BAB 4
HASIL PENELITIAN

4.1 RINGKASAN HASIL PENELITIAN

Berdasarkan penelitian yang dilakukan, tujuan utama dari studi ini adalah untuk menganalisis dan memprediksi kelulusan mahasiswa Tugas Akhir di Fakultas Teknologi dan Informatika (FTTI) Universitas Jenderal Achmad Yani Yogyakarta dengan menggunakan metode *Decision Tree*. Tahapan dalam penelitian mencakup proses pengumpulan data, *preprocessing*, pembagian data, penerapan model, hingga evaluasi hasil prediksi. Data yang digunakan berasal dari bagian akademik universitas, dengan total sebanyak 440 data mahasiswa. Dari keseluruhan data tersebut, tercatat 136 mahasiswa telah mengambil mata kuliah Tugas Akhir. Untuk keperluan pelatihan dan pengujian model, data dibagi menjadi dua bagian, yaitu 219 data sebagai data *training* dan 94 data sebagai data *testing*.

Metode Decision Tree yang digunakan dalam penelitian ini menerapkan algoritma CART (Classification and Regression Tree), yang membentuk struktur pohon biner berdasarkan pemilihan atribut terbaik dengan kriteria Gini Index. Algoritma ini memungkinkan identifikasi pola kelulusan mahasiswa berdasarkan atribut-atribut akademik secara hierarkis dan mudah dipahami. Berdasarkan hasil pengujian model, Decision Tree mampu memprediksi kelulusan mahasiswa dengan cukup baik. Dari 136 mahasiswa yang dianalisis, sebanyak 128 mahasiswa yang mengambil Tugas Akhir diprediksi akan lulus, sedangkan 8 mahasiswa diprediksi tidak lulus. Jika dinyatakan dalam persentase, sebesar 94,11% mahasiswa diprediksi lulus, sementara 5,88% sisanya diprediksi tidak lulus. Hasil ini menunjukkan bahwa metode Decision Tree dengan algoritma CART memiliki potensi yang baik dalam mengidentifikasi peluang kelulusan mahasiswa berdasarkan data akademik yang tersedia.

4.2 PEMBAHASAN

Model prediksi kelulusan mahasiswa yang dikembangkan dengan algoritma Classification and Regression Tree (CART) dalam metode Decision Tree mampu

memberikan hasil klasifikasi yang sangat baik. Evaluasi terhadap kinerja model menghasilkan akurasi sebesar 93,62%, precision 93,33%, recall 96,55%, dan F1-score 94,92%, yang menunjukkan bahwa model mampu membedakan mahasiswa yang berpotensi lulus dan tidak lulus secara seimbang dan akurat. Dari struktur pohon keputusan yang dihasilkan, Indeks Prestasi Kumulatif (IPK) muncul sebagai variabel paling berpengaruh, disusul oleh SKS Kumulatif (SKSK). Pola ini menunjukkan bahwa kelulusan sangat dipengaruhi oleh capaian akademik mahasiswa, di mana IPK tinggi dan jumlah SKS yang memadai menjadi indikator kuat terhadap keberhasilan menyelesaikan tugas akhir.

Untuk mendukung interpretasi hasil, penelitian ini juga menyediakan dashboard interaktif berbasis Streamlit yang menyajikan data dan visualisasi secara informatif dan real-time. Fitur seperti grafik batang, diagram lingkaran, distribusi prediksi berdasarkan program studi, serta visualisasi pohon keputusan, membantu pengguna memahami kecenderungan dan pola dalam data. Sebagian besar mahasiswa diproyeksikan akan lulus, namun tetap ditemukan sejumlah kasus mahasiswa yang diprediksi tidak lulus, khususnya pada program studi tertentu. Informasi ini dapat digunakan oleh pihak kampus untuk melakukan pemantauan serta tindakan preventif terhadap mahasiswa yang teridentifikasi memiliki risiko tidak lulus. Pembahasan selanjutnya akan menjelaskan secara teknis tahapan yang dilakukan dalam membangun model, dimulai dari tahap *preprocessing data*.

4.2.1 *Preprocessing* Data

Data yang telah dikumpulkan selanjutnya melalui tahapan *preprocessing* untuk memastikan kualitas dan konsistensinya sebelum digunakan dalam proses pemodelan. Tahapan ini mencakup dimulai dari menggabungkan data mahasiswa, membersihkan outlier, standarisasi dan membersihkan kolom kemudian terakhir encoding kategori

1. Menggabungkan Data Mahasiswa

Pada tahap awal preprocessing, data mahasiswa yang berasal dari berbagai sumber digabungkan menjadi satu dataset utama. Data ini bisa meliputi informasi akademik seperti nilai IPS, IPK, jumlah SKS, serta data tambahan seperti kehadiran

atau data personal. Proses penggabungan dilakukan dengan mencocokkan kunci unik seperti Nomor Induk Mahasiswa (NIM), sehingga semua atribut yang relevan dari berbagai tabel dapat digabung secara akurat. Hal ini memastikan bahwa seluruh informasi penting untuk analisis telah terintegrasi dalam satu file data yang utuh.

```
def preprocess_year_sheet(sheet_name, tahun):
    df = df_sheets[sheet_name].copy()

    df.columns = df.iloc[0]
    df = df[1:]

     df.columns = ["NO", "NIM", "NAMA", "PRODI", "IPS", "SKS",
"IPK", "SKSK", "SEMESTER_LULUS"]


    df["TAHUN"] = tahun
    df["LULUS"] = df["SEMESTER_LULUS"].notna().astype(int)


     return df[["NIM", "NAMA", "PRODI", "IPS", "SKS", "IPK",
"SKSK", "SEMESTER_LULUS", "LULUS", "TAHUN"]]


data_training = pd.concat([
    preprocess_year_sheet("2018", 2018),
    preprocess_year_sheet("2019", 2019),
    preprocess_year_sheet("2020", 2020),
    preprocess_year_sheet("2021", 2021)
], ignore_index=True)


data_training.head()
```


