PERBANDINGAN PERFORMA ALGORITMA *GRADIENT BOOSTING* UNTUK PREDIKSI AWAL KETEPATAN WAKTU KELULUSAN MAHASISWA BERBASIS DATA AKADEMIK HINGGA SEMESTER ENAM



Disusun Oleh:

N a m a : Rifandito Daniswara

NIM : 22523219

PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM INDONESIA

2026

HALAMAN PENGESAHAN DOSEN PEMBIMBING

PERBANDINGAN PERFORMA ALGORITMA *GRADIENT BOOSTING* UNTUK PREDIKSI AWAL KETEPATAN WAKTU KELULUSAN MAHASISWA BERBASIS DATA AKADEMIK HINGGA SEMESTER ENAM

TUGAS AKHIR

Disusun Oleh:

N a m a : Rifandito Daniswara

NIM : 22523219

Yogyakarta, 24 Juli 2026

Pembimbing,

(Chanifah Indah Ratnasari, S.Kom., M.Kom.)

HALAMAN PENGESAHAN DOSEN PENGUJI

PERBANDINGAN PERFORMA ALGORITMA *GRADIENT BOOSTING* UNTUK PREDIKSI AWAL KETEPATAN WAKTU KELULUSAN MAHASISWA BERBASIS DATA AKADEMIK HINGGA SEMESTER ENAM

TUGAS AKHIR

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Informatika – Program Sarjana di Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 24 Juli 2026

Tim Penguji

Chanifah I Ratnasari, S.Kom., M.Kom. _____

Anggota 1

Erika Ramadhani, S.T., M.Eng. _____

Anggota 2

Mukhammad Andri Setiawan, S.T., M.Sc., Ph.D. _____

Mengetahui,

Ketua Program Studi Informatika – Program Sarjana

Fakultas Teknologi Industri

Universitas Islam Indonesia

(Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D.)

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Rifandito Daniswara

NIM : 22523219

Tugas akhir dengan judul:

PERBANDINGAN PERFORMA ALGORITMA *GRADIENT BOOSTING* UNTUK PREDIKSI AWAL KETEPATAN WAKTU KELULUSAN MAHASISWA BERBASIS DATA AKADEMIK HINGGA SEMESTER ENAM

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila di kemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung risiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 24 Juli 2026



(Rifandito Daniswara)

HALAMAN PERSEMBAHAN

Alhamdulillah, Alhamdulillah, Alhamdulillah penulis ucapkan ke hadirat Allah Swt. atas segala limpahan rahmat serta hidayah-Nya, sehingga skripsi ini terselesaikan dengan baik. Selawat dan salam semoga selalu tercurahkan kepada junjungan Nabi besar Muhammad saw. yang telah memberikan bimbingan untuk menuju jalan yang Allah ridai. Dengan segala ketulusan dan keikhlasan, karya ini penulis persembahkan kepada:

1. Ayah (Hargijanto, S.E.) dan Ibu (Yanti Riftina, S.Pd.) tercinta, rida orang tua adalah rida Allah, puluhan tahun waktu telah berlalu, selama itu pula doa, kasih sayang, dan pengorbanan yang Ayah dan Ibu berikan untukku. Aku sadar bahwa sebanyak apapun usahaku untuk membalasnya, tidak akan pernah sebanding dengan apa yang telah Ayah dan Ibu berikan untukku. Meskipun hanya sedikit, aku berharap dengan terselesaikannya skripsi ini, bisa menjadi salah satu bentuk baktiku kepada Ayah dan Ibu tercinta. Semoga Allah selalu menyayangi dan memberikan yang terbaik bagi beliau berdua.
2. Kakak tersayang (Alm. Arieq Badhra Pranaya), Kak, meskipun kita belum pernah bertemu di dunia, aku tau kamu selalu mengawasiku di sisi-Nya, juga doa yang kamu panjatkan untuk keluarga kita. Dengan ini, kupersembahkan karya pertama ku untukmu.
3. Adik tersayang (Khayyiratri Anindita), terima kasih karena selalu memberiku motivasi dan doa untuk kesuksesan skripsi ini. Semoga Allah melimpahkan rahmat yang tiada henti-hentinya kepadamu.
4. Dosen Pembimbing, Ibu Chanifah Indah Ratnasari, S.Kom., M.Kom., terima kasih sebesar-besarnya karena telah bersedia meluangkan waktu untuk membimbing, menasehati, mencurahkan segenap waktu, tenaga, dan pikiran demi terselesaikannya skripsi ini.
5. Almamaterku Jurusan Informatika, Fakultas Teknologi Industri, Universitas Islam Indonesia yang ku banggakan.
6. Pribadiku, selamat telah menuntaskan salah satu ujianmu dengan baik. Tetaplah semangat dalam menghadapi ujian-ujian berikutnya. Jangan lari, hadapilah dengan sabar dan salat.

HALAMAN MOTO

“Hai orang-orang yang beriman, jadikanlah sabar dan salat sebagai penolongmu, sesungguhnya Allah beserta orang-orang yang sabar.” Q.S Al-Baqarah : 153

KATA PENGANTAR

Assalamu'alaikum warahmatullahi wabarakatuh.

Bismillahirrahmaanirrahiim, segala puji dan syukur penulis ucapkan ke hadirat Allah Swt. atas limpahan rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan tugas akhir ini dengan baik. Laporan skripsi ini berjudul "Perbandingan Performa Algoritma *Gradient Boosting* Untuk Prediksi Awal Ketepatan Waktu Kelulusan Mahasiswa Berbasis Data Akademik Hingga Semester Enam", yang disusun untuk memenuhi syarat perolehan gelar Sarjana (S-1) pada Program Studi Informatika, Fakultas Teknologi Industri, Universitas Islam Indonesia.

