



Direktorat Jenderal Pendidikan Tinggi, Riset, dan, Teknologi
Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi
Republik Indonesia



MICROCREDENTIAL: ASSOCIATE DATA SCIENTIST

01 November – 10 Desember 2021

Pertemuan ke-12

Membangun Model 3 (Regresi dengan Tree)



[ditjen.dikti](https://www.facebook.com/ditjen.dikti)



@ditjendikt
i



[ditjen.dikti](https://www.instagram.com/ditjen.dikti/)



Ditjen
Diktristek



<https://dikti.kemdikbud.go.id/>



Profil Pengajar: Nama Lengkap dan Gelar Akademik



Poto
Pengajar

Contak Pengajar:

Ponsel:

xxxxxx

Email:

xxxxxxx

Jabatan Akademik:

Latar Belakang Pendidikan:

- S1:
- S2:
- S3:

Riwayat/Pengalaman Pekerjaan:

- Dosen
- XXXX
- XXXX
- XXXX
- XXXX



KODE UNIT : **J.62DMI00.013.1**

JUDUL UNIT : **Membangun Model**

DESKRIPSI UNIT: Unit kompetensi ini berhubungan dengan pengetahuan, keterampilan, dan sikap kerja yang dibutuhkan dalam membangun model.

ELEMEN KOMPETENSI	KRITERIA UNJUK KERJA
1. Menyiapkan parameter model	1.1 Parameter-parameter yang sesuai dengan model diidentifikasi. 1.2 Nilai toleransi parameter evaluasi pengujian ditetapkan sesuai dengan tujuan teknis.
2. Menggunakan tools pemodelan	2.1 Tools untuk membuat model diidentifikasi sesuai dengan tujuan teknis <i>data science</i> . 2.2 Algoritma untuk teknik pemodelan yang ditentukan dibangun menggunakan <i>tools</i> yang dipilih. 2.3 Algoritma pemodelan dieksekusi sesuai dengan skenario pengujian dan <i>tools</i> untuk membuat model yang telah ditetapkan. 2.4 Parameter model algoritma dioptimasi untuk menghasilkan nilai parameter evaluasi yang sesuai dengan skenario pengujian.

1. Konteks variabel

- 1.1 Termasuk di dalam skenario pengujian adalah komposisi *data training* dan *data testing*, cara pemilihan data *training* dan data *testing* seperti *percentage splitting*, *random selection*, atau *cross validation*.
- 1.2 Yang dimaksud dengan parameter model di antaranya arsitektur model, banyaknya *layer* atau simpul, *learning rate* untuk *neural network*, nilai *k* untuk *k-means*, nilai *pruning* untuk *decision tree*.
- 1.3 Nilai parameter evaluasi adalah nilai ambang batas (*threshold*) yang bisa diterima.
- 1.4 Yang dimaksud dengan *tools* pemodelan di antaranya perangkat lunak *data science* di antaranya: *rapid miner*, *weka*, atau *development* untuk bahasa pemrograman tertentu seperti *python* atau *R*.



Definisi Kursus

Pelatihan ini menjelaskan regresi dan bagaimana membangun model (regresi), yaitu:

- a. menyiapkan parameter model
- b. menggunakan tools pemodelan

Selanjutnya menjelaskan algoritma dan menggunakan Regresi dengan tree, dan performansi regresi dengan Python dan Scikit-learn.



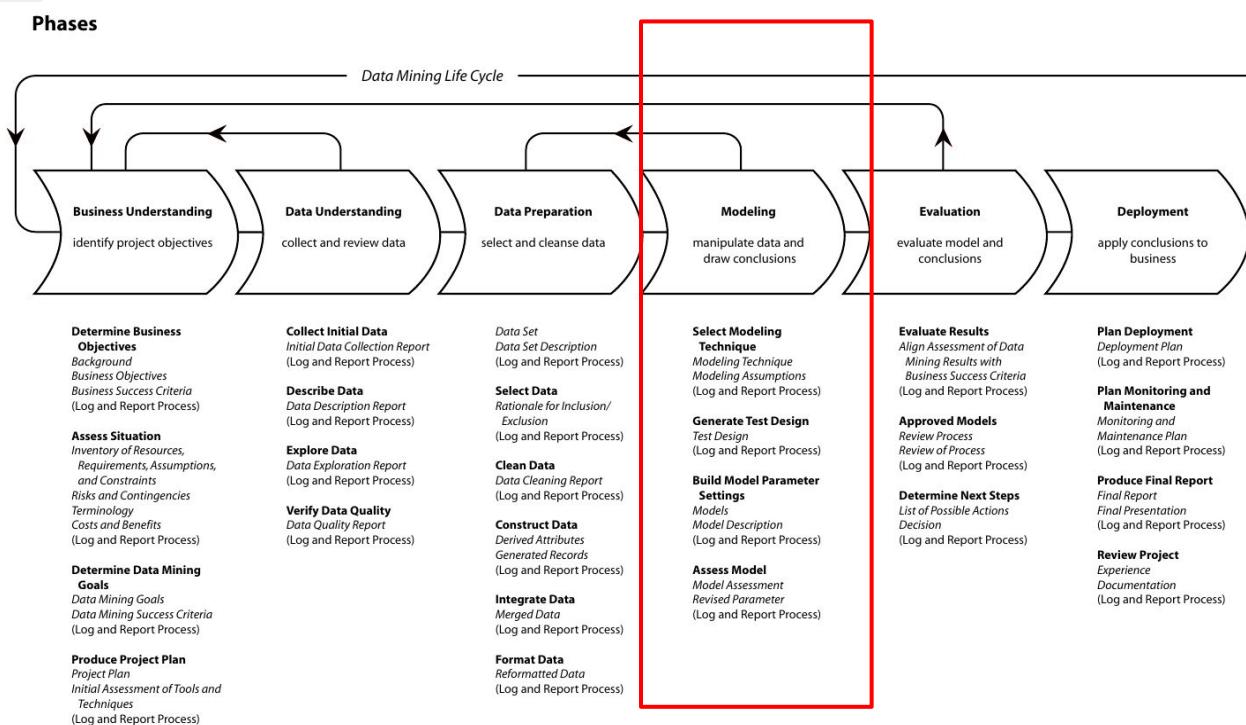
Capaian Pembelajaran

Peserta dapat menjelaskan, menyiapkan, dan mengimplementasikan model regresi dengan algoritma: Regresi dengan tree. Beserta pengukuran performansinya menggunakan Python dan Scikit-learn.



Tujuan Pembelajaran

Peserta mempelajari pengertian, cara menyiapkan, dan cara implementasi model regresi dengan algoritma Regresi dengan Tree. Beserta pengukuran performansinya menggunakan Python dan Scikit-learn.



a visual guide to CRISP-DM methodology

SOURCE CRISP-DM 1.0

<http://www.crisp-dm.org/download.htm>

DESIGN Nicole Leaper

<http://www.nicoleleaper.com>



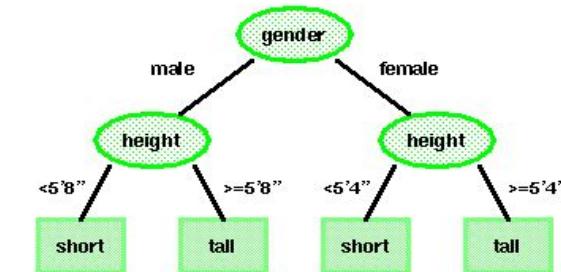


Decision Tree Regression



Overview

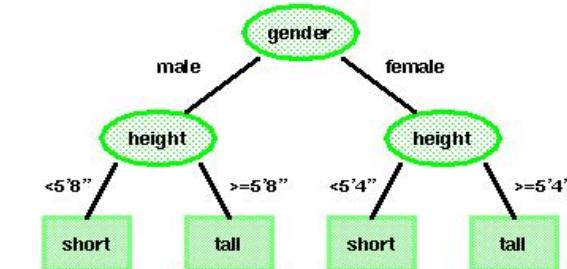
- Teknik sebelumnya terdiri dari vektor fitur bernilai nyata (atau bernilai diskrit) dan ukuran jarak alami (mis., Euclidean).
- Pertimbangkan masalah klasifikasi yang melibatkan data nominal – data yang dijelaskan oleh daftar atribut (misalnya, mengkategorikan orang sebagai pendek atau tinggi menggunakan jenis kelamin, tinggi badan, usia, dan etnis).
- Bagaimana kita bisa menggunakan data nominal seperti itu untuk klasifikasi? Bagaimana kita bisa mempelajari kategori data tersebut? Metode nonmetrik seperti pohon keputusan menyediakan cara untuk menangani data tersebut.
- Pohon keputusan berusaha untuk mengklasifikasikan suatu pola melalui urutan pertanyaan. Misalnya, atribut seperti jenis kelamin dan tinggi dapat digunakan untuk mengklasifikasikan orang sebagai pendek atau tinggi. Tetapi ambang batas terbaik untuk tinggi badan bergantung pada jenis kelamin.
- Sebuah pohon keputusan terdiri dari node dan daun, dengan setiap daun menunjukkan kelas.
- Kelas (tinggi atau pendek) adalah output dari pohon.
- Atribut (jenis kelamin dan tinggi badan) adalah seperangkat fitur yang menggambarkan data.
- Data input terdiri dari nilai-nilai atribut yang berbeda. Menggunakan nilai atribut ini, pohon keputusan menghasilkan kelas sebagai output untuk setiap data input.





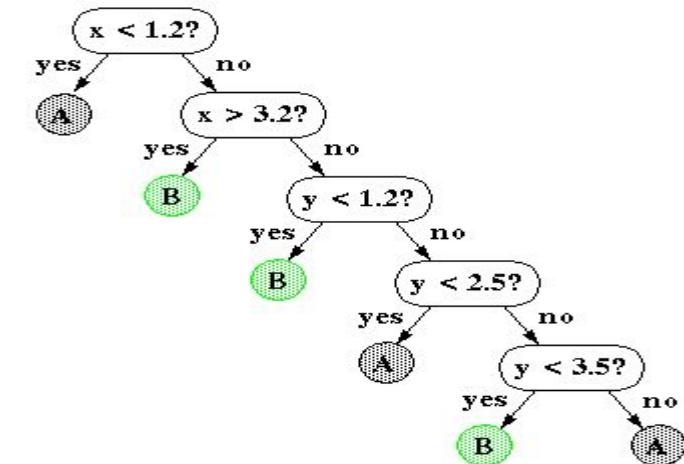
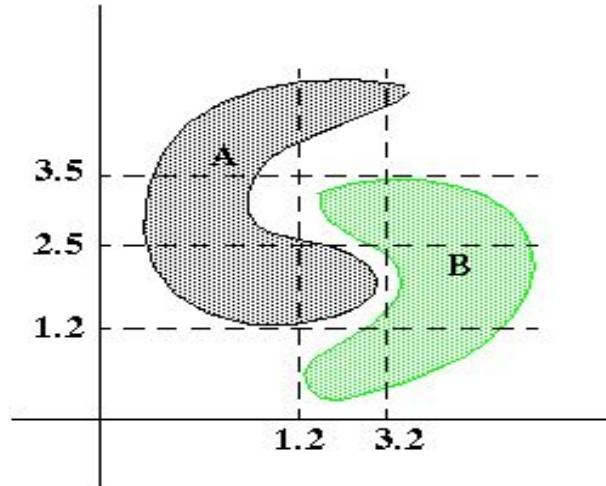
Basic Principles

- Bagian atas, atau simpul pertama, disebut simpul akar.
- Tingkat terakhir dari node adalah node daun dan berisi klasifikasi akhir.
- Node perantara adalah lapisan turunan atau "tersembunyi".
- Pohon biner, seperti yang ditunjukkan di sebelah kanan, adalah jenis pohon yang paling populer. Namun, pohon M-ary (cabang M di setiap node) dimungkinkan.
- Node dapat berisi satu pertanyaan lagi. Dalam pohon biner, dengan konvensi jika jawaban atas pertanyaan adalah "ya", cabang kiri dipilih. Perhatikan bahwa pertanyaan yang sama dapat muncul di beberapa tempat dalam jaringan.
- Pohon keputusan memiliki beberapa manfaat dibandingkan pendekatan tipe jaringan saraf, termasuk kemampuan interpretasi dan pembelajaran berbasis data.
- Pertanyaan kunci termasuk bagaimana menumbuhkan pohon, bagaimana menghentikan pertumbuhan, dan bagaimana memangkas pohon untuk meningkatkan generalisasi.
- Pohon keputusan sangat kuat dan dapat memberikan kinerja yang sangat baik pada pengujian set tertutup. Generalisasi adalah sebuah tantangan.



Nonlinear Decision Surfaces

- Pohon keputusan dapat menghasilkan permukaan keputusan nonlinier:



- Mereka adalah alternatif yang menarik untuk pengklasifikasi lain yang telah dipelajari karena mereka didorong oleh data dan dapat memberikan tingkat presisi yang tinggi pada data pelatihan.
- Tapi... generalisasi menjadi tantangan.



Classification and Regression Trees (CART)

- Pertimbangkan satu set D dari data pelatihan berlabel dan satu set properti (atau pertanyaan), T.
- Bagaimana kita mengatur pohon untuk menghasilkan kesalahan klasifikasi terendah?
- Setiap pohon keputusan akan secara berurutan membagi data menjadi himpunan bagian yang lebih kecil dan lebih kecil. Akan ideal jika semua sampel yang terkait dengan simpul daun berasal dari kelas kecil. Subset seperti itu, atau simpul, dianggap murni dalam kasus ini.
- Metodologi pertumbuhan pohon generik, yang dikenal sebagai CART, membagi node secara berurutan hingga menjadi murni. Enam pertanyaan kunci:
 1. Haruskah pertanyaannya biner (misalnya, apakah jenis kelamin laki-laki atau perempuan) atau numerik (misalnya, apakah tinggi badan $\geq 5'4''$) atau multi-nilai (misalnya, ras)?
 2. Properti mana yang harus diuji pada setiap node?
 3. Kapan sebuah node dinyatakan sebagai daun?
 4. Jika pohon menjadi terlalu besar, bagaimana bisa dipangkas?
 5. Jika simpul daun tidak murni, kategori apa yang harus diberikan padanya?
 6. Bagaimana seharusnya data yang hilang ditangani?



Machine Learning Methods

- Berbeda gaya dibanding inferensi statistik parametris klasik
- Sering memanfaatkan idea dari ilmu komputer dan teknik dimana karakteristik ketidakpastian (uncertainty) dikurangi dengan memanfaatkan pendekatan algoritmis alih-alih stokastik
- Mempelajari data ketimbang mengatur data menjadi suatu model linear atau terstruktur
- Mengapa menggunakan Tree
 - Menghasilkan model yang sederhana dan mudah
 - Seringkali lebih akurat daripada pendekatan parametris
 - Metoda dapat digunakan untuk sebarang jumlah variabel
 - Dapat memisahkan prediktor yang relevan dan tidak relevan
- Tree adalah salah satu penggunaan metode machine learning
 - Classification Tree: When Y (outcome) is binary/unordered categorical, you want to assign each subject to a category $Y=k$, Error assessment through misclassification cost.
 - Regression Tree: Y is continuous or ordered discrete values. Prediction error measured by squared or relative absolute difference between observed and predicted values.

Classification Tree

Pohon Klasifikasi: 3 label kelas, dua prediktor, partisi
Ruang X (ruang fitur) menjadi set persegi panjang

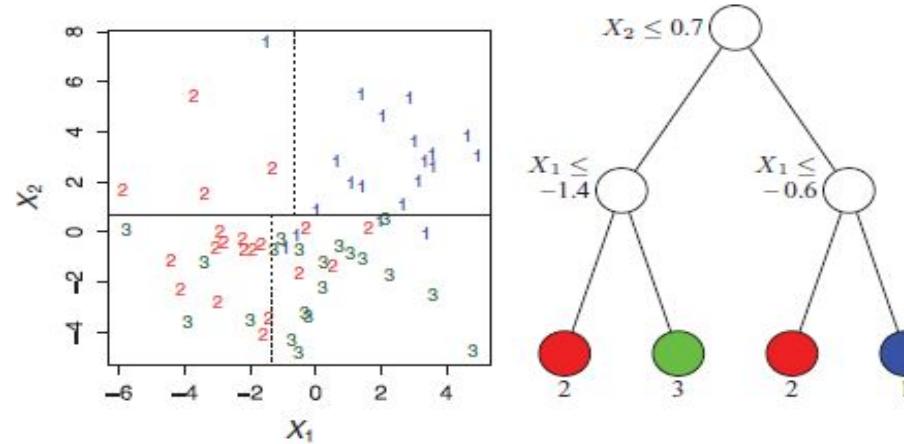
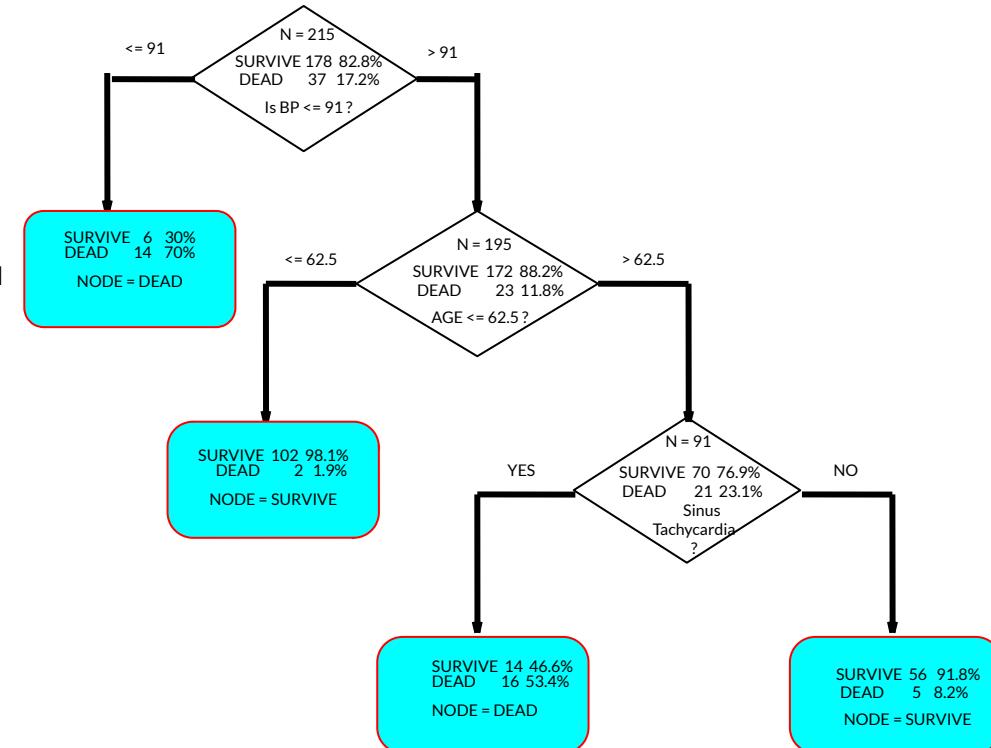


FIGURE 1 | Partitions (left) and decision tree structure (right) for a classification tree model with three classes labeled 1, 2, and 3. At each intermediate node, a case goes to the left child node if and only if the condition is satisfied. The predicted class is given beneath each leaf node.

Loh et al, 2011

Contoh sederhana

- Suatu Tree untuk Klasifikasi
- Variabel Tak Bebas adalah Binary (SURVIVE, DEATH)
- Ingin memprediksi anggota kelas
- Tree dengan variabel bebas yang kontinyu adalah Tree untuk Regresi





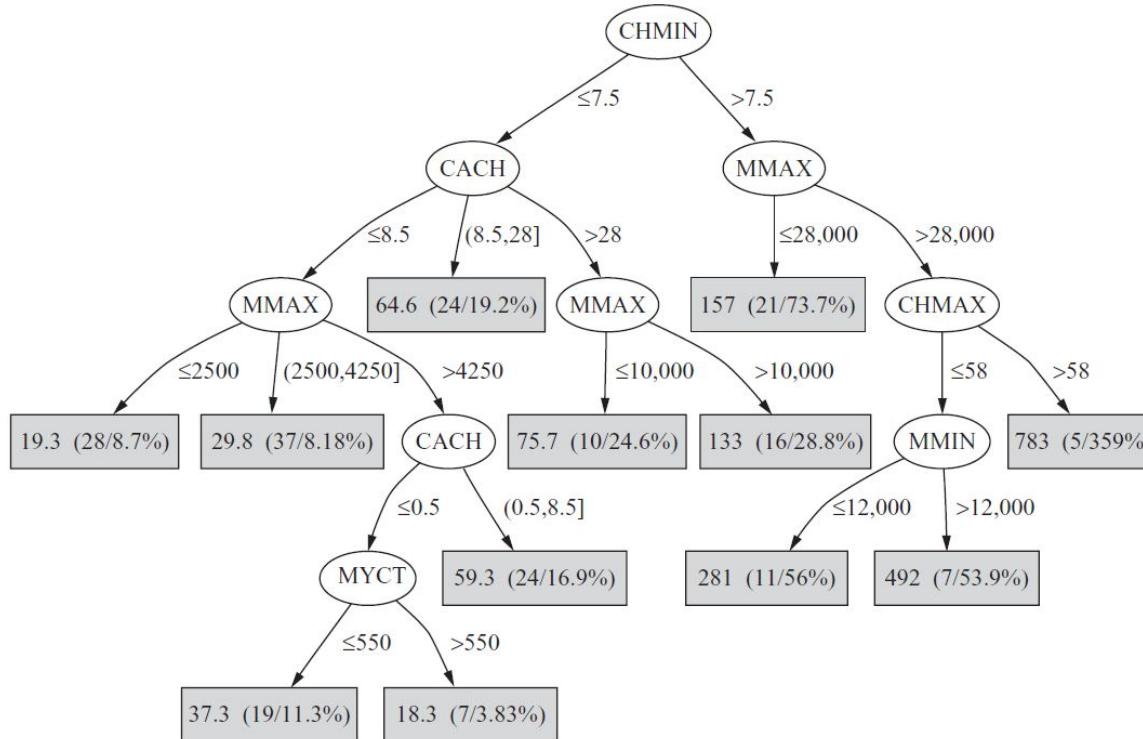
Hal penting dari Tree

- Seluruh Tree merepresentasikan analisis lengkap atau suatu model
- Merupakan bentuk dari Decision Tree
- Root dari Tree terbalik berisi semua data
- Root akan memberikan node anak, yang akan memberikan node anak lain. Hingga ke node akhir (node terminal)
- Node terminal mengklasifikasikan subyek → suatu klas tunggal
- Jalur Tree tersebut diatur oleh jawaban atas pertanyaan atau Rule (Aturan) dari Tree tersebut

Decision Tree Regression

- Decision Tree Regression (DTR) membangun model regresi dalam bentuk struktur pohon.
- DTR memecah dataset menjadi subset yang lebih kecil dan lebih kecil sementara pada saat yang sama pohon keputusan terkait dikembangkan secara bertahap.
- Hasil akhirnya adalah pohon dengan simpul keputusan dan simpul daun.
- Dengan titik data tertentu, DTR dijalankan sepenuhnya melalui seluruh pohon dengan menjawab pertanyaan Benar/Salah hingga mencapai simpul daun
- Prediksi terakhir adalah rata-rata dari nilai variabel dependen dalam simpul daun tertentu. Melalui beberapa iterasi, Pohon mampu memprediksi nilai yang tepat untuk titik data.

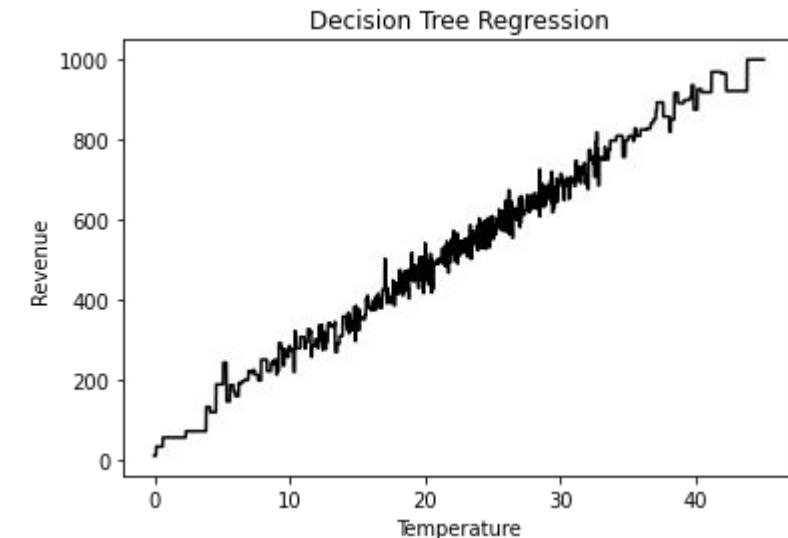
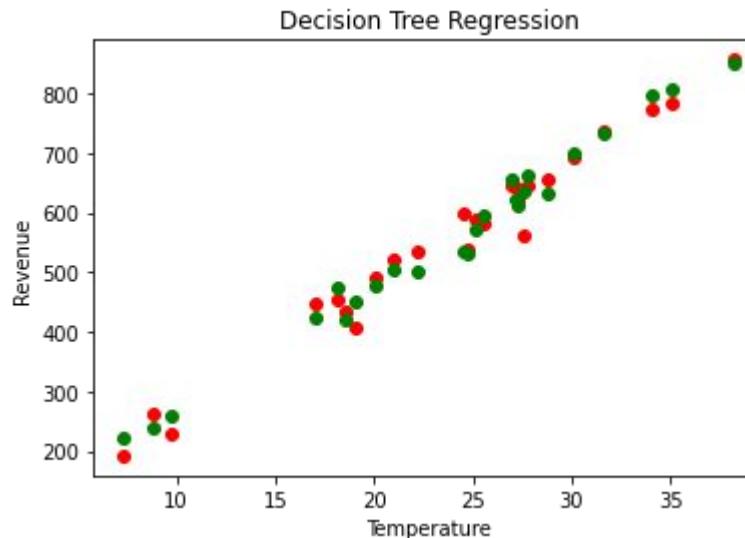
Decision Tree Regression



Hasil dari pemodelan regresi berupa Decision Tree dengan leaves berupa nilai rata-rata dari data yang ada di leaf tersebut dalam bentuk numerik sebagaimana tampak pada Gambar 17.

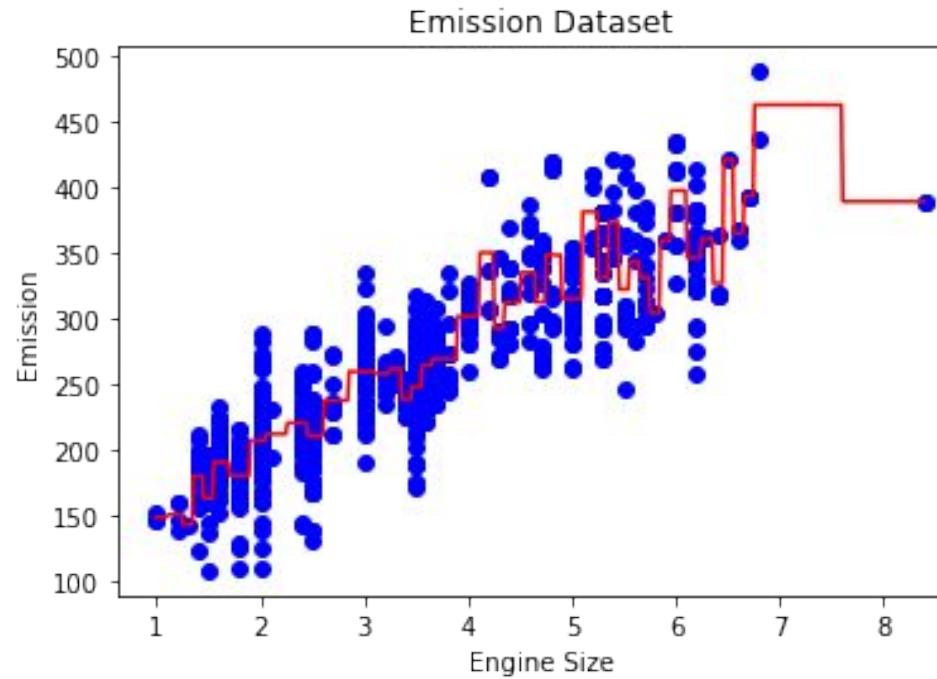


Decision Tree Regression - Contoh

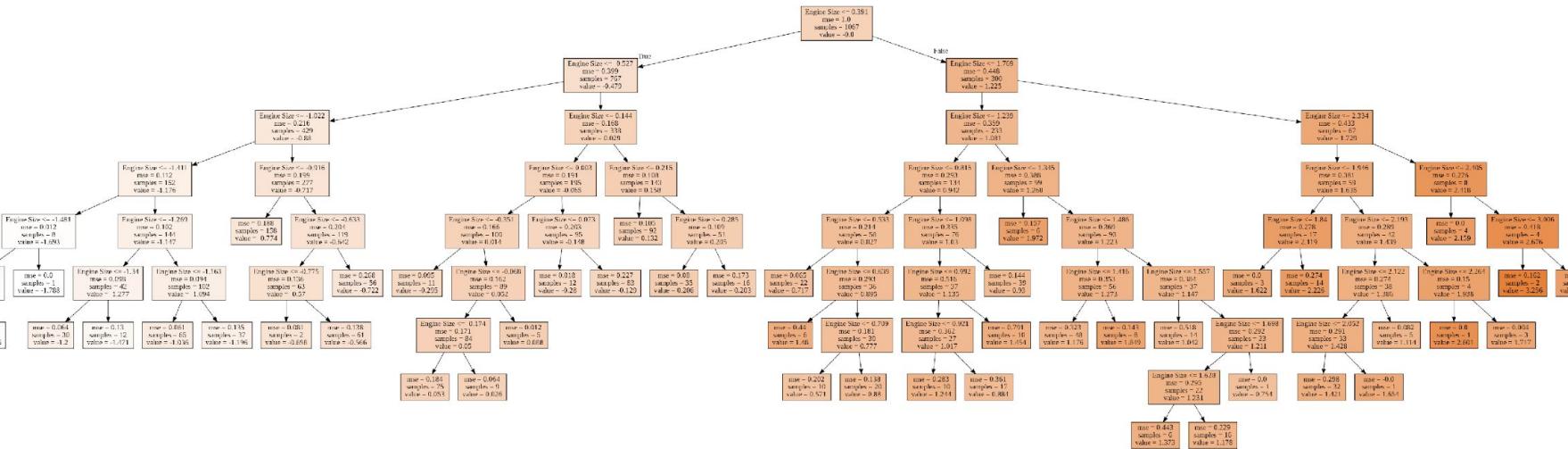




Aplikasi DTR pada Regresi Sederhana Emission Dataset



Visualisasi Struktur Tree DTR – Emission Dataset



Hasil pemodelan DTR pada Emission Dataset tampak pada Gambar 18 berupa garis patah-patah warna merah, sedangkan hasil tree-nya dapat dilihat pada Gambar 19.



Lab

- Jalankan file Jupyter Notebook untuk Decision Tree Regression



Home Page - Select or Decision_Tree_Regressi +

jupyter Decision_Tree_Regression Last Checkpoint: 3 minutes ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

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 realistik 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]:

```
# Mengambil "Engine Size" sebagai variabel independen (regressor)
X = cdf.iloc[:, 0].values
# Mengambil "Emission" sebagai variabel dependen
y = cdf.iloc[:, 3].values
# Reshape data karena hanya menggunakan satu fitur "Engine Size"
X = X.reshape(-1,1)
# Reshape data karena hanya satu fitur
y = y.reshape(-1,1)
```

Proses splitting dataset pelatihan dan pengujian

In [11]:

```
# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0) ← Split data
```

Normalisasi atau scaling dataset

In [12]:

```
# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc_X = StandardScaler()
sc_y = StandardScaler()
X = sc_X.fit_transform(X)
y = sc_y.fit_transform(y) ← Scaling
```

Pembuatan Model

In [13]:

```
# Fitting Decision Tree Regression to the dataset
from sklearn.tree import DecisionTreeRegressor
regressor = DecisionTreeRegressor()
regressor.fit(X, y)
```

Out[13]: DecisionTreeRegressor()

Decision Tree



Home Page - Select or Decision_Tree_Regressi +
127.0.0.1:8888/notebooks/Decision_Tree_Regression.ipynb

jupyter Decision_Tree_Regression Last Checkpoint: 4 minutes ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Pembuatan Model

In [15]: # Fitting Decision Tree Regression to the dataset
from sklearn.tree import DecisionTreeRegressor
regressor = DecisionTreeRegressor()
regressor.fit(X, y)

Out[15]: DecisionTreeRegressor()

Prediksi nilai baru dengan model yang telah dibentuk

In [16]: # Predicting a new result

y_pred = regressor.predict(sc_X.transform(np.array([[5.4]])))
#To transform 5.4 to the scaled X value, we first need to convert it into the array form
#Since the transform method of StandardScaler Library only accepts arrays

y_pred = sc_y.inverse_transform(y_pred)
#Now the prediction gives us the scaled value of y
#Thus we need inverse transformation of the scaled value for the real results

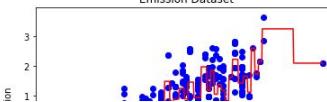
print(y_pred)

[2.09668256]

Visualisasi Hasil dalam nilai yang darsaling:

In [17]: # Visualising the Decision Tree Regression results (higher resolution)
X_grid = np.arange(min(X), max(X), 0.01)
X_grid = X_grid.reshape((len(X_grid), 1))
plt.scatter(X, y, color = 'blue')
plt.plot(X_grid, regressor.predict(X_grid), color = 'red')
plt.title('Emission Dataset')
plt.xlabel('Engine Size')
plt.ylabel('Emission')
plt.show()

Emission Dataset



Decision Tree

Decision Tree Regresi

Lakukan Prediksi untuk variabel bebas



Home Page - Select or Decision_Tree_Regress +

127.0.0.1:8888/notebooks/Decision_Tree_Regression.ipynb

jupyter Decision_Tree_Regression Last Checkpoint: 5 minutes ago (unsaved changes) Logout Trusted Python 3

File Edit View Insert Cell Kernel Widgets Help

Visualisasi Hasil dalam nilai yang discaling:

```
In [17]: # Visualising the Decision Tree Regression results (higher resolution)
X_grid = np.arange(min(X), max(X), 0.01)
X_grid = X_grid.reshape((len(X_grid), 1))
plt.scatter(X, y, color = 'blue')
plt.plot(X_grid, regressor.predict(X_grid), color = 'red')
plt.title('Emission Dataset')
plt.xlabel('Engine Size')
plt.ylabel('Emission')
plt.show()
```

Emission Dataset

Visualisasi Hasil dalam nilai asalnya:

```
In [16]: # Visualising the Decision Tree Regression results (higher resolution)
X_grid = np.arange(min(X), max(X), 0.01)
X_grid = X_grid.reshape((len(X_grid), 1))
plt.scatter(sc_X.inverse_transform(X), sc_y.inverse_transform(y), color = 'blue')
plt.plot(sc_X.inverse_transform(X_grid), sc_y.inverse_transform(regressor.predict(X_grid)), color = 'red')
plt.title('Emission Dataset')
plt.xlabel('Engine Size')
plt.ylabel('Emission')
plt.show()
```

Emission Dataset

Decision Tree

Plot decision tree regresi (merah)



Home Page - Select or Decision_Tree_Regressi Regresi_Linier_Sederha Random_Forest_Regres +

jupyter Decision_Tree_Regression (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help

Not Trusted Python 3

Emission Dataset

Visualisasi Tree (dengan nilai yang discaling):

```
In [17]: import graphviz
# DOT data
dot_data = export_graphviz(regressor, out_file=None, feature_names=['Engine Size'],
                           filled=True)

# Draw graph
graph = graphviz.Source(dot_data, format="png")
graph
```

ModuleNotFoundError Traceback (most recent call last)
<ipython-input-17-d40b3fd4c007> in <module>
----> 1 import graphviz
2 # DOT data
3 dot_data = export_graphviz(regressor, out_file=None, feature_names=['Engine Size'],
4 filled=True)
5
ModuleNotFoundError: No module named 'graphviz'

Visualisasi Tree dalam file PNG

```
In [ ]: graph.render("decision_tree_graphviz")
'decision_tree_graphviz.png'
```

Decision Tree

File Edit View Bookmarks Settings Help
localhost:~ # pip install graphviz
Collecting graphviz
 Downloading graphviz-0.17-py3-none-any.whl (18 kB)
Installing collected packages: graphviz
Successfully installed graphviz-0.17
localhost:~ #

Install dulu:
program graphviz
pip install graphviz



Home Page - Select or Decision_Tree_Regress x +

jupyter Decision_Tree_Regression Last Checkpoint: 9 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Visualisasi Tree (dengan nilai yang discaling):

```
In [22]: import graphviz
# DOT data
dot_data = sklearn.tree.export_graphviz(regressor, out_file=None, feature_names=['Engine Size'], filled=True)

# Draw graph
graph = graphviz.Source(dot_data, format="png")
graph
```

Out[22]:

```
graph TD
    Root["Engine n sa"] --> Left["Engine Size <= -1.411  
mse = 0.112"]
    Root --> Right["Engine n"]
```

Decision Tree

Perbaikan:
sklearn.tree.export_graphviz



Home Page - Select or Decision_Tree_Regress +

127.0.0.1:8888/notebooks/Decision_Tree_Regression.ipynb

Jupyter Decision_Tree_Regression Last Checkpoint: 10 hours ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Visualisasi Tree dalam file PNG

In [24]: graph.render("decision_tree_graphviz")
'decision_tree_graphviz.png'

Out[24]: 'decision_tree_graphviz.png'

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 [21]: from sklearn.metrics import r2_score

test_x = np.asanyarray(cdf[['ENGINESIZE']])
test_y = np.asanyarray(cdf[['CO2EMISSIONS']])
test_y_ = sc_y.inverse_transform(regressor.predict(sc_X.transform(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: 254.18
Residual sum of squares (MSE): 68620.04
R2-score: -209582.70

In []:

Decision Tree

Menyimpan image decision tree

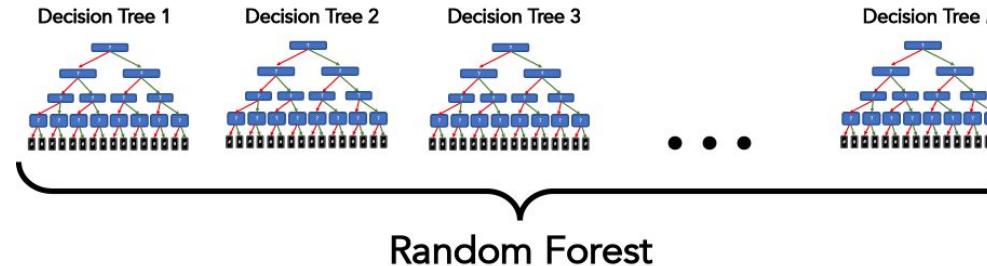
Evaluasi,



Random Forest Regression



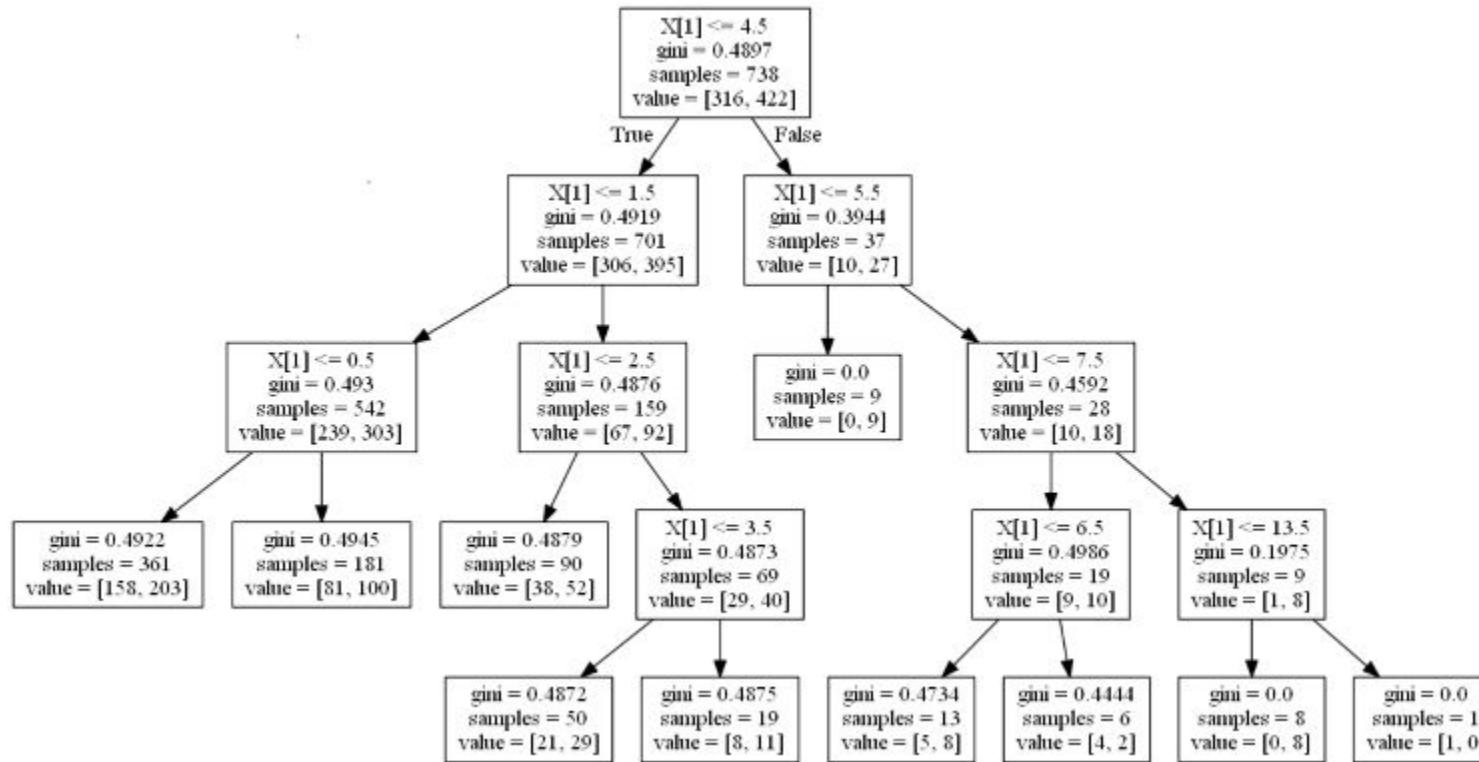
Random Forest Regression (RFR)



- Pohon Keputusan (Decision Tree) adalah algoritma yang mudah dipahami dan diinterpretasikan dan karenanya satu pohon mungkin tidak cukup bagi model untuk mempelajari fitur-fiturnya. Di sisi lain, Random Forest juga merupakan algoritma berbasis "Pohon" yang menggunakan fitur kualitas dari beberapa Pohon Keputusan untuk membuat keputusan.
- Oleh karena itu, dapat disebut sebagai 'Forest' atau 'Hutan' dari pohon-pohon dan karenanya disebut "Random Forest". Istilah 'Random' atau 'Acak' disebabkan oleh fakta bahwa algoritma ini adalah hutan dari 'Pohon Keputusan' atau Decision Tree yang dibuat secara acak atau random'.
- Algoritma Decision Tree memiliki kelemahan utama yaitu menyebabkan over-fitting. Masalah ini dapat diatasi dengan menerapkan Regresi Random Forest (Random Forest Regression) sebagai pengganti DTR. Selain itu, algoritma Random Forest juga sangat cepat dan kuat dibandingkan model regresi lainnya.

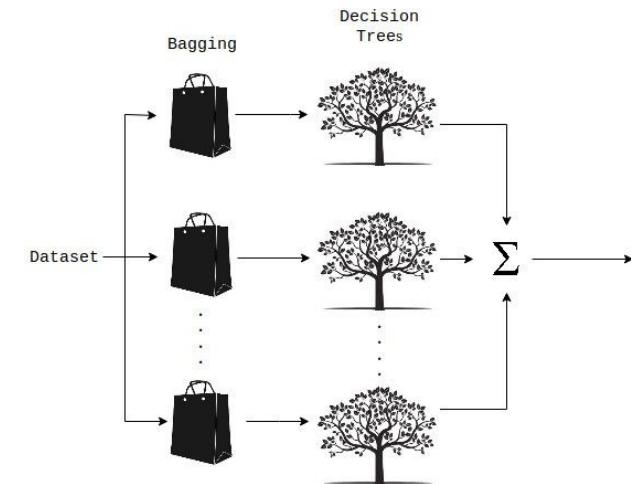
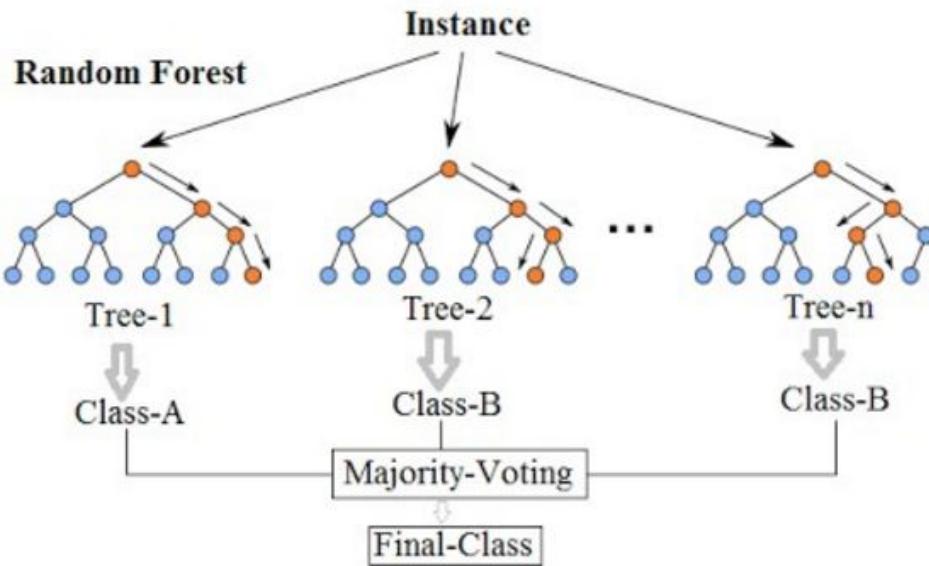


Random Forest Regression (RFR)



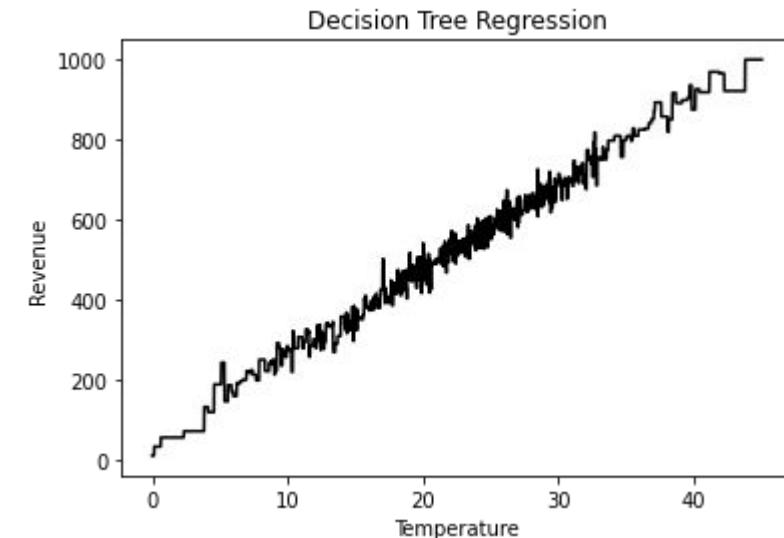
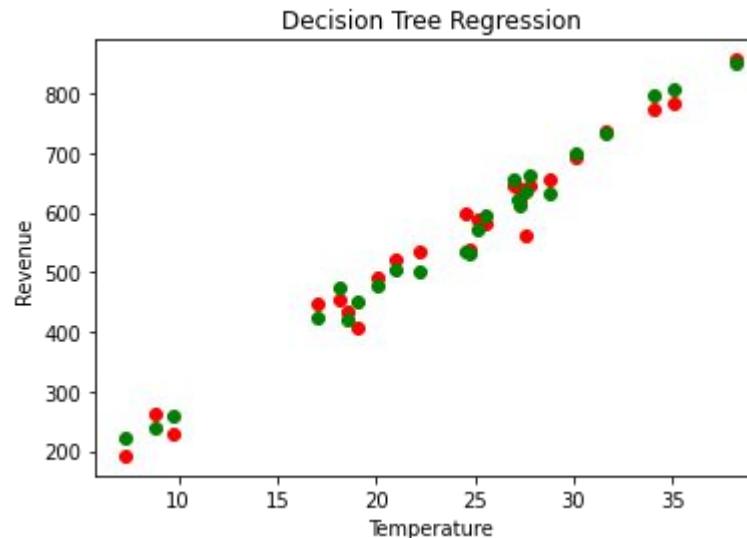
Random Forest

Random Forest Simplified

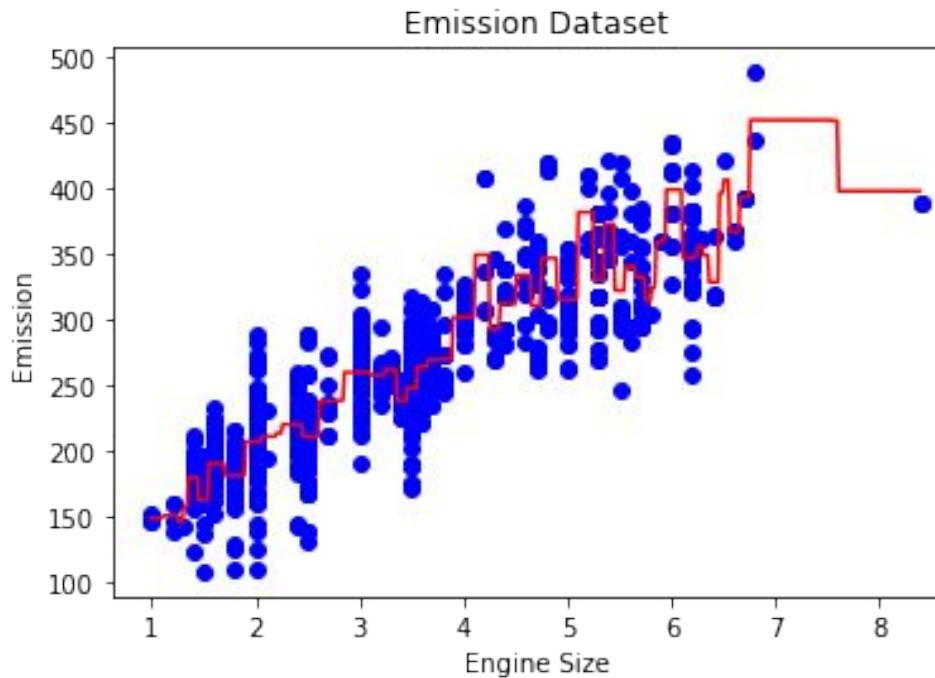




Random Forest Regression - Contoh

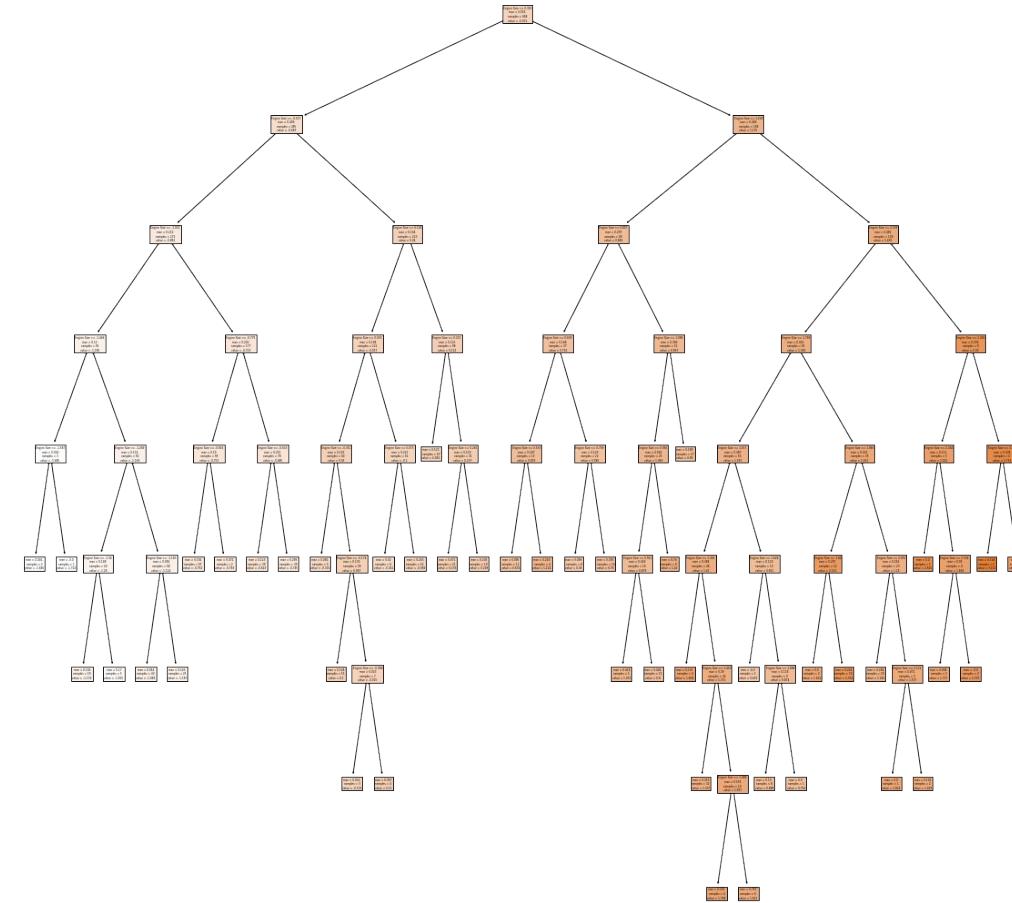


Aplikasi RFR pada Regresi Sederhana Emission Dataset



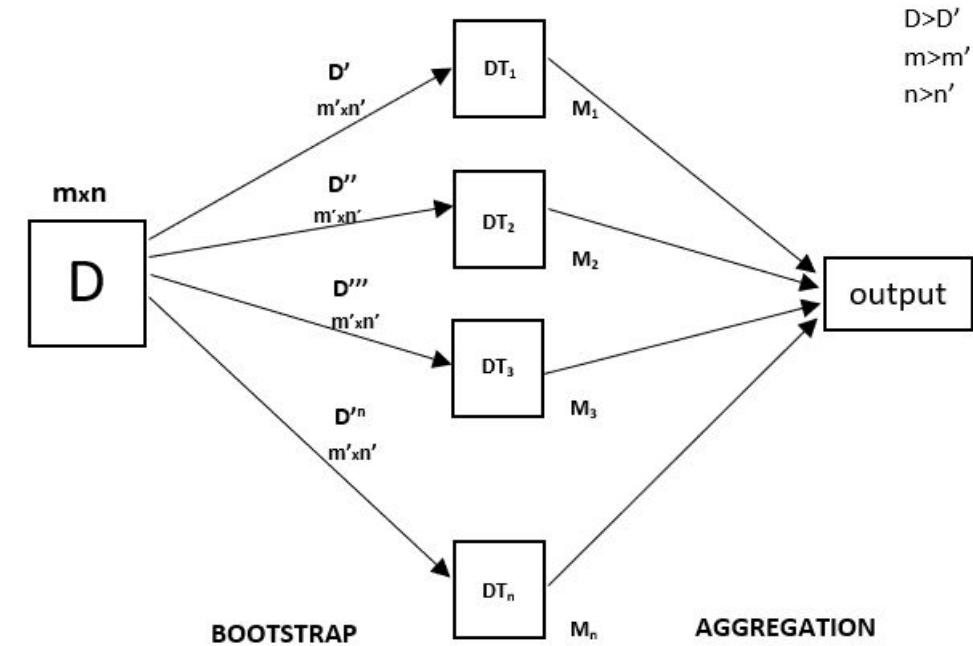
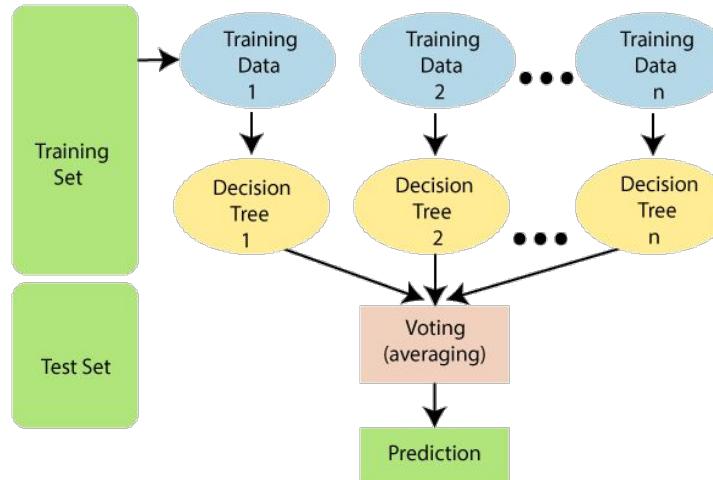
Hasil model dari aplikasi RFR pada Emission Dataset tampak pada Gambar 21, dengan garis putus-putus berwarna merah.

