BAB IV

HASIL DAN PEMBAHASAN

4.1 Pembahasan

Bab ini menjelaskan implementasi langkah-langkah penelitian yang dirancang pada Bab 3 secara rinci, mulai dari proses pengambilan data lapangan, teknik pembersihan dan pengolahan data, hingga penerapan metode analisis statistik dan pemodelan. Setiap tahapan dirancang untuk memastikan data yang digunakan relevan, valid, dan mampu menghasilkan model prediksi yang akurat sesuai dengan tujuan penelitian. Setiap langkah akan diuraikan secara sistematis, mencakup pengambilan data dari sumber terpercaya, pengolahan data dengan teknik preprocessing yang sesuai, validasi menggunakan analisis statistik, serta pemodelan dan evaluasi menggunakan metrik yang tepat untuk memastikan akurasi prediksi. Hasil yang diperoleh pada setiap langkah juga dianalisis secara mendalam untuk menjawab tujuan penelitian.

Proses penelitian ini menggunakan pendekatan statistik dan metode prediksi berbasis Generalized Linear Model (GLM) dengan fitur Polynomial, Random Forest (RF), serta analisis tambahan menggunakan Vector Autoregression (VAR).

- Generalized Linear Model (GLM) dengan pendekatan polynomial dipilih untuk memodelkan hubungan non-linear antara variabel prediktor dan respons, memungkinkan fleksibilitas dalam distribusi data.
- Random Forest (RF) sebuah algoritma ensemble berbasis pohon keputusan, digunakan untuk menangkap hubungan kompleks antar variabel dan menghasilkan prediksi yang robust pada dataset yang besar.
- Vector Autoregression (VAR) digunakan untuk menganalisis hubungan dinamis antar variabel iklim dalam konteks data deret waktu, memberikan wawasan mendalam mengenai interaksi antar variabel selama periode tertentu.

Dataset yang digunakan mencakup data curah hujan serta variabel iklim seperti suhu rata-rata harian, kelembapan, dan tekanan udara di Kota Kediri. Periode pengamatan meliputi data selama 5 tahun terakhir, dengan fokus pada data harian dari bulan Januari hingga Oktober 2024. Periode ini mencakup musim hujan dan kemarau untuk menangkap perbedaan pola curah hujan yang signifikan antara kedua musim tersebut. Analisis ini bertujuan memberikan pemahaman yang lebih mendalam tentang fluktuasi curah hujan, yang berpotensi memengaruhi sektor pertanian dan pengelolaan sumber daya air di wilayah tersebut.

Pengolahan data dilakukan menggunakan teknik khusus, seperti pengisian data hilang dengan metode interpolasi linier untuk menjaga kontinuitas data. Pola data dianalisis menggunakan visualisasi grafik dari Matplotlib dan Seaborn untuk mengidentifikasi tren musiman. Selanjutnya, model prediksi curah hujan diterapkan dengan dukungan algoritma Generalized Linear Model (GLM), Random Forest (RF) dan Vector Autoregression (VAR). Kombinasi metode ini diharapkan memberikan hasil prediksi yang signifikan serta mendukung tujuan penelitian, yaitu meningkatkan akurasi prediksi curah hujan di Kota Kediri.

4.1.1 Pengambilan Data

Data yang digunakan dalam penelitian ini diambil dari situs resmi BMKG. Beberapa langkah penting terkait pengambilan dan pengolahan data meliputi :

- Data diambil dari bulan Januari hingga Oktober 2024 untuk mencakup musim hujan dan kemarau di Kota Kediri.
- Frekuensi data harian dipilih untuk mendapatkan granularitas yang memadai dalam analisis curah hujan.
- Data awalnya berada dalam format Microsoft Excel dan kemudian diubah ke format CSV untuk kompatibilitas dengan pustaka Python seperti Pandas.

Untuk memastikan kualitas dan konsistensi dataset, beberapa langkah dilakukan, yaitu:

1. Menghapus judul tabel yang tidak konsisten dan menggantinya dengan format standar untuk memastikan keseragaman.

2. Membersihkan data, termasuk menangani nilai yang hilang atau outlier menggunakan metode interpolasi.
3. Memvisualisasikan distribusi awal data untuk memahami pola dan anomali.

1. Import Library

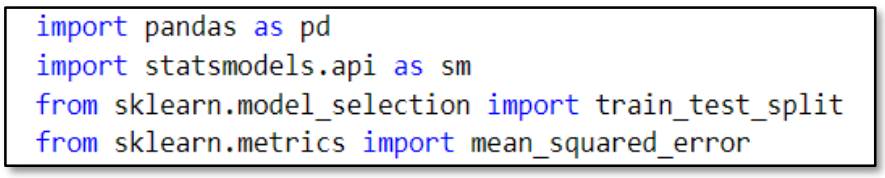
Berikut adalah pustaka yang digunakan dalam penelitian ini :

1) Metode Generalized Linear Model (GLM)

Berikut langkah-langkah menggunakan GLM untuk analisis data:

- Data dibaca dari file CSV menggunakan pustaka pandas. Pustaka ini memungkinkan manipulasi data dalam bentuk DataFrame, yang mempermudah proses analisis.
- Dataset dibagi menjadi data latih dan data uji menggunakan train_test_split dari Scikit-learn. Pembagian ini dilakukan untuk memastikan model dapat dievaluasi dengan data yang belum pernah dilihat selama pelatihan.
- Model GLM dibangun menggunakan pustaka statsmodels. Model ini menggunakan distribusi Gaussian untuk memprediksi variabel target berdasarkan variabel independen.
- Hasil prediksi model dievaluasi menggunakan metrik mean_squared_error untuk mengukur seberapa baik model memprediksi data uji.

Kode Implementasi GLM :



Gambar 4. 1 Proses Import Library untuk Implementasi Generalized Linear Model (GLM)

Gambar tersebut menunjukkan proses mengimpor pustaka-pustaka yang diperlukan untuk pemodelan GLM. Berikut langkah-langkah yang ditunjukkan:

1. Mengimpor Library Pandas menggunakan alias `pd`. Library ini digunakan untuk membaca file CSV dan memanipulasi data dalam format DataFrame.
2. Mengimpor Library Statsmodels diimpor dengan alias `sm`. Library ini berfungsi untuk membangun model GLM dengan distribusi Gaussian.
3. Mengimpor Fungsi train_test_split ini membagi dataset menjadi data latih (80%) dan data uji (20%) secara acak.
4. Mengimpor Metrik Evaluasi mean_squared_error digunakan untuk mengevaluasi seberapa besar kesalahan antara nilai prediksi model dengan nilai aktual.

2) Metode Random Forest (RF)

Berikut langkah-langkah menggunakan RF untuk analisis data:

- Dataset dibaca dari file CSV menggunakan pustaka pandas. Data ini akan diolah dalam format DataFrame untuk mempermudah manipulasi.

- Dataset dibagi menjadi data latih dan data uji menggunakan fungsi train_test_split. Langkah ini memastikan model dievaluasi dengan data yang belum pernah digunakan saat pelatihan.
- Model RF dibuat menggunakan RandomForestRegressor. Model ini bekerja dengan membuat beberapa pohon keputusan dari subset data secara acak dan menggabungkan hasil prediksi dengan rata-rata.
- Prediksi model dibandingkan dengan data aktual menggunakan metrik mean_squared_error. Nilai MSE yang lebih kecil menunjukkan model yang lebih baik.

