Sistem Deteksi Dini Penurunan Performa Belajar Siswa Berdasarkan Analisis Kualitas Infrastruktur Dan Kapasistas Pengajar

Dosen Pengampuh RUNAL REZKIAWAN, S.Kom.,M.T

Oleh:

Nama : ALDI

Kelas : 5AI-A

PROGRAM STUDI INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS MUHAMMADIYAH MAKASSAR

2025

BAB 1 BUSSINES UNDERSTANDING (PEMAHAMAN BISNIS)

1.1 Latar Belakang

Pendidikan merupakan pilar utama dalam peningkatan kualitas sumber daya manusia. Di Indonesia, standar kualitas pendidikan telah diatur dalam Undang-Undang No. 20 Tahun 2003, yang mewajibkan pemerintah dan penyelenggara pendidikan untuk menyediakan sarana prasarana serta tenaga pendidik yang layak [1]. Namun, realita di lapangan menunjukkan adanya disparitas mutu yang signifikan antar wilayah, khususnya di tingkat Sekolah Dasar (SD).

Salah satu indikator utama penurunan mutu pendidikan adalah menurunnya performa belajar siswa. Studi menunjukkan bahwa performa siswa tidak berdiri sendiri, melainkan sangat dipengaruhi oleh ekosistem sekolah. Studi dari Hanushek (2011) menunjukkan bahwa kualitas guru dan rasio siswa-guru memiliki korelasi langsung terhadap capaian akademik siswa [2]. Selain itu, kondisi fisik lingkungan belajar, seperti ruang kelas yang rusak, terbukti menurunkan motivasi dan konsentrasi belajar siswa secara signifikan [3].

Sayangnya, pemantauan terhadap kondisi fisik dan kapasitas guru seringkali dilakukan secara manual dan birokratis. Dinas Pendidikan kerap terlambat menyadari adanya sekolah yang mengalami "krisis" (kekurangan guru akut atau kerusakan berat) hingga akhirnya berdampak pada anjloknya nilai akreditasi atau prestasi siswa. Oleh karena itu, diperlukan pendekatan berbasis data (*Data Driven*) untuk mendeteksi masalah ini lebih awal.

Pemanfaatan teknologi *Machine Learning*, khususnya algoritma *Random Forest*, telah terbukti efektif dalam membangun sistem peringatan dini (*Early Warning System*) di bidang pendidikan. Metode ini mampu mengolah data historis profil sekolah untuk memprediksi risiko di masa depan dengan akurasi tinggi [4]. Dengan memanfaatkan data Dapodik (Data Pokok Pendidikan), sistem ini dapat membantu pemangku kebijakan untuk memprioritaskan intervensi ke sekolah yang paling membutuhkan sebelum penurunan mutu terjadi secara permanen [5].

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, maka rumusan masalah dalam penelitian ini adalah:

a. Bagaimana cara mengintegrasikan dan melakukan pra-pemrosesan data profil sekolah (Siswa, Guru, dan Fasilitas) yang bersumber dari *file* terpisah menjadi dataset yang siap untuk pemodelan *Machine Learning*?
b. Bagaimana kinerja algoritma *Random Forest* dalam mengklasifikasikan status risiko sekolah (AMAN atau WASPADA) berdasarkan indikator rasio guru dan kondisi infrastruktur?
c. Bagaimana merancang implementasi sistem deteksi dini berbasis *web* yang dapat memberikan informasi status dan rekomendasi manajerial secara mudah kepada pengguna?

1.3 Tujuan Proyek

a. Menghasilkan dataset terintegrasi yang bersih dari *noise* dan siap digunakan untuk analisis data.
b. Membangun model *Machine Learning* dengan algoritma *Random Forest* yang memiliki tingkat akurasi tinggi dalam memprediksi penurunan mutu sekolah.
c. Menyediakan aplikasi antarmuka (*User Interface*) berbasis web yang dapat digunakan oleh pihak sekolah atau dinas terkait untuk simulasi deteksi dini secara *real-time*.