Visualisasi Struktur Tree RFR – Emission Dataset



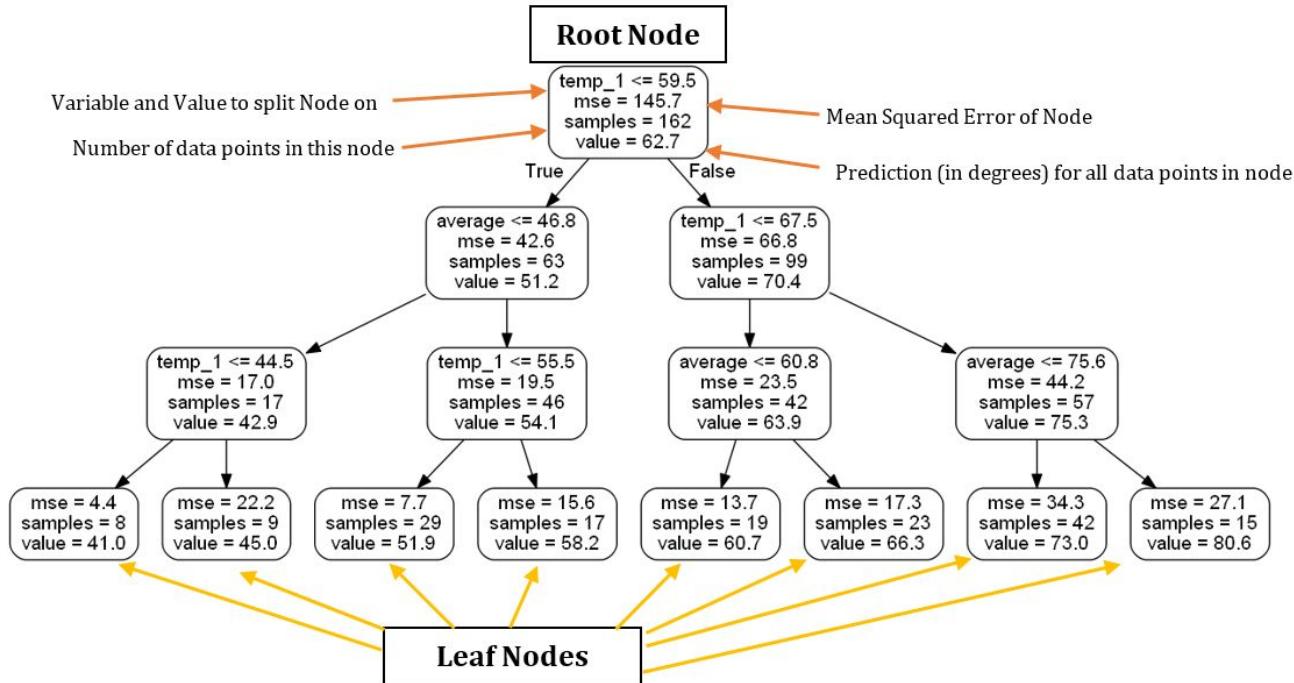
Visualisasi model yang dihasilkan oleh RFR pada Emission Dataset tampak pada Gambar 22.

Random Forest in Python





Random Forest in Python





Penggunaan Random Forest

- Random Forest jarang digunakan pada produksi karena algoritma lain menunjukkan hasil yang lebih baik.
- RF wajib digunakan untuk menguji hipotesis, karena akan memberikan insight yang berharga. Seringkali RF mengalahkan neural network yang kompleks untuk deteksi fraud
- Untuk penggunaan Klasifikasi
 - Fraud Detection (Classification) – terkadang mengalahkan neural network yang kompleks pada suatu tugas yang tidak jelas
 - Credit Scoring (Classification) – suatu solusi yang dibutuhkan pada perbankan. Banyak solusi menggunakan neural network kompleks untuk kebutuhan ini, tetapi dengan Random Forest yang sederhana seringkali diperoleh hasil yang sama
 - E-commerce case (Classification) – misal untuk memprediksi apakah konsumen akan menyukai produk atau tidak
 - Problem klasifikasi dengan suatu tabel data
- Untuk regresi, digunakan bila::
 - It is not a time series problem
 - The data has a non-linear trend and extrapolation is not crucial
- For example, Random Forest is frequently used in value prediction (value of a house or a packet of milk from a new brand)



Pro- Cons

Pros.



- Excellent prediction performance.
- Feature normalization is not required.
- No extensive parameter tuning.
- Easily handle a mixture of feature types.

Cons.



- Trained models are usually hard for human interpretation.
- A slow training process for high-dimensional tasks.



Lab

- Jalankan file Jupyter Notebook untuk Random Forest Regression



Home Page - Select or < x Random_Forest_Regre: x +
127.0.0.1:8888/notebooks/Random_Forest_Regression.ipynb

jupyter Random_Forest_Regression (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3

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 realistik 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]: # Mengambil "Engine Size" sebagai variabel independen (regressor)
X = cdf.iloc[:, 0].values
Mengambil "Emission" sebagai variabel dependen
y = cdf.iloc[:, 3].values
Reshape data karena hanya menggunakan satu fitur "Engine Size"
X = X.reshape(-1,1)
Reshape data karena hanya satu fitur
y = y.reshape(-1,1) ←

Proses splitting dataset pelatihan dan pengujian

In [11]: # Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0)

Normalisasi atau scaling dataset

In [12]: # Feature Scaling
from sklearn.preprocessing import StandardScaler
sc_X = StandardScaler()
sc_y = StandardScaler()
X = sc_X.fit_transform(X)
y = sc_y.fit_transform(y)

Pembuatan Model

In [13]: # Fitting Random Forest Regression to the dataset
from sklearn.ensemble import RandomForestRegressor
regressor = RandomForestRegressor(n_estimators = 100)
regressor.fit(X, y)

/usr/lib/python3.6/site-packages/ipykernel_launcher.py:4: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
after removing the cwd from sys.path.

Random Forest Regression

Library yang harus terinstal



Home Page - Select or × Random_Forest_Regre × +

127.0.0.1:8888/notebooks/Random_Forest_Regression.ipynb

jupyter Random_Forest_Regression (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

In [13]: # Fitting Random Forest Regression to the dataset
from sklearn.ensemble import RandomForestRegressor
regressor = RandomForestRegressor(n_estimators = 100)
regressor.fit(X, y)

/usr/lib/python3.6/site-packages/ipykernel_launcher.py:4: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
after removing the cwd from sys.path.

Out[13]: RandomForestRegressor()

Prediksi nilai baru dengan model yang telah dibentuk

In [14]: # Predicting a new result

y_pred = regressor.predict(sc_X.transform(np.array([[5.4]])))
#To transform 5.4 to the scaled X value, we first need to convert it into the array form
#Since the transform method of StandardScaler Library only accepts arrays

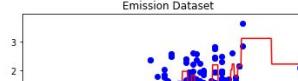
y_pred = sc_y.inverse_transform(y_pred)
#Now the prediction gives us the scaled value of y
#Thus we need inverse transformation of the scaled value for the real results

print(y_pred)
[372.16469389]

Visualisasi Hasil dalam nilai yang discaling:

In [15]: # Visualising the Random Forest Regression results (higher resolution)
X_grid = np.arange(min(X), max(X), 0.01)
X_grid = X_grid.reshape((len(X_grid), 1))
plt.scatter(X, y, color = 'blue')
plt.plot(X_grid, regressor.predict(X_grid), color = 'red')
plt.title('Emission Dataset')
plt.xlabel('Engine Size')
plt.ylabel('Emission')
plt.show()

Emission Dataset



Random Forest Regression



Home Page - Select or < Random_Forest_Regressor.ipynb

jupyter Random_Forest_Regression (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

[372.16469389]

Visualisasi Hasil dalam nilai yang discaling:

```
In [15]: # Visualising the Random Forest Regression results (higher resolution)
X_grid = np.arange(min(X), max(X), 0.01)
X_grid = X_grid.reshape((len(X_grid), 1))
plt.scatter(X, y, color = 'blue')
plt.plot(X_grid, regressor.predict(X_grid), color = 'red')
plt.title('Emission Dataset')
plt.xlabel('Engine Size')
plt.ylabel('Emission')
plt.show()
```

Emission Dataset

Visualisasi Hasil dalam nilai asalnya:

```
In [16]: # Visualising the Random Forest Regression results (higher resolution)
X_grid = np.arange(min(X), max(X), 0.01)
X_grid = X_grid.reshape((len(X_grid), 1))
sc_X = StandardScaler()
sc_y = StandardScaler()
sc_X.fit(X)
sc_y.fit(y)
X_grid = sc_X.transform(X_grid)
y_grid = sc_y.inverse_transform(X_grid)
regressor = RandomForestRegressor(n_estimators=10, random_state=0)
regressor.fit(X, y)
y_grid = regressor.predict(X_grid)
plt.scatter(sc_X.inverse_transform(X), sc_y.inverse_transform(y), color = 'blue')
plt.plot(sc_X.inverse_transform(X_grid), sc_y.inverse_transform(regressor.predict(X_grid)), color = 'red')
plt.title('Emission Dataset')
plt.xlabel('Engine Size')
plt.ylabel('Emission')
plt.show()
```

Emission Dataset

Random Forest Regression



Home Page - Select or Random_Forest_Regre +

127.0.0.1:8888/notebooks/Random_Forest_Regression.ipynb

jupyter Random_Forest_Regression (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Visualisasi Hasil dalam nilai asalnya:

```
In [16]: # Visualising the Random Forest Regression results (higher resolution)
X_grid = np.arange(min(X), max(X), 0.01)
X_grid = X_grid.reshape((len(X_grid), 1))
plt.scatter(sc_X.inverse_transform(X), sc_y.inverse_transform(y), color = 'blue')
plt.plot(sc_X.inverse_transform(X_grid), sc_y.inverse_transform(regressor.predict(X_grid)), color = 'red')
plt.title('Emission Dataset')
plt.xlabel('Engine Size')
plt.ylabel('Emission')
plt.show()
```

Emission Dataset

Visualisasi Tree (dengan nilai yang discaling):

```
In [17]: from sklearn import tree
plt.figure(figsize=(20,20))
graph = tree.plot_tree(regressor.estimators_[0], feature_names=['Engine Size'], filled=True)
```

Random Forest Regression



Home Page - Select or Random_Forest_Regre... +

127.0.0.1:8888/notebooks/Random_Forest_Regression.ipynb

jupyter Random_Forest_Regression (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 Logout

Visualisasi Tree (dengan nilai yang discaling):

```
In [17]: from sklearn import tree
plt.figure(figsize=(20,20))
graph = tree.plot_tree(regressor.estimators_[0], feature_names=['Engine Size'], filled=True)
```

Random Forest Regression



Home Page - Select or > Random_Forest_Regres > +

127.0.0.1:8888/notebooks/Random_Forest_Regression.ipynb

jupyter Random_Forest_Regression (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 Logout

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, misalkan 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 [18]:

```
from sklearn.metrics import r2_score

test_x = np.asarray(cdf[['ENGINESIZE']])
test_y = np.asarray(cdf[['CO2EMISSIONS']])
test_y_ = sc_y.inverse_transform(regressor.predict(sc_X.transform(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: 68.06
Residual sum of squares (MSE): 7266.74
R2-score: 0.78

In []:

In []:

Random Forest Regression

Evaluasi



Metriks Evaluasi





Error

	ENGINESIZE	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	
7	3.7	6	11.1	
8	3.7	6	11.6	
9	2.4	4	9.2	

Test

y

Actual values

232
255
267
212

Prediction
6 234
7 256
8 267
9 210

\hat{y}

↓
Predicted values

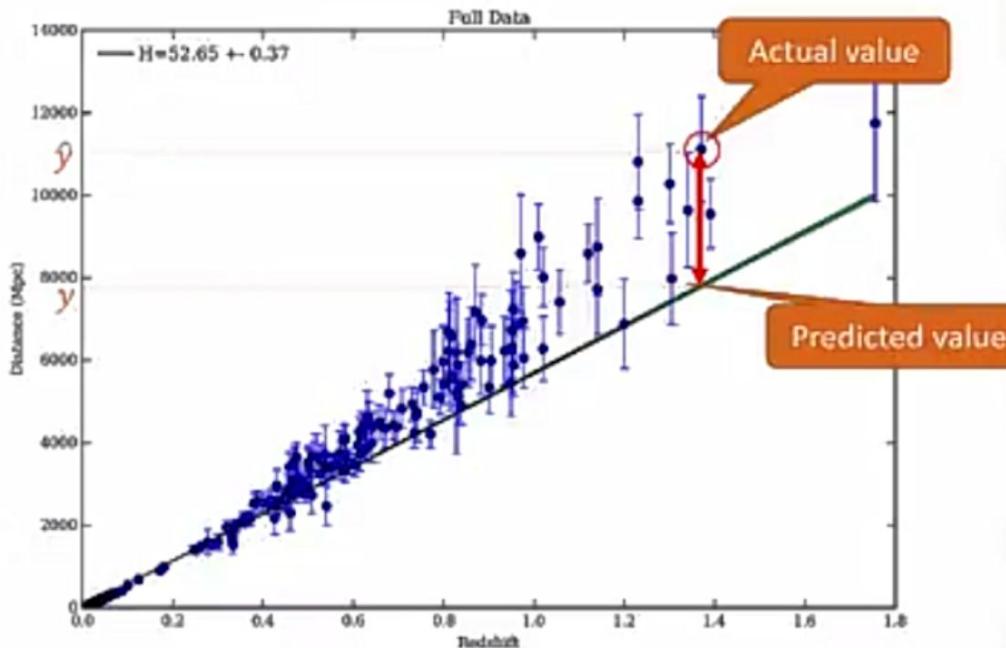
$$\text{Error} = \frac{(232 - 234) + (255 - 256) + \dots}{4}$$

$$\text{Error} = \frac{1}{n} \sum_{j=1}^n |y_j - \hat{y}_j|$$

- MAE
- MSE
- RMSE
- ...

- Metrik evaluasi yang paling umum digunakan adalah Error yang terdiri dari berbagai formula antara lain Mean Absolute Error sebagaimana pada Gambar 23.
- Selain itu terdapat Mean Squared Error, Root Mean Squared Error, dan lainnya.

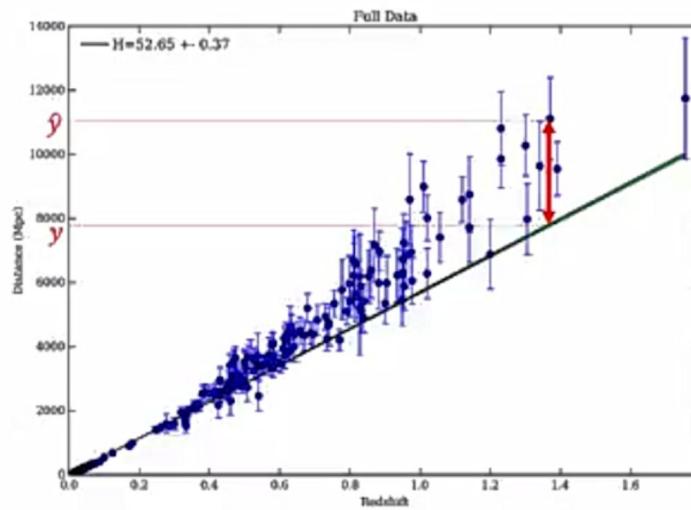
Error dari Model



Error: measure of how far the data is from the fitted regression line.

- Error didefinisikan sebagai ukuran sejauh mana data aktual terhadap garis regresi yang sudah dicocokan sebagaimana diilustrasikan pada Gambar 24.
- Garis regresi tersebut memuat nilai prediksi sedangkan titik-titik biru yang ada adalah nilai aktualnya.

Error dari Model



$$MAE = \frac{1}{n} \sum_{j=1}^n |y_j - \hat{y}_j|$$

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

$$RMSE = \sqrt{\frac{1}{n} \sum_{j=1}^n (y_j - \hat{y}_j)^2}$$

$$RAE = \frac{\sum_{j=1}^n |y_j - \hat{y}_j|}{\sum_{j=1}^n |y_j - \bar{y}|}$$

$$RSE = \frac{\sum_{j=1}^n (y_j - \hat{y}_j)^2}{\sum_{j=1}^n (y_j - \bar{y})^2}$$

$$R^2 = 1 - RSE$$

- MSE, MAE, RMSE lebih disukai digunakan untuk membandingkan kinerja antara model regresi yang berbeda.
- Menggunakan MSE jika nilainya tidak terlalu besar atau menggunakan MAE saat tidak ingin menghukum kesalahan prediksi yang besar
- R-Square dan Adjusted R-Square lebih baik digunakan untuk menjelaskan kesesuaian model. Harus selalu ada keseimbangan karena nilai R² yang sangat rendah berarti model underfitting sedangkan nilai R² yang sangat tinggi berarti model overfitting.



Perbandingan





Regresi Linier (LR) vs DTR

- DTR mendukung non linearitas, di mana RL hanya mendukung solusi linier.
- Ketika ada sejumlah besar fitur dengan lebih sedikit kumpulan data (dengan noise rendah), regresi linier dapat mengungguli DTR/Random Forest Regression (RFR). Dalam kasus umum, DTR akan memiliki akurasi rata-rata yang lebih baik.
- Untuk variabel bebas kategorikal, DTR lebih baik daripada regresi linier.
- DTR menangani kolinearitas lebih baik daripada LR.



RL vs SVR

- SVR mendukung solusi linier dan non-linier menggunakan trik kernel.
- SVR menangani outlier lebih baik daripada RL.
- Keduanya berkinerja baik ketika data pelatihan lebih sedikit, dan ada banyak fitur.



DTR vs RFR

- RFR adalah kumpulan DT, suara (vote) mayoritas atau rata-rata dari forest dipilih sebagai keluaran yang diprediksi.
- RFR akan kurang rentan terhadap overfitting daripada DTR, dan memberikan solusi yang lebih general.
- RFR lebih robust dan akurat daripada pohon keputusan.



Perbandingan

Regression Model	Advantages	Disadvantages
Linear Regression	<ol style="list-style-type: none">1. Works well irrespective of the dataset size.2. Gives information about the relevance of features.	<ol style="list-style-type: none">1. The assumptions of Linear Regression.
Polynomial Regression	<ol style="list-style-type: none">1. Works on any size of the dataset.2. Works very well on non-linear problems.	<ol style="list-style-type: none">1. We need to choose the right polynomial degree for good bias/ variance tradeoff.
Support Vector Regression	<ol style="list-style-type: none">1. Easily adaptable.2. Works very well on non-linear problems.3. Not biased by outliers (object that deviates significantly from the rest).	<ol style="list-style-type: none">1. Compulsory to apply feature scaling.2. Not well known.3. Difficult to understand.
Decision Tree Regression	<ol style="list-style-type: none">1. Interpretability.2. Works well on both linear and non-linear problems.3. No need to apply feature scaling.	<ol style="list-style-type: none">1. Poor results on small datasets.2. Overfitting can easily occur.
Random Forest Regression	<ol style="list-style-type: none">1. Powerful.2. Accurate.3. Good performance on many problems including non-linear.	<ol style="list-style-type: none">1. No interpretability.2. Overfitting can easily occur.3. We need to choose the number of trees.



Tools / Lab Online

- Scikit-Learn
- Jupyter Notebook
- Conda / Anaconda
- Google Colaboratory
- PyPI (pip)



Summary

- Regresi adalah prediksi nilai kontinu atau numerik dari variabel tak bebas berdasarkan variabel bebas atau predictor.
- Regresi ada dua tipe sederhana atau satu variabel dan variabel jamak.
- Masing-masing regresi tersebut terdapat dua pendekatan terhadap data: linier dan non-linier.
- Evaluasi pemodelan regresi menggunakan metrik berdasar error seperti MAE, MSE, maupun kecocokan model terhadap data seperti R^2
- Terdapat banyak algoritma regresi yang dapat digunakan dengan kelebihan dan kekurangan masing-masing.



Quiz / Tugas

Quiz dapat diakses melalui <https://spadadikti.id/>



Terima kasih