Kode tersebut digunakan untuk menggabungkan data mahasiswa dari berbagai tahun akademik menjadi satu dataset yang terintegrasi sebagai bagian dari tahap awal preprocessing. Fungsi *preprocess_year_sheet()* bertugas memproses data dari setiap sheet berdasarkan tahun, dengan cara menyalin data, mengatur ulang nama kolom agar konsisten, serta menambahkan dua kolom baru, yaitu TAHUN untuk menandai tahun asal data, dan LULUS sebagai label biner yang menunjukkan status kelulusan mahasiswa (1 jika sudah lulus, 0 jika belum). Setelah fungsi ini dijalankan untuk masing-masing sheet tahun 2018 hingga 2021, data dari keempat tahun tersebut digabung menggunakan *pd.concat()* menjadi satu DataFrame bernama data_training. Dengan penggabungan ini, seluruh informasi akademik

mahasiswa dari berbagai angkatan dapat dianalisis secara menyeluruh dalam satu dataset yang siap untuk diproses lebih lanjut dalam tahap preprocessing

2. Membersihkan outlier

Setelah data tergabung, langkah selanjutnya adalah membersihkan outlier, yaitu nilai-nilai ekstrem yang secara signifikan berbeda dari mayoritas data dan dapat memengaruhi hasil analisis atau model. Pendeteksian outlier dilakukan menggunakan metode statistik seperti *Interquartile Range (IQR)*. Setelah outlier teridentifikasi, dilakukan penanganan yang sesuai, misalnya dengan menghapus baris data tersebut, mengganti dengan nilai median, atau melakukan clipping agar nilai-nilai berada dalam batas wajar. Tujuannya adalah memastikan data yang digunakan benar-benar merepresentasikan pola umum. Kode yang digunakan untuk membersihkan outlier adalah sebagai berikut:

```
data_clean = data_training.drop(outlier_df.index,
errors='ignore')
print(f"Data awal: {len(data_training)}, setelah hapus outlier:
{len(data_clean)}")
```

Kode tersebut berfungsi untuk menghapus data yang teridentifikasi sebagai outlier dari dataset utama data_*training*, dengan menyimpan hasilnya ke dalam variabel baru bernama data_*clean*. Proses ini dilakukan menggunakan fungsi *drop()* dengan referensi indeks dari *outlier_df.index,* yaitu kumpulan baris yang sebelumnya telah dikenali sebagai outlier melalui metode statistik tertentu seperti IQR. Penggunaan parameter *errors='ignore'* memastikan bahwa proses penghapusan tetap berjalan meskipun ada indeks yang tidak ditemukan. Selanjutnya, perintah print() digunakan untuk menampilkan jumlah data sebelum dan sesudah penghapusan, sehingga dapat diketahui berapa banyak data yang telah dibersihkan. Berdasarkan output yang dihasilkan, jumlah data berkurang dari 440 menjadi 372, yang berarti sebanyak 68 data outlier telah berhasil dihapus dari dataset.

3. Standarisasi dan Membersihkan Kolom

Tahapan ini bertujuan untuk merapikan dan menyamakan format data di seluruh kolom. Beberapa langkah yang dilakukan antara lain membersihkan data teks dari

spasi berlebih, menyamakan kapitalisasi huruf, dan memperbaiki ketidakkonsistenan penulisan. Selain itu, tipe data juga distandarisasi—misalnya, nilai numerik yang awalnya dalam bentuk teks akan dikonversi ke format angka. Nilai kosong (*missing value*) juga diperiksa dan ditangani dengan cara menghapus baris kosong atau mengisinya menggunakan metode imputasi seperti rata-rata atau nilai terbanyak. Kode yang digunakan untuk standarisasi dan membersihkan kolom adalah sebagai berikut:

```
data_training["PRODI"] = (
    data_training["PRODI"]
    .astype(str)
    .str.strip()
    .str.replace(r"\s+", " ", regex=True)
)

data_training["PRODI"].value_counts()

from sklearn.preprocessing import LabelEncoder

data_training["PRODI"]                                                    =
data_training["PRODI"].astype(str).str.strip()
print("PRODI unik:", data_training["PRODI"].unique())


print(data_training["PRODI"].value_counts())
```


Kode tersebut digunakan untuk membersihkan dan menstandarisasi data pada kolom "PRODI" agar nilai-nilainya konsisten dan siap digunakan dalam analisis. Langkah pertama dilakukan dengan mengubah seluruh isi kolom menjadi tipe string, menghapus spasi di awal dan akhir teks, serta menyederhanakan spasi ganda menjadi satu spasi agar tidak terjadi perbedaan penulisan yang secara teknis dianggap berbeda meskipun secara visual tampak sama. Setelah pembersihan, dilakukan pengecekan terhadap nilai-nilai unik pada kolom tersebut menggunakan fungsi unique() dan value_counts() untuk melihat sebaran data dan memastikan tidak ada duplikasi akibat inkonsistensi penulisan. Tahap ini sangat penting karena ketidakkonsistenan dalam data kategori seperti nama program studi dapat menyebabkan kesalahan saat dilakukan encoding atau analisis lebih lanjut, sehingga perlu dirapikan terlebih dahulu untuk menghasilkan dataset yang valid dan terstruktur

