BAB Ⅲ

METODE PENELITIAN

3.1 Objek Penelitian

Penelitian ini berfokus pada prediksi tingkat stres manusia berdasarkan faktor lingkungan dan aktivitas fisik dengan menggunakan algoritma *Random Forest*. Objek penelitian dalam studi ini adalah individu yang memiliki pola aktivitas harian yang berbeda-beda, di mana tingkat stres mereka dipengaruhi oleh kelembaban udara, suhu lingkungan, dan jumlah langkah harian.

3.2 Prosedur Penelitian

Prosedur penelitian terdiri dari 5 tahap utama:

1. Dataset: Dataset diambil dari *website Kaggle* (Laavanya Rachakonda, 2022)
2. *EDA*: Proses ini bertujuan untuk memahami pola dan distribusi data, mengidentifikasi nilai yang hilang, serta hubungan antar variabel.
3. *Preprocessing*: Menyiapkan data dengan mengatasi nilai hilang, normalisasi fitur, dan *encoding* variabel kategori jika diperlukan.
4. Pemodelan: Pembangunan model menggunakan algoritma *Random Forest* dengan validasi silang *5-fold.*
5. Evaluasi: Menggunakan metrik seperti ***accuracy, precision, recall, F1-score***, dan ***ROC curve*** untuk menilai kinerja model.

Gambar 3. 1 Diagram Alur Proses

3.3 Dataset

Dataset yang digunakan dalam penelitian ini berisi informasi tentang tingkat stres manusia yang dipengaruhi oleh faktor kelembaban udara, suhu lingkungan, dan jumlah langkah. Dataset ini digunakan untuk membangun model prediksi dengan algoritma Random

Forest. Namun, dataset ini tidak mencakup informasi demografis, seperti usia, jenis kelamin, atau status kesehatan individu yang terlibat dalam pengumpulan data. Keterbatasan ini dapat mempengaruhi kemampuan generalisasi model, karena model yang dibangun hanya relevan untuk individu dewasa sehat dan tidak dapat digeneralisasi pada populasi yang lebih beragam, seperti individu dengan gangguan kesehatan atau kelompok usia lainnya.

Penelitian ini menggunakan data sekunder yang diperoleh dari penyedia open data di situs *web Kaggle*. Dataset yang dipilih, berjudul *Human Stress Detection* (Laavanya Rachakonda, 2022) https://www.kaggle.com/ datasets/laavanya/stress-level-detection, digunakan untuk melatih dan menguji model *Random Forest* dalam memprediksi tingkat stres. Dataset ini dipilih karena memuat variabel yang relevan dan telah divalidasi dalam penelitian sebelumnya, dengan kriteria Variabel Sampel 2001 baris (80% latih = 1600, 20% uji = 401).

Tabel 3. 1 Variabel Dataset

Kolom	Tipe Data	Range/Nilai	Sumber Pengukuran
Humidity	Numerik	0–80% RH (±3% accuracy)	Sensor Kelembapan (Palmar Sweat)
Temperature	Numerik	-10°F hingga +85°F	Sensor Suhu Kontak
Step Count	Numerik	0–200 langkah/hari	Akselerometer 3-Axis
Stress Level	Kategorik	0 (Rendah) 1 (Normal) 2 (Tinggi)	DNN berbasis data sensor fisiologis

Dataset ini memiliki 4 variabel utama yang digunakan dalam analisis, yaitu:

1. *Humidity* (Kelembaban) (%) : Mengukur kadar keringat di telapak tangan. Stres tinggi menyebabkan peningkatan sekresi keringat (berkorelasi dengan aktivitas sistem saraf simpatik). Sensor menggunakan prinsip resistansi listrik kulit (GSR).
2. *Temperature* (Suhu)°F: Memantau suhu kulit, terutama di ekstremitas. Stres akut menyebabkan vasokonstripsi, sehingga suhu turun. Sensor ditempatkan di pergelangan tangan atau telapak tangan.
3. *Step Count* (Jumlah Langkah) : Menghitung langkah melalui gerakan 3 sumbu (x, y, z). Pola langkah tidak teratur atau intensitas abnormal bisa indikasi stres. Data diambil dari aktivitas harian.
4. *Stress Level* (Tingkat Stres) : Variabel target yang dikategorikan sebagai:

1) Stres Rendah (0)
2) Stres Normal (1)
3) Stres Tinggi (2)

Berikut tampilan struktur dataset yang digunakan dalam penelitian ini:

Tabel 3. 2 Struktur Dataset

No	Humidity	Temperature	Step Count	Stress Level
1	21.33	90.33	123	1
2	21.41	90.41	93	1
3	27.12	96.12	196	2
4	27.64	96.64	177	2
5	10.87	79.87	87	0
…	…	…	…	…
1997	21.82	90.82	96	1
1998	10.45	79.45	45	0
1999	27.22	96.22	135	2
2000	12.46	81.46	64	0

3.4 Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) adalah tahap awal dalam analisis data yang bertujuan untuk memahami karakteristik dataset secara menyeluruh sebelum dilakukan pemodelan machine learning. EDA membantu dalam mengidentifikasi pola, tren, serta anomali dalam data yang dapat memengaruhi hasil analisis. Selain itu, tahap ini juga berguna untuk menemukan potensi masalah seperti *missing values*, *outlier*, *multikolinearitas*, serta distribusi data yang dapat berpengaruh terhadap performa model (haloryan, 2024)

Dalam praktiknya, EDA dilakukan dengan berbagai teknik, baik secara deskriptif maupun visualisasi data. Analisis deskriptif mencakup perhitungan statistik dasar seperti mean**,** median**,** modus**,** standar deviasi**,** minimum**,** maksimum, dan distribusi data untuk masing-

masing variabel. Sementara itu, visualisasi data sering kali menggunakan grafik seperti histogram**,** box plot**,** scatter plot**,** pair plot, atau heatmap untuk memahami hubungan antarvariabel (Kurniawati Ery Yulia, 2025)

Dengan melakukan EDA secara menyeluruh, kita dapat memperoleh pemahaman yang lebih dalam mengenai data, mengidentifikasi permasalahan yang mungkin terjadi, serta menentukan strategi terbaik sebelum melakukan pemodelan machine learning. Hal ini sangat penting untuk memastikan bahwa model yang dibangun memiliki akurasi tinggi dan mampu memberikan prediksi yang lebih andal.

3.4.1 Analisis Statistik Deskriptif

Tabel 3. 3 Statistik Deskiptif

Statistik	Humidity	Temperature	Step Count	Stress Level
Count	2001	2001	2001	2001
Mean (Rata-rata)	20.00	89.00	100.14	1.10
Std (Standar Deviasi)	5.78	5.78	58.18	0.77
Min (Minimum)	10.00	79.00	0.00	0.00
25% (Q1)	15.00	84.00	50.00	0.00
50% (Median/Q2)	20.00	89.00	101.00	1.00
75% (Q3)	25.00	94.00	150.00	2.00
Max (Maksimum)	30.00	99.00	200.00	2.00
Skewness	0.00	~0.00	~0.00	-0.18
Kurtosis	-1.20	-1.20	-1.21	-1.30

Pada tahap ini, dilakukan analisis statistik dasar untuk memahami distribusi data dalam dataset. Analisis ini membantu dalam mengidentifikasi pola umum, sebaran data, serta potensi anomali yang dapat memengaruhi hasil model. Beberapa metrik statistik utama yang digunakan meliputi:

1. Mean

Mean atau rata-rata adalah nilai yang diperoleh dengan menjumlahkan semua data dalam suatu variabel lalu membaginya dengan jumlah total observasi. Mean memberikan gambaran umum mengenai pusat data, tetapi sangat sensitif terhadap ***outlier*** (nilai ekstrem).

Rumus Mean:

$$\bar{X} = \frac{\sum X_i}{N}$$

Keterangan:

$\bar{X}$ = adalah Mean rata-rata

$\sum X_i$ = adalah jumlah seluruh nilai dalam dataset

N = jumlah total dalam dataset

2. Median

Median adalah nilai yang membagi dataset menjadi dua bagian yang sama besar ketika data diurutkan dari nilai terkecil hingga terbesar. Median lebih ***robust*** terhadap *outlier* dibandingkan mean, sehingga lebih baik digunakan untuk data yang tidak berdistribusi normal atau memiliki pencilan ekstrem.

Cara menghitung median**:**

1) Jika jumlah data ganjil, median adalah nilai tengah.
2) Jika jumlah data genap, median dihitung sebagai rata-rata dari dua nilai tengah.

3. Standard Deviation

Standard deviation mengukur **sebaran** atau **dispersi** data terhadap nilai rata-rata. Jika simpangan baku kecil, berarti data berkumpul dekat dengan mean; jika besar, berarti data tersebar lebih jauh.

