LAPORAN PROJECT AKHIR

ANALISA BIG DATA

KELAS A

Penerapan Teknologi Big Data Dalam Pengembangan Komoditas Buah Tomat

Zarkasyi Fahriza[1], Bintang Widya Narendra[2], Muhamad Farhan Ismail Dewanata[3]

[1] **Teknik Informatika, Universitas Trunojoyo Madura (220411100001@student.trunojoyo.ac.id)**
[2] **Teknik Informatika, Universitas Trunojoyo Madura (220411100037@student.trunojoyo.ac.id)**
[3] **Teknik Informatika, Universitas Trunojoyo Madura (220411100068@student.trunojoyo.ac.id)**

Abstract

Penelitian ini bertujuan untuk menganalisis penerapan teknologi Big Data dalam pengelolaan komoditas tomat guna meningkatkan efisiensi produksi, distribusi, dan stabilitas harga. Berbagai metode analisis data diterapkan, termasuk regresi linier untuk peramalan harga, Support Vector Machine (SVM) untuk klasifikasi tingkat kematangan tomat, dan algoritma K-Means untuk pengelompokan wilayah berdasarkan pola harga. Data yang digunakan meliputi dataset gambar dan data harga komoditas dari sumber terpercaya. Hasil penelitian menunjukkan bahwa model regresi linier berhasil memprediksi harga tomat dengan tingkat akurasi yang memadai, metode SVM mencapai akurasi klasifikasi hingga 82,83% untuk tiga kategori kematangan tomat (belum matang, matang, dan rusak), serta algoritma K-Means efektif dalam membentuk empat klaster wilayah pasar berdasarkan pola harga, yang memberikan wawasan penting bagi pengambilan keputusan terkait distribusi dan produksi. Temuan ini memperkuat potensi teknologi Big Data dalam mendukung pertanian cerdas dan berkelanjutan di Indonesia, khususnya pada komoditas pangan strategis seperti tomat. Simpulan dari penelitian ini adalah bahwa pengintegrasian teknologi Big Data mampu menghadirkan solusi inovatif untuk mengatasi tantangan produksi dan distribusi komoditas tomat, sekaligus memberikan landasan yang kuat untuk pengembangan kebijakan berbasis data.

Keywords: *Big Data, Tomat, Produksi*

1. Introduction

Dalam beberapa tahun terakhir, perkembangan teknologi Big Data telah mengalami lonjakan signifikan dan berperan penting dalam transformasi berbagai sektor, termasuk sektor pertanian. Penerapan Big Data dalam pertanian dikenal dengan istilah Agriculture 4.0, yang berfokus pada pengumpulan, pengolahan, dan analisis data dalam skala besar untuk meningkatkan efisiensi dan produktivitas. Salah satu komoditas yang sangat potensial untuk dioptimalkan dengan teknologi ini adalah buah tomat, yang menjadi komoditas pangan strategis dan banyak dikonsumsi di seluruh dunia, termasuk di Indonesia.[1]

Sebagai salah satu komoditas hortikultura, tomat memegang peranan penting dalam rantai pasokan pangan. Namun, komoditas ini menghadapi berbagai tantangan yang kompleks, seperti fluktuasi harga yang ekstrem, ketidakefisienan dalam produksi, dan masalah distribusi yang tidak merata. Fluktuasi harga tomat sering kali terjadi akibat ketidakseimbangan antara penawaran dan permintaan, kurangnya informasi pasar yang tepat waktu, serta keterbatasan akses terhadap data produksi yang akurat. Selain itu, proses produksi juga menghadapi ketidaksempurnaan dalam hal kualitas hasil panen dan ketepatan waktu distribusi, yang pada akhirnya memengaruhi stabilitas harga di pasaran.[2]

Dalam konteks ini, teknologi Big Data menawarkan solusi potensial untuk mengatasi tantangan tersebut dengan mengumpulkan data dalam jumlah besar dari berbagai sumber, seperti data cuaca, data harga pasar, data distribusi, serta data kualitas hasil panen. Melalui analisis Big Data, data terkait produksi dan harga tomat dari berbagai wilayah dapat diolah untuk menghasilkan wawasan yang lebih akurat dan terkini. Teknologi ini dapat digunakan dalam berbagai metode seperti peramalan harga (forecasting), klasifikasi tingkat kematangan tomat menggunakan data gambar, serta pengelompokan wilayah (clustering) berdasarkan harga komoditas. [3]

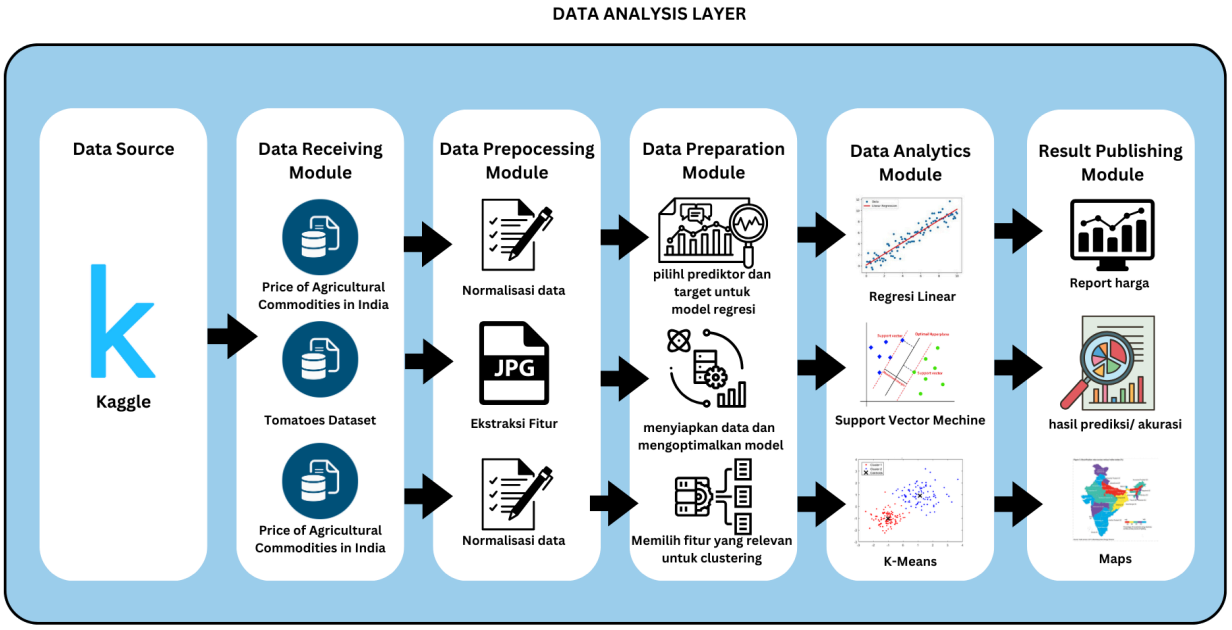
Penelitian terdahulu tentang penerapan metode regresi linier dalam peramalan (forecasting) telah banyak digunakan di berbagai industri, terutama untuk membantu pengambilan keputusan dalam manajemen inventaris dan produksi. Salah satu penelitian yang relevan adalah peramalan kebutuhan bahan baku pada industri pangan, seperti tahu, di mana prediksi yang akurat atas jumlah bahan baku menjadi krusial untuk menjaga efisiensi produksi.[4]

Dalam klasifikasi tingkat kematangan buah dengan metode Support Vector Machine (SVM) telah menunjukkan keefektifan algoritma ini dalam mendeteksi objek berdasarkan fitur tertentu, seperti warna, tekstur, atau bentuk.[5] menurut adzan et al(2023) Salah satu studi yang relevan adalah klasifikasi tingkat kematangan buah tomat menggunakan model warna berbasis ekstraksi fitur dari gambar RGB, HSV, YCbCr, dan CIELab. Penelitian ini menggunakan SVM sebagai algoritma utama untuk memproses dan mengklasifikasikan tingkat kematangan tomat berdasarkan data warna. Studi ini menunjukkan hasil akurasi sebesar 82,83%, menandakan bahwa SVM efektif dalam mengidentifikasi tingkat kematangan dengan menggunakan model warna

Sebuah studi di Indonesia yang dilakukan oleh Eka Prasetyaningrum dan Puji Susanti. (2023) menunjukkan bahwa penggunaan teknologi Big Data untuk pemetaan hasil produksi Hasilnya, algoritma K-Means memberikan performa yang lebih baik[6] Untuk menjawab tantangan yang dihadapi dalam pengelolaan komoditas tomat, proyek ini mengusulkan penerapan beberapa metode analisis data yang relevan.

Melalui penerapan teknologi Big Data ini, diharapkan bahwa para petani, pedagang, dan pemangku kebijakan akan dapat membuat keputusan yang lebih tepat dan berbasis data, sehingga tantangan yang selama ini dihadapi dalam pengelolaan komoditas tomat dapat diatasi dengan lebih efektif. Penelitian ini juga akan memberikan kontribusi penting dalam mewujudkan pertanian yang cerdas dan berkelanjutan di Indonesia, dengan memanfaatkan teknologi canggih untuk mengoptimalkan komoditas pangan yang krusial bagi ketahanan pangan nasional.

2. Arsitektur Sistem



Berikut adalah penjelasan singkat tentang arsitektur sistem penyelesaian proyek:

1. **Data Source**: Data diambil dari Kaggle, termasuk dataset harga komoditas di India dan gambar tomat untuk analisis.
2. **Data Receiving Module**: Mengimpor dataset untuk diproses lebih lanjut.
3. **Data Preprocessing Module**: Melakukan normalisasi data numerik dan ekstraksi fitur dari gambar (misalnya, warna dan tekstur).
4. **Data Preparation Module**:
    - Menyiapkan prediktor dan target untuk model regresi.
    - Memilih fitur relevan untuk analisis clustering.