4. Encoding Kategori

Data kategorikal seperti nama program studi, jenis kelamin, atau status mahasiswa tidak dapat digunakan secara langsung dalam model pembelajaran mesin yang membutuhkan input numerik. Oleh karena itu, encoding dilakukan untuk mengubah data kategori menjadi bentuk angka. Teknik encoding yang umum digunakan adalah label encoding, yang memberi nilai numerik pada setiap kategori, atau one-hot encoding, yang mengubah setiap kategori menjadi kolom biner (0 atau 1). Proses ini penting agar model dapat mengolah data kategori tanpa kehilangan makna aslinya.

```
le = LabelEncoder()
data_training["PRODI_ENCODED"]                                    =
le.fit_transform(data_training["PRODI"])

mapping_prodi = dict(zip(le.classes_, le.transform(le.classes_)))
print("Mapping PRODI:", mapping_prodi)

# Cek hasil
data_training[["PRODI", "PRODI_ENCODED"]].drop_duplicates()
```

Kode tersebut digunakan untuk mengubah data kategorikal pada kolom "PRODI" menjadi format numerik melalui proses label encoding, agar dapat digunakan dalam algoritma pembelajaran mesin yang hanya menerima input berupa angka. Dengan menggunakan objek LabelEncoder() dari pustaka sklearn.preprocessing, setiap kategori unik dalam kolom "PRODI" diubah menjadi nilai numerik dan disimpan dalam kolom baru bernama "PRODI_ENCODED". Selanjutnya, dibuat dictionary mapping_prodi untuk memetakan setiap kategori asli dengan nilai encoding-nya, sehingga hubungan antara label dan angka tetap dapat dilacak. Terakhir, hasil encoding diverifikasi dengan menampilkan kombinasi unik antara nilai asli dan hasil encoding menggunakan drop_duplicates(). Proses ini penting untuk memastikan bahwa informasi dalam data kategorikal tetap dipertahankan meskipun telah diubah ke dalam bentuk numerik yang dapat diolah oleh model.

4.2.2 Pembagian Data

Setelah proses preprocessing selesai, data dibagi menjadi dua bagian, yaitu data latih (training data) dan data uji (testing data), dengan rasio 70% untuk pelatihan dan 30% untuk pengujian. Data latih digunakan untuk membentuk model dengan mempelajari pola dan hubungan antar variabel, sedangkan data uji berfungsi untuk mengevaluasi kinerja model terhadap data yang belum pernah dilihat sebelumnya. Evaluasi ini bertujuan untuk mengukur sejauh mana model mampu melakukan prediksi secara akurat dan generalisasi terhadap data nyata di luar data pelatihan. Melalui proses ini, dapat diketahui apakah model yang dibangun memiliki performa yang baik. Kode yang digunakan untuk pembagian data adalah sebagai berikut:

```
X = train_data[['PRODI_ENCODED', 'IPS', 'SKS', 'IPK', 'SKSK',
'TAHUN']]
y = train_data['LULUS']

X_train_split, X_test_split, y_train_split, y_test_split =
train_test_split(
    X, y, test_size=0.3, random_state=42, stratify=y
)
print(f"Jumlah data training: {len(X_train_split)}")
print(f"Jumlah data testing: {len(X_test_split)}")
```


Kode tersebut digunakan untuk membagi dataset menjadi dua bagian, yaitu data latih dan data uji, sebagai persiapan sebelum pelatihan model. Variabel fitur (X) terdiri dari beberapa kolom yang dianggap berpengaruh terhadap kelulusan mahasiswa, yaitu PRODI_ENCODED, IPS, SKS, IPK, SKSK, dan TAHUN, sedangkan variabel target (y) adalah kolom LULUS yang menunjukkan status kelulusan mahasiswa dalam bentuk biner. Fungsi *train_test_split()* digunakan untuk membagi data dengan proporsi 70% sebagai data latih dan 30% sebagai data uji. Parameter *random_state=42* digunakan agar hasil pembagian data bersifat tetap dan dapat direproduksi, sementara *stratify=y* memastikan distribusi kelas target tetap seimbang antara data latih dan data uji. Setelah pembagian dilakukan, jumlah data pada masing-masing subset ditampilkan, yaitu sebanyak 219 data untuk

pelatihan dan 94 data untuk pengujian, yang menunjukkan bahwa proses pembagian telah berhasil dan proporsional.

