TUGAS AKHIR DSC

REGRESI LINEAR SEDERHANA

O

L

E

H

Nama : Elisabeth Woli Wadan

Nim : 192400008

UNIVERSITAS PGRI ADIBUANA SURABAYA

TAHUN 2022/2023

Berikut adalah penjelasan penjabaran regresi linear berganda di phyton:

Data set yang digunakan dalam kasus ini adalah Pengaruh Promosi Terhadap Keputusan Konsumen dalam membeli suatu produk. Data diambil dari sumber google dengan jumlah data sebanyak 20 responden dan telah diolah untuk kemudahan analisis.

Responden	promosi	Keputusan konsumen
A	10	23.4
B	2	7.5
C	4	15.7
D	6	17.8
E	8	23.8
F	7	22.7
G	4	10.5
H	6	14.4
I	7	20.5
J	6	19.8
K	5	18.7
L	7	13.8
M	6	30.5
N	7	25.4
O	6	8.6
P	4	10.7
Q	8	27.7
R	6	20.4
S	4	24.4
T	9	16.8

1) **Langkah pertama yang harus dilakukan adalah mengimport library yang diperlukan untuk mengimport data.**

untuk Mengimport data csv ke python dapat menggunakan library ’numpy’ dan ’pandas’. misalkan ellsa = pd.read_csv(’Pengaruh Promosi Terhadap Keputusan Pelanggan.csv’) digunakan untuk mengimport dataset. ”ellsa” disini merupakan nama data setnya yang akan dipanggil untuk dirunning di phyton, sedangkan ”pengaruh promosi terhadap keputusan pelanggan.csv” merupakan judul dari data yang akan dianalisis menggunakan phyton

In [4]:
```
#Importing numpy and pandas libraries to read

#Supress Warnings
import warnings
warnings.filterwarnings('ignore')

#import the numpy and pandas package
import numpy as np
import pandas as pd

ellsa = pd.read_csv('PENGARUH PROMOSI TERHADAP KEPUTUSAN KONSUMEN.csv')
ellsa
```

Hasil outputnya :

Out[4]:

	Responden	promosi	Keputusan konsumen
0	A	10	23.4
1	B	2	7.5
2	C	4	15.7
3	D	6	17.8
4	E	8	23.8
5	F	7	22.7
6	G	4	10.5
7	H	6	14.4
8	I	7	20.5
9	J	6	19.8
10	K	5	18.7
11	L	7	13.8
12	M	6	30.5
13	N	7	25.4
14	O	6	8.6
15	P	4	10.7
16	Q	8	27.7
17	R	6	20.4
18	S	4	24.4
19	T	9	16.8

2) **Memahami data.**

Dalam memahami data dapat menggunnakan library ’shape’, ’info’, dan ’describe’ yang berfungsi untuk melakukan analisis deskriptif secara otomatis terhadap dataset yang dipilih

```
In [5]: # shape of our dataset
        ellsa.shape

        #info our dataset
        ellsa.info()

        #describe our dataset
        ellsa.describe()
```

Hasil outputnya:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20 entries, 0 to 19
Data columns (total 3 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Responden           20 non-null     object
 1   promosi             20 non-null     int64
 2   Keputusan konsumen  20 non-null     float64
dtypes: float64(1), int64(1), object(1)
memory usage: 608.0+ bytes
```

Out[5]:

	promosi	Keputusan konsumen
count	20.000000	20.000000
mean	6.100000	18.655000
std	1.916686	6.422123
min	2.000000	7.500000
25%	4.750000	14.250000
50%	6.000000	19.250000
75%	7.000000	23.500000
max	10.000000	30.500000

Dari hasil output, menampilkan antara lain: nilai mean (rata-rata) dari kedua variabel yakni variabel promosi dengan nilai mean 6.1 dan variabel keputusan konsumen dengan nilai mean 18.65. output juga menampilkan nilai dari standar deviasi, nilai minimum dan maksimum, dll dari kedua variabel.

3) **Memvisualisasikan data.**

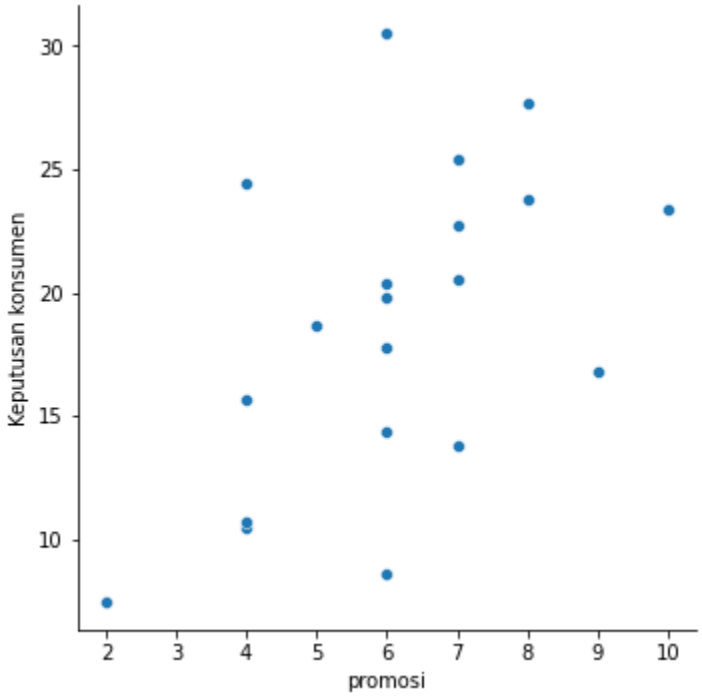
Untuk visualisasi data kita dapat menggunakan library ’matplotlib’ dan ’seaborn’ yang berfungsi untuk menampilkan grafik plot. Dalam analisis ini, koordinat X merupakan ’Promosi’ sedangkan koordinat Y merupakan ’keputusan pelanggan’