7.

- Mengoptimalkan data dan model.

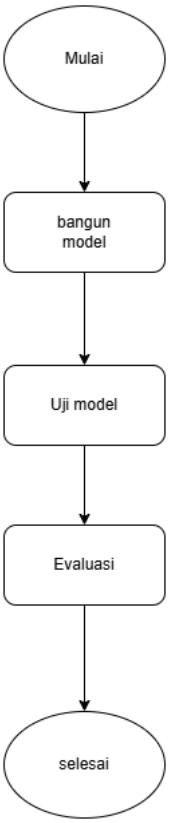
5. **Data Analytics Module**: Menerapkan model analitik seperti:
   - **Regresi Linear** untuk prediksi harga.
   - **SVM** untuk klasifikasi (contohnya, tingkat kematangan tomat).
   - **K-Means** untuk clustering data.
6. **Result Publishing Module**: Menyajikan hasil analisis dalam bentuk laporan harga, visualisasi geografis (maps), dan metrik evaluasi akurasi.

Sistem ini memastikan data diolah secara terstruktur dari sumber hingga hasil akhir yang dapat digunakan untuk pengambilan keputusan.

2.1Forecasting

Dalam penelitian ini, Regresi Linier dan SARIMA digunakan sebagai metode forecasting untuk memprediksi harga modal tomat. Regresi Linier adalah salah satu algoritma statistik yang memodelkan hubungan antara variabel independen (harga minimum dan harga maksimum) dengan variabel dependen (harga modal tomat) melalui persamaan linier. Persamaan umum Regresi Linier dapat dituliskan sebagai:

2.1.1 Algoritma Model



Rumus Normalisasi:

$$\chi scaled = \frac{\chi - \mu}{\sigma}$$

Di mana:

- X adalah nilai asli,
- μ adalah nilai rata-rata,
- σ adalah standar deviasi.[8]

Namun, ketika data memiliki pola musiman atau tren yang signifikan, model SARIMA digunakan untuk meningkatkan akurasi prediksi. SARIMA (Seasonal Autoregressive Integrated Moving Average) merupakan model yang memperluas ARIMA dengan komponen musiman. Rumus umum SARIMA adalah:

$$SARIMA(p, d, q\ X\ P, D, Q)m$$

Di mana:

- p,d,,q adalah parameter untuk komponen non-musiman,
- P,D,Q adalah parameter untuk komponen musiman,
- m adalah periode musiman.

2.1.2 Pelatihan Model

- **Model**: Sistem forecasting menggunakan **Regresi Linier**, model yang cocok untuk memprediksi nilai berkelanjutan berdasarkan hubungan linier antar variabel.

Rumus Regresi Linier:

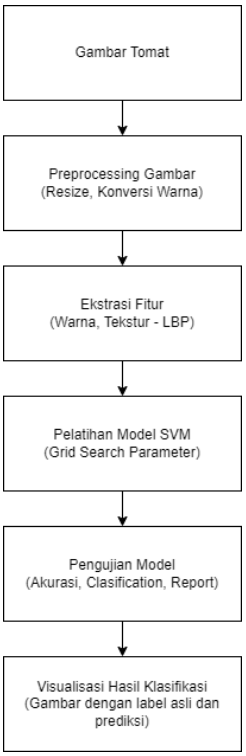
$$\hat{y} = \beta 0 + \beta 1\chi 1 + \beta 2\chi 2 + \epsilon$$

Di mana:

- $\hat{y}$ = adalah nilai prediksi harga modal,
- β0 = adalah intercept (nilai tetap),
- $\beta 1\ dan\ \beta 2$ = adalah koefisien regresi untuk harga minimum dan harga maksimum,
- $\chi 1\ dan\ \chi 2$ = adalah harga minimum dan maksimum,
- ϵ = adalah error atau noise.[9]

2.2 Klasifikasi

Arsitektur sistem klasifikasi tomat ini mencakup langkah-langkah mulai dari pengambilan data (gambar), preprocessing, ekstraksi fitur, pelatihan model klasifikasi, hingga pengujian dan visualisasi hasil klasifikasi. Berikut penjelasan detail mengenai arsitektur sistem ini[10]:



2.2.1 Data Acquisition (Akuisisi Data)

- **Input:** Gambar tomat dari berbagai tingkat kematangan (Belum Matang, Matang, Rusak).
- **Sumber Data:** Dataset gambar diambil dari Google Drive yang telah diunggah oleh pengguna. Gambar-gambar tersebut disimpan dalam folder tertentu di Google Drive[11].

2.2.2 Image Preprocessing (Pra-pemrosesan Gambar)

- **Tujuan**: Menyiapkan gambar agar dapat digunakan untuk ekstraksi fitur dan pelatihan model[12].
- **Langkah-langkah**:
  - **Resize (ubah ukuran)**: Gambar diubah ukurannya menjadi dimensi standar (128x128 piksel) agar konsisten dan memudahkan pengolahan.
  - **Konversi Warna**: Gambar dikonversi menjadi format yang lebih sesuai untuk ekstraksi fitur, seperti HSV (Hue, Saturation, Value) untuk analisis warna

2.2.3 Feature Extraction (Ekstraksi Fitur)

Tujuan: Mengambil informasi penting dari gambar untuk digunakan sebagai input bagi model klasifikasi. Fitur yang diambil terdiri dari:

- **Fitur Warna**: Rata-rata nilai Hue, Saturation, dan Value (HSV) dari gambar[10].
- **Fitur Tekstur**: Menggunakan **Local Binary Patterns (LBP)** untuk menganalisis pola tekstur pada gambar[13].

Rumus Ekstraksi Fitur Warna (HSV): Rata-rata dari masing-masing komponen HSV dihitung dari rumus berikut.

$$H_{mean} = \frac{1}{N}\sum_{i=1}^{n} H(i)$$

$$S_{mean} = \frac{1}{N}\sum_{i=1}^{n} S(i)$$

$$V_{mean} = \frac{1}{N}\sum_{i=1}^{n} V(i)$$

Dimana:

- $H(i), S(i), V(i)$ adalah nilai Hue, Saturation, dan Value dari setiap pixel pada gambar.
- N adalah jumlah pixel pada gambar.

Rumus Ekstraksi Fitur Tekstur (LBP): Histogram dari nilai LBP dihitung untuk mendapatkan pola tekstur dari gambar.

$$LBP\left(x_c, y_c\right) = \sum_{p=0}^{p-1} s\left(i_p - i_c\right) \times 2^p$$

Dimana:

- x_c, y_c adalah koordinat pixel pusat.
- i_p adalah intensitas pixel tetangga.
- i_c adalah intensitas pixel pusat.
- $S(x) = 1\ jika\ x \geq 0\ dan\ s(x) = 0\ jika\ x < 0$

2.2.4 Model Training (Pelatihan Model) Menggunakan SVM

Berikut adalah penjelasan **tahapan SVM (Support Vector Machine)** dalam sistem klasifikasi tomat yang mencakup tiga kelas kematangan: **Belum Matang, Matang dan Rusak**.

1. **Input Data**
   - SVM menerima data berupa fitur yang telah diekstrak dari gambar tomat, yaitu:
     - Fitur Warna: Nilai rata-rata Hue, Saturation, dan Value (HSV).
     - Fitur Tekstur: Pola tekstur yang diperoleh melalui Local Binary Patterns (LBP).

2. **Pemisahan Kelas**
   - Data fitur yang diterima dibagi ke dalam tiga kelas berdasarkan tingkat kematangan tomat:
     - Belum Matang
     - Matang
     - rusak
   - SVM mencari hyperplane yang **memisahkan kelas-kelas tersebut** dengan margin maksimal

3. **Penentuan Hyperplane**
   - SVM menggunakan **kernel function** untuk memproyeksikan data ke dimensi yang lebih tinggi (jika diperlukan) agar data dapat dipisahkan dengan lebih baik.
   - Hyperplane didefinisikan sebagai:

$$f(x) = \sum_{i=1}^{n} a_i y_i K(x_i, x) + b$$

Dimana:

- x: Input fitur.
- y_i: Label kelas asli.
- a_i: Koefisien yang diperoleh dari pelatihan.
- K(x_i, x): Kernel function untuk memproyeksikan data.
- b: Bias.

4. **Pengoptimalan Parameter**

- Untuk memastikan hyperplane yang ditemukan memberikan performa terbaik, dilakukan **Grid Search** untuk mencari kombinasi optimal dari parameter:
  - **C (penalty parameter)**: Mengontrol trade-off antara margin besar dan kesalahan klasifikasi pada data pelatihan.
  - **γ (gamma)**: Mengatur pengaruh setiap titik data terhadap bentuk hyperplane (berkaitan dengan kernel RBF atau lainnya).

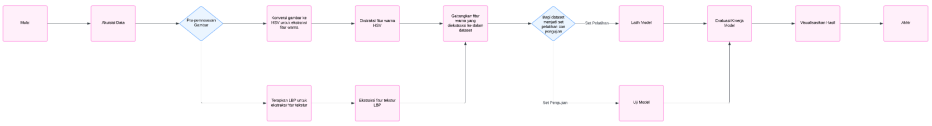
5. **Fungsi Keputusan**

- Setelah menemukan hyperplane terbaik, fungsi keputusan SVM digunakan untuk memprediksi kelas berdasarkan posisi data terhadap hyperplane:
  - Jika f(x) > 0, maka data termasuk kelas tertentu.
  - Jika f(x) < 0, maka data termasuk kelas lain.

