Simple Linear Regression

Studi Kasus 1

Kali ini kita akan menggunakan dataset dimana didalamnya terdiri dari beberapa atribut/fitur yaitu ID, Income dan Consumption.

Tujuan dari latihan ini adalah untuk memprediksi tingkat konsumsi (Consumption) berdasarkan pendapatan (Income) yang dimiliki apabila Income yang diperoleh sebesar 100, 150 dan 200.

Hipotesis awal : Semakin tinggi Income maka semakin tinggi tingkat konsumsinya.

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

Load Library

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
```

Load Dataset

```
df = pd.read_csv('/content/drive/MyDrive/MATERI/Pembelajaran Mesin/Praktikum Genap 20222023/data-SLR.csv')

print(df)
```

```
    ID  Income  Consumption
0    1     119          154
1    2      85          123
2    3      97          125
3    4      95          130
4    5     120          151
5    6      92          131
6    7     105          141
7    8     110          141
8    9      98          130
9   10      98          134
10  11      81          115
11  12      81          117
12  13      91          123
13  14     105          144
14  15     100          137
15  16     107          140
16  17      82          123
17  18      84          115
18  19     100          134
19  21     108          147
20  21     116          144
21  22     115          144
22  23      93          126
23  24     105          141
24  25      89          124
25  26     104          144
26  27     108          144
27  28      88          129
28  29     109          137
29  30     112          144
30  31      96          132
31  32      89          125
32  33      93          126
33  34     114          140
34  35      81          120
35  36      84          118
36  37      88          119
37  38      96          131
38  39      82          127
39  40     114          150
```

Sneak Peak Data

```
#Melihat 5 baris teratas dari data
#Independent variabel(x) adalah Income
#Dependent variabel(y) adalah Consumption
df.head()
```

	ID	Income	Consumption

```
#Melihat informasi data kita mulai dari jumlah data, tipe data, memory yang digunakan
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 40 entries, 0 to 39
Data columns (total 3 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   ID           40 non-null     int64
 1   Income       40 non-null     int64
 2   Consumption  40 non-null     int64
dtypes: int64(3)
memory usage: 1.1 KB
```

```
#Melihat statistical description dari data mulai dari mean, kuartil, standard deviasi
df.describe()
```

	ID	Income	Consumption
count	40.000000	40.000000	40.000000
mean	20.525000	98.350000	133.000000
std	11.690425	11.802977	10.717826
min	1.000000	81.000000	115.000000
25%	10.750000	88.750000	124.750000
50%	21.000000	97.500000	131.500000

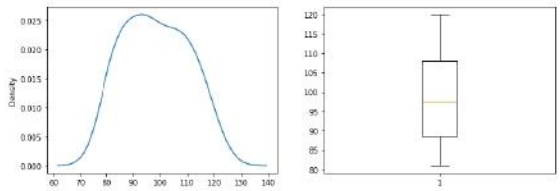
Handling Missing Values

```
#Mencari dan menangani missing values
#Ternyata data kita tidak ada missing values
df.isnull().sum()
```

```
ID             0
Income         0
Consumption    0
dtype: int64
```

Exploratory Data Analysis (EDA)

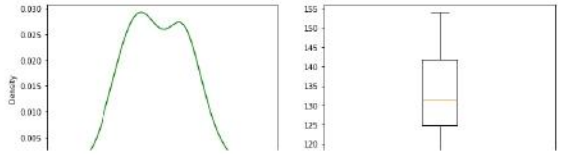
```
#Univariate analysis Income
#Melihat distribusi dari Income
f = plt.figure(figsize=(12,4))
f.add_subplot(1,2,1)
df['Income'].plot(kind='kde')
f.add_subplot(1,2,2)
plt.boxplot(df['Income'])
plt.show()
```



Dapat dilihat bahwa density dari Income paling tinggi ada pada kisaran nilai 90an.

Distribusinya sangat mendekati dengan distribusi normal dan persebaran data cukup merata.

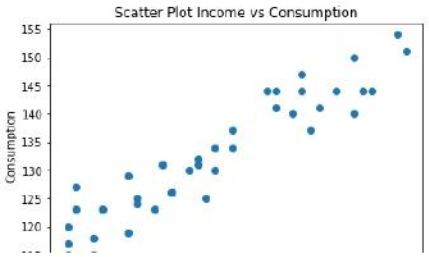
```
#Univariate analysis Consumption
#Melihat distribusi dari Consumption
f = plt.figure(figsize=(12,4))
f.add_subplot(1,2,1)
df['Consumption'].plot(kind='kde', c='g')
f.add_subplot(1,2,2)
plt.boxplot(df['Consumption'])
plt.show()
```



Dapat dilihat bahwa density dari Consumption paling tinggi ada pada kisaran nilai 120an.

Distribusinya mendekati dengan distribusi normal dan persebaran data cukup merata.

