Belajar Machine Learning : Simple Linear Regression di Python

Adipta Martulandi Follow
Sep 7, 2019 · 6 min read

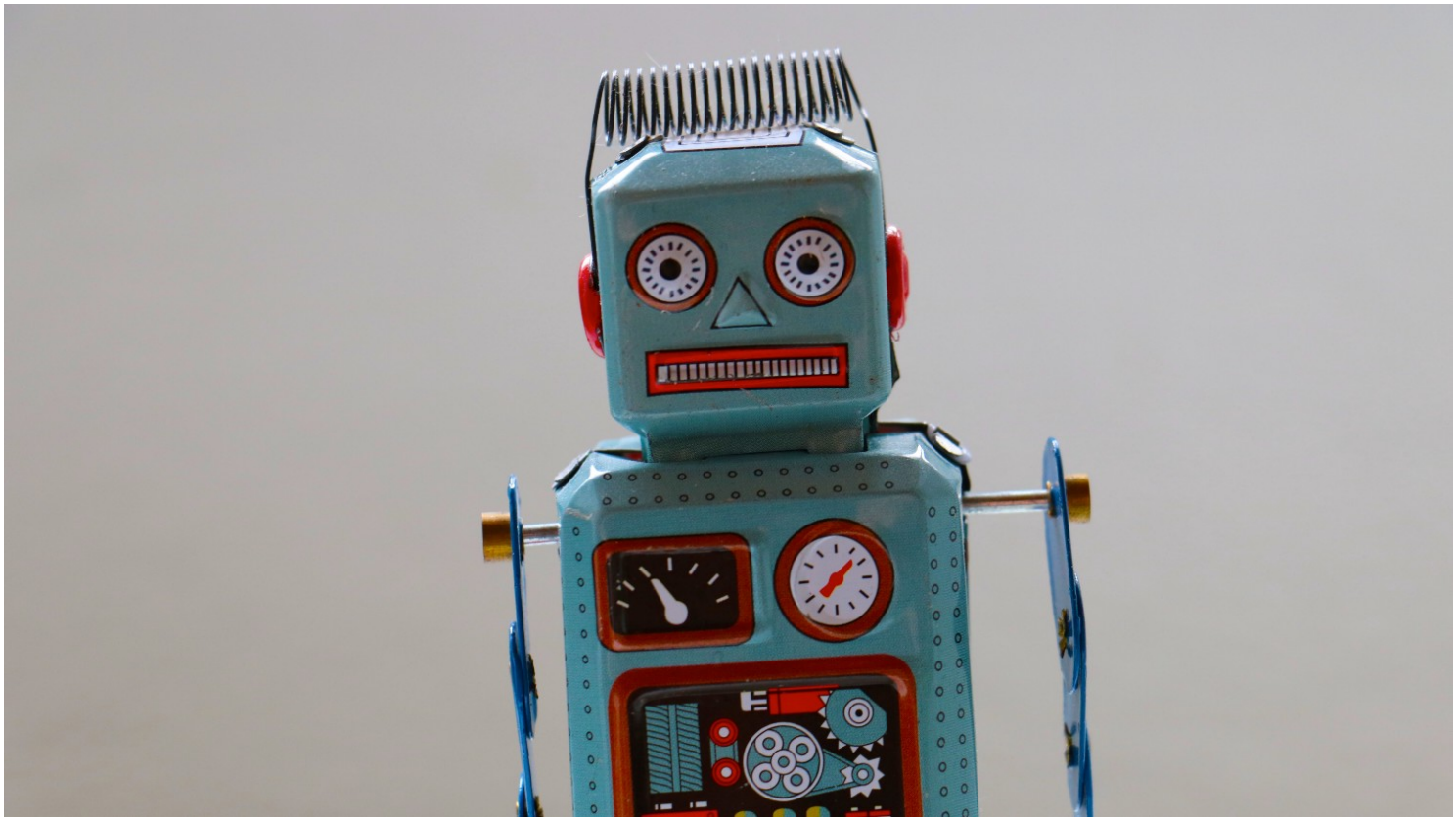


Photo by Rock'n Roll Monkey on Unsplash

Ada 3 jenis *Machine Learning (ML)* yang sekarang lagi populer, yaitu *Supervised Learning, Unsupervised Learning,* dan *Reinforcement Learning*. Pada kesempatan kali ini kita akan belajar salah satu algoritma *Supervised Learning* yaitu *Simple Linear Regression*. *Simple linear Regression* hanya mempunyai 1 independent variabel (x).

Walaupun sederhana, algoritma ini merupakan salah satu algoritma yang sangat populer karena *simple* tapi *powerful*.

Secara matematis, persamaan dari *Simple Linear Regression* adalah sebagai berikut:

> $y = mx + b + e$

y = dependent variable
m = slope dari garis (persamaan diatas merupakan sebuah garis)
x = independent variable
b = intercept
e = error

Jadi, secara sederhana tujuan dari *Simple Linear Regression* adalah untuk memprediksi nilai dari **y** dengan mengetahui nilai **x** dan menemukan nilai **m** dan **b** yang errornya paling minimal. Karena ini merupakan sebuah prediksi, maka persamaan diatas harus ditambahkan nilai error. Pada tutorial kali ini, kita akan memprediksi harga (y) dari sebuah mobil berdasarkan jumlah horsepower (x) dari mobil tersebut.

Dalam perhitungan kali ini akan menggunakan **Scikit-Learn**, salah satu library python yang sangat populer untuk *Machine Learning*. Dataset yang digunakan terdiri dari 1 variabel dependent (y) dan 1 variabel independent (x).

Dataset dan Full code bisa di DOWNLOAD di Github saya dan seluruh pengerjaan dilakukan di *Jupyter Notebook*.

> *Untuk mengikuti tutorial ini, setidaknya kalian harus tau terkait:*
> *1. Dasar pemrograman dengan **Python**.*