```
In [6]: import matplotlib.pyplot as plt
        import seaborn as sns

        #using pairplot we'll vizualize the data for correlation
        sns.pairplot(ellsa, x_vars=['promosi'],
                     y_vars='Keputusan konsumen', size=5, aspect=1,kind='scatter')

        plt.show()
```

Hasil otputnya :



Dari output dapat dilihat hasil visualisasi yang mana menampilkan scatterplot hubungan antara variabel promosi dan variabel keputusan pelanggan

4) **Melakukan regresi linear sederhana**.

Persamaan regresi linear sederhana

Y= c + mX

dalam kasus yang kami ambil:

Y = c + m*(promosi)

Nilai m dikenal sebagai **koefisien model** atau **patrameter model**

Untuk melakukan regresi linear sederhana, dapat dilakukan melalui 4 langkah:

a. Membuat X dan Y
b. Buat training dan testing set
c. Melatih model yang kita miliki
d. Evaluasi model

- Untuk menggeneralisasi, variabel independen mewakili X, dan Y merepresentasikan variebal target dalam model regresi linear sederhana.
- Membuat training and testing set
  Kita perlu membagi data menjadi dua bagian yakni data training (training set) dan data test (test set). Untuk data training 70% (0.7) dan untuk data test adalah 30% (0.3) dalam membagi data dapat mengimport train_test_split dari library sklearn.model_selection
  Lalu kemudian melihat kumpulan data trainingnya

In [7]:
```
#creating x and y
x= ellsa['promosi']
y= ellsa['Keputusan konsumen']
```

In [8]:
```
#splitting the variables as training and testing
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)
```

In [9]:
```
#take a look at the train dataset
x_train
y_train
```

Hasil outputnya:

```
In [9]: #take a look at the train dataset
        x_train
        y_train
```

```
Out[9]: 16    27.7
        1      7.5
        9     19.8
        14     8.6
        12    30.5
        5     22.7
        2     15.7
        4     23.8
        10    18.7
        0     23.4
        15    10.7
        7     14.4
        3     17.8
        8     20.5
        Name: Keputusan konsumen, dtype: float64
```

- ❖ Membangun dan melatih model

Dengan menggunakan dua paket berikut, kita dapat membangun model regresi linear sederhana

1. Statsmodel
2. Sklearn

A. library Statsmodel

Untuk yang pertama kami akan menggunakan library statsmodel untuk membuat model regresi sederhana.

Secara default, library statsmodel menyesuaikan dengan garis yang melewati asal. Tetapi jika kita mengamati persamaan regresi linear sederhana Y= c +mX, ia memiliki nilai intersep sebagai c. Jadi, untuk memiliki intersep kita perlu menambahkan add_constant atribut secara manual.

Setelah kita menambahkan konstanta, kita dapat menyesuaikan garis regresi menggunakan OLS metode (Ordinary Least Square) yang ada di statsmodel. Setelah itu, kita akan melihat parameternya, yaitu c dan m dari garis lurus.

Untuk melihat ringkasan dari semua parameter berbeda dari garis regresi yang dipasang seperti R^2, probabilitas F-statistic dan p-value dapat menggunakan library lr.summary().

In [10]:
```
#importing statsmodels.api library from stamodel package
import statsmodels.api as sm

