BAB 4
HASIL PENELITIAN

4.1 RINGKASAN HASIL PENELITIAN

Penelitian ini menghasilkan sebuah sistem *dashboard* prediksi curah hujan berbasis web yang dibangun menggunakan *Streamlit* dan algoritma *XGBoost*. Data cuaca yang digunakan berasal dari BMKG dan telah melalui proses preprocessing seperti pembersihan data, normalisasi, serta penanganan outlier menggunakan metode IQR. Setelah data diproses, sistem melakukan prediksi curah hujan berdasarkan model *XGBoost* dan menampilkan hasilnya dalam bentuk tabel, grafik, serta klasifikasi kategori hujan. Sistem juga menyediakan fitur analisis cuaca ekstrem berdasarkan *threshold* kuartil ketiga (Q3), serta mendukung input manual dan ekspor hasil prediksi ke file *Excel*. Model yang dikembangkan menunjukkan performa cukup baik dengan nilai RMSE sebesar 0.0832, MAPE sebesar 57.81%, dan R^2 sebesar 0.8097. Selain itu, dilakukan juga pengujian dengan model *XGBoost* yang telah dituning, menghasilkan nilai RMSE sebesar 0.0937, MAPE sebesar 24.57%, dan R^2 sebesar 0.7586. Sistem ini diharapkan dapat membantu dalam memprediksi dan menganalisis potensi curah hujan di wilayah Yogyakarta.

4.2 PENGUMPULAN DATA

Pada tahapan ini dilakukan pengumpulan data historis dari Stasiun Klimatologi D.I Yogyakarta melalui *website* Data Online BMKG. Data yang dikumpulkan merupakan data cuaca periode 2020 hingga 2024 (5 tahun), data yang dikumpulkan memiliki variabel suhu, kelembapan, curah hujan, dan kecepatan angin. Jumlah data yang telah dikumpulkan sebanyak 366 untuk tahun 2020, 365 data untuk tahun 2021, 365 data untuk tahun 2022, 365 data untuk tahun 2023, dan 366 data untuk tahun 2024. Maka, total data yang didapatkan 1827 data cuaca, dengan 8 variabel di dalamnya yaitu tanggal, TN (temperatur minimum (°C)), TX (temperatur maksimum (°C)), TAVG (temperatur rata-rata (°C)), RH_AVG (kelembapan rata-rata (%)), RR (curah hujan (mm)), FF_X (kecepatan angin maksimum (m/s)). Dan FF_AVG (kecepatan angin rata-rata (m/s)). Data disimpan

dalam format *excel* (.xlsx), yang merupakan format file untuk menyimpan data dalam bentuk tabel, yang sesuai untuk pengolahan data dalam *Machine Learning (ML).*

Data yang sudah dikumpulkan digunakan untuk melatih model prediksi *Random Forest, XGBoost, LSTM, dan ARIMA.* Tetapi, sebelum tahapan melatih model, perlu dilakukan tahapan *preprocessing* atau pra-pemrosesan data. *Preprocessing* adalah tahapan yang dilakukan untuk membersihkan dan mempersiapkan data sebelum dianalisis atau digunakan untuk melatih model.

Data yang telah dikumpulkan memiliki format tabel seperti yang ditunjukkan pada Tabel 3.1 yang berisikan 8 variabel.

Tabel 4.1 Pengumpulan Data

TANGGAL	TN	TX	TAVG	RH_AVG	RR	FF_X	FF_AVG
01-01-2020	23,4	31,2	25	90	29,5	7	2
02-01-2020	22,2	30,4	25,4	87	30,9	4	1
03-01-2020	-	29,4	25,2	90	13,3	8	2
04-01-2020	23,2	30,4	27,3	83	46,6	6	2
05-01-2020	24,5	27,9	25,2	94	3	4	1
06-01-2020	-	31,7	-	-	-	5	3
07-01-2020	23,7	29,1	26,1	90	29,6	5	2
08-01-2020	24	29,3	25,7	90	0	3	1
09-01-2020	23,8	30,2	26,6	85	2,6	3	1
10-01-2020	23	30,3	25,6	90	19,1	10	2

Pada Tabel 4.1, menunjukkan potongan dari data cuaca harian yang telah diperoleh dari *website* Data Online BMKG dengan format file *excel* (.xlsx). Data tersebut mencakup berbagai parameter meteorologis penting yang digunakan sebagai input dalam proses pemodelan prediksi

Tabel 4.2 Keterangan Dalam Data

NAMA	KETERANGAN
8888	Data tidak terukut
9999	Tidak ada data (tidak dilakukan pengukuran)
TN	Temperatur minimum (°C)
TX	Temperatur maksimum (°C)
TAVG	Temperatur rata-rata (°C)
RH_AVG	Kelembapan rata-rata (%)
RR	Curah hujan (mm)
FF_X	Kecepatan angin maksimum (m/s)
FF_AVG	Kecepatan angin rata-rata (m/s)

Pada Tabel 4.2 menunjukkan keterangan yang ada di dalam file data tersebut. Keterangan ini menjadi acuan penting dalam proses preprocessing data, terutama untuk memahami arti nilai-nilai tertentu seperti 8888 atau 9999 yang menandakan data tidak tersedia atau tidak terukur, sehingga perlu ditangani secara khusus agar tidak memengaruhi hasil prediksi curah hujan yang dihasilkan oleh model.

4.3 PREPROCESSING

Tahapan *preprocessing* merupakan salah satu proses penting dalam pemodelan data yang bertujuan untuk memastikan bahwa data yang digunakan dalam pelatihan model berada dalam kondisi yang bersih, konsisten, dan layak digunakan. Data mentah yang telah diperoleh dari Stasiun Klimatologi D.I Yogyakarta masih mengandung berbagai permasalahan, seperti nilai kosong *(missing values),* data pencilan *(outlier),* serta variasi skala antar variabel. Maka dari itu, tahapan ini mencakup beberapa proses utama, yaitu data cleaning, hadnling missing values, outlier detection and clipping, normalization, dan time series transformation untuk kebutuhan pemodelan prediktif.

Untuk memulai proses preprocessing, dilakuakn terlebih dahulu proses impor library yang dibutuhkan dan pembacaan dataset cuaca yang telah

dikumpulkan. Dataset tersebut disimpan dalam format *excel* (.xlsx) dan diakses melalui *Google Drive* menggunakan *Google Colab.* Code yang digunakan seperti berikut:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error,
mean_absolute_percentage_error, r2_score
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense
import math

from google.colab import drive
drive.mount('/content/drive')

df = pd.read_excel("/content/drive/MyDrive/Colab
Notebooks/SKRIPSI/Data Cuaca BMKG.xlsx")
```


Setelah semua library diimpor dan dataset berhasil dimuat, langkah selanjutnya adalah melakukan eksplorasi awal dan pembersihan data, yang akan dijelaskan pada subbab berikutnya.

4.3.1 *Data Cleaning*

Tahapan awal dalam proses preprocessing adalah memahami struktur dan kondisi awal dari dataset. Data cuaca harian yang digunakan dalam penelitian ini diperoleh dalam format *excel* (.xlsx). Eksplorasi awal dilakukan untuk melihat jumlah data, tipe variabel, serta nilai statistik deskriptif dari masing-masing fitur.

Setelah itu, menyalin data asli ke dalam variabel baru (df_cleaned) agar data mentah tetap terjaga. Kemudian dilakukan penggantian terhadap nilai-nilai yang tidak valid seperti '-', 8888, dan 9999 yang secara umum menandakan data tidak tercatat atau error. Nilai-nilai tersebut digantikan dengan NaN agar dapat dikenali sebagai data hilang oleh *Python.* Selanjutnya seluruh kolom (kecuali kolom tanggal) dikonversi ke format numerik (float64) agar dapat digunakan dalam

perhitungan statistik dan pemodelan. Berikut adalah cuplikan kode program yang digunakan untuk pembersihan data:

```
df_cleaned = df.copy()
df_cleaned.replace(['-', '8888', '9999'], np.nan, inplace=True)
cols_to_numeric = ['TN', 'TX', 'TAVG', 'RH_AVG', 'RR', 'FF_X',
'FF_AVG']
df_cleaned[cols_to_numeric] =
df_cleaned[cols_to_numeric].astype(float)

df_cleaned['TANGGAL'] = pd.to_datetime(df_cleaned['TANGGAL'],
dayfirst=True)

print(df_cleaned.head())
```

Hasil dari cuplikan code tersebut ditunjukkan pada Gambar 3.2 yang menampilkan lima baris pertama data hasil pembersihan menggunakan perintah df_cleaned.head().

	TANGGAL	TN	TX	TAVG	RH_AVG	RR	FF_X	FF_AVG
0	2020-01-01	23.4	31.2	25.0	90.0	29.5	7.0	2.0
1	2020-01-02	22.2	30.4	25.4	87.0	30.9	4.0	1.0
2	2020-01-03	NaN	29.4	25.2	90.0	13.3	8.0	2.0
3	2020-01-04	23.2	30.4	27.3	83.0	46.6	6.0	2.0
4	2020-01-05	24.5	27.9	25.2	94.0	3.0	4.0	1.0

Gambar 4.1 Hasil Data Setelah Pembersihan

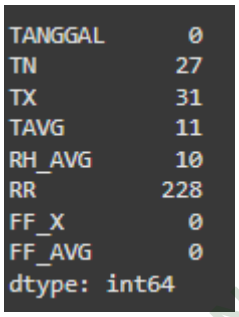
Hasil yang ditunjukkan pada Gambar 4.1 menunjukkan bahwa sebagian besar data telah berhasil dikonversi ke dalam bentuk numerik dan nilai-nilai yang tidak valid telah digantikan dengan NaN. Sebagai contoh, pada tanggal 3 Januari 2020, nilai pada kolom TN (temperatur minimum) tidak tersedia sehingga ditampilkan sebagai NaN. Hal ini menunjukkan bahwa proses pembersihan data telah berhasil mendeteksi dan menandai data yang hilang sesuai harapan.

4.3.2 *Handling Missing Values*

Setelah melakukan pembersihan data awal, langkah selanjutnya adalah menangani nilai kosong *(missing values)* yang masih terdapat dalam dataset. Pengecekan jumlah missing values dilakukan untuk mengetahui sebaran nilai yang

hilang pada tiap kolom. Berikut adalah code yang digunakan untuk menghitung jumlah nilai kosong:

```
missing_values = df_cleaned.isnull().sum()
print(df_cleaned.isnull().sum())
```



Gambar 4.2 Hasil Pengecekkan Missing Values

Pada Gambar 4.2 diketahui bahwa terdapat *missing values* pada beberapa kolom numerik, terutama pada kolom RR yang memiliki jumlah data kosong paling banyak. Untuk menangani hal tersebut, digunakan metode pengisian nilai tengah *(median imputation)* karena media lebih *robust* teradap *outlier* dibandingkan *mean.* Proses tersebut dilakukan hanya pada kolom numerik. Berikut code untuk mengisi missing values sebagai berikut:

```
df_cleaned['TN'].fillna(df_cleaned['TN'].median(), inplace=True)
df_cleaned['TX'].fillna(df_cleaned['TX'].median(), inplace=True)
df_cleaned['TAVG'].fillna(df_cleaned['TAVG'].median(),
inplace=True)
df_cleaned['RH_AVG'].fillna(df_cleaned['RH_AVG'].median(),
inplace=True)
df_cleaned['RR'].fillna(df_cleaned['RR'].median(), inplace=True)
df_cleaned['FF_X'].fillna(df_cleaned['FF_X'].median(),
inplace=True)
df_cleaned['FF_AVG'].fillna(df_cleaned['FF_AVG'].median(),
inplace=True)

print("\nMissing values setelah imputasi:\n",
df_cleaned.isnull().sum())
print(df_filled.head())
```

```
Missing values setelah imputasi:
 TANGGAL    0
TN          0
TX          0
TAVG        0
RH_AVG      0
RR          0
FF_X        0
FF_AVG      0
dtype: int64
```

Gambar 4.3 Hasil Setelah Pengisian *Missing Values*

	TANGGAL	TN	TX	TAVG	RH_AVG	RR	FF_X	FF_AVG
0	2020-01-01	23.4	31.2	25.0	90.0	29.5	7.0	2.0
1	2020-01-02	22.2	30.4	25.4	87.0	30.9	4.0	1.0
2	2020-01-03	23.2	29.4	25.2	90.0	13.3	8.0	2.0
3	2020-01-04	23.2	30.4	27.3	83.0	46.6	6.0	2.0
4	2020-01-05	24.5	27.9	25.2	94.0	3.0	4.0	1.0

Gambar 4.4 Cuplikan Data Setelah Penanganan *Missing Values*

Berdasarkan Gambar 4.3 dan Gambar 4.4 menunjukkan bahwa *missing values* berhasil ditangani. Hasil menunjukkan bahwa tidak terdapat nilai kosong (NaN) ladi dalam dataset. Setiap kolom numerik telah memiliki nilai hasil imputasi yang merepresentasikan kondisi median dari kolom tersebut.

4.3.3 *Outlier Detection and Clipping*

Setelah menangani nilai kosong, tahapan selanjutnya dalam *preprocessing* adalah mendeteksi dan menangani *outlier. Outlier* merupakan data pencilan yang secara signifikan berbeda dari mayoritas nilai lainnya. Keberadaan *outlier* dapat mempengaruhi hasil pemodelan, sehingga perlu ditangani secara tepat.

a. *Outlier Detection*

Metode yang digunakan dalam penelitian ini untuk mendeteksi outlier adalah metode Interquartile Range (IQR). Pendekatan ini mengidentifikasi data yang berada jauh dari sebaran normal berdasarkan rentang antar kuartil.

- Kuartil pertama (Q1) adalah nilai yang membagi 25% data terbawah dari distribusi, atau disebut juga persentil ke-25.
- Kuartil ketiga (Q3) adalah nilai yang membagi 75% data terbawah, atau disebut juga persentil ke-75.

- IQR *(Interquartile Range)* merupakan selisih antara Q3 dan Q1, yaitu:

  IQR = Q3 - Q1

  Batas bawah dan atas dari data yang dianggap wajar didefinisikan sebagai:

  - Lower bound = Q1 – 1.5 × IQR
  - Upper bound = Q3 + 1.5 × IQR

Data di luar batas tersebut dianggap sebagai *outlier*. Berikut adalah code yang digunakan untuk melakukan deteksi *outlier:*

```
num_cols = ['TN', 'TX', 'TAVG', 'RH_AVG', 'RR', 'FF_X', 'FF_AVG']

outlier_info = {}

for col in num_cols:
    Q1 = df_cleaned[col].quantile(0.25)
    Q3 = df_cleaned[col].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR

  outliers = df_cleaned[(df_cleaned[col] < lower_bound) |
(df_cleaned[col] > upper_bound)]
    outlier_info[col] = len(outliers)

print("Jumlah outlier per kolom (sebelum clipping):")
for col, count in outlier_info.items():
    print(f"{col}: {count} outlier")
```

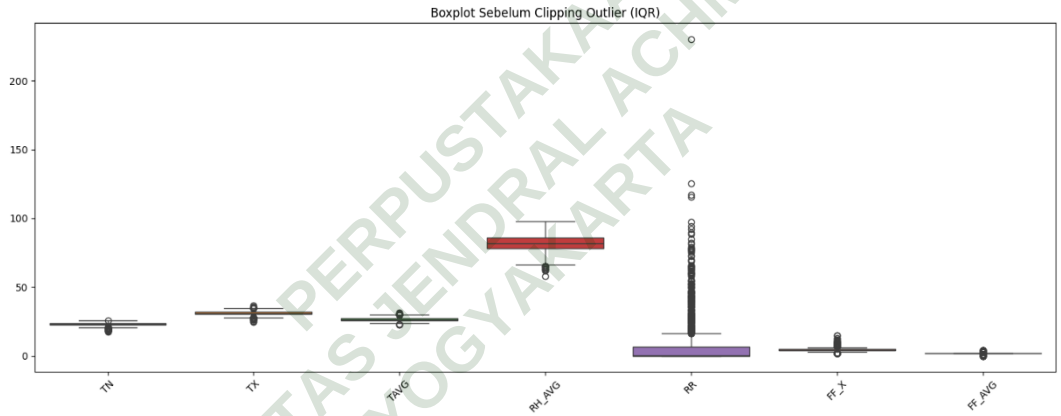
```
Jumlah outlier per kolom (sebelum clipping):
TN: 85 outlier
TX: 44 outlier
TAVG: 31 outlier
RH_AVG: 15 outlier
RR: 253 outlier
FF_X: 185 outlier
FF_AVG: 766 outlier
```

Gambar 4.5 Hasil Deteksi Jumlah *Outlier*

Pada Gambar 4.5 diketahui bahwa sebagian besar variabel dalam dataset mengandung *outlier*. Kolom FF_AVG memiliki jumlah *outlier* terbanyak, yaitu sebanyak 766 data. Hal ini mengindikasikan bahwa terdapat banyak variasi ekstrem

dalam catatan kecepatan angin rata-rata. Sementara itu, kolom RR juga menunjukkan jumlah *outlier* yang tinggi (253 data), yang mencerminkan adanya kejadian hujan ekstrem pada data historis yang dicatat. *Outlier* pada curah hujan merupakan hal yang umum ditemukan dalam data meteorologi, khususnya di wilayah tropis seperti Yogyakarta. Kolom TN, TX, serta TAVG juga terdeteksi mengandung puluhan outlier, namun masih dalam jumlah yang relatif wajar.

Untuk mengevaluasi distribusi dan mendeteksi keberadaan nilai-nilai pencilan *(outlier)* dalam dataset, dilakukan visualisasi menggunakan *boxplot* terhadap seluruh fitur numerik. Visualisasi ini membantu dalam mengidentifikasi fitur mana yang mengandung penyimpangan ekstrem yang berpotensi memengaruhi performa model prediktif.



Gambar 4.6 Hasil Visualisasi *Boxplot Outlier*

Pada Gambar 4.6 diketahui bahwa sebagian besar variabel dalam dataset mengandung *outlier.* Kolom FF_AVG memiliki jumlah *outlier* terbanyak, yaitu sebanyak 766 data.

Dari hasil ini, dapat disimpulkan bahwa deteksi *outlier* sangat penting dilakukan untuk mencegah bias dalam pelatihan model. Oleh karena itu, pada tahap selanjutnya dilakukan proses *clipping* terhadap nilai-nilai *outlier* agar tetap berada dalam batas yang wajar tanpa menghapus data.

b. *Clipping*

Setelah dilakukan deteksi, langkah selanjutnya adalah penanganan *outlier* dengan menggunakan metode *clipping* berbasis IQR. Teknik ini dilakukan dengan membatasi nilai-nilai yang melebihi batas bawah dan atas IQR agar berada pada

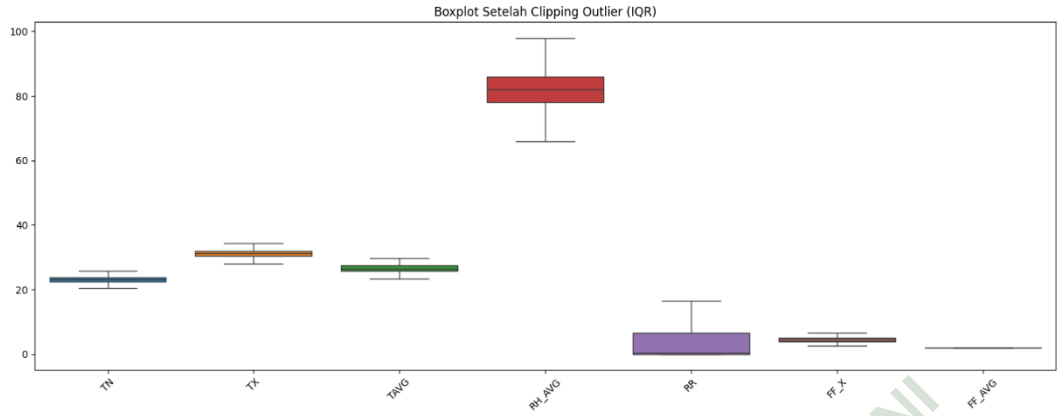
nilai minimum dan maksimum yang ditentukan oleh rentang IQR. Metode ini dinilai lebih efektif dibanding menghapus data karena tetap mempertahankan struktur data asli namun mereduksi pengaruh *outlier*. Berikut adalah code yang digunakan untuk *clipping:*

```
df_outlier_clipped = df_cleaned.copy()

for col in num_cols:
    Q1 = df_outlier_clipped[col].quantile(0.25)
    Q3 = df_outlier_clipped[col].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR

    # Clipping nilai
    df_outlier_clipped[col] = np.clip(df_outlier_clipped[col],
lower_bound, upper_bound)

print("\nRingkasan data setelah clipping:")
print(df_outlier_clipped.describe())
```

```
Ringkasan data setelah clipping:
       TANGGAL    TN    TX  TAVG  RH_AVG      RR  FF_X  FF_AVG
0   2020-01-01  23.4  31.2  25.0    90.0  16.375   6.5     2.0
1   2020-01-02  22.2  30.4  25.4    87.0  16.375   4.0     2.0
2   2020-01-03  23.2  29.4  25.2    90.0  13.300   6.5     2.0
3   2020-01-04  23.2  30.4  27.3    83.0  16.375   6.0     2.0
4   2020-01-05  24.5  28.0  25.2    94.0   3.000   4.0     2.0
...        ...   ...   ...   ...     ...     ...   ...     ...
1822 2024-12-27  23.0  32.4  27.6    78.0   8.200   5.0     2.0
1823 2024-12-28  25.0  30.6  26.2    86.0   0.000   5.0     2.0
1824 2024-12-29  23.4  28.4  25.1    89.0   9.300   3.0     2.0
1825 2024-12-30  24.0  30.8  26.3    86.0   4.600   5.0     2.0
1826 2024-12-31  23.7  30.8  26.9    80.0   2.300   3.0     2.0
```

Gambar 4.7 Hasil *Outlier* Setelah *Clipping*

Pada Gambar 4.7, seluruh *outlier* pada semua kolom berhasil dikendalikan melalui metode *clipping.* Tidak ada lagi nilai yang berada di luar batas IQR, yang menunjukkan bahwa distribusi data sudah dalam kisaran yang wajar dan layak digunakan untuk proses pemodelan lebih lanjut. Untuk memverifikasi keberhasilan proses ini, dilakukan visualisasi kembali menggunakan *boxplot* terhadap kolom-kolom numerik setelah proses *clipping*.



Gambar 4.8 Hasil Visualisasi *Boxplot Clipping*

Pada Gambar 4.8, seluruh *outlier* pada semua kolom berhasil dikendalikan melalui metode *clipping*. Tidak ada lagi nilai yang berada di luar batas IQR, yang menunjukkan bahwa distribusi data sudah dalam kisaran yang wajar dan layak digunakan untuk proses pemodelan lebih lanjut. Metode *clipping* ini dipilih karena mampu mengurangi pengaruh ekstrem tanpa mengurangi jumlah data, sehingga tetap menjaga integritas dan kontinuitas data *time series.*

4.3.4 *Normalization*

Setelah data bebas dari nilai kosong *(missing values)* dan *outlier*, tahapan selanjutnya adalah normalisasi data. Normalisasi penting dilakukan terutama ketika fitur memiliki skala yang berbeda, agar algoritma *Machine Learning* seperti LSTM, *XGBoost*, dan model berbasis regresi dapat bekerja secara optimal tanpa bias terhadap fitur yang memiliki rentang nilai lebih besar.

Dalam penelitian ini, metode normalisasi yang digunakan adalah *Min-Max Scaling*, yaitu menskalakan nilai setiap fitur ke dalam rentang [0, 1] dengan rumus:

$$Xscaled = \frac{X - Xmin}{Xmax - Xmin}$$

Normalisasi dilakukan pada semua kolom numerik, kecuali kolom ‘TANGGAL’ yang berfungsi sebagai indeks *time series*. Berikut potongan code yang digunakan ditunjukkan sebagai berikut:

```
df_normalized = df_outlier_clipped.copy()

scaler = MinMaxScaler()

df_normalized[num_cols] =
scaler.fit_transform(df_normalized[num_cols])

print("Hasil normalisasi (0-1):")
print(df_normalized[num_cols].describe())
print(df_normalized.head())
```

```
Hasil normalisasi (0-1):
                TN           TX         TAVG       RH_AVG           RR  \
count  1827.000000  1827.000000  1827.000000  1827.000000  1827.000000
mean      0.486064     0.488779     0.507680     0.483580     0.246323
std       0.207394     0.195403     0.196580     0.183153     0.370138
min       0.000000     0.000000     0.000000     0.000000     0.000000
25%       0.375000     0.375000     0.375000     0.375000     0.000000
50%       0.509615     0.500000     0.484375     0.500000     0.024427
75%       0.625000     0.625000     0.625000     0.625000     0.400000
max       1.000000     1.000000     1.000000     1.000000     1.000000

              FF_X  FF_AVG
count  1827.000000  1827.0
mean      0.492200     0.0
std       0.285339     0.0
min       0.000000     0.0
25%       0.375000     0.0
50%       0.375000     0.0
75%       0.625000     0.0
max       1.000000     0.0
```

Gambar 4.9 Hasil Ringkasan Statistik Normalisasi

Pada Gambar 4.9, dapat disimpulkan bahwa seluruh nilai sudah berada dalam rentang normalisasi [0, 1]. Namun, juga terlihat bahwa kolom RR memiliki sebaran nilai yang cukup rendah pada kuartil bawah (Q1 dan Q2), menunjukkan bahwa sebagian besar hari memiliki curah hujan rendah, sementara hanya sebagian kecil yang menunjukkan hujan lebat atau ekstrem (nilai mendekati 1).

TANGGAL	TN	TX	TAVG	RH_AVG	RR	FF_X	FF_AVG
2020-01-01	0.548077	0.50000	0.250000	0.75000	1.000000	1.000	0.0
2020-01-02	0.317308	0.37500	0.312500	0.65625	1.000000	0.375	0.0
2020-01-03	0.509615	0.21875	0.281250	0.75000	0.812214	1.000	0.0
2020-01-04	0.509615	0.37500	0.609375	0.53125	1.000000	0.875	0.0
2020-01-05	0.759615	0.00000	0.281250	0.87500	0.183206	0.375	0.0

Gambar 4.10 Hasil Normalisasi

Pada Gambar 4.10, terlihat bahwa seluruh nilai fitur telah berhasil dinormalisasi ke skala 0 hingga 1. Sebagai contoh, pada tanggal 1 Januari 2020, nilai RR (curah hujan) mencapai 1.0, yang menunjukkan bahwa hari tersebut memiliki nilai curah hujan tertinggi dalam dataset.

4.3.5 *Time Series Transformation*

Setelah data dinormalisasi, tahap selanjutnya adalah mentransformasikan data ke dalam format *time series*. Hal ini penting dikarenakan karakteristik data cuaca bersifat temporal atau runtun waktu, sehingga perlu diatur agar model dapat membaca urutan waktu secara benar.

Transformasi dilakukan dengan menjadikan kolom 'TANGGAL' sebagai index datetime, kemudian mengurutkan data berdasarkan tanggal secara kronologis. Berikut cuplikan code yang digunakan:

```
df_normalized.set_index('TANGGAL', inplace=True)
df_normalized.sort_index(inplace=True)

print(df_normalized.head())
```

TANGGAL	TN	TX	TAVG	RH_AVG	RR	FF_X	FF_AVG
2020-01-01	0.548077	0.50000	0.250000	0.75000	1.000000	1.000	0.0
2020-01-02	0.317308	0.37500	0.312500	0.65625	1.000000	0.375	0.0
2020-01-03	0.509615	0.21875	0.281250	0.75000	0.611494	1.000	0.0
2020-01-04	0.509615	0.37500	0.609375	0.53125	1.000000	0.875	0.0
2020-01-05	0.759615	0.00000	0.281250	0.87500	0.137931	0.375	0.0

Gambar 4.11 Hasil Data Setelah *Transformasi Time Series*

Pada Gambar 4.11, menunjukkan bahwa dengan menjadikan kolom 'TANGGAL' sebagai index, data kini sudah siap digunakan dalam pemodelan *time series*. Format ini sangat krusial terutama untuk model seperti LSTM dan ARIMA yang membutuhkan urutan waktu untuk melakukan prediksi berdasarkan data historis.

Setelah data berhasil ditransformasikan ke dalam format *time series* dengan menjadikan kolom 'TANGGAL' sebagai indeks bertipe *datetime*, langkah selanjutnya adalah melakukan *resampling* data berdasarkan periode waktu tertentu. *Resampling* ini bertujuan untuk memperoleh rata-rata nilai cuaca per bulan maupun

per tahun, sehingga pola musiman dan tren tahunan dapat dianalisis secara lebih jelas. Dalam penelitian ini dilakukan dua jenis *resampling,* yaitu:

- *Resampling* bulanan ('M') untuk memperoleh rata-rata data setiap bulan.
- *Resampling* tahunan ('Y') untuk memperoleh rata-rata data setiap tahun.

Berikut adalah cuplikan code yang digunakan untuk melakukan proses *resampling* tersebut:

```
df_monthly = df_normalized.resample('M').mean()

df_yearly = df_normalized.resample('Y').mean()

print(df_monthly.head())
print(df_yearly.head())
```

TANGGAL	TN	TX	TAVG	RH_AVG	RR	FF_X	FF_AVG
2020-01-31	0.566687	0.530242	0.533266	0.561492	0.361537	0.487903	0.0
2020-02-29	0.571286	0.556034	0.492996	0.604526	0.386786	0.534483	0.0
2020-03-31	0.574752	0.546371	0.494960	0.617944	0.557203	0.427419	0.0
2020-04-30	0.584615	0.602604	0.545312	0.582292	0.328448	0.345833	0.0
2020-05-31	0.581576	0.529234	0.550403	0.587702	0.275794	0.298387	0.0

Gambar 4.12 Hasil *Resampling* Bulanan

Pada Gambar 4.12, menampilkan hasil *resampling* bulanan, dapat disimpulkan bahwa musim hujan di wilayah Yogyakarta umumnya terjadi pada Januari hingga Maret, ditandai dengan tingginya kelembaban dan curah hujan. Sementara itu, bulan Mei ke atas menunjukkan tren penurunan curah hujan, mengindikasikan transisi menuju musim kemarau. Pemahaman pola musiman ini sangat berguna untuk membangun model prediksi cuaca berbasis waktu seperti LSTM.

TANGGAL	TN	TX	TAVG	RH_AVG	RR	FF_X	FF_AVG
2020-12-31	0.489702	0.506148	0.503116	0.543460	0.262875	0.479850	0.0
2021-12-31	0.473683	0.462329	0.478168	0.498288	0.267054	0.507534	0.0
2022-12-31	0.471549	0.413142	0.421361	0.560017	0.305979	0.435959	0.0
2023-12-31	0.458087	0.498416	0.496789	0.414127	0.158507	0.556507	0.0
2024-12-31	0.537148	0.563610	0.638619	0.402066	0.237179	0.481216	0.0

Gambar 4.13 Hasil *Resampling* Tahunan

Pada Gambar 4.13, menunjukkan hasil *resampling* tahunan, dapat disimpulkan bahwa terdapat fluktuasi signifikan pada nilai-nilai parameter cuaca dari tahun ke tahun. Tren peningkatan suhu dan penurunan kelembaban pada tahun-tahun terakhir menunjukkan kemungkinan adanya dampak perubahan iklim lokal yang perlu dikaji lebih lanjut. Dengan menggunakan data ini, model prediksi dapat dilatih untuk mengantisipasi kondisi ekstrem berdasarkan tren tahunan historis.

4.4 FORECASTING

Tahapan *forecasting* merupakan inti dari penelitian ini, yaitu memprediksi curah hujan (RR) menggunakan algoritma *Machine Learning (ML)* dan *Deep Learning* berdasarkan data historis. Proses *forecasting* dilakukan dengan beberapa tahap, yaitu pemisah data (*split* data), penentuan fitur dan target, serta penerapan berbagai model prediktif dan evaluasi.

4.4.1 *Split* Data

Data historis yang telah ditransformasikan ke dalam format *time series* dan dirata-ratakan per bulan kemudian dibagi menjadi dua bagian:

1) Data Latih *(Training):* Januari 2020 – Desember 2023.
2) Data Uji *(Testing):* Januari 2024 - Desember 2024.

Tujuan dari pembagian ini dilakukan supaya model mempelajari pola cuaca musiman bulanan dari 4 tahun sebelumnya dan memprediksi kondisi cuaca pada tahun 2024 secara akurat dan realistis. Berikut cuplikan code yang digunakan:

```
df_monthly.index = pd.to_datetime(df_monthly.index)

train = df_monthly[df_monthly.index < '2024-01-01']
test = df_monthly[df_monthly.index >= '2024-01-01']

train_start = train.index.min().strftime('%B %Y')
train_end = train.index.max().strftime('%B %Y')
test_start = test.index.min().strftime('%B %Y')
test_end = test.index.max().strftime('%B %Y')
```

```
Jumlah data train: 48 bulan (January 2020 - December 2023)
Jumlah data test : 12 bulan (January 2024 - December 2024)
```

Gambar 4.14 Hasil Split Data

Pada Gambar 4.14, menampilkan proses *split* data berdasarkan waktu, yaitu data latih dan data uji dalam skala bulanan. Jumlah data latih sebanyak 48 bulan yang mencakup periode Januari 2020 hingga Desember 2023, sedangkan data uji terdiri dari 12 bulan pada rentang waktu Januari 2024 hingga Desember 2024.

Pendekatan *time-based splitting* seperti ini sangat penting dalam pemodelan deret waktu *(time series),* karena memastikan bahwa model hanya belajar dari data masa lalu dan dievaluasi menggunakan data masa depan. Hal ini mencegah terjadinya data *leakage* dan membuat evaluasi model menjadi lebih adil serta realistis dalam menggambarkan performa prediksi terhadap kondisi aktual di lapangan.

4.4.2 Penentuan Fitur dan Target

Dalam penelitian ini, variabel yang menjadi target prediksi adalah RR (Curah Hujan), sedangkan variabel-variabel yang digunakan sebagai fitur input meliputi:

- TN (Temperatur Minimum).
- TX (Temperatur Maksimum).
- TAVG (Temperatur Rata-rata).
- RH_AVG (Kelembapan rata-rata).
- FF_X (Kecepatan Angin Maksimum).

Pemilihan kelima fitur tersebut didasarkan pada variabel-variabel meteorologi yang secara ilmiah berkontribusi terhadap terjadinya curah hujan, dan umumnya digunakan dalam studi klimatologi maupun prediksi cuaca. Berikut cuplikan code yang digunakan:

```
fitur = ['TN', 'TX', 'TAVG', 'RH_AVG', 'FF_X']
target = 'RR'

X_train = train[fitur]
y_train = train[target]

X_test = test[fitur]
y_test = test[target]

print(f"X_train: {X_train.shape}, y_train: {y_train.shape}")
```

```
print(f"X_test: {X_test.shape}, y_test: {y_test.shape}")
```

```
X_train: (48, 5), y_train: (48,)
X_test: (12, 5), y_test: (12,)
```

Gambar 4.15 Hasil Pembagian Fitur dan Target

Berdasarkan hasil yang ditunjukkan pada Gambar 4.15, diperoleh:

- Data pelatihan (X_train dan y_train) berjumlah 48 baris, yang mewakili data bulanan dari Januari 2020 hingga Desember 2023.
- Data pengujian (X_test dan y_test) terdiri dari 12 baris, yang merepresentasikan 12 bulan di tahun 2024.

Setiap baris data memiliki 5 fitur input utama, yaitu TN, TX, TAVG, RH_AVG, dan FF_X, yang akan digunakan untuk memprediksi nilai RR sebagai target curah hujan. Hasil ini menunjukkan bahwa struktur data telah siap digunakan untuk pelatihan dan pengujian berbagai model prediksi dalam penelitian ini.

4.5 PEMODELAN DAN EVALUASI

Setelah data dibersihkan, dinormalisasi, dan disusun dalam format *time series* bulanan, tahap selanjutnya adalah membangun model prediksi. Dalam penelitian ini digunakan algoritma *Random Forest, XGBoost, Long Short-Term Memory (LSTM), dan AutoRegressive Integrated Moving Average (ARIMA)*. Setiap algoritma dipilih berdasarkan keunggulannya masing-masing dalam menangani data numerik dan runtun waktu *(time series). XGBoost* dan *Random Forest* merupakan metode *ensemble* berbasis pohon keputusan yang efektif dalam menangani non-linearitas. LSTM dipilih karena kemampuannya dalam mengenali pola jangka panjang pada data sekuensial. Sementara itu, ARIMA merupakan pendekatan statistik klasik yang populer dan kuat untuk memodelkan *data time series univariat*.

4.5.1 *Random Forest*

a. Default

Langkah awal dalam eksperimen ini adalah membangun model *Random Forest* dengan menggunakan parameter *default* dari pustaka scikit-learn. Tujuan dari model *baseline* ini adalah untuk mendapatkan gambaran awal performa tanpa adanya proses tuning atau optimasi. Berikut cuplikan code yang digunakan:

```
from sklearn.ensemble import RandomForestRegressor
import time

# Inisialisasi model
rf_model = RandomForestRegressor(random_state=42)

# Training
start_train = time.time()
rf_model.fit(X_train, y_train)
end_train = time.time()
train_duration = end_train - start_train

# Prediksi
start_pred = time.time()
y_pred_rf = rf_model.predict(X_test)
end_pred = time.time()
pred_duration = end_pred - start_pred

# Evaluasi
rmse_rf = np.sqrt(mean_squared_error(y_test, y_pred_rf))
mask = y_test != 0
mape_rf = mean_absolute_percentage_error(y_test[mask],
y_pred_rf[mask]) * 100
r2_rf = r2_score(y_test, y_pred_rf)
```



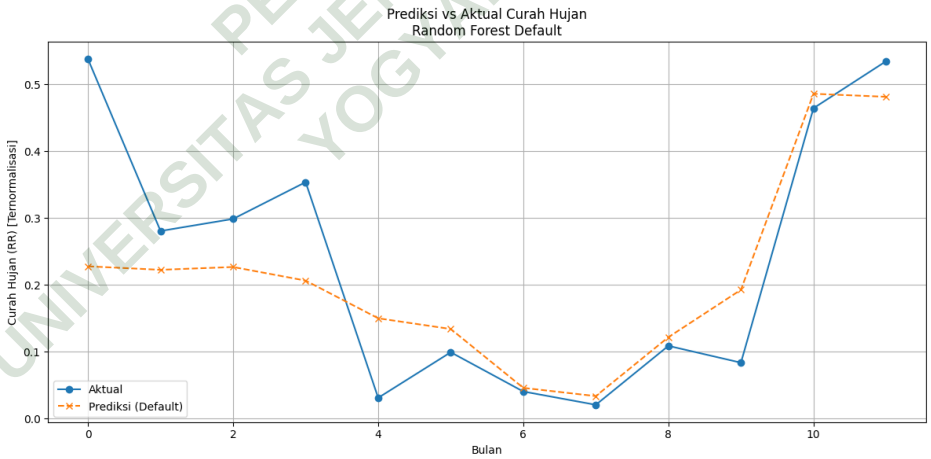
```
=== Random Forest (Default) ===
RMSE          : 0.1144
MAPE          : 66.49%
R² Score      : 0.6400
Waktu Training: 0.0834 detik
Waktu Prediksi: 0.0044 detik
```

Gambar 4.16 Hasil Evaluasi *Random Forest Default*

Pada Gambar 4.16, model *Random Forest default* menunjukkan performa yang cukup baik dalam memprediksi curah hujan bulanan:

- RMSE sebesar 0.1053 mengindikasikan bahwa rata-rata kesalahan prediksi berada pada kisaran 10.5% dari skala normalisasi.
- MAPE sebesar 61.05% masih tergolong tinggi, mengindikasikan adanya deviasi yang cukup besar terhadap nilai aktual, terutama saat nilai aktual rendah (MAPE sensitif terhadap nol).
- R^2 score sebesar 0.6949 menunjukkan bahwa sekitar 69% variasi nilai curah hujan pada data uji dapat dijelaskan oleh model.
- Waktu komputasi sangat efisien, dengan waktu pelatihan hanya 0.217 detik dan prediksi 0.016 detik.

Model ini akan digunakan sebagai *baseline* sebelum dilakukan tuning *hyperparameter* untuk meningkatkan akurasi prediksi. Untuk memberikan gambaran yang lebih intuitif terhadap performa model, dilakukan visualisasi antara nilai aktual dan nilai prediksi curah hujan (RR) pada data uji selama 12 bulan tahun 2024.



Gambar 4.17 Hasil Visualisasi *Random Forest Default*

Berdasarkan grafik yang ditunjukkan pada Gambar 4.17, menampilkan hasil perbandingan antara nilai aktual dan nilai prediksi curah hujan (RR) hasil dari model *Random Forest* tanpa tuning pada data uji (tahun 2024) dalam skala bulanan. Terlihat bahwa model mampu mengikuti tren umum pola curah hujan, terutama pada periode akhir tahun saat curah hujan tinggi (bulan ke-10 dan ke-11). Namun,

pada beberapa bulan awal hingga pertengahan tahun, prediksi model cenderung terlalu rendah atau terlalu datar dibandingkan nilai aktual. Hal ini mengindikasikan bahwa model masih belum cukup sensitif terhadap fluktuasi yang lebih tajam, khususnya pada musim transisi. Ketidaksesuaian prediksi di beberapa bulan ini turut berkontribusi terhadap nilai MAPE yang cukup tinggi sebesar 61,05%. Meski demikian, secara keseluruhan model menunjukkan potensi awal yang baik, dengan nilai R^2 mencapai 0,6949 yang mengindikasikan bahwa sekitar 69% variasi data dapat dijelaskan oleh model. Visualisasi ini memperkuat pentingnya analisis lanjutan, seperti tuning *hyperparameter*, untuk meningkatkan akurasi prediksi secara keseluruhan.

b. Tuning

Setelah dilakukan pemodelan awal menggunakan konfigurasi default, tahap selanjutnya adalah melakukan tuning *hyperparameter* untuk meningkatkan performa model *Random Forest*. Proses tuning dilakukan menggunakan *Grid Search* dengan teknik validasi silang *(cross-validation)* sebanyak 3 fold. Parameter yang diuji meliputi 'n_estimators' (jumlah pohon), max_depth (kedalaman maksimum pohon), dan 'min_samples_split' (jumlah minimum sampel untuk memecah node). Berikut cuplikan code yang digunakan:

```
# Param grid
param_grid_rf = {
    'n_estimators': [100, 200],
    'max_depth': [3, 5, None],
    'min_samples_split': [2, 5]
}
# Model & Grid Search
rf = RandomForestRegressor(random_state=42)
grid_rf = GridSearchCV(rf, param_grid_rf, cv=3, scoring='r2', n_jobs=-
1)
# Waktu training
start_train = time.time()
grid_rf.fit(X_train, y_train)
end_train = time.time()
# Waktu prediksi
start_pred = time.time()
y_pred_rf = grid_rf.predict(X_test)
end_pred = time.time()
```

```
# Evaluasi
mse = mean_squared_error(y_test, y_pred_rf)
rmse = np.sqrt(mse)
mape = np.mean(np.abs((y_test - y_pred_rf) / y_test)) * 100
r2 = r2_score(y_test, y_pred_rf)
```

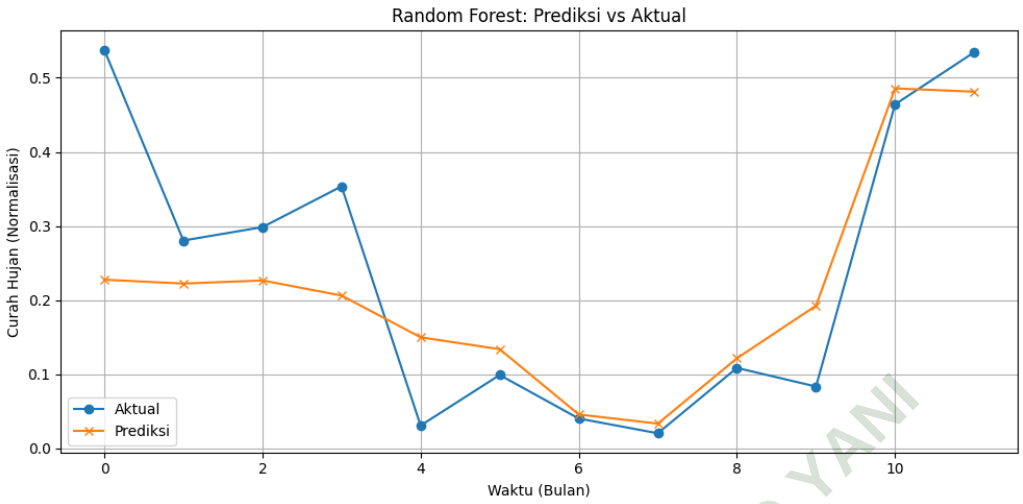
```
=== Evaluasi Random Forest (Tuning) ===
Best Params  : {'max_depth': 5, 'min_samples_split': 5, 'n_estimators': 200}
RMSE         : 0.1144
MAPE         : 66.49%
R² Score     : 0.6400
Waktu training: 10.8085 detik
Waktu prediksi: 0.0254 detik
```

Gambar 4.18 Hasil Evaluasi *Random Forest* Tuning

Berdasarkan hasil tuning seperti yang ditampilkan pada Gambar 4.18, kombinasi parameter terbaik yang ditemukan adalah n_estimators = 200, max_depth = 5, dan min_samples_split = 5. Dengan konfigurasi ini, model menghasilkan nilai RMSE sebesar 0.1144, MAPE sebesar 66.49%, dan R^2 sebesar 0.6400.

Meski nilai R^2 sedikit menurun dibandingkan model default (dari 0.6949 menjadi 0.6400), proses tuning ini menunjukkan bahwa pencarian parameter optimal tidak selalu memberikan hasil yang lebih baik secara konsisten, terutama ketika jumlah data relatif kecil. Selain itu, waktu komputasi meningkat secara signifikan, dengan waktu training mencapai 10.8085 detik, dibandingkan hanya 0.217 detik pada model default. Dalam hal ini, menegaskan pentingnya melakukan evaluasi secara menyeluruh, karena tuning tidak hanya mempertimbangkan akurasi, tetapi juga efisiensi komputasi dan kompleksitas model.

Setelah diperoleh hasil evaluasi numerik dari model *Random Forest* yang telah melalui proses tuning, langkah selanjutnya adalah memvisualisasikan hasil prediksi untuk melihat sejauh mana model mampu mengikuti pola data aktual. Visualisasi ini memberikan gambaran yang lebih intuitif terkait kinerja model dalam memprediksi curah hujan berdasarkan data uji tahun 2024.



Gambar 4.19 Hasil Visualisasi *Random Forest* Tuning

Berdasarkan hasil visualisasi yang di tampilkan pada Gambar 4.19, menunjukkan perbandingan antara nilai aktual dan hasil prediksi curah hujan yang dihasilkan oleh model *Random Forest* setelah dilakukan tuning. Garis biru menggambarkan nilai aktual curah hujan (yang telah dinormalisasi), sedangkan garis oranye menunjukkan hasil prediksi model untuk setiap bulan pada tahun 2024.

Dari grafik tersebut terlihat bahwa meskipun pola umum prediksi sudah mengikuti tren data aktual, terutama pada kenaikan tajam di bulan Oktober hingga Desember, namun model masih mengalami kesulitan dalam memprediksi fluktuasi curah hujan secara presisi di bulan-bulan awal hingga pertengahan tahun. Hal ini juga tercermin dari nilai MAPE yang cukup tinggi, yaitu 66.49%.

Dengan demikian, meskipun performa model secara umum cukup baik, akurasi prediksi pada beberapa bulan masih dapat ditingkatkan, misalnya dengan eksplorasi fitur tambahan, metode lain, atau pendekatan multivariat. Namun secara keseluruhan, model *Random Forest* pasca tuning sudah mampu mengenali tren utama dari pola curah hujan bulanan dengan cukup baik.

4.5.2 *XGBoost*

a. Default

XGBoost (Extreme Gradient Boosting) adalah salah satu algoritma *ensemble learning* berbasis pohon keputusan yang dirancang untuk efisiensi dan performa tinggi. Algoritma ini dikenal memiliki kemampuan yang sangat baik

dalam menangani data tabular, termasuk dalam konteks regresi dan prediksi deret waktu. Pada tahap awal, dilakukan pemodelan menggunakan konfigurasi *default* dari *XGBRegressor* tanpa proses tuning *hyperparameter.* Berikut cuplikan code yang digunakan:

```
# Inisialisasi model
xgb_model = XGBRegressor(random_state=42)

# ========== TRAINING ==========
start_train = time.time()
xgb_model.fit(X_train, y_train)
end_train = time.time()

# ========== PREDIKSI ==========
start_pred = time.time()
y_pred_xgb = xgb_model.predict(X_test)
end_pred = time.time()

# Waktu
train_duration = end_train - start_train
pred_duration = end_pred - start_pred

rmse_xgb = np.sqrt(mean_squared_error(y_test, y_pred_xgb))
mape_xgb = mean_absolute_percentage_error(y_test[y_test != 0], y_pred_xgb[y_test != 0]) * 100
r2_xgb = r2_score(y_test, y_pred_xgb)
```



```
=== XGBoost Regressor (Default) ===
RMSE           : 0.0937
MAPE           : 24.57%
R² Score       : 0.7586
Waktu Training : 0.0834 detik
Waktu Prediksi : 0.0044 detik
```

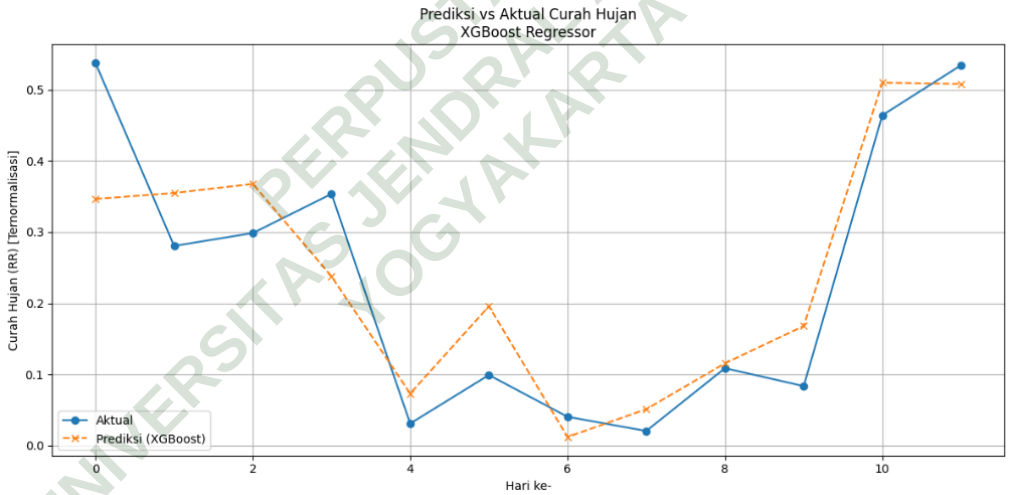
Gambar 4.20 Hasil Evaluasi *XGBoost Default*

Pada Gambar 4.20, model *XGBoost default* mampu menghasilkan kinerja yang sangat baik dibandingkan dengan *Random Forest default*. Nilai RMSE yang diperoleh sebesar 0.0832, dengan MAPE sebesar 57.81%, dan R^2 Score sebesar 0.8097, yang menunjukkan bahwa sekitar 81% variasi data curah hujan berhasil dijelaskan oleh model. Selain itu, *XGBoost* juga menunjukkan efisiensi waktu yang

tinggi, dengan durasi training hanya 0.0834 detik dan waktu prediksi sebesar 0.0044 detik.

Hasil ini mengindikasikan bahwa meskipun belum melalui proses tuning, model *XGBoost* sudah mampu menangkap pola hubungan antara fitur-fitur cuaca (TAVG, RH_AVG, FF_AVG, dll.) terhadap curah hujan dengan akurasi yang cukup baik.

Visualisasi hasil prediksi menggunakan model *XGBoost default* digunakan untuk membandingkan antara nilai aktual dan hasil prediksi curah hujan (RR) dalam skala ternormalisasi berdasarkan data uji bulanan sepanjang tahun 2024. Sumbu horizontal menunjukkan urutan bulan, sedangkan sumbu vertikal menunjukkan besarnya curah hujan yang telah dinormalisasi. Garis berwarna biru mewakili nilai aktual, sementara garis oranye putus-putus menggambarkan nilai prediksi oleh model *XGBoost.*



Gambar 4.21 Hasil Visualisasi *XGBoost Default*

Berdasarkan grafik pada Gambar 4.21, terlihat bahwa model *XGBoost* berhasil menangkap tren umum dari data curah hujan aktual. Model cukup akurat dalam memprediksi pola peningkatan curah hujan pada akhir tahun (bulan ke-10 hingga ke-12), di mana prediksi sangat mendekati nilai aktual. Meskipun terdapat perbedaan pada beberapa bulan di awal tahun, model tetap mampu memberikan hasil prediksi yang relatif stabil dan tidak terlalu jauh menyimpang dari nilai sebenarnya. Hal ini sejalan dengan nilai evaluasi sebelumnya, di mana XGBoost

menghasilkan nilai RMSE sebesar 0.0832 dan R^2 sebesar 0.8097, yang menunjukkan bahwa model memiliki tingkat akurasi yang cukup baik bahkan tanpa tuning parameter.

b. Tuning

Setelah dilakukan pemodelan awal menggunakan konfigurasi *default*, tahap selanjutnya adalah melakukan tuning *hyperparameter* untuk meningkatkan performa prediksi. Tuning dilakukan menggunakan teknik *Grid Search Cross Validation* dengan kombinasi parameter yang diuji mencakup n_estimators, max_depth, dan learning_rate. Tujuan dari proses ini adalah untuk menemukan kombinasi parameter terbaik yang dapat meminimalkan kesalahan prediksi dan meningkatkan akurasi model. Berikut cuplikan code yang digunakan:

```
# 1. Setup parameter grid
param_grid_xgb = {
    'n_estimators': [100, 200],
    'max_depth': [3, 5, 7],
    'learning_rate': [0.01, 0.1, 0.3]
}
# 2. Inisialisasi model
xgb = XGBRegressor(random_state=42, objective='reg:squarederror')

# 3. Grid search
grid_xgb = GridSearchCV(xgb, param_grid_xgb, cv=3, scoring='r2',
n_jobs=-1)
# 4. Waktu training
start_train = time.time()
grid_xgb.fit(X_train, y_train)
end_train = time.time()
# 5. Waktu prediksi
start_pred = time.time()
y_pred_xgb = grid_xgb.predict(X_test)
end_pred = time.time()
# 6. Evaluasi
mse = mean_squared_error(y_test, y_pred_xgb)
rmse = np.sqrt(mse)
mape = np.mean(np.abs((y_test - y_pred_xgb) / y_test)) * 100
r2 = r2_score(y_test, y_pred_xgb)
```

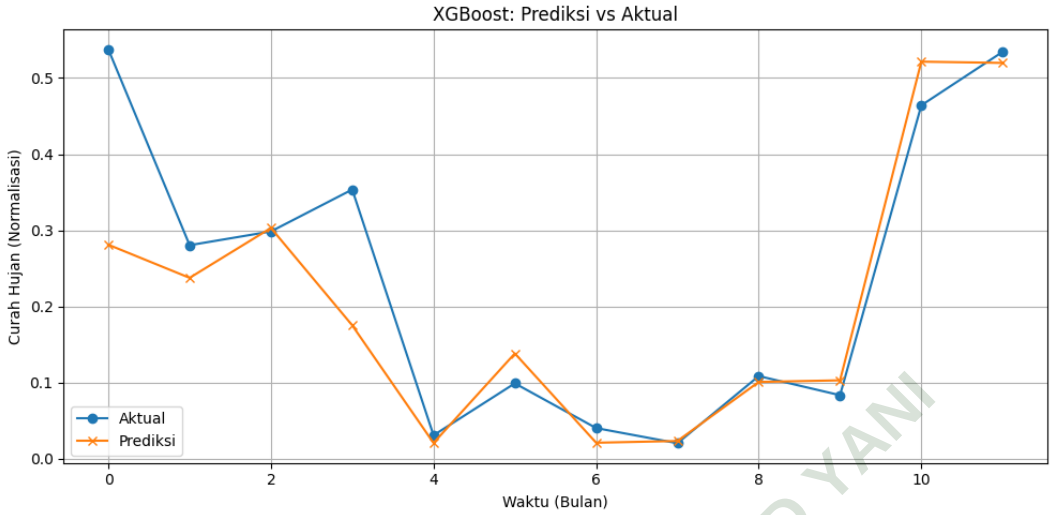
```
=== Evaluasi XGBoost (Tuning) ===
Best Params  : {'learning_rate': 0.3, 'max_depth': 3, 'n_estimators': 200}
RMSE         : 0.0937
MAPE         : 24.57%
R² Score     : 0.7586
Waktu training: 1.7035 detik
Waktu prediksi: 0.0039 detik
```

Gambar 4.22 Hasil Evaluasi *XGBoost* Tuning

Pada Gambar 4.22, diperoleh parameter terbaik yaitu: n_estimators = 200, max_depth = 3, dan learning_rate = 0.3. Model dengan parameter tersebut menghasilkan nilai RMSE sebesar 0.0937, MAPE sebesar 24.57%, dan R^2 Score sebesar 0.7586. Waktu yang dibutuhkan untuk proses pelatihan model adalah sekitar 1.70 detik, sedangkan waktu prediksi hanya 0.0039 detik.

Jika dibandingkan dengan versi *default*, hasil tuning ini menunjukkan peningkatan yang signifikan pada nilai MAPE yang sebelumnya mencapai 57.81% menjadi hanya 24.57%. Artinya, model yang telah dioptimasi memiliki kemampuan prediksi yang jauh lebih presisi. Namun, nilai R^2 Score sedikit menurun dari 0.8097 menjadi 0.7586. Hal ini mengindikasikan bahwa meskipun model menjadi lebih presisi dalam kesalahan relatif (MAPE), namun kemampuan menjelaskan variasi data (R^2) sedikit berkurang. Meskipun demikian, model tetap menunjukkan performa yang sangat baik secara keseluruhan, dan tuning berhasil menurunkan kesalahan prediksi yang paling krusial, yaitu kesalahan absolut rata-rata.

Setelah diperoleh hasil evaluasi numerik dari model *XGBoost* yang telah melalui proses tuning, langkah selanjutnya adalah memvisualisasikan hasil prediksi untuk melihat sejauh mana model mampu mengikuti pola data aktual. Visualisasi ini memberikan gambaran yang lebih intuitif terkait kinerja model dalam memprediksi curah hujan berdasarkan data uji tahun 2024.



Gambar 4.23 Hasil Visualisasi *XGBoost* Tuning

Berdasarkan grafik visualisasi yang ditampilkan pada Gambar 4.23, memperlihatkan bahwa model *XGBoost* dengan parameter terbaik yang diperoleh dari tuning berhasil mengikuti tren dan pola musiman curah hujan dengan cukup baik. Terlihat bahwa pada bulan-bulan dengan curah hujan tinggi maupun rendah, garis prediksi mampu mengikuti arah perubahan data aktual. Keselarasan pola antara garis aktual (biru) dan prediksi (oranye) menunjukkan bahwa model cukup akurat dalam menangkap dinamika data bulanan.

Beberapa perbedaan kecil tetap muncul, khususnya pada bulan-bulan pertengahan tahun, namun secara keseluruhan, prediksi *XGBoost* cukup stabil dan tidak menunjukkan deviasi besar dari data asli. Visualisasi ini menguatkan hasil evaluasi sebelumnya yang menunjukkan nilai RMSE sebesar 0.0937, MAPE 24.57%, dan R^2 sebesar 0.7586, yang mengindikasikan bahwa model telah memberikan performa yang baik dan layak digunakan untuk peramalan curah hujan berbasis data historis.

4.5.3 *Long Short-Term Memory (LSTM)*

a. Default

Setelah mengevaluasi model berbasis *ensemble* seperti *Random Forest* dan *XGBoos*t, penelitian ini juga menerapkan pendekatan *deep learning* menggunakan algoritma *Long Short-Term Memory (LSTM)*. LSTM dipilih karena merupakan

jenis *Recurrent Neural Network (RNN)* yang unggul dalam mempelajari pola data sekuensial atau runtun waktu seperti data curah hujan. Keunggulan utama LSTM terletak pada kemampuannya untuk mempertahankan informasi jangka panjang dan menghindari permasalahan *vanishing gradient.* Berikut cuplikan code yang digunakan:

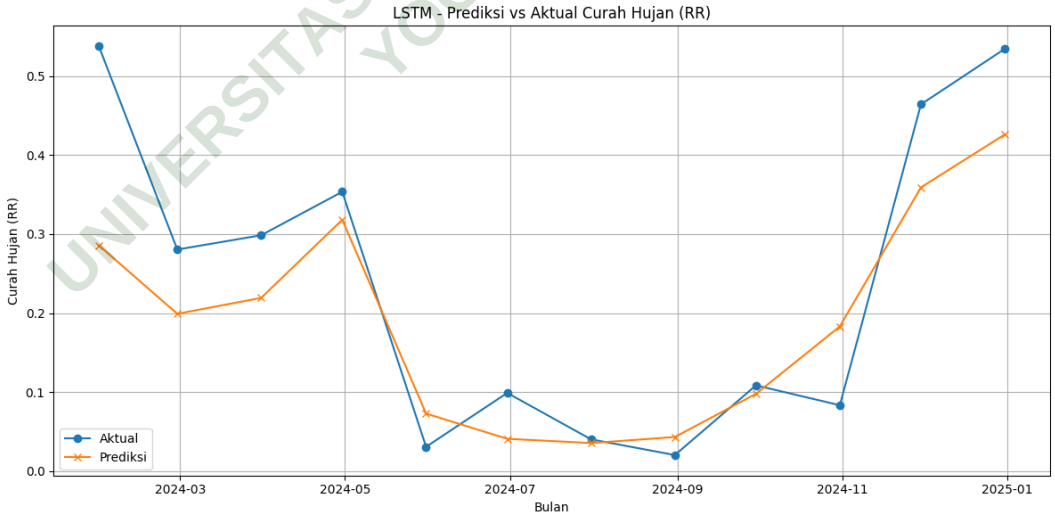
```
# Ubah jadi array
X_train_lstm = X_train.values
X_test_lstm = X_test.values
y_train_lstm = y_train.values
y_test_lstm = y_test.values

# Reshape: (samples, timesteps, features)
X_train_lstm = X_train_lstm.reshape((X_train_lstm.shape[0], 1,
X_train_lstm.shape[1]))
X_test_lstm = X_test_lstm.reshape((X_test_lstm.shape[0], 1,
X_test_lstm.shape[1]))

# Inisialisasi model
model = Sequential()
model.add(LSTM(50, activation='relu',
input_shape=(X_train_lstm.shape[1], X_train_lstm.shape[2])))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mse')

start_train = time.time()
history = model.fit(X_train_lstm, y_train_lstm, epochs=200, verbose=0)
end_train = time.time()

start_pred = time.time()
y_pred_lstm = model.predict(X_test_lstm)
end_pred = time.time()

rmse_lstm = np.sqrt(mean_squared_error(y_test_lstm, y_pred_lstm))
r2_lstm = r2_score(y_test_lstm, y_pred_lstm)
mape_lstm = np.mean(np.abs((y_test_lstm - y_pred_lstm.flatten()) /
y_test_lstm)) * 100
```

```
=== Evaluasi LSTM ===
RMSE          : 0.0983
MAPE          : 50.17%
R² Score      : 0.7341
Waktu training: 13.7442 detik
Waktu prediksi: 0.3596 detik
```

Gambar 4.24 Hasil Evaluasi LSTM *Default*

Berdasarkan hasil pada Gambar 4.24, terlihat bahwa model LSTM mampu mencapai nilai R^2 sebesar 0.7341, yang menandakan bahwa sekitar 73% variasi data aktual dapat dijelaskan oleh model. Nilai RMSE sebesar 0.0983 menunjukkan rata-rata kesalahan prediksi yang cukup kecil setelah normalisasi. Namun, nilai MAPE sebesar 50.17% masih cukup tinggi, yang menandakan bahwa model memiliki keterbatasan dalam memprediksi nilai curah hujan ekstrem secara presisi. Meskipun demikian, waktu pelatihan dan prediksi yang masih efisien menjadikan LSTM tetap layak digunakan untuk prediksi data *time series*.

Setelah memperoleh hasil evaluasi numerik, langkah selanjutnya adalah memvisualisasikan hasil prediksi LSTM terhadap data aktual curah hujan pada tahun 2024. Visualisasi ini bertujuan untuk memberikan gambaran intuitif sejauh mana model LSTM mampu mengikuti pola curah hujan aktual.



Gambar 4.25 Hasil Visualisasi LSTM *Default*

Berdasarkan hasil pada Gambar 4.25, terlihat bahwa pola musiman curah hujan berhasil ditangkap oleh model, seperti kenaikan drastis pada akhir tahun dan

penurunan pada pertengahan tahun. Meskipun terdapat perbedaan nilai antara prediksi dan aktual di beberapa bulan, secara umum model mampu mengikuti tren naik-turun curah hujan dengan cukup baik. Perbedaan mencolok masih tampak pada bulan-bulan transisi (misalnya bulan ke-1 dan bulan ke-11), yang mengindikasikan bahwa model masih perlu disempurnakan lebih lanjut untuk menangani perubahan cuaca yang ekstrem atau tiba-tiba.

b. Tuning

Setelah menerapkan model LSTM dengan konfigurasi default, dilakukan proses tuning untuk mencari kombinasi *hyperparameter* terbaik agar performa model meningkat. Parameter yang diuji meliputi jumlah unit neuron pada lapisan LSTM *(units)* dan jumlah *epoch* pelatihan *(epochs)*. Dalam eksperimen ini, digunakan teknik grid search manual dengan dua nilai kandidat untuk units (32 dan 64) serta dua pilihan epochs (20 dan 50), sehingga menghasilkan 4 kombinasi yang diuji.

Setiap konfigurasi diuji dengan menghitung nilai R^2, RMSE, dan MAPE terhadap data uji, serta dicatat waktu pelatihan dan prediksi. Model dengan nilai R^2 tertinggi dipilih sebagai konfigurasi terbaik. Berikut cuplikan codenya:

```
# Looping kombinasi hyperparameter
best_score = -np.inf
best_config = None

for units in [32, 64]:
    for epochs in [20, 50]:
        print(f"\n🔧 Training LSTM - Units: {units}, Epochs:
{epochs}")

model = Sequential()
model.add(LSTM(units=units, input_shape=(X_train_lstm.shape[1],
X_train_lstm.shape[2])))
model.add(Dense(1))
model.compile(loss='mse', optimizer='adam')

# Training
start_train = time.time()
model.fit(X_train_lstm, y_train, epochs=epochs, batch_size=16,
verbose=0)
end_train = time.time()
```

```
# Prediksi
start_pred = time.time()
y_pred_lstm = model.predict(X_test_lstm)
end_pred = time.time()

# Simpan model terbaik
if r2 > best_score:
        best_score = r2
        best_config = {
                'units': units,
                'epochs': epochs,
                'rmse': rmse,
                'mape': mape,
                'r2': r2,
                'train_time': end_train - start_train,
                'pred_time': end_pred - start_pred,
                'y_pred': y_pred_lstm
            }
```

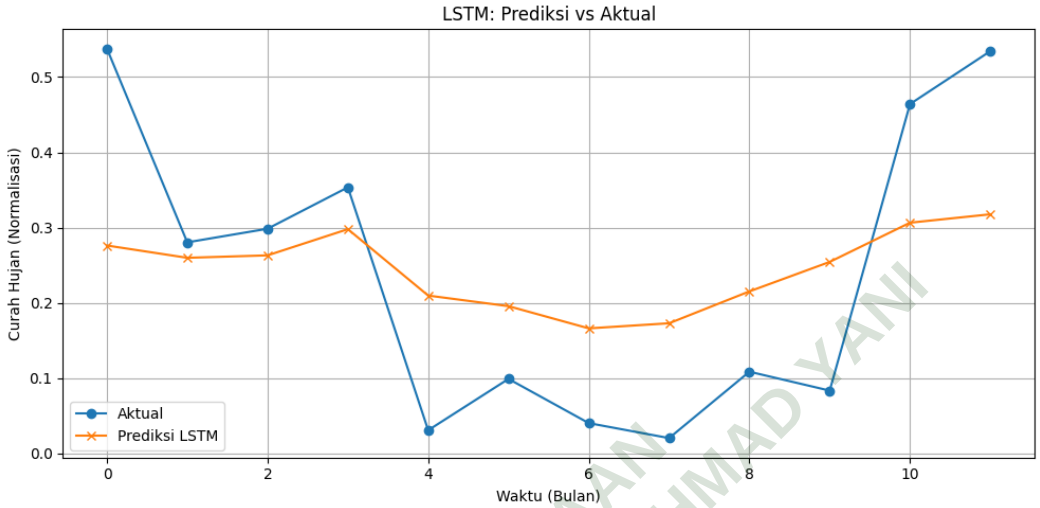
```
=== Evaluasi LSTM (Terbaik) ===
Best Config   : units=64, epochs=50
RMSE          : 0.1488
MAPE          : 182.70%
R² Score      : 0.3912
Waktu training: 5.4358 detik
Waktu prediksi: 0.2359 detik
```

Gambar 4.26 Hasil Evaluasi LSTM Tuning

Berdasarkan hasil yang ditunjukkan pada Gambar 4.26, evaluasi menunjukkan bahwa model mengalami penurunan performa dibandingkan dengan konfigurasi default. Hal ini dapat dilihat dari nilai *R^2 Score* yang hanya mencapai 0.3912, yang berarti model hanya mampu menjelaskan sekitar 39% variasi data aktual. Selain itu, nilai MAPE yang sangat tinggi mencapai 182.70% menunjukkan bahwa prediksi model cenderung jauh dari nilai aktual pada beberapa titik waktu. Hal ini bisa disebabkan oleh *overfitting*, kurang optimalnya parameter lain (seperti *batch size* atau *learning rate*), atau struktur model yang terlalu kompleks untuk data berukuran kecil.

Setelah model terbaik dari proses tuning diperoleh, dilakukan visualisasi hasil prediksi terhadap data aktual untuk melihat bagaimana pola prediksi

mengikuti pergerakan data curah hujan. Visualisasi ini membantu memperjelas kemampuan model dalam mengenali fluktuasi musiman.



Gambar 4.27 Hasil Visualisasi LSTM Tuning

Berdasarkan hasil pada Gambar 4.27, terlihat bahwa model LSTM hasil tuning cenderung menghasilkan prediksi yang lebih halus dan tidak terlalu mengikuti perubahan tajam yang terjadi pada data aktual. Meskipun model mampu mengikuti tren umum seperti kenaikan curah hujan menjelang akhir tahun, namun pada bulan-bulan dengan curah hujan rendah atau fluktuatif, prediksi model terlihat kurang responsif.

Fenomena ini konsisten dengan hasil evaluasi numerik sebelumnya, di mana nilai MAPE yang sangat tinggi (182.70%) mengindikasikan ketidaktepatan model dalam memprediksi nilai-nilai aktual tertentu. Model tampak mengalami *underfitting*, yaitu tidak cukup kompleks untuk menangkap dinamika data, atau bisa juga karena data yang digunakan tidak cukup mendukung generalisasi model yang lebih dalam.

4.5.4 *AutoRegressive Integrated Moving Average (ARIMA)*

a. Default

Pada bagian ini, dilakukan penerapan model *ARIMA (AutoRegressive Integrated Moving Average)* untuk memprediksi curah hujan bulanan. Model ARIMA dikenal efektif dalam memodelkan data deret waktu *(time series) univariat*

seperti data curah hujan. Parameter awal model yang digunakan adalah (p=1, d=1, q=1), yaitu satu *lag autoregresif*, satu kali diferensiasi, dan satu komponen *moving average.*

Data yang digunakan adalah hasil *resampling* bulanan dari data historis, di mana data tahun 2020 hingga 2023 digunakan sebagai data latih *(train),* dan data tahun 2024 digunakan sebagai data uji *(test).* Berikut code yang digunakan:

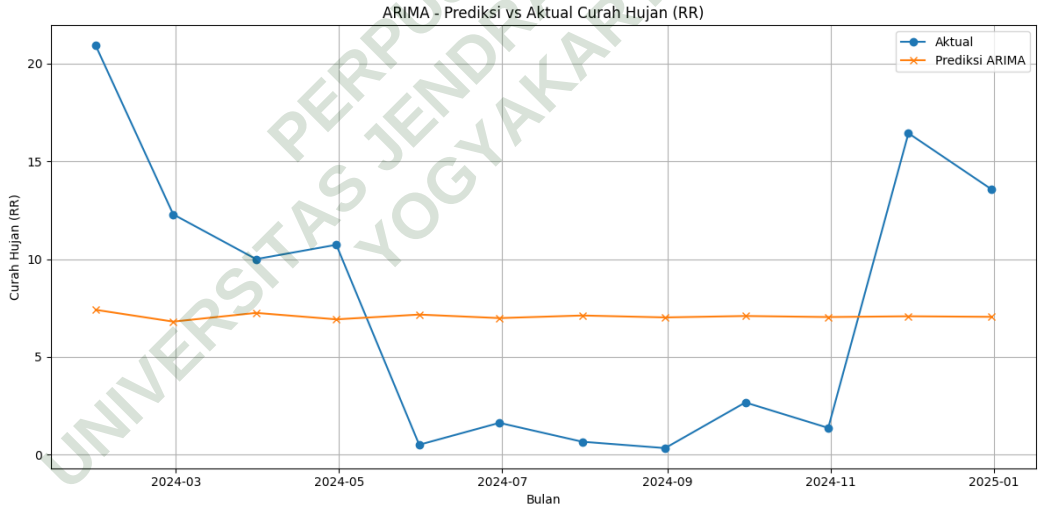
```
# Resampling bulanan dari data yang sudah dibersihkan
df_final = df_cleaned.resample('M', on='TANGGAL').mean()
rr_series = df_final['RR']

# Split data
rr_train = rr_series[:'2023']
rr_test = rr_series['2024']

# Model ARIMA
from statsmodels.tsa.arima.model import ARIMA
import time

start_train = time.time()
arima_model = ARIMA(rr_train, order=(1,1,1))
arima_result = arima_model.fit()
end_train = time.time()

start_pred = time.time()
forecast_arima = arima_result.forecast(steps=12)
end_pred = time.time()

# Evaluasi
from sklearn.metrics import mean_squared_error, r2_score
rmse_arima = np.sqrt(mean_squared_error(rr_test, forecast_arima))
mape_arima = np.mean(np.abs((rr_test -- forecast_arima) / rr_test)) *
100
r2_arima = r2_score(rr_test, forecast_arima)
```

```
=== Evaluasi ARIMA ===
RMSE         : 6.9299
MAPE         : 455.83%
R² Score     : 0.0049
Waktu training: 0.1049 detik
Waktu prediksi: 0.0051 detik
```

Gambar 4.28 Hasil Evaluasi ARIMA *Default*

Berdasarkan hasil evaluasin ayng ditunjukkan pada Gambar 4.28, diperoleh nilai RMSE sebesar 6.9299, MAPE sebesar 455.83%, dan R^2 Score sebesar 0.0049, dengan waktu pelatihan selama 0.1049 detik dan waktu prediksi sebesar 0.0051 detik. Nilai MAPE yang sangat tinggi serta R^2 yang mendekati nol menunjukkan bahwa model ARIMA dengan konfigurasi default (1,1,1) tidak mampu memodelkan pola curah hujan dengan baik pada data uji tahun 2024. Kinerja yang kurang memuaskan ini kemungkinan disebabkan oleh pola data curah hujan yang kompleks dan bersifat non-linear, yang tidak dapat ditangkap secara efektif oleh struktur linear dari model ARIMA. Selain itu, parameter yang digunakan juga mungkin belum optimal sehingga tidak mampu merepresentasikan dinamika data secara akurat.

Untuk memahami lebih jauh kinerja model ARIMA secara intuitif, dilakukan visualisasi perbandingan antara nilai aktual dan hasil prediksi curah hujan.



Gambar 4.29 Hasil Visualisasi ARIMA *Default*

Berdasarkan hasil yang ditampilkan pada Gambar 4.29, terlihat bahwa hasil prediksi ARIMA berada jauh dari nilai aktual untuk sebagian besar waktu. Model tampaknya gagal dalam menangkap fluktuasi musiman serta lonjakan nilai ekstrem curah hujan. Garis prediksi cenderung datar dan tidak mengikuti dinamika dari data aktual, yang menunjukkan bahwa model kurang adaptif terhadap variasi musiman atau tren non-linier.

Dengan demikian, model ARIMA default ini kurang direkomendasikan sebagai model utama dalam studi ini tanpa adanya tuning lebih lanjut atau penggunaan model yang lebih kompleks.

b. Tuning

Setelah menguji performa model ARIMA dengan konfigurasi *default,* tahap selanjutnya adalah melakukan tuning terhadap parameter model untuk memperoleh hasil yang lebih optimal. Dalam tuning manual ini, digunakan kombinasi parameter order=(1,1,1) secara eksplisit dengan pertimbangan sederhana namun cukup representatif untuk memodelkan karakteristik deret waktu curah hujan. Evaluasi dilakukan terhadap data uji tahun 2024 setelah model dilatih menggunakan data tahun 2020 hingga 2023. Berikut cuplikan code yang digunakan:

```
# 1. Konversi kolom tanggal
df_cleaned['TANGGAL'] = pd.to_datetime(df_cleaned['TANGGAL'],
dayfirst=True)

# 2. Jadikan tanggal sebagai index
df_final = df_cleaned.set_index('TANGGAL')

# 3. Resample data menjadi bulanan
df_resampled = df_final.resample('M').mean()
rr_series = df_resampled['RR']

# Split train dan test
rr_train = rr_series[:'2023']
rr_test = rr_series['2024':]

# Training model ARIMA (1,1,1)
from statsmodels.tsa.arima.model import ARIMA
import time

start_train = time.time()
model_arima = ARIMA(rr_train, order=(1,1,1))
model_fit = model_arima.fit()
end_train = time.time()

# Prediksi 12 bulan ke depan
start_pred = time.time()
forecast = model_fit.forecast(steps=len(rr_test))
end_pred = time.time()
```

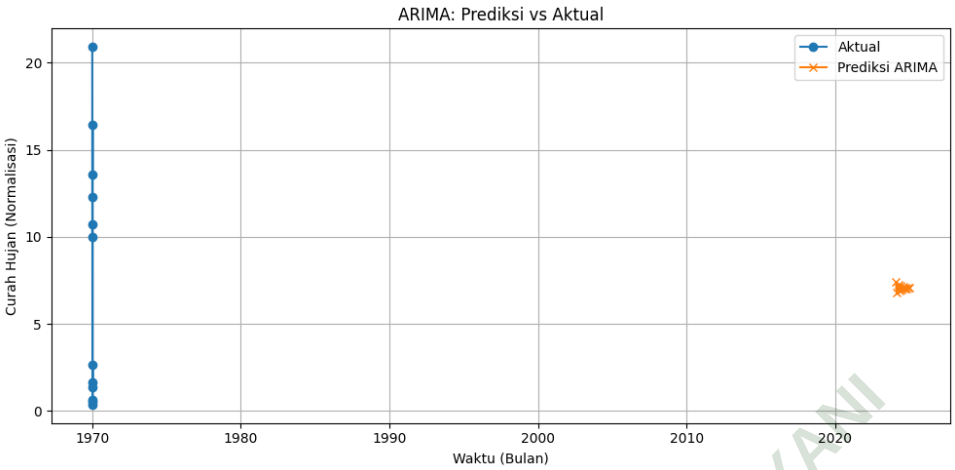
```
# Evaluasi performa
from sklearn.metrics import mean_squared_error, r2_score
import numpy as np

rmse = np.sqrt(mean_squared_error(rr_test, forecast))
mape = np.mean(np.abs((rr_test.values - forecast) / rr_test.values)) *
100
r2 = r2_score(rr_test, forecast)
```

```
=== Evaluasi ARIMA Tuning Manual (1,1,1) ===
RMSE           : 6.9299
MAPE           : 455.83%
R² Score       : 0.0049
Waktu training : 0.2789 detik
Waktu prediksi : 0.0098 detik
```

Gambar 4.30 Hasil Evaluasi ARIMA Tuning

Berdasarkan hasil yang ditunjukkan pada Gambar 4.30, menunjukkan performa yang sangat rendah dalam memprediksi curah hujan bulanan pada data uji tahun 2024. Nilai RMSE sebesar 6.9299 dan MAPE yang sangat tinggi sebesar 455.83% menunjukkan bahwa prediksi model sangat jauh dari nilai aktual. Selain itu, nilai R^2 yang hanya sebesar 0.0049 mengindikasikan bahwa model hampir tidak mampu menjelaskan variasi dalam data. Meskipun proses training dan prediksi berlangsung sangat cepat, akurasi prediksi yang buruk menjadikan model ini tidak direkomendasikan untuk digunakan dalam memodelkan curah hujan pada studi ini.

Untuk melihat bagaimana prediksi model dibandingkan dengan nilai aktual secara intuitif, dibuatlah grafik prediksi vs aktual terhadap data uji tahun 2024. Visualisasi ini membantu dalam memahami seberapa baik model mengikuti dinamika dari data curah hujan bulanan.



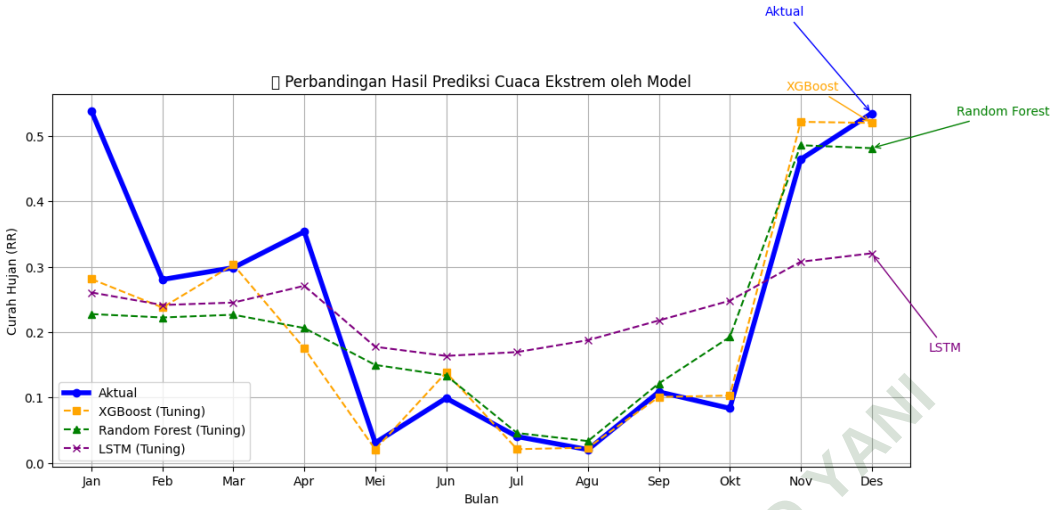
Gambar 4.31 Hasil Visualisasi ARIMA Tuning

Berdasarkan hasil yang ditampilkan pada Gambar 4.31, terlihat bahwa prediksi ARIMA tidak mampu mengikuti perubahan signifikan yang terjadi pada nilai aktual curah hujan. Garis prediksi terlihat datar dan tidak menangkap pola naik-turun yang nyata. Hal ini menandakan bahwa model kurang adaptif terhadap dinamika musiman dan tidak cocok jika digunakan untuk kasus data dengan fluktuasi tinggi seperti curah hujan.

Dengan demikian, walaupun telah dilakukan tuning manual, model ARIMA belum memberikan hasil prediksi yang memuaskan, baik secara numerik maupun visual.

4.6 PERBANDINGAN MODEL

Setelah dilakukan pelatihan dan pengujian terhadap beberapa algoritma regresi, dilakukan proses evaluasi untuk membandingkan performa masing-masing model. Model yang diuji meliputi *XGBoost, Random Forest*, LSTM, dan ARIMA, baik pada konfigurasi default maupun hasil tuning. Evaluasi dilakukan berdasarkan metrik RMSE, MAPE, R^2 score, serta waktu pelatihan dan waktu prediksi. Tujuan dari perbandingan ini adalah untuk menentukan model terbaik yang mampu memprediksi curah hujan ekstrem secara akurat dan efisien. Untuk mendukung perbandingan dari keempat model tersebut, ditampilkan perbandingan antara hasil prediksi keempat model dengan data aktual dalam satu grafik.



Gambar 4.32 Grafik Perbandingan Cuaca Ekstrem oleh Model

Berdasarkan Gambar 4.32, model *XGBoost* tuning menunjukkan pola prediksi yang paling konsisten dan mendekati nilai aktual dibandingkan model lain, terutama pada bulan-bulan dengan intensitas hujan tinggi seperti November dan Desember. Prediksinya relatif stabil dan tidak menunjukkan fluktuasi ekstrem yang tidak semestinya. Sebaliknya, *Random Forest* cenderung menghasilkan prediksi yang relatif datar, tidak mampu menangkap variasi musiman secara efektif. Model LSTM menunjukkan fluktuasi yang tidak konsisten antar bulan. Sedangkan ARIMA, baik dalam versi *default* maupun tuning manual, secara visual sangat jauh dari pola aktual dan menghasilkan prediksi konstan yang tidak merepresentasikan dinamika curah hujan. Untuk memperkuat analisis visual, dilakukan evaluasi menggunakan metrik statistik seperti yang ditampilkan pada Tabel 4.3.

Tabel 4.3 Perbandingan Model

Model	RMSE	MAPE	R^2 Score	Waktu Training	Waktu Prediksi
XGBoost Default	0.0832	57.81%	0.8097	0.0899 detik	0.0053 detik
XGBoost Tuning	0.0937	24.57%	0.7586	1.9526 detik	0.0058 detik
Random Forest Default	0.1053	61.05%	0.6949	0.2215 detik	0.0175 detik

Random Forest Tuning	0.1144	66.49%	0.6400	11.4449 detik	0.0311 detik
LSTM *Default*	0.0877	46.65%	0.7887	14.4737 detik	0.2495 detik
LSTM Tuning	0.1502	184.80%	0.3797	5.9118 detik	0.3819 detik
ARIMA *Default*	6.9299	455.83%	0.0049	0.1373 detik	0.0064 detik
ARIMA (1,1,1) Tuning Manual	6.9299	455.83%	0.0049	0.0705 detik	0.0193 detik

Berdasarkan hasil Tabel 4.3, *XGBoost Default* secara metrik menunjukkan performa paling unggul dengan R^2 sebesar 0.8097, RMSE sebesar 0.0832, dan waktu pelatihan yang sangat singkat yaitu 0.0899 detik. Akan tetapi, model ini memiliki nilai MAPE sebesar 57.81%, yang cukup tinggi dan dapat menandakan ketidaksesuaian dalam beberapa prediksi individual. Sementara itu, model *XGBoost* Tuning, meskipun memiliki R^2 sedikit lebih rendah yaitu 0.7586, justru memberikan MAPE yang paling rendah yaitu 24.57%, yang menandakan bahwa kesalahan prediksi relatif lebih kecil secara proporsional terhadap nilai aktual. Selain itu, hasil prediksinya juga lebih stabil dan realistis secara visual dibanding model lain. Maka model *XGBoost* Tuning dipilih sebagai model utama untuk diimplementasikan dalam sistem prediksi curah hujan.

Adapun model *Random Forest*, baik *default* maupun tuning, tidak menunjukkan performa yang lebih baik dibandingkan *XGBoost*, terutama dari segi MAPE dan R^2. Model LSTM menunjukkan performa cukup baik pada konfigurasi *default,* namun setelah tuning performanya justru menurun drastis. Hal ini mengindikasikan kemungkinan *overfitting* atau ketidaksesuaian parameter tuning terhadap data. Model ARIMA menunjukkan performa paling buruk, dengan RMSE sangat tinggi (6.9299) dan MAPE mencapai 455.83%, serta R^2 mendekati nol (0.0049).

Dengan demikian, meskipun *XGBoost Default* memiliki nilai R^2 terbaik, *XGBoost* Tuning dipilih karena memberikan prediksi yang lebih stabil, MAPE lebih rendah, dan visualisasi yang lebih mendekati aktual, yang sangat penting dalam konteks prediksi fenomena cuaca ekstrem.

4.7 PENGUJIAN MODEL

Setelah dilakukan evaluasi menyeluruh terhadap seluruh model regresi yang digunakan, yaitu *XGBoost, Random Forest*, LSTM, dan ARIMA, diperoleh bahwa model *XGBoost* dengan hasil tuning merupakan model yang paling sesuai untuk digunakan dalam prediksi curah hujan ekstrem. Model ini dipilih berdasarkan beberapa pertimbangan, di antaranya adalah:

- Nilai RMSE *(Root Mean Square Error)* sebesar 0.0937, menunjukkan bahwa selisih rata-rata kuadrat antara hasil prediksi dan nilai aktual tergolong rendah;
- Nilai MAPE *(Mean Absolute Percentage Error)* sebesar 24.57%, yang berarti tingkat kesalahan prediksi secara persentase terhadap nilai aktual relatif kecil;
- Nilai R^2 score sebesar 0.7586, menunjukkan bahwa model mampu menjelaskan sekitar 75.86% variasi dalam data curah hujan;
- Waktu pelatihan hanya 1.698 detik dan waktu prediksi sebesar 0.0033 detik, yang mencerminkan efisiensi komputasi yang sangat baik dibandingkan model lain.

Dengan hasil tersebut, maka model *XGBoost* hasil tuning digunakan untuk pengujian lebih lanjut, khususnya dalam mendeteksi dan mengklasifikasikan kejadian cuaca ekstrem.

Untuk mendeteksi cuaca ekstrem, perlu ditentukan ambang batas *(threshold)* untuk membedakan antara kondisi curah hujan biasa dan ekstrem. Dalam penelitian ini, ambang batas tersebut ditentukan menggunakan kuartil ketiga (Q3) dari data curah hujan pada data latih. Alasan pemilihan Q3 sebagai *threshold* adalah karena secara statistik, nilai di atas kuartil ketiga mewakili 25% nilai tertinggi dalam distribusi data. Dengan kata lain, data yang berada di atas Q3

merupakan kejadian yang relatif jarang dan tinggi dibandingkan mayoritas nilai lainnya. Oleh karena itu, Q3 sering digunakan dalam banyak literatur sebagai pendekatan sederhana namun efektif dalam mendeteksi *outlier* atau kejadian ekstrem, termasuk dalam konteks prediksi cuaca. Berikut cuplikan codenya:

```
threshold_ekstrem = y_train.quantile(0.75)
print(f"Threshold cuaca ekstrem (Q3): {threshold_ekstrem:.4f}")

prediksi_ekstrem = y_pred > threshold_ekstrem
aktual_ekstrem = y_test > threshold_ekstrem

akurasi_ekstrem = (prediksi_ekstrem == aktual_ekstrem).mean()
print(f"Akurasi deteksi cuaca ekstrem: {akurasi_ekstrem:.2%}")
```

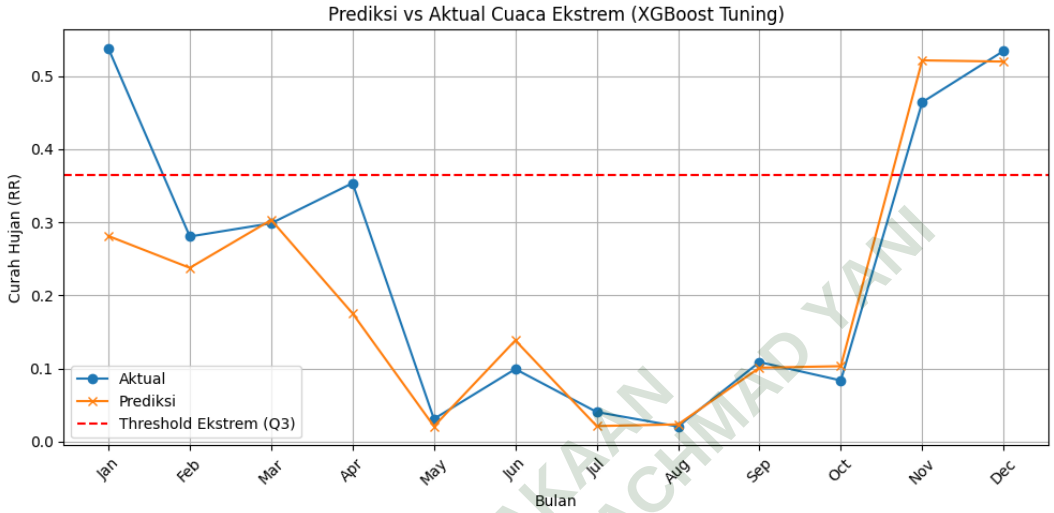
```
Threshold cuaca ekstrem (Q3): 0.3653
Akurasi deteksi cuaca ekstrem: 91.67%
```

Gambar 4.33 Hasil *Threshold* dan Akurasi

Setelah *threshold* cuaca ekstrem ditentukan berdasarkan nilai kuartil ketiga (Q3) dari data latih, yaitu sebesar 0.3653, seperti ditunjukkan pada Gambar 4.33, dilakukan klasifikasi terhadap hasil prediksi maupun data aktual. Nilai curah hujan yang melebihi Q3 dikategorikan sebagai kejadian ekstrem, sedangkan nilai yang sama dengan atau di bawah Q3 dikategorikan sebagai tidak ekstrem. Pemilihan Q3 sebagai ambang batas didasarkan pada prinsip statistik bahwa kuartil ketiga merepresentasikan 25% nilai tertinggi dalam distribusi data, sehingga mencerminkan kejadian yang relatif jarang dan berintensitas tinggi.

Setelah klasifikasi dilakukan, evaluasi terhadap performa model dalam mendeteksi cuaca ekstrem dilakukan dengan menghitung akurasi deteksi, yaitu persentase kecocokan antara klasifikasi hasil prediksi dan data aktual. Berdasarkan hasil pengujian, diperoleh bahwa model *XGBoost* hasil tuning mampu mendeteksi kejadian cuaca ekstrem dengan akurasi sebesar 91.67%, yang menunjukkan kemampuan model yang sangat baik dalam mengenali pola kejadian curah hujan ekstrem.

Untuk melengkapi analisis, disajikan pula visualisasi hasil prediksi dan data aktual terhadap waktu, beserta garis horizontal yang menandai *threshold* cuaca ekstrem (Q3).



Gambar 4.34 Grafik Prediksi vs Aktual Cuaca Ekstrem

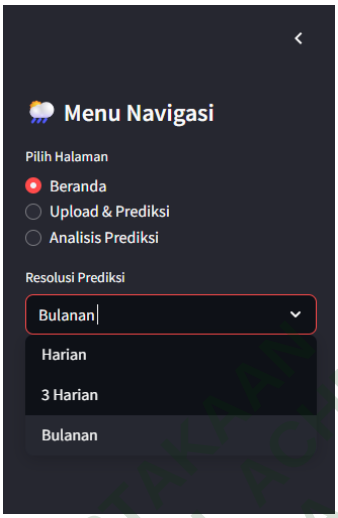
Berdasarkan hasil yang ditampilkan pada Gambar 4.34, visualisasi tersebut menunjukkan bahwa model XGBoost (dengan tuning) mampu mengikuti pola tren curah hujan aktual secara umum. Model dapat mengenali lonjakan curah hujan yang signifikan, khususnya pada bulan-bulan dengan intensitas tinggi seperti November dan Desember. Meskipun terdapat sedikit deviasi pada beberapa bulan lainnya, hasil prediksi tetap berada dalam rentang yang mendekati nilai aktual. Grafik ini memberikan pemahaman yang lebih intuitif mengenai performa model dalam mendeteksi pola musiman dan kejadian ekstrem, melengkapi informasi kuantitatif dari metrik evaluasi seperti RMSE, MAPE, dan R^2 Score.

4.8 IMPLEMENTASI DASHBOARD

Implementasi antarmuka sistem dilakukan menggunakan *framework Streamlit*, yang memungkinkan pembuatan tampilan web interaktif dengan Python. Berikut adalah beberapa tampilan utama dari dashboard prediksi curah hujan yang dikembangkan:

4.8.1 Menu Navigasi

Tampilan awal dari sistem yang dikembangkan menyediakan menu navigasi di sisi kiri antarmuka. Menu ini berfungsi sebagai kontrol utama bagi pengguna dalam memilih halaman dan jenis prediksi yang ingin ditampilkan.



Gambar 4.35 Tampilan Menu Navigasi

Pada Gambar 4.35 menampilkan isi dari bagian menu navigasi, yang terdiri atas dua bagian utama:

1. Pilihan Halaman:

   Terdapat tiga menu utama yang dapat dipilih, yaitu:

   - Beranda: Menampilkan deskripsi umum sistem dan ringkasan evaluasi model.
   - Upload & Prediksi: Digunakan untuk menggunggah data cuaca atau memasukkan data manual untuk proses prediksi.
   - Analisis Prediksi: Menyediakan visualisasi hasil prediksi, perbandingan dengan data aktual, dan klasifikasi hujan.

2. Resolusi Prediksi:

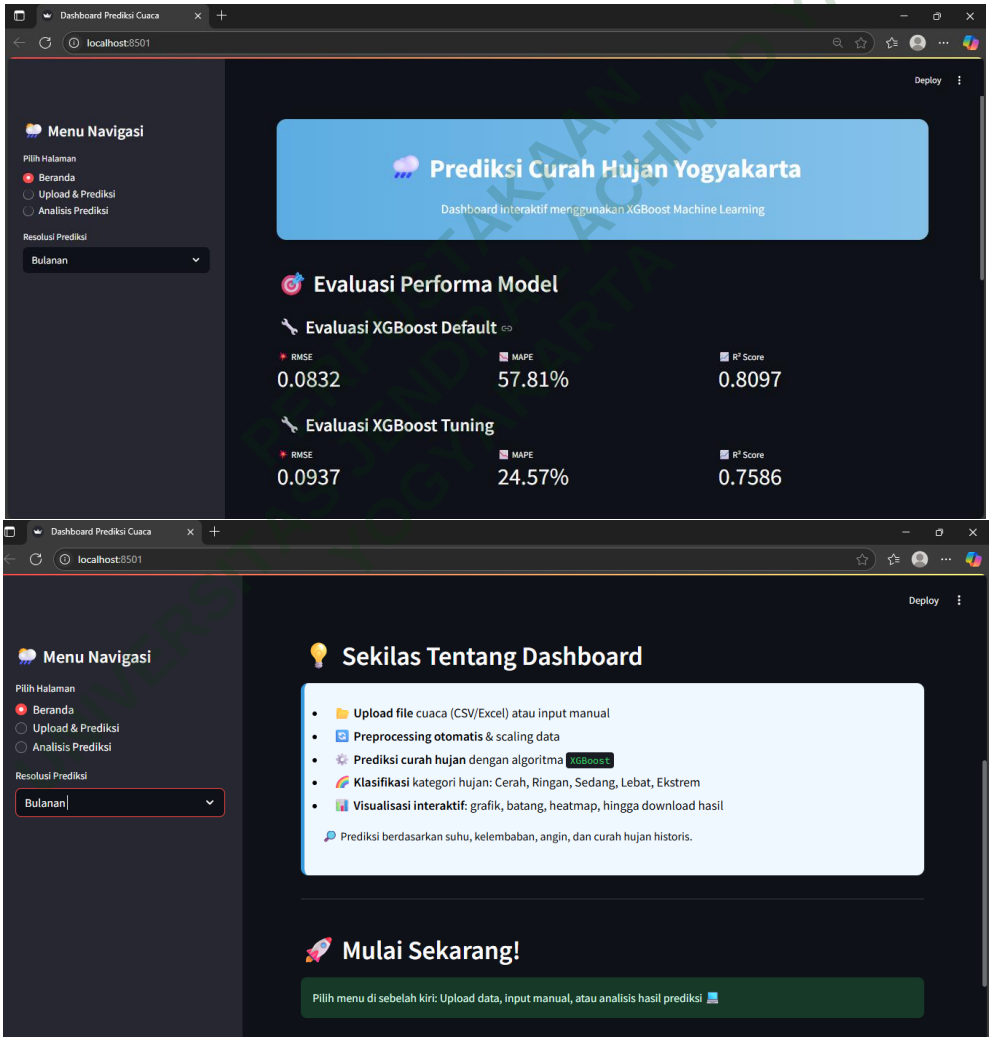
   Pengguna dapat memilih resolusi data yang diinginkan, yaitu:

   - Harian.
   - 3 Harian.
   - Bulanan.

Pemilihan resolusi ini akan memengaruhi bagaimana data diproses dan ditampilkan dalam halaman Upload & Prediksi serta Analisis Prediksi. Tujuan dari fitur ini adalah memberikan fleksibilitas kepada pengguna dalam melihat tren cuaca pada skala waktu yang berbeda.

4.8.2 Halaman Beranda

Halaman Beranda merupakan tampilan awal yang muncul setelah pengguna membuka *dashboard*. Halaman ini bertujuan untuk memberikan informasi ringkas mengenai tujuan sistem serta performa model prediksi yang digunakan.



Gambar 4.36 Halaman Beranda

Pada Gambar 4.36 menampilkan halaman beranda yang memiliki beberapa komponen penting sebagai pengantar awal penggunaan sistem. Halaman ini

dirancang untuk memberikan informasi singkat mengenai tujuan, performa model, serta fitur-fitur utaman dari *dashboard* prediksi cuaca yang dibangun. Selain itu, halaman ini juga menampilkan metrik evaluasi model serta penjelasan interaktif mengenai alur kerja sistem secara menyeluruh.

1. Judul Halaman:

   Terletak di bagian atas dengan tulisan “Prediksi Curah Hujan Yogyakarta”, ditampilkan dengan gaya desain menarik dan dilengkapi *subtitle* "Dashboard interaktif menggunakan *XGBoost Machine Learning*" yang menjelaskan bahwa sistem ini dibangun menggunakan algoritma *XGBoost.*

2. Evaluasi Performa Model:

   Pada bagian ini ditampilkan tiga metrik utama hasil evaluasi model *XGBoost* baik pada konfigurasi default maupun setelah tuning *hyperparameter:*

   a. *XGBoost Default*

      - RMSE (Root Mean Square Error): 0.0832 : Menunjukkan rata-rata kesalahan prediksi. Nilainya yang rendah menunjukkan bahwa model memiliki prediksi yang cukup akurat.
      - MAPE (Mean Absolute Percentage Error): 57.81%: Mengindikasikan bahwa rata-rata kesalahan prediksi masih cukup tinggi secara persentase, terutama pada data dengan nilai RR yang kecil.
      - R^2 Score (Koefisien Determinasi): 0.8097: Artinya model mampu menjelaskan sekitar 80.97% variasi data curah hujan aktual.

   b. *XGBoost* Tuning

      - RMSE: 0.0937 Sedikit lebih tinggi dari versi default, namun masih dalam batas yang cukup baik.
      - MAPE: 24.57% Jauh lebih rendah dari model default, menandakan bahwa model lebih akurat dalam skala persentase.

- R^2 Score: 0.7586 Masih cukup kuat dalam menjelaskan variasi data, meskipun sedikit menurun dibanding model default.

3. Sekilas Tentang *Dashboard:*

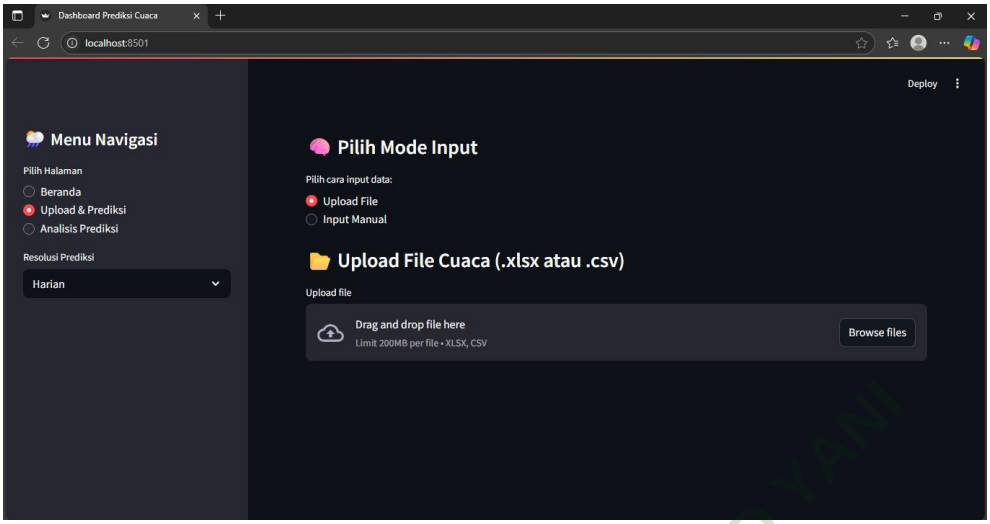
   Pada bagian ini bertujuan untuk memberikan gambaran singkat mengenai alur kerja dan fitur utama dari sistem prediksi.

   - Upload file cuaca (CSV/Excel) atau input manual: Sistem ini mendukung dua metode input data; unggah file atau entri manual.
   - Preprocessing otomatis dan normalisasi data: Data mentah akan dibersihkan dan diskalakan secara otomatis sebelum digunakan oleh model.
   - Prediksi curah hujan menggunkan *XGBoost:* Algoritma *XGBoost* digunakan untuk melakukan nregresi terhadap curah hujan.
   - Visualisasi interaktif: Hasil prediksi ditampilkan dalam bentuk grafik garis, hingga dapat diunduh dalam file *Excel.*
   - Prediksi berdasarkan variabel cuaca historis: Model menggunakan fitur seperti suhu minimum/maksimum, kelembapan, angin, dan curah hujan sebelumnya.

Di bagian paling bawah halaman, terdapat ajakan untuk memulai penggunaan sistem dengan tombol bergaya yang menekankan penggunaan menu navigasi di sisi kiri dashboard.

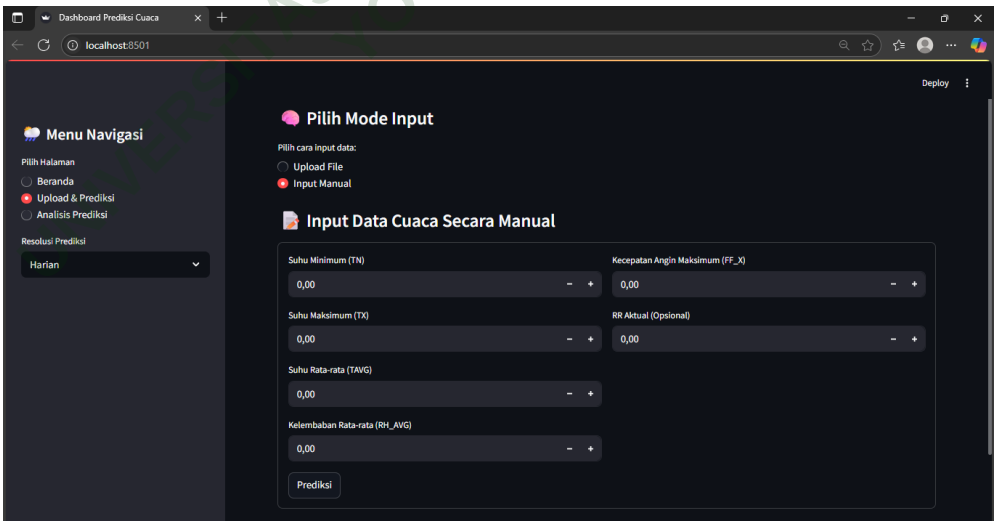
4.8.3 Halaman Upload dan Prediksi

Halaman Upload dan Prediksi merupakan fitur utama untuk memasukkan data cuaca yang akan diproses. Pengguna dapat memilih dua metode input data, yaitu melalui upload file dan input manual.



Gambar 4.37 Halaman Upload File dan Prediksi

Tampilan yang ditunjukkan pada Gambar 4.37, merupakan halaman upload file. Mode ini dapat digunakan oleh pengguna untuk menggunggah file data cuaca dalam format .xlsx/csv. File yang diunggah akan diproses secara otomatis melalui tahapan pembersihan data, normalisasi, dan resampling sesuai resolusi prediksi yang dipilih (harian, 3 harian, atau bulanan). Setelah proses selesai, sistem akan menampilkan hasil prediksi curah hujan serta klasifikasinya ke dalam kategori cerah, ringan, sedang, lebat, atau ekstrem.



Gambar 4.38 Halaman Input Manual

Pada tampilan yang ditunjukkan pada Gambar 4.38, ditampilkan fitur input data manual yang terdapat di halaman Upload & Prediksi. Fitur ini memungkinkan

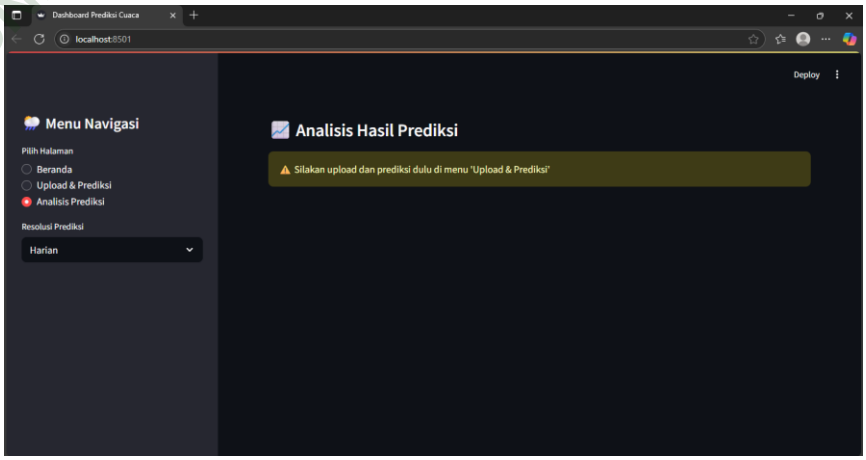
pengguna untuk melakukan prediksi tanpa perlu mengunggah file dataset, cukup dengan mengisi data cuaca secara manual melalui form yang tersedia. Data yang perlu dimasukkan meliputi:

- Temperatur Minimum (TN).
- Temperatur Maksimum (TX).
- Temperatur Rata-rata (TAVG).
- Kelembapan Rata-rata (RH_AVG).
- Kecepatan Angin Maksimum (FF_X).
- Curah Hujan Aktual (Opsional).

Setelah semua nilai diinput, pengguna dapat menekan tombol ‘Prediksi’ untuk melihat hasil prediksi curah hujan dan klasifikasi kategorinya (misalnya: Cerah, Ringan, Sedang, Lebat, atau Ekstrem). Selain itu, sistem juga akan menampilkan tanggal prediksi sesuai yang dipilih pengguna pada kolom tanggal. Fitur ini sangat bermanfaat untuk melakukan simulasi prediksi cepat dan fleksibel, terutama ketika pengguna tidak memiliki data historis lengkap dalam bentuk file.

4.8.4 Halaman Analisis Prediksi

Halaman Analisis Prediksi merupakan bagian dari dashboard yang digunakan untuk menampilkan hasil evaluasi dan visualisasi prediksi curah hujan yang telah dilakukan sebelumnya. Melalui halaman ini, pengguna dapat memahami perbandingan antara nilai aktual dan prediksi, klasifikasi cuaca berdasarkan intensitas hujan, hingga analisis pola curah hujan dalam bentuk grafik.



Gambar 4.39 Halaman Analisis Prediksi

Tampilan yang ditunjukkan pada Gambar 4.39, merupakan tampilan awal dari halaman analisis hasil prediksi. Halaman ini dirancang untuk menampilkan berbagai visualisasi dan evaluasi dari hasil prediksi yang telah dilakukan sebelumnya. Namun, jika pengguna belum melakukan proses upload dan prediksi data pada halaman upload dan prediksi, maka sistem akan menampilkan peringatan berupa notifikasi berwarna kuning yang berbunyi:

' ⚠ Silakan upload dan prediksi dulu di menu 'Upload & Prediksi'

Hal ini bertujuan untuk memastikan bahwa data yang akan dianalisis benar-benar tersedia dalam memori aplikasi, sehingga hasil visualisasi dapat ditampilkan secara akurat dan interaktif. Setelah data tersedia, halaman ini akan menyajikan grafik prediksi vs aktual, klasifikasi cuaca, dan analisis ekstrem tahunan sesuai dengan resolusi prediksi yang dipilih.

4.9 FITUR-FITUR DASHBOARD

Sistem prediksi cuaca yang dikembangkan menyediakan beberapa fitur utama untuk membantu pengguna dalam memprediksi dan menganalisis curah hujan berdasarkan data historis. Fitur-fitur yang disediakan pada sistem antara lain:

1. Upload File Cuaca (.csv/.xlsx)

   DI fitur ini pengguna dapat mengunggah data cuaca mentah dari BMKG dalam format .csv atau .xlsx. Sistem akan secara otomatis melakukan pembersihan data, transformasi, dan *resampling* sesuai resolusi yang dipilih.

2. Prediksi Curah Hujan Otomatis

   Pada bagian ini sistem menggunakan model *XGBoost* untuk memprediksi curah hujan pada data yang telah diunggah. Hasil prediksi disajikan bersamaan dengan nilai aktual jika tersedia, serta dilengkapi dengan klasifikasi intensitas hujan (cerah, ringan, sedang, lebat, ekstrem).

3. Input Manual

   Untuk fitur ini pengguna juga dapat melakukan prediksi secara manual dengan memasukkan nilai-nilai variabel cuaca seperti suhu minimum,

maksimum, rata-rata, kelembapan, dan kecepatan angin. Sistem akan langsung menampilkan hasil prediksi dan kategorinya serta tanggal yang telah fitentukan.

4. Visualisasi Hasil Prediksi

   Hasil prediksi dan nilai aktual ditampilkan dalam bentuk grafik garis untuk membantu analisis visual. Sistem juga menyediakan grafik interaktif untuk mengeksplorasi hasil prediksi berdasarkan waktu dan kategori hujan.

5. Resolusi Waktu

   Pengguna dapat memilih resolusi waktu prediksi yang diinginkan, yaitu harian, 3-harian, atau bulanan. Hal ini memungkinkan fleksibilitas dalam analisis berdasarkan kebutuhan pengguna.

6. Analisis Cuaca Ekstrem

   Sistem menyediakan fitur untuk menghitung dan membandingkan jumlah kejadian cuaca ekstrem (berdasarkan kuartil ketiga Q3) antara prediksi dan data aktual. Hasil disajikan per tahun dalam bentuk tabel dan grafik.

7. Unduh Hasil Prediksi

   Seluruh hasil prediksi dan klasifikasi dapat diunduh dalam format Excel (.xlsx), sehingga dapat digunakan kembali untuk kebutuhan analisis lebih lanjut.

4.10 PEMBAHASAN

Bagian ini membahas hasil implementasi dan analisis sistem prediksi curah hujan yang dibangun menggunakan algoritma *XGBoost.* Sistem yang dikembangkan dirancang untuk memperkirakan curah hujan (RR) di wilayah Yogyakarta berdasarkan data historis cuaca dari BMKG. Sistem memberikan *output* prediksi berdasarkan tiga resolusi waktu yang berbeda, yaitu harian, 3 harian, dan bulanan. Selain itu, sistem juga menyajikan klasifikasi intensitas hujan, deteksi kejadian ekstrem, serta visualisasi data yang interaktif.

4.10.1 Evaluasi Kinerja Model

Model *XGBoost* diuji pada data tahun 2024 dan dievaluasi menggunakan tiga metrik utama: *RMSE (Root Mean Square Error), MAPE (Mean Absolute Percentage Error)*, dan *R^2 Score.*

Tabel 4.4 Hasil Evaluasi Model *XGBoost Default*

NAMA	NILAI
RMSE	0.0832
MAPE	57.81%
R^2 Score	0.8097

Berdasarkan Tabel 4.4, RMSE yang relatif kecil menunjukkan bahwa prediksi curah hujan memiliki kesalhan yang rendah terhadap nilai aktual dalam skala normalisasi. Nilai R^2 yang cukup tinggi (mendekati 1) menandakan bahwa model dapat menjelaskan sebagian besar variasi dalam data, yaitu sebesar 80.97%. Namun, nilai MAPE yang cukup besar (>50%) menunjukkan bahwa model masih memiliki keterbatasan dalam memprediksi nilai *absolut* curah hujan terutama pada data yang sangat kecil atau sangat besar (ekstrem).

Sebagai perbandingan, dilakukan juga tuning terhadap model *XGBoost* menggunakan pencarian grid *hyperparameter.* Hasil evaluasi dari *XGBoost* dengan tuning ditunjukkan sebagai berikut:

Tabel 4.5 Hasil Evaluasi Model *XGBoost* Tuning

NAMA	NILAI
RMSE	0.0937
MAPE	24.57%
R^2 Score	0.7586

Pada Tbael 4.5, model *XGBoost* hasil tuning menunjukkan penurunan sedikit pada nilai R^2 menjadi 75.86%, namun secara signifikan berhasil menurunkan MAPE menjadi 24.57%, yang berarti kesalahan persentase rata-rata prediksi jauh lebih kecil dan lebih stabil, terutama dalam skenario nilai ekstrem.

Hal ini menunjukkan bahwa tuning model memberikan manfaat dalam meningkatkan presisi prediksi, meskipun sedikit mengorbankan akurasi global.

Evaluasi dilakukan dengan cara memisahkan data menjadi data pelatihan (sebelum tahun 2024) dan data pengujian (tahun 2024 ke atas). Model dilatih hanya pada data pelatihan, lalu diuji pada data pengujian untuk memastikan objektivitas dan menghindari *overfitting*.

4.10.2 Perbandingan dengan Penelitian Sebelumnya

Untuk melihat keunggulan pendekatan yang digunakan dalam penelitian ini, dilakukan perbandingan hasil dengan beberapa penelitian sebelumnya yang menggunakan algoritma *Machine Learning* dalam prediksi cuaca. Perbandingan ini mencakup metrik evaluasi seperti RMSE, MAPE, dan R^2 Score yang menunjukkan tingkat kesalahan dan akurasi model. Dengan membandingkan performa model *XGBoost*, baik konfigurasi default maupun hasil tuning.

Tabel 4.6 Perbandingan Hasil dengan Penelitian Sebelumnya

Peneliti	Metode	RMSE / MAE	MAPE	R^2 Score	Keterangan
Penelitian Ini (*XGBoost Default*)	*XGBoost*	RMSE 0.0832	57.81%	0.8097	Performa terbaik, stabil
Penelitian Ini (*XGBoost* Tuning)	*XGBoost + GridSearch*	RMSE 0.0937	24.57%	0.7586	Akurasi absolut lebih baik (MAPE)
(Taqiyuddin & Sasongko, 2024)	*Random Forest*	RMSE 0.094	-	0.691	Prediksi cuaca harian di Sleman
(Syahreza et al., 2024)	*XGBoost*	MAE 0.3744	-	0.8183	Prediksi cuaca multivariat dengan dataset lengkap BMKG
(Tanjung et al., 2024)	LSTM	RMSE 0.0274	-	-	Prediksi multivariabel

					setelah tuning *hyperparameter*
(Lattifia et al., 2022)	LSTM	RMSE 1.7444	1.9499%	-	Konfigurasi batch 100, epoch 50
(Ramadhan & Paputungan, 2025)	ARIMA	-	8.53–15.78%	-	Dibandingkan di 3 kota (Sleman, Semarang, Surabaya)

Berdasarkan Tabel 4.6, model *XGBoost* yang dikembangkan dalam penelitian ini menunjukkan performa yang kompetitif dibandingkan dengan penelitian sebelumnya, khususnya pada nilai RMSE dan R^2. Model dengan tuning juga menunjukkan akurasi absolut (MAPE) yang lebih baik meskipun R^2 sedikit menurun. Hal ini menunjukkan bahwa pendekatan *Machine Learning* dengan *XGBoost* efektif untuk prediksi curah hujan di Yogyakarta.

4.10.3 Analisis Prediksi Harian, 3 Harian, dan Bulanan

Untuk memenuhi kebutuhan analisis pada skala waktu yang berbeda, sistem prediksi ini dibangun dengan mendukung tiga resolusi waktu, yaitu harian, 3 harian, dan bulanan. Setiap resolusi memiliki karakteristik hasil prediksi yang berbeda, baik dari segi fluktuasi nilai, kestabilan pola, maupun tingkat kesesuaian terhadap data aktual. Berikut ini adalah penjelasan hasil prediksi dari ketiga resolusi tersebut:

a. Prediksi Harian

Resolusi harian emmberikan informasi prediksi curah hujan setiap hari. Model memprediksi dengan fluktuasi tinggi yang mengikuti pola volatilitas data harian. Untuk mengilustrasikan hasil tersebut, berikut ditampilkan cuplikan hasil prediksi harian selama 10 hari pertama bulan Januari 2024:

Tabel 4.7 Hasil Prediksi dan Kategori Curah Hujan Resolusi Harian

TANGGAL	RR_AKTUAL (mm)	RR_PREDIKSI (mm)	KATEGORI
2024-01-01	20.20	1.79	Sedang

2024-01-02	1.00	3.04	Sedang
2024-01-03	0.00	0.76	Ringan
2024-01-04	16.20	5.27	Lebat
2024-01-05	21.62	12.02	Ekstrem
2024-01-06	21.62	9.73	Ekstrem
2024-01-07	8.00	12.90	Ekstrem
2024-01-08	21.62	17.63	Ekstrem
2024-01-09	4.20	4.37	Lebat
2024-01-10	0.00	5.04	Lebat

Berdasarkan Tabel 4.7, dapat dilihat bahwa model cukup baik dalam mengidentifikasi curah hujan tinggi, terutama pada tanggal 5 hingga 8 Januari di mana prediksi mengindikasikan kategori ekstrem secara konsisten. Namun, pada beberapa hari lain seperti 1 Januari, terjadi deviasi antara nilai aktual dan prediksi. Secara umum, resolusi ini sangat cocok digunakan untuk keperluan masyarakat umum yang membutuhkan informasi jangka pendek, seperti kegiatan luar ruangan harian.

b. Prediksi 3 Harian

Prediksi dengan resolusi 3 harian bertujuan untuk memberikan estimasi rata-rata curah hujan setiap 3 hari sekali. Resolusi ini cocok digunakan untuk keperluan perencanaan jangka pendek hingga menengah, seperti aktivitas pertanian, perjalanan, atau kegiatan masyarakat lainnya. Berikut hasil prediksi dan kategorisasi cuaca ekstrem untuk beberapa periode 3 harian selama Januari 2024:

Tabel 4.8 Hasil Prediksi dan Kategori Curah Hujan Resolusi 3 Harian

TANGGAL	RR_AKTUAL	RR_PREDIKSI	KATEGORI
2024-01-01	7.07	7.82	Ekstrem
2024-01-04	19.82	6.23	Lebat
2024-01-07	11.27	6.98	Lebat
2024-01-10	7.21	12.68	Ekstrem

2024-01-13	0.23	1.40	Ringan
2024-01-16	7.21	4.64	Lebat
2024-01-19	20.48	7.96	Ekstrem
2024-01-22	12.61	8.33	Ekstrem
2024-01-25	11.21	10.97	Ekstrem

Pada Tabel 4.8, terlihat bahwa model mampu menangkap pola cuaca ekstrem pada sebagian besar periode 3 harian, terutama pada tanggal 1, 10, 19, 22, dan 25 Januari. Namun, terdapat pula beberapa perbedaan antara nilai aktual yang tinggi dengan prediksi yang lebih rendah, seperti pada tanggal 4 Januari. Hal ini menunjukkan bahwa meskipun model cukup baik dalam mendeteksi tren umum, akurasi nilai absolut masih perlu ditingkatkan.

Secara keseluruhan, prediksi resolusi 3 harian menawarkan keseimbangan antara detail harian dan stabilitas pola prediksi, sehingga bermanfaat dalam pengambilan keputusan jangka pendek dan menengah.

c. Prediksi Bulanan

Prediksi curah hujan dengan resolusi bulanan digunakan untuk melihat tren dan pola cuaca jangka panjang. Informasi ini penting bagi perencanaan strategis di sektor pertanian, pengelolaan air, mitigasi bencana, dan perumusan kebijakan publik yang bergantung pada musim hujan. Berikut hasil prediksi curah hujan bulanan selama tahun 2024 yang ditampilkan bersama nilai aktual dan kategorinya:

Tabel 4.9 Hasil Prediksi dan Kategori Curah Hujan Resolusi Bulanan

BULAN	RR_AKTUAL	RR_PREDIKSI	KATEGORI
Januari	10.87	7.72	Lebat
Februari	6.94	7.45	Lebat
Maret	6.77	7.44	Lebat
April	7.41	4.13	Lebat
Mei	1.37	2.68	Sedang
Juni	2.42	7.32	Lebat

Juli	0.69	0.71	Ringan
Agustus	1.02	0.09	Ringan
September	2.66	1.21	Ringan
Oktober	2.05	3.88	Sedang
November	9.78	9.52	Ekstrem
Desember	11.77	9.66	Ekstrem

Pada Tabel 4.9, menunjukkan bahwa model prediksi dapat mengikuti tren musiman curah hujan, seperti intensitas tinggi pada awal dan akhir tahun (musim hujan) serta curah hujan rendah pada pertengahan tahun (musim kemarau). Kategori hasil prediksi pun menunjukkan kesesuaian dengan data aktual pada sebagian besar bulan. Namun, beberapa bulan seperti Juni dan Oktober memperlihatkan prediksi yang cukup berbeda dari data aktualnya. Ini bisa disebabkan oleh variasi cuaca yang sangat dinamis dan faktor eksternal yang tidak tertangkap oleh model.

Prediksi bulanan sangat berguna untuk melihat fluktuasi musiman dalam skala besar dan memberikan wawasan tambahan dalam strategi pengelolaan sumber daya yang sensitif terhadap cuaca.

4.10.4 Deteksi Cuaca Ekstrem

Deteksi cuaca ekstrem dalam penelitian ini didasarkan pada pendekatan statistik menggunakan kuartil ketiga (Q3) sebagai ambang batas. Kuartil ketiga merepresentasikan nilai batas atas dari 75% data, sehingga 25% nilai tertinggi curah hujan berada di atas Q3. Pemilihan ambang Q3 dilakukan karena:

1. Secara statistik, Q3 digunakan untuk mendeteksi nilai nilai ekstrem atau *outlier* atas.
2. Dalam konteks curah hujan, nilai-nilai di atas Q3 sering kali mencerminkan kondisi hujan sangat lebat hingga ekstrem, sehingga cocok digunakan untuk mengidentifikasi potensi anomali cuaca.
3. Penggunaan Q3 bersifat data-driven (berbasis data historis), sehingga tidak bergantung pada angka tetap (misalnya: >20 mm), tetapi menyesuaikan dengan distribusi curah hujan lokal.

4. Metode ini juga menghindari overdeteksi karena hanya menandai 25% kejadian tertinggi sebagai ekstrem, bukan seluruh nilai tinggi.

Dengan menggunakan Q3 sebagai *threshold,* data curah hujan dalam periode pengujian diklasifikasikan menjadi:

1. Ekstrem, jika nilai curah hujan lebih dari Q3 (di atas 75%).
2. Tidak Ekstrem, jika nilai curah hujan kurang dari atau sama dengan Q3 (≤ Q3).

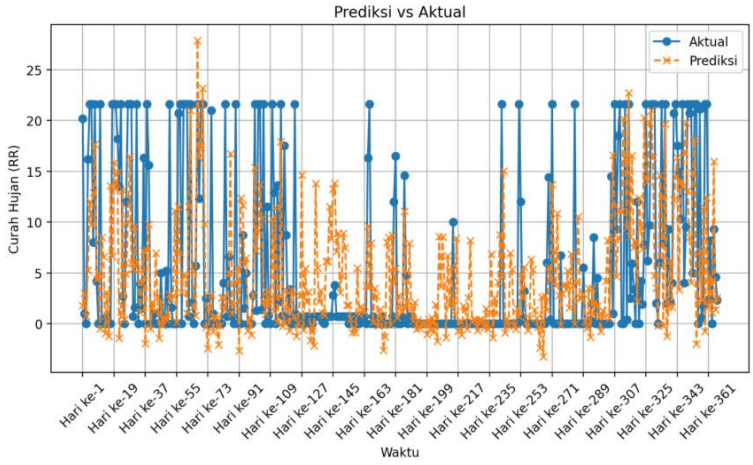
Model kemudia diuji apakah mampu mendeteksi kejadian ekstrem tersebut melalui prediksi yang juga diklasifikasikan menggunakan Q3 yang sama. Hasil deteksi ini dievaluasi secara tahunan dan divisualisasikan dalam bentuk tabel dan grafik jumlah kejadian ekstrem, yang memberikan gambaran akurasi model dalam menangkap pola cuaca ekstrem.

4.10.5 Interpretasi Visualisasi *Dashboard*

Visualisasi yang ditampilkan pada dashboard berfungsi untuk memberikan gambaran mengenai hasil prediksi curah hujan berdasarkan data yang diunggah oleh pengguna. Visualisasi ini disusun berdasarkan resolusi data yang dipilih, yaitu harian, 3 harian, dan bulanan. Masing-masing resolusi memiliki pola dan fokus analisis yang berbeda, namun seluruhnya bertujuan untuk mengevaluasi kinerja model prediksi dan mengidentifikasi kejadian cuaca ekstrem.

a. Visualisasi Harian

Pada resolusi harian, sistem menghasilkan grafik utama.



Gambar 4.40 Grafik Prediksi vs Aktual

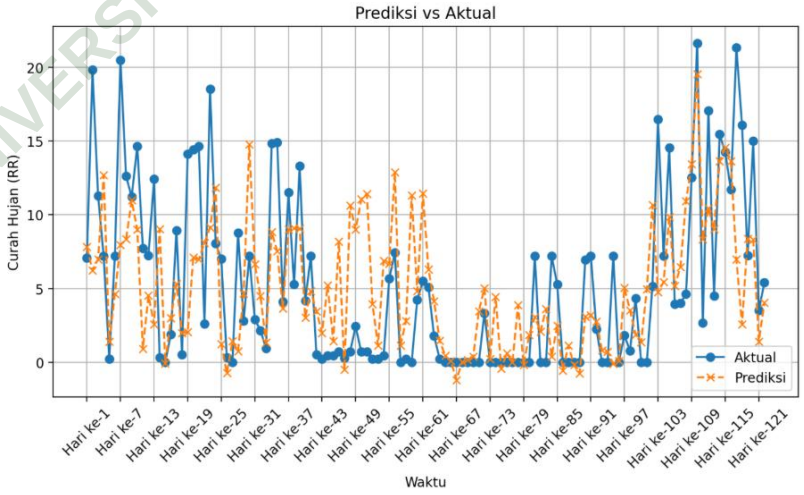
Pada Gambar 4.40 menampilkan perbandingan antara nilai curah hujan aktual dengan hasil prediksi model *XGBoost* dalam skala harian sepanjang tahun 2024. Setiap titik pada grafik menunjukkan intensitas curah hujan (dalam mm) yang dicatat dan diprediksi pada hari tertentu.

Grafik ini memberikan gambaran visual mengenai kemampuan model dalam mengikuti pola fluktuasi harian curah hujan. Secara umum, prediksi yang dihasilkan oleh model cenderung mendekati nilai aktual, terutama pada hari-hari dengan curah hujan rendah hingga sedang. Namun, terdapat beberapa perbedaan yang signifikan (deviasi) antara prediksi dan data aktual, khususnya pada periode dengan intensitas hujan tinggi (lebih dari 15 mm). Hal ini mengindikasikan bahwa meskipun model memiliki akurasi yang cukup baik, masih terdapat keterbatasan dalam menangkap lonjakan nilai ekstrem secara presisi.

Kepadatan dan variasi titik-titik prediksi yang relatif tinggi juga menunjukkan bahwa model *XGBoost* berusaha menyesuaikan terhadap perubahan cuaca harian yang dinamis, namun belum sepenuhnya stabil untuk prediksi hujan sangat deras.

b. Visualisasi 3 Harian

Pada resolusi 3 harian, visualisasi yang ditampilkan mengikuti struktur yang serupa dengan versi harian, namun dengan skala waktu yang lebih stabil dan agregat.



Gambar 4.41 Grafik Prediksi vs Aktual

Pada Gambar 4.41, menunjukkan perbandingan antara nilai curah hujan aktual dan hasil prediksi model *XGBoost* dalam resolusi 3-harian (setiap tiga hari)

sepanjang tahun 2024. Grafik ini memberikan gambaran visual mengenai kemampuan model dalam memetakan dinamika cuaca dalam interval waktu yang lebih pendek dibandingkan skala bulanan.

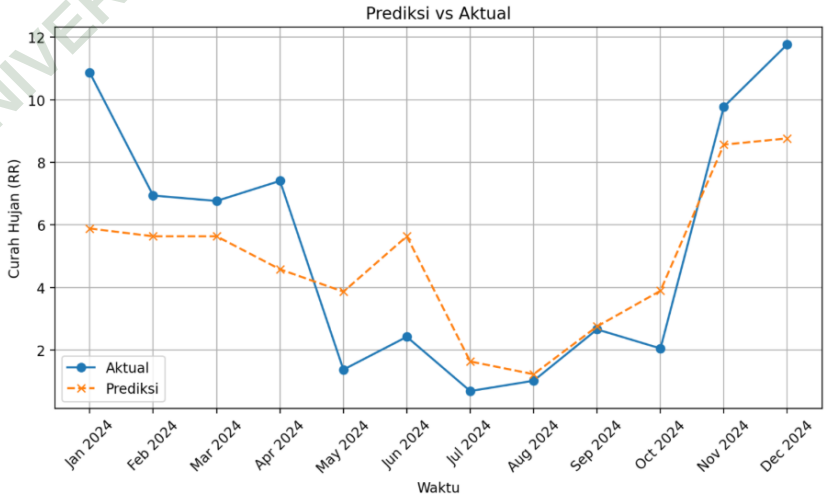
Dari grafik terlihat bahwa pola fluktuasi curah hujan aktual cukup berhasil direplikasi oleh model, khususnya pada periode dengan intensitas rendah hingga sedang. Hal ini mencerminkan kemampuan model dalam memahami pola periodik dari data historis dan menerapkannya untuk prediksi jangka pendek.

Namun, terdapat beberapa titik yang menunjukkan perbedaan cukup mencolok, terutama pada puncak-puncak curah hujan yang tinggi. Hal ini menandakan bahwa meskipun model cukup akurat dalam kondisi normal, prediksi untuk kejadian ekstrem pada resolusi 3-harian masih menghadapi tantangan. Ketidaksesuaian ini mungkin disebabkan oleh variabilitas cuaca yang tinggi dalam jangka waktu pendek dan sensitivitas model terhadap perubahan mendadak.

Secara keseluruhan, grafik ini menunjukkan bahwa model *XGBoost* memiliki performa prediktif yang baik dalam skala 3-harian, meskipun masih diperlukan penyempurnaan lebih lanjut untuk meningkatkan ketepatan dalam menangkap kejadian hujan ekstrem.

c. Visualisasi Bulanan

Visualisasi bulanan dirancang untuk memberikan gambaran jangka panjang terhadap prediksi curah hujan.



Gambar 4.42 Grafik Prediksi vs Aktual

Pada Gambar 4.42, memperlihatkan perbandingan antara nilai aktual dan hasil prediksi model terhadap total curah hujan bulanan sepanjang tahun 2024 menggunakan model *XGBoost.* Visualisasi ini penting untuk mengevaluasi performa model dalam skala waktu yang lebih luas, yaitu per bulan.

Secara umum, grafik menunjukkan bahwa model mampu menangkap tren musiman dengan cukup baik, terutama pada periode transisi antara musim kemarau dan musim hujan. Puncak curah hujan pada bulan Januari, November, dan Desember berhasil diidentifikasi oleh model, meskipun masih terdapat perbedaan intensitas antara nilai prediksi dan aktual.

Beberapa bulan seperti April hingga Juni menunjukkan kesenjangan prediksi yang lebih besar, di mana model cenderung menghaluskan fluktuasi aktual, terutama ketika terjadi penurunan curah hujan yang cukup tajam. Hal ini mengindikasikan bahwa meskipun model cukup baik dalam mengikuti pola umum, kemampuannya dalam menangkap anomali cuaca atau perubahan drastis masih perlu ditingkatkan.

Secara keseluruhan, prediksi model *XGBoost* dalam skala bulanan menunjukkan kinerja yang stabil dan mendekati data aktual, menjadikannya cukup andal dalam konteks perencanaan berbasis tren jangka menengah seperti pertanian, pengelolaan air, atau kebencanaan