> *2. Library **Pandas** untuk data analysis tools.*
> *3. Library **Matplotlib** untuk visualisasi data.*
> *4. Library **Scikit-Learn** untuk Machine Learning*
> *5. Jupyter Notebook*

Tahapan dalam penggunaan *Simple Linear Regression* di artikel kali ini adalah sebagai berikut:
1. Load library python
2. Load dataset

3. Sneak peak data
4. Handling missing values
5. Exploratory Data Analysis (EDA)
6. Modelling
7. Prediction

1 Memuat beberapa library python yang akan digunakan dalam tutorial ini. Library tersebut adalah:

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

- Module *LinearRegression* digunakan untuk memanggil algoritma *Linear Regression.*
- Module *train_test_split* digunakan untuk membagi data kita menjadi training dan testing set.

2 Memuat dataset yang akan digunakan menggunakan library pandas dengan function *read_csv* (karena file kita extensionya csv).

```
df = pd.read_csv('data.csv', usecols=['horsepower', 'price'])
```

3 Melihat beberapa general information dari dataset kita agar kita lebih familiar dengan data yang kita punya.

```
#Melihat 5 baris teratas dari data.
#Independent variabel(x) adalah horsepower.
#Dependent variabel(y) adalah price.

df.head()
```

	horsepower	price
0	111.0	13495.0
1	111.0	16500.0
2	154.0	16500.0
3	102.0	13950.0
4	115.0	17450.0

5 Data Teratas

```
#Mengetahui jumlah kolom dan baris dari data.
#Data kita mempunya 2 kolom dengan 200 baris.
```

df.shape

```
(200, 2)
```

Jumlah baris dan kolom

```
#Melihat informasi data kita mulai dari jumlah data, tipe data,
memory yang digunakan dll.
```

df.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 2 columns):
horsepower    200 non-null float64
price         200 non-null float64
dtypes: float64(2)
memory usage: 3.2 KB
```

Informasi Data