Dalam pelaksanaan penyusunan laporan ini, penulis mendapat banyak bantuan, bimbingan, dan arahan dari berbagai pihak. Oleh sebab itu, dalam kesempatan ini penulis ingin menyampaikan ucapan terima kasih yang tulus kepada:

1. Bapak Prof. Dr. Ir. Hari Purnomo, M.T., IPU., ASEAN Eng., selaku Rektor Universitas Islam Indonesia, beserta seluruh jajarannya.
2. Bapak Dr. Raden Teduh Dirgahayu, S.T., M.Sc., selaku Ketua Jurusan Informatika, beserta seluruh jajarannya.
3. Bapak Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D., selaku Ketua Program Studi Informatika – Program Sarjana UII.
4. Ibu Nurlatifah Risti Julianingrum, S.E., selaku Kepala Divisi Administrasi Akademik Fakultas Teknologi Industri, Universitas Islam Indonesia, yang telah berjasa dalam proses pengumpulan data akademik yang digunakan pada penelitian ini.
5. Ibu Chanifah Indah Ratnasari, S.Kom., M.Kom., selaku dosen pembimbing yang selalu sabar dalam membantu perjalanan penulis selama proses penyusunan skripsi ini hingga selesai.
6. Seluruh dosen pengajar dan staf prodi Informatika yang telah memberikan bekal ilmu dan bantuannya dalam proses belajar, semoga menjadi amal kebaikan Bapak/Ibu.
7. Pihak keluarga, terutama Ayah dan Ibu yang selalu memberikan semangat dan doa untuk penulis.
8. Seluruh pihak yang membantu dan tidak dapat penulis sebutkan satu-persatu, semoga amal kebaikan yang telah diberikan kepada penulis mendapat ganjaran kebaikan berlipat ganda dari Allah Swt.

Penulis menyadari bahwa dalam penyusunan laporan ini masih banyak kekurangan, sehingga masih jauh dari kesempurnaan. Oleh karena itu, penulis mengharapkan kritik dan saran yang bersifat membangun dari semua pihak. Semoga laporan skripsi ini bermanfaat bagi penulis pada khususnya, bagi dunia pendidikan dan pembaca pada umumnya.

Wassalamu'alaikum warahmatullahi wabarakatuh.

Yogyakarta, 24 Juli 2026



(Rifandito Daniswara)

SARI

Ketepatan waktu kelulusan mahasiswa merupakan salah satu indikator penting dalam evaluasi pendidikan tinggi karena berkaitan dengan efisiensi masa studi dan kinerja program studi. Penelitian ini bertujuan untuk membangun model prediksi awal ketepatan waktu kelulusan mahasiswa berdasarkan data akademik hingga semester enam menggunakan algoritma klasifikasi berbasis *gradient boosting*. Data yang digunakan berasal dari mahasiswa Program Studi Informatika Universitas Islam Indonesia angkatan 2019, 2020, dan 2021. Setelah proses pembersihan data, *dataset* yang digunakan berjumlah 370 data mahasiswa dengan distribusi kelas seimbang, yaitu 185 mahasiswa lulus terlambat dan 185 mahasiswa lulus tepat waktu. Tahapan penelitian meliputi *data cleaning*, *data splitting*, *feature engineering*, *feature selection* menggunakan korelasi Pearson, optimasi *hyperparameter*, optimasi *threshold*, evaluasi model, dan demonstrasi model menggunakan Streamlit. Model yang dibandingkan adalah *Histogram-Based Gradient Boosting* (HGB), *Extreme Gradient Boosting* (XGB), dan *Light Gradient Boosting Machine* (LGBM). Hasil evaluasi menunjukkan bahwa XGB memperoleh performa terbaik pada data uji dengan *accuracy* sebesar 0,7973, *precision* sebesar 0,8056, *recall* sebesar 0,7838, *macro F1-score* sebesar 0,7973, dan AUC-ROC sebesar 0,8145. Analisis *permutation feature importance* menunjukkan bahwa fitur beban SKS dan dinamika capaian IP berkontribusi terhadap hasil prediksi. Berdasarkan hasil tersebut, XGB dipilih sebagai model terbaik untuk prediksi awal ketepatan waktu kelulusan mahasiswa.

Kata kunci: prediksi kelulusan, *gradient boosting*, *feature engineering*, *feature selection*.

GLOSARIUM

Ensemble Learning
: Paradigma pembelajaran mesin yang menggabungkan prediksi dari beberapa model untuk menghasilkan satu prediksi akhir. Pendekatan ini diterapkan untuk meningkatkan akurasi, stabilitas, serta performa generalisasi model dibandingkan jika hanya menggunakan satu model tunggal.

Gradient Boosting
: Salah satu metode dalam *ensemble learning* yang membangun model prediksi secara bertahap dan berurutan. Metode ini bekerja dengan cara melatih model baru pada setiap iterasi untuk memperbaiki kesalahan atau residu prediksi dari model sebelumnya, dengan memanfaatkan arah gradien negatif dari fungsi kerugian (*loss function*) untuk meminimalkan kesalahan secara optimal.

Hyperparameter
: Sebuah parameter yang nilainya ditentukan sebelum proses pelatihan model dilakukan dan tidak dipelajari secara langsung dari data. Fungsinya adalah mengatur perilaku suatu algoritma pembelajaran mesin.

Instance
: Satu kesatuan unit data tunggal di dalam sebuah kumpulan data. Dalam bentuk tabel, *instance* diwakili oleh satu baris yang berisi nilai-nilai fitur dan label target untuk satu objek.

Loss Function
: Fungsi matematika yang digunakan untuk menghitung selisih atau besarnya kesalahan antara hasil prediksi model dengan nilai target yang sebenarnya.