1) Rumus Simpangan Baku untuk populasi (σ) :

$$\sigma = \sqrt{\frac{\sum(X_i -)^2}{N}}$$

2) Rumus Simpangan Baku untuk Sampel ($\boldsymbol{s}$):

$$\boldsymbol{s = \sqrt{\frac{\sum(X_i - \bar{X})^2}{N-1}}}$$

Keterangan:

$\boldsymbol{\sigma}$ = Simpangan baku populasi

$\boldsymbol{s}$ = Simpangan baku sampel

$\boldsymbol{X_i}$ = Nilai individu dalam dataset

$\boldsymbol{\mu}$ = Mean (rata-rata populasi)

$\boldsymbol{\bar{X}}$ = Mean (rata-rata sampel)

$\boldsymbol{N}$ = Jumlah total data dalam populasi

$\boldsymbol{N-1}$ = Digunakan pada sampel untuk **menghindari bias estimasi.**

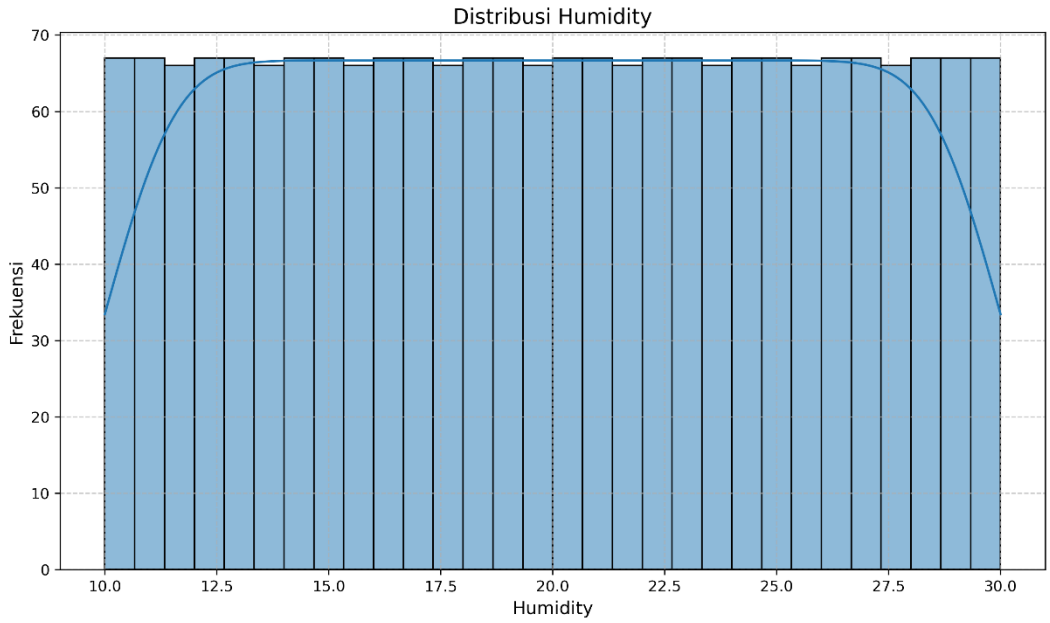
4. Minimum dan Maksimum

Minimum dan maksimum menunjukkan **nilai terkecil dan terbesar** dalam dataset, yang berguna untuk memahami rentang data serta mendeteksi ***outlier***.

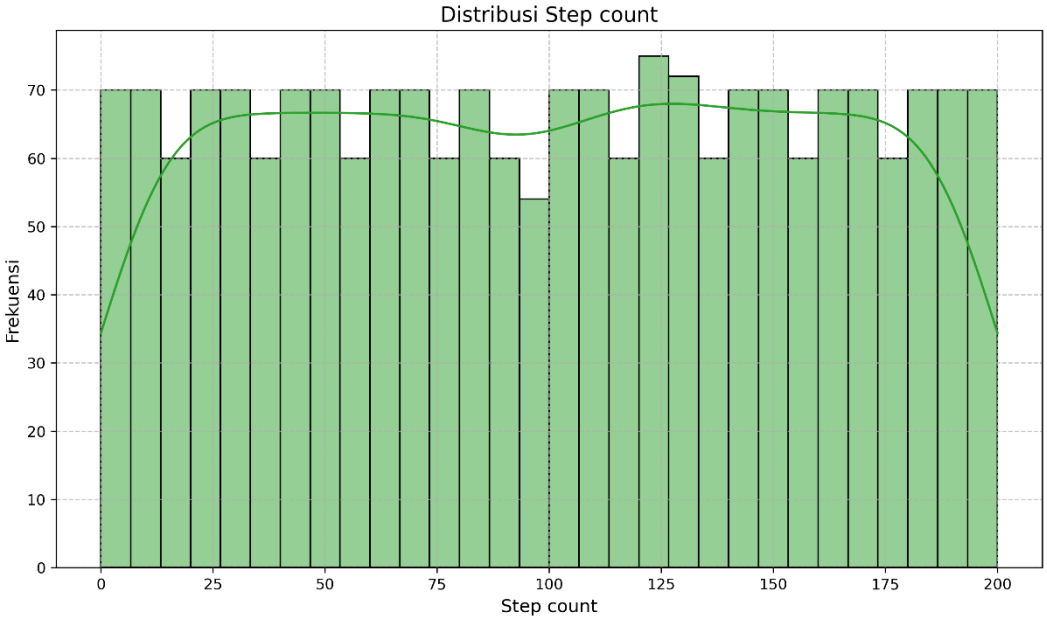
3.4.2 Analisis Visualisasi Distribusi Data

Analisis distribusi data bertujuan untuk memahami karakteristik masing-masing fitur (variabel) dalam dataset secara mendalam. Proses ini dilakukan menggunakan teknik visualisasi, seperti **histogram, boxplot,** dan **pairplot,** yang memberikan gambaran menyeluruh mengenai bentuk distribusi data, keberadaan outlier, serta potensi hubungan antar variabel. Hasil dari analisis visualisasi ini menjadi dasar dalam pengambilan keputusan pada tahap **preprocessing data.**

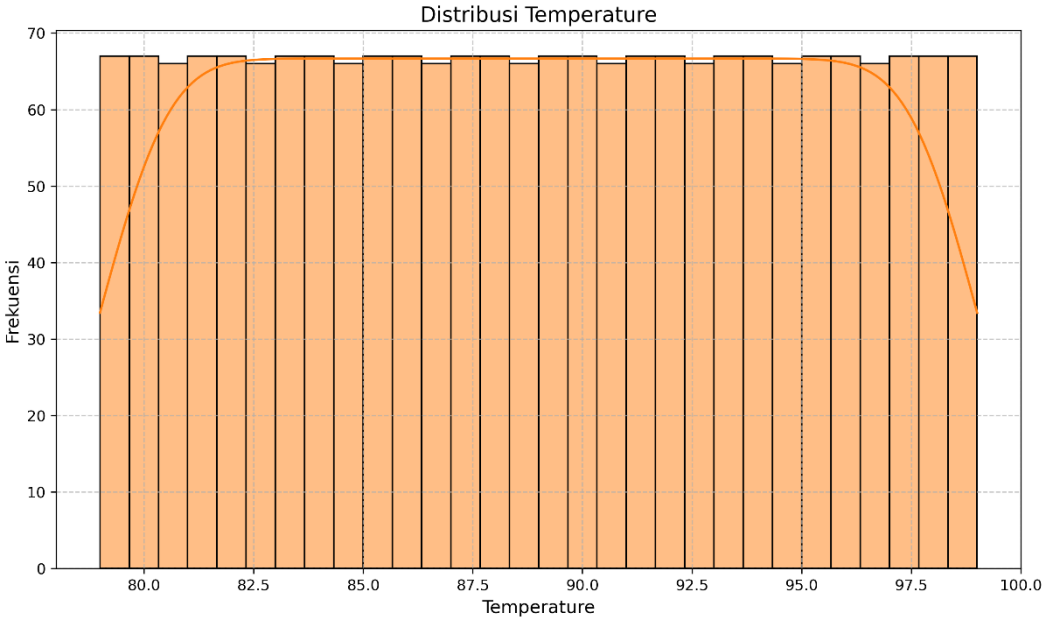
Visualisasi Histogram:



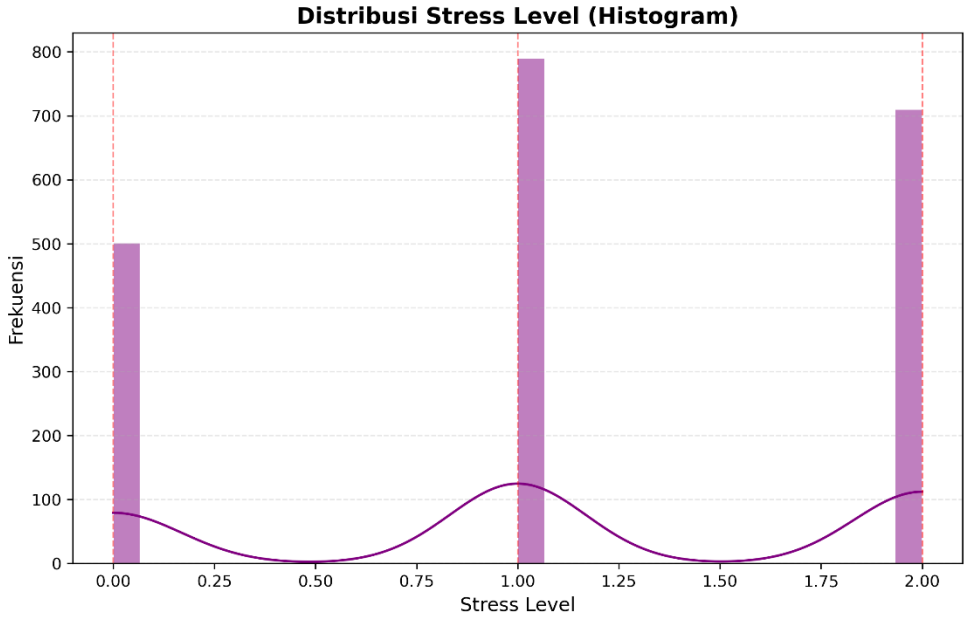
Gambar 3. 2 Visualisasi Histogram Distribusi Humidity