4.2.3 Implementasi Decision tree

Model klasifikasi *Decision Tree* diterapkan untuk mengidentifikasi pola kelulusan mahasiswa berdasarkan atribut-atribut akademik, seperti Indeks Prestasi Kumulatif (IPK), jumlah SKS yang telah ditempuh, status pengambilan Tugas Akhir (TA), dan tingkat stres akademik. Melalui struktur pohon keputusan yang dihasilkan, model dapat memprediksi status kelulusan mahasiswa yang mengambil TA dengan mempertimbangkan variabel-variabel pendukung tersebut. Kode yang digunakan untuk implementasi model adalah sebagai berikut:

```
model = DecisionTreeClassifier(max_depth=5, random_state=42)
model.fit(X, y)

plt.figure(figsize=(23, 13))
plot_tree(
    model,
    feature_names=features,
    class_names=model.classes_,
    filled=True,
    rounded=True,
    fontsize=12
)
plt.title("Pohon Keputusan Prediksi Kelulusan)", fontsize=20)
plt.tight_layout()
plt.show()
```


Kode di atas digunakan untuk membangun dan memvisualisasikan model pohon keputusan (Decision Tree) yang bertujuan untuk memprediksi kelulusan mahasiswa berdasarkan sejumlah variabel, seperti IPK, SKS, SKSK, dan IPS. Model dibuat menggunakan DecisionTreeClassifier dengan kedalaman maksimal lima level (max_depth=5) untuk menghindari overfitting, serta menggunakan parameter random_state=42 agar hasilnya konsisten saat dijalankan berulang. Algoritma yang digunakan dalam model ini adalah CART (Classification and Regression Tree), yaitu metode pembentukan pohon keputusan yang membagi data berdasarkan nilai-nilai atribut dengan menghitung indeks Gini sebagai ukuran ketidakmurnian (impurity). Setelah model dilatih menggunakan data fitur X dan label target y, visualisasi pohon dibuat menggunakan fungsi plot_tree, yang

menampilkan struktur pohon secara lengkap dengan informasi nama fitur, kelas prediksi, serta pewarnaan node berdasarkan kategori kelas mayoritas. Visualisasi ini juga dilengkapi dengan elemen estetika seperti sudut membulat, pewarnaan penuh, serta ukuran font yang diperbesar untuk meningkatkan keterbacaan. Dengan melihat hasil visualisasi ini, kita dapat memahami bagaimana model mengambil keputusan, misalnya dengan melihat cabang-cabang keputusan berdasarkan nilai IPK atau SKSK, dan bagaimana kombinasi variabel tersebut memengaruhi prediksi kelulusan mahasiswa.

4.2.4 Evaluasi Model

Selanjutnya, model dievaluasi menggunakan data uji dengan metrik akurasi, *precision*, *recall, F1-score,* dan *confusion matrix* untuk menilai kinerja model secara menyeluruh. Kode yang digunakan untuk evaluasi model adalah sebagai berikut:

```
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score

accuracy = accuracy_score(y_test_split, y_pred_split) * 100
precision = precision_score(y_test_split, y_pred_split, zero_division=0) * 100
recall = recall_score(y_test_split, y_pred_split, zero_division=0) * 100
f1 = f1_score(y_test_split, y_pred_split, zero_division=0) * 100

print(f"\nMetrik Evaluasi Model")
print(f"Akurasi : {accuracy:.2f}%")
print(f"Precision: {precision:.2f}%")
print(f"Recall   : {recall:.2f}%")
print(f"F1 Score : {f1:.2f}%")
```

Kode di atas digunakan untuk menghitung metrik evaluasi model berupa akurasi, *precision*, *recall*, dan *F1-score*. Metrik tersebut digunakan untuk menilai sejauh mana model mampu melakukan prediksi dengan benar dan seimbang. Hasil dari evaluasi model dapat dilihat pada tabel 4.1.

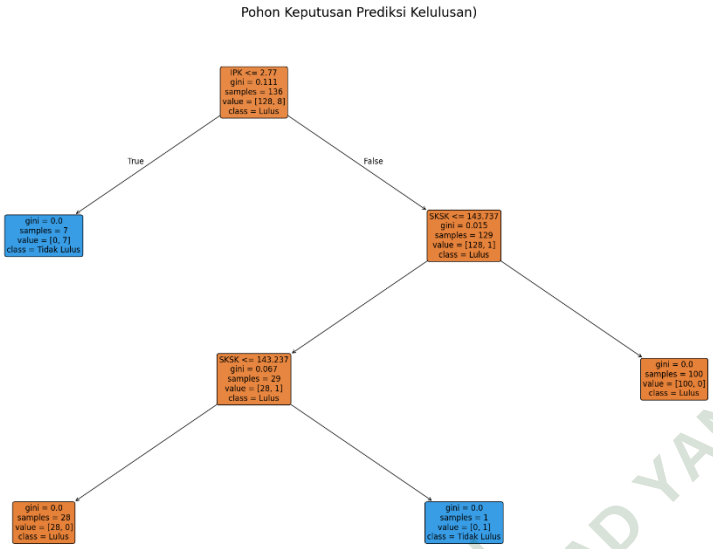
Tabel 4.1 Akurasi Model

Akurasi	Nilai
Accuracy	93.62%
Precision	93.33%
Recall	96.55%
F1-Score	94.92%

Pada Tabel 4.1 diperoleh hasil evaluasi model dengan akurasi sebesar 93,62%, yang menunjukkan bahwa model memiliki tingkat ketepatan yang tinggi dalam mengklasifikasikan data kelulusan mahasiswa. Nilai precision sebesar 93,33% mengindikasikan bahwa sebagian besar mahasiswa yang diprediksi lulus oleh model memang benar-benar lulus. Sementara itu, recall sebesar 96,55% menunjukkan bahwa model mampu mendeteksi hampir seluruh mahasiswa yang lulus dengan benar. Nilai F1-Score sebesar 94,92% mencerminkan keseimbangan yang sangat baik antara precision dan recall. Dengan demikian, model dapat dikatakan memiliki kinerja yang sangat baik dan andal dalam memprediksi kelulusan mahasiswa tugas akhir.