BAB 2. DATA UNDERSTANDING (PEMAHAMAN DATA)

2.1 Sumber Data

Data yang digunakan adalah data sekunder publik dari Portal Data Kemendikdasmen (https://data.kemendikdasmen.go.id/dataset/pendidikan-2) tahun ajaran terbaru untuk jenjang SD (Sekolah Dasar) di wilayah Kabupaten Gowa. Dataset terdiri dari tiga file terpisah:

- Data Siswa: jumlah-siswa-menurut-kelompok-umur...xlsx (Berisi jumlah peserta didik).
- Data Guru: jumlah-kepala-sekolah-dan-guru...xlsx (Berisi jumlah tenaga pengajar).
- Data Fasilitas: jumlah-ruang-kelas-menurut-kondisi...xlsx (Berisi kondisi fisik ruang kelas).

2.2 Jenis Data

Dataset ini terdiri dari beberapa jenis data, yaitu:

a. Data Kategorikal (Nominal):
   - Kecamatan: Nama wilayah kecamatan (contoh: Bontonompo, Pallangga).
   - Status: Status kepemilikan sekolah (Negeri / Swasta).
   - Prediksi_Status: Label target hasil klasifikasi ("AMAN" / "WASPADA").

b. Data Numerik (Kontinu & Diskrit):
   - Total_Siswa: Jumlah siswa (Integer).
   - Total_Guru: Jumlah guru (Integer).
   - Total_Kelas: Jumlah total ruang kelas (Integer).
   - Kelas_Rusak: Jumlah ruang kelas yang rusak (Integer).
   - Rasio_Siswa_Guru: Hasil pembagian siswa dengan guru (Float/Desimal).
   - Persentase_Kerusakan: Persentase kerusakan kelas (Float/Desimal).

2.3 Deskripsi Fitur dan Target

Dalam proses *Machine Learning*, data dibagi menjadi dua peran utama:

a. Fitur (*Features* / Variabel Input X) Fitur adalah atribut yang digunakan oleh model untuk mempelajari pola risiko penurunan mutu sekolah. Fitur yang dipilih meliputi:

- Total_Siswa: Menggambarkan beban populasi murid yang harus dilayani.
- Total_Guru: Menggambarkan kapasitas sumber daya manusia (SDM) pengajar.
- Total_Kelas: Menggambarkan kapasitas daya tampung infrastruktur sekolah.
- Kelas_Rusak: Menggambarkan kualitas fisik lingkungan belajar (kenyamanan dan keamanan).

b. Target (*Label* / Variabel Output y) Target adalah variabel yang ingin diprediksi oleh sistem.

   Prediksi_Status:

   - Merupakan label klasifikasi biner.
   - Bernilai "WASPADA (Berisiko)" jika sekolah memiliki rasio guru yang sangat timpang (1 guru mengajar > 28 siswa) atau kondisi kerusakan fisik yang parah (> 30%).
   - Bernilai "AMAN" jika indikator sekolah masih dalam batas wajar standar pelayanan minimal.

BAB 3. DATA PREPARATION (PERSIAPAN DATA)

Tahap persiapan data (*Data Preparation*) merupakan langkah krusial untuk memastikan data mentah siap diproses oleh algoritma *Machine Learning*. Pada tahap ini, dilakukan Tiga proses utama yaitu Penggabungan Data pembersihan data (*data cleaning*) dan transformasi data.

3.1 Penggabungan Data

Data sumber yang diperoleh dari portal Kemendikbud terpisah menjadi tiga *file* Excel yang berbeda, yaitu data Siswa, data Guru, dan data Fasilitas (Ruang Kelas). Untuk melakukan analisis secara menyeluruh, ketiga data tersebut perlu digabungkan menjadi satu kesatuan dataset (*merged dataset*).

Proses penggabungan dilakukan menggunakan fungsi merge pada library Pandas dengan metode Inner Join. Kunci utama (*primary key*) yang digunakan untuk mencocokkan data antar-tabel adalah kolom "Kecamatan" dan "Status Sekolah". Hal ini memastikan bahwa data siswa, guru, dan kelas yang digabungkan benar-benar berasal dari wilayah dan jenis sekolah yang sama.

