Nama : Firman Gani Heriansyah

NIM : 21120122130043

Kelas : C

Metode Linear dan Metode Pangkat Sederhana

Link GitHub : https://github.com/Frmngh/Tugas-Metnum-Regresi

Source Code:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
import time

# Load data
data = pd.read_csv('Student_Performance.csv')

# Extract relevant columns
TB = data['Hours Studied'].values
NT = data['Performance Index'].values

# Model Linear (Metode 1)
def linear_regression(TB, NT):
    model = LinearRegression()
    TB = TB.reshape(-1, 1)
    model.fit(TB, NT)
    NT_pred = model.predict(TB)
    return model, NT_pred

# Model Pangkat Sederhana (Metode 2)
def power_law(TB, NT):
    # Transform data for linearization
    log_TB = np.log(TB)
    log_NT = np.log(NT)

    # Fit linear model to transformed data
    model = LinearRegression()
    log_TB = log_TB.reshape(-1, 1)
    model.fit(log_TB, log_NT)
    log_NT_pred = model.predict(log_TB)

    # Transform predictions back to original scale
    NT_pred = np.exp(log_NT_pred)
    return model, NT_pred

# Calculate RMS error
def calculate_rms_error(NT, NT_pred):
    return np.sqrt(mean_squared_error(NT, NT_pred))

# Plot results
```

```
def plot_results(TB, NT, NT_pred_linear, NT_pred_power):
    plt.figure(figsize=(14, 6))

    # Scatter plot of actual data
    plt.subplot(1, 2, 1)
    plt.scatter(TB, NT, color='blue', label='Actual Data')
    plt.plot(TB, NT_pred_linear, color='red', label='Linear Fit')
    plt.xlabel('Sample Question Papers Practiced')
    plt.ylabel('Performance Index')
    plt.title('Linear Regression')
    plt.legend()

    # Scatter plot of actual data
    plt.subplot(1, 2, 2)
    plt.scatter(TB, NT, color='blue', label='Actual Data')
    plt.plot(TB, NT_pred_power, color='green', label='Power Law Fit')
    plt.xlabel('Sample Question Papers Practiced')
    plt.ylabel('Performance Index')
    plt.title('Power Law Regression')
    plt.legend()

    plt.show()

# Run analysis
start_time = time.time()
linear_model, NT_pred_linear = linear_regression(TB, NT)
linear_time = time.time() - start_time
linear_rms_error = calculate_rms_error(NT, NT_pred_linear)

start_time = time.time()
power_model, NT_pred_power = power_law(TB, NT)
power_time = time.time() - start_time
power_rms_error = calculate_rms_error(NT, NT_pred_power)

# Print results
print(f"Linear Model RMS Error: {linear_rms_error}, Execution Time:
{linear_time} seconds")
print(f"Power Law Model RMS Error: {power_rms_error}, Execution Time:
{power_time} seconds")

# Plot results
plot_results(TB, NT, NT_pred_linear, NT_pred_power)
```

Konsep:

Kode ini membandingkan dua model regresi untuk menganalisis hubungan antara jumlah latihan soal yang dikerjakan (jam belajar) dan performa indeks siswa. Dua model yang dibandingkan adalah regresi linier dan regresi pangkat sederhana. Regresi linier mengasumsikan hubungan linear antara jam belajar dan performa, sedangkan regresi pangkat sederhana mengasumsikan hubungan non-linear yang berbentuk pangkat antara jam belajar dan performa.

Alur Kode:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
import time

# Load data
data = pd.read_csv('Student_Performance.csv')

# Extract relevant columns
TB = data['Hours Studied'].values
NT = data['Performance Index'].values
```

Pada proses pertama, mengimpor modul-modul yang diperlukan seperti numpy, pandas, matplotlib, dan sklearn. Kemudian, data dari `'Student_Performance.csv'` dimasukkan ke dalam data frame pandas. Nantinya, dari data tersebut, kolom `'Hours Studied'` dan `'Performance Index'` akan diekstraksi dan disimpan ke dalam vaaariabel TB dan NT sebagai array numpy.

```
# Model Linear (Metode 1)
def linear_regression(TB, NT):
    model = LinearRegression()
    TB = TB.reshape(-1, 1)
    model.fit(TB, NT)
    NT_pred = model.predict(TB)
    return model, NT_pred

# Model Pangkat Sederhana (Metode 2)
def power_law(TB, NT):
    # Transform data for linearization
    log_TB = np.log(TB)
    log_NT = np.log(NT)

    # Fit linear model to transformed data
    model = LinearRegression()
    log_TB = log_TB.reshape(-1, 1)
    model.fit(log_TB, log_NT)
    log_NT_pred = model.predict(log_TB)

    # Transform predictions back to original scale
    NT_pred = np.exp(log_NT_pred)
    return model, NT_pred
```