4.2.5 Analisis Hasil

Selanjutnya, dilakukan analisis untuk mengidentifikasi faktor-faktor utama yang memengaruhi kelulusan mahasiswa tugas akhir. Analisis ini dilakukan dengan mengamati struktur pohon keputusan yang dihasilkan dari proses klasifikasi. Setiap percabangan pada pohon mencerminkan atribut yang berperan dalam pengambilan keputusan, seperti Indeks Prestasi Kumulatif (IPK), jumlah SKS yang telah ditempuh. Hasil dari pohon keputusan dapat dilihat pada Gambar 4.1.



Gambar 4.1 Pohon keputusan

Gambar 4.1 menunjukkan hasil visualisasi dari model pohon keputusan yang digunakan untuk memprediksi kelulusan mahasiswa berdasarkan sejumlah parameter akademik, yaitu Indeks Prestasi Kumulatif (IPK), Indeks Prestasi Semester (IPS), jumlah Satuan Kredit Semester (SKS), dan SKS Kumulatif (SKSK). Model ini dibangun menggunakan algoritma CART (Classification and Regression Tree), yang menghasilkan struktur pohon biner di mana setiap node merepresentasikan keputusan berdasarkan nilai ambang dari suatu atribut. Berdasarkan visualisasi, model memulai proses pemisahan dari akar pohon (root node) menggunakan atribut IPK dengan ambang batas sebesar 2.77. Jika IPK mahasiswa kurang dari atau sama dengan 2.77, maka data diarahkan ke cabang kiri dan diklasifikasikan sebagai "Tidak Lulus". Hal ini ditunjukkan oleh simpul dengan nilai gini sebesar 0.0, jumlah sampel 7 mahasiswa, dan seluruhnya tidak lulus, yang mencerminkan bahwa IPK rendah merupakan indikator awal yang sangat kuat terhadap kemungkinan ketidaklulusan. Sebaliknya, jika IPK lebih dari 2.77, maka data diarahkan ke cabang kanan, di mana pemisahan selanjutnya dilakukan berdasarkan atribut SKSK dengan ambang batas 143.737.

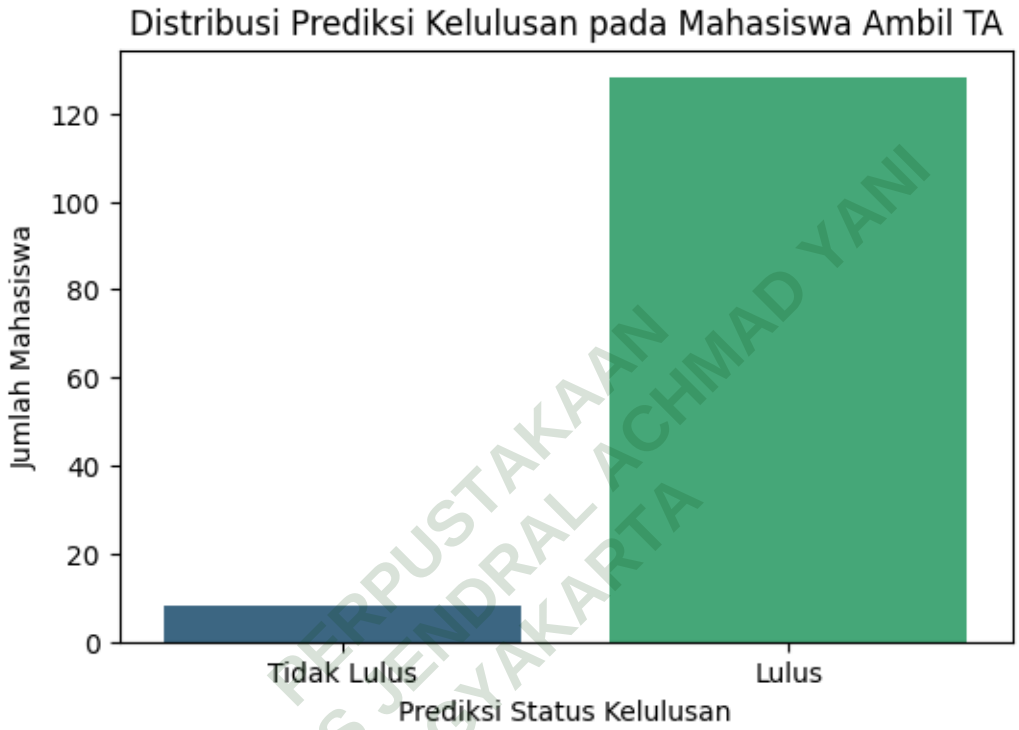
Pada percabangan ini, jika SKSK mahasiswa kurang dari atau sama dengan 143.737, proses dilanjutkan ke simpul berikutnya yang memisahkan kembali data

berdasarkan ambang SKSK sebesar 143.237. Jika nilai SKSK kurang dari atau sama dengan 143.237, maka seluruh mahasiswa pada simpul tersebut (sebanyak 28 mahasiswa) diklasifikasikan sebagai “Lulus” dengan nilai gini sebesar 0.0. Sebaliknya, jika SKSK lebih dari 143.237, hanya terdapat satu mahasiswa yang diklasifikasikan sebagai “Tidak Lulus”. Sementara itu, jika SKSK mahasiswa lebih dari 143.737, maka data langsung diklasifikasikan sebagai “Lulus” tanpa perlu pemisahan lebih lanjut, dengan 100 mahasiswa dalam simpul tersebut dan nilai gini sebesar 0.0 yang menunjukkan homogenitas sempurna. Setiap simpul pada pohon keputusan ini menampilkan informasi penting seperti nilai gini, jumlah sampel, distribusi kelas (value), dan hasil klasifikasi akhir (class). Nilai gini menggambarkan tingkat ketidakmurnian pada setiap node, di mana semakin mendekati nol berarti semakin murni atau homogen data pada simpul tersebut.