Berikut adalah implementasi kode untuk proses penggabungan data:

```python
import pandas as pd
import os

print("--- MENCARI FILE & MEMBACA SECARA OTOMATIS ---")

# 1. Cari file berdasarkan kata kunci
files = os.listdir()
file_siswa = next((f for f in files if "siswa" in f.lower() and "sd" in f.lower() and ".xlsx" in f), None)
file_guru = next((f for f in files if "guru" in f.lower() and "sd" in f.lower() and ".xlsx" in f), None)
file_kelas = next((f for f in files if "kelas" in f.lower() and "sd" in f.lower() and ".xlsx" in f), None)

if None in [file_siswa, file_guru, file_kelas]:
    print("❌ ERROR: File Excel (.xlsx) belum ditemukan.")
    print("Pastikan Anda meng-upload file ASLI dari Kemdikbud (format .xlsx), bukan hasil convert.")
else:
    print(f"✅ FILE EXCEL DITEMUKAN:\n 1. {file_siswa}\n 2. {file_guru}\n 3. {file_kelas}")

    # 2. Baca File Excel (Gunakan read_excel, BUKAN read_csv)
    df_siswa = pd.read_excel(file_siswa, header=2)
    df_guru = pd.read_excel(file_guru, header=2)
    df_kelas = pd.read_excel(file_kelas, header=2)

    # 3. Bersihkan Nama Kolom (Hapus spasi yang tidak perlu)
    df_siswa.columns = df_siswa.columns.str.strip()
    df_guru.columns = df_guru.columns.str.strip()
    df_kelas.columns = df_kelas.columns.str.strip()

    # 4. Gabungkan Data (Merge)
    df_gabung = pd.merge(df_siswa, df_guru, on=['Kecamatan', 'Status'], how='inner', suffixes=('_Siswa', '_Guru'))
    df_final = pd.merge(df_gabung, df_kelas, on=['Kecamatan', 'Status'], how='inner')

    # 5. Pilih & Rapikan Kolom Penting
    df_clean = pd.DataFrame()
    df_clean['Lokasi'] = df_final['Kecamatan'] + " (" + df_final['Status'] + ")"

    # Biasanya: Jumlah_Siswa (dari suffix tadi), Jumlah_Guru, dan Jumlah (dari file kelas)
    df_clean['Total_Siswa'] = df_final['Jumlah_Siswa']
    df_clean['Total_Guru'] = df_final['Jumlah_Guru']

    # Kolom kelas biasanya bernama 'Jumlah' (karena dia file terakhir yang di-merge)
    if 'Jumlah' in df_final.columns:
        df_clean['Total_Kelas'] = df_final['Jumlah']
    else:
        # Jika tidak ada, cari kolom lain yg mengandung 'Jumlah'
        col_kelas = [c for c in df_final.columns if 'Jumlah' in c and c not in ['Jumlah_Siswa', 'Jumlah_Guru']][-1]
        df_clean['Total_Kelas'] = df_final[col_kelas]

    # Hitung Kelas Rusak
    df_clean['Kelas_Rusak'] = df_final['Rusak Sedang'] + df_final['Rusak Berat']

    # 6. Buat Target Label Otomatis (Sistem Deteksi Dini)
    def cek_status(row):
        if row['Total_Guru'] == 0: return 'Data Error'

        rasio_guru = row['Total_Siswa'] / row['Total_Guru']
        persen_rusak = (row['Kelas_Rusak'] / row['Total_Kelas']) * 100 if row['Total_Kelas'] > 0 else 0

        # Logika: Jika 1 Guru pegang > 28 murid ATAU Kerusakan > 30% -> WASPADA
        if rasio_guru > 28 or persen_rusak > 30:
            return 'WASPADA (Berisiko)'
        return 'AMAN'

    df_clean['Prediksi_Status'] = df_clean.apply(cek_status, axis=1)

    print("\n--- DATASET FINAL SIAP SCREENSHOT ---")
    display(df_clean)
```

