**Nama : Dzaki Amri Zaidaan**

**NIM : 21120122130044**

**Kelas : Metode Numerik D**

Program ini dirancang untuk menganalisis dan memprediksi nilai ujian siswa berdasarkan dua variabel utama: durasi waktu belajar (TB) dan jumlah latihan soal yang dipraktikkan (NL). Metode yang digunakan dalam analisis ini adalah regresi linear dan regresi pangkat. Dengan program ini, kita dapat mengevaluasi efektivitas dari kedua metode regresi tersebut dan menentukan mana yang lebih baik dalam memprediksi kinerja siswa.

Link Github : <https://github.com/Dzakiamriz22/Aplikasi-Regresi_Dzaki-Amri-Zaidaan>

Persiapan data

def prepare\_data():

    # Baca data dari CSV

    data = pd.read\_csv('Student\_Performance.csv')

    # Ambil kolom yang relevan

    data = data[['Hours Studied', 'Sample Question Papers Practiced', 'Performance Index']]

    data.columns = ['Durasi Waktu Belajar(TB)', 'Jumlah Latihan Soal(NL)', 'Nilai Ujian Siswa (NL)']

    data.to\_csv('processed\_student\_data.csv', index=False)

    print(data.head())

Fungsi prepare\_data() bertanggung jawab untuk membaca data dari file CSV yang berisi informasi tentang kinerja siswa, memilih kolom yang relevan, mengubah nama kolom ke dalam bahasa Indonesia, dan menyimpan data yang telah diproses ke file baru.

Pembagian data

def load\_data():

    # Baca data yang sudah diproses

    data = pd.read\_csv('processed\_student\_data.csv')

    return data

Fungsi load\_data() digunakan untuk membaca data yang telah diproses dan disimpan dalam file CSV baru.

Regresi Linear

def linear\_regression(X\_train, y\_train, X\_test):

    linear\_model = LinearRegression()

    linear\_model.fit(X\_train, y\_train)

    y\_pred\_linear = linear\_model.predict(X\_test)

    return y\_pred\_linear

Fungsi linear\_regression() menjalankan regresi linear dengan menggunakan data pelatihan dan mengembalikan prediksi untuk data uji.

Regresi Model Pangkat Sederhana

def power\_law\_regression(X\_train, y\_train, X\_test):

    # Transformasi data menggunakan log

    X\_train\_power = np.log(X\_train + 1)

    X\_test\_power = np.log(X\_test + 1)

    y\_train\_power = np.log(y\_train + 1)

    power\_model = LinearRegression()

    power\_model.fit(X\_train\_power, y\_train\_power)

    y\_pred\_power\_log = power\_model.predict(X\_test\_power)

    y\_pred\_power = np.exp(y\_pred\_power\_log) - 1

    return y\_pred\_power

Fungsi power\_law\_regression() melakukan transformasi log pada data untuk menerapkan regresi pangkat, kemudian mengembalikan hasil prediksi dengan mengembalikan data ke skala aslinya.

Hitung RMS

def calculate\_rms(y\_test, y\_pred):

    return np.sqrt(mean\_squared\_error(y\_test, y\_pred))

Visualisasi Hasil

def visualize\_results(y\_test, y\_pred\_linear, y\_pred\_power):

    # Visualisasi hasil regresi linear

    plt.figure(figsize=(12, 6))

    plt.subplot(1, 2, 1)

    plt.scatter(y\_test, y\_pred\_linear, color='blue')

    plt.plot([y\_test.min(), y\_test.max()], [y\_test.min(), y\_test.max()], 'k--', lw=2)

    plt.xlabel('Observed')

    plt.ylabel('Predicted')

    plt.title('Linear Regression')

    # Visualisasi hasil regresi pangkat

    plt.subplot(1, 2, 2)

    plt.scatter(y\_test, y\_pred\_power, color='red')

    plt.plot([y\_test.min(), y\_test.max()], [y\_test.min(), y\_test.max()], 'k--', lw=2)

    plt.xlabel('Observed')

    plt.ylabel('Predicted')

    plt.title('Power Law Regression')

    plt.tight\_layout()

    plt.show()