Struktur pohon yang terbentuk menunjukkan bahwa IPK merupakan faktor paling dominan yang memengaruhi kelulusan mahasiswa karena menjadi pemisah pertama dalam pohon. Mahasiswa dengan IPK lebih dari 2.77 sebagian besar diklasifikasikan sebagai “Lulus”, sementara yang memiliki IPK sama dengan atau di bawah 2.77 cenderung “Tidak Lulus”. Selain itu, atribut SKSK juga menjadi penentu penting karena muncul dalam beberapa simpul sebagai kriteria pemisah lanjutan. Hal ini mengindikasikan bahwa mahasiswa dengan jumlah SKS kumulatif yang lebih tinggi cenderung memiliki progres akademik yang baik dan peluang kelulusan yang lebih tinggi. Meskipun dalam visualisasi ini atribut IPS dan SKS tidak muncul secara eksplisit dalam struktur pohon, bukan berarti keduanya tidak berpengaruh, melainkan IPK dan SKSK memiliki kontribusi paling signifikan dalam proses pembentukan pohon berdasarkan algoritma CART.

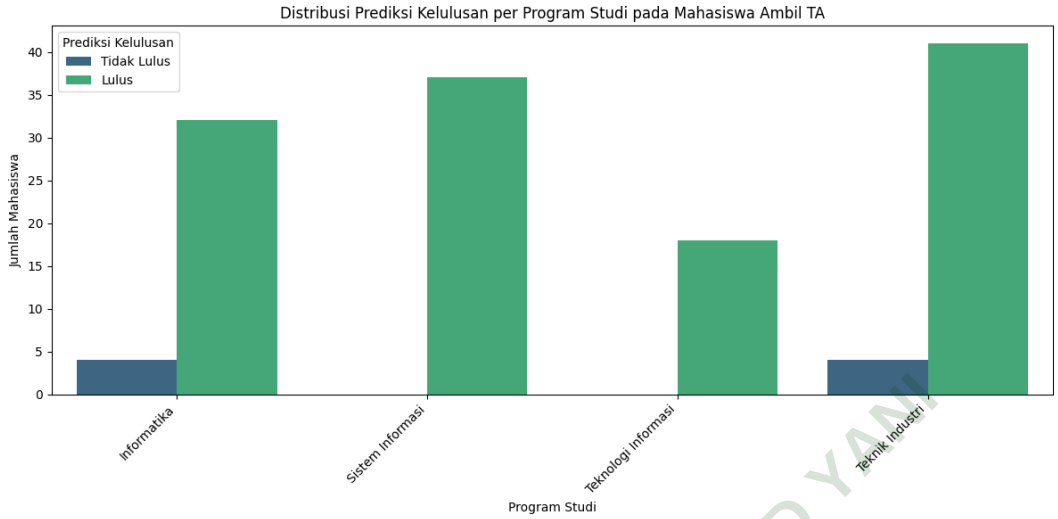
Secara keseluruhan, pohon keputusan ini memberikan gambaran visual yang jelas mengenai pola hubungan antara atribut akademik dengan hasil kelulusan mahasiswa. Model ini tidak hanya memberikan prediksi, tetapi juga mampu menjelaskan logika pengambilan keputusan yang dilakukan berdasarkan nilai-nilai atribut tertentu. Berdasarkan hasil prediksi yang dilakukan oleh model pohon keputusan, diketahui bahwa dari seluruh data mahasiswa yang dianalisis, sebanyak 112 mahasiswa (82,35%) diprediksi akan lulus, sedangkan 24 mahasiswa (17,65%)

diprediksi tidak lulus. Hasil ini menunjukkan bahwa mayoritas mahasiswa dalam dataset memiliki karakteristik yang memenuhi kriteria kelulusan menurut model, khususnya dalam hal IPK dan SKSK yang tinggi. Visualisasi untuk hasil analisis prediksi ini dapat dilihat pada Gambar 4.2.



Gambar 4.2 Distribusi Prediksi kelulusan

Dari Gambar 4.2, grafik tersebut menunjukkan distribusi prediksi kelulusan mahasiswa yang sedang menempuh Tugas Akhir (TA). Terlihat bahwa jumlah mahasiswa yang diprediksi akan lulus jauh lebih besar dibandingkan dengan yang diprediksi tidak lulus. Secara visual, kolom berlabel *Lulus* memiliki tinggi yang signifikan, menunjukkan bahwa sebanyak 128 mahasiswa diprediksi lulus, sedangkan hanya 8 mahasiswa yang diprediksi tidak lulus. Selanjutnya adalah visualisasi dari prediksi kelulusan setiap prodi, visualisasi dapat dilihat pada Gambar 4.3.