#adding a constant to get an intercept
x_train_sm = sm.add_constant(x_train)
```

In [11]:
```
#Fitting the resgression line using 'OLS'
lr = sm.OLS(y_train, x_train_sm).fit()

#printing the parameters
lr.params
```

Out[11]:
```
const      4.441935
promosi    2.348387
dtype: float64
```

In [12]:
```
#performing a summary to list out all the different parameters of the regression line fitted
lr.summary()
```

Hasil outputnya:

Out[12]:

OLS Regression Results

Dep. Variable:	Keputusan konsumen	**R-squared:**	0.462
Model:	OLS	**Adj. R-squared:**	0.417
Method:	Least Squares	**F-statistic:**	10.29
Date:	Tue, 28 Dec 2021	**Prob (F-statistic):**	0.00753
Time:	22:29:12	**Log-Likelihood:**	-41.936
No. Observations:	14	**AIC:**	87.87
Df Residuals:	12	**BIC:**	89.15
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	4.4419	4.660	0.953	0.359	-5.711	14.595
promosi	2.3484	0.732	3.207	0.008	0.753	3.944

Omnibus:	3.489	**Durbin-Watson:**	2.513
Prob(Omnibus):	0.175	**Jarque-Bera (JB):**	1.119
Skew:	0.413	**Prob(JB):**	0.572
Kurtosis:	4.111	**Cond. No.**	21.7

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Dari hasil output dapat dilihat bahwa untuk nilai R- square nya = 0.462, untuk nilai F-statisticnya = 10,29, nilai coefisien konstanta= 4,4419 sedangkan coefisien untuk promosi = 2, 3484 dll dapat dilihat pada hasil output

Jadi statistik yang terutama diperhatikan untuk menentukan apakah model tersebut layak atau tidak adalah:

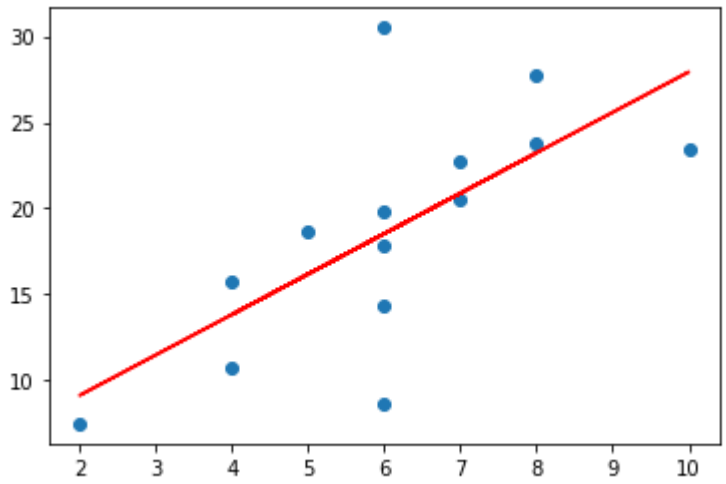
The coefficient dan p-value (signifikansi)
Nilai R-square
F-statistic dan signifikansinya

kita juga perlu memastikan bahwa nilai p harus selalu lebih kecil agar koefisiennya signifikan. Jika sudah signifikan maka dapat dilanjutkan dan memvisualisasikan seberapa cocok garis lurus dengan plot sebar antara Promosi dan Keputusan pelanggan.

- Memvisualisasikan garis regresi