Sparse
: Kondisi matriks atau tabel data yang sebagian besar elemen atau selnya bernilai nol atau tidak memiliki isian data (*missing value*).

DAFTAR ISI

HALAMAN JUDUL i
HALAMAN PENGESAHAN DOSEN PEMBIMBING ii
HALAMAN PENGESAHAN DOSEN PENGUJI iii
HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR iv
HALAMAN PERSEMBAHAN v
HALAMAN MOTO vi
KATA PENGANTAR vii
SARI ix
GLOSARIUM x
DAFTAR ISI xi
DAFTAR TABEL xiii
DAFTAR GAMBAR xiv
BAB I PENDAHULUAN 1
1.1 Latar Belakang 1
1.2 Rumusan Masalah 3
1.3 Batasan Masalah 4
1.4 Tujuan Penelitian 5
1.5 Manfaat Penelitian 5
1.6 Metodologi Penelitian 6
1.7 Sistematika Penulisan 7
BAB II LANDASAN TEORI 9
2.1 *Educational Data Mining* 9
2.2 Prediksi Dini Ketepatan Waktu Kelulusan Mahasiswa 10
2.3 Data Akademik 12
2.3.1 *Data Collection* 13
2.3.2 *Labeling* 13
2.4 *Feature Engineering* 14
2.4.1 Pembentukan Fitur Perubahan Antarperiode 15
2.4.2 Pembentukan Fitur Ringkasan Statistik 15
2.5 *Feature Selection* 15
2.6 *Gradient Boosting* 17
2.6.1 *Extreme Gradient Boosting* 18
2.6.2 *Histogram-based Gradient Boosting* 19
2.6.3 *Light Gradient Boosting Machine* 20
2.7 Pemilihan *Hyperparameter* Menggunakan *Random Search* 21
2.7.1 Mekanisme *Random Search* 22
2.7.2 Parameter Kunci dalam *Pipeline Gradient Boosting* 22
2.7.3 Strategi Pemilihan *Hyperparameter* dalam *Pipeline* 24
2.8 Teknik Evaluasi 24
2.8.1 *Confusion Matrix* 25
2.8.2 Metrik Evaluasi 26
2.8.3 Evaluasi Kinerja dengan *Cross-validation* 27
2.8.4 Optimasi *Threshold* 28
2.8.5 AUC-ROC 30
2.8.6 *Permutation Feature Importance* 31
2.9 Penelitian Terdahulu 32
2.10 Kerangka Pemikiran 38

BAB III METODOLOGI PENELITIAN 40
3.1 Alur Penelitian 40
3.2 Lingkungan Penelitian 41
3.3 *Data Collection and Labeling* 42
3.3.1 *Data Collection* 42
3.3.2 *Labeling* 43
3.4 *Data Cleaning* 45
3.5 *Data Splitting* 46
3.5.1 Skenario Data Gabungan 46
3.5.2 Skenario Data Berdasarkan Angkatan 47
3.6 *Feature Engineering* 47
3.7 *Feature Selection* 49
3.7.1 Analisis Korelasi Awal 49
3.7.2 Penentuan *Threshold* Korelasi Awal 50
3.7.3 Mekanisme *Feature Selection* 51
3.7.4 Implementasi dalam *Pipeline* 52
3.7.5 Integrasi dengan *Random Search* 52
3.8 *Modeling* dengan *Gradient Boosting* 52
3.9 Pemilihan *Hyperparameter* 53
3.10 Evaluasi Model 56
3.10.1 Evaluasi Utama pada Data Gabungan 56
3.10.2 Evaluasi Tambahan pada Data Berdasarkan Angkatan 59
BAB IV HASIL DAN PEMBAHASAN 61
4.1 Hasil *Data Collection and Labeling* 61
4.2 Hasil *Data Cleaning* 62
4.3 Hasil *Data Splitting* 64
4.3.1 Hasil pada Data Gabungan 65
4.3.2 Hasil pada Data Berdasarkan Angkatan 66
4.4 Hasil *Feature Engineering* 66
4.5 Hasil *Feature Selection* 68
4.6 Hasil *Modeling* dan Optimasi *Hyperparameter* 72
4.7 Hasil Evaluasi Model 76
4.7.1 Hasil Evaluasi Utama pada Data Gabungan 76
4.7.2 Hasil Evaluasi Tambahan pada Data Berdasarkan Angkatan 90
4.8 Demonstrasi Model berbasis Streamlit 93
4.9 Pembahasan Temuan 95
4.10 Keterbatasan Penelitian 100
BAB V KESIMPULAN DAN SARAN 102
5.1 Kesimpulan 102
5.2 Saran 103
DAFTAR PUSTAKA 106
LAMPIRAN 111