6. **Evaluasi Hasil**

- Model diuji pada data uji untuk memeriksa performanya dalam memprediksi tingkat kematangan tomat.
- Metrik evaluasi meliputi:
  - Akurasi: Proporsi prediksi yang benar.
  - Precision: Ketepatan model dalam memprediksi kelas tertentu.
  - Recall: Kemampuan model mendeteksi semua data dari kelas tertentu.
  - F1-Score: Harmonik antara precision dan recall.

Flowchart



2.2.5 Testing (Pengujian Model)

2.2.5.1 Tujuan Pengujian:

Untuk mengevaluasi performa model dalam memprediksi tingkat kematangan tomat pada data yang belum pernah dilihat sebelumnya (data uji). Hasil prediksi akan dibandingkan dengan label asli untuk menilai kemampuan model.

2.2.5.2 Proses Pengujian:

- Data uji (20% dari dataset) yang tidak digunakan dalam pelatihan akan dimasukkan ke dalam model yang sudah dilatih (SVM) untuk menghasilkan prediksi tingkat kematangan.
- Model SVM akan memberikan prediksi kelas untuk setiap gambar pada data uji, yaitu 'Belum Matang', 'Setengah Matang', atau 'Matang'.

2.2.5.3 Evaluasi Model:

- **Akurasi:** Mengukur seberapa sering model berhasil mengklasifikasikan gambar dengan benar.
- **Classification Report:** Menyediakan metrik evaluasi lainnya seperti Precision, Recall, dan F1-Score untuk setiap kelas.
- Precision menunjukkan proporsi prediksi benar dari semua prediksi untuk kelas tertentu.
- Recall mengukur seberapa baik model dalam menemukan semua data yang benar-benar berada dalam kelas tersebut.
- F1-Score adalah rata-rata harmonis dari precision dan recall, memberikan pandangan menyeluruh mengenai performa model pada setiap kelas.

2.2.5.4 Rumus Akurasi:

Rumus ini digunakan untuk menghitung persentase prediksi yang benar dari total datauji.

2.2.5.5 Analisis Hasil:

Model kemudian dievaluasi untuk melihat apakah hasil prediksi sejalan dengan label asli dari gambar tomat dalam data uji. Hal ini memberikan wawasan tentang ketepatan dan konsistensi model dalam memprediksi tingkat kematangan.

2.2.6 Visualization (Visualisasi Hasil Klasifikasi)

Pada tahap visualisasi, gambar tomat dari data uji akan ditampilkan bersama dengan label asli dan hasil prediksi dari model SVM. Gambar ini menunjukkan tingkat kematangan tomat yang sebenarnya serta hasil prediksi dari model. Visualisasi ini membantu pengguna memahami performa model secara intuitif, apakah model dapat mengklasifikasikan gambar dengan benar atau tidak.

2.2.7 Skenario Pengujian

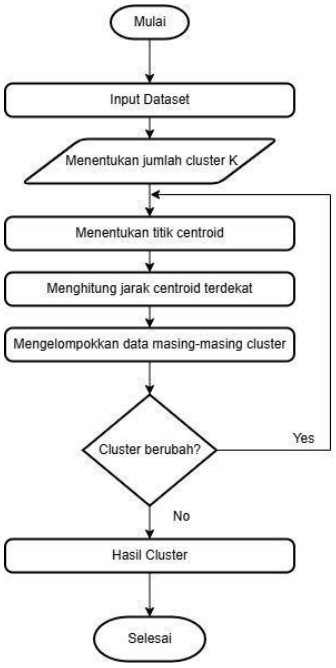
- **Data Latih dan Data Uji:** Dataset dibagi menjadi 80% untuk pelatihan dan 20% untuk pengujian.
- **Grid Search:** Dilakukan pencarian parameter optimal untuk SVM.
- **Evaluasi:** Model dievaluasi berdasarkan akurasi dan laporan klasifikasi.
- **Visualisasi:** Hasil visualisasi ditampilkan untuk menunjukkan performa model pada data uji.

2.3 Clustering

Pada penelitian ini proses clustering digunakan untuk memperoleh klaster atau kelompok dari wilayah pasar pertanian di india berdasarkan harga komoditas tomat pada setiap kota dan distrik di inida. Data yang digunakan berasal dari website kaggle https://www.kaggle.com/datasets/anshtanwar/current-daily-price-of-various-commodities-india yang merupakan platform populer untuk berbagi dan menemukan dataset yang relevan untuk penelitian dan pengembangan model mechine learning.

2.3.1 Algoritma K-Means Clustering

Pada penelitian ini, algoritma K-Means digunakan untuk mengelompokkan wilayah pasar berdasarkan pola harga yang ada, sehingga dapat memberikan wawasan yang lebih baik bagi petani, pedagang, dan pembuat kebijakan dalam pengambilan keputusan terkait produksi dan distribusi komoditas. Proses yang dilakukan memiliki beberapa tahapan yaitu sebagai berikut.



1. Melakukan pembersihan data.
2. Menginput data yang sudah *cleaning*
3. Tentukan nilai k sebagai jumlah cluster yang akan dibentuk menggunakan metode *elbow criterion* dengan rumus:

$$SSE = \sum_{k=1}^{k} \sum_{x_i = S_k} ||N_i - C_k||$$

4. Tentukan k sebagai pusat cluster (*Centroid*) awal yang dilakukan secara random. Penentuan *centroid* dilakukan secara acak dari data yang ada sebanyak k *cluster*. Untuk menentukan posisi *centroid* ke-*i* yang baru, dapat menggunakan rumus:

$$v = \frac{\sum_{i=1}^{n} X_i}{n} : i = 1, 2, 3, \ldots n$$

5. Hitung jarak masing-masing data dengan posisi setiap centroid menggunakan Euclidean Distance dengan rumus:

$$d(x, y) = ||x - y|| = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} : i = 1, 2, 3, \ldots n$$

6. Alokasikan masing-masing data ke dalam *centroid* yang paling dekat. Cara mengalokasikan data ialah dengan mengukur
7. Lakukan iterasi pada setiap data dan lakukan pembaruan posisi centroid dengan persamaan pada langkah 4
8. Lakukan perulangan dari langkah 3-5 hingga posisi semua centroid tidak mengalami perubahan.

3. Hasil dan Pembahasan

3.1. Karakteristik Big Data

a. Volume

Volume dalam konteks big data mengacu pada jumlah data yang disimpan. Pada proyek ini banyak data yang di dapat sebanyak 495 untuk klasifikasi, 2741 untuk forecasting, dan 671 untuk clustering. Sehingga untuk total data yang digunakan untuk proyek analisis big data ini sebanyak 3.907 Banyaknya data yang diperoleh nantinya akan mempengaruhi kualitas output yang dihasilkan pada setiap model. Sehingga volume data ini penting untuk memastikan bahwa model machine learning yang digunakan pada setiap pendekatan memiliki data yang cukup untuk melatih dan menguji setiap model yang akan digunakan pada masing-masing pendekatan

b. Velocity

Velocity dalam konteks ini mengacu pada kecepatan proses pengumpulan dan pemrosesan data. Pada proyek analisis big data ini dataset tidak dikumpulkan secara real-time, melainkan sebagai data statis yang diunduh dari berbagai sumber di internet dan ditambah dengan beberapa data yang diambil secara manual. Meski pengumpulan data tidak melibatkan aliran data yang konstan, kecepatan pemrosesan tetap penting saat data ini diolah menggunakan algoritma machine learning. Model harus mampu memproses input secara cepat agar dapat berfungsi secara efektif dalam aplikasi real-time atau sistem otomatis. Oleh karena itu, pemilihan algoritma dan optimasi model menjadi kunci dalam memastikan kecepatan proses yang tinggi. .

c. Veracity

Veracity berkaitan dengan kepercayaan dan keakuratan data yang diperoleh. Dalam proyek analisis big data ini, sumber data utama adalah web di Internet. Web-web tersebut adalah Kaggle. web tersebut dipilih utnuk menjadi sumber data karena web tersebut menyajikan data-data yang dapat dipertanggungjawabkan kualitasnya. Kaggle, platform ini dikenal menyediakan dataset berkualitas untuk tujuan penelitian dan pengembangan machine learning. Selain data yang diambil dari internet, data juga diperoleh dari melakukan pengambilan foto. Foto tersebut digunakan untuk menambah data untuk klasifikasi. Dengan demikian, data yang digunakan untuk proyek ini memiliki tingkat kepercayaan yang tinggi sehingga hasil yang dihasilkan nantinya juga memiliki kualitas yang sama terpercayanya dengan data yang digunakan untuk pelatihan setiap model.

d. Variety

Variety menggambarkan keragaman format data yang tersedia. Pada proyek analisis big data ini, data yang diperoleh dari setiap sumber memiliki jenis yang beragam. Jenis data yang diperoleh tersebut meliputi dataset berisi gambar tomat yang terbagi dalam tiga kategori visual utama: tomat belum matang (Unripe), tomat matang (ripe) dan tomat rusak (damaged). Kemudian data time series juga digunakan untuk proyek ini. Dataset ini mencakup data harga tomat dari **16 Juni 2013** hingga **13 Mei 2021**, memberikan rentang waktu yang cukup panjang untuk melakukan analisis time series dan forecasting yang berkualitas.. dengan masing-masing 12 data diperoleh pada setiap tahunnya. Dan yang terakhir ada data tabular yang digunakan untuk clustering. Data tabular tersebut berisi data harga komoditas buah tomat pada setiap kota di India pada tahun 2023 Keberagaman pada jenis data yang digunakan akan menambah kompleksitas dalam menghasilkan insight, namun juga memperkaya variasi yang dibutuhkan oleh setiap model untuk mendapatkan insight yang diinginkan.

