	7.587943e+10
7	1967	7.205703e+10
8	1968	6.999350e+10
9	1969	7.871882e+10

Kemudian menggunakan fungsi **head()** yang merupakan fungsi untuk mendapatkan 10 data dari batas teratas seperti gambar diatas.

Menampilkan Ploting Dataset

```
# Plotting dataset
x_data, y_data = (df["Year"].values, df["Value"].values)

plt.figure (figsize = (8, 5))
plt.plot(x_data, y_data, 'ro')
plt.ylabel("GDP")
plt.xlabel("Year")
plt.show()
```



Setelah itu melakukan eksplorasi data dengan menampilkan plotting atribute year dan value (GDP), seperti gambar diatas. Kemudian membuat model sigmoid dengan melakukan normalisasi data dan menentukan nilai Beta_1 dan Beta_2, Selanjutnya melakukan prediksi model.

```
# Membuat Model

def model_sigmoid(x, Beta_1, Beta_2):
    ymodel = 1 / (1 + np.exp (-Beta_1 * (x - Beta_2)))
    return ymodel
```

```
[ ] # Melakukan Normalisasi Data

    x_scaled = (x_data - min (x_data)) / (max(x_data) - min(x_data))
    y_scaled = (y_data - min (y_data)) / (max(y_data) - min(y_data))
```

```
[ ] # Menentukan Nilai Beta_1 dan Beta_2

    from scipy.optimize import curve_fit
    popt, pcov = curve_fit(model_sigmoid, x_scaled, y_scaled)
    print("Beta_1 = {} Beta_2 = {}".format(popt[0], popt[1]))
```

```
Beta_1 = 18.869350407536267 Beta_2 = 0.8966631945557113
```

```
# Hitung y_prediksi normalisasi

y_prediksi= model_sigmoid(x_scaled, *popt)
```

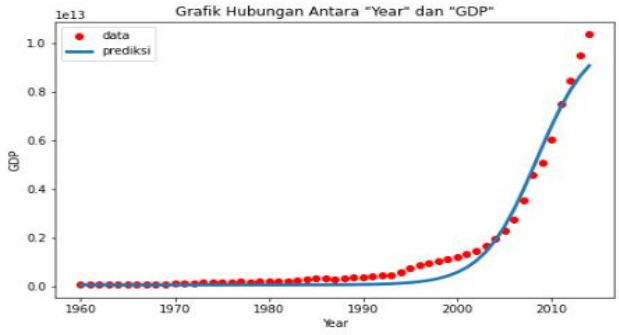
1. **Tambahkan evaluasi model dengan menghitung nilai *Mean Absolute Error* (MAE), *Mean Squared Error* (MSE), *Root Mean Squared Error* (RMSE)**

```
[ ] #menghitung MAE, MSE, dan RMSE
    mae = np.mean(np.absolute(y_scaled - y_prediksi))
    mse = np.mean((y_scaled - y_prediksi)**2)
    rmse = np.sqrt(mse)

    print("Mean Absolute Error = " + str(round(mae, 4)))
    print("Mean Squared Error = " + str(round(mse, 4)))
    print("Root Mean Squared Error = " + str(round(rmse, 4)))
```

```
Mean Absolute Error = 0.0273
Mean Squared Error = 0.0014
Root Mean Squared Error = 0.0376
```

2. **Tambahkan grafik yang menunjukkan hubungan antara "Year" dan "GDP" dengan nilai yang sebenarnya (tidak dinormalisasi)**

```
# Menambahkan grafik yang menunjukkan hubungan antara "Year" dan "GDP" dengan nilai yang sebenarnya (tidak dinormalisasi)

y_prediksi2 = (y_prediksi*(max(y_data)-min(y_data)))+min(y_data)
plt.figure(figsize=(8, 5))
plt.plot(x_data, y_data, 'ro', label = 'data')
plt.plot(x_data, y_prediksi2, linewidth = 3.0, label = 'prediksi')
plt.legend(loc = 'best')
plt.ylabel("GDP")
plt.xlabel("Year")
plt.title('Grafik Hubungan Antara "Year" dan "GDP"')
plt.show()
```



3. Tambahkan grafik yang menunjukkan prediksi GDP untuk tahun 2015 – 2030 menggunakan nilai yang sebenarnya (tidak dinormalisasi)

```
x_uji = np.arange(2015, 2031)
x_uji_scaled = (x_uji-min(x_data))/(max(x_data)-min(x_data))
y_prediksi3 = model_sigmoid(x_uji_scaled, *popt)
y_prediksi4 = (y_prediksi3*(max(y_data)-min(y_data)))+min(y_data)
plt.figure(figsize=(8, 5))
plt.plot(x_uji, y_prediksi4, linewidth = 3.0, label = 'prediksi')
plt.legend(loc = 'best')
plt.ylabel("GDP")
plt.xlabel("Year")
plt.title('Grafik Hubungan Antara "Year" dan "GDP" Untuk Tahun 2015 - 2030')
plt.show()
```