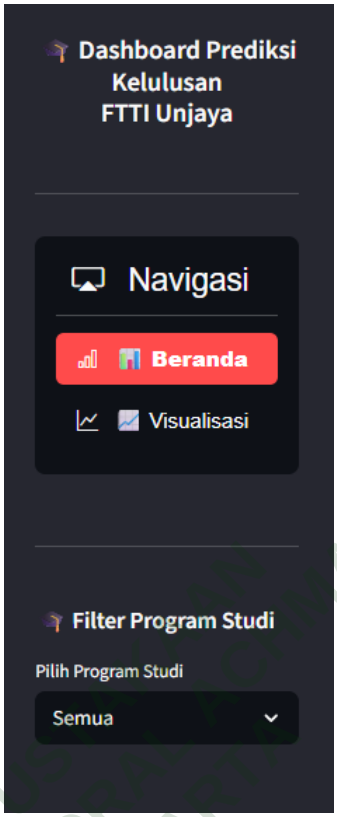
Gambar 4.3 Distribusi Prediksi Kelulusan Setiap Prodi

Gambar 4.3 menunjukkan distribusi prediksi kelulusan mahasiswa berdasarkan program studi pada mahasiswa yang mengambil Tugas Akhir. Secara keseluruhan, jumlah mahasiswa yang diprediksi lulus lebih tinggi dibandingkan yang tidak lulus di setiap program studi. Program Studi Teknik Industri memiliki jumlah prediksi kelulusan tertinggi dengan sekitar 40 mahasiswa, diikuti oleh Sistem Informasi dengan sekitar 37 mahasiswa. Sementara itu, Teknologi Informasi menunjukkan jumlah prediksi kelulusan paling rendah, yakni sekitar 18 mahasiswa, namun tidak terdapat mahasiswa yang diprediksi tidak lulus pada program studi ini. Program Studi Informatika dan Teknik Industri juga menunjukkan jumlah mahasiswa yang diprediksi tidak lulus yang relatif menonjol, masing-masing sekitar 4 mahasiswa. Hal ini mengindikasikan bahwa meskipun sebagian besar mahasiswa diprediksi lulus, masih terdapat beberapa program studi yang memiliki jumlah prediksi tidak lulus yang perlu menjadi perhatian.

4.3 IMPLEMENTASI DASHBOARD

4.3.1 Menu Navigasi

Pada tampilan awal *dashboard*, ditampilkan menu navigasi yang terletak di sisi kiri antarmuka. Menu ini berfungsi sebagai kontrol utama yang memungkinkan pengguna untuk memilih dan mengakses halaman yang diinginkan. Tampilan menu navigasi tersebut dilihat pada Gambar 4.4.

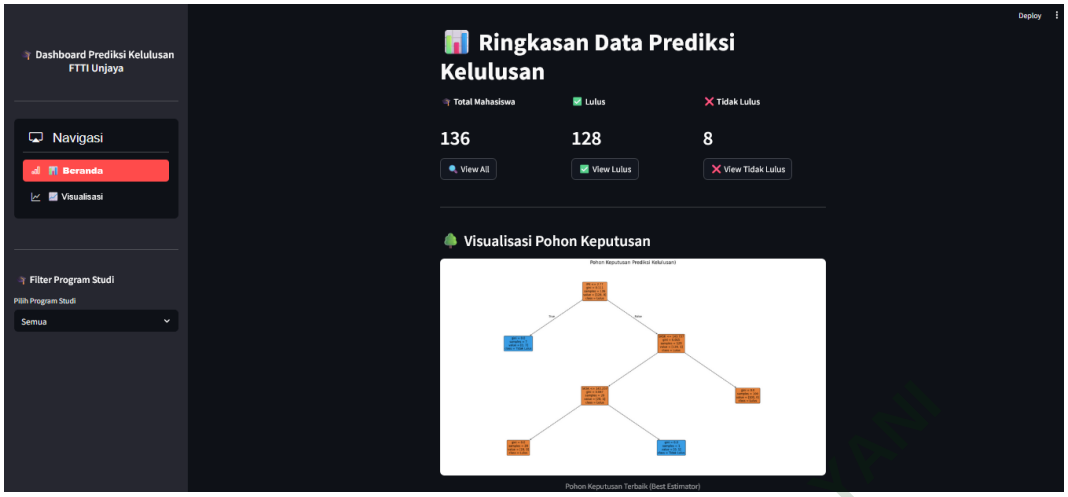


Gambar 4.4 Menu Navigasi

Gambar 4.4 menampilkan menu navigasi yang terletak di sisi kiri antarmuka *dashboard*. Menu ini dirancang untuk memudahkan pengguna dalam mengakses fitur utama, yaitu Beranda dan Visualisasi, dengan tampilan ikon yang informatif dan indikator warna untuk menandai menu yang aktif. Di bawahnya terdapat fitur Filter Program Studi yang memungkinkan pengguna menyaring data berdasarkan program studi atau menampilkan seluruh data secara sekaligus. Desain antarmuka yang sederhana dan responsif ini bertujuan untuk meningkatkan kemudahan penggunaan serta mendukung proses analisis data secara efisien.