3.2. Forecasting

Pada bagian ini, menjelaskan implementasi metode forecasting harga menggunakan model sarima.

1. library yang di gunakan

```
1. import pandas as pd
2. import numpy as np
3. from statsmodels.tsa.statespace.sarimax import SARIMAX
4. import matplotlib.pyplot as plt
5. from sklearn.metrics import mean_squared_error, mean_absolute_error
```

2. import data

```
# Load the dataset
df = pd.read_csv('Tomato.csv')
print("Dataset loaded successfully.")
# Convert the 'Date' column to datetime format
df['Date'] = pd.to_datetime(df['Date'])
print("Date column converted to datetime format.")
# Set the 'Date' column as the index
df.set_index('Date', inplace=True)
print("Date column set as index.")
# Use the 'Average' column for forecasting
data = df['Average']
print("Data for forecasting extracted.")

```

Dataset loaded successfully.
Date column converted to datetime format.
Date column set as index.
Data for forecasting extracted.

3. penanganan outlayer

```
# Detect outliers using IQR
Q1 = data.quantile(0.25)
Q3 = data.quantile(0.75)
IQR = Q3 - Q1

# Define a threshold to identify outliers
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# Filter out outliers
data = data[(data >= lower_bound) & (data <= upper_bound)]
print("Outliers filtered out.")
```

Outliers filtered out.

4. tahap persiapan data

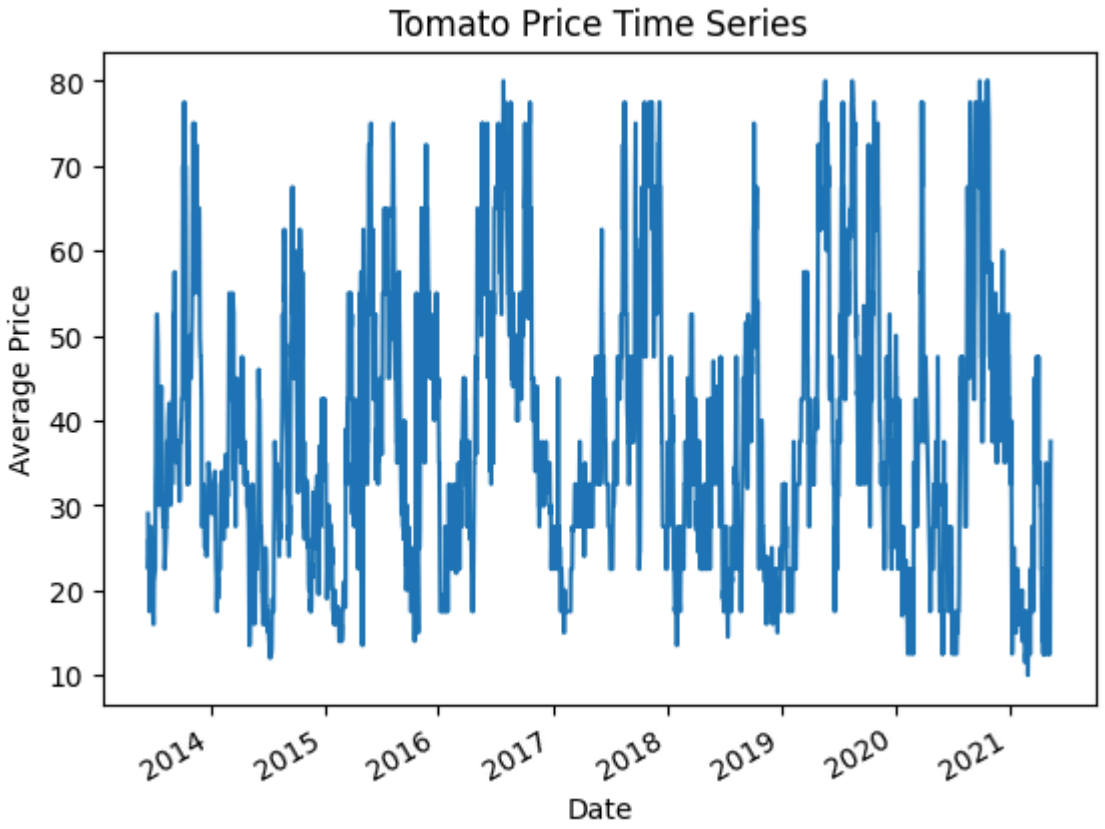
```
# 2. Prepare the data for forecasting
if 'Date' in data.columns:
    data['Date'] = pd.to_datetime(data['Date'])
    data.set_index('Date', inplace=True)
else:
    raise ValueError("The dataset must contain a 'Date' column for
time series analysis.")

# Sort by date
data = data.sort_index()

# Check for missing values and handle them (e.g., forward fill)
data = data['Average'].fillna(method='ffill')

# Split data into train and test sets
train_size = int(len(data) * 0.8)
train, test = data.iloc[:train_size], data.iloc[train_size:]
```

5. model

```
# Plot the time series data
print("Plotting time series data...")
```

7.

```
data.plot(title='Tomato Price Time Series')
plt.xlabel('Date')
plt.ylabel('Average Price')
plt.show()
```



6. split data

```
# Split data into training and testing sets
train_data = data[:-7]
test_data = data[-7:]
print("Data split into training and testing sets.")
```

Data split into training and testing sets.

7. tahap optimasi

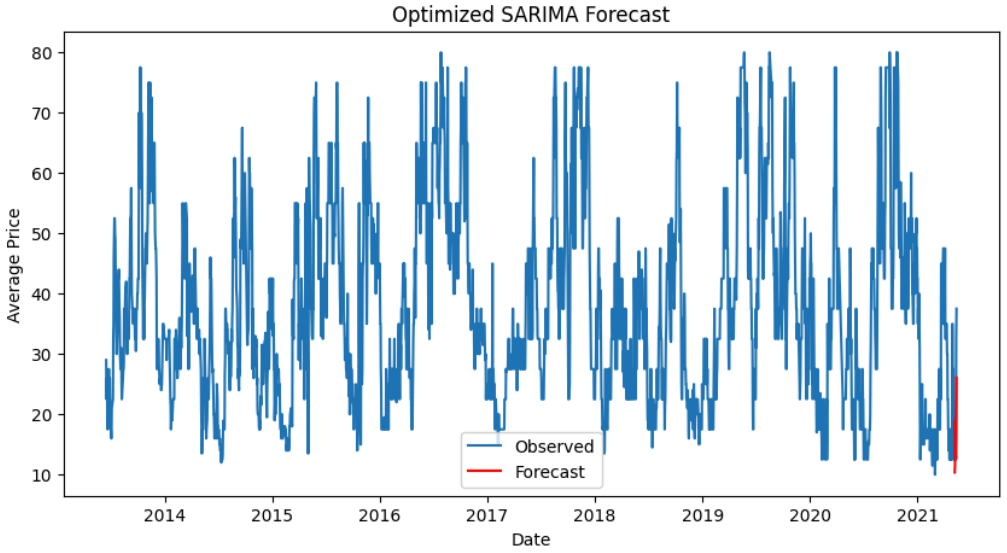
```
print("Starting parameter optimization...")
for p in range(3):
    for d in range(2):
        for q in range(3):
            for P in range(2):
                for D in range(2):
                    for Q in range(2):
                        try:
                            # Build the SARIMA model
                            model = SARIMAX(train_data, order=(p, d, q), seasonal_order=(P, D, Q, 7))
                            model_fit = model.fit(disp=False)

                            # Forecasting
                            forecast = model_fit.forecast(steps=len(test_data))

```

```
                                # Calculate MSE
                                mse = mean_squared_error(test_data,
    forecast)

                                if mse < best_mse:
                                    best_mse = mse
                                    best_order = (p, d, q)
                                    best_seasonal_order = (P, D, Q, 7)
                            except:
                                continue

print(f'Best MSE: {best_mse}')
print(f'Best Order: {best_order}')
print(f'Best Seasonal Order: {best_seasonal_order}')
```

```
Best MSE: 68.60348015163353
Best Order: (2, 0, 2)
Best Seasonal Order: (1, 1, 0, 7)
```