Gambar 3. 3 Visualisasi Histogram Distribusi Step Count



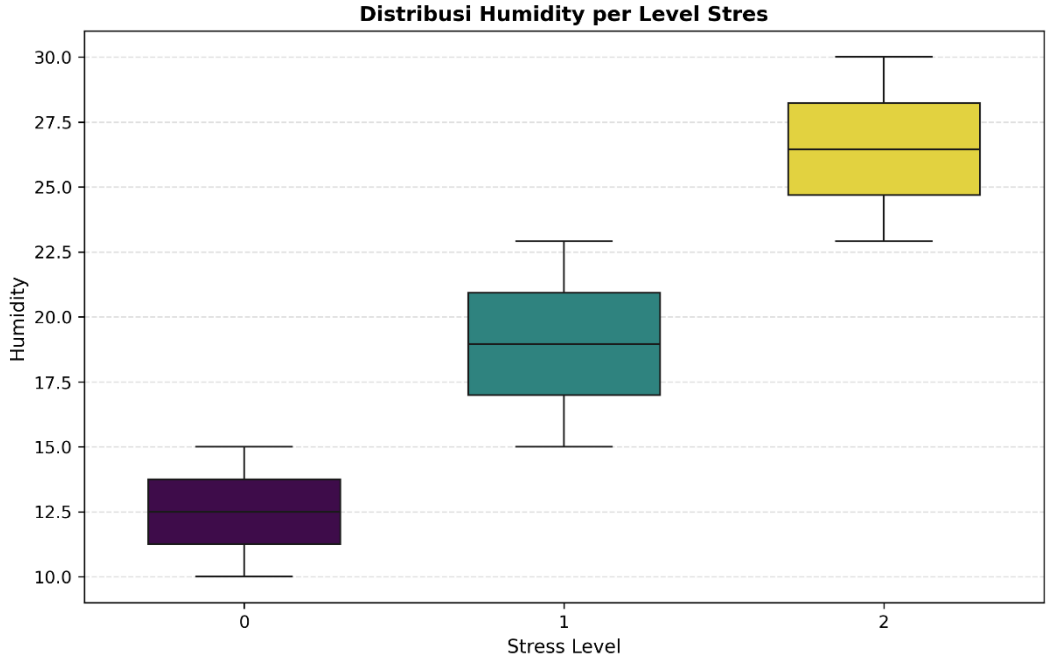
Gambar 3. 4 Visualisasi Histogram Distribusi Temperature



Gambar 3. 5

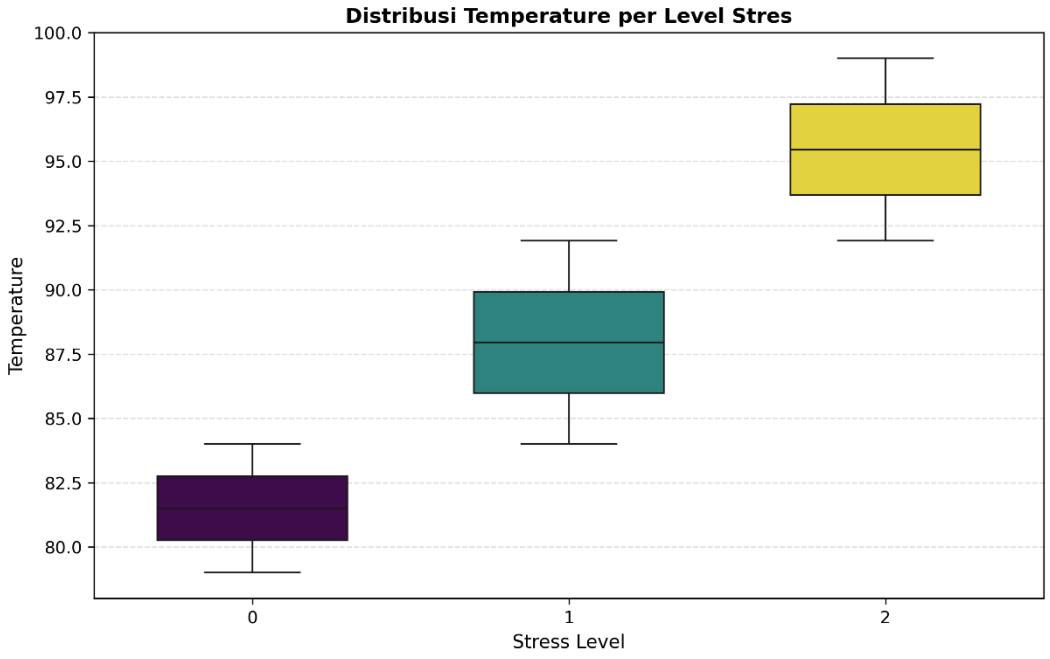
Visualisasi Histogram Distribusi Stress Level

Visualisasi Boxplot:

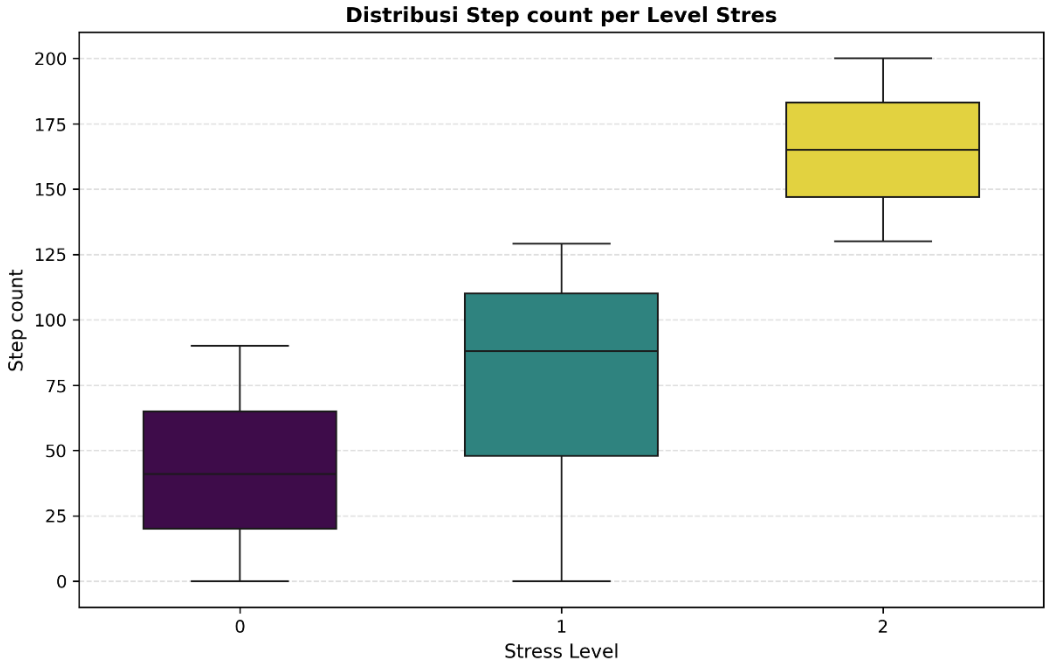


Gambar 3.

6 Visualisasi Boxplot Distribusi Humidity

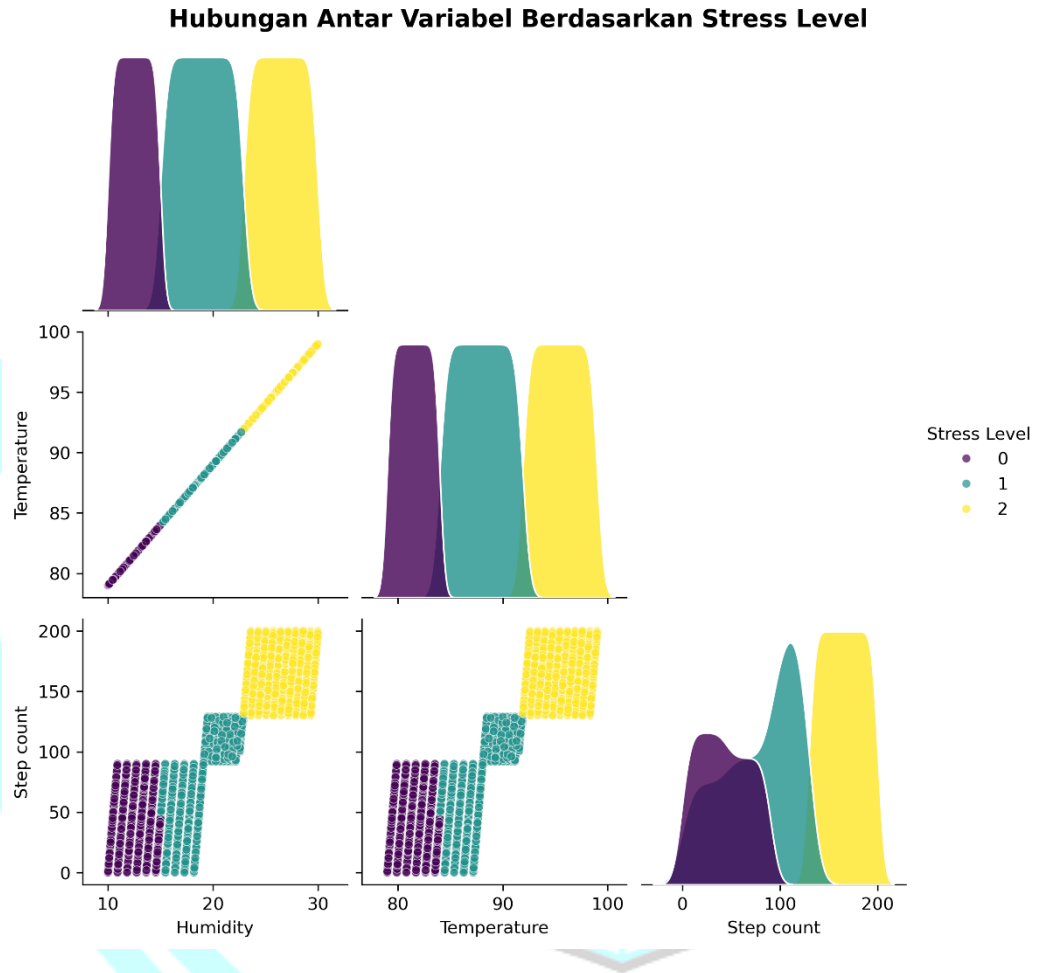


Gambar 3. 7 Visualisasi Boxplot Distribusi Temperature



Gambar 3. 8 Visualisasi Boxplot Distribusi Stress Level

Visualisasi Pairplot:



Gambar 3. 9 Visualisasi Pairplot Hubungan Antar Fitur

3.4.3 Analisis Korelasi Antar Variabel

Analisis korelasi dilakukan untuk mengidentifikasi hubungan linear dan non-linear antar variabel. Koefisien korelasi Pearson digunakan untuk mengukur hubungan linear, sedangkan Spearman digunakan untuk hubungan monoton. Signifikansi statistik diuji dengan p-value (α=0.05). Multikolinearitas dinilai menggunakan Variance Inflation Factor (VIF), dengan nilai >10 menunjukkan redudansi fitur.