4.3.2 Halaman Beranda

Halaman Beranda merupakan tampilan pertama yang muncul saat pengguna mengakses dashboard. Halaman ini menyajikan ringkasan data yang digunakan dalam penelitian sebagai informasi awal. Tampilan Halaman Beranda dapat dilihat pada Gambar 4.5

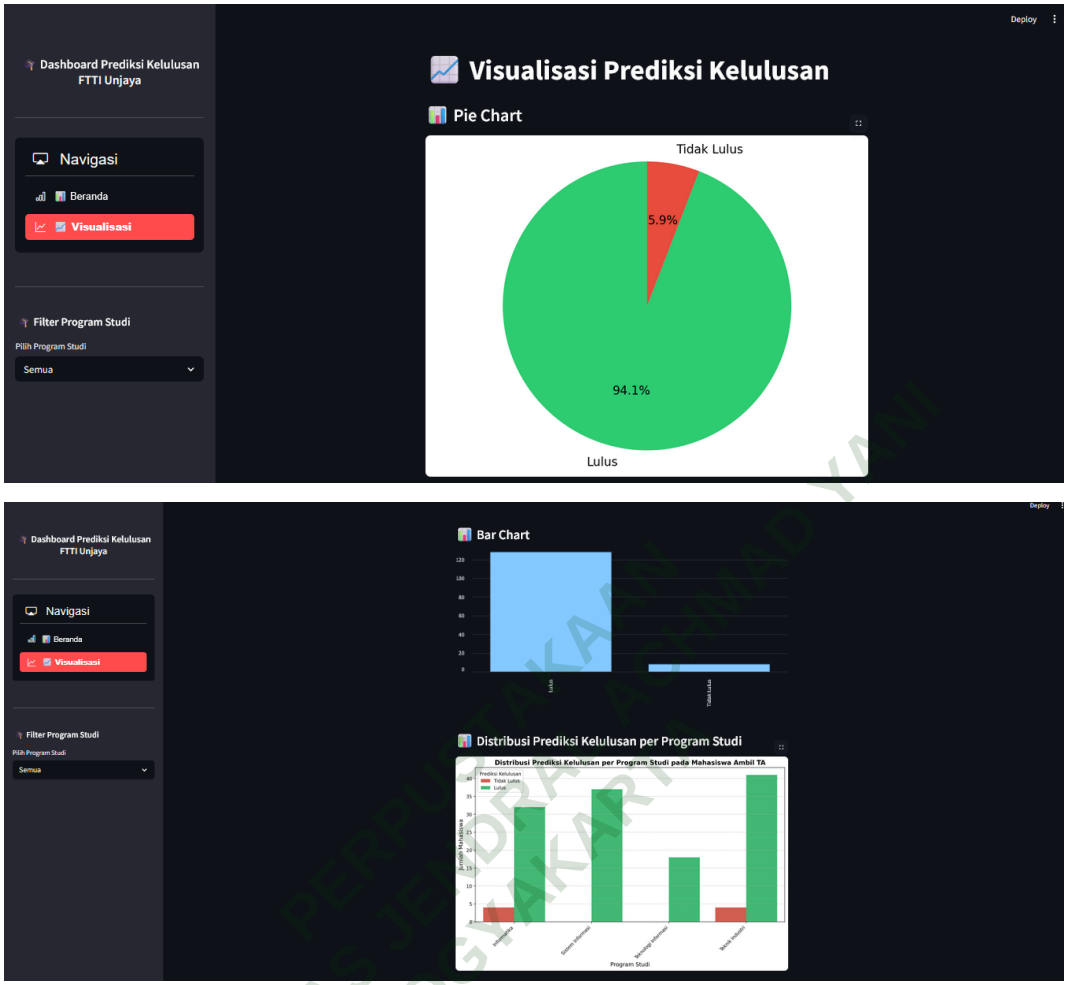


Gambar 4.5 Halaman Beranda

Pada Gambar 4.5 menampilkan halaman beranda yang berisi Ringkasan Data Prediksi Kelulusan dan Visualisasi Pohon Keputusan. Ringkasan data menunjukkan total mahasiswa yang dianalisis sebanyak 136 orang, dengan rincian 128 mahasiswa diprediksi lulus dan 8 mahasiswa diprediksi tidak lulus. Informasi ini disajikan secara jelas melalui elemen visual yang interaktif, seperti ikon, tombol *View All*, *View* Lulus, dan *View* Tidak Lulus untuk memudahkan pengguna dalam menelusuri data lebih detail. Di bagian bawah, terdapat visualisasi pohon keputusan hasil dari model Decision Tree dengan algoritma CART yang menampilkan alur pemisahan data berdasarkan atribut paling berpengaruh. Setiap node memperlihatkan nilai Gini, jumlah sampel, serta prediksi kelas pada masing-masing cabang, yang berguna untuk memahami logika klasifikasi model secara visual dan informatif.

4.3.3 Halaman Visualisasi

Halaman Visualisasi menampilkan grafik untuk menunjukkan tingkat kelulusan mahasiswa dari berbagai prodi yang ada di FTTI UNJAYA. Tampilan Halaman Visualisasi dapat dilihat pada Gambar 4.6.



Gambar 4. 6 Halaman Visualisasi

Pada Gambar 4.6 menampilkan halaman Visualisasi Prediksi Kelulusan yang menyajikan hasil prediksi dalam bentuk grafik untuk memudahkan pemahaman terhadap distribusi data. Terdapat tiga jenis visualisasi utama, yaitu *Pie Chart, Bar Chart*, dan grafik distribusi prediksi berdasarkan program studi. *Pie Chart* menunjukkan proporsi mahasiswa yang diprediksi lulus sebesar 94,1% dan tidak lulus sebesar 5,9%. *Bar Chart* menggambarkan jumlah mahasiswa dalam masing-masing kategori kelulusan secara lebih eksplisit. Sementara itu, grafik Distribusi Prediksi Kelulusan per Program Studi menampilkan sebaran jumlah mahasiswa lulus dan tidak lulus pada tiap program studi yang mengikuti mata kuliah Tugas Akhir.