DAFTAR TABEL

Tabel 2.1 Tabel *Confusion Matrix* 25
Tabel 2.2 Penelitian Terdahulu 33
Tabel 3.1 Perangkat Lunak dan Pustaka Pendukung 41
Tabel 3.2 Cakupan Data Akademik per Angkatan 43
Tabel 3.3 Atribut *Dataset* Setelah *Labeling* dan Pembatasan Semester 45
Tabel 3.4 Ruang Pencarian *Hyperparameter* 54
Tabel 4.1 Persebaran Label Data Awal Berdasarkan Angkatan 61
Tabel 4.2 Hasil Penyaringan Label pada *Dataset* 63
Tabel 4.3 Distribusi Label Setelah *Data Cleaning* 63
Tabel 4.4 Persentase *Missing Values* pada *Dataset* 64
Tabel 4.5 Distribusi Label pada Skenario Data Gabungan 65
Tabel 4.6 Distribusi Label pada Skenario Data Berdasarkan Angkatan 66
Tabel 4.7 Hasil *Feature Engineering* 68
Tabel 4.8 Perbandingan Hasil *Feature Selection* pada Setiap Model 70
Tabel 4.9 Hasil Optimasi *Hyperparameter* pada Setiap Model 74
Tabel 4.10 Hasil *Best CV Score* pada Setiap Model 75
Tabel 4.11 Perbandingan *Train* dan *Cross-validation* pada Setiap Model 77
Tabel 4.12 Hasil Evaluasi Awal dengan *Threshold* 0,50 79
Tabel 4.13 *Confusion Matrix* Evaluasi Awal dengan *Threshold* 0,50 79
Tabel 4.14 Hasil Optimasi *Threshold* Menggunakan Data Latih 82
Tabel 4.15 Hasil Evaluasi Akhir dengan *Threshold* Terbaik 83
Tabel 4.16 *Confusion Matrix* Evaluasi Akhir dengan *Threshold* Terbaik 83
Tabel 4.17 Perubahan Setelah Optimasi *Threshold* 83
Tabel 4.18 Hasil AUC-ROC pada Setiap Model 85
Tabel 4.19 Hasil *Permutation Feature Importance* pada Setiap Model 88
Tabel 4.20 Ringkasan Perbandingan Performa Model 90
Tabel 4.21 Hasil Evaluasi Tambahan XGB pada Data Berdasarkan Angkatan 91
Tabel 4.22 *Confusion Matrix* Evaluasi Tambahan XGB pada Data Berdasarkan Angkatan 92

DAFTAR GAMBAR

Gambar 2.1 Ilustrasi grafik kurva AUC-ROC 31
Gambar 3.1 Alur penelitian 40
Gambar 3.2 *Correlation Heatmap* dari *dataset* 50
Gambar 4.1 Kode *label filtering* dan *feature dropping* pada *Dataset* Awal 63
Gambar 4.2 Kode Implementasi *Feature Engineering* dalam Sebuah Kelas 67
Gambar 4.3 Kode Implementasi *Feature Selection* dalam Sebuah Kelas 70
Gambar 4.4 Kode Implementasi *Modeling* dan *Hyperparameter Tuning* 73
Gambar 4.5 Kode Evaluasi Kinerja *Train* dan *Cross-validation* 77
Gambar 4.6 Kode Evaluasi Awal dengan *Threshold* 0.50 79
Gambar 4.7 Kode Optimasi *Threshold* Menggunakan *Out-of-Fold Probability* 81
Gambar 4.8 Kode Analisis AUC-ROC 85
Gambar 4.9 Kurva ROC pada Model XGB 86
Gambar 4.10 Kode Analisis *Feature Importance* 87
Gambar 4.11 Kode Pemanggilan Model pada Streamlit 93
Gambar 4.12 Tampilan Form Input Data 94
Gambar 4.13 Tampilan Hasil Prediksi 95

BAB I
PENDAHULUAN

1.1 Latar Belakang

Program Studi Informatika pada Program Sarjana di Universitas Islam Indonesia (UII) telah beroperasi sejak tahun 1994 dan memiliki rekam jejak penyelenggaraan pendidikan tinggi di bidang informatika selama lebih dari tiga dekade. Dalam periode tersebut, program studi ini terus menghadapi kebutuhan untuk menjaga kualitas penyelenggaraan akademik, salah satunya melalui pemenuhan standar akreditasi. Saat ini, Program Studi Informatika pada Program Sarjana UII memperoleh status Akreditasi Unggul dari LAM INFOKOM yang berlaku hingga 13 Desember 2029 (LAM INFOKOM, 2024). Pencapaian akreditasi unggul tersebut tidak bersifat statis, melainkan menuntut upaya berkelanjutan dalam menjaga dan meningkatkan kualitas penyelenggaraan pendidikan. Proses akreditasi ulang yang dilakukan secara berkala mengharuskan program studi untuk secara konsisten memenuhi standar penilaian yang ditetapkan, termasuk memperbarui data kinerja dan menunjukkan perbaikan yang terukur dari periode sebelumnya. Salah satu aspek yang secara langsung dinilai dalam proses akreditasi adalah kinerja kelulusan mahasiswa, yang mencerminkan efektivitas proses pembelajaran dan layanan akademik yang diberikan oleh program studi. Kinerja kelulusan yang tidak terpantau dengan baik berisiko menurunkan nilai akreditasi pada periode penilaian berikutnya, sehingga pemantauan dan antisipasi dini terhadap keterlambatan kelulusan menjadi hal yang penting untuk dilakukan.

Dalam pedoman akreditasi nomor 13 tahun 2025 yang diterbitkan oleh Badan Akreditasi Nasional Perguruan Tinggi (BAN-PT) untuk program sarjana, disebutkan bahwa perguruan tinggi yang hendak memperoleh status akreditasi Unggul harus memenuhi tolak ukur kelulusan tertentu (Badan Akreditasi Nasional Perguruan Tinggi, 2025, Lampiran 3g). Hal ini mencakup persentase mahasiswa yang masuk pada tiga tahun sebelum tahun evaluasi dan lulus tepat waktu dalam satu masa tempuh kurikulum (PK1MTK) $\geq$ 45%, proporsi lulusan yang menyelesaikan studi dalam 1,5 masa tempuh kurikulum terhadap total lulusan dalam tiga tahun terakhir (RPK1,5MTK) $\leq$ 30%, dan persentase lulusan dalam dua masa tempuh kurikulum dari mahasiswa yang masuk enam tahun sebelum tahun evaluasi (PK2MTK) $\geq$ 75%. Indikator-indikator ini merupakan bagian dari penilaian Efektivitas Kinerja Program Studi yang secara langsung memengaruhi hasil akreditasi institusi. Oleh karena itu, pemantauan terhadap pola kelulusan mahasiswa secara proaktif menjadi hal yang perlu diperhatikan oleh program studi.

