

Regresi Linear Variabel Jamak

Tentang Notebook

Dalam notebook ini, akan dipelajari cara menggunakan scikit-learn untuk mengimplementasikan regresi linier variabel jamak. Data yang dipakai adalah kumpulan data yang terkait dengan, ukuran mesin, silinder, konsumsi bahan bakar dengan 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/CognitiveClass/ML0101Env3
#/Labs/FuelConsumptionCo2.csv')
#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	FUELTYPE	FUELCONSUMPTION_CITY	FUELCON
0	2014	ACURA	ILX	COMPACT	2.0	4	AS5	Z	9.9	
1	2014	ACURA	ILX	COMPACT	2.4	4	M6	Z	11.2	
2	2014	ACURA	ILX HYBRID	COMPACT	1.5	4	AV7	Z	6.0	
3	2014	ACURA	MDX 4WD	SUV - SMALL	3.5	6	AS6	Z	12.7	
4	2014	ACURA	RDX AWD	SUV - SMALL	3.5	6	AS6	Z	12.1	

Eksplorasi Data

Eksplorasi deskriptif data yang diunduh.

```
In [4]: # merangkum data
df.describe()
```

Out[4]:

	MODELYEAR	ENGINESIZE	CYLINDERS	FUELCONSUMPTION_CITY	FUELCONSUMPTION_HWY	FUELCONSUMPTION_COMB	FUELCONSUM
count	1067.0	1067.000000	1067.000000	1067.000000	1067.000000	1067.000000	
mean	2014.0	3.346298	5.794752	13.296532	9.474602	11.580881	
std	0.0	1.415895	1.797447	4.101253	2.794510	3.485595	
min	2014.0	1.000000	3.000000	4.600000	4.900000	4.700000	
25%	2014.0	2.000000	4.000000	10.250000	7.500000	9.000000	
50%	2014.0	3.400000	6.000000	12.600000	8.800000	10.900000	
75%	2014.0	4.300000	8.000000	15.550000	10.850000	13.350000	
max	2014.0	8.400000	12.000000	30.200000	20.500000	25.800000	

Beberapa fitur dapat dieksplorasi dengan cara berikut.

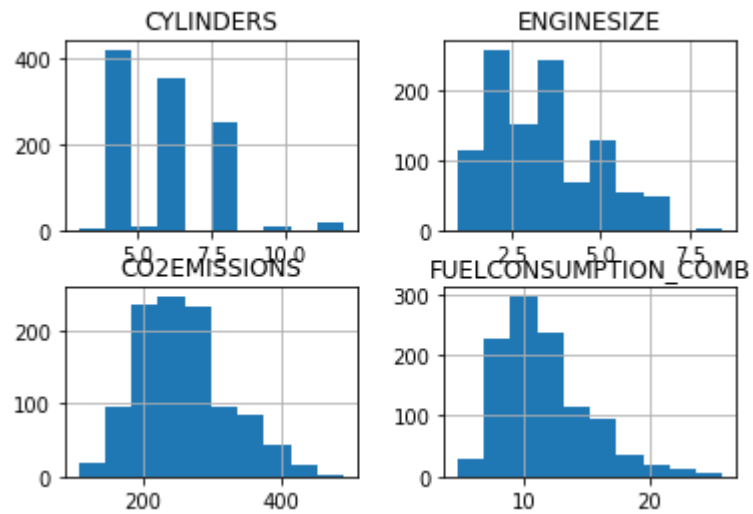
```
In [5]: ▶ cdf = df[['ENGINE_SIZE', 'CYLINDERS', 'FUELCONSUMPTION_COMB', 'CO2EMISSIONS']]  
cdf.head(10)
```

Out[5]:

	ENGINE_SIZE	CYLINDERS	FUELCONSUMPTION_COMB	CO2EMISSIONS
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

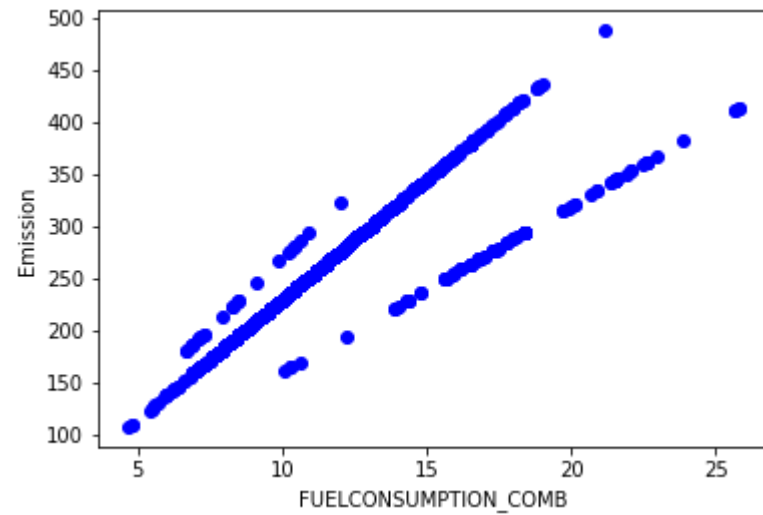
Fitur-fitur tersebut dapat diplot sebagai berikut:

```
In [6]: viz = cdf[['CYLINDERS', 'ENGINE SIZE', 'CO2EMISSIONS', 'FUELCONSUMPTION_COMB']]
viz.hist()
plt.show()
```

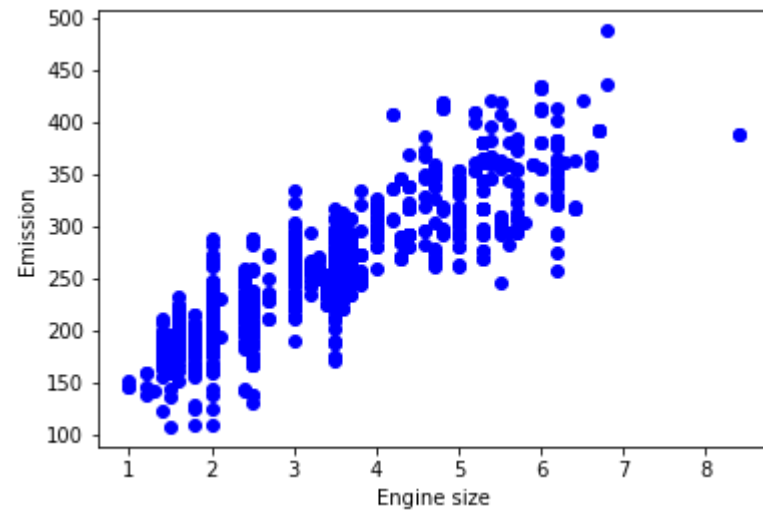


Plot fitur-fitur vs emisi dapat dibuat dan dapat dilihat linearitas hubungannya.

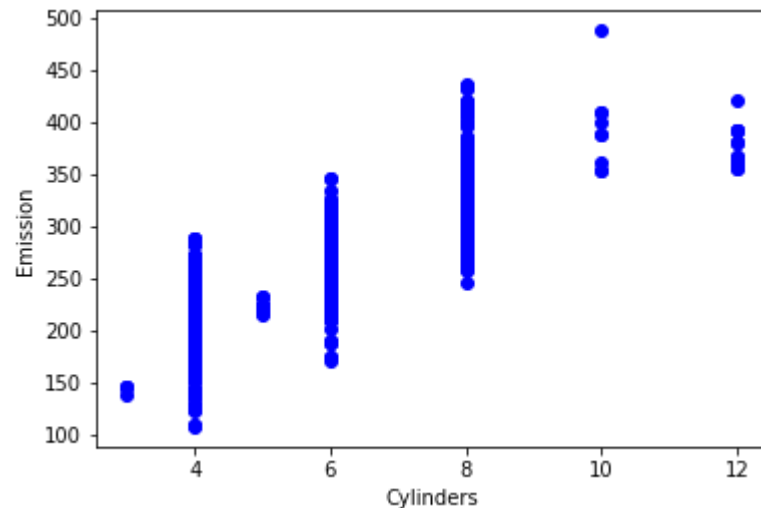
```
In [7]: ▶ plt.scatter(cdf.FUELCONSUMPTION_COMB, cdf.CO2EMISSIONS, color='blue')  
plt.xlabel("FUELCONSUMPTION_COMB")  
plt.ylabel("Emission")  
plt.show()
```



```
In [8]: ▶ plt.scatter(cdf.ENGINESIZE, cdf.CO2EMISSIONS, color='blue')  
plt.xlabel("Engine size")  
plt.ylabel("Emission")  
plt.show()
```




```
In [9]: ▶ plt.scatter(cdf.CYLINDERS, cdf.CO2EMISSIONS, color='blue')
plt.xlabel("Cylinders")
plt.ylabel("Emission")
plt.show()
```



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.

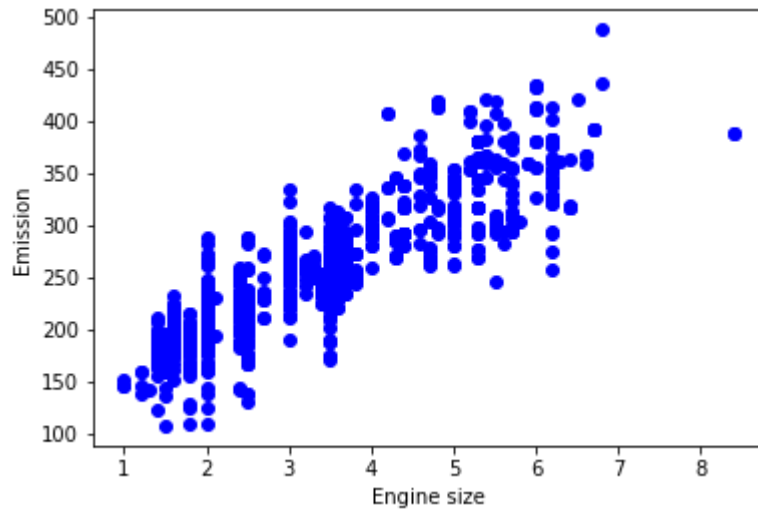
```
In [10]: ▶ msk = np.random.rand(len(df)) < 0.8
train = cdf[msk]
test = cdf[~msk]
```

Model Regresi Variabel Jamak

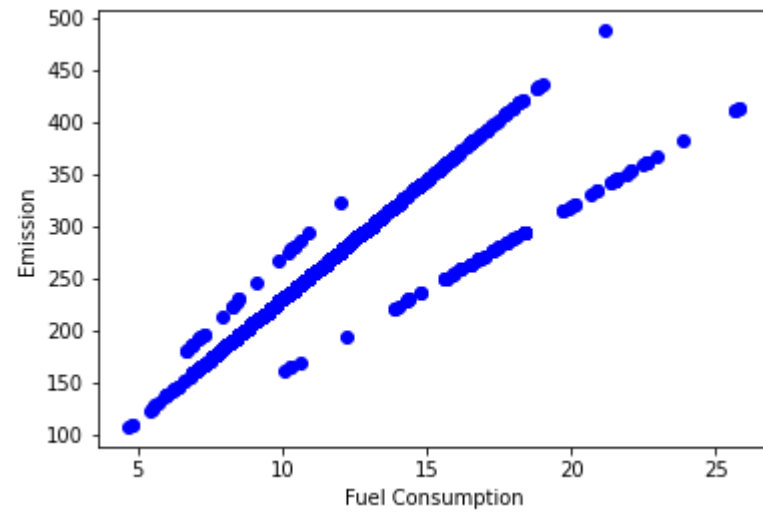
Regresi Linier cocok dengan model linier dengan koefisien $B = (B_1, \dots, B_n)$ untuk meminimalkan 'jumlah kuadrat sisa (residu)' antara x bebas (X_1, \dots, X_n) dalam kumpulan data, dan y tak bebas dengan pendekatan linier.

Distribusi Data Pelatihan

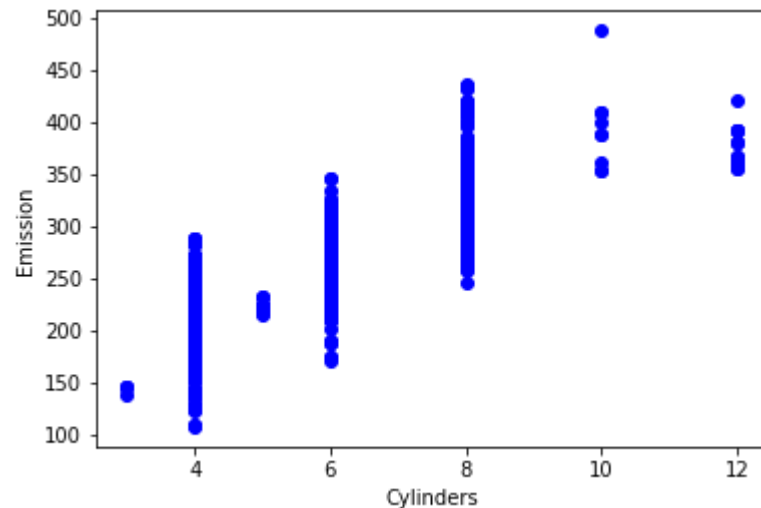
```
In [11]: ▶ plt.scatter(cdf.ENGINESIZE, cdf.CO2EMISSIONS, color='blue')  
plt.xlabel("Engine size")  
plt.ylabel("Emission")  
plt.show()
```



```
In [12]: ▶ plt.scatter(cdf.FUELCONSUMPTION_COMB, cdf.CO2EMISSIONS, color='blue')  
plt.xlabel("Fuel Consumption")  
plt.ylabel("Emission")  
plt.show()
```



```
In [13]: ▶ plt.scatter(cdf.CYLINDERS, cdf.CO2EMISSIONS, color='blue')
plt.xlabel("Cylinders")
plt.ylabel("Emission")
plt.show()
```



Pemodelan

Menggunakan sklearn package untuk memodelkan data.

```
In [14]: ▶ from sklearn import linear_model
regr = linear_model.LinearRegression()
train_x = np.asanyarray(train[['ENGINE_SIZE', 'FUELCONSUMPTION_COMB', 'CYLINDERS']])
train_y = np.asanyarray(train[['CO2EMISSIONS']])
regr.fit(train_x, train_y)
# The coefficients
print('Coefficients: ', regr.coef_)
print('Intercept: ', regr.intercept_)
```

```
Coefficients: [[9.76004394 9.74963679 7.96346191]]
Intercept: [63.80507573]
```

Seperti disebutkan sebelumnya, **koefisien** dan **intercept** dalam regresi linier sederhana, adalah parameter dari garis yang cocok dengan data.

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 [15]: ► from sklearn.metrics import r2_score

test_x = np.asanyarray(test[['ENGINE_SIZE', 'FUELCONSUMPTION_COMB', 'CYLINDERS']])
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: 16.85
Residual sum of squares (MSE): 514.11
R2-score: 0.85