8. plot forecasting

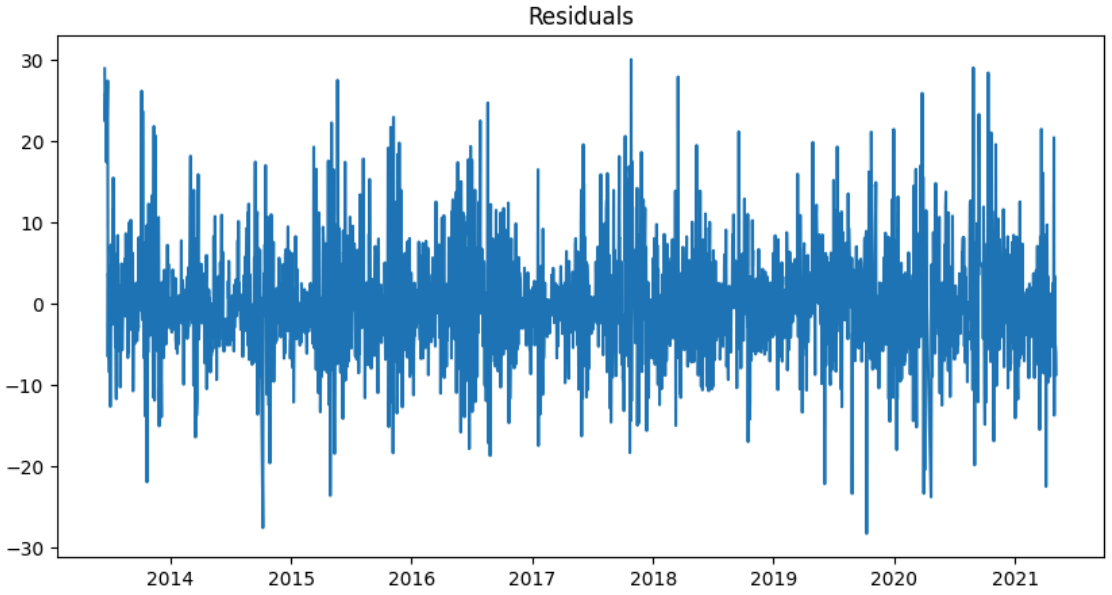
```
# Plot the best forecast
print("Plotting the best forecast...")
model = SARIMAX(train_data, order=best_order,
seasonal_order=best_seasonal_order)
model_fit = model.fit(disp=False)
forecast = model_fit.forecast(steps=len(test_data))

plt.figure(figsize=(10, 5))
plt.plot(data.index, data, label='Observed')
plt.plot(test_data.index, forecast, label='Forecast', color='red')
plt.title('Optimized SARIMA Forecast')
plt.xlabel('Date')
plt.ylabel('Average Price')
plt.legend()
plt.show()
```



8. plot residual

7.

```
# Plot the best forecast
print("Plotting the best forecast...")
model = SARIMAX(train_data, order=best_order,
seasonal_order=best_seasonal_order)
model_fit = model.fit(disp=False)
forecast = model_fit.forecast(steps=len(test_data))

plt.figure(figsize=(10, 5))
plt.plot(data.index, data, label='Observed')
plt.plot(test_data.index, forecast, label='Forecast', color='red')
plt.title('Optimized SARIMA Forecast')
plt.xlabel('Date')
plt.ylabel('Average Price')
plt.legend()
plt.show()
```



3.3. Klasifikasi

Pada tugas klasifikasi gambar tomat belum matang, matang, dan rusak kami menggunakan metode SVM, berikut untuk potongan kode implementasi beserta screenshot UI yang telah dibuat:

1. Pengaturan Lingkungan dan Pustaka yang Digunakan

```
# 1. Import Libraries
import os
```

```
import cv2
import numpy as np
from sklearn import svm
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import accuracy_score, classification_report
import joblib
from google.colab import drive
from skimage.feature import local_binary_pattern
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler  # Import
   StandardScaler here
# Instead of keras.preprocessing.image, import from
   tensorflow.keras.preprocessing.image
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

```
# 2. Connect to Google Drive
drive.mount('/content/drive')
# 3. Define Dataset Folders Individually
categories = ['damaged','ripe', 'unripe']

# Define individual paths for each category manually
damaged_folder = "/content/drive/My Drive/Big Data/tomat/Damaged"
ripe_folder = "/content/drive/My Drive/Big Data/tomat/Ripe"
unripe_folder = "/content/drive/My Drive/Big Data/tomat/Unripe"

# Store all paths in a list
category_folders = [damaged_folder, ripe_folder, unripe_folder]
```

2. Data Preprocessing (Prapemrosesan Data Gambar)

Menyiapkan gambar agar dapat digunakan untuk ekstraksi fitur dan pelatihan model, dengan melakukan Resize (Mengubah Ukuran) dan (Konversi Warna).

```
# 5. Load and preprocess the dataset
def load_and_preprocess_image(image_path, size=(128, 128)):
    image = cv2.imread(image_path)
    if image is None:
        raise ValueError(f"Could not read image: {image_path}")
    resized_image = cv2.resize(image, size)
    # Convert the image to grayscale
    grayscale_image = cv2.cvtColor(resized_image, cv2.COLOR_BGR2GRAY)
    return grayscale_image
```

3. Feature Extraction (Ekstraksi Fitur)

Mengambil informasi penting dari gambar untuk digunakan sebagai input bagi model klasifikasi. Fitur yang diambil terdiri dari Fitur Warna dan Fitur Tekstur

```
# 6. Feature Extraction Functions
```

7.

```
def extract_color_features(image):
    hsv_image = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
    mean_hue = np.mean(hsv_image[:, :, 0])
    mean_saturation = np.mean(hsv_image[:, :, 1])
    mean_value = np.mean(hsv_image[:, :, 2])
    return [mean_hue, mean_saturation, mean_value]

def extract_lbp_features(image):
    # Konversi gambar ke grayscale
    gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    # Pastikan tipe data adalah uint8
    gray_image = gray_image.astype(np.uint8)
    # Hitung LBP
    lbp = local_binary_pattern(gray_image, P=8, R=1,
    method='uniform')
    # Hitung histogram LBP
    (hist, _) = np.histogram(lbp.ravel(),bins=np.arange(0, 11),
     density=True)
    return hist
```

4. Pembagian data

Dataset dibagi menjadi data training dan testing, masing-masing dimuat dari direktori train dan test. Ukuran gambar disesuaikan.

```
# 8. Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y,
test_size=0.2, random_state=42)
```

5. Pelatihan Model

```
# 9. Initialize and Train SVM with Grid Search
param_grid = {
    'C': [0.01, 0.1, 1, 10, 100],
    'gamma': [0.001, 0.01, 0.1, 1]
}
grid = GridSearchCV(svm.SVC(kernel='rbf'), param_grid, refit=True,
verbose=2)
grid.fit(X_train, y_train)
```

6. Evaluasi Model

```
# 11. Evaluate Model Accuracy
accuracy = accuracy_score(y_test, y_pred)
print(f'Model accuracy: {accuracy * 100:.2f}%')
```

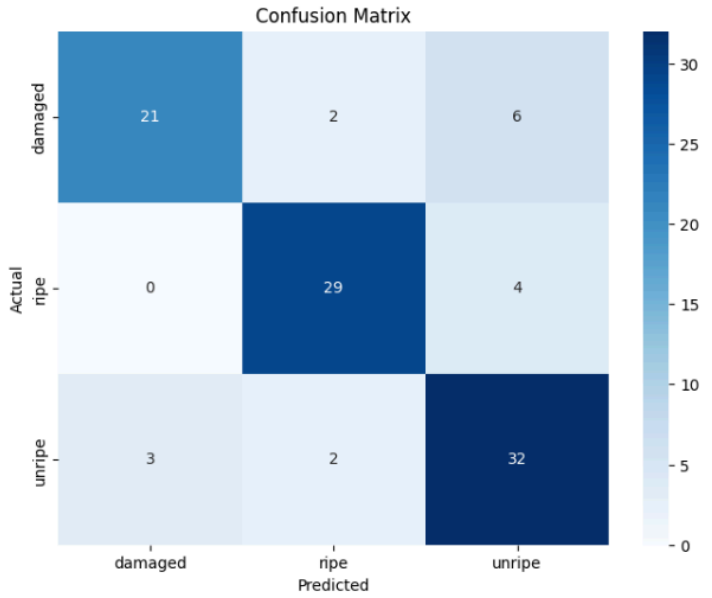
```
[CV] END .....................................C=100, gamma=1; total time=   0.0s
Model accuracy: 82.83%
```

7. Visualisasi Evaluasi

```
# Visualize the confusion matrix as well (optional)
from sklearn.metrics import confusion_matrix
import seaborn as sns
```

```
4.
5.  cm = confusion_matrix(y_test, y_pred, labels=unique_labels)
6.  plt.figure(figsize=(8, 6))
7.  sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
    xticklabels=categories, yticklabels=categories)
8.  plt.ylabel('Actual')
9.  plt.xlabel('Predicted')
10. plt.title('Confusion Matrix')
11. plt.show()
```

```
Classification Report:
              precision    recall  f1-score   support

     damaged       0.88      0.72      0.79        29
       ripe        0.88      0.88      0.88        33
     unripe        0.76      0.86      0.81        37

    accuracy                           0.83        99
   macro avg       0.84      0.82      0.83        99
weighted avg       0.83      0.83      0.83        99
```



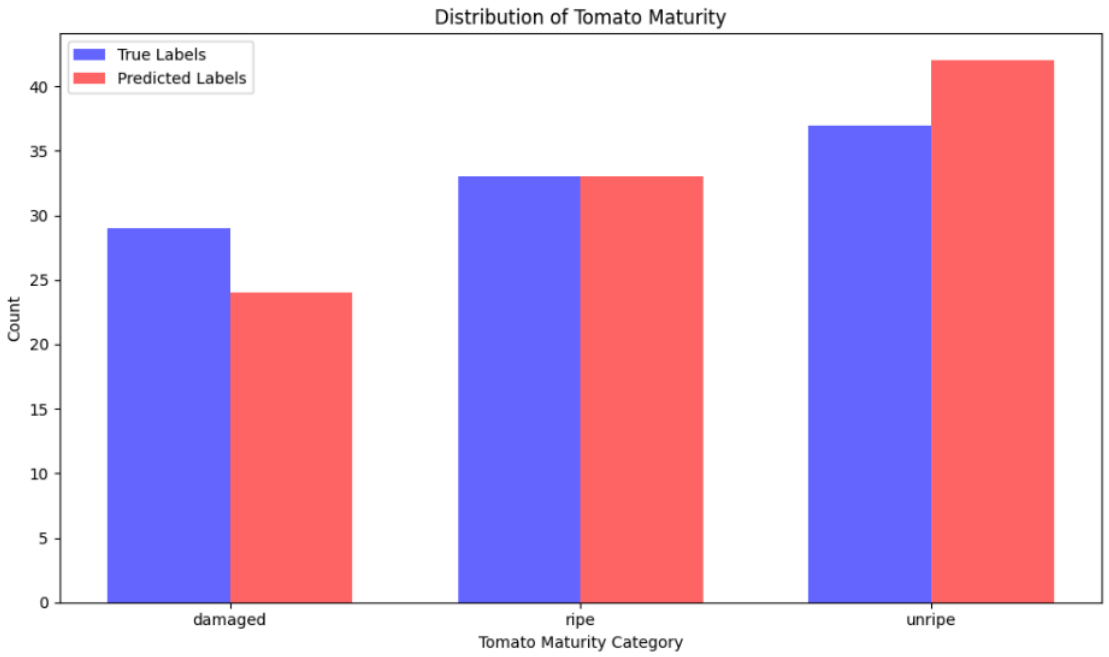
8. Visualisasi prediction