1. Korelasi Pearson

Rumus Analisis Korelasi Pearson:

$$r = \frac{\sum(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum(X_i - \bar{X})^2 \cdot \sum(Y_i - \bar{Y})^2}}$$

Keterangan:

r = Koefisien korelasi Pearson, nilai yang menunjukkan hubungan linier Antar dua variabel.

X_i = Nilai individu dari variabel **X** (misalnya suhu).

Y_i = Nilai individu dari variabel **Y** (misalnya tingkat stres).

$\bar{X}$ = Mean (rata-rata) dari variabel **X.**

$\bar{Y}$ = Mean (rata-rata) dari variabel Y.

$\sum(X_i - \bar{X})(Y_i - \bar{Y})$ = Jumlah hasil perkalian selisih nilai individu dari rata-rata untuk kedua variabel.

$\sum(X_i - \bar{X})^2$ = Jumlah kuadrat dari selisih nilai individu dengan rata-rata untuk variabel X.

$\sum(Y_i - \bar{Y})^2$ = Jumlah kuadrat dari selisih nilai individu dengan rata-rata untuk variabel Y.

$\sqrt{\sum(X_i - \bar{X})^2 \cdot \sum(Y_i - \bar{Y})^2}$ = Hasil kali dari akar kuadrat varians masing-masing variabel.

Setelah perhitungan, hasilnya divisualisasikan menggunakan ***heatmap*** **korelasi** untuk mempermudah interpretasi hubungan antar variabel.

2. Korelasi Spearman

Rumus analisis Korelasi Spearman:

$$r_s = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)}$$

Keterangan:

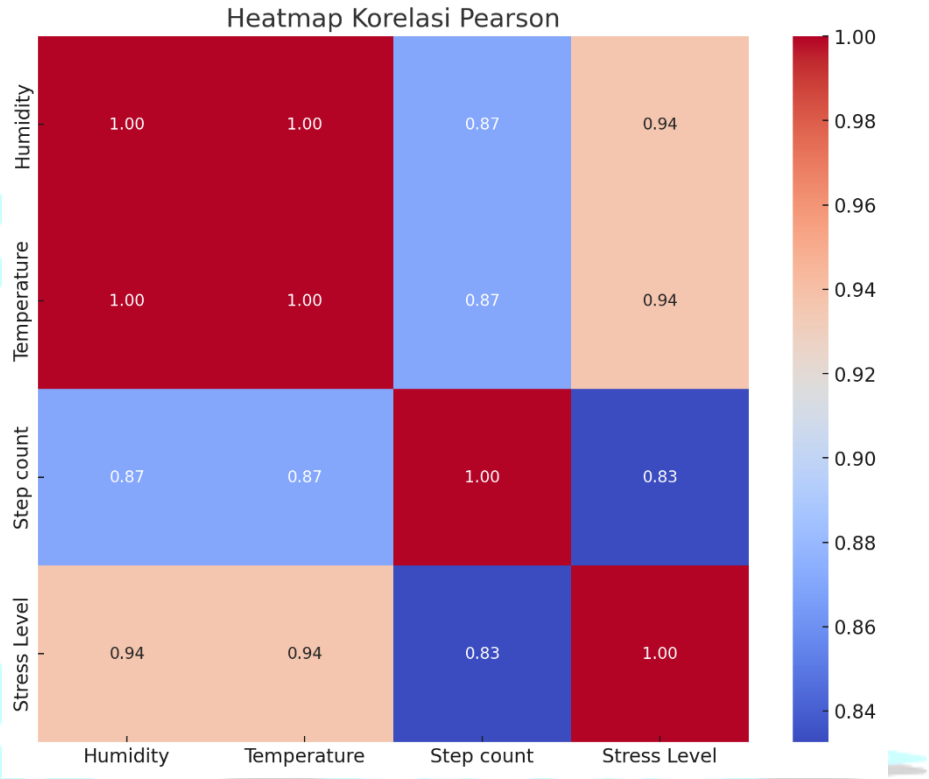
d_i = Selisih peringkat antar pasangan

r_s = Koefisien korelasi Spearman

n = Jumlah pasangan data

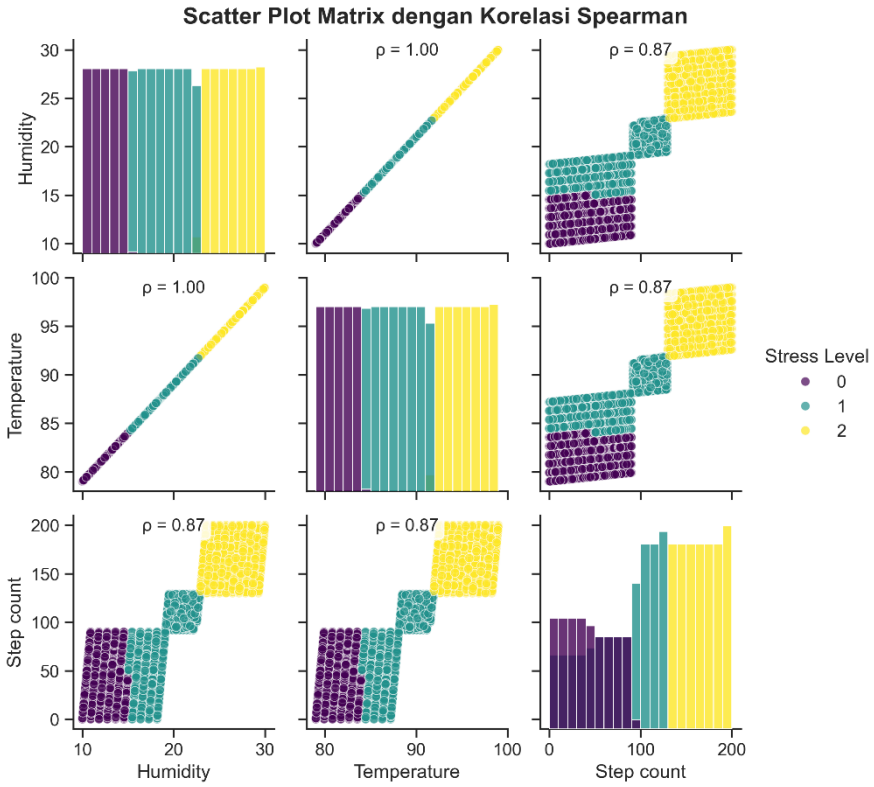
Setelah perhitungan, hasilnya divisualisasikan menggunakan ***scatter*** untuk mempermudah interpretasi hubungan antar variabel.

Visualisasi Heatmap Korelasi Pearson:



Gambar 3. 10 Visualisasi Heatmap Korelasi Pearson

Visualisasi Scatter Plot:



Gambar 3. 11 Visualisasi Scatter Plot Korelasi Spearman

3.4.4 Analisis Multikolinearitas

Dalam penelitian ini, pengujian multikolinearitas dilakukan untuk memastikan bahwa tidak terdapat hubungan linear yang kuat antar variabel independen, yang dapat menyebabkan bias pada hasil estimasi regresi. Pengujian dilakukan dengan menghitung **Variance Inflation Factor (VIF)** berdasarkan langkah-langkah sebagai berikut:

1. Menghitung matriks korelasi Pearson untuk mengidentifikasi pasangan variabel yang memiliki hubungan linear kuat secara awal.
2. Melakukan regresi linier berganda untuk setiap variabel independen terhadap variabel independen lainnya.
3. Menghitung nilai VIF menggunakan library statsmodels, dengan rumus:

$$VIF_i = \frac{1}{1 - R_i^2}$$

di mana R_i^2 adalah koefisien determinasi dari regresi variabel independen ke- i terhadap variabel independen lainnya.

Kriteria Interpretasi VIF:

1. **VIF < 5**: Tidak terdapat multikolinearitas signifikan.

2. **5 ≤ VIF ≤ 10**: Terdapat indikasi multikolinearitas moderat.
3. **VIF > 10**: Menunjukkan adanya multikolinearitas kuat yang memerlukan penanganan lebih lanjut.

Strategi Penanganan Multikolinearitas:

Jika nilai VIF menunjukkan adanya multikolinearitas, maka beberapa strategi penanganan yang dipertimbangkan dalam penelitian ini meliputi:

1. **Eliminasi variabel** dengan nilai VIF tertinggi, terutama jika variabel tersebut tidak terlalu signifikan dalam model.
2. **Transformasi variabel,** seperti melakukan standardisasi atau menerapkan teknik reduksi dimensi seperti Principal Component Analysis (PCA).
3. **Penggunaan metode regularisasi** seperti regresi Ridge (L2) atau Lasso (L1) untuk mengurangi efek multikolinearitas pada model.

3.4.5 Identifikasi Outlier

Outlier merupakan titik data yang secara signifikan berbeda dari sebagian besar data, sehingga penting diidentifikasi untuk menjaga kualitas data sebelum dilakukan pemodelan. Analisis outlier diidentifikasi menggunakan dua pendekatan, yaitu analisis visualisasi boxplot dan metode statistik Interquartile Range (IQR).