Fungsi pertama pada metode linear, menerima dua parameter yaitu TB dan NT Dimana TB akan diubah menjadi array 2D, dan model linear dilatih dengan TB sebagai fitur dan NT sebagai target. `return model, NT_pred` prediksi NT akan dihasilkan menggunakan model yang telah dilatih. Berbeda dengan model pangkat sederhana yang mentransformasikan data TB dan NT

menggunakan logaritma natural untuk linearisasi. Model regresi linear akan dilatih pada data yang dtransformasikan, dan pada formula `NT_pred = np.exp(log_NT_pred)` nilai prediksi akan dihasilkan dalam skala logaritmik, serta akan dikembalikan ke dalam skala asli dengan eksponensial.

```
# Calculate RMS error
def calculate_rms_error(NT, NT_pred):
    return np.sqrt(mean_squared_error(NT, NT_pred))

# Plot results
def plot_results(TB, NT, NT_pred_linear, NT_pred_power):
    plt.figure(figsize=(14, 6))

    # Scatter plot of actual data
    plt.subplot(1, 2, 1)
    plt.scatter(TB, NT, color='blue', label='Actual Data')
    plt.plot(TB, NT_pred_linear, color='red', label='Linear Fit')
    plt.xlabel('Sample Question Papers Practiced')
    plt.ylabel('Performance Index')
    plt.title('Linear Regression')
    plt.legend()

    # Scatter plot of actual data
    plt.subplot(1, 2, 2)
    plt.scatter(TB, NT, color='blue', label='Actual Data')
    plt.plot(TB, NT_pred_power, color='green', label='Power Law Fit')
    plt.xlabel('Sample Question Papers Practiced')
    plt.ylabel('Performance Index')
    plt.title('Power Law Regression')
    plt.legend()

    plt.show()
```

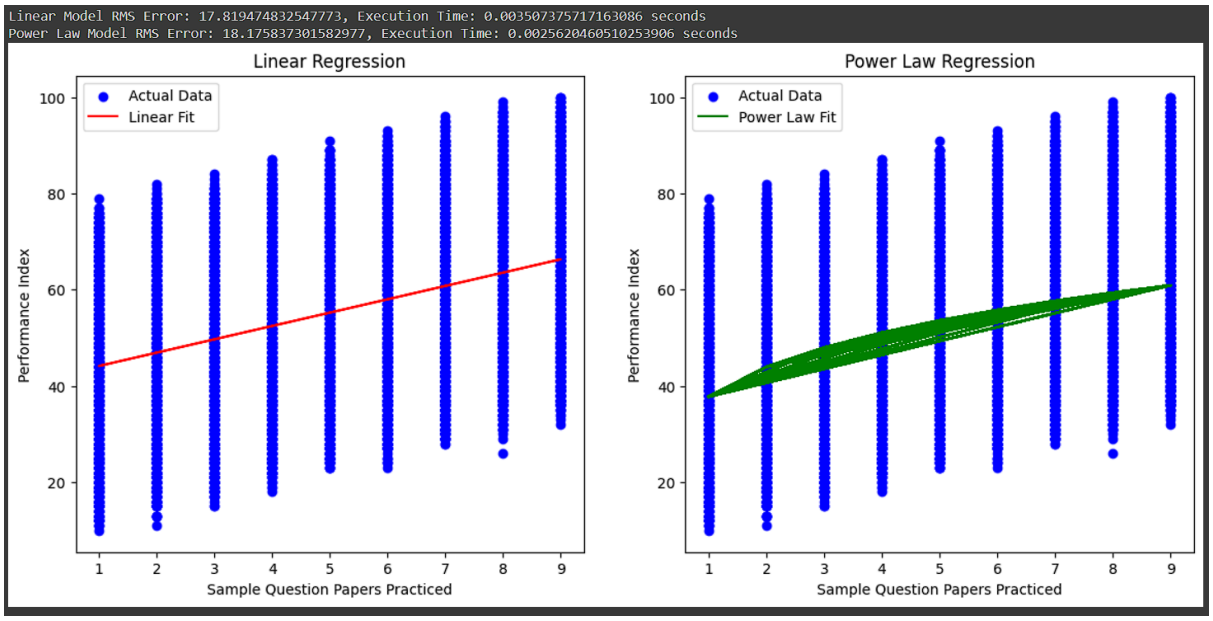
Kode tersebut menggunakan dua fungsi. Fungsi pertama yaitu `calculate_rms_error` untuk menghitung galat RMS antara NT dan NT_pred. Data NT akan diplot dengan TB bersama dengan garis linear `NT_pred_linear`. Pada plot results, data NT akan diplot lagi dengan TB dengan garis power law `NT_pred_power`. Nantinya, kedua fungsi tersebut akan menampilkan hasil berupa label, judul, untuk visualisasi dan perbandingan nilai.

```
# Run analysis
start_time = time.time()
linear_model, NT_pred_linear = linear_regression(TB, NT)
linear_time = time.time() - start_time
linear_rms_error = calculate_rms_error(NT, NT_pred_linear)

start_time = time.time()
power_model, NT_pred_power = power_law(TB, NT)
power_time = time.time() - start_time
power_rms_error = calculate_rms_error(NT, NT_pred_power)

# Print results
```

```
print(f"Linear Model RMS Error: {linear_rms_error}, Execution Time:
{linear_time} seconds")
print(f"Power Law Model RMS Error: {power_rms_error}, Execution Time:
{power_time} seconds")

# Plot results
plot_results(TB, NT, NT_pred_linear, NT_pred_power)
```