```
In [13]: #vizulizing the regression line
plt.scatter(x_train, y_train)
plt.plot(x_train, 4.4419 + 2.3484*x_train, 'r')
plt.show()
```

Hasil outputnya:



Dari hasil output visusliasasi data menggunakan scatterplot dapat kita ketahui bahwa semakin tingginya promosi yang dilakukan pada suatu produk maka semakin tinggi pula keputusan konsumen untuk membeli produk tersebut

- Analisis Sisa

Salah satu asumsi utama dari model regresi linear adalah istilah kesalahan berdistribusi normal

Error = Actual y value – y predicted value

dari dataset, kita perlu memprediksi nilai y dari data set pelatihan(training)X menggunakan predict atribut. Setelah itu, kita akan menggunakan istilah kesalahan (residual) dari data yang diprediksi.

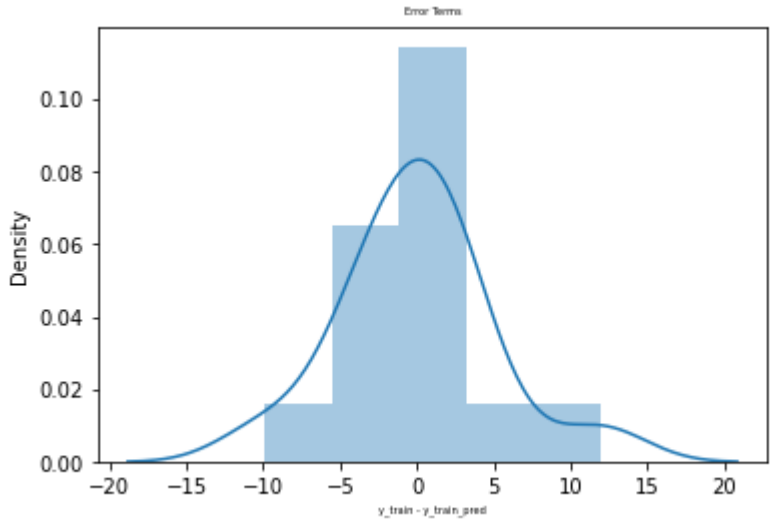
Kita dapat membuat plot histogram untuk melihat apakah data tampak seperti berdistribusi normal atau tidak dengan menggunakan sns.distplot(res,bins = 5)

```
In [14]: #predicting y_value using traingn data of x
         y_train_pred = lr.predict(x_train_sm)

         #creating residuals from the y_train data and predicted y_data
         res = (y_train - y_train_pred)
```

```
In [15]: #plotting the histogram using the residual values
         fig = plt.figure()
         sns.distplot(res,bins = 5)
         plt.title('Error Terms', fontsize = 5)
         plt.xlabel('y_train - y_train_pred', fontsize = 5)
         plt.show()
```

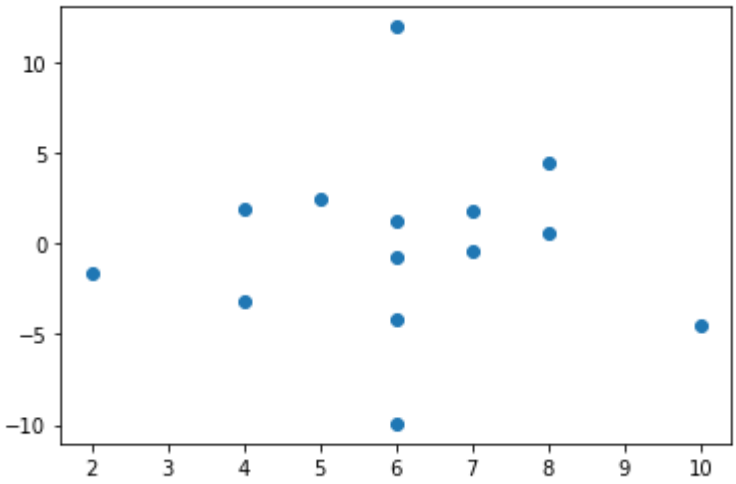
Hasil outputnya:



Lalu kemudian memastikan residu tidak memngikuti pola tertentu dengan membuat scatterplot plt.scatter(x_train,res)

```
In [16]: #looking for any patterns in the residuals
         plt.scatter(x_train,res)
         plt.show()
```

Hasil outputnya:



❖ Prediksi Pada Data Uji (Data Testing) Atau Mengevaluasi Model

Untuk membuat beberapa prediksi pada data pengujian(testing), mirip dengan set data pelatihan(training), kita perlu menggunakan add_constant menguji data dan memeprediksi nilai y menggunakan predict atribut yang ada di statsmodel, kemudian menambahkan sebuah konstanta ke x_test lalu memprediksi nilai y sesuai dengan x_test_sm dan kemudian mencetak 10 nilai prediksi pertama

```
In [17]: #adding a constant to x_test
         x_test_sm = sm.add_constant(x_test)

         #predicting the y values corresponding to x_test_sm
         y_test_pred = lr.predict(x_test_sm)

         #printing the first 10 predicted values
         y_test_pred
```

Hasil outputnya:

```
Out[17]: 17    18.532258
         19    25.577419
         11    20.880645
         18    13.835484
         13    20.880645
         6     13.835484
         dtype: float64
```

Setelah keputusan konsumen dapat prediksi, perusahaan atau pemilik usaha dapat menentukan tingkat promosi yang diperlukan untuk mencapai keuntungan yang maksimal. Namun model tersebu perlu dievaluasi untuk memeriksa kebaikannya dengan

menggunakan R kuadrat untuk nilai y yang akan diprediksi. Nila R kuadrat berada di antara 0 sampai 1.

Kita dapat melakukannya hanya dengan mengimport library r^2_score dari sklearn_matrics Jika nilai R^2. Sebuah model dianggap cukup baik jika nilai R kuadrat mendekat 1 yang artinya apa yang telah dipelajari dimodel pada set pelatihan(training) dapat digeneralisasikan pada set pengujian(testing) yang tidak terlihat.

Berikut syntax dan hasil outputnya:

```
In [18]: #importing r2_square
         from sklearn.metrics import r2_score

         #checking the R-square value
         r_squared = r2_score(y_test, y_test_pred)
         r_squared

Out[18]: -0.5647653259841916
```

Karena berdasarkan output nilai R kuadrat berada jauh dari nilai 1, sehingga dapat dikatakan model belum sesuai. Karenanya dalam hal ini dapat dikatakan bahwa sekitar 70% dari total variansi dari keputusan konsumen tidak dapat dijelaskan oleh promosi.

Ini adalah cara membangun model regresi linear menggunakan paket statsmodel. Selain dengan menggunakan statmodel, kita dapat membangun model regresi linear menggunakan sklearn.

B. library Sklearn

Berikut akan dijelaskan cara membangun model regresi linear menggunakan sklearn. Dengan menggunakan library linear_model dari sklearn, kita dapat membuat model regresi linear sederhana.

- Mirip dengan statsmodel, data akan dibagi data menjadi train dan test

```
In [20]: #splitting the data into train and test
         x_train_lm, x_test_lm, y_train_lm, y_test_lm = train_test_split(x, y, train_size = 0.7,
                                                                         test_size = 0.3, random_state = 100)
```

- Untuk regresi linear sederhana perlu menambahkan kolom untuk melakukan penyesuaian regresi dengan benar. Berikut syntax dan output yang ditampilkan:

```
In [21]: #shape of the train set without adding column
         x_train_lm.shape

         #adding additional column to the train and test data
         x_train_lm = x_train_lm.values.reshape(-1,1)
         x_test_lm = x_test_lm.values.reshape(-1,1)

         print(x_train_lm.shape)
         print(x_test_lm.shape)
```

```
(14, 1)
(6, 1)
```

Bentuk X_training sebelum menambahkan kolom adalah (14,1). Kemudian Bentuk x untuk data training dan testing adalah (6,1)

- Selanjutnya adalah mengimport library LinearRegression dari file sklearn.linear_model
  Berikut syntax dan hasil outputnya:

```
In [22]: from sklearn.linear_model import LinearRegression

         #creating an object of Linear Regression
         lm = LinearRegression()

         #fit the model using .fit() method
         lm.fit(x_train_lm, y_train_lm)
```

```
Out[22]: LinearRegression()
```

- Mencari nilai koefisien modelnya

```
In [24]: #intercept value
         print('intercept :',lm.intercept_)

         #slope value
         print('slope :',lm.coef_)
```

```
intercept : 4.441935483870974
slope : [2.3483871]
```

Nilai koefisiennya adalah 2. 3484

- Persamaan regresi linear sederhana yang didapatkan untuk nilai-nilai diatas adalah:
  Keputusan Konsumen = 4.4419 + 2.3484 * Promosi
  Jika dimati persamaan yang didapat sama dengan yang didapatkan di statsmodel.

- Setelah itu kita akan membuat prediksi dan data serta mengevaluasi model dengan membandingkan nilai R^2.
  Syntax dan outputnya adalah sebagai berikut:

```
In [25]: #making predictions of y_value
         y_train_pred = lm.predict(x_train_lm)
         y_test_pred = lm.predict(x_test_lm)

         #comparing the r2 value of both train and test data
         print(r2_score(y_train,y_train_pred))
         print(r2_score(y_test,y_test_pred))
```

```
0.4615425391497575
-0.5647653259841909
```

Nilai R^2 dari data train dan test adalah
R^2 train_data = 0.4615
R^2 test_data = -0,5647
Karena berdasarkan output nilai R kuadrat berada jauh dari nilai 1 yakni -0,5647, sehingga dapat dikatakan model belum sesuai. Karenanya dalam hal ini dapat dikatakan bahwa sekitar 70% dari total variansi dari keputusan konsumen tidak dapat dijelaskan oleh promosi.