1. Metode Identifikasi Outlier
   1) Interquartile Range (IQR)
      a) Menghitung selisih antara kuartil ke-3 (Q3) dan kuartil ke-1 (Q1):

$$\mathbf{IQR} = \boldsymbol{Q_3} - \boldsymbol{Q_1}$$

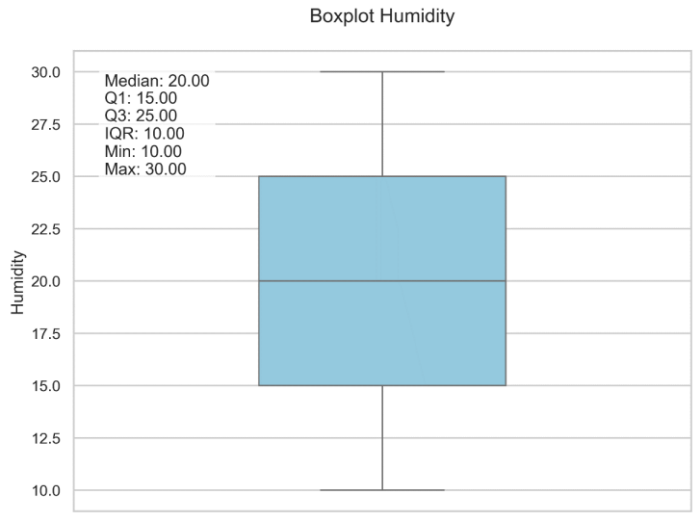
      b) Batas bawah dan atas outlier:

$$\textbf{Batas Bawah } \boldsymbol{Q_1} - \mathbf{1.5} \times \mathbf{IQR} \quad \textbf{dan} \quad \textbf{Batas Atas: } \boldsymbol{Q_3} + \mathbf{1.5} \times \mathbf{IQR}$$

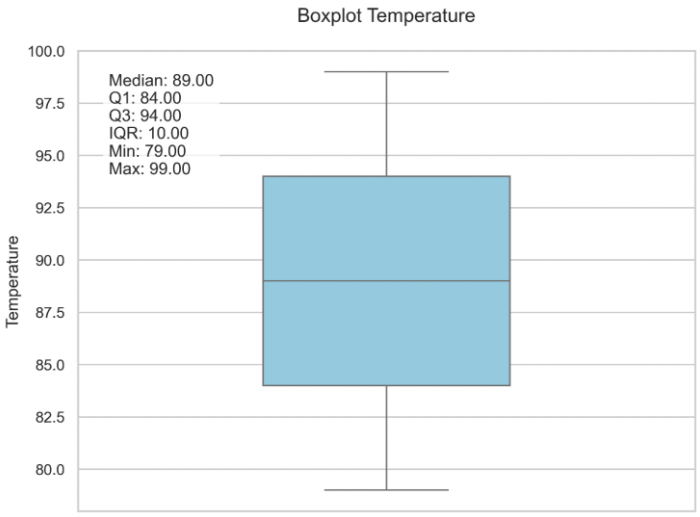
2. Visualisasi Boxplot

Boxplot berfungsi untuk mengidentifikasi nilai ekstrem yang berada di luar rentang normal (whisker) yang ditentukan berdasarkan 1.5 kali Interquartile Range (IQR) dari kuartil pertama (Q1) dan kuartil ketiga (Q3).Melalui analisis ini, diharapkan dapat diketahui apakah terdapat nilai-nilai pencilan (outlier) yang berpotensi mempengaruhi distribusi data atau performa

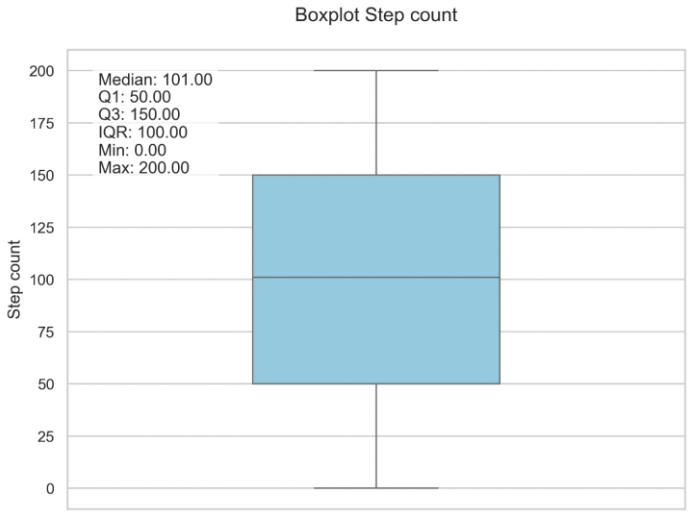
model. Jika ditemukan outlier, akan dipertimbangkan tindakan seperti penghapusan (removal) atau transformasi data pada tahap preprocessing.



Gambar 3. 12 Visualisasi Boxplot Humidity



Gambar 3. 13 Visualisasi Boxplot Temperature



Gambar 3. 14 Visualisasi Boxplot Step Count

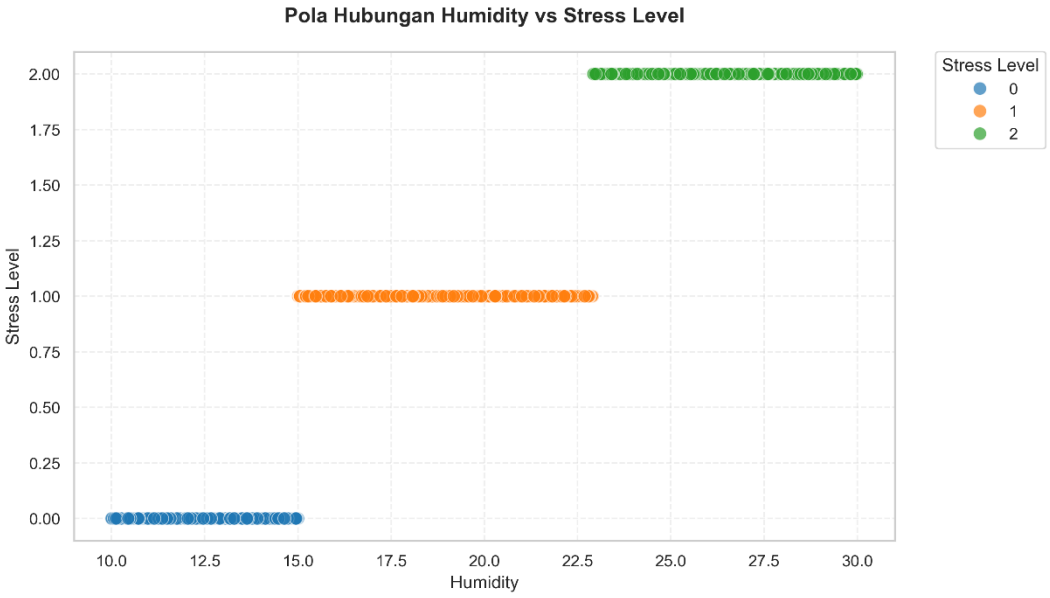
3.4.6 Pemetaan Pola Data

Pemetaan pola data bertujuan untuk memahami hubungan antar variabel secara visual dan mengidentifikasi struktur atau tren tersembunyi dalam dataset. Dalam konteks penelitian ini, fokus pemetaan dilakukan untuk melihat bagaimana Humidity, Temperature, dan Step Count berhubungan dengan Stress Level. Beberapa teknik visualisasi yang digunakan meliputi:

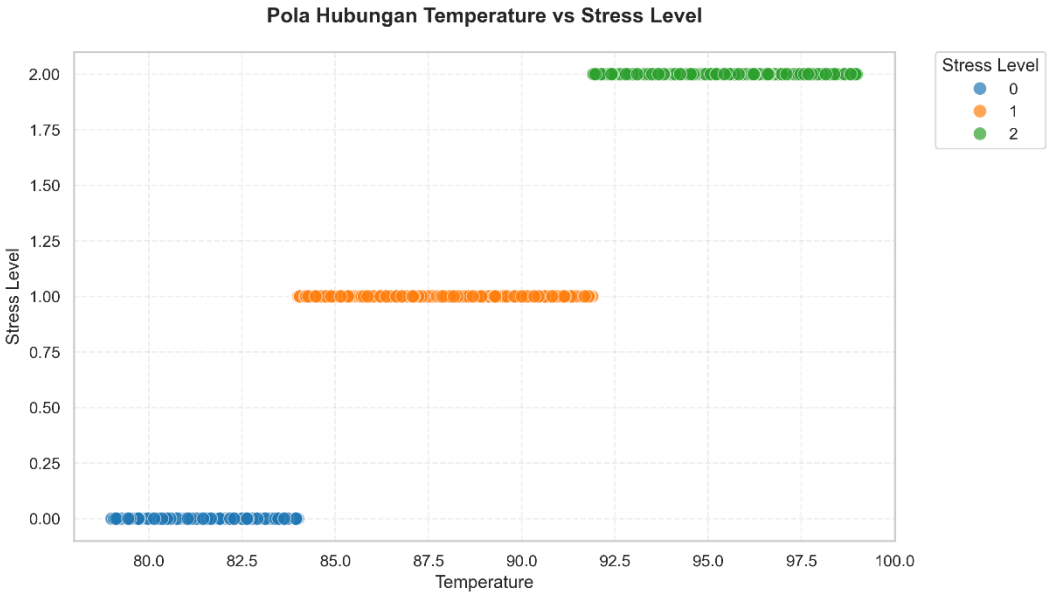
1. Scatter Plot

Scatter plot digunakan untuk melihat penyebaran data antara dua fitur sekaligus dan bagaimana mereka mempengaruhi *Stress Level*. Dalam penelitian ini, scatter plot dibuat antara:

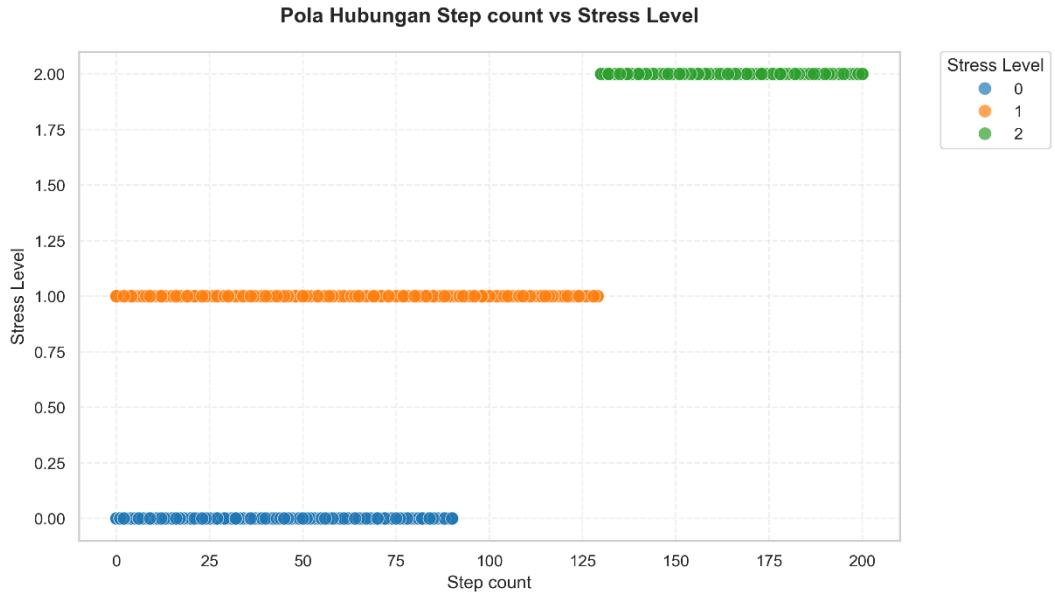
1) Humidity vs Stress Level
2) Temperature vs Stress Level
3) Step Count vs Stress Level