```
#Melihat statistical description dari data mulai dari mean, kuartil,
standard deviation dll.
```

df.describe()

	horsepower	price
count	200.000000	200.000000
mean	103.320000	13230.375000
std	37.468615	7960.155239
min	48.000000	5118.000000
25%	70.000000	7775.000000
50%	95.000000	10320.000000
75%	116.000000	16500.750000
max	262.000000	45400.000000

Statistical Description

4 Krosscek dan Menangani *missing values* di data jika ada, jika tidak ada maka bisa dilanjutkan ke tahap *exploration data*.

```
#Mencari dan menangani missing values.
#Ternyata data kita tidak ada missing values.
```

df.isnull().sum()

```
horsepower    0
price         0
dtype: int64
```

Jumlah data yang missing

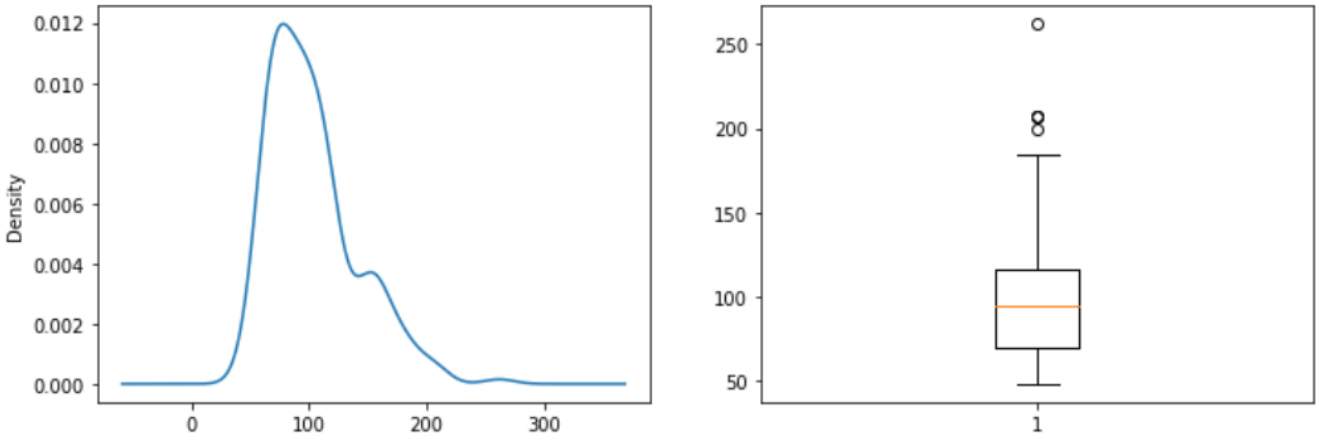
5 Melakukan *Exploratory Data Analysis (EDA)* untuk lebih mengenal data kita dan menemukan insights dari data.

```
#Univariate analysis horsepower.
#Melihat distribusi dari horsepower.

f = plt.figure(figsize=(12,4))

f.add_subplot(1,2,1)
df['horsepower'].plot(kind='kde')