```
1.  # 14. Visualize Predictions
2.  def visualize_predictions(images, y_true, y_pred, categories):
3.      fig, axes = plt.subplots(1, len(images), figsize=(15, 5))
4.      for i, ax in enumerate(axes):
5.          ax.imshow(cv2.cvtColor(images[i], cv2.COLOR_BGR2RGB))
6.          true_label = categories[y_true[i]]
7.          pred_label = categories[y_pred[i]]
8.          ax.set_title(f"True: {true_label}\nPred: {pred_label}")
9.          ax.axis("off")
10.     plt.tight_layout()
11.     plt.show()
12.
13. # Show 5 sample predictions
14. visualize_predictions(processed_images[:12], y_test[:12], y_pred[:12],
    categories)
```

7.

9. Grafik Distribusi Kematangan Tomat

```
# 15. Plot Maturity Distribution
def plot_maturity_distribution(y_true, y_pred, categories):
    true_counts = np.bincount(y_true, minlength=len(categories))
    pred_counts = np.bincount(y_pred, minlength=len(categories))

    bar_width = 0.35
    index = np.arange(len(categories))

    plt.figure(figsize=(10, 6))
    plt.bar(index, true_counts, bar_width, label="True Labels",
color='b', alpha=0.6)
    plt.bar(index + bar_width, pred_counts, bar_width,
label="Predicted Labels", color='r', alpha=0.6)

    plt.xlabel('Tomato Maturity Category')
    plt.ylabel('Count')
    plt.title('Distribution of Tomato Maturity')
    plt.xticks(index + bar_width / 2, categories)
    plt.legend()
    plt.tight_layout()
    plt.show()

# Call the maturity distribution plot
plot_maturity_distribution(y_test, y_pred, categories)
```



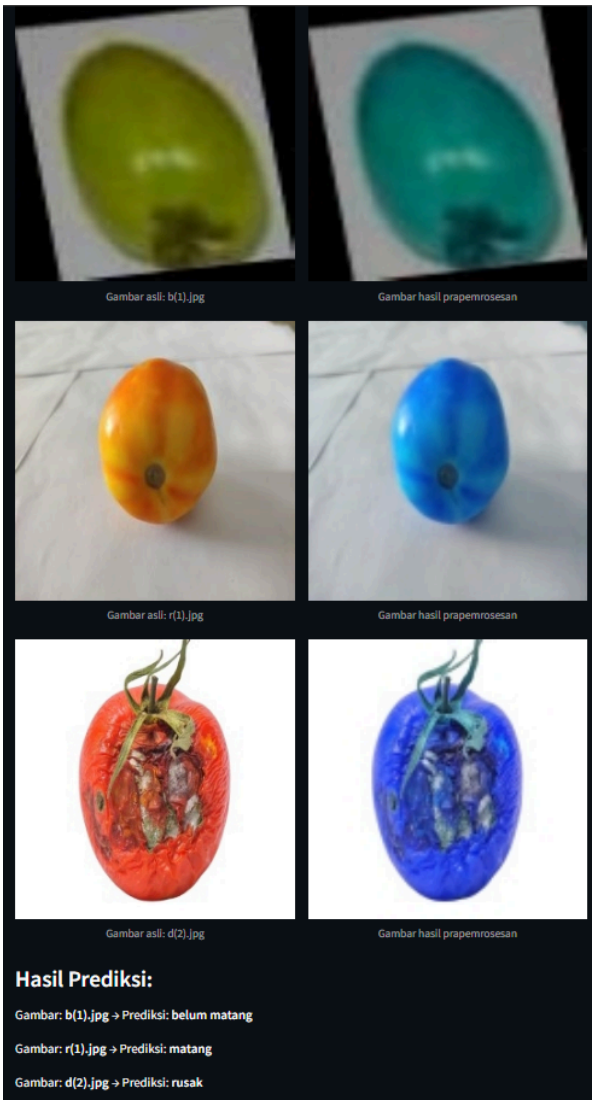
10. Menyimpan Model

Model yang telah dilatih disimpan ke Google Drive untuk memudahkan akses di masa mendatang atau pengujian lebih lanjut tanpa perlu melatih ulang model.

```
# 16. Save the Model to Google Drive
model_path = '/content/drive/My Drive/Big Data/model_svm_tomat.pkl'
joblib.dump(grid, model_path)
print(f"Model saved at {model_path}")
```

11. Hasil Klasifikasi

7.



dalam klasifikasi ini terdapat ekstrasi fitur warna dan tekstur, dengan acuan dari model yanng sudah di training sebelumnya dengan akurasi 82,83% untuk klasifikasi 3 tingkat kematangan tomat. dan data uji (data sudah di traaining) digunakan sebagai acuan dalamm melakukan klasifikasi seperti hasil diatas.

3.4. Clustering

Pada tugas clusterisasi untuk pemetaan wilayah pasar pertanian di india berdasarkan harga komoditas, kami menggunakan metode K-Means, berikut untuk potongan kode:

1. Import pustaka untuk membantu dalam proses analisis data, visualisasi, dan clustering.

```
1. import pandas as pd
2. import numpy as np
3. import matplotlib.pyplot as plt
4. import matplotlib.dates as mdates
5. import seaborn as sns
6. import plotly.express as px
7. import seaborn as sb
8. from sklearn.cluster import KMeans
9. from google.colab import drive
```

2. Menyambungkan dengan drive untuk mengambil dataset

```
1. df = pd.read_csv('/content/drive/MyDrive/BIG DATA/ANBIG_DATASET.csv')
```

3. Memilih komoditas buat tomat saja

```
1. df_Tomato = df[df['Commodity'] == 'Tomato']
```

4. Menyimpan dataset yang komoditasnya buah tomat saja

```
1. df_Tomato.to_csv('Dataset_Tomato.csv', index=False)
```

5. Menentukan nilai K menggunakan metode elbow

```
1. k_rng = range(1,10) #Membuat rentang dari 1 hingga 9 (batas atas tidak termasuk). Ini adalah jumlah klaster yang akan diuji.
2. sse = [] #list kosong untuk menyimpan nilai inertia (SSE) untuk setiap jumlah klaster.
3. for k in k_rng:
4.     km = KMeans(n_clusters=k)#Membuat objek model K-Means untuk jumlah klaster k.
5.     km.fit(df[['Max Price', 'Modal Price']])# Melatih model
6.     sse.append(km.inertia_)#menyimpan nilai inertia(sse)dari model K-means kedalam list sse
7.
8. plt.title('Metode Elbow')
9. plt.xlabel('Jumlah Cluster')
10. plt.ylabel('WCSS')
11. plt.plot(k_rng,sse)
```

6. Menentukan jumlah cluster

```
1. km = KMeans(n_clusters=4, init = 'k-means++', random_state= 42  ) #menentukan bahwa algoritma K-Means akan membagi data menjadi 4 klaster.
```

7. Mengelompokkan data berdasarkan harga Max Price dan Modal Price, dan hasilnya disimpan dalam kolom baru bernama cluster di dataframe

```
1. y_predicted = km.fit_predict(df[['Max Price', 'Modal Price']])
2. df['cluster'] = y_predicted
3. df.head()
```

8. Visualisasi untuk melihat distribusi data berdasarkan dua variabel: 'Max Price' dan 'Modal Price', serta memisahkan data berdasarkan klaster yang dihasilkan oleh model K-Means. Setiap klaster diwakili oleh warna yang berbeda (hijau, merah, biru) untuk memudahkan analisis.