Berdasarkan data akademik Program Studi Informatika UII, dari tiga angkatan yang menjadi objek penelitian ini, diperoleh gambaran mengenai pola kelulusan mahasiswa. Perlu dicatat bahwa data yang diperoleh dari pihak administrasi akademik belum memiliki label status kelulusan, sehingga pelabelan dilakukan secara manual menggunakan formula yang mengacu pada kriteria kelulusan yang ditetapkan oleh program studi. Selain itu, analisis hanya melibatkan mahasiswa yang telah memperoleh label tepat waktu atau terlambat berdasarkan hasil pelabelan tersebut, sementara mahasiswa yang masih aktif atau belum memenuhi kriteria kelulusan tidak diikutsertakan. Perlu diketahui juga bahwa nilai yang tidak tersedia pada beberapa atribut data tidak diperlakukan sebagai kesalahan, melainkan dipertahankan sebagai informasi yang bermakna dalam proses pemodelan. Data yang tersedia mencakup rekam jejak akademik hingga semester tiga belas untuk angkatan 2019, semester sebelas untuk angkatan 2020, dan semester sembilan untuk angkatan 2021, sehingga status kelulusan setiap mahasiswa dapat ditentukan berdasarkan posisi mereka relatif terhadap batas masa tempuh kurikulum pada masing-masing titik waktu tersebut. Dari mahasiswa yang telah menyelesaikan studi, angkatan 2019 mencatat 57 mahasiswa lulus tepat waktu dan 86 mahasiswa lulus terlambat, angkatan 2020 mencatat 47 mahasiswa lulus tepat waktu dan 87 mahasiswa lulus terlambat, sedangkan angkatan 2021 mencatat 81 mahasiswa lulus tepat waktu dan 12 mahasiswa lulus terlambat. Kondisi ini menunjukkan bahwa keterlambatan kelulusan merupakan fenomena yang nyata terjadi di program studi, khususnya terlihat pada angkatan 2019 dan 2020 yang rekam jejaknya sudah lebih lengkap, sementara pola angkatan 2021 belum dapat disimpulkan secara menyeluruh mengingat data yang tersedia baru mencakup hingga semester sembilan. Oleh karena itu, kemampuan mengidentifikasi mahasiswa yang berisiko terlambat sejak dini menjadi kebutuhan yang relevan bagi program studi dalam upaya mempertahankan standar indikator kelulusan yang ditetapkan BAN-PT. Pendekatan berbasis *machine learning* menawarkan potensi untuk menjawab kebutuhan tersebut, dengan memanfaatkan data akademik yang sudah tersedia untuk membangun model prediksi yang dapat memberikan informasi awal mengenai risiko keterlambatan kelulusan mahasiswa.

Mengingat karakteristik data akademik mahasiswa yang digunakan dalam penelitian ini mencakup 370 data setelah *preprocessing*, pendekatan *machine learning* konvensional dipilih sebagai metode pemodelan yang lebih sesuai dibandingkan *deep learning*. Penelitian terdahulu menunjukkan bahwa model *deep learning* umumnya membutuhkan data dalam jumlah besar untuk dapat belajar secara efektif, sementara pada *dataset* berskala kecil hingga menengah, algoritma *machine learning* cenderung memberikan performa yang lebih stabil dan tidak rentan

terhadap *overfitting* (Pawitra dkk., 2024). Dalam konteks ini, algoritma berbasis *gradient boosting* dipilih karena kemampuannya menangani *missing value* secara *native* tanpa memerlukan imputasi terlebih dahulu, sehingga nilai yang tidak tersedia dalam data akademik tidak harus dihilangkan, melainkan dapat diperlakukan sebagai informasi yang bermakna dalam proses pembelajaran model (Aydin & Ozturk, 2021; Chen & Guestrin, 2016). Lebih lanjut, penelitian ini membandingkan tiga varian *gradient boosting* yaitu *Histogram-Based Gradient Boosting* (Pedregosa dkk., 2011), *Extreme Gradient Boosting* (Chen & Guestrin, 2016), dan *Light Gradient Boosting Machine* (Ke dkk., 2017). Ketiganya dipilih karena memiliki karakteristik implementasi yang berbeda dalam hal strategi *binning* dan regularisasi, sehingga perlu dievaluasi untuk menentukan varian mana yang paling sesuai dengan karakteristik data akademik yang digunakan. Dengan demikian, penelitian ini bertujuan membandingkan performa ketiga algoritma *gradient boosting* tersebut dalam memprediksi ketepatan waktu kelulusan mahasiswa, sehingga dapat memberikan rekomendasi model yang paling sesuai sebagai dasar sistem pendukung keputusan akademik di Program Studi Informatika UII.

1.2 Rumusan Masalah

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

a. Bagaimana penerapan algoritma klasifikasi berbasis *gradient boosting* dalam memprediksi ketepatan waktu kelulusan mahasiswa berdasarkan data akademik hingga semester enam?
b. Bagaimana performa dan hasil evaluasi model prediksi ketepatan waktu kelulusan mahasiswa saat diuji menggunakan metrik klasifikasi, seperti *accuracy*, *precision*, *recall*, *macro F1-score*, *confusion matrix*, dan AUC-ROC?
c. Bagaimana perbandingan performa algoritma *Histogram-Based Gradient Boosting* (HGB), *Extreme Gradient Boosting* (XGB), dan *Light Gradient Boosting Machine* (LGBM) dalam konteks prediksi awal ketepatan waktu kelulusan mahasiswa?
d. Bagaimana kontribusi fitur akademik terhadap hasil prediksi pada model terbaik berdasarkan analisis menggunakan *permutation feature importance*?