Gambar 3. 15 Visualisasi Scatter Plot hubungan Humidity dengan Stress Level



Gambar 3. 16 Visualisasi Scatter Plot hubungan Temperature dengan Stress Level

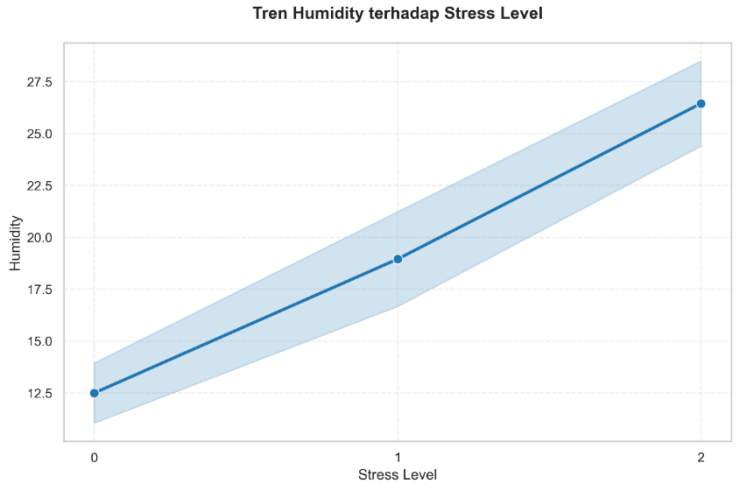


Gambar 3. 17 Visualisasi Scatter Plot hubungan Step count dengan Stress Level

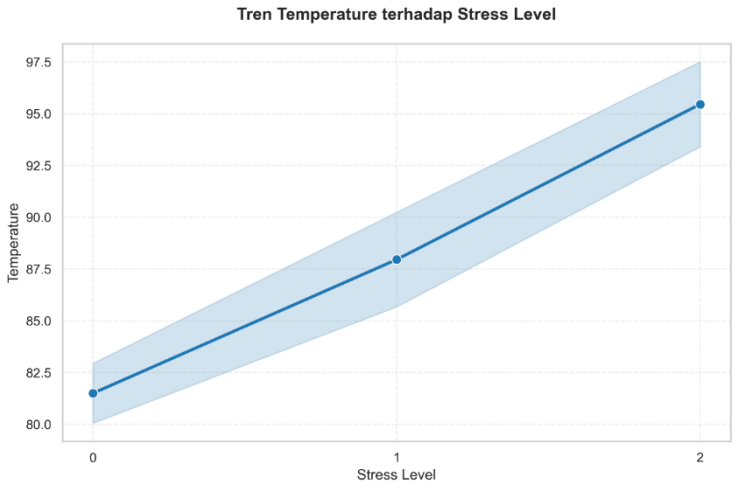
2. Line Plot

Line plot digunakan untuk melihat tren perubahan Humidity, Temperature, dan Step Count terhadap Stress Level. Dengan mengelompokkan data berdasarkan rentang suhu atau jumlah langkah, Dalam penelitian ini, scatter plot dibuat antara:

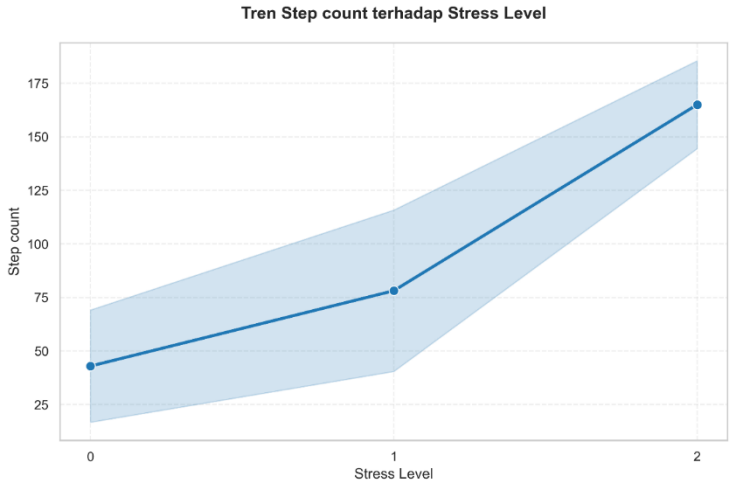
1) Humidity vs Stress Level
2) Temperature vs Stress Level
3) Step Count vs Stress Level



Gambar 3. 18 Visualisasi Line Plot Tren Humidity terhadap Stress Level



Gambar 3. 19 Visualisasi Line Plot Tren Temperature terhadap Stress Level



Gambar 3. 20 Visualisasi Line Plot Tren Step count terhadap Stress Level

3.4.7 Analisis Clustering

Untuk mengidentifikasi pola alami atau pengelompokan data berdasarkan karakteristik lingkungan, dilakukan analisis klaster menggunakan dua pendekatan berikut:

1. K-Means Clustering (k = 3)

Metode K-Means digunakan untuk mengelompokkan data menjadi tiga klaster yang diasumsikan mewakili tingkat stres Low, Normal, dan High. Pengelompokan ini didasarkan pada kombinasi fitur Humidity, Temperature, dan Step Count. Hasil klasterisasi divisualisasikan menggunakan scatter plot, yang mempermudah pemahaman distribusi data dan pemisahan antar klaster.

2. Hierarchical Clustering (Agglomerative dengan Ward Linkage)

Untuk memvalidasi hasil klasterisasi dari K-Means, diterapkan Hierarchical Clustering menggunakan metode agglomerative dengan teknik penggabungan Ward linkage. Visualisasi hasil dilakukan dalam bentuk dendrogram, yang menunjukkan proses penggabungan data secara bertahap dan mendukung identifikasi jumlah klaster optimal.

Penggunaan kedua pendekatan ini bertujuan untuk memastikan bahwa pola alami dalam data dapat diidentifikasi dengan lebih akurat serta mendukung keandalan segmentasi sebelum pemodelan prediktif dilakukan.

3.4.8 Analisis Keseimbangan Kelas

Pada tahap ini, penting untuk memeriksa distribusi kelas dalam dataset untuk memastikan bahwa model yang dikembangkan tidak akan terpengaruh oleh ketidakseimbangan kelas yang signifikan. Ketidakseimbangan kelas dapat mempengaruhi kemampuan model untuk memprediksi kelas minoritas dengan akurat.

1. Pemeriksaan Distribusi Kelas, dataset yang digunakan dalam penelitian ini memiliki variabel target Stress Level, yang terdiri dari tiga kelas:
    1) Kelas 0 (Stres Rendah)
    2) Kelas 1 (Stres Normal)
    3) Kelas 2 (Stres Tinggi)

Distribusi kelas akan dianalisis untuk memastikan apakah terdapat kelas yang mendominasi dataset atau jika distribusinya seimbang. Untuk melakukan ini, dilakukan analisis deskriptif terhadap frekuensi setiap kelas dalam variabel target.

2. Penanganan Ketidakseimbangan Kelas
    1) Jika ditemukan ketidakseimbangan yang signifikan, beberapa teknik dapat diterapkan untuk menangani masalah ini, seperti:
    2) Penyesuaian Bobot Kelas (Class Weighting): Dalam algoritma Random Forest, parameter class_weight='balanced' akan digunakan untuk memberikan bobot lebih pada kelas yang lebih jarang, sehingga model lebih memperhatikan kelas minoritas.
    3) Oversampling atau Undersampling: Jika ketidakseimbangan lebih ekstrem, teknik seperti SMOTE (Synthetic Minority Over-sampling Technique) atau pengurangan sampel kelas mayoritas dapat diterapkan untuk menyamakan distribusi kelas dalam dataset pelatihan.

3. Evaluasi Keseimbangan Kelas

Untuk mengevaluasi keseimbangan performa model pada setiap kelas, metrik seperti Precision, Recall, F1-Score, dan Confusion Matrix akan digunakan. Metrik-metrik ini memberikan gambaran yang lebih detail tentang bagaimana model memprediksi masing-masing kelas, terutama dalam kasus ketidakseimbangan kelas.

Jika kelas dalam dataset terbagi tidak merata, penggunaan weighted metrics untuk menghitung Precision, Recall, dan F1-Score akan dilakukan untuk memberikan bobot tambahan pada kelas minoritas dan memastikan bahwa model tidak hanya mengoptimalkan akurasi pada kelas mayoritas.

3.5 Data Preprocessing

Setelah melakukan ***Exploratory Data Analysis (EDA)***, tahap **Data *Preprocessing*** diperlukan untuk memastikan bahwa data yang akan digunakan dalam pemodelan **bersih, bebas dari anomali, dan hanya berisi fitur yang relevan**. Preprocessing data sangat penting karena **model *machine learning* sangat bergantung pada kualitas data.** Jika data mengandung banyak *noise*, *missing values*, atau fitur yang tidak relevan, maka model yang dihasilkan bisa menjadi tidak akurat.

Tahapan utama dalam *preprocessing* meliputi:

1. Pemeriksaan dan Penanganan Nilai Hilang (Missing Values)

Pada tahap ini, dataset akan diperiksa untuk mengetahui adanya nilai yang hilang (missing values). Nilai hilang dapat memengaruhi kualitas model dan menyebabkan distorsi pada hasil prediksi. Untuk menangani nilai hilang, beberapa metode yang umum digunakan adalah imputasi (pengisian nilai hilang) dengan menggunakan rata-rata (mean), median, modus, atau pendekatan yang lebih kompleks seperti menggunakan model prediktif. Metode imputasi yang dipilih pada penelitian ini adalah **mean imputation** untuk fitur numerik, yang dipilih karena memungkinkan penanganan nilai hilang dengan cara yang sederhana dan tidak mengubah distribusi data secara signifikan.

2. Penyamaan Format Kolom

Selama tahap ini, dilakukan pemeriksaan terhadap format kolom dalam dataset. Penyamaan format kolom penting untuk memastikan bahwa seluruh data memiliki tipe yang konsisten dan sesuai dengan kebutuhan analisis. Misalnya, kolom yang berisi data numerik yang

diperlakukan sebagai string akan diubah menjadi format numerik, dan kolom yang berisi data kategorikal akan disesuaikan dengan tipe data kategori yang sesuai.

3. Encoding Label Target

Label target pada dataset ini berupa kategori tingkat stres yang perlu dikonversi menjadi format numerik agar dapat digunakan dalam algoritma machine learning. Oleh karena itu, dilakukan encoding terhadap label target dengan menggunakan metode **Label Encoding**. Label encoding mengubah kategori menjadi angka yang merepresentasikan kelas target, sehingga memudahkan algoritma dalam memproses data tersebut.

4. Normalisasi Fitur Numerik (Standardization)

Fitur numerik dalam dataset memiliki skala yang berbeda-beda, seperti kelembaban, suhu, dan jumlah langkah. Agar model tidak bias terhadap fitur dengan skala yang lebih besar, dilakukan normalisasi menggunakan **standardization**. Metode ini mengubah nilai fitur agar memiliki distribusi dengan rata-rata 0 dan standar deviasi 1. Dengan demikian, setiap fitur memiliki skala yang setara, yang membantu meningkatkan performa model.

5. Pemisahan Fitur dan Target

Pada tahap ini, dataset dibagi menjadi dua bagian: fitur (independent variables) dan target (dependent variable). Fitur yang digunakan dalam penelitian ini terdiri dari kelembaban, suhu, dan jumlah langkah, sedangkan target yang diprediksi adalah tingkat stres manusia. Proses pemisahan ini dilakukan dengan menggunakan fungsi pemisahan data seperti `train_test_split` pada Python, yang membagi data menjadi set pelatihan dan set pengujian dengan proporsi yang telah ditentukan.

3.6 Modelling Data

Tahap Modelling Data merupakan proses utama dalam penelitian ini, di mana algoritma Random Forest digunakan untuk membangun model prediksi tingkat stres berdasarkan variabel kelembaban, suhu, dan jumlah langkah. Pada tahap ini, data yang telah melalui proses preprocessing akan dibagi menjadi data latih dan data uji sebelum dimasukkan ke dalam model.

3.6.1 Pembagian Dataset

Sebelum membangun model, dataset perlu dibagi menjadi dua bagian utama agar model dapat dilatih dan diuji secara adil, yaitu:

1. **Data Latih (Training Set):** 80% dari dataset digunakan untuk melatih model.

2. **Data Uji (Testing Set):** 20% dari dataset digunakan untuk menguji performa model terhadap data yang belum pernah dilihat sebelumnya.

Pembagian ini dilakukan menggunakan metode *train-test split*, yang memastikan model dapat melakukan generalisasi dengan baik terhadap data baru. Berikut beberapa alasan mengapa pembagian 80%-20% sangat efektif:

1. Keseimbangan Antara Pelatihan dan Evaluasi

Pembagian ini memberikan jumlah data yang cukup untuk model belajar, sambil tetap memastikan evaluasi yang akurat. Jika terlalu banyak data digunakan untuk pelatihan, data uji menjadi tidak representatif, sedangkan jika terlalu sedikit data digunakan untuk pelatihan, model tidak mendapatkan cukup informasi untuk belajar.

2. Generalization Performance (Kemampuan Generalisasi)

Model machine learning harus mampu menggeneralisasi pola dari data latih ke data uji yang belum pernah dilihat sebelumnya. Pembagian 80%-20% memastikan bahwa model dapat belajar dengan cukup banyak data sementara juga diuji pada data yang dapat mewakili data yang tidak pernah dilihat sebelumnya.

3. Menghindari Overfitting dan Underfitting

Pembagian ini membantu menghindari masalah overfitting (model terlalu mengingat data latih) dan underfitting (model tidak belajar dengan baik dari data latih), yang bisa terjadi jika dataset terlalu kecil untuk pelatihan atau pengujian.

4. Praktik Terbaik dalam Industri dan Riset:

Pembagian 80%-20% adalah standar yang telah terbukti dalam berbagai eksperimen dan aplikasi, memberikan hasil yang konsisten dan stabil. Meskipun dalam kasus dataset yang sangat besar, rasio ini bisa sedikit disesuaikan, tetapi 80%-20% tetap menjadi pilihan utama karena konsistensinya.

5. Representasi yang Cukup untuk Evaluasi:

Dengan 20% data uji, kita masih mendapatkan gambaran yang representatif mengenai kinerja model, dan memastikan bahwa variabilitas dalam hasil pengujian tidak terlalu tinggi.

3.6.2 Penerapan Algoritma Random Forest

Algoritma Random Forest dipilih karena kemampuannya dalam menangani dataset dengan banyak variabel, mencegah overfitting, dan memberikan interpretasi melalui feature importance. Langkah-langkah implementasi Random Forest adalah sebagai berikut:

Menentukan Parameter Model:

1. n_estimators = 100: Jumlah pohon keputusan yang dibangun dalam Random Forest. Nilai ini umumnya digunakan dan memberikan stabilitas prediksi pada dataset berukuran sedang.
2. max_depth = None: Tidak ada batasan kedalaman pohon, sehingga setiap pohon akan tumbuh hingga seluruh daun murni atau hingga jumlah sampel pada node lebih kecil dari min_samples_split.
3. min_samples_split = 2: Minimum jumlah sampel yang dibutuhkan untuk memisahkan node internal.
4. min_samples_leaf = 1: Minimum jumlah sampel pada setiap daun pohon.
5. max_features = 'sqrt': Pada setiap split, hanya akar kuadrat dari jumlah total fitur yang dipertimbangkan untuk pemilihan split terbaik (standar untuk klasifikasi).
6. bootstrap = True: Mengaktifkan teknik bagging dengan pengambilan sampel acak (dengan pengembalian) untuk membangun setiap pohon.
7. random_state = 42: Digunakan untuk memastikan bahwa hasil eksperimen dapat direplikasi dengan konsisten.

Pada tahap awal, model akan dilatih menggunakan parameter default untuk menilai kinerja model. Jika diperlukan, eksperimen lebih lanjut akan dilakukan menggunakan GridSearchCV atau RandomizedSearchCV untuk mengeksplorasi pengaruh parameter terhadap akurasi model.

3.6.3 Pelatihan Model

1. Model dilatih menggunakan data latih dengan parameter yang telah ditentukan.
2. Proses pelatihan ini melibatkan pembuatan banyak pohon keputusan yang bekerja secara kolektif untuk menghasilkan prediksi terbaik.
3. Setelah model dilatih, dilakukan prediksi terhadap data uji.
4. Hasil prediksi dibandingkan dengan data aktual untuk mengukur kinerja model.

3.6.4 Cross-Validation

Untuk memastikan bahwa model Random Forest yang dikembangkan tidak hanya bekerja baik pada subset data tertentu, digunakan teknik k-fold cross-validation dengan nilai k = 5. Dalam metode ini, dataset dibagi menjadi lima bagian (fold) yang sama besar. Proses

pelatihan dan evaluasi dilakukan sebanyak lima kali, di mana setiap fold bergantian digunakan sebagai data uji, dan empat lainnya sebagai data latih.

Pemilihan 5-fold dibanding 10-fold didasarkan pada pertimbangan komputasi dan stabilitas hasil. Pada dataset dengan ukuran sedang 2001, 5-fold memberikan keseimbangan antara:

1) Waktu komputasi yang lebih efisien,
2) Ukuran subset validasi yang cukup besar (≈400 sampel per fold) untuk mengurangi varians evaluasi,

Hasil evaluasi dari kelima iterasi tersebut kemudian dirata-rata untuk memperoleh estimasi performa model yang lebih stabil dan bebas dari bias terhadap subset data tertentu. Metode ini membantu menghindari masalah overfitting dan memberikan gambaran yang lebih umum terhadap performa model pada data yang belum pernah dilihat sebelumnya.

3.6.5 Evaluasi Model

Setelah model Random Forest dilatih menggunakan data latih, tahap selanjutnya adalah melakukan evaluasi untuk mengukur kinerja model dalam memprediksi tingkat stres berdasarkan variabel kelembaban, suhu, dan jumlah langkah.Untuk menilai performa model, beberapa metrik evaluasi utama yang digunakan dalam penelitian ini adalah sebagai berikut:

1. Accuracy

Akurasi adalah **persentase prediksi yang benar dibandingkan dengan total data uji.**
Rumus *Accuracy*:

$$\boldsymbol{Accuracy = \frac{TP + TN}{TP + TN + FP + FN}}$$

Keterangan:

TP (True Positive): Prediksi stres benar

TN (True Negative): Prediksi tidak stres benar

FP (False Positive): Prediksi stres tetapi sebenarnya tidak

FN (False Negative): Prediksi tidak stres tetapi sebenarnya iya

2. Precision

Precision untuk Mengukur seberapa banyak prediksi positif yang benar dibandingkan dengan total prediksi positif.

Rumus *Precision*:

$$\boldsymbol{Precision = \frac{TP}{TP + FP}}$$

3. ***Recall***

Recall mengukur kemampuan model dalam menemukan semua kasus positif yang sebenarnya ada.

Rumus *Recall*:

$$\boldsymbol{Recall = \frac{TP}{TP + FN}}$$

4. ***F1-Score***

F1-Score merupakan Kombinasi *Precision* dan *Recall* untuk keseimbangan antara keduanya.