Output

```
--- MENCARI FILE & MEMBACA SECARA OTOMATIS ---
✅ FILE EXCEL DITEMUKAN:
 1. jumlah-siswa-menurut-kelompok-umur-tiap-provinsi-kab-gowa-sd-2022.xlsx
 2. jumlah-kepala-sekolah-dan-guru-menurut-golongan-tiap-provinsi-kab-gowa-sd-2021.xlsx
 3. jumlah-ruang-kelas-menurut-kondisi-tiap-propinsi-kab-gowa-sd-2024.xlsx

--- DATASET FINAL SIAP SCREENSHOT ---
```

	Lokasi	Total_Siswa	Total_Guru	Total_Kelas	Kelas_Rusak	Prediksi_Status
0	Kec. Bontonompo (Negeri)	4157.0	283.0	207.0	97.0	WASPADA (Berisiko)
1	Kec. Bajeng (Negeri)	5897.0	373.0	240.0	111.0	WASPADA (Berisiko)
2	Kec. Pallangga (Negeri)	12095.0	534.0	296.0	99.0	WASPADA (Berisiko)
3	Kec. Somba Opu (Negeri)	13180.0	635.0	280.0	111.0	WASPADA (Berisiko)
4	Kec. Bontomarannu (Negeri)	4801.0	237.0	138.0	32.0	AMAN
5	Kec. Parang Loe (Negeri)	1874.0	159.0	124.0	48.0	WASPADA (Berisiko)
6	Kec. Tinggi Moncong (Negeri)	2484.0	182.0	149.0	64.0	WASPADA (Berisiko)
7	Kec. Bungaya (Negeri)	1336.0	159.0	112.0	55.0	WASPADA (Berisiko)
8	Kec. Tompobulu (Negeri)	2601.0	216.0	140.0	37.0	AMAN
9	Kec. Barombong (Negeri)	4074.0	179.0	114.0	58.0	WASPADA (Berisiko)
10	Kec. Biring Bulu (Negeri)	2598.0	242.0	157.0	104.0	WASPADA (Berisiko)

3.2 Pembersihan Data (Data Cleaning)

Tujuan dari pembersihan data adalah menghilangkan gangguan (*noise*) yang dapat menurunkan akurasi model. Langkah-langkah yang dilakukan meliputi:

a. Penanganan Data Duplikat: Memeriksa dan menghapus baris data yang memiliki nilai identik (ganda) untuk menjaga integritas analisis.
b. Penanganan Nilai Kosong (*Missing Values*): Menghapus baris data yang memiliki nilai NaN atau kosong. Hal ini penting karena algoritma *Random Forest* tidak dapat memproses data yang tidak lengkap.
c. Penanganan *Outlier*: Membersihkan data pencilan yang tidak masuk akal, seperti data sekolah yang tercatat memiliki jumlah siswa 0 atau negatif, yang kemungkinan disebabkan oleh kesalahan input.

implementasi kode untuk tahap pembersihan data:

```
# 1. Load Data
df = pd.read_csv(filename)

# 2. Cleaning (Hapus baris kosong/NaN jika ada)
print(f"Jumlah data awal: {len(df)}")
df = df.dropna()
print(f"Jumlah data bersih: {len(df)}")
```

Outout

```
✅ Library berhasil di-install dan di-import!
Jumlah data awal: 25
Jumlah data bersih: 25

✅ Data berhasil disiapkan.
Data Latih: 20 baris
Data Uji  : 5 baris
```

3.3 Transformasi Data (Data Transformation)

Setelah data dibersihkan, dilakukan tahap transformasi agar format data sesuai dengan kebutuhan algoritma *Machine Learning*.