1.3 Batasan Masalah

Agar penelitian lebih terarah dan sesuai dengan tujuan yang telah ditetapkan, maka batasan masalah dalam penelitian ini adalah sebagai berikut:

a. Data yang digunakan merupakan data akademik mahasiswa Program Studi Informatika Program Sarjana Universitas Islam Indonesia pada tiga angkatan yang menjadi objek penelitian.
b. Data akademik yang digunakan dalam proses pemodelan dibatasi hingga semester enam, sehingga prediksi yang dilakukan bersifat prediksi awal sebelum mahasiswa memasuki masa akhir studi.
c. Atribut yang digunakan dalam penelitian ini dibatasi pada data akademik mahasiswa, khususnya indeks prestasi semester, indeks prestasi kumulatif, dan jumlah satuan kredit semester yang ditempuh pada setiap semester.
d. Data mentah yang diperoleh tidak memiliki label target secara eksplisit, sehingga penentuan label kelas dilakukan secara manual (*manual labeling*) berdasarkan aturan kelulusan resmi Program Studi Informatika UII yang dikonversi ke dalam logika sistem.
e. Penelitian ini difokuskan pada prediksi ketepatan waktu kelulusan mahasiswa dalam bentuk klasifikasi biner, yaitu lulus tepat waktu dan lulus terlambat. Mahasiswa dengan status lain, seperti masih aktif, belum lulus, atau tidak termasuk ke dalam dua kelas tersebut, tidak digunakan dalam proses pemodelan.
f. Algoritma yang digunakan dalam penelitian ini dibatasi pada tiga algoritma klasifikasi berbasis *gradient boosting*, yaitu *Histogram-Based Gradient Boosting* (HGB), *Extreme Gradient Boosting* (XGB), dan *Light Gradient Boosting Machine* (LGBM).
g. Evaluasi model dilakukan menggunakan beberapa metrik klasifikasi, yaitu *accuracy*, *precision*, *recall*, *macro F1-score*, *confusion matrix*, dan AUC-ROC. Analisis kontribusi fitur terhadap model terbaik dilakukan menggunakan *permutation feature importance* sebagai analisis tambahan.
h. Penelitian ini berfokus pada pengembangan dan evaluasi model prediksi, sehingga implementasi antarmuka aplikasi hanya digunakan sebagai media demonstrasi dan bukan menjadi fokus utama penelitian.
i. Penelitian ini tidak membahas penerapan model secara langsung dalam sistem akademik operasional maupun perancangan intervensi akademik bagi mahasiswa yang diprediksi berisiko terlambat lulus.

1.4 Tujuan Penelitian

Secara umum, penelitian ini bertujuan untuk mengembangkan model *machine learning* untuk memprediksi ketepatan waktu kelulusan mahasiswa berdasarkan data akademik hingga semester enam. Adapun secara khusus, tujuan penelitian ini adalah sebagai berikut:

a. Membangun model prediksi awal ketepatan waktu kelulusan mahasiswa berdasarkan data akademik hingga semester enam menggunakan algoritma klasifikasi berbasis *gradient boosting*.
b. Mengevaluasi kinerja model prediksi ketepatan waktu kelulusan mahasiswa menggunakan metrik klasifikasi, seperti *accuracy*, *precision*, *recall*, *macro F1-score, confusion matrix*, dan AUC-ROC.
c. Membandingkan performa algoritma *Histogram-Based Gradient Boosting* (HGB), *Extreme Gradient Boosting* (XGB), dan *Light Gradient Boosting Machine* (LGBM) dalam konteks prediksi awal ketepatan waktu kelulusan mahasiswa.
d. Menganalisis kontribusi fitur akademik terhadap hasil prediksi pada model terbaik menggunakan *permutation feature importance*.

1.5 Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan manfaat sebagai berikut:

a. Manfaat Teoritis

   Penelitian ini diharapkan dapat menjadi referensi dalam pengembangan penelitian di bidang prediksi ketepatan waktu kelulusan mahasiswa menggunakan pendekatan *machine learning*, khususnya algoritma berbasis *gradient boosting*. Selain itu, penelitian ini juga diharapkan dapat memberikan gambaran mengenai penerapan tahapan pemodelan, evaluasi performa, dan analisis kontribusi fitur pada data akademik mahasiswa.

b. Manfaat Praktis

   1. Bagi Program Studi Informatika Universitas Islam Indonesia, penelitian ini diharapkan dapat memberikan gambaran awal mengenai potensi pemanfaatan data akademik hingga semester enam untuk memprediksi ketepatan waktu kelulusan mahasiswa.
   2. Bagi pengelola akademik, hasil penelitian ini diharapkan dapat menjadi bahan pertimbangan awal dalam mendukung proses pemantauan akademik mahasiswa, khususnya dalam mengidentifikasi kemungkinan risiko keterlambatan kelulusan berdasarkan hasil prediksi model.

3. Bagi peneliti selanjutnya, penelitian ini diharapkan dapat menjadi referensi dalam pengembangan model prediksi kelulusan mahasiswa, baik melalui penggunaan atribut akademik yang lebih beragam, metode pemodelan yang berbeda, maupun pengujian pada data dari program studi atau institusi lain.
4. Implementasi antarmuka aplikasi dalam penelitian ini diharapkan dapat menjadi contoh demonstrasi sederhana mengenai bagaimana model prediksi dapat disajikan kepada pengguna, tanpa dimaksudkan sebagai sistem akademik operasional yang siap diterapkan secara langsung.