Rumus *F1-Score*:

$$\boldsymbol{F1 - Score = \frac{Precision \times Recall}{Precision + Recall}}$$

Jika model memiliki ketidakseimbangan kelas (misalnya, lebih banyak data dengan stres rendah dibanding stres tinggi), maka ***F1-Score*** **lebih baik digunakan daripada akurasi.**

5. Confusion Matrix

Matriks ini menunjukkan jumlah prediksi yang benar dan salah dalam setiap kategori tingkat stres. Confusion matrix membantu memvisualisasikan distribusi **TP, TN, FP,** dan **FN** untuk memahami kinerja model lebih mendalam.

6. Penanganan Ketidakseimbangan Kelas

Jika ditemukan ketidakseimbangan kelas yang signifikan, teknik penyesuaian bobot kelas akan diterapkan pada model menggunakan parameter `class_weight ='balanced'` pada algoritma Random Forest. Penyesuaian ini akan memberikan bobot lebih pada kelas minoritas, sehingga model tidak akan bias terhadap kelas mayoritas. Selain itu, metrik seperti **weighted F1-score** dan **macro average** akan digunakan untuk memastikan bahwa performa model diperhitungkan secara proporsional, tanpa mengabaikan kelas yang lebih sedikit jumlahnya dalam dataset.

7. Evaluasi Kemampuan Generalisasi Model

Untuk menilai kemampuan generalisasi model, dilakukan perbandingan antara performa model pada data latih dan data uji. Jika model menunjukkan performa yang jauh lebih baik pada data latih dibandingkan dengan data uji, ini bisa menunjukkan adanya **overfitting**. Sebaliknya, jika performa pada kedua data rendah, model mungkin mengalami **underfitting.** Berdasarkan hasil evaluasi, strategi lebih lanjut seperti **regularisasi** atau **hyperparameter tuning** mungkin diperlukan untuk meningkatkan kinerja model.

3.6.6 Analisis Hasil Evaluasi

Setelah mendapatkan hasil dari metrik evaluasi, analisis dilakukan untuk menentukan apakah model sudah bekerja dengan baik atau masih perlu perbaikan. Beberapa hal yang perlu diperhatikan dalam analisis hasil evaluasi adalah sebagai berikut:

1. Akurasi Tinggi tetapi F1-Score Rendah

Jika akurasi tinggi tetapi F1-score rendah, ini menunjukkan kemungkinan adanya ketidakseimbangan kelas dalam data. Hal ini dapat menyebabkan model terlalu fokus pada kelas mayoritas dan mengabaikan kelas minoritas, sehingga menyebabkan rendahnya kemampuan model dalam mendeteksi kelas yang lebih jarang. Dalam kasus seperti ini, metrik seperti F1-score, Recall, dan Precision akan lebih memberikan gambaran yang lebih adil tentang kinerja model, terutama ketika data tidak seimbang.

2. Recall Rendah

Jika Recall rendah, ini menunjukkan bahwa model melewakan banyak kasus positif (misalnya, banyak sampel stres yang sebenarnya ada tidak terdeteksi). Hal ini dapat terjadi jika model lebih mengutamakan Precision dan tidak cukup sensitif dalam mendeteksi seluruh sampel yang seharusnya diprediksi positif. Dalam kasus ini, mungkin perlu dilakukan penyesuaian pada threshold klasifikasi atau penambahan bobot pada kelas positif agar model lebih fokus pada mendeteksi kasus positif.

3. Precision Rendah

Jika Precision rendah, ini berarti model sering membuat kesalahan dengan mengklasifikasikan sampel yang tidak stres sebagai stres. Dalam hal ini, model menghasilkan banyak False Positive. Hal ini menunjukkan bahwa model mungkin terlalu sensitif dalam memprediksi kelas positif, atau ada ketidakseimbangan kelas yang mengarah pada banyak kesalahan deteksi. Untuk memperbaiki ini, Anda mungkin perlu menyesuaikan bobot kelas atau menggunakan penyaringan threshold yang lebih ketat.

4. Overfitting

Jika model menunjukkan akurasi sangat tinggi pada data pelatihan tetapi rendah pada data uji, ini menandakan bahwa model mengalami overfitting, yaitu model terlalu mengingat detail dari data pelatihan dan tidak dapat menggeneralisasi dengan baik pada data yang belum dilihat sebelumnya. Untuk mengatasi overfitting, beberapa pendekatan dapat dilakukan, seperti:

1) Pengurangan kompleksitas model: Mengurangi jumlah pohon (estimators) atau kedalaman pohon (depth) dalam Random Forest.
2) Penyesuaian parameter: Melakukan hyperparameter tuning untuk menemukan parameter yang lebih optimal dan mengurangi overfitting.
3) Regularisasi: Menggunakan teknik regularisasi untuk membatasi kompleksitas model.

3.6.7 ROC Curve dan AUC Score

1. *Receiver Operating Characteristic (ROC)*

ROC Curve adalah grafik yang menggambarkan hubungan antara ***True Positive Rate (TPR)*** dan ***False Positive Rate (FPR)*** pada berbagai *threshold* klasifikasi. Grafik ini membantu dalam menganalisis *trade-off* antara sensitivitas (kemampuan model mendeteksi kelas positif) dan spesifisitas (kemampuan model menghindari kesalahan deteksi kelas negatif).

1) Rumus *True Positive Rate* :

$$\boldsymbol{TPR = \frac{TP}{TP + FN}}$$

Mewakili proporsi sampel positif yang berhasil diklasifikasikan dengan benar.

2) Rumus *False Positive Rate* :

$$\boldsymbol{FPR = \frac{FP}{FP + TN}}$$

Mengukur proporsi sampel negatif yang salah diklasifikasikan sebagai positif.

Dengan memplot nilai *TPR* terhadap *FPR* pada berbagai *threshold*, kita mendapatkan *ROC Curve* yang menunjukkan bagaimana performa model berubah tergantung pada ambang batas probabilitas yang digunakan untuk klasifikasi.

2. *AUC (Area Under Curve)*

AUC adalah ukuran numerik yang menghitung luas di bawah *ROC Curve*. Nilai *AUC* berkisar antara 0 hingga 1.

1) ***AUC* = 1** menunjukkan model sempurna yang mampu membedakan kelas positif dan negatif tanpa kesalahan.
2) ***AUC* = 0.5** menunjukkan model tidak lebih baik dari tebakkan acak.
3) ***AUC* < 0.5** menunjukkan model berkinerja lebih buruk daripada tebakan acak.

Semakin tinggi nilai *AUC*, semakin baik model dalam membedakan antara kelas yang berbeda. *AUC* juga dapat digunakan untuk membandingkan beberapa model guna menentukan mana yang memiliki performa terbaik.

3.6.8 Hyperparameter Tuning

Jika hasil evaluasi menunjukkan bahwa model belum optimal, maka dilakukan tuning parameter untuk mencari kombinasi parameter terbaik yang dapat meningkatkan kinerja model. Beberapa teknik yang digunakan dalam hyperparameter tuning adalah sebagai berikut:

1. Grid Search CV:

Grid Search CV mencari kombinasi parameter terbaik dengan menguji semua kemungkinan nilai parameter secara exhaustif. Meskipun teknik ini memberikan hasil yang lebih lengkap, Grid Search cenderung memakan waktu lebih lama karena memeriksa setiap kombinasi dari parameter yang diberikan.

2. Randomized Search CV:

Randomized Search CV mencari kombinasi parameter terbaik dengan memilih nilai parameter secara acak dari ruang parameter yang telah ditentukan. Teknik ini lebih efisien karena tidak menguji semua kemungkinan, tetapi memberikan hasil yang hampir sebanding dengan Grid Search CV dalam waktu yang lebih singkat.

3. Rencana Eksperimen Tuning:

Pada tahap eksperimen tuning parameter, beberapa parameter utama yang akan diuji menggunakan GridSearchCV dan RandomizedSearchCV adalah:

1) n_estimators (jumlah pohon dalam Random Forest),
2) max_depth (kedalaman maksimal pohon),
3) max_features (jumlah fitur yang dipertimbangkan pada setiap pemisahan).

Tujuan dari eksperimen ini adalah untuk menemukan kombinasi parameter yang dapat meningkatkan akurasi model, sambil mempertimbangkan waktu pelatihan yang dibutuhkan.

Jika eksperimen ini menunjukkan peningkatan yang signifikan dalam akurasi tanpa menambah waktu pelatihan secara drastis, maka parameter tersebut akan digunakan dalam model akhir. Namun, jika peningkatan akurasi tidak signifikan atau memakan waktu pelatihan yang terlalu lama, parameter default akan tetap dipilih.

3.6.9 Library dan Tools yang Digunakan

Seluruh proses preprocessing, pemodelan, dan evaluasi pada penelitian ini dilakukan menggunakan bahasa pemrograman Python dengan bantuan beberapa pustaka (library) populer dalam pengolahan data dan machine learning. Pustaka-pustaka yang digunakan antara lain:

1. Pandas

Digunakan untuk manipulasi dan eksplorasi data, seperti pengolahan data dalam bentuk DataFrame dan operasi terkait (misalnya, filter, agregasi, dan manipulasi kolom).

2. numpy

Mendukung operasi numerik dan array multidimensi. NumPy digunakan untuk melakukan operasi matematis dan aljabar linear, serta menyediakan struktur data array yang efisien.

3. matplotlib dan seaborn

Kedua pustaka ini digunakan untuk visualisasi data, seperti membuat histogram, boxplot, pairplot, dan heatmap.

   1) matplotlib digunakan untuk visualisasi dasar dan customisasi grafik.
   2) seaborn digunakan untuk visualisasi statistik yang lebih mudah dengan sintaks yang lebih ringkas.

4. scikit-learn

Digunakan untuk implementasi model Random Forest, preprocessing data (misalnya, menggunakan StandardScaler untuk normalisasi), evaluasi model (confusion matrix, classification report, accuracy score, dll.), serta metode validasi seperti train-test split dan k-fold cross-validation.

Pemrograman dilakukan menggunakan Jupyter Notebook dan Google Colab sebagai environment karena kedua platform ini mendukung eksplorasi data secara interaktif serta mempermudah visualisasi dan dokumentasi hasil penelitian.