

LAPORAN ANALISIS PREDIKTIF ENERGI DARI LIMBAH PERTANIAN MENGGUNAKAN RANDOM FOREST REGRESSOR

Laporan ini Dibuat untuk Memenuhi Tugas Ujian Tengah Semester
Matakuliah Machine Learning



Dosen Pengampu: Jusmardi, S.Kom., M.Pd.T

Disusun oleh kelompok 2 yang beranggotakan:

- | | |
|---------------------------|----------|
| 1. Aryanahta Putra | 22346002 |
| 2. Davit Zarly | 22346004 |
| 3. Ade Adinda | 22346028 |
| 4. Jovanka Sabila Pertiwi | 22346033 |
| 5. Muhammad Irzan Ali | 22346039 |

PROGRAM STUDI INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS NEGERI PADANG

2025/2026

PENDAHULUAN

Indonesia sebagai negara agraris memiliki produksi pertanian dan perkebunan yang sangat melimpah sepanjang tahun. Kegiatan ini menghasilkan limbah organik dalam jumlah besar, seperti sekam padi, bonggol jagung, ampas tebu, dan limbah kelapa sawit. Sayangnya, sebagian besar limbah ini belum dimanfaatkan secara maksimal dan sering kali dibuang begitu saja, padahal limbah tersebut menyimpan potensi energi terbarukan yang cukup besar. Berdasarkan penelitian terbaru, potensi biomassa di Indonesia diperkirakan mencapai 146,7 juta ton per tahun, setara dengan kapasitas listrik sebesar 24,46 GW yang dapat dimanfaatkan untuk mendukung transisi energi nasional menuju energi bersih.

Kebutuhan akan energi di Indonesia terus meningkat seiring dengan pertumbuhan ekonomi dan jumlah penduduk. Di sisi lain, ketergantungan terhadap energi berbasis fosil menimbulkan berbagai permasalahan, seperti penurunan cadangan energi, emisi karbon, dan kerusakan lingkungan. Oleh karena itu, pemanfaatan energi terbarukan menjadi solusi strategis. Salah satu bentuk energi terbarukan yang berpotensi besar adalah energi biomassa, terutama yang berasal dari limbah pertanian.

Untuk mengoptimalkan pemanfaatan limbah pertanian sebagai energi, diperlukan metode yang efektif dalam memprediksi potensi energi yang dapat dihasilkan dari berbagai jenis limbah tersebut. Machine Learning, khususnya algoritma **Random Forest Regressor**, dapat menjadi solusi untuk membangun model prediksi yang akurat dan andal, sehingga potensi energi dari limbah pertanian dapat diestimasi secara sistematis dan digunakan sebagai dasar pengambilan keputusan.

Link video bisa diakses menggunakan link berikut:

Untuk google collab bisa diakses menggunakan link di bawah ini :

PEMBAHASAN

1. Pra-pemrosesan Data

1.1 Impor Library

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error,
mean_squared_error, r2_score
from sklearn.preprocessing import LabelEncoder, StandardScaler
```

1) import pandas as pd

Mengimpor pustaka pandas yang digunakan untuk memanipulasi dan menganalisis data dalam format tabular, seperti CSV, Excel, dan lainnya. Pandas menyediakan berbagai fungsi untuk membersihkan, mengolah, dan memvisualisasikan data.

2) import numpy as np

Pustaka ini digunakan untuk komputasi numerik. Numpy menyediakan struktur data array multidimensi dan fungsi matematika untuk melakukan operasi pada array, yang sangat berguna untuk analisis data berbasis angka.

3) import matplotlib.pyplot as plt

Pustaka ini digunakan untuk visualisasi data dengan membuat grafik, seperti grafik batang, garis, dan lain-lain. Dapat digunakan untuk mengeksplorasi pola dalam data.

4) import seaborn as sns

Seaborn adalah pustaka visualisasi berbasis matplotlib yang menyediakan cara yang lebih mudah dan estetik untuk membuat grafik statistik.

5) `from sklearn.model_selection import train_test_split`

Digunakan untuk membagi dataset menjadi dua bagian: data latih (training) dan data uji (testing), yang diperlukan untuk mengembangkan model machine learning.

6) `from sklearn.ensemble import RandomForestRegressor`

Pustaka ini digunakan untuk mengimpor algoritma Random Forest, yang merupakan teknik ensemble learning yang dapat digunakan untuk regresi atau klasifikasi. Random Forest digunakan untuk memprediksi estimasi energi berdasarkan input data.

7) `from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score`

Digunakan untuk mengukur kinerja model setelah pelatihan, menggunakan metrik seperti Mean Absolute Error (MAE), Mean Squared Error (MSE), dan R² Score.

8) `from sklearn.preprocessing import LabelEncoder, StandardScaler`

Pustaka ini digunakan untuk melakukan prapemrosesan data. LabelEncoder digunakan untuk mengubah data kategorikal menjadi format numerik, sementara StandardScaler digunakan untuk normalisasi data numerik, seperti berat dan kadar air.

1.2 Memuat Dataset

1. `df = pd.read_csv('dataset_limbah_energi.csv')`

Fungsi `read_csv()` dari pandas digunakan untuk membaca file CSV yang berisi data tentang limbah pertanian dan estimasi energi. Data yang dibaca dimasukkan ke dalam sebuah DataFrame yang memungkinkan analisis lebih lanjut.

1.3 Menangani Data yang Hilang, Encoding Kolom Kategori, Normalisasi Kolom Numerik, dan Menampilkan Data yang Telah Diproses

Berdasarkan hasil pemeriksaan menggunakan fungsi `isnull().sum()`, tidak ditemukan nilai yang hilang pada seluruh kolom dalam dataset. Semua entri pada kolom 'Jenis Limbah', 'Berat (kg)', 'Kadar Air (%)', dan 'Estimasi Energi (kWh)' memiliki data yang lengkap. Dengan demikian, tidak diperlukan teknik imputasi atau penghapusan data. Tidak ada nilai yang hilang dalam semua kolom.

Kolom 'Jenis Limbah' berisi data kategori dalam bentuk teks, seperti "Organik", "Anorganik", dan sebagainya. Agar dapat digunakan dalam proses analisis berbasis numerik, kolom ini dikodekan menggunakan `LabelEncoder`. Setiap kategori diberikan label numerik unik dan hasilnya disimpan dalam kolom yang sama.

Contoh:

- Organik \rightarrow 0
- Anorganik \rightarrow 1
- Logam \rightarrow 2

Kolom 'Berat (kg)' dan 'Kadar Air (%)' dinormalisasi menggunakan metode `StandardScaler`, yaitu dengan rumus Z-score normalization:

$$z = \frac{(x - \mu)}{\sigma}$$

Proses ini bertujuan untuk menyetarakan skala antar fitur, agar model tidak bias terhadap fitur dengan nilai besar. Hasil normalisasi tetap disimpan dalam kolom aslinya.

- Menampilkan Informasi Dataset

Setelah proses encoding dan normalisasi selesai, struktur dataset ditampilkan menggunakan fungsi `df.info()`. Fungsi ini memperlihatkan jumlah entri non-null, tipe data tiap kolom, serta konsumsi memori dataset.

- Contoh Data yang Diproses

Lima baris pertama dari dataset ditampilkan menggunakan `df.head()`. Dari hasil ini dapat diamati bahwa:

- a. Kolom 'Jenis Limbah' telah dikonversi menjadi angka (0, 1, 2, dst),
- b. Kolom 'Berat (kg)' dan 'Kadar Air (%)' telah bernilai dalam bentuk standar Z-score (rata-rata 0 dan deviasi standar 1),
- c. Kolom 'Estimasi Energi (kWh)' tetap dalam bentuk aslinya karena merupakan target/output yang tidak perlu diubah skalanya.

```
# Cek missing value
print(df.isnull().sum())

# Encoding variabel kategori (Jenis Limbah)
le = LabelEncoder()
df['Jenis Limbah'] = le.fit_transform(df['Jenis Limbah'])

# Normalisasi berat dan kadar air
scaler = StandardScaler()
df[['Berat (kg)', 'Kadar Air (%)']] = scaler.fit_transform(df[['Berat (kg)', 'Kadar Air (%)']])

# Tampilkan informasi struktur data
df.info()

# Tampilkan 5 baris pertama setelah diproses
df.head()
```

Output dan penjelasan:

1. Informasi Dataset yang Diproses

Dataset terdiri dari 2000 baris dan 4 kolom: 'Jenis Limbah', 'Berat (kg)', 'Kadar Air (%)', dan 'Estimasi Energi (kWh)'. Semua kolom telah bertipe numerik dan tidak ada nilai yang hilang. Dataset ini memerlukan memori sebesar ± 64 KB, dan siap digunakan untuk proses analisis atau pelatihan model.

2. Contoh Data yang Telah Diproses

Contoh lima data teratas menunjukkan bahwa kolom 'Jenis Limbah' sudah dalam bentuk angka kategori, sementara 'Berat (kg)' dan 'Kadar Air (%)' telah distandarkan. Ini menunjukkan bahwa seluruh proses transformasi telah berhasil dilakukan dan data siap untuk analisis lanjutan seperti regresi atau klasifikasi.

```
... Jenis Limbah      0
    Berat (kg)        0
    Kadar Air (%)     0
    Estimasi Energi (kwh) 0
    dtype: int64
    <class 'pandas.core.frame.DataFrame'>
    RangeIndex: 2000 entries, 0 to 1999
    Data columns (total 4 columns):
    #   Column              Non-Null Count  Dtype
    ---  ---
    0   Jenis Limbah         2000 non-null   int64
    1   Berat (kg)           2000 non-null   float64
    2   Kadar Air (%)        2000 non-null   float64
    3   Estimasi Energi (kwh) 2000 non-null   float64
    dtypes: float64(3), int64(1)
    memory usage: 62.6 KB

...

```

	Jenis Limbah	Berat (kg)	Kadar Air (%)	Estimasi Energi (kWh)
0	7	-0.344948	-1.647673	758.29
1	2	1.605917	1.090811	1360.26
2	4	-1.553512	-0.779269	126.66
3	1	1.056166	1.220642	747.19
4	1	0.313369	1.423556	531.73

2. Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) dilakukan untuk memahami karakteristik data, melihat distribusi fitur, serta mengidentifikasi pola atau hubungan antar variabel yang dapat berguna dalam proses modeling. Analisis ini mencakup visualisasi distribusi berat limbah, kadar air limbah, korelasi antar fitur, serta identifikasi fitur penting berdasarkan tingkat kontribusinya.

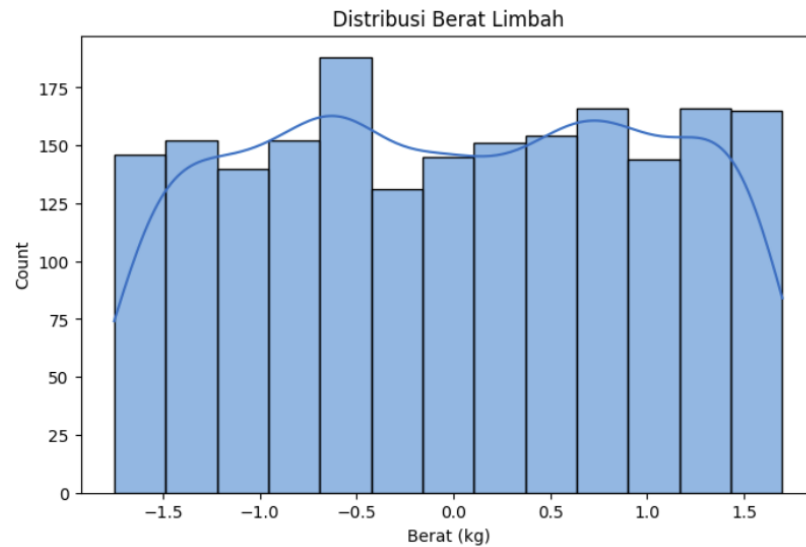
2.1 Visualisasi Distribusi Berat Limbah

```
plt.figure(figsize=(8,5))
sns.histplot(df['Berat (kg)'], kde=True)
plt.title('Distribusi Berat Limbah')
plt.show()
```

Visualisasi ini menggabungkan histogram dan kurva KDE (Kernel Density Estimation) untuk memberikan gambaran tentang pola

distribusi data pada kolom "Berat (kg)".

- Histogram (batang biru transparan): Menunjukkan jumlah atau frekuensi data yang jatuh dalam rentang berat tertentu.
- KDE (garis lengkung biru): Memberikan gambaran halus tentang bentuk distribusi data, membantu memahami apakah distribusi tersebut simetris, miring, atau multimodal.



Interpretasi:

1. Pola Distribusi:

- Data tampak relatif merata dengan sedikit fluktuasi di tengah, menunjukkan distribusi tidak terlalu condong ke kiri (negatif) maupun ke kanan (positif).
- Kurva KDE tidak menunjukkan puncak tunggal yang mencolok, yang mengindikasikan data mungkin mendekati distribusi uniform (seragam) daripada distribusi normal.

2. Rentang Data:

- Nilai berat berada di kisaran sekitar -1.7 kg hingga 1.7 kg, kemungkinan besar ini adalah data hasil standarisasi (z-score), karena nilai negatif masih muncul untuk berat yang logis.

3. Kepadatan dan Frekuensi:

- Setiap bar mewakili kelompok berat yang memiliki sekitar 130–190 kemunculan, menunjukkan bahwa jumlah data tersebar cukup merata dalam tiap kelompok.

4. Tidak Ada Outlier Ekstrem:

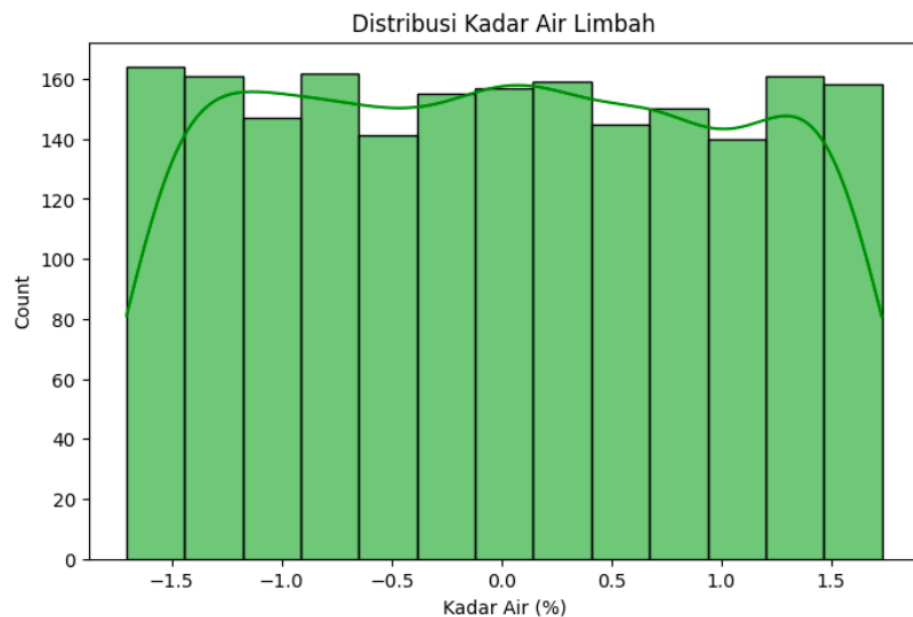
- Tidak tampak batang yang terpencil jauh dari kelompok utama, yang berarti tidak ada outlier berat yang mencolok.

2.2 Visual Distribusi Kadar Air Limbah

```
# Visualisasi distribusi kadar air
plt.figure(figsize=(8,5))
sns.histplot(df['Kadar Air (%)'], kde=True,
             color='green')
plt.title('Distribusi Kadar Air Limbah')
plt.show()
```

Grafik ini menyajikan distribusi kadar air limbah dengan menggabungkan histogram dan kurva KDE (Kernel Density Estimation) yang ditampilkan berwarna hijau.

- Histogram (batang hijau): Mewakili jumlah data yang masuk dalam setiap kelompok kadar air (%). Ini menunjukkan frekuensi kemunculan data dalam rentang tertentu.
- Kurva KDE (garis hijau halus): Menggambarkan estimasi distribusi data secara kontinu, membantu memahami bentuk dan pola persebarannya secara lebih halus.



Interpretasi:

1. Pola Distribusi:

- Histogram terlihat cukup merata dengan ketinggian bar yang tidak terlalu jauh berbeda. Ini menandakan distribusi kadar air cenderung uniform (seragam).
- Kurva KDE naik di awal, lalu menurun sedikit dan stabil, menunjukkan tidak ada puncak ekstrem atau outlier besar yang mendominasi data.

2. Rentang Nilai:

- Nilai kadar air berkisar antara -1.5 hingga 1.5, mengindikasikan bahwa data kemungkinan telah melalui proses standarisasi (misalnya z-score), di mana nilai diubah menjadi deviasi terhadap rata-rata.

3. Keseimbangan Data:

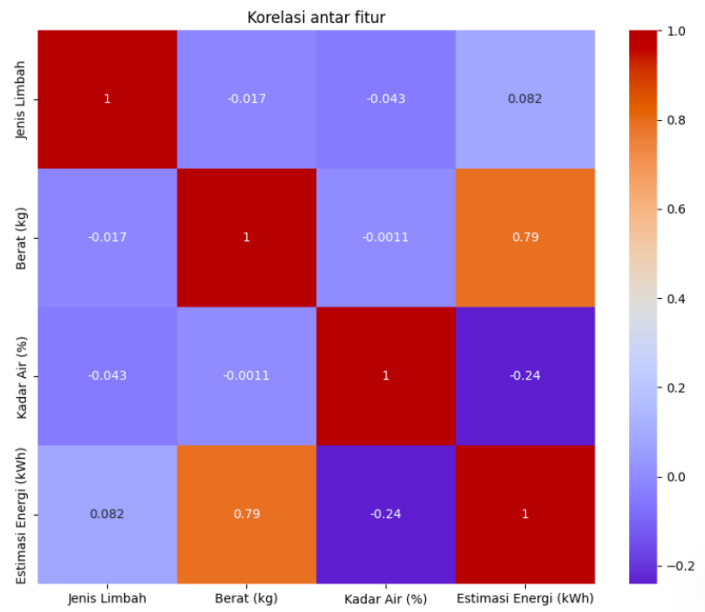
- Tidak ada lonjakan atau penurunan tajam di histogram atau KDE, yang mengindikasikan tidak ada dominasi nilai tertentu. Artinya, model machine learning yang digunakan nantinya tidak akan terlalu bias terhadap rentang nilai kadar air tertentu.

2.3 Visual Distribusi Kadar Air Limbah

```
plt.figure(figsize=(10,8))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
plt.title('Korelasi antar fitur')
plt.show()
```

Visualisasi ini menunjukkan matriks korelasi antar fitur dalam dataset limbah menggunakan warna dari palet coolwarm. Korelasi ini diukur menggunakan koefisien Pearson, yang nilainya berkisar antara -1 hingga 1:

- Nilai mendekati 1: korelasi positif kuat.
- Nilai mendekati -1: korelasi negatif kuat.
- Nilai mendekati 0: hampir tidak ada korelasi linear.



Interpretasi Setiap Hubungan:

1. Berat (kg) dan Estimasi Energi (kWh): 0.79

Korelasi positif kuat, artinya semakin berat limbah, semakin besar pula estimasi energi yang dihasilkan. Hubungan ini masuk akal karena semakin banyak massa limbah yang diproses, energi yang dibutuhkan atau dihasilkan cenderung meningkat.

2. Kadar Air (%) dan Estimasi Energi (kWh): -0.24

Korelasi negatif lemah, menunjukkan bahwa kadar air yang lebih tinggi mungkin menurunkan efisiensi energi. Air dalam limbah bisa menyerap panas dan menurunkan potensi energi.

3. Jenis Limbah dengan fitur lain: korelasi sangat rendah

Misalnya, dengan Estimasi Energi (kWh): 0.082, dan dengan Berat (kg): -0.017. Hal ini mengindikasikan bahwa jenis limbah tidak memiliki hubungan linear yang signifikan terhadap nilai numerik lainnya dalam dataset ini.

4. Kadar Air (%) dan Berat (kg): -0.0011

Hampir tidak ada korelasi, artinya kadar air tidak bergantung pada berat total limbah.

3. Pemilihan Algoritma dan Pelatihan Model

3.1 Pemilihan Algoritma

Pada penelitian ini, algoritma yang digunakan adalah Random Forest Regressor, salah satu metode ensemble learning berbasis decision tree. Random Forest bekerja dengan membangun sejumlah pohon keputusan (decision tree) secara acak dan menggabungkan hasil prediksi dari setiap pohon untuk menghasilkan prediksi akhir yang lebih akurat dan stabil. Keunggulan utama dari algoritma ini adalah kemampuannya dalam:

- Mengurangi risiko overfitting melalui proses bagging,
- Menangani data non-linear dengan baik,
- Bekerja secara efektif pada dataset dengan fitur yang kompleks dan saling berinteraksi.

3.2 Pembentukan Fitur dan Target

Pemodelan dimulai dengan memisahkan fitur (variabel independen) dan target (variabel dependen). Pada kasus ini, target yang ingin diprediksi adalah kolom **Estimasi Energi (kWh)**.

```
X = df.drop('Estimasi Energi (kWh)', axis=1)
y = df['Estimasi Energi (kWh)']
```

- X menyimpan semua kolom fitur yang akan digunakan untuk pelatihan model.
- y menyimpan nilai target, yaitu estimasi energi dalam satuan kilowatt-jam (kWh).

3.3 Pembagian Data Latih dan Uji

Data dibagi menjadi dua bagian, yaitu **data latih (training)** dan **data uji (testing)** dengan perbandingan 80:20. Tujuan pembagian ini adalah untuk melatih model pada sebagian data dan mengujinya pada data yang belum pernah dilihat oleh model, sehingga dapat dievaluasi performanya.

```
X_train, X_test, y_train, y_test = train_test_split(X,
y, test_size=0.2, random_state=42)
```

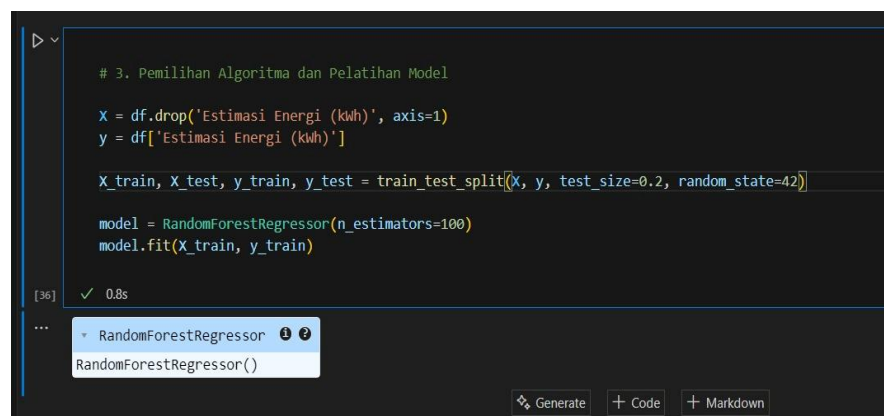
- `test_size=0.2` berarti 20% data digunakan untuk pengujian.
- `random_state=42` digunakan untuk memastikan hasil pembagian data tetap konsisten setiap kali dijalankan.

3.4 Pelatihan Model dengan Random Forest Regressor

Setelah data dibagi, langkah selanjutnya adalah melatih model menggunakan algoritma **Random Forest Regressor**. Model ini dibuat dengan 100 pohon keputusan sebagai parameter awal.

```
model = RandomForestRegressor(n_estimators=100,
                              random_state=42)
model.fit(X_train, y_train)
# Menampilkan model yang telah dilatih
model
```

- `n_estimators=100` menunjukkan jumlah pohon dalam ensemble.
- `fit()` adalah fungsi untuk melatih model dengan data latih.
- Pemanggilan model di akhir sel akan menampilkan detail model yang telah dilatih (berlaku di Jupyter Notebook atau Google Colab).



The screenshot shows a Jupyter Notebook interface. The code cell contains the following Python code:

```
# 3. Pemilihan Algoritma dan Pelatihan Model

X = df.drop('Estimasi Energi (kwh)', axis=1)
y = df['Estimasi Energi (kwh)']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

model = RandomForestRegressor(n_estimators=100)
model.fit(X_train, y_train)
```

Below the code cell, the output is displayed as a string representation of the `RandomForestRegressor` object:

```
[36] ✓ 0.8s
...
RandomForestRegressor()
```

At the bottom right of the notebook interface, there are buttons for "Generate", "+ Code", and "+ Markdown".

Output tersebut bukan hasil prediksi atau evaluasi, melainkan hanya representasi dari objek model yang sudah dilatih, yaitu: `RandomForestRegressor(...)` → menunjukkan bahwa objek ini adalah model regresi berbasis Random Forest dari pustaka `sklearn`.

4. Evaluasi Model

Setelah model dilatih, langkah selanjutnya adalah mengevaluasi performa model terhadap data uji. Evaluasi dilakukan dengan menghitung tiga metrik umum pada regresi:

```
y_pred = model.predict(X_test)

mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Absolute Error (MAE): {mae:.2f}")
print(f"Mean Squared Error (MSE): {mse:.2f}")
print(f"R2 Score: {r2:.2f}")
```

1. Mean Absolute Error (MAE):

Mengukur rata-rata selisih absolut antara nilai aktual dan prediksi. MAE memberikan informasi seberapa besar kesalahan rata-rata model dalam satuan asli (misalnya kWh). Semakin kecil nilai MAE, semakin baik model.

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i|$$

2. Mean Squared Error (MSE):

Mirip dengan MAE, tetapi menghitung selisih kuadrat. MSE memberi penalti lebih besar terhadap kesalahan yang besar (karena dikuadratkan), sehingga lebih sensitif terhadap outlier dibanding MAE. Nilainya juga dalam satuan kuadrat (misalnya kWh²).

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

3. R² Score (Koefisien Determinasi)

Mengukur seberapa baik model menjelaskan variasi data. Nilainya berkisar antara 0 hingga 1. **Semakin mendekati 1**, semakin baik model dalam menjelaskan variasi data.

$$R^2 = 1 - \frac{\sum (y_i - \hat{y}_i)^2}{\sum (y_i - \bar{y})^2}$$

```
# 4. Evaluasi Model

y_pred = model.predict(X_test)

mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Absolute Error (MAE): {mae:.2f}")
print(f"Mean Squared Error (MSE): {mse:.2f}")
print(f"R2 Score: {r2:.2f}")
```

✓ 0.1s

Mean Absolute Error (MAE): 24.12
Mean Squared Error (MSE): 1170.90
R2 Score: 0.99

- Mean Absolute Error (MAE): 24.12 artinya, model rata-rata meleset sebesar 24.12 kWh dari nilai yang seharusnya. Semakin kecil MAE, semakin baik. Dalam konteks energi, ini sangat tergantung pada skala datanya.
- Mean Squared Error (MSE): 1170.90 berarti rata-rata kuadrat kesalahan prediksi adalah sebesar itu, dalam satuan kWh².
- R² Score (Koefisien Determinasi): 0.99 berarti 99% variasi dalam nilai "Estimasi Energi (kWh)" dapat dijelaskan oleh model. Ini merupakan indikasi bahwa model sangat akurat.

5. Validasi Model

Validasi model penting untuk memastikan prediksi yang akurat dan konsisten pada data baru. Metode Cross Validation sering dipakai untuk menguji performa model secara menyeluruh dengan membagi data menjadi beberapa bagian. Selain itu, analisis error dan fitur penting membantu memahami kualitas prediksi dan faktor utama yang memengaruhi hasil model.

```
# VALIDASI MODEL DENGAN CROSS VALIDATION
from sklearn.model_selection import
    cross_val_score

# Menggunakan cross-validation 5-fold
cv_scores = cross_val_score(model, X, y,
    cv=5, scoring='r2')

print("\nHasil Cross Validation (R2 tiap
    fold):", cv_scores)
print("Rata-rata R2:", cv_scores.mean())

# ANALISIS DISTRIBUSI ERROR (RESIDUAL)
import matplotlib.pyplot as plt
import seaborn as sns

residuals = y_test - y_pred

plt.figure(figsize=(8,5))
sns.histplot(residuals,          kde=True,
    color='red')
plt.title('Distribusi Error Prediksi
    (Residuals)')
plt.xlabel('Error')
plt.ylabel('Frekuensi')
plt.show()

# FEATURE IMPORTANCE
importances = model.feature_importances_
feature_names = X.columns

plt.figure(figsize=(8,5))
sns.barplot(x=importances,
    y=feature_names)
plt.title('Pentingnya Fitur dalam
    Prediksi Energi')
plt.xlabel('Importance Score')
plt.ylabel('Fitur')
plt.show()
```


1. Cross Validation (5-Fold):

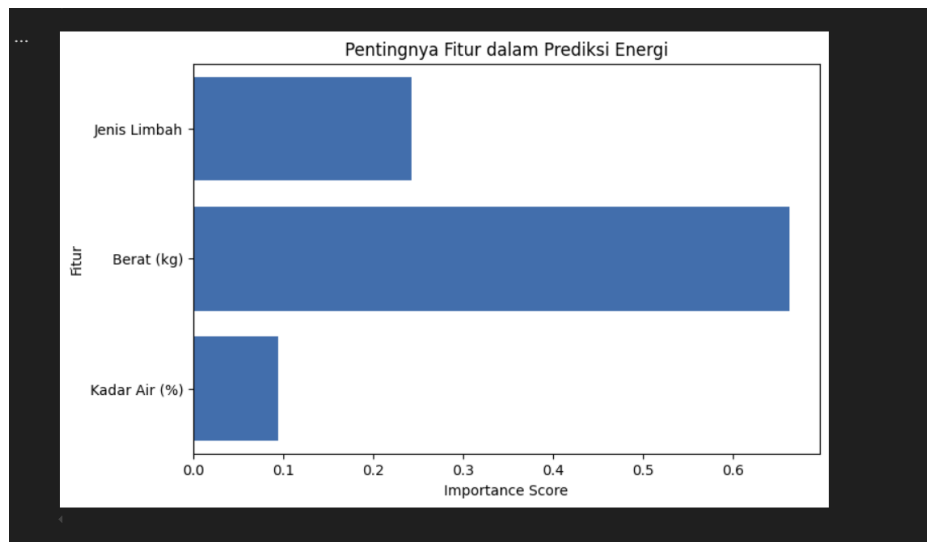
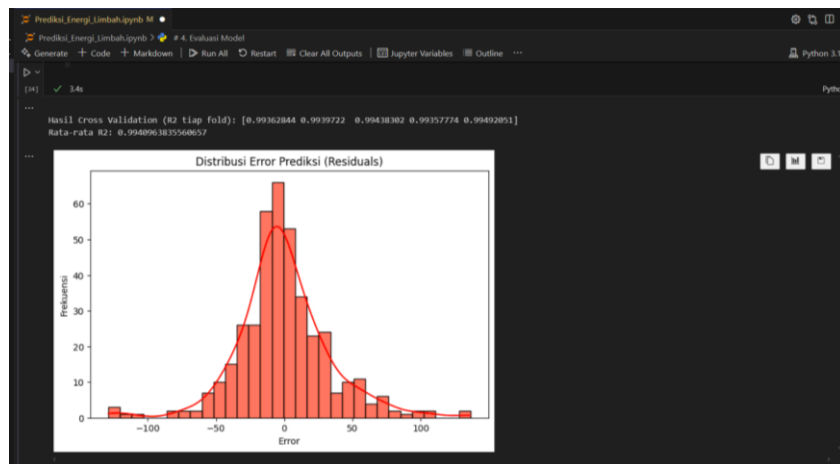
Menggunakan `cross_val_score()` dengan 5-fold untuk menguji kestabilan dan akurasi model menggunakan metrik **R² Score**. Hasil berupa 5 skor R² dan rata-ratanya menunjukkan performa model secara keseluruhan. Hasil program kami konsisten tinggi: **rata-rata R² = 0.9941**, artinya model mampu menjelaskan sekitar **99,41% variasi dalam data**, yang sangat baik untuk prediksi energi dari limbah.

2. Distribusi Error (Residual):

Menghitung residual = $y_{\text{test}} - y_{\text{pred}}$ lalu memvisualisasikannya dalam histogram. Pola distribusi normal dan simetris di sekitar nol menandakan prediksi akurat dan tidak bias. Grafik tampak seperti lonceng (normal distribution) dengan puncak di sekitar nol. Distribusi error simetris dan mengelompok di sekitar nol, artinya sebagian besar prediksi cukup dekat dengan nilai sebenarnya. Tidak ada outlier ekstrem yang mencolok, sehingga model stabil dan tidak bias ke arah tertentu (over atau under predicting). Ini mendukung bahwa model memiliki akurasi yang tinggi dan error yang wajar.

3. Feature Importance (Random Forest)

Mengambil `model.feature_importances_` untuk melihat kontribusi tiap fitur terhadap prediksi. Visualisasi bar chart menunjukkan fitur mana yang paling berpengaruh terhadap keluaran model. Grafik *feature importance* menunjukkan seberapa besar kontribusi masing-masing fitur input terhadap hasil prediksi model Random Forest. Berdasarkan hasil analisis, fitur **berat limbah (kg)** memberikan kontribusi terbesar, yaitu sekitar 0,65. Hal ini sangat logis mengingat semakin berat limbah, maka semakin besar pula potensi energi yang dapat dihasilkan. Selanjutnya, **jenis limbah** menyumbang sekitar 0,25 terhadap hasil prediksi. Meskipun kontribusinya tidak sebesar berat, fitur ini tetap berpengaruh karena jenis limbah berperan dalam menentukan kualitas bahan bakar atau energi yang dapat dihasilkan. Terakhir, **kadar air (%)** memiliki kontribusi terkecil, sekitar 0,10. Kandungan air yang tinggi memang dapat mengurangi efisiensi proses pembakaran atau konversi energi, namun pengaruhnya relatif kecil dibandingkan dengan berat limbah.



6. Optimasi Model

Untuk meningkatkan performa model lebih lanjut, dilakukan optimasi dengan mencoba kombinasi nilai hyperparameter `n_estimators` dan `max_depth`. Proses ini dikenal dengan hyperparameter tuning.

```

best_r2 = r2
best_model = model

for n in [50, 100, 150]:
    for depth in [5, 10, None]:
        temp_model =
RandomForestRegressor(n_estimators=n, max_depth=depth,
random_state=42)
        temp_model.fit(X_train, y_train)
        temp_r2 = r2_score(y_test,
temp_model.predict(X_test))
        if temp_r2 > best_r2:
            best_r2 = temp_r2
            best_model = temp_model

print(f"Best R2 Score after tuning: {best_r2:.2f}")

```

- a. Parameter yang Diuji
- `n_estimators`: Jumlah pohon dalam hutan (nilai yang diuji: 50, 100, 150)
 - `max_depth`: Kedalaman maksimum tiap pohon keputusan (nilai yang diuji: 5, 10, dan None (tanpa batas kedalaman))
- b. Proses
- Model dilatih berulang kali menggunakan setiap kombinasi nilai `n_estimators` dan `max_depth`.
 - Setiap model diuji menggunakan data uji (`X_test`) dan diukur performanya menggunakan R^2 Score.
 - Jika model saat ini memiliki R^2 lebih tinggi dari sebelumnya, maka model tersebut disimpan sebagai model terbaik (`best_model`), dan nilai R^2 -nya juga diperbarui (`best_r2`).
- c. Hasil
- Setelah semua kombinasi dicoba, diperoleh model terbaik berdasarkan nilai R^2 tertinggi.
 - Nilai R^2 akhir ditampilkan menggunakan `print(f"Best R2 Score after tuning: {best_r2:.2f}")`.

```
# 5. Optimasi Model (Hyperparameter Tuning)

best_r2 = r2
best_model = model

for n in [50, 100, 150]:
    for depth in [5, 10, None]:
        temp_model = RandomForestRegressor(n_estimators=n, max_depth=depth, random_state=42)
        temp_model.fit(X_train, y_train)
        temp_r2 = r2_score(y_test, temp_model.predict(X_test))
        if temp_r2 > best_r2:
            best_r2 = temp_r2
            best_model = temp_model

print(f"Best R2 Score after tuning: {best_r2:.2f}")
```

✓ 9.9s

Best R2 Score after tuning: 0.99

Arti Nilai $R^2 = 0.99$:

Ini berarti model mampu menjelaskan 99% variansi dari data target (`y_test`) dengan sangat baik. Nilai R^2 mendekati 1 menunjukkan model memiliki akurasi prediksi yang sangat tinggi.

KESIMPULAN

Berdasarkan hasil proyek ini, dapat disimpulkan bahwa model machine learning menggunakan **Random Forest Regressor** mampu memprediksi estimasi energi (dalam satuan kWh) yang dapat dihasilkan dari limbah pertanian dengan tingkat akurasi yang sangat tinggi. Proses dimulai dari pra-pemrosesan data dengan teknik encoding dan normalisasi, dilanjutkan dengan eksplorasi data untuk memahami distribusi dan korelasi antar fitur.

Model dilatih dengan data yang telah dibagi secara proporsional dan menghasilkan performa evaluasi yang sangat baik dengan nilai **R^2 sebesar 0.99** pada data uji, serta **rata-rata R^2 sebesar 0.994** dari validasi menggunakan **5-Fold Cross Validation**, yang membuktikan kestabilan model. Visualisasi distribusi residual menunjukkan error prediksi tersebar normal di sekitar nol, dan analisis feature importance mengonfirmasi bahwa **berat limbah** adalah faktor paling dominan dalam prediksi energi, disusul jenis limbah dan kadar air.

Secara keseluruhan, proyek ini menunjukkan bahwa pendekatan machine learning dapat diandalkan dalam membantu memprediksi potensi energi dari limbah pertanian secara akurat, sehingga dapat dimanfaatkan dalam perencanaan energi terbarukan, pengelolaan limbah yang lebih efisien, dan pembangunan yang berkelanjutan di sektor agrikultur.