implementasi kode untuk tahap transformasi data:

```python
# 1. Encoding (Target jadi Angka)
le = LabelEncoder()
df['Prediksi_Status'] = le.fit_transform(df['Prediksi_Status'])

# --- TAMBAHAN AGAR MUNCUL DI LAYAR ---
print("✅ Hasil Encoding (Target menjadi 0/1):")
print(df[['Prediksi_Status']].head()) # Menampilkan 5 baris kolom target saja
print("-" * 30)

# 2. Normalisasi (Angka jadi 0-1)
scaler = MinMaxScaler()
kolom_angka = ['Total_Siswa', 'Total_Guru', 'Total_Kelas', 'Kelas_Rusak']
df[kolom_angka] = scaler.fit_transform(df[kolom_angka])

# --- TAMBAHAN AGAR MUNCUL DI LAYAR ---
print("✅ Hasil Normalisasi (Angka menjadi desimal 0-1):")
display(df[kolom_angka].head()) # Menampilkan tabel angka yang sudah berubah
print(f"Jumlah data bersih: {len(df)}")
```

Output

```
✅ Hasil Encoding (Target menjadi 0/1):
   Prediksi_Status
0                1
1                1
2                1
3                1
4                0
------------------------------
✅ Hasil Normalisasi (Angka menjadi desimal 0-1):
```

	Total_Siswa	Total_Guru	Total_Kelas	Kelas_Rusak
0	0.313788	0.439490	0.696246	0.873874
1	0.446118	0.582803	0.808874	1.000000
2	0.917484	0.839172	1.000000	0.891892
3	1.000000	1.000000	0.945392	1.000000
4	0.362765	0.366242	0.460751	0.288288

```
Jumlah data bersih: 25
```

Tahap ini terdiri dari dua proses utama:

a. Encoding Data Kategorikal: Kolom target Prediksi_Status berisi data teks ("AMAN" dan "WASPADA"). Komputer tidak dapat memproses teks secara langsung, sehingga dilakukan *Label Encoding* untuk mengubah data tersebut menjadi format numerik:

- AMAN diubah menjadi 0
- WASPADA diubah menjadi 1

b. Normalisasi Data (*Normalization*): Fitur-fitur numerik seperti Total_Siswa, Total_Guru, Total_Kelas, dan Kelas_Rusak memiliki rentang nilai yang sangat berbeda (ratusan vs puluhan). Untuk mencegah bias pada model, dilakukan normalisasi menggunakan metode Min-Max Scaler. Metode ini mengubah semua nilai numerik ke dalam rentang skala 0 sampai 1.

3.4 Pembagian Data

Sebelum proses pelatihan, dataset dibagi menjadi dua komponen utama:

- Fitur (X): Variabel input (Total Siswa, Guru, Kelas, dan Kelas Rusak).
- Target (y): Label status yang ingin diprediksi (AMAN/WASPADA).

Selanjutnya, data dipisah menggunakan teknik *Train-Test Split* dengan proporsi 80:20, di mana 80% data digunakan untuk melatih model dan 20% sisanya disembunyikan untuk pengujian nanti.

Berikut adalah implementasi kodenya:

```
# 3. Tentukan Fitur (X) dan Target (y)
X = df[['Total_Siswa', 'Total_Guru', 'Total_Kelas', 'Kelas_Rusak']]
y = df['Prediksi_Status']

# 4. Bagi Data (80% Latih, 20% Uji)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

print("\n✅ Data berhasil disiapkan.")
print(f"Data Latih: {len(X_train)} baris")
print(f"Data Uji  : {len(X_test)} baris")

# Tampilkan 5 data teratas untuk Laporan
print("\n--- SAMPEL DATA ---")
display(df.head())
```

Output:

```
--- SAMPEL DATA ---
```

	Lokasi	Total_Siswa	Total_Guru	Total_Kelas	Kelas_Rusak	Prediksi_Status
0	Kec. Bontonompo (Negeri)	0.313788	0.439490	0.696246	0.873874	1
1	Kec. Bajeng (Negeri)	0.446118	0.582803	0.808874	1.000000	1
2	Kec. Pallangga (Negeri)	0.917484	0.839172	1.000000	0.891892	1
3	Kec. Somba Opu (Negeri)	1.000000	1.000000	0.945392	1.000000	1
4	Kec. Bontomarannu (Negeri)	0.362765	0.366242	0.460751	0.288288	0

BAB 4. MODELING (PEMODELAN)

Pada tahap modeling, data yang telah dipersiapkan digunakan untuk melatih algoritma *Machine Learning* agar dapat mengenali pola dan melakukan prediksi status sekolah.

4.1 Pemilihan Algoritma

Algoritma yang dipilih untuk menyelesaikan permasalahan klasifikasi ini adalah Random Forest Classifier.

Random Forest adalah algoritma *ensemble learning* yang bekerja dengan cara membangun banyak pohon keputusan (*decision trees*) pada saat pelatihan. Hasil prediksi akhir ditentukan melalui sistem pemungutan suara terbanyak (*majority voting*) dari seluruh pohon yang terbentuk.

4.2 Alasan Pemilihan Algoritma

Pemilihan *Random Forest Classifier* didasarkan pada beberapa keunggulan berikut yang relevan dengan karakteristik dataset sekolah:

a. Akurasi Tinggi & Stabilitas: Berbeda dengan *Decision Tree* tunggal yang cenderung menghafal data (*overfitting*), *Random Forest* menggabungkan hasil dari ratusan pohon. Hal ini membuat model lebih stabil dan akurat saat memprediksi data baru yang belum pernah dilihat sebelumnya.
b. Tahan Terhadap *Outlier*: Data pendidikan seringkali memiliki variasi yang unik (misalnya sekolah kecil dengan murid sedikit atau sekolah besar dengan fasilitas minim). *Random Forest* memiliki kemampuan yang baik dalam menangani data pencilan (*outlier*) tanpa merusak performa model secara keseluruhan.
c. Analisis *Feature Importance*: Algoritma ini memiliki kemampuan bawaan untuk mengukur seberapa penting setiap fitur (variabel). Hal ini sangat krusial bagi proyek ini untuk mengetahui faktor mana yang paling dominan menyebabkan penurunan mutu sekolah (apakah karena kurangnya guru atau rusaknya fasilitas).

4.3 Proses Implementasi

Berikut adalah implementasi kode untuk melatih model menggunakan data latih (*training data*) yang telah disiapkan pada bab sebelumnya. Model dikonfigurasi dengan

parameter n_estimators=100, yang berarti model akan membangun 100 pohon keputusan.

```
print("\n⏳ Sedang melatih model Random Forest...")

# Inisialisasi Model
model = RandomForestClassifier(n_estimators=100, random_state=42)

# Latih Model dengan Data Training
model.fit(X_train, y_train)

print("✅ Model BERHASIL dilatih!")
```

Output:

```
⏳ Sedang melatih model Random Forest...
✅ Model BERHASIL dilatih!
```

BAB 5. EVALUATION

5.1 Metode Evaluasi

Metode evaluasi yang dipilih untuk proyek ini adalah Confusion Matrix dan Classification Report.

a. Confusion Matrix: Digunakan untuk melihat detail kesalahan prediksi model. Matriks ini menunjukkan berapa banyak data yang benar diprediksi sebagai "AMAN" (True Positive) dan berapa yang salah diprediksi (misalnya sebenarnya "WASPADA" tapi diprediksi "AMAN").
b. Classification Report: Digunakan untuk mendapatkan metrik yang lebih komprehensif daripada sekadar akurasi biasa. Metrik ini meliputi:
    - Precision: Tingkat ketepatan saat model memprediksi suatu kelas.
    - Recall: Kemampuan model menemukan seluruh data positif yang sebenarnya.
    - F1-Score: Rata-rata harmonis antara *precision* dan *recall*, sangat berguna jika jumlah data tidak seimbang.

5.2 Hasil Evaluasi Model

Berikut adalah implementasi kode untuk menampilkan hasil evaluasi berdasarkan metode yang dipilih.

```
print("\n--- HASIL EVALUASI MODEL ---")

# Prediksi Data Uji
y_pred = model.predict(X_test)

# 1. Skor Akurasi
akurasi = accuracy_score(y_test, y_pred)
print(f"Akurasi Model: {akurasi * 100:.2f}%")

# 2. Laporan Detail
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
```

Output

```
--- HASIL EVALUASI MODEL ---
Akurasi Model: 100.00%

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         2
           1       1.00      1.00      1.00         3

    accuracy                           1.00         5
   macro avg       1.00      1.00      1.00         5
weighted avg       1.00      1.00      1.00         5
```

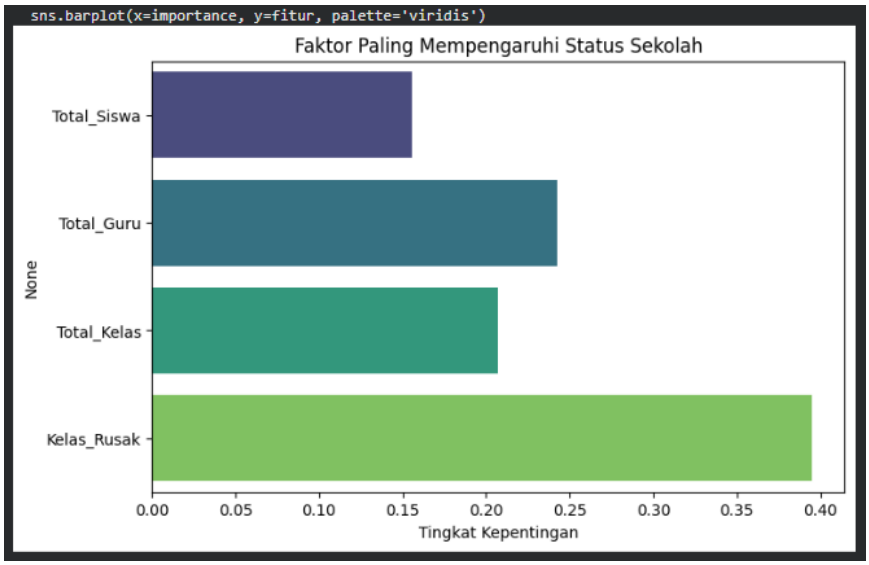
5.3 Analisis Feature Importance (Faktor Penentu)

Selain evaluasi performa, dilakukan juga analisis untuk mengetahui variabel apa yang paling berpengaruh dalam penentuan status sekolah. Analisis ini memanfaatkan fitur bawaan dari algoritma *Random Forest*.

implementasi kode

```
# 3. Grafik Feature Importance (Faktor apa yang paling penting?)
importance = model.feature_importances_
fitur = X.columns

plt.figure(figsize=(8, 5))
sns.barplot(x=importance, y=fitur, palette='viridis')
plt.title('Faktor Paling Mempengaruhi Status Sekolah')
plt.xlabel('Tingkat Kepentingan')
plt.show()
```

Output



BAB 6. DEPLOYMENT (IMPLEMENTASI)

6.1 Rancangan Sistem Deployment

Sistem ini dirancang menggunakan pustaka Gradio, sebuah *framework* Python yang memungkinkan pembuatan antarmuka pengguna (*User Interface*) berbasis web secara cepat.

Alur kerja aplikasi adalah sebagai berikut:

1. Input: Pengguna memasukkan data profil sekolah (Jumlah Siswa, Jumlah Guru, Total Kelas, dan Jumlah Kelas Rusak).
2. Proses: Sistem melakukan normalisasi data input agar sesuai dengan format pelatihan model, kemudian Model *Random Forest* memprediksi status risiko.
3. Output: Sistem menampilkan status ("AMAN" atau "WASPADA") disertai rekomendasi tindakan manajerial.

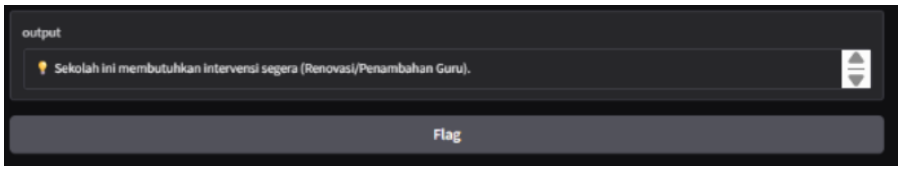
6.2 Implementasi Kode Antarmuka

Berikut adalah implementasi kode program untuk membangun antarmuka web tersebut. Kode ini mencakup fungsi logika prediksi yang menerjemahkan hasil numerik (0/1) kembali menjadi teks status, serta memberikan saran otomatis.

```python
# Fungsi Prediksi untuk Web
def prediksi_sekolah(total_siswa, total_guru, total_kelas, kelas_rusak):
    # Masukkan input user ke dalam format DataFrame (agar sama dengan saat training)
    data_baru = pd.DataFrame([[total_siswa, total_guru, total_kelas, kelas_rusak]],
                             columns=['Total_Siswa', 'Total_Guru', 'Total_Kelas', 'Kelas_Rusak'])

    # Lakukan Prediksi
    hasil_prediksi = model.predict(data_baru)[0]

    # Hitung rasio untuk memberi info tambahan
    rasio_guru = total_siswa / total_guru if total_guru > 0 else 0
    persen_rusak = (kelas_rusak / total_kelas) * 100 if total_kelas > 0 else 0

    # Pesan Output
    pesan = f"Status Sekolah: {hasil_prediksi}\n\n"
    pesan += f"📊 Analisis Singkat:\n"
    pesan += f"- Beban Guru: 1 Guru mengajar {int(rasio_guru)} Siswa\n"
    pesan += f"- Kerusakan Fisik: {int(persen_rusak)}% dari total kelas"

    # Tambahkan rekomendasi sederhana
    if "WASPADA" in hasil_prediksi:
        pesan += "\n\n⚠️ REKOMENDASI: Sekolah ini membutuhkan intervensi segera (Renovasi/Penambahan Guru)."
    else:
        pesan += "\n\n✅ SARAN: Pertahankan kualitas manajemen sekolah."

    return pesan

# Membuat Tampilan Web dengan Gradio
interface = gr.Interface(
    fn=prediksi_sekolah,
    inputs=[
        gr.Number(label="Total Siswa", value=300),
        gr.Number(label="Total Guru", value=10),
        gr.Number(label="Total Ruang Kelas", value=10),
        gr.Number(label="Jumlah Kelas Rusak", value=2),
    ],
    outputs="text",
    title="Sistem Deteksi Dini Sekolah (Kab. Gowa)",
    description="Masukkan data profil sekolah untuk memprediksi risiko penurunan mutu pendidikan.",
    theme="default"
)

print("\n✅ Aplikasi Web Siap! Silakan gunakan di bawah ini:")
interface.launch(share=True) # share=True agar bisa dibuka di HP/Link publik
```

Setelah kode dijalankan, sistem menghasilkan tautan web publik yang dapat diakses melalui browser.

Output


output
Sekolah ini membutuhkan intervensi segera (Renovasi/Penambahan Guru).
Flag

Daftar Pustaka

[1] Republik Indonesia, "Undang-Undang Nomor 20 Tahun 2003 tentang Sistem Pendidikan Nasional," Jakarta: Sekretariat Negara, 2003.

[2] E. A. Hanushek, "The Economic Value of Higher Teacher Quality," *Economics of Education Review*, vol. 30, no. 3, pp. 466-479, 2011.

[3] P. Barrett, F. Davies, Y. Zhang, and L. Barrett, "The impact of classroom design on pupils' learning: Final results of a holistic, multi-level analysis," *Building and Environment*, vol. 89, pp. 118-133, 2015.

[4] B. K. Baradwaj and S. Pal, "Mining Educational Data to Analyze Students' Performance," *International Journal of Advanced Computer Science and Applications*, vol. 2, no. 6, 2011.

[5] C. Romero and S. Ventura, "Educational Data Mining: A Review of the State of the Art," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 40, no. 6, pp. 601-618, 2010.

[6] Sumber Data: Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi. (2025). *Data Pokok Pendidikan (Dapodik) Kabupaten Gowa*. Diakses dari https://data.kemendikdasmen.go.id/dataset/pendidikan-2