f.add_subplot(1,2,2)
plt.boxplot(df['horsepower'])
plt.show()
```



Plot distribusi dan boxplot horsepower

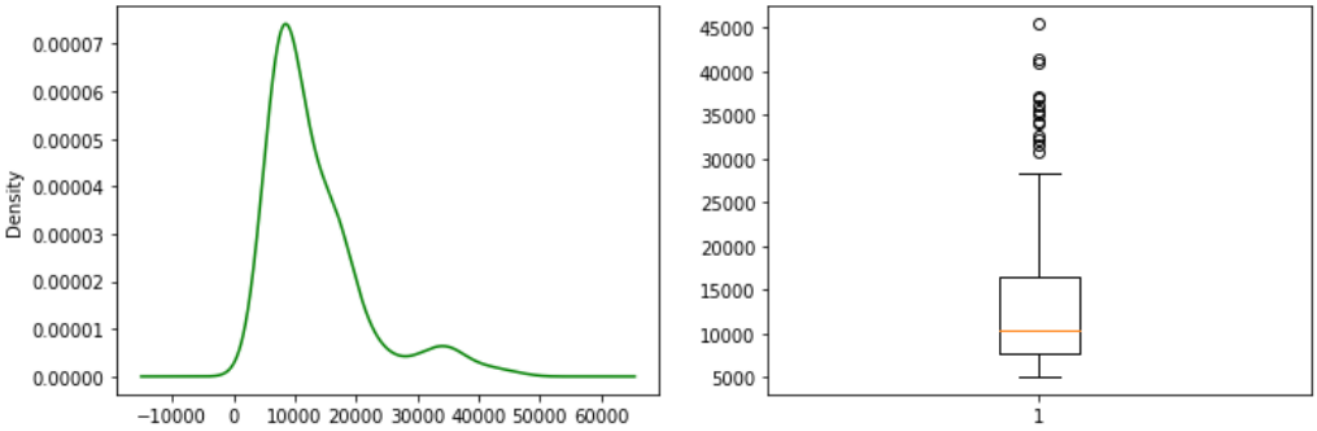
- Dapat dilihat bahwa *mean dan median* dari horsepower terpusat di sekitar nilai 100.
- Distribusinya hampir mirip dengan distribusi normal namun persebaran data kurang merata (memiliki *standard deviasi* yang tinggi) karena memiliki *whiskers boxplot* yang panjang.
- Terdapat 3 *outliers* data yang bisa dilihat di *boxplot*.

```
#Univariate analysis price.
#Melihat distribusi dari price.

f = plt.figure(figsize=(12,4))

f.add_subplot(1,2,1)
df['price'].plot(kind='kde', c='g')
```

```
f.add_subplot(1,2,2)
plt.boxplot(df['price'])
plt.show()
```

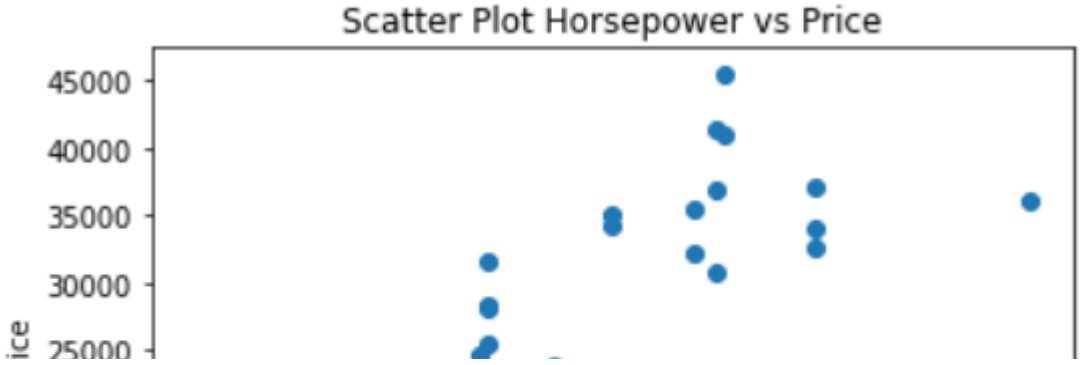


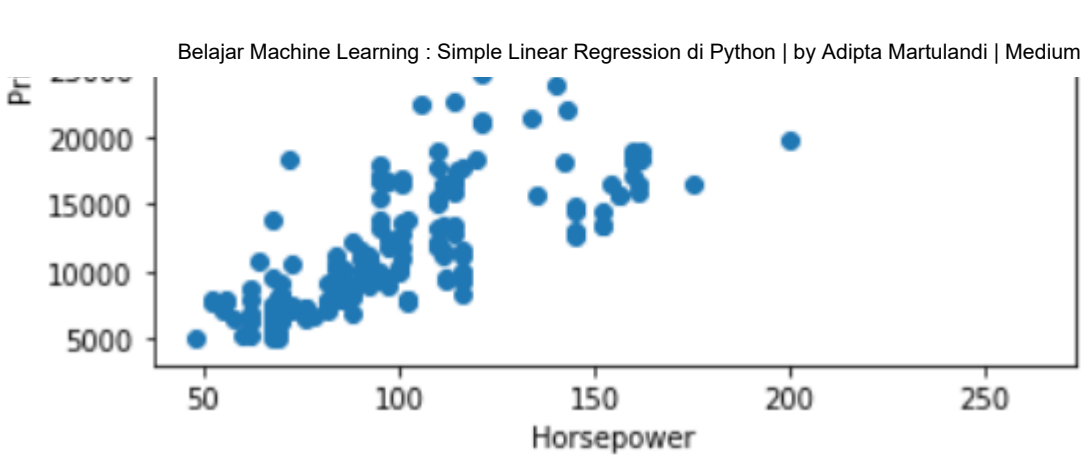
Plot distribusi dan boxplot price

- *Mean dan median* dari horsepower terpusat di sekitar nilai 10000.
- Distribusinya lebih ke *right skew* dan persebaran data kurang merata (memiliki standard deviasi yang tinggi).
- Data memiliki banyak *outliers* jika dibandingkan dengan horsepower.