```
#Bivariate analysis Income dan Consumption
#Menggunakan scatter plot
plt.scatter(df['Income'], df['Consumption'])
plt.xlabel('Income')
plt.ylabel('Consumption')
plt.title('Scatter Plot Income vs Consumption')
plt.show()
```



Dari scatter plot dapat dilihat bahwa data memiliki korelasi positif yang cukup signifikan. Hal ini berarti dengan bertambahnya nilai dari Income maka nilai Consumption pun akan bertambah

```
#Mengetahui nilai korelasi dari Income dan Consumption
#Nilai korelasinya adalah 0,93 merupakan kategori yang sangat tinggi
df.corr()
```

	ID	Income	Consumption
ID	1.000000	-0.113235	-0.093318
Income	-0.113235	1.000000	0.939073

Modelling

```
#Recall data kita
df.head()
```

	ID	Income	Consumption
0	1	119	154
1	2	85	123
2	3	97	125
3	4	95	130

```
#Pertama, buat variabel x dan y
x = df['Income'].values.reshape(-1,1)
y = df['Consumption'].values.reshape(-1,1)
```

```
#Kedua, kita split data kita menjadi training and testing dengan porsi 80:20
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2)
```

```
#Ketiga, kita bikin object linear regresi
lin_reg = LinearRegression()
```

```
#Keempat, train the model menggunakan training data yang sudah displit
lin_reg.fit(x_train, y_train)
```

```
LinearRegression()
```

```
#Kelima, cari tau nilai slope/koefisien (m) dan intercept (b)
print(lin_reg.coef_)
print(lin_reg.intercept_)
```

```
[[0.86438392]]
[47.52500829]
```

dari nilai m dan b diatas, kalau dimasukkan kedalam rumus menjadi : Y = 50,62 + 0,84X

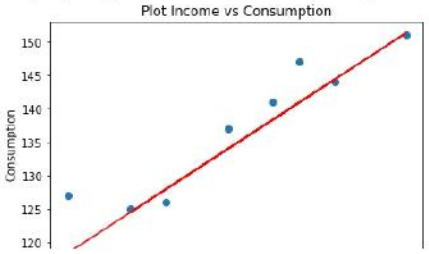
```
#Keenam, kita cari tahu accuracy score dari model kita menggunakan testing data
lin_reg.score(x_test, y_test)
```

```
0.8179495616530155
```

Model ini mendapat akurasi sebesar 93,83%

```
#Garis merah merupakan garis regresi dari persamaan yang kita dapat tadi
y_prediksi = lin_reg.predict(x_test)
plt.scatter(x_test, y_test)
plt.plot(x_test, y_prediksi, c='r')
plt.xlabel('Income')
plt.ylabel('Consumption')
plt.title('Plot Income vs Consumption')
```

```
Text(0.5, 1.0, 'Plot Income vs Consumption')
```



Prediction

Yuk kita prediksi tingkat konsumsi ketika memiliki income 100, 150, dan 200

