

Regresi Linear Sederhana

Tentang Notebook

Dalam notebook ini, akan dipelajari cara menggunakan scikit-learn untuk mengimplementasikan regresi linier sederhana. Data yang dipakai adalah kumpulan data yang terkait dengan ukuran mesin dan emisi karbon dioksida mobil. Kemudian, data dibagi menjadi data pelatihan dan data pengujian, kemudian pembuatan model menggunakan dataset pelatihan, evaluasi model menggunakan dataset pengujian, dan akhirnya penggunaan model untuk memprediksi nilai yang tidak diketahui

Import packages yang diperlukan

In [1]:

```
import matplotlib.pyplot as plt
import pandas as pd
import pylab as pl
import numpy as np
%matplotlib inline
```

Pengunduhan Data

Untuk mengunduh data, gunakan wget dengan URL yang diberikan.

In [2]:

```
#install wget
#import wget
#menggunakan env windows
#wget.download('https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/Cognit
#SUDAH TERDOWNLOAD
```

Memahami Data

FuelConsumption.csv :

Dataset **FuelConsumption.csv** yang diunduh mengandung model spesifik untuk rating konsumsi bahan bakar (fuel consumption) dan estimasi emisi karbon dioksida untuk kendaraan ringan yang dijual di Kanada. [Dataset source \(http://open.canada.ca/data/en/dataset/98f1a129-f628-4ce4-b24d-6f16bf24dd64\)](http://open.canada.ca/data/en/dataset/98f1a129-f628-4ce4-b24d-6f16bf24dd64).

- **MODELYEAR** e.g. 2014
- **MAKE** e.g. Acura
- **MODEL** e.g. ILX
- **VEHICLE CLASS** e.g. SUV
- **ENGINE SIZE** e.g. 4.7
- **CYLINDERS** e.g 6

- **TRANSMISSION** e.g. A6
- **FUEL CONSUMPTION in CITY**(L/100 km) e.g. 9.9
- **FUEL CONSUMPTION in HWY** (L/100 km) e.g. 8.9
- **FUEL CONSUMPTION COMB** (L/100 km) e.g. 9.2
- **CO2 EMISSIONS** (g/km) e.g. 182 --> low --> 0

Membaca Data

In [3]:

```
df = pd.read_csv("FuelConsumptionCo2.csv")

# melihat dataset
df.head()
```

Out[3]:

	MODELYEAR	MAKE	MODEL	VEHICLECLASS	ENGINE SIZE	CYLINDERS	TRANSMISSION
0	2014	ACURA	ILX	COMPACT	2.0	4	A6
1	2014	ACURA	ILX	COMPACT	2.4	4	A6
2	2014	ACURA	ILX HYBRID	COMPACT	1.5	4	A6
3	2014	ACURA	MDX 4WD	SUV - SMALL	3.5	6	A6
4	2014	ACURA	RDX AWD	SUV - SMALL	3.5	6	A6

Eksplorasi Data

Eksplorasi deskriptif data yang diunduh.

In [4]:

```
# merangkum data
df.describe()
```

Out[4]:

	MODELYEAR	ENGINE SIZE	CYLINDERS	FUEL CONSUMPTION_CITY	FUEL CONSUMPT
count	1067.0	1067.000000	1067.000000	1067.000000	1067.000000
mean	2014.0	3.346298	5.794752	13.296532	13.296532
std	0.0	1.415895	1.797447	4.101253	4.101253
min	2014.0	1.000000	3.000000	4.600000	4.600000
25%	2014.0	2.000000	4.000000	10.250000	10.250000
50%	2014.0	3.400000	6.000000	12.600000	12.600000
75%	2014.0	4.300000	8.000000	15.550000	15.550000
max	2014.0	8.400000	12.000000	30.200000	30.200000

Beberapa fitur dapat dieksplorasi dengan cara berikut.

In [5]:

```
cdf = df[['ENGINE SIZE', 'CYLINDERS', 'FUEL CONSUMPTION_COMB', 'CO2 EMISSIONS']]
cdf.head(10)
```

Out[5]:

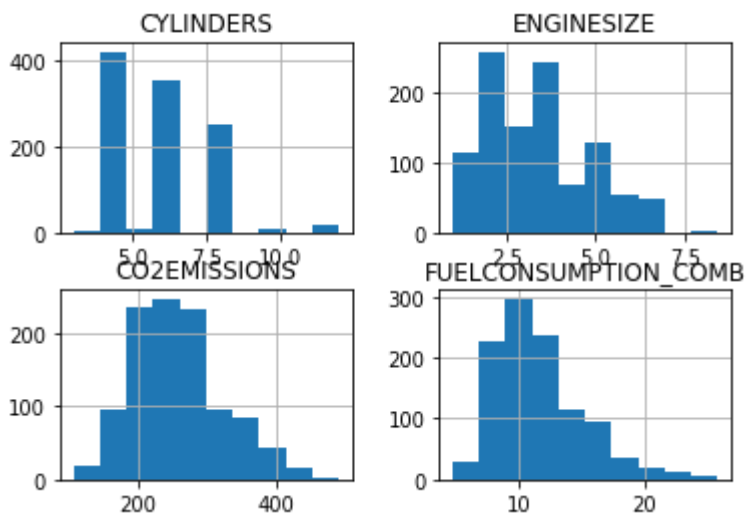
	ENGINE SIZE	CYLINDERS	FUEL CONSUMPTION_COMB	CO2 EMISSIONS
0	2.0	4	8.5	196
1	2.4	4	9.6	221
2	1.5	4	5.9	136
3	3.5	6	11.1	255
4	3.5	6	10.6	244
5	3.5	6	10.0	230
6	3.5	6	10.1	232
7	3.7	6	11.1	255
8	3.7	6	11.6	267
9	2.4	4	9.2	212

Fitur-fitur tersebut dapat diplot sebagai berikut:

In [6]:



```
viz = cdf[['CYLINDERS', 'ENGINE SIZE', 'CO2EMISSIONS', 'FUELCONSUMPTION_COMB']]
viz.hist()
plt.show()
```

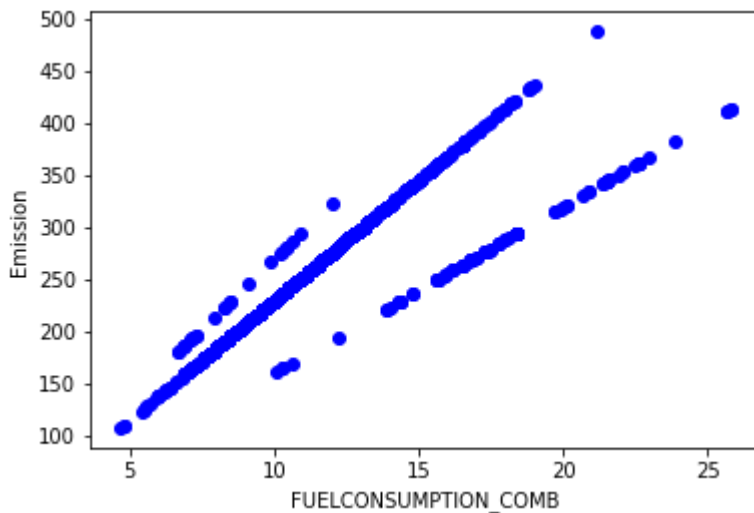


Plot fitur-fitur vs emisi dapat dibuat dan dapat dilihat linearitas hubungannya.

In [7]:



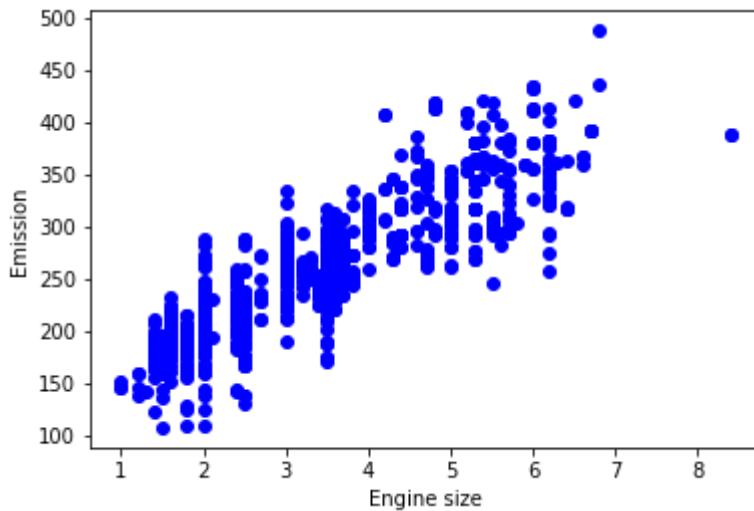
```
plt.scatter(cdf.FUELCONSUMPTION_COMB, cdf.CO2EMISSIONS, color='blue')
plt.xlabel("FUELCONSUMPTION_COMB")
plt.ylabel("Emission")
plt.show()
```



In [8]:



```
plt.scatter(cdf.ENGINESIZE, cdf.CO2EMISSIONS, color='blue')
plt.xlabel("Engine size")
plt.ylabel("Emission")
plt.show()
```



Praktik

plot **CYLINDER** vs the Emission, untuk melihat hubungan linearnya:

In [9]:



```
# write your code here
```

Double-click **di_sini** untuk solusi.

Pembuatan dataset pelatihan dan pengujian

Pemisahan data latih/uji melibatkan pemisahan dataset menjadi dataset pelatihan dan pengujian, yang saling eksklusif. Setelah itu, dataset pelatihan dapat digunakan untuk membuat model dan dataset pengujian untuk pengujian. Hal ini akan memberikan evaluasi yang lebih akurat pada akurasi out-of-sample karena dataset pengujian bukan merupakan bagian dari dataset yang telah digunakan untuk melatih data. Ini lebih realistis untuk masalah dunia nyata.

Ini berarti bahwa hasil dari setiap titik data dalam kumpulan data ini diketahui, sehingga sangat bagus untuk data pengujian. Dataset pengujian belum digunakan untuk melatih model, sehingga model tidak memiliki pengetahuan tentang hasil dari data ini, sehingga dapat disebut pengujian di luar sampel.

In [10]:

```
msk = np.random.rand(len(df)) < 0.8
train = cdf[msk]
test = cdf[~msk]
```

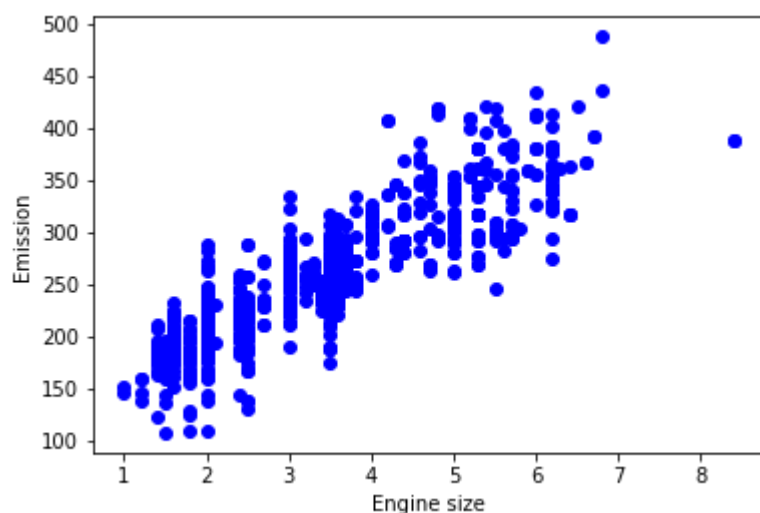
Model Regresi Sederhana

Regresi Linier cocok dengan model linier dengan koefisien $B = (B_1, \dots, B_n)$ untuk meminimalkan 'jumlah kuadrat sisa (residu)' antara x bebas dalam kumpulan data, dan y tak bebas dengan pendekatan linier.

Distribusi Data Pelatihan

In [11]:

```
plt.scatter(train.ENGINESIZE, train.CO2EMISSIONS, color='blue')
plt.xlabel("Engine size")
plt.ylabel("Emission")
plt.show()
```



Pemodelan

Menggunakan sklearn package untuk memodelkan data.

In [12]:

```
from sklearn import linear_model
regr = linear_model.LinearRegression()
train_x = np.asanyarray(train[['ENGINE_SIZE']])
train_y = np.asanyarray(train[['CO2EMISSIONS']])
regr.fit(train_x, train_y)
# The coefficients
print('Coefficients: ', regr.coef_)
print('Intercept: ', regr.intercept_)
```

```
Coefficients: [[38.95868737]]
Intercept: [125.96840085]
```

Seperti disebutkan sebelumnya, **koefisien** dan **intercept** dalam regresi linier sederhana, adalah parameter dari garis yang cocok dengan data. Mengingat bahwa ini adalah regresi linier sederhana, dengan hanya 2 parameter, dan mengetahui bahwa parameternya adalah intercept dan kemiringan atau gradien garis (koefisien), sklearn dapat memperkirakannya langsung dari data. Perhatikan bahwa semua data harus tersedia untuk menghitung parameter.

Plot output model

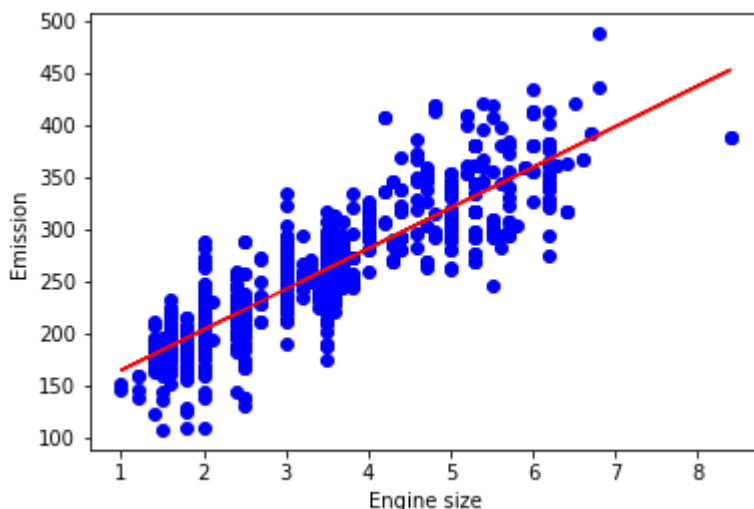
Plotting garis yang mencocoki terhadap data:

In [13]:

```
plt.scatter(train.ENGINESIZE, train.CO2EMISSIONS, color='blue')
plt.plot(train_x, regr.coef_[0][0]*train_x + regr.intercept_[0], '-r')
plt.xlabel("Engine size")
plt.ylabel("Emission")
```

Out[13]:

Text(0, 0.5, 'Emission')



Evaluasi

Nilai aktual dan nilai prediksi dapat dibandingkan untuk menghitung akurasi dari model regresi. Metrik evaluasi sangat penting untuk pengembangan model karena memberikan pengetahuan untuk perbaikan model.

Ada berbagai metrik untuk evaluasi model, misalnya MSE sebagai error untuk mengetahui akurasi dari model yang dibangun yang dihitung dari MSE model terhadap data pengujian:

- Mean Absolute Error (MAE): Rerata dari nilai absolut dari error. MAE adalah metrik paling mudah dipahami karena hanya rata-rata dari error.
- Mean Squared Error (MSE): adalah rerata dari error dikuadratkan. MSE lebih populer dibanding MAE karena fokus pada error yang besar karena dikuadratkan sehingga berdampak lebih besar terhadap error yang lebih besar dibandingkan error yang lebih kecil.
- Root Mean Squared Error (RMSE).
- R-squared bukan error namun metrik yang populer yang merepresentasikan sejauh mana data cocok dengan garis regresi yang didapatkan. Semakin besar R-squared akan semakin baik pencocokan garis terhadap data. Nilai terbaik adalah 1.0 dan dapat bernilai negatif.

In [14]:



```
from sklearn.metrics import r2_score

test_x = np.asanyarray(test[['ENGINE_SIZE']])
test_y = np.asanyarray(test[['CO2EMISSIONS']])
test_y_ = regr.predict(test_x)

print("Mean absolute error: %.2f" % np.mean(np.absolute(test_y_ - test_y)))
print("Residual sum of squares (MSE): %.2f" % np.mean((test_y_ - test_y) ** 2))
print("R2-score: %.2f" % r2_score(test_y_ , test_y) )
```

Mean absolute error: 23.54

Residual sum of squares (MSE): 952.69

R2-score: 0.68