```
1. df1 = df[df.cluster==0]
2. df2 = df[df.cluster==1]
3. df3 = df[df.cluster==2]
4. df4 = df[df.cluster==3]
5.
6. plt.scatter(df1['Max Price'], df1['Modal Price'], color='green')
7. plt.scatter(df2['Max Price'], df2['Modal Price'], color='red')
8. plt.scatter(df3['Max Price'], df3['Modal Price'], color='blue')
9. plt.scatter(df4['Max Price'], df4['Modal Price'], color='yellow')
10.
11. plt.xlabel('Max Price')
12. plt.ylabel('Modal Price')
13. plt.show()
```

9. Melakukan normalisasi pada kolom Max Price dan Modal Price, dengan tujuan untuk mengubah nilai dalam kedua kolom tersebut ke dalam rentang [0, 1]. Normalisasi ini penting agar data lebih terstandardisasi dan lebih mudah digunakan dalam model machine learning, yang seringkali sensitif terhadap skala data yang berbeda-beda.

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
scaler.fit(df[['Max Price']])
df[['Max Price']] = scaler.transform(df[['Max Price']])

scaler.fit(df[['Modal Price']])
df[['Modal Price']] = scaler.transform(df[['Modal Price']])
```

10. Mengelompokkan data berdasarkan dua fitur, yaitu Max Price dan Modal Price, ke dalam 4 cluster menggunakan algoritma K-means clustering. Hasil clustering ini disimpan dalam kolom baru cluster di dataframe

```
km = KMeans(n_clusters=4)
y_predicted = km.fit_predict(df[['Max Price', 'Modal Price']])
df['cluster'] = y_predicted
df.head()
```

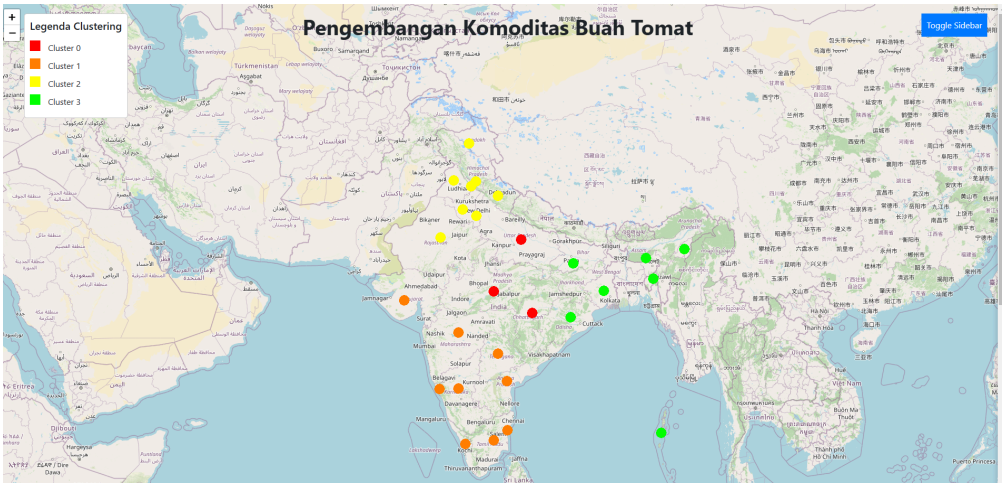
11. Memvisualisasikan hasil clustering menggunakan K-means, dengan masing-masing cluster ditampilkan dalam warna yang berbeda, dan posisi centroid setiap cluster ditandai dengan simbol 'X' berwarna hitam. Visualisasi ini membantu untuk memahami bagaimana data tersebar dan bagaimana masing-masing cluster terpisah dalam ruang fitur berdasarkan harga.

```
km.cluster_centers_
km.labels_
df1 = df[df.cluster==0]
df2 = df[df.cluster==1]
df3 = df[df.cluster==2]
df4 = df[df.cluster==3]

plt.scatter(df1['Max Price'], df1['Modal Price'], color='green', label='cluster 0')
plt.scatter(df2['Max Price'], df2['Modal Price'], color='red', label='cluster 1')
plt.scatter(df3['Max Price'], df3['Modal Price'], color='blue', label='cluster 2')
plt.scatter(df4['Max Price'], df4['Modal Price'], color='yellow', label='cluster 3')

plt.scatter(km.cluster_centers_[:,0], km.cluster_centers_[:,1], color='black', marker='X', label='centroid')
plt.legend()
plt.xlabel('Max Price')
plt.ylabel('Modal Price')
```

12. Evaluasi model menggunakan MSE

```
from sklearn.metrics import silhouette_score

def evaluate_kmeans(X, labels, model_name="K-Means"):
    silhouette_avg = silhouette_score(X, labels)

    print(f"\nEvaluasi {model_name}:")
    print(f"Silhouette Score: {silhouette_avg:.2f}")

    return silhouette_avg

# Asumsi 'kmeans_sel' adalah model KMeans yang telah di-fit
labels = kmeans_sel.predict(cluster_data)
metrics_kmeans = evaluate_kmeans(cluster_data, labels, "K-Means")
```

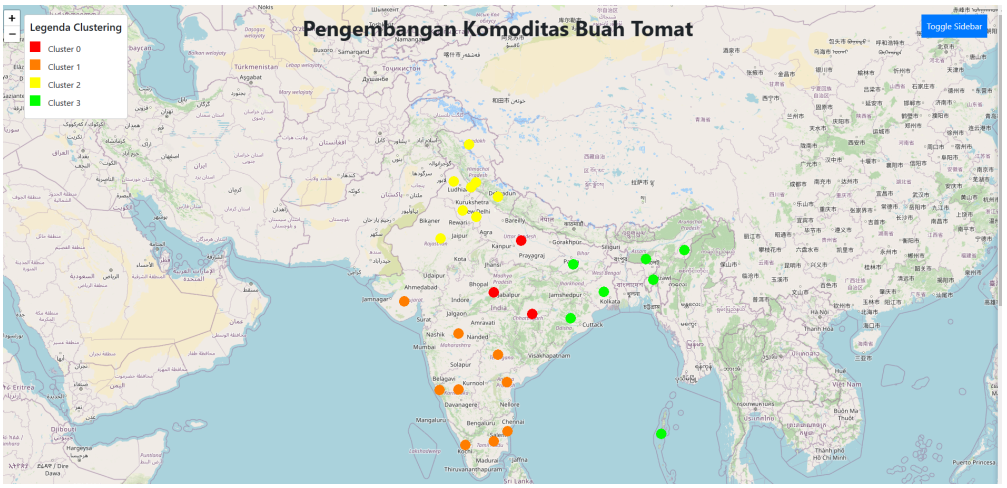
13. Hasil Clustering



Informasi label warna plot clustering, untuk cluster 1 yang berwarna merah artinya wilayah tersebut harga komoditas buah tomatnya rendah (wilayah pasar lokal atau daerah pertanian yang melimpah), untuk cluster 2 berwarna oranye artinya wilayah tersebut harga komoditasnya menengah (Wilayah pasar sedang atau daerah yang memiliki akses lebih terbatas ke produk pertanian, cluster 3 berwarna kuning artinya wilayah dengan harga komoditas tinggi (Wilayah perkotaan besar atau pasar premium), cluster 4 berwarna hijau artinya wilayah dengan harga komoditas yang sangat tinggi atau tidak stabil, seperti pasar expor atau daerah dengan fluktuasi harga musiman. Untuk evaluasi model yang saya dapatkan yaitu 0,59 dengan menggunakan evaluasi Silhouette Score

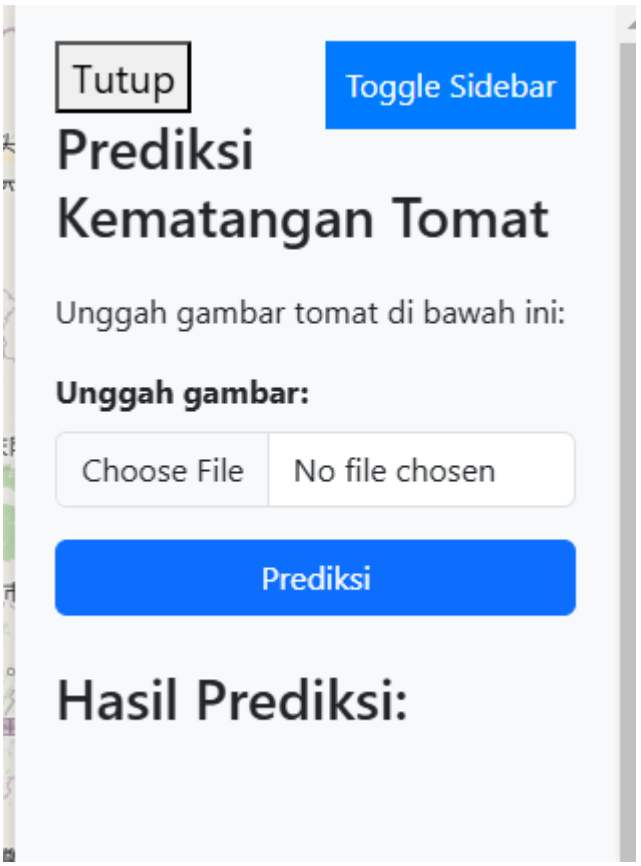
3.5. Insight

3.5.1 mapping

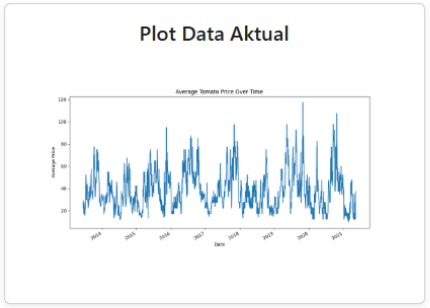


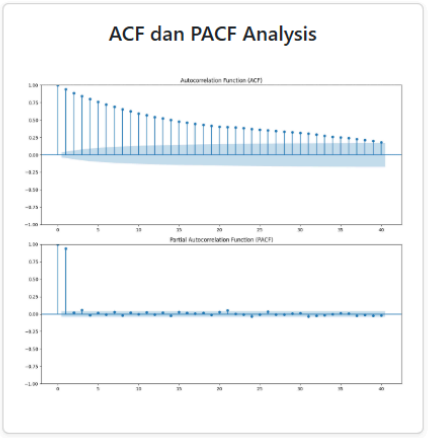
3.5.2 prediksi

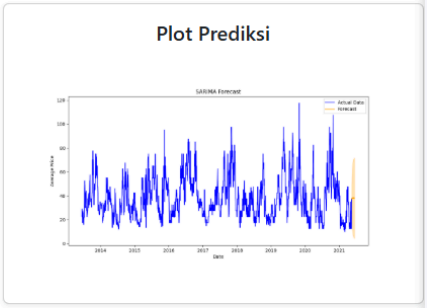
7.