```
#Bivariate analysis horsepower dan price.
#Menggunakan scatter plot.

plt.scatter(df['horsepower'], df['price'])
plt.xlabel('Horsepower')
plt.ylabel('Price')
plt.title('Scatter Plot Horsepower vs Price')
plt.show()
```





Scatter plot horsewer dan price

- Dari scatter plot dapat dilihat secara kasat mata bahwa data memiliki korelasi positif yang cukup signifikan.
- Hal ini berarti dengan bertambahnya nilai dari horsepower maka nilai price pun akan bertambah.

```
#Mengetahui nilai korelasi dari horsepower dan price.
#Nilai korelasinya adalah 0.81 termasuk kategori sangat tinggi.

df.corr()
```

	horsepower	price
horsepower	1.000000	0.811097
price	0.811097	1.000000

Nilai korelasi horsepower dan price

6 Setelah kita mengetahui karakteristik dari data kita, maka tahapan selanjutnya adalah Modelling.

1. Pertama, buat variabel (x) dan (y)

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

2. 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)
```

3. Ketiga, kita bikin object linear regresi.

```
lin_reg = LinearRegression()
```

4. Keempat, training the model menggunakan training data yang sudah displit sebelumnya.

```
lin_reg.fit(x_train, y_train)
```

5. Kelima, cari tau nilai slope/koefisien (m) dan intercept (b).

```
print(lin_reg.coef_)
print(lin_reg.intercept_)
```

```
[[164.73707883]]
[-3903.3911837]
```

Nilai m dan b

- Dari nilai **m** dan **b** diatas, kalau dimasukan ke dalam rumus menjadi:
  Y = 164.73x — 3911.83

6. Keenam, kita cari tahu accuracy score dari model kita menggunakan testing data yang sudah displit sebelumnya.

```
lin_reg.score(x_test, y_test)
```