Pada langkah terakhir, `time.time` digunakan untuk mencatat regresi linear yang kemudian akan dibangun dan prediksi nilai dihasilkan. Menghitung waktu eksekusi dengan menggunakan formula `time.time() - start_time`. Pada kode tersebut, galat RMS juga dihitung. Sama hal nya pada fungsi kedua. Setelah itu, hasil kedua model akan dicetak, dan grafik kedua model akan diplot untuk membandingkan prediksi dan data asli dengan formula `print(f"Linear Model RMS Error: {linear_rms_error}, Execution Time: {linear_time} seconds")`.

Analisis Hasil:



Grafik di sebelah kiri menampilkan hasil regresi linier, di mana garis merah menunjukkan fit linear terhadap data aktual yang ditampilkan dalam titik-titik biru. Hasilnya menunjukkan bahwa model linier memberikan garis lurus yang sedikit miring ke atas, menunjukkan bahwa peningkatan kinerja yang linear namun relatif lemah dengan meningkatnya jumlah latihan soal. Grafik tesebut juga mencantumkan galat RMS (RMS Error) sebesar 17.819 yang mana menandakan bahwa tingkat kesalahan pada model ini cukup tinggi.

Sedangkan grafik di sebelah kanan menunjukkan hasil model pangkat sederhana, di mana kurva hijau menunjukkan fit model power law terhadap data aktual. Galat RMS untuk model ini adalah 18.176, yang sedikit lebih tinggi daripada model linier, hal ini menunjukkan

bahwa meskipun power law regression memberikan visualisasi yang lebih baik, namun secara kuantitatif model ini tidak secara signifikan lebih akurat daripada model linier berdasarkan galat RMS yang diberikan.

Kseimpulan:

Pada analisis hasil menunjukkan bahwa model Linear dan Pangkat Sederhana memiliki kelebihan dan kekurangannya masing-masing. Model Linear menunjukkan garis fit yang relatif miring ke atas, artinya ada peningkatan kinerja yang linear namun lemah dengan meningkatnya jumlah latihan soal. Model ini tidak sepenuhnya akurat dalam memprediksi data aktual, dibuktikan dengan tingkat kesalahan model ini yang diukur dengan galat RMS adalah sebesar 17.819. Sedangkan, model Pangkat Sederhana emnunjukkan kurva fit yang lebih sesuai dengan data aktual namun tingkat kesalahannya sebesar 18.176 yang mana lebih tinggi daripada model Linear. Ditunjukkan juga bahwa waktu eksekusi pada model Pangkat Sederhana lebih cepat dibandingkan model Linear. Sehingga, dapat disimpulkan bahwa meskipun model Pangkat Sederhana menghasilkan visualisasi yang lebih sesuai, namun secara kuantitatif model ini tidak signifikan lebih akurat dibanding model Linear. Oleh karena itu, penting untuk mempertimbangkan konteks dan kebutuhan spesifik dari analisis yang dilakukan.