**PRAKTIKUM**

**MACHINE LEARNING**



Nama : Nabilah Amaalina Syafa

NIM    : 2141720198

Kelas : TI-3D

JURUSAN TEKNIK INFORMATIKA POLITEKNIK NEGERI MALANG 2023

Jl. Soekarno Hatta No.9, Jatimulyo, Kec. Lowokwaru,

Kota Malang, Jawa Timur

TUGAS PRAKTIKUM

Tugas 1: Multiple Linear Regression

1. Identifikasi variabel-variabel yang akan digunakan sebagai variabel bebas (fitur) dan variabel target (biaya medis personal).
2. Bagi dataset menjadi data latih (train) dan data uji (test) dengan proporsi yang sesuai.
3. Lakukan feature scaling jika diperlukan.
4. Buat model multiple linear regression menggunakan Scikit-Learn.
5. Latih model pada data latih dan lakukan prediksi pada data uji.
6. Evaluasi model dengan menghitung metrik seperti R-squared, MSE, dan MAE. Tampilkan hasil evaluasi.

Jawab :

Langkah Pertama Import Library agar bisa memanggil objeknya  
#Import Library

from sklearn.linear\_model import LinearRegression

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

from sklearn.metrics import mean\_squared\_error, mean\_absolute\_error, r2\_score

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import pandas as pd

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

from sklearn.preprocessing import LabelEncoder, StandardScaler

dpath = '/content/drive/MyDrive/ML/WEEK03/insurance.csv'

df = pd.read\_csv(dpath)

df.head()

1. Variabel bebas (fitur) : age,sex,bmi,children,smoker,region

Variabel target (biaya medis personal) :charges



1. Menggunakan data independen (X) “age” dan “bmi” untuk data dependen (y) yaitu “charges”. Pembagian data latih dan data uji dengan proporsi 7:3

#Jawaban No 2

# Membuat variabel bebas X dan Y, contoh pengambilan dari analisis korelasi sebelumnya

X = df[['age','bmi']]

#X = data['Length of Membership']: Ini adalah perintah yang digunakan untuk membuat variabel X. Variabel X adalah variabel independen atau fitur dalam analisis.

y = df['charges']

#y = data['Yearly Amount Spent']: Ini adalah perintah yang digunakan untuk membuat variabel y. Variabel y adalah variabel dependen atau target dalam analisis.

# Pembagian data latih dan data uji dengan proporsi 7:3 (70% data latih dan 30% data uji)

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

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, train\_size=0.7, test\_size=0.3, random\_state=100)

# # Training model

# import statsmodels.api as sm

# # Menambahkan konstanta (intercept) ke data latih

# X\_train\_sm = sm.add\_constant(X\_train)AA

# # Melatih model regresi linier

# lr = sm.OLS(y\_train, X\_train\_sm).fit()

# print(lr.summary())

1. Scalling menggunakan MinMaxScaler yang bertujuan untuk mengubah nilai-nilai dalam dataset ke dalam rentang tertentu, seringkali antara 0 dan 1. Ini dilakukan dengan cara mengurangkan nilai minimum dari setiap data dan kemudian membaginya dengan selisih antara nilai maksimum dan minimum.

#Jawaban No 3

from sklearn.preprocessing import MinMaxScaler

# Inisialisasi objek scaler

scaler = MinMaxScaler()

# Melakukan scaling pada data latih

X\_train\_scaled = scaler.fit\_transform(X\_train)

# Melakukan scaling pada data uji dengan menggunakan parameter yang sama seperti pada data latih

X\_test\_scaled = scaler.transform(X\_test)

#Jawaban No 4

from sklearn.linear\_model import LinearRegression

# Inisialisasi objek model

model = LinearRegression()

1. Di bawah ini merupakan code untuk melatih model data latih dan melakukan prediksi pada data uji yang telah di scaling

#Jawaban No 5

# # Melatih model dengan data latih

# model.fit(X\_train, y\_train)

# # Membuat prediksi menggunakan data uji

# y\_pred = model.predict(X\_test)

# Melatih model dengan data latih yang telah di-scaled

model.fit(X\_train\_scaled, y\_train)

# Melakukan prediksi pada data uji yang telah di-scaled

y\_pred = model.predict(X\_test\_scaled)

1. Berikut adalah code untuk menghitung metrik evaluasi dan visualisasi multiple linier regression

#Jawaban No 6

# Menghitung metrik evaluasi

mse = mean\_squared\_error(y\_test, y\_pred)

mae = mean\_absolute\_error(y\_test, y\_pred)

r2 = r2\_score(y\_test, y\_pred)

# Menghitung metrik evaluasi (misalnya, RMSE dan R-squared)

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

# Menampilkan hasil evaluasi

print("Mean Squared Error (MSE):", mse)

print("Mean Absolute Error (MAE):", mae)

print("R-squared (R2):", r2)

print("Root Mean Squared Error (RMSE):", rmse)

# Membuat plot multiple linier regression untuk age

plt.scatter(X\_test['age'], y\_test, color='blue', label='Data Asli')

plt.scatter(X\_test['age'], y\_pred, color='red', label='Prediksi')

plt.xlabel("Umur (age)")

plt.ylabel("Biaya (charges)")

plt.title("Multiple Linier Regression untuk Umur (age)")

plt.legend()

plt.show()

# Membuat plot multiple linier regression untuk bmi

plt.scatter(X\_test['bmi'], y\_test, color='blue', label='Data Asli')

plt.scatter(X\_test['bmi'], y\_pred, color='red', label='Prediksi')

plt.xlabel("Indeks Massa Tubuh (bmi)")

plt.ylabel("Biaya (charges)")

plt.title("Multiple Linier Regression untuk Indeks Massa Tubuh (bmi)")

plt.legend()

plt.show()