```
#Prediksi Consumption dengan Income 100
lin_reg.predict([[100]])
```

```
array([[133.9633998]])
```

```
#Prediksi Consumption dengan Income 100
lin_reg.predict([[150]])
```

```
array([[177.18259555]])
```

```
#Prediksi Consumption dengan Income 100
lin_reg.predict([[200]])
```

```
array([[220.40179131]])
```

dengan melihat prediksi diatas, didapatkan bahwa hipotesis awal kita memang benar yaitu ketika income yang dimiliki 200 maka konsumsinya semakin meningkat!

Studi Kasus 2

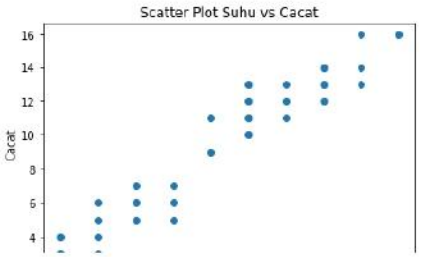
```
data = pd.read_csv('/content/drive/MyDrive/MATERI/Pembelajaran Mesin/Praktikum Genap 20222023/data-SLR2.csv')

print(data)
```

```
    Tanggal  Suhu  Cacat
0         1    24     10
1         2    22      5
2         3    21      6
3         4    20      3
4         5    22      6
5         6    19      4
6         7    20      5
7         8    23      9
8         9    24     11
9        10    25     13
10       11    21      7
11       12    20      4
12       13    20      6
13       14    19      3
14       15    25     12
15       16    27     13
16       17    28     16
17       18    25     12
18       19    26     14
19       20    24     12
20       21    27     16
21       22    23      9
22       23    24     13
23       24    23     11
```

```
24      25    22      7
25      26    21      5
26      27    26     12
27      28    25     11
28      29    26     13
29      30    27     14
```

```
#Bivariate analysis Suhu dan Cacat Produksi
#Menggunakan scatter plot
plt.scatter(data['Suhu'], data['Cacat'])
plt.xlabel('Suhu')
plt.ylabel('Cacat')
plt.title('Scatter Plot Suhu vs Cacat')
plt.show()
```




```
#Pertama, buat variabel x dan y
x = data['Suhu'].values.reshape(-1,1)
y = data['Cacat'].values.reshape(-1,1)
```

```
#Kedua, kita split data kita menjadi training and testing dengan porsi 80:20
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2)
```

```
#Ketiga, kita bikin object linear regresi
lin_reg = LinearRegression()
```

```
#Keempat, train the model menggunakan training data yang sudah displit
lin_reg.fit(x_train, y_train)
```

```
▾ LinearRegression
LinearRegression()
```

```
#Kelima, cari tau nilai slope/koefisien (m) dan intercept (b)
print(lin_reg.coef_)
print(lin_reg.intercept_)
```

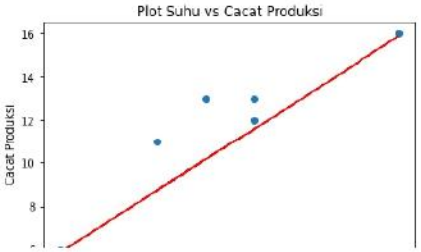
```
[[1.42586512]]
[-24.06264208]
```

```
#Keenam, kita cari tahu accuracy score dari model kita menggunakan testing data
lin_reg.score(x_test, y_test)
```

```
0.7185919740755646
```

```
#Garis merah merupakan garis regresi dari persamaan yang kita dapat tadi
y_prediksi = lin_reg.predict(x_test)
plt.scatter(x_test, y_test)
plt.plot(x_test, y_prediksi, c='r')
plt.xlabel('Suhu')
plt.ylabel('Cacat Produksi')
plt.title('Plot Suhu vs Cacat Produksi')
```

```
Text(0.5, 1.0, 'Plot Suhu vs Cacat Produksi')
```




```
#Prediksi Cacat Produksi dengan Suhu 30
lin_reg.predict([[30]])
```

```
array([[18.71331144]])
```