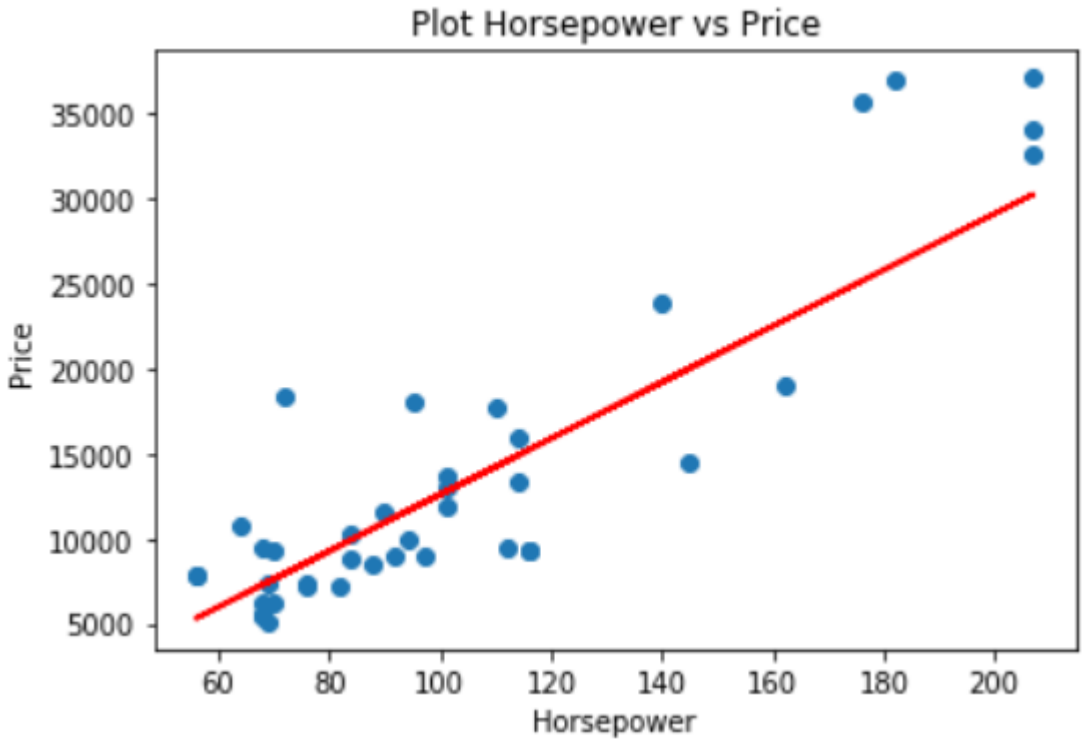
```
0.7867512368331953
```

Accuracy score model

- Model kita mendapatkan accuracy score sebesar 78.67%

7. Ketujuh, visualisasi *Regression Line* menggunakan data testing.

```
y_prediksi = lin_reg.predict(x_test)
plt.scatter(x_test, y_test)
plt.plot(x_test, y_prediksi, c='r')
plt.xlabel('Horsepower')
plt.ylabel('Price')
plt.title('Plot Horsepower vs Price')
```



Regression Line model

- Garis merah merupakan *Regression Line* dari model yang telah dibuat sebelumnya.

7 Setelah kita yakin dengan model yang dibuat, selanjutnya adalah prediksi dari harga mobil dengan horsepower 100, 150, dan 200.

```
#Prediksi harga mobil dengan horsepower 100.

lin_reg.predict([[100]])
```

```
array([[12570.3166989]])
```

Harga mobil dengan horsepower 100

```
#Prediksi harga mobil dengan horsepower 150.

lin_reg.predict([[150]])
```

```
array([[20807.1706402]])
```

Harga mobil dengan horsepower 150

```
#Prediksi harga mobil dengan horsepower 200.

lin_reg.predict([[200]])
```

```
array([[29044.0245815]])
```

Harga mobil dengan horsepower 200

> ***NOTES :***
> ***1****. Jangan lupa baca* ***asumsi-asumsi*** *yang harus dipenuhi ketika kalian akan menggunakan algoritma* ***Simple Linear Regression*** *yak!*

Well done! Kita telah menyelesaikan tutorial *Machine Learning* menggunakan algoritma *Simple Linear Regression*. Terimakasih telah membaca artikel ini, jika ada **saran atau kritik** bisa langsung comment di bawah ini. Saya yakin saran atau kritik yang kalian

berikan akan sangat membantu saya agar terus [redacted]kan skill saya di bidang **Data Science**.

Data Science Machine Learning Artificial Intelligence Linear Regression Python