Main.py

import pandas as pd

import numpy as np

from sklearn.model\_selection import train\_test\_split

import matplotlib.pyplot as plt

from analisis import prepare\_data, load\_data, linear\_regression, power\_law\_regression, calculate\_rms, visualize\_results

def main():

    prepare\_data()

    data = load\_data()

    # Memisahkan fitur dan target

    X = data[['Durasi Waktu Belajar(TB)', 'Jumlah Latihan Soal(NL)']]

    y = data['Nilai Ujian Siswa (NL)']

    # Membagi data menjadi training dan testing

    X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

    # Fungsi untuk menjalankan analisis berdasarkan metode yang dipilih

    def run\_analysis(method):

        y\_pred\_linear = None

        y\_pred\_power = None

        if method == 'linear' or method == 'keduanya':

            y\_pred\_linear = linear\_regression(X\_train, y\_train, X\_test)

            rms\_linear = calculate\_rms(y\_test, y\_pred\_linear)

            print(f'RMS Error for Linear Model: {rms\_linear}')

        if method == 'pangkat' or method == 'keduanya':

            y\_pred\_power = power\_law\_regression(X\_train, y\_train, X\_test)

            rms\_power = calculate\_rms(y\_test, y\_pred\_power)

            print(f'RMS Error for Power Law Model: {rms\_power}')

        if method == 'keduanya':

            visualize\_results(y\_test, y\_pred\_linear, y\_pred\_power)

        elif method == 'linear':

            plt.figure(figsize=(6, 6))

            plt.scatter(y\_test, y\_pred\_linear, color='blue')

            plt.plot([y\_test.min(), y\_test.max()], [y\_test.min(), y\_test.max()], 'k--', lw=2)

            plt.xlabel('Observed')

            plt.ylabel('Predicted')

            plt.title('Linear Regression')

            plt.show()

        elif method == 'pangkat':

            plt.figure(figsize=(6, 6))

            plt.scatter(y\_test, y\_pred\_power, color='red')

            plt.plot([y\_test.min(), y\_test.max()], [y\_test.min(), y\_test.max()], 'k--', lw=2)

            plt.xlabel('Observed')

            plt.ylabel('Predicted')

            plt.title('Power Law Regression')

            plt.show()

        else:

            print("Metode tidak dikenal. Pilih 'linear', 'pangkat', atau 'keduanya'.")

    # Loop untuk meminta input metode dari pengguna

    while True:

        method = input("Pilih metode regresi (linear/pangkat/keduanya) atau ketik 'keluar' untuk berhenti: ").strip().lower()

        if method == 'keluar':

            print("Program dihentikan.")

            break

        elif method in ['linear', 'pangkat', 'keduanya']:

            run\_analysis(method)

        else:

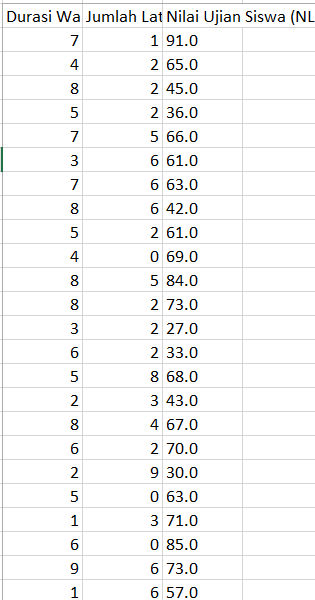
            print("Metode tidak dikenal. Pilih 'linear', 'pangkat', 'keduanya', atau 'keluar'.")

if \_\_name\_\_ == "\_\_main\_\_":

    main()

Fungsi ini memanggil fungsi untuk mempersiapkan dan memuat data, membagi data menjadi set pelatihan dan uji, dan mengelola pemilihan metode analisis serta visualisasi hasil berdasarkan input pengguna.

Contoh data yang sudah diproses



Tampilan Aplikasi Regresi