1.6 Metodologi Penelitian

Penelitian ini dilakukan melalui beberapa tahapan untuk memastikan proses penelitian berjalan secara sistematis dan terstruktur. Adapun tahapan penelitian yang dilakukan adalah sebagai berikut:

a. *Data Collection and Labeling*

   Tahap pengumpulan data dilakukan dengan memperoleh data akademik mahasiswa Program Studi Informatika Universitas Islam Indonesia melalui pihak administrasi akademik Fakultas Teknologi Industri Universitas Islam Indonesia.

b. *Data Cleaning*

   Tahap *cleaning* dilakukan untuk mempersiapkan data sebelum digunakan dalam proses pemodelan. Proses yang dilakukan meliputi penghapusan atribut yang tidak relevan serta proses *label filtering* sesuai dengan kebutuhan penelitian.

c. *Data Splitting*

   Tahap *data splitting* dilakukan menggunakan metode *stratified split* untuk membagi *dataset* menjadi data pelatihan dan data pengujian dengan proporsi 80:20. Metode ini digunakan agar distribusi kelas pada data pelatihan dan data pengujian tetap seimbang, mempertahankan rasio asli dari keseluruhan *dataset*.

d. *Feature Engineering*

   Tahap *feature engineering* dilakukan dengan membentuk fitur turunan berdasarkan data akademik mahasiswa. Fitur turunan dibentuk untuk merepresentasikan pola perkembangan akademik mahasiswa antarsemester sehingga dapat mendukung proses pemodelan.

e. *Feature Selection*

Tahap *feature selection* dilakukan untuk memilih fitur yang digunakan dalam proses pemodelan berdasarkan analisis korelasi Pearson. Proses ini dilakukan untuk mengurangi redundansi antarfitur dan mempertahankan fitur yang dianggap relevan terhadap proses prediksi.

f. *Modeling*

Tahap *modeling* dilakukan menggunakan algoritma *Histogram-Based Gradient Boosting* (HGB), *Extreme Gradient Boosting* (XGB), dan *Light Gradient Boosting Machine* (LGBM). Ketiga algoritma tersebut digunakan untuk membangun model prediksi ketepatan waktu kelulusan mahasiswa berdasarkan data akademik hingga semester enam.

g. *Hyperparameter Optimization*

Tahap pemilihan *hyperparameter* dilakukan menggunakan *Randomized Search Cross Validation* untuk memperoleh kombinasi parameter terbaik pada masing-masing algoritma.

h. *Model evaluation*

Tahap evaluasi dilakukan untuk mengukur performa model menggunakan metrik klasifikasi, seperti *accuracy*, *precision*, *recall*, *macro F1-score*, *confusion matrix*, dan AUC-ROC. Pada tahap ini, model terlebih dahulu dievaluasi menggunakan *threshold* bawaan sebesar 0,50 sebagai dasar. Selanjutnya, dilakukan optimasi *threshold* menggunakan probabilitas *out-of-fold* pada data latih untuk menentukan batas klasifikasi yang lebih sesuai. *Threshold* yang terpilih kemudian digunakan pada evaluasi akhir terhadap data uji. Hasil evaluasi digunakan untuk membandingkan performa masing-masing algoritma yang digunakan dalam penelitian.

i. *Model demonstration*

Tahap akhir dilakukan dengan menampilkan implementasi sederhana model prediksi menggunakan antarmuka berbasis Streamlit sebagai media demonstrasi penggunaan model prediksi ketepatan waktu kelulusan mahasiswa.

1.7 Sistematika Penulisan

Untuk memberikan gambaran umum mengenai penelitian ini, penulisan disusun dalam lima bab yang saling berkaitan dengan sistematika sebagai berikut:

a. BAB I PENDAHULUAN

Bab ini berisi latar belakang penelitian yang membahas ketepatan waktu kelulusan mahasiswa serta perannya dalam penilaian akreditasi perguruan tinggi. Selain itu, dibahas

pula tantangan dalam mempertahankan rasio kelulusan tepat waktu serta pemanfaatan *machine learning* untuk prediksi ketepatan waktu kelulusan mahasiswa. Penelitian ini juga menyoroti karakteristik data akademik yang memiliki nilai kosong (*missing value*), di mana *missing value* tidak selalu merepresentasikan kesalahan data, tetapi dapat mengandung informasi akademik tertentu. Oleh karena itu, penelitian ini menggunakan algoritma *gradient boosting* yang mampu menangani *missing value* secara *native* dalam proses pemodelan. Bab ini juga memuat rumusan masalah, tujuan penelitian, batasan masalah, manfaat penelitian, metodologi penelitian, dan sistematika penulisan.

b. BAB II LANDASAN TEORI

Bab ini menyajikan teori-teori yang menjadi landasan penelitian, mencakup *Educational Data Mining*, konsep prediksi ketepatan waktu kelulusan mahasiswa, data akademik, *feature engineering*, *feature selection*, algoritma *gradient boosting* seperti *Histogram-Based Gradient Boosting* (HGB), *Extreme Gradient Boosting* (XGB), dan *Light Gradient Boosting Machine* (LGBM), pemilihan *hyperparameter*, teknik evaluasi klasifikasi, serta tinjauan penelitian terdahulu yang berkaitan dengan prediksi kelulusan mahasiswa menggunakan pendekatan *machine learning*.

c. BAB III METODOLOGI PENELITIAN

Bab ini berisi kerangka kerja penelitian yang mencakup *data collection and labeling*, *data cleaning*, *data splitting*, *feature engineering*, *feature selection*, pemilihan serta konfigurasi algoritma HGB, XGB, dan LGBM, pemilihan *hyperparameter*, serta metode evaluasi yang digunakan.

d. BAB IV HASIL DAN PEMBAHASAN

Bab ini berisi hasil *data collection and labeling*, *data cleaning*, *data splitting*, *feature engineering*, *feature selection*, *modeling* dan pemilihan *hyperparameter*, evaluasi model, perbandingan performa algoritma HGB, XGB, dan LGBM, pembahasan temuan, demo penerapan menggunakan Streamlit, serta keterbatasan penelitian.

e. BAB V KESIMPULAN DAN SARAN

Bab ini berisi kesimpulan yang diperoleh dari hasil penelitian terkait efektivitas metode *gradient boosting* untuk prediksi ketepatan waktu kelulusan mahasiswa, serta saran-saran untuk penelitian selanjutnya.