3.5.3 forecasting







3.6. Decision

Berdasarkan temuan dan analisis yang dilakukan, beberapa keputusan strategis dapat diambil untuk mengoptimalkan pengelolaan komoditas tomat:

1. Perencanaan Produksi Berbasis Prediksi Harga
   Dengan adanya model prediksi harga yang akurat, petani dan pelaku agribisnis dapat menyusun jadwal produksi yang lebih efisien, menghindari overproduksi saat harga rendah, serta memaksimalkan produksi pada periode harga tinggi. Keputusan ini membantu stabilitas pendapatan petani dan mengurangi pemborosan hasil panen.

2. Penerapan Sistem Otomatisasi Seleksi Buah
   Keberhasilan klasifikasi tingkat kematangan tomat memberikan peluang untuk mengembangkan sistem otomatisasi berbasis komputer vision di lapangan. Hal ini memungkinkan seleksi buah dilakukan lebih cepat dan akurat, mengurangi kesalahan manual, serta meningkatkan efisiensi dalam rantai pasokan.

3. Optimasi Distribusi Berbasis Klasterisasi Wilayah
   Informasi klasterisasi wilayah pasar memungkinkan distribusi tomat yang lebih terarah berdasarkan pola harga. Pedagang dan distributor dapat menyesuaikan volume pengiriman ke wilayah tertentu untuk menghindari kelebihan pasokan atau kekurangan stok, sehingga membantu menstabilkan harga di tingkat regional.

4. Pemanfaatan Data Big Data untuk Kebijakan Agrikultur
   Penggunaan data dengan volume besar dan beragam format memberikan landasan bagi pemangku kebijakan untuk menyusun regulasi dan kebijakan berbasis data. Contohnya, subsidi harga dapat diarahkan pada wilayah dengan fluktuasi harga ekstrem, atau insentif produksi diberikan kepada petani di wilayah yang memiliki potensi pasar tinggi.

5. Pengembangan Infrastruktur Data untuk Keberlanjutan Sistem
   Untuk memastikan keberlanjutan sistem, diperlukan pengembangan infrastruktur data yang mendukung pengumpulan, pemrosesan, dan analisis data secara real-time. Hal ini mencakup investasi pada sensor lapangan, sistem pengolahan data cloud, dan integrasi dengan platform digital yang dapat diakses oleh seluruh pemangku kepentingan.

Keputusan-keputusan ini memberikan kerangka kerja yang terintegrasi dalam mendukung pertanian cerdas berbasis data, sekaligus memastikan keberlanjutan pengelolaan komoditas tomat yang lebih efektif dan responsif terhadap dinamika pasar.

7.

Conclusion

Penelitian ini berhasil menjawab sebagian besar masalah yang diangkat terkait tantangan dalam pengelolaan komoditas tomat, seperti fluktuasi harga, ketidakefisienan produksi, dan distribusi yang tidak merata. Melalui penerapan teknologi Big Data dan metode analisis yang relevan, beberapa solusi telah diusulkan dan diuji, dengan hasil sebagai berikut:

1. Peramalan Harga

   Fluktuasi harga tomat dapat ditangani lebih efektif menggunakan model SARIMA (*Seasonal Autoregressive Integrated Moving Average*). SARIMA memberikan prediksi akurat untuk data *time series*, dengan nilai MSE Best MSE: 68.60348015163353 sementara regresi linier memprediksi tren berdasarkan variabel terkait. Prediksi ini membantu petani dan pelaku pasar merencanakan produksi dan distribusi dengan efisien, meminimalkan risiko kerugian akibat perubahan harga.

2. Klasifikasi Tingkat Kematangan Tomat

   Klasifikasi tingkat kematangan menggunakan metode SVM mencapai akurasi yang baik (82,83%), memberikan solusi untuk otomatisasi seleksi buah berdasarkan kategori kematangan. Hasil ini menunjukkan bahwa pendekatan berbasis data mampu mengurangi kesalahan dalam seleksi manual.

3. Klasterisasi Wilayah Pasar

   Dengan algoritma K-Means, wilayah pasar berhasil dikelompokkan berdasarkan pola harga, memberikan wawasan yang dapat digunakan untuk strategi distribusi dan stabilisasi harga.
   Namun, beberapa tantangan masih perlu ditangani lebih lanjut, seperti peningkatan akurasi prediksi dan klasifikasi dengan menambahkan lebih banyak data atau fitur yang relevan, serta pengembangan sistem real-time untuk analisis data. Meskipun demikian, penelitian ini telah membuktikan bahwa integrasi teknologi Big Data dapat memberikan solusi inovatif dan praktis dalam mengatasi tantangan pengelolaan komoditas tomat. Kesimpulannya, penelitian ini telah menyelesaikan sebagian besar masalah utama yang diangkat dan memberikan landasan kuat untuk pengembangan lebih lanjut di bidang pertanian cerdas berbasis data.

References

[1] R. R. Rachmawati, “Smart Farming 4.0 Untuk Mewujudkan Pertanian Indonesia Maju, Mandiri, Dan Modern,” *Forum Penelit. Agro Ekon.*, vol. 38, no. 2, p. 137, 2021, doi: 10.21082/fae.v38n2.2020.137-154.

[2] D. sarjon Juliansa hengki and Sumijan, “Model Manajemen Big Data Komoditas Beras untuk Kebijakan Pangan Nasional,” *Resti*, vol. 1, no. 1, pp. 19–25, 2017.

[3] L. Maysofa, K. Umam Syaliman, and Sapriadi, “Implementasi Forecasting Pada Penjualan Inaura Hair Care Dengan Metode Single Exponential Smoothing,” *J. Test. dan Implementasi Sist. Inf.*, vol. 1, no. 2, pp. 82–91, 2023.

[4] E. Kwok and W. Susanti, “Penerapan Metode Regresi Linier dalam Aplikasi Sistem Peramalan Jumlah Bahan Baku untuk Produksi Tahu,” *Mhs. Apl. Teknol. Komput. dan Inf.*, vol. 1, no. 2, pp. 1–8, 2019.

[5] K. Tingkat *et al.*, “Tomat Dengan Variasi Model Warna,” vol. 1, no. 2, pp. 203–210, 2023.

[6] E. Prasetyaningrum and P. Susanti, “Perbandingan Algoritma K-Means Dan K-Medoids Untuk Pemetaan Hasil Produksi Buah-Buahan,” *J. Media Inform. Budidarma*, vol. 7, no. 4, p. 1775, 2023, doi: 10.30865/mib.v7i4.6477.

[7] W. B. Sebayang, “Adolescent Childbirth with Asphyxia Neonatorum,” *J. Aisyah J. Ilmu Kesehat.*, vol. 7, no. 2, pp. 669–672, 2022, doi: 10.30604/jika.v7i2.1507.

[8] F. Rozak, “Prediksi Jumlah Kunjungan Pasien Rawat Jalan Menggunakan Metode Resgresi Linier Sederhana,” *PROSISKO J. Pengemb. Ris. dan Obs. Sist. Komput.*, vol. 11, no. 2, pp. 284–293, 2024, doi: 10.30656/prosisko.v11i2.8756.

[9] J. Pebralia, “Analisis Curah Hujan Menggunakan Machine Learning Metode Regresi Linier Berganda Berbasis Python dan Jupyter Notebook,” *J. Ilmu Fis. dan Pembelajarannya*, vol. 6, no. 2, pp. 23–30, 2022, doi: 10.19109/jifp.v6i2.13958.

[10] S. Thayyil *et al.*, “Hypothermia for moderate or severe neonatal encephalopathy in low-income and middle-income countries (HELIX): a randomised controlled trial in India, Sri Lanka, and Bangladesh,” *Lancet Glob. Heal.*, vol. 9, no. 9, pp. e1273–e1285, 2021, doi: https://doi.org/10.1016/S2214-109X(21)00264-3.

[11] N. Sulistianingsih, F. Astutik, and A. Rahman, “Optimasi Seleksi Fitur Untuk Perbaikan Akurasi Support Vector Machine Classifier Pada Klasifikasi Citra Tanaman Rimpang,” vol. 14, no. 2, pp. 526–532, 2024.

[12] F. Liantoni, “Klasifikasi Daun Dengan Perbaikan Fitur Citra Menggunakan Metode K-Nearest Neighbor,” *J. Ultim.*, vol. 7, no. 2, pp. 98–104, 2016, doi: 10.31937/ti.v7i2.356.

[13] A. A. Vernanda, A. Faisol, and N. Vendyansyah, “Penerapan Metode K-Means Clustering Untuk Pemetaan Daerah Rawan Kecelakaan Lalu Lintas Di Kota Malang Berbasis Website,” *JATI (Jurnal Mhs. Tek. Inform.*, vol. 5, no. 2, pp. 836–844, 2021, doi: 10.36040/jati.v5i2.3791.

[14] D. Rika Widianita, “No 主観的健康感を中心とした在宅高齢者における 健康関連指標に関する共分散構造分析Title,” *AT-TAWASSUTH J. Ekon. Islam*, vol. VIII, no. I, pp. 1–19, 2023.

7.