Kode Implementasi RF :

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error
```

Gambar 4. 2 Proses Import Library untuk Implementasi Random Forest (RF)

Gambar tersebut menunjukkan proses mengimpor pustaka-pustaka yang diperlukan untuk pemodelan RF. Berikut langkah-langkah yang ditunjukkan:

1. Mengimpor Library Pandas pada baris pertama, library pandas diimpor menggunakan alias `pd`. Library ini digunakan untuk membaca file CSV dan memanipulasi data dalam format DataFrame.

2. Mengimpor Fungsi train_test_split pada baris kedua, fungsi `train_test_split` dari pustaka Scikit-learn diimpor. Fungsi ini membagi dataset menjadi data latih (80%) dan data uji (20%) secara acak.
3. Mengimpor Library RandomForestRegressor pada baris ketiga, pustaka Scikit-learn diimpor untuk membangun model Random Forest Regressor menggunakan fungsi `RandomForestRegressor`.
4. Mengimpor Metrik Evaluasi mean_squared_error pada baris keempat, metrik `mean_squared_error` dari pustaka Scikit-learn diimpor. Metrik ini digunakan untuk mengevaluasi performa model dengan menghitung kesalahan antara nilai prediksi model dan nilai aktual.

3) Metode Vector Autoregresion (VAR)

Berikut langkah-langkah menggunakan VAR untuk analisis data:

- Dataset dibaca dari file CSV menggunakan pustaka pandas dan diolah dalam format DataFrame.
- Dataset dapat dibagi menjadi data latih dan data uji untuk mengevaluasi performa model.
- Model VAR dibuat menggunakan pustaka statsmodels. Model ini mempelajari hubungan antar variabel dan menghasilkan prediksi berdasarkan nilai sebelumnya.
- Prediksi dibandingkan dengan data aktual menggunakan metrik seperti mean_squared_error untuk mengukur error.

Kode Implementasi VAR :

```
import pandas as pd
from sklearn.model_selection import train_test_split
from statsmodels.tsa.api import VAR
from sklearn.metrics import mean_squared_error
```

Gambar 4. 3 Proses Import Library untuk Implementasi Vector Autoregresion (VAR)

Gambar tersebut menunjukkan langkah awal dalam membangun model prediksi menggunakan VAR. Berikut langkah-langkah yang ditunjukkan:

1. Mengimpor Library Pandas menggunakan alias `pd`. Library ini digunakan untuk membaca file CSV dan memanipulasi data dalam format DataFrame.
2. Mengimpor Fungsi train_test_split ini membagi dataset menjadi data latih (80%) dan data uji (20%) secara acak.
3. Mengimpor pustaka `VAR` (Vector Autoregression) dari statsmodels diimpor. Pustaka ini digunakan untuk menganalisis hubungan antar variabel dalam data time series.
4. Mengimpor Metrik Evaluasi mean_squared_error digunakan untuk mengevaluasi kesalahan antara nilai prediksi dan nilai aktual.

2. Membaca Dataset dari File CSV

Langkah Implementasi :

- file_path: Menyimpan path menuju file CSV yang berisi dataset. Anda dapat mengganti path ini sesuai lokasi file Anda.

- pd.read_csv(file_path): Fungsi dari pustaka pandas untuk membaca file CSV ke dalam format DataFrame sehingga data dapat diolah.

Kode Implementasi :

```
# Load dataset dari file CSV
file_path = (r'C:\Users\DIRPENDIK\Documents\datasetskripsi.csv')  # Ganti dengan path file CSV Anda
data = pd.read_csv(file_path)
```

Gambar 4. 4 Proses Membaca Dataset menggunakan Pandas dari File CSV

Gambar di atas menunjukkan proses membaca dataset dari file CSV menggunakan library pandas. Langkah-langkah implementasi yang ditunjukkan adalah sebagai berikut :

1. Mendefinisikan Path File Dataset variabel `file_path` menyimpan path atau lokasi file dataset dalam format CSV. Lokasi ini merujuk pada file `datasetskripsi.csv` yang terletak di folder `Documents` pengguna. Anda dapat menyesuaikan path sesuai dengan lokasi file dataset Anda.
2. Membaca Dataset ke dalam DataFrame Pandas perintah `pd.read_csv(file_path)` digunakan untuk membaca file CSV yang telah ditentukan sebelumnya. Hasil pembacaan ini disimpan ke dalam variabel `data`, yang berbentuk DataFrame. DataFrame adalah struktur data dua dimensi yang disediakan oleh pandas, sangat cocok untuk analisis dan manipulasi data.

3. Menampilkan Nama Kolom dalam Dataset

Langkah Implementasi:

- data.columns.tolist(): Fungsi ini digunakan untuk mengambil semua nama kolom dari dataset dan mengubahnya menjadi sebuah daftar (list). Hal ini mempermudah verifikasi struktur dataset dan memastikan bahwa nama-nama kolom sesuai dengan yang dibutuhkan untuk analisis lebih lanjut.

Kode Implementasi :

```
# Tampilkan nama-nama kolom
print("Nama-nama kolom dalam dataset:")
print(data.columns.tolist())
```

Gambar 4. 5 Proses Menampilkan Nama Kolom pada Dataset yang Dibaca

Gambar di atas menunjukkan proses untuk menampilkan nama-nama kolom yang terdapat dalam dataset. Langkah-langkah implementasi yang ditunjukkan adalah sebagai berikut:

1. Menampilkan Informasi perintah `print("Nama-nama kolom dalam dataset:")` digunakan untuk mencetak teks statis di layar. Teks ini berfungsi sebagai keterangan awal untuk memberikan informasi kepada pengguna tentang apa yang akan ditampilkan.
2. Mengakses Nama Kolom Dataset perintah `data.columns.tolist()` digunakan untuk mengambil daftar nama kolom dari DataFrame `data`. Metode `.columns` mengembalikan objek berupa indeks kolom,

sedangkan metode `.tolist()` mengonversi indeks tersebut menjadi daftar (list) Python.

3. Mencetak Daftar Nama Kolom hasil dari metode `tolist()` kemudian dicetak di layar menggunakan perintah `print()` untuk memberikan informasi kepada pengguna mengenai struktur data yang sedang mereka kerjakan.

4. Memastikan Data Ter-load dengan Benar

Langkah Implementasi:

- data.head(): Fungsi ini digunakan untuk menampilkan 5 baris pertama dari dataset secara default. Hal ini berguna untuk memeriksa apakah data ter-load dengan benar dan apakah formatnya sesuai dengan yang diinginkan, seperti apakah ada kolom yang hilang atau data kosong.

Kode Implementasi :

```
# Cek beberapa baris awal dari dataset untuk memastikan data ter-load dengan benar
print("\nBeberapa baris awal dari dataset:")
print(data.head())
```

Gambar 4. 6 Proses Verifikasi untuk Memastikan Data Ter-load dengan Benar

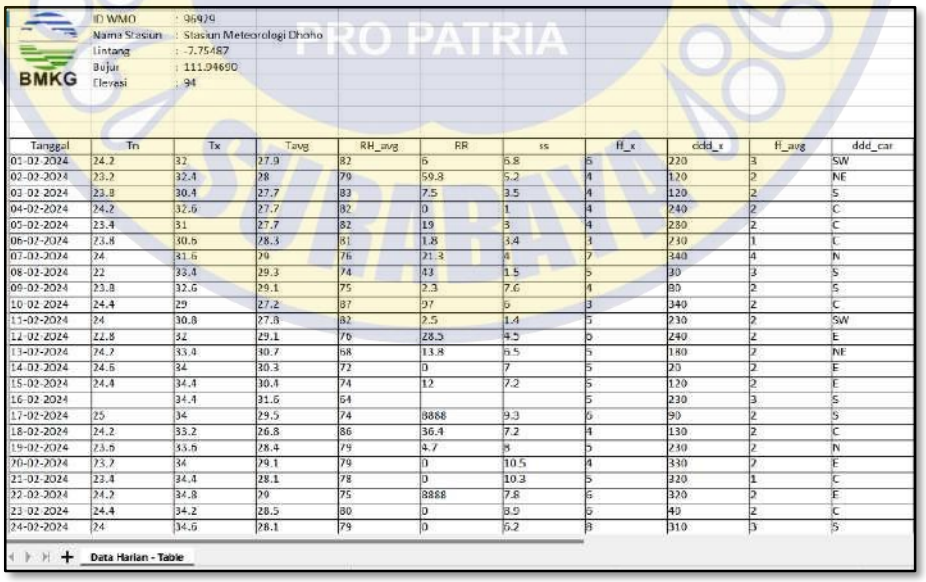
Gambar di atas menunjukkan proses untuk memverifikasi data dengan menampilkan beberapa baris awal dari dataset. Langkah-langkah implementasi yang ditunjukkan adalah sebagai berikut:

1. Menampilkan Informasi perintah print("\nBeberapa baris awal dari dataset:") digunakan untuk mencetak teks statis di layar. Teks ini berfungsi sebagai keterangan awal untuk memberikan informasi kepada pengguna tentang data yang akan ditampilkan.

2. Menampilkan Beberapa Baris Data perintah data.head() digunakan untuk menampilkan 5 baris pertama dari dataset secara default. Fungsi ini sangat berguna untuk memeriksa apakah data ter-load dengan benar, apakah ada kolom yang hilang, atau format data sesuai harapan.

3. Mencetak Data Awal hasil dari metode head() kemudian dicetak di layar menggunakan perintah print() untuk memberikan informasi awal kepada pengguna mengenai struktur data yang sedang mereka kerjakan.

4.1.2 Pengolahan Data



BMKG

ID WMO	: 96979
Nama Stasiun	: Stasiun Meteorologi Dhoho
Lintang	: -7.75487
Bujur	: 111.94690
Elevasi	: 94

Tanggal	Tn	Tx	Tavg	RH_avg	RR	ss	ff_x	ddd_x	ff_avg	ddd_car
01-02-2024	24.2	32	27.9	82	6	6.8	6	220	3	SW
02-02-2024	23.2	32.4	28	79	59.8	5.2	4	120	2	NE
03-02-2024	23.8	30.4	27.7	83	7.5	3.5	4	120	2	S
04-02-2024	24.2	32.6	27.7	82	0	1	4	240	2	C
05-02-2024	23.4	31	27.7	82	19	3	4	280	2	C
06-02-2024	23.8	30.6	28.3	81	1.8	3.4	4	230	1	C
07-02-2024	24	31.6	29	76	21.3	4	7	340	4	N
08-02-2024	22	33.4	29.3	74	43	1.5	5	30	3	S
09-02-2024	23.8	32.6	29.1	75	2.3	7.6	4	80	2	S
10-02-2024	24.4	29	27.2	87	97	6	3	340	2	C
11-02-2024	24	30.8	27.8	82	2.5	1.4	5	230	2	SW
12-02-2024	22.8	32	29.1	76	28.5	4.5	6	240	2	E
13-02-2024	24.2	33.4	30.7	68	13.8	6.5	5	180	2	NE
14-02-2024	24.6	34	30.3	72	0	7	5	20	2	E
15-02-2024	24.4	34.4	30.4	74	12	7.2	5	120	2	E
16-02-2024		34.4	31.6	64			5	230	3	S
17-02-2024	25	34	29.5	74	8888	9.3	6	90	2	S
18-02-2024	24.2	33.2	26.8	86	36.4	7.2	4	130	2	C
19-02-2024	23.6	33.6	28.4	79	4.7	8	5	230	2	N
20-02-2024	23.2	34	29.1	79	0	10.5	4	330	2	E
21-02-2024	23.4	34.4	28.1	78	0	10.3	5	320	1	C
22-02-2024	24.2	34.8	29	75	8888	7.8	6	320	2	E
23-02-2024	24.4	34.2	28.5	80	0	8.9	6	40	2	C
24-02-2024	24	34.6	28.1	79	0	6.2	8	310	3	S

Data Harian - Table

Gambar 4. 7 Dataset Penelitian

Gambar di atas menunjukkan contoh dataset yang digunakan dalam penelitian ini. Dataset ini mencakup data cuaca harian dari Stasiun Meteorologi Dholo yang mencakup berbagai parameter penting terkait kondisi cuaca. Data tersebut diperoleh dari sumber terpercaya, yaitu BMKG (Badan Meteorologi, Klimatologi, dan Geofisika), yang mencatat berbagai variabel iklim untuk analisis lebih lanjut.

Dataset ini mencakup 10 parameter utama terkait kondisi cuaca, yaitu:

1. Tn : Suhu terendah yang tercatat dalam periode tertentu (°C).
2. Tx : Suhu tertinggi dalam periode tertentu (°C).
3. Tavg : Rata-rata suhu harian, dihitung dari nilai Tn dan Tx (°C).
4. RH_avg : Kelembapan udara rata-rata dalam persentase (%).
5. RR : Jumlah curah hujan (mm), merupakan target yang ingin dituju.
6. Ss : Lama paparan sinar matahari setiap hari (jam).
7. ff_x : Kecepatan angin tertinggi yang tercatat (m/s).
8. ddd_x : Arah angin saat kecepatan tertinggi terjadi (°).
9. ff_avg : Kecepatan rata-rata angin selama periode tertentu (m/s).
10. ddd_car : Arah angin yang paling sering muncul selama periode tertentu (°).

Pemilihan fitur dalam dataset ini didasarkan pada relevansi variabel terhadap prediksi curah hujan. Tavg (suhu rata-rata harian) dipilih karena suhu memiliki peran penting dalam proses penguapan dan pembentukan awan, yang merupakan tahap awal dalam siklus hujan. RH_avg (kelembapan udara rata-rata) juga dipilih karena kelembapan yang tinggi berpotensi meningkatkan curah hujan,

menjadikannya faktor utama dalam analisis cuaca. Ss (durasi penyinaran matahari) turut dipertimbangkan karena radiasi matahari memengaruhi pemanasan atmosfer dan pola penguapan yang berpengaruh langsung pada curah hujan.

Selain itu, ff_avg (kecepatan rata-rata angin) dipilih karena angin berperan dalam distribusi uap air dan pembentukan pola awan. Variabel target, RR (curah hujan), dipilih sebagai output karena tujuan utama analisis ini adalah untuk memprediksi curah hujan. Dengan memilih variabel-variabel ini, model prediksi dapat lebih efektif dalam menangkap pola cuaca yang signifikan, sekaligus mengurangi pengaruh dari variabel lain yang kurang relevan.

Tidak semua variabel digunakan dalam analisis ini, dengan fokus pada parameter yang paling berhubungan dengan prediksi curah hujan. Berikut adalah beberapa parameter yang terpilih:

1. Fitur yang Digunakan (X)

   Variabel yang dipilih untuk model prediksi adalah :

   - Tavg : Suhu rata-rata harian, karena berkorelasi dengan penguapan dan pembentukan awan.
   - RH_avg : Kelembapan udara, yang memengaruhi potensi hujan.
   - Ss : Durasi penyinaran matahari, karena berperan dalam pemanasan atmosfer.
   - ff_avg : Rata-rata kecepatan angin, yang memengaruhi distribusi awan dan pola cuaca.

2. Target Prediksi (Y) :

    - RR : Curah hujan, sebagai nilai yang ingin diprediksi berdasarkan fitur di atas

Pendekatan yang digunakan dalam penelitian ini melibatkan beberapa metode pemodelan data dengan melakukan prediksi curah hujan. Setiap metode memiliki langkah-langkah persiapan data yang spesifik, seperti menentukan variabel independen (fitur) dan variabel dependen (target). Berikut adalah penjelasan masing-masing metode:

1. **Metode Generalized Linear Model (GLM)**

Langkah Implementasi:

Pada model Generalized Linear Model (GLM), kita menggunakan sejumlah variabel independen untuk memprediksi variabel dependen, dalam hal ini adalah curah hujan. Berikut adalah penjelasan terkait variabel yang digunakan dalam model:

- features: Daftar variabel independen yang dianggap memengaruhi curah hujan. Dalam hal ini, features berisi variabel seperti temperature, humidity, sunshine, dan windspeed.
- X: Data input yang berisi nilai-nilai dari variabel independen, yaitu features yang telah dipilih sebelumnya.
- y: Variabel target yang akan diprediksi oleh model, yaitu curah hujan (rainfall).

Kode Implementasi:

```
# Menyiapkan data untuk GLM
features = ['temperature', 'humidity', 'sunshine', 'windspeed']
X = data[features]
y = data['rainfall']
```

Gambar 4. 8 Proses Pengolahan Data Menggunakan Generalized Linear Model (GLM)

Gambar di atas menunjukkan langkah-langkah dalam menyiapkan data untuk digunakan dalam model GLM. Berikut penjelasan setiap bagian kode:

- Variabel `features` didefinisikan sebagai daftar kolom atau variabel yang dianggap memengaruhi curah hujan. Dalam contoh ini, *temperature*, *humidity*, *sunshine*, dan *windspeed* dipilih sebagai variabel independen.
- Perintah `X = data[features]` digunakan untuk mengambil nilai-nilai dari variabel independen yang terdapat dalam dataset. Nilai-nilai ini akan digunakan sebagai data input untuk model.
- Variabel `y` didefinisikan sebagai kolom `rainfall` dalam dataset. Data ini merupakan variabel target yang akan diprediksi oleh model GLM.

2. Metode Random Forest (RF)

Langkah Implementasi:

Model Random Forest (RF) digunakan untuk memprediksi curah hujan dengan menggunakan beberapa variabel independen sebagai input. Berikut adalah penjelasan variabel yang digunakan:

- Features: Daftar variabel independen yang dianggap memengaruhi curah hujan, seperti temperature, humidity, sunshine, dan windspeed.
- X: Data input yang berisi nilai-nilai dari variabel independen.
- Y: Variabel target yang akan diprediksi, yaitu curah hujan (rainfall).

Kode Implementasi :

```
# Menyiapkan data untuk Random Forest
features = ['temperature', 'humidity', 'sunshine', 'windspeed']
X = data[features]
y = data['rainfall']
```

Gambar 4. 9 Proses Pengolahan Data Menggunakan Random Forest (RF)

Gambar di atas menunjukkan langkah-langkah dalam menyiapkan data untuk digunakan dalam model RF. Berikut penjelasan setiap bagian kode:

- Variabel `features` didefinisikan sebagai daftar kolom atau variabel yang dianggap memengaruhi curah hujan. Dalam contoh ini, *temperature*, *humidity*, *sunshine*, dan *windspeed* dipilih sebagai variabel independen.
- Perintah `X = data[features]` digunakan untuk mengambil nilai-nilai dari variabel independen yang terdapat dalam dataset. Nilai-nilai ini akan digunakan sebagai data input untuk model.

- Variabel `y` didefinisikan sebagai kolom `rainfall` dalam dataset. Data ini merupakan variabel target yang akan diprediksi oleh model RF.

3. Metode Vector Autoregression (VAR)

Langkah Implementasi :

Model Vector Autoregression (VAR) digunakan untuk menganalisis hubungan dinamis antar variabel dalam data time series. Dalam hal ini, kita mempertimbangkan beberapa variabel sebagai input untuk menganalisis pengaruh satu sama lain, dengan memperhitungkan efek tunda (lag). Berikut penjelasan variabel yang digunakan:

- Data: Memilih subset kolom yang relevan dari dataset, seperti temperature, humidity, sunshine, windspeed, dan rainfall. Semua variabel ini akan dianalisis bersama sebagai bagian dari model VAR.
- Model VAR: Menggunakan seluruh variabel (fitur dan target) untuk menganalisis hubungan dinamis antarvariabel, termasuk pengaruh dari waktu tunda.

Kode Implementasi :

```
# Menyiapkan data untuk VAR, fokus pada fitur dan label
# Dalam hal ini, kita menggunakan semua fitur dan 'rainfall' sebagai variabel target
data = data[['temperature', 'humidity', 'sunshine', 'windspeed', 'rainfall']]
```

Gambar 4. 10 Proses Pengolahan Data Menggunakan Vector Autoregression (VAR)

Gambar di atas menunjukkan langkah-langkah dalam menyiapkan data untuk digunakan dalam model VAR. Berikut penjelasan setiap bagian kode:

- data_var memilih subset kolom yang relevan untuk digunakan dalam analisis VAR, yang mencakup variabel-variabel seperti temperature, humidity, sunshine, windspeed, dan rainfall.

4.1.3 Validasi Data

Proses validasi dilakukan untuk memastikan:

- Kesesuaian format data.
- Tidak ada anomali seperti nilai ekstrim yang tidak wajar.
- Dataset siap digunakan untuk tahap pemodelan.

Pada tahap validasi data, ditemukan bahwa data asli mengandung banyak nilai nol atau angka yang kurang memadai. Oleh karena itu, untuk memperbaiki distribusi data dan mengurangi efek nilai ekstrim, dilakukan transformasi logaritma. Proses ini bertujuan untuk mengubah data yang memiliki nilai ekstrim atau nol menjadi lebih terdistribusi secara normal.

Transformasi Logaritma

Transformasi logaritma dilakukan menggunakan rumus:

$$X' = \log(X + 1) \quad (3)$$

Keterangan :

- X adalah nilai asli dari data.
- X′adalah hasil setelah transformasi logaritma.

Kebalikan Transformasi Logaritma

Setelah melakukan transformasi logaritma, hasil prediksi yang diperoleh dari model juga berada dalam bentuk yang sudah tertransformasi. Untuk mengembalikan hasil prediksi ke bentuk asli, kita perlu melakukan kebalikan dari transformasi logaritma.

Kebalikan dari transformasi logaritma dilakukan dengan rumus :

$$x_{pred} = 10^{ypred} - 1 \quad (4)$$

Keterangan :

- Ypred adalah nilai yang diprediksi setelah transformasi logaritma.
- Xpred adalah nilai yang diprediksi dalam bentuk asli.

Dengan demikian, proses kebalikan logaritma memungkinkan kita untuk mengembalikan hasil prediksi ke skala yang sesuai dengan data asli.

Contoh Perhitungan :

1. Transformasi Logaritma: Misalkan x=10, maka :

$$y = \log(10 + 1) = \log(11) \approx 1.0414$$

Hasil transformasi logaritma dari nilai x adalah y ≈ 1.0414

2. Prediksi Menggunakan Data yang Ditansformasi: Misalkan hasil prediksi untuk data yang sudah ditransformasi adalah ypred =2, maka:
3. Kebalikan Transformasi Logaritma untuk mengembalikan hasil prediksi ke nilai asli, kita gunakan rumus kebalikan:

$$\text{xpred} = 10\text{ypred} - 1 = 102 - 1 = 100 - 1 = 99$$

Dengan demikian, hasil prediksi yang sesungguhnya adalah xpred = 99. Selanjutnya, pembagian dataset menjadi data latih dan data uji dilakukan untuk memastikan bahwa model dapat dilatih dan dievaluasi dengan data yang berbeda, sehingga menghasilkan performa yang lebih objektif. Berikut adalah penjelasan mengenai metode pembagian data pada masing-masing model:

1. Metode Generalized Linear Model (GLM) dan Random Forest (RF)

```
# Membagi dataset menjadi training dan testing (80% training, 20% testing)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Gambar 4. 11 Proses Validasi Data Menggunakan Generalized Linear Model (GLM) dan Random Forest (RF)

Gambar di atas menunjukkan langkah-langkah dalam proses validasi untuk digunakan dalam model GLM dan RF. Berikut penjelasan setiap bagian kode:

1. train_test_split(X, y, test_size=0.2, random_state=42):

- X: Merupakan variabel input atau fitur (independent variables) yang digunakan untuk melatih model.
- y: Merupakan variabel target atau label (dependent variable) yang ingin diprediksi oleh model.
- test_size=0.2: Menentukan proporsi data yang digunakan untuk data uji (testing set). Dalam hal ini, 20% dari total data akan digunakan untuk pengujian, dan sisanya 80% akan digunakan untuk pelatihan.
- random_state=42: Menentukan nilai acak untuk pembagian data agar hasilnya dapat direproduksi. Angka 42 adalah nilai acak yang digunakan, tetapi bisa diganti dengan nilai lain sesuai kebutuhan. Penggunaan nilai acak ini memastikan bahwa pembagian data yang dilakukan selalu konsisten setiap kali kode dijalankan.

2. X_train, X_test, y_train, y_test:

- X_train: Data latih untuk variabel input (fitur).
- X_test: Data uji untuk variabel input (fitur).
- y_train: Data latih untuk variabel target.
- y_test: Data uji untuk variabel target.

Dengan menggunakan train_test_split, dataset dibagi menjadi data latih dan data uji, yang akan digunakan untuk melatih dan mengevaluasi model secara terpisah.

2. Metode Vector Autoregression (VAR)

```
# Membagi dataset menjadi training dan testing (80% training, 20% testing)
train_size = int(len(data) * 0.8)
train, test = data.iloc[:train_size], data.iloc[train_size:]
```

Gambar 4. 12 Proses Validasi Data Menggunakan Vector Autoregression (VAR)

Gambar di atas menunjukkan langkah-langkah dalam proses validasi untuk digunakan dalam model VAR. Berikut penjelasan setiap bagian kode:

1. train_size = int(len(data) * 0.8):
   - len(data): Menghitung jumlah baris dalam dataset data.
   - 0.8: Persentase data yang digunakan untuk training (80%).
   - int(): Mengonversi hasil perhitungan menjadi bilangan bulat.
   - Variabel train_size akan berisi jumlah baris yang digunakan untuk data latih (training set), yaitu 80% dari jumlah total data.
2. train, test = data.iloc[:train_size], data.iloc[train_size:]:
   - data.iloc[:train_size]: Memilih data dari baris pertama hingga baris yang sesuai dengan train_size (80% pertama dari dataset) untuk training set.
   - data.iloc[train_size:]: Memilih data dari baris setelah train_size hingga akhir dataset (20% terakhir) untuk testing set.

- train: Menyimpan data latih (80% dari total data).
- test: Menyimpan data uji (20% dari total data).

Kode ini membagi dataset data menjadi dua bagian: 80% untuk training dan 20% untuk testing, yang umum digunakan dalam pembelajaran mesin untuk melatih dan menguji model.

4.1.4 Pemodelan Data (Modelling)

Pada bagian ini, proses pemodelan dilakukan menggunakan tiga metode, yaitu Generalized Linear Model (GLM), Random Forest, dan Vector Autoregression (VAR). Setiap metode memiliki pendekatan yang berbeda untuk menangkap pola dalam dataset dan diuji untuk menentukan performa terbaik dalam memprediksi curah hujan.

1. **Metode Generalized Linear Model (GLM)**

Metode Generalized Linear Model (GLM) digunakan untuk menangkap hubungan linear antara fitur dan target dengan asumsi distribusi normal. GLM adalah model yang sangat berguna untuk data yang memiliki hubungan linear atau hampir linear antara fitur dan target. Dalam implementasinya, GLM dilatih menggunakan pustaka statsmodels dengan fungsi link Gaussian, yang mengasumsikan bahwa kesalahan (error) mengikuti distribusi normal.

Selain itu, konstanta ditambahkan ke dalam fitur untuk memasukkan intercept dalam model. Tanpa intercept, model diasumsikan melalui titik asal (0,0), yang sering kali tidak mencerminkan hubungan yang sebenarnya dalam data. GLM memberikan hasil prediksi yang cukup baik, dan dengan menghitung Mean Squared

Error (MSE), kita dapat mengevaluasi apakah model ini dapat menangkap pola dalam data dengan baik.

```
# Menambahkan intercept ke dalam model
X_train = sm.add_constant(X_train)
X_test = sm.add_constant(X_test)

# Membuat model GLM dengan data training
model = sm.GLM(y_train, X_train, family=sm.families.Gaussian())
results = model.fit()
```

Gambar 4. 13 Proses Pemodelan dengan Generalized Linear Model (GLM)

Gambar di atas menunjukkan langkah-langkah dalam proses pemodelan data untuk digunakan dalam model GLM. Berikut penjelasan setiap bagian kode :

1. Menambahkan Intercept ke Model :

- X_train = sm.add_constant(X_train) dan X_test = sm.add_constant(X_test): Fungsi add_constant() dari pustaka statsmodels digunakan untuk menambahkan kolom konstanta (intercept) ke dalam data fitur.
  - X_train: Data fitur untuk pelatihan.
  - X_test: Data fitur untuk pengujian.

Intercept diperlukan dalam model regresi untuk memastikan model dapat menangkap pergeseran nilai target secara vertikal (y-intercept). Tanpa intercept, model diasumsikan melalui titik asal (0,0), yang sering kali tidak akurat dan dapat menghasilkan prediksi yang bias.

2. Membuat Model GLM:

- sm.GLM(y_train, X_train, family=sm.families.Gaussian()):

  Fungsi GLM() digunakan untuk membuat model Generalized Linear Model (GLM).

  - y_train: Variabel target dari data pelatihan.
  - X_train: Data fitur yang telah ditambahkan konstanta (intercept).
  - family=sm.families.Gaussian(): Menentukan distribusi error. Dalam hal ini, distribusi Gaussian (normal) digunakan, karena regresi linier adalah kasus khusus dari GLM dengan distribusi normal. Distribusi Gaussian mengasumsikan bahwa kesalahan model memiliki distribusi normal.

3. Melatih Model:

- results = model.fit():

  Fungsi fit() digunakan untuk melatih model GLM menggunakan data latih (X_train dan y_train).

Objek results menyimpan informasi penting seperti parameter model, nilai statistik, dan hasil prediksi yang dapat digunakan untuk evaluasi lebih lanjut. Dengan objek results, kita bisa memeriksa koefisien regresi, nilai p, dan informasi lainnya yang penting untuk menilai kinerja model.

Kode ini bertujuan untuk membangun model GLM yang dapat digunakan untuk menganalisis hubungan antara variabel independen (X_train) dan variabel target (y_train). Setelah model dilatih, kita dapat menggunakannya untuk membuat

prediksi pada data uji (X_test) dan mengevaluasi akurasi model dengan menggunakan metrik seperti MSE.

2. Metode Random Forest (RF)

Random Forest adalah algoritma ensemble yang menggunakan beberapa pohon keputusan (decision trees) untuk membuat prediksi. Setiap pohon dalam forest memberikan hasil yang berbeda, dan hasil akhirnya adalah kombinasi dari semua pohon yang ada. Metode ini unggul dalam menangani hubungan non-linear dan interaksi antar variabel. Random Forest memiliki kemampuan untuk meningkatkan akurasi model dengan mengurangi risiko overfitting, karena menggunakan beberapa model pohon keputusan.

```
# Membuat model Random Forest
model = RandomForestRegressor(n_estimators=100, random_state=42)
model.fit(X_train, y_train)
```

Gambar 4. 14 Proses Pemodelan dengan Random Forest (RF)

Gambar di atas menunjukkan langkah-langkah dalam proses pemodelan data untuk digunakan dalam model RF. Berikut penjelasan setiap bagian kode :

1. RandomForestRegressor(n_estimators=100, random_state=42):
    - RandomForestRegressor: Model regresi berbasis ensemble yang menggunakan beberapa pohon keputusan (decision trees) untuk membuat prediksi.

- n_estimators=100: Menentukan jumlah pohon keputusan yang akan dibuat dalam ensemble. Semakin banyak pohon, semakin stabil dan akurat prediksi, tetapi juga membutuhkan lebih banyak waktu komputasi.
- random_state=42: Menetapkan nilai random state agar hasil pembagian data dan pelatihan model dapat direproduksi.

2. model.fit(X_train, y_train):
    - X_train: Data latih untuk fitur (variabel independen).
    - y_train: Data latih untuk target (variabel dependen).
    - Metode fit akan melatih model dengan data latih, sehingga model mempelajari hubungan antara fitur dan target.

3. Metode Vector Autoregression (VAR)

VAR digunakan untuk menganalisis hubungan temporal antar variabel dalam dataset. Metode ini cocok untuk data deret waktu dengan hubungan yang saling memengaruhi antar variabel. Model dilatih menggunakan pustaka statsmodels, dengan parameter lag ditentukan berdasarkan validasi performa. VAR tidak memberikan hasil yang lebih baik dibandingkan baseline, kemungkinan karena variabel dalam dataset tidak memiliki hubungan temporal yang signifikan.

```
# Membuat model VAR
model = VAR(train)

# Melatih model
results = model.fit(maxlags=15, ic='aic')
```

Gambar 4. 15 Proses Pemodelan dengan Vector Autoregression (VAR)

Gambar di atas menunjukkan langkah-langkah dalam proses pemodelan data untuk digunakan dalam model VAR. Berikut penjelasan setiap bagian kode :

1. model = VAR(train):
    - VAR(train): Membuat model VAR dengan menggunakan data latih (train).
    - train: Data input untuk model VAR. Data ini harus berupa data multivariat (berisi beberapa variabel) yang akan digunakan untuk menganalisis hubungan antarvariabel secara dinamis.
2. results = model.fit(maxlags=15, ic='aic'):
    - fit(): Melatih model VAR dengan data yang diberikan.
    - maxlags=15: Menentukan jumlah maksimum lag (tunda waktu) yang dipertimbangkan dalam model. Model akan mencari hubungan hingga 15 periode sebelumnya.
    - ic='aic': Menggunakan kriteria informasi Akaike (AIC) untuk memilih jumlah lag terbaik. AIC membantu dalam memilih model yang paling sesuai dengan data tanpa overfitting.

Pembagian Data (Presentasi Split)

1. Data Pelatihan (Training Set): Bagian ini terdiri dari 80% data dan digunakan untuk melatih model. Melalui pelatihan ini, model akan belajar mengenali pola dan hubungan antar variabel.
2. Data Pengujian (Testing Set): Sisanya, yaitu 20% data, digunakan untuk menguji seberapa baik model dapat memprediksi curah hujan pada data yang belum pernah dilihat sebelumnya.

Pembagian ini dilakukan dengan menggunakan fungsi train_test_split dari pustaka sklearn, yang secara acak memisahkan data menjadi dua bagian dengan proporsi yang telah ditentukan. Pembagian yang tepat adalah kunci untuk memastikan bahwa model tidak hanya mengingat data yang dilatih (overfitting), tetapi juga mampu menggeneralisasi hasil ke data yang baru. Berikut ini penjelasan proses pelatihan dan pengujian model :

a. Proses Pelatihan Model (Training)

Dengan data pelatihan yang telah disiapkan (80%), model mulai dilatih menggunakan masing-masing metode (GLM, RF, dan VAR). Selama pelatihan, model belajar membangun hubungan matematis yang menghubungkan variabel input dengan output yang ingin diprediksi (curah hujan).

b. Proses Pengujian Model (Testing)

Setelah model dilatih, data pengujian (20%) digunakan untuk menguji kemampuan model dalam memprediksi curah hujan pada data yang belum pernah dilihat. Evaluasi dilakukan dengan menghitung Mean Squared Error (MSE)**,** yang mengukur perbedaan antara nilai yang diprediksi oleh model dengan nilai aktual dari data pengujian. MSE yang lebih kecil menunjukkan prediksi yang lebih akurat. Dengan menggunakan MSE sebagai metrik evaluasi, performa masing-masing model dapat diukur secara objektif. Model yang memiliki MSE paling rendah akan dianggap lebih baik dalam memprediksi curah hujan pada data yang tidak terlihat sebelumnya.

4.1.5 Evaluasi

Evaluasi dilakukan untuk menilai kualitas dan kinerja model yang dibangun menggunakan metrik utama, yaitu:

1. MSE (Mean Squared Error) Mengukur rata-rata kuadrat dari kesalahan prediksi. Metrik ini memberikan penalti yang lebih besar pada kesalahan besar.

Langkah evaluasi mencakup :

- Menghitung metrik MSE untuk model serta membandingkannya dengan baseline (nilai rata-rata target).
- Menginterpretasikan hasil, di mana nilai MSE yang lebih kecil menunjukkan performa model yang lebih baik.

Untuk memahami implementasi evaluasi ini, berikut adalah penjelasan kode yang digunakan pada beberapa metode pemodelan :

1. **Metode Generalized Linear Model (GLM) dan Random Forest (RF)**

```
# Menghitung dan menampilkan Mean Squared Error sebagai evaluasi kinerja model
mse = mean_squared_error(y_test, predictions)
print(f"\nMean Squared Error: {mse}")
```

Gambar 4. 16 Proses Evaluasi Hasil Prediksi Menggunakan Generalized Linear Model (GLM) dan Random Forest (RF)

Gambar di atas menunjukkan langkah-langkah dalam proses evaluasi hasil prediksi untuk digunakan dalam model GLM dan RF. Berikut penjelasan setiap bagian kode :

- mean_squared_error(): Fungsi ini digunakan untuk menghitung nilai MSE dengan membandingkan y_test (nilai target sebenarnya) dan predictions (nilai yang diprediksi oleh model) pada data uji.
- y_test: Merupakan data target asli (sesuai dengan data pengujian).
- predictions: Merupakan hasil prediksi yang dihasilkan oleh model yang sedang diuji.
- print(f"\nMean Squared Error: {mse}"): Menampilkan nilai MSE ke layar untuk analisis lebih lanjut.

2. **Metode Vector Autoregression (VAR)**

```
# Menghitung dan menampilkan Mean Squared Error sebagai evaluasi kinerja model
mse = mean_squared_error(comparison_df['Actual'], comparison_df['Predicted'])
print(f"\nMean Squared Error: {mse}")
```

Gambar 4. 17 Proses Evaluasi Hasil Prediksi Menggunakan Vector Autoregression (VAR)

Gambar di atas menunjukkan langkah-langkah dalam proses evaluasi hasil prediksi untuk digunakan dalam model VAR. Berikut penjelasan setiap bagian kode :

- comparison_df['Actual']: Kolom dalam DataFrame comparison_df yang berisi nilai target sebenarnya.
- comparison_df['Predicted']: Kolom dalam comparison_df yang berisi nilai yang diprediksi oleh model VAR.
- mean_squared_error(): Menghitung nilai MSE dengan membandingkan nilai aktual dan prediksi dari comparison_df.
- print(f"\nMean Squared Error: {mse}"): Menampilkan hasil perhitungan MSE ke layar.

4.1.6 Model Prediksi

Proses prediksi dilakukan dengan langkah berikut :

1. Dataset dibagi menjadi 80% data pelatihan dan 20% data pengujian untuk memastikan model memiliki data yang cukup untuk belajar dan diuji.
2. Data pelatihan digunakan untuk membangun model berdasarkan hubungan antara fitur (X) dan target (Y).

3. Data pengujian digunakan untuk mengukur kemampuan model dalam melakukan prediksi pada data baru yang belum pernah dilihat.

Prediksi akhir dilakukan menggunakan model yang telah dilatih, dengan hasil prediksi dikonversi kembali dari transformasi logaritma ke skala aslinya. Langkah ini penting untuk memastikan hasil prediksi dapat diinterpretasikan dalam konteks data asli. Setelah konversi, hasil prediksi dibandingkan dengan nilai aktual untuk mengukur akurasi model, menggunakan metrik seperti Mean Squared Error (MSE). Berikut implementasi prediksi dan evaluasi menggunakan beberapa metode:

1. Metode Generalized Linear Model (GLM)

```
# Menampilkan nilai aktual dan prediksi
comparison_df = pd.DataFrame({'Actual': y_test, 'Predicted': predictions})
print("\nPerbandingan nilai aktual dan prediksi:")
print(comparison_df.reset_index(drop=True))

# Simpan Prediksi
predictions_df = pd.DataFrame({
    'Actual': y_test,
    'Predicted': predictions
})
predictions_df.to_csv('GLM_HasilPrediksi_Results.csv', index=False)
print("\nPredicted results saved to 'GLM_HasilPrediksi_Results.csv'.")
```

Gambar 4. 18 Proses Penerapan Metode Generalized Linear Model (GLM) dalam Model Prediksi

Gambar di atas menunjukkan langkah-langkah dalam proses model prediksi untuk digunakan dalam model GLM. Berikut penjelasan setiap bagian kode :

1. Menampilkan Nilai Aktual dan Prediksi

- comparison_df = pd.DataFrame({'Actual': y_test, 'Predicted': predictions}): Membuat DataFrame baru yang berisi dua kolom, yaitu 'Actual' (nilai aktual dari data uji) dan 'Predicted' (hasil prediksi).
- print("\nPerbandingan nilai aktual dan prediksi:"): Menampilkan pesan sebelum mencetak DataFrame.
- print(comparison_df.reset_index(drop=True)): Mencetak DataFrame yang berisi perbandingan nilai aktual dan prediksi. reset_index(drop=True) digunakan untuk menghilangkan index lama.

2. Menyimpan Prediksi ke File CSV
   - predictions_df = pd.DataFrame({'Actual': y_test, 'Predicted': predictions}): Membuat DataFrame yang sama dengan sebelumnya, namun di sini digunakan untuk menyimpan data ke file CSV.
   - predictions_df.to_csv('GLM_HasilPrediksi_Results.csv', index=False): Menyimpan DataFrame predictions_df ke dalam file CSV bernama GLM_HasilPrediksi_Results.csv. index=False berarti index tidak akan disertakan dalam file.
   - print("\nPredicted results saved to 'GLM_HasilPrediksi_Results.csv'."): Menampilkan pesan bahwa hasil prediksi telah disimpan.

2. Metode Random Forest (RF)

```
# Menampilkan nilai aktual dan prediksi
comparison_df = pd.DataFrame({'Actual': y_test, 'Predicted': predictions})
print("\nPerbandingan nilai aktual dan prediksi:")
print(comparison_df.reset_index(drop=True))

# Simpan Prediksi
predictions_df = pd.DataFrame({
    'Actual': y_test,
    'Predicted': predictions
})
predictions_df.to_csv('RF_HasilPrediksi_Results.csv', index=False)
print("\nPredicted results saved to 'RF_HasilPrediksi_Results.csv'.")
```

Gambar 4. 19 Proses Penerapan Metode Random Forest (RF) dalam Model Prediksi

Gambar di atas menunjukkan langkah-langkah dalam proses model prediksi untuk digunakan dalam model RF. Berikut penjelasan setiap bagian kode :

1. Menampilkan Nilai Aktual dan Prediksi:
    - comparison_df = pd.DataFrame({'Actual': y_test, 'Predicted': predictions}): Membuat DataFrame yang berisi dua kolom, 'Actual' untuk nilai aktual (dari data uji) dan 'Predicted' untuk hasil prediksi.
    - print("\nPerbandingan nilai aktual dan prediksi:"): Menampilkan pesan yang menunjukkan bahwa yang akan ditampilkan adalah perbandingan antara nilai aktual dan prediksi.
    - print(comparison_df.reset_index(drop=True)): Menampilkan DataFrame yang berisi nilai aktual dan prediksi. reset_index(drop=True) digunakan untuk menghilangkan index lama agar hasilnya lebih rapi.
2. Menyimpan Prediksi ke File CSV:

- predictions_df = pd.DataFrame({'Actual': y_test, 'Predicted': predictions}): Membuat DataFrame lagi untuk menyimpan hasil prediksi ke dalam file CSV. Data yang disimpan adalah kolom 'Actual' dan 'Predicted'.
- predictions_df.to_csv('RF_HasilPrediksi_Results.csv', index=False): Menyimpan DataFrame predictions_df ke dalam file CSV dengan nama RF_HasilPrediksi_Results.csv. index=False berarti index tidak akan disertakan dalam file CSV.
- print("\nPredicted results saved to 'RF_HasilPrediksi_Results.csv'."): Menampilkan pesan bahwa hasil prediksi telah berhasil disimpan ke dalam file CSV.

3. Metode Vector Autoregression (VAR)

```
# Menampilkan nilai aktual dan prediksi
print("\nPerbandingan nilai aktual dan prediksi:")
print(comparison_df.reset_index(drop=True))

# Simpan hasil prediksi ke file CSV
comparison_df.to_csv('VAR_HasilPrediksi_Results.csv', index=False)
print("\nPredicted results saved to 'VAR_HasilPrediksi_Results.csv'.")
```

Gambar 4. 20 Proses Penerapan Metode Vector Autoregression (VAR) dalam Model Prediksi

Gambar di atas menunjukkan langkah-langkah dalam proses model prediksi untuk digunakan dalam model VAR. Berikut penjelasan setiap bagian kode :

1. Menampilkan Perbandingan Nilai Aktual dan Prediksi:

- comparison_df.reset_index(drop=True): Mengatur ulang indeks pada DataFrame agar dimulai dari 0 dan menghilangkan indeks sebelumnya.
- print(comparison_df): Menampilkan tabel yang berisi nilai aktual dan hasil prediksi untuk analisis visual.

2. Menyimpan Hasil Prediksi ke CSV:
    - to_csv(): Metode yang digunakan untuk menyimpan DataFrame ke file CSV.
    - index=False: Menghilangkan kolom indeks agar file CSV lebih rapi.
    - VAR_HasilPrediksi_Results.csv: Nama file yang digunakan untuk menyimpan hasil prediksi.

4.1.7 Hasil Penelitian

Pada bagian ini, dilakukan analisis dan interpretasi hasil penelitian yang diperoleh dari evaluasi performa model prediksi. Fokus utama adalah pada nilai Mean Squared Error (MSE) dan persentase selisih nilai aktual terhadap nilai prediksi untuk menilai seberapa baik model bekerja.

1. Performa Model Berdasarkan MSE

Tabel berikut menyajikan nilai MSE dari tiga metode yang digunakan :

Tabel 4. 1 Hasil Analisis MSE

Metode	Hasil MSE	Keterangan

Generalized Linear Model (GLM)	0.22	Model ini menunjukkan kinerja yang cukup baik, namun ada ruang untuk perbaikan.
Random Forest (RF)	0.21	Menunjukkan hasil yang lebih baik daripada GLM, dengan tingkat kesalahan yang sedikit lebih rendah.
Vector Autoregression (VAR)	0.12	Metode ini memberikan hasil terbaik dengan MSE terendah, menunjukkan kemampuan prediksi yang lebih akurat.

Analisis dan Interpretasi

1. Generalized Linear Model (GLM)

   GLM menghasilkan nilai MSE sebesar 0.22, menunjukkan bahwa model ini mampu memberikan prediksi yang cukup baik. Namun, nilai kesalahan yang relatif lebih tinggi dibandingkan metode lain menunjukkan bahwa masih ada ruang untuk perbaikan.

2. Random Forest (RF)

   Model RF memiliki nilai MSE sebesar 0.21, sedikit lebih rendah dibandingkan GLM. Hal ini menunjukkan bahwa RF memiliki kemampuan yang lebih baik dalam menangani data, terutama data yang lebih kompleks, meskipun keunggulannya tidak terlalu signifikan dibanding GLM.

3. Vector Autoregression (VAR)

VAR memberikan hasil terbaik dengan nilai MSE sebesar 0.12. Kesalahan prediksi yang rendah ini menunjukkan bahwa model VAR lebih akurat dalam memprediksi data dan mampu menangkap pola atau hubungan antar variabel dalam dataset dengan lebih baik.

Dari ketiga metode, Vector Autoregression (VAR) adalah pilihan terbaik untuk meminimalkan kesalahan prediksi karena memiliki nilai MSE terendah. Namun, pemilihan model juga dapat mempertimbangkan faktor lain seperti interpretabilitas atau kebutuhan komputasi.

2. Hasil Prediksi

Tabel berikut menyajikan perbandingan antara nilai aktual dan nilai prediksi yang dihasilkan oleh masing-masing model, disertai dengan persentase selisihnya :

Tabel 4. 2 Hasil Prediksi Model GLM, RF, dan VAR Berdasarkan Selisih Nilai Aktual

Metode	Nilai Aktual	Nilai Prediksi	Selisih
Generalized Linear Model (GLM)	1.38	0.92	33.03%
	1.08	0.67	37.26%.
	0.81	0.46	43.21%
Random Forest (RF)	0.78	1.06	37.09%
	1.05	0.64	39.05%
	0.81	0.44	45.34%
	0.36	0.25	29.53%

Vector Autoregression (VAR)	0.36	0.25	30.56%
	1.16	0.70	39.35%

Analisis dan Interpretasi

1. Generalized Linear Model (GLM)

   GLM menunjukkan performa yang cukup baik, tetapi memiliki selisih prediksi terbesar, yaitu 43.21%. Hal ini menunjukkan bahwa model ini kurang akurat pada beberapa titik data tertentu.

2. Random Forest (RF)

   RF menghasilkan prediksi yang lebih baik dibanding GLM pada beberapa data, tetapi cenderung memberikan prediksi yang lebih tinggi daripada nilai aktual. Selisih terbesar terjadi pada data keenam dengan nilai 45.34%, yang mengindikasikan bahwa model ini lebih rentan terhadap overfitting pada titik data tertentu.

3. Vector Autoregression (VAR)

   VAR memberikan hasil prediksi yang paling konsisten dibandingkan model lainnya. Selisih terkecil ditemukan pada data pertama dan kedua, sekitar 30%, sedangkan selisih terbesar terjadi pada data ketiga dengan nilai 39.35%. Hal ini menunjukkan bahwa meskipun VAR lebih stabil, masih terdapat ruang untuk perbaikan dalam meningkatkan akurasi prediksinya

Secara keseluruhan, VAR menunjukkan hasil yang lebih konsisten dibandingkan GLM dan RF, dengan selisih prediksi yang lebih kecil secara umum. Namun, semua model masih memiliki kekurangan dalam meminimalkan selisih prediksi pada beberapa titik data tertentu. Oleh karena itu, VAR direkomendasikan jika konsistensi menjadi prioritas utama dalam pemodelan prediksi curah hujan ini.