BAB II
LANDASAN TEORI

Bab ini menyusun landasan teori yang menjadi dasar bagi perancangan dan analisis model prediksi ketepatan waktu kelulusan mahasiswa. Pembahasan diawali dengan *Educational Data Mining* sebagai bidang yang berkaitan dengan pemanfaatan data pendidikan untuk menemukan pola, membangun model prediktif, dan mendukung proses pengambilan keputusan akademik. Selanjutnya, bab ini membahas konsep prediksi ketepatan waktu kelulusan mahasiswa serta ruang lingkup data akademik yang digunakan dalam penelitian. Aspek pengolahan data dijelaskan melalui tahapan *data collection, labeling, feature engineering*, dan *feature selection* untuk memastikan data yang digunakan sesuai dengan kebutuhan pemodelan. Fokus kemudian diarahkan pada algoritma *gradient boosting* yang digunakan, yaitu *Histogram-Based Gradient Boosting* (HGB), *Extreme Gradient Boosting* (XGB), dan *Light Gradient Boosting Machine* (LGBM), beserta proses pemilihan *hyperparameter* dan optimasi *threshold*. Selain itu, bab ini juga menjelaskan metrik evaluasi klasifikasi yang digunakan untuk menilai performa model. Pada bagian akhir, disajikan penelitian terdahulu yang relevan sebagai landasan dalam merancang metodologi penelitian pada bab berikutnya.

2.1 *Educational Data Mining*

Educational Data Mining (EDM) merupakan bidang yang menerapkan metode *data mining* untuk mengeksplorasi data yang berasal dari lingkungan pendidikan. Romero & Ventura (2010) menjelaskan bahwa EDM digunakan untuk menganalisis data pendidikan dalam rangka menemukan pola, membangun model, dan menghasilkan pengetahuan yang relevan terhadap permasalahan pendidikan. Dalam konteks pendidikan tinggi, data yang dianalisis dapat berupa data akademik mahasiswa, seperti capaian indeks prestasi, jumlah kredit yang ditempuh, dan status kelulusan.

Secara umum, penerapan EDM dapat diarahkan pada beberapa tujuan analisis, seperti memahami pola perilaku atau capaian mahasiswa, mengelompokkan mahasiswa berdasarkan karakteristik tertentu, serta memprediksi kondisi akademik mahasiswa di masa mendatang. Pada pendekatan deskriptif, EDM digunakan untuk menemukan pola atau karakteristik yang terdapat pada data pendidikan tanpa secara langsung membangun model prediksi terhadap kelas target tertentu. Pendekatan ini dapat digunakan, misalnya, untuk mengidentifikasi pola

umum performa akademik mahasiswa atau mengelompokkan mahasiswa berdasarkan kemiripan karakteristik akademiknya.

Berbeda dengan pendekatan deskriptif, pendekatan prediktif dalam EDM bertujuan membangun model yang mampu memperkirakan keluaran tertentu berdasarkan data historis. Dalam konteks pendidikan tinggi, pendekatan prediktif dapat digunakan untuk memperkirakan risiko akademik, keberhasilan studi, capaian performa mahasiswa, maupun ketepatan waktu kelulusan. Apabila keluaran yang diprediksi berupa kategori atau kelas, maka permasalahan tersebut dapat diformulasikan sebagai tugas klasifikasi. Pendekatan klasifikasi umumnya menggunakan *supervised learning*, yaitu model dilatih menggunakan data yang telah memiliki label target sehingga dapat mempelajari hubungan antara fitur input dan kelas keluaran. Penerapan pendekatan prediktif ini terbukti efektif dalam membantu institusi memantau performa akademik mahasiswa pada tiap semester secara berkelanjutan (Diekuu dkk., 2025).

Penelitian ini mengadopsi pendekatan prediktif berbasis klasifikasi *supervised* karena tujuan utama penelitian adalah memprediksi status ketepatan waktu kelulusan mahasiswa berdasarkan data akademik historis. Dalam penelitian ini, status kelulusan mahasiswa diformulasikan sebagai dua kelas keluaran, yaitu lulus tepat waktu dan lulus terlambat. Penggunaan data akademik historis untuk memprediksi status kelulusan pada berbagai tahapan studi ini sejalan dengan penelitian yang menunjukkan bahwa model klasifikasi mampu mendeteksi risiko kelulusan mahasiswa sejak dini (Fernandez-Garcia dkk., 2021).

Selanjutnya, Romero & Ventura (2020) memperluas pembahasan EDM dengan menempatkannya dalam siklus penemuan pengetahuan (*knowledge discovery*) yang mencakup data pendidikan, *data pre-processing*, penerapan metode analisis, serta interpretasi hasil untuk menghasilkan pengetahuan baru. Kerangka tersebut menjadi dasar bahwa penelitian EDM tidak hanya berfokus pada pemilihan algoritma, tetapi juga memerlukan rangkaian tahapan pendukung seperti pembentukan fitur, seleksi fitur, pemilihan *hyperparameter*, penyesuaian ambang keputusan klasifikasi, serta pemilihan metrik evaluasi yang sesuai dengan tujuan penelitian.

2.2 Prediksi Dini Ketepatan Waktu Kelulusan Mahasiswa

Sebagaimana telah diuraikan pada subbab Latar Belakang, ketepatan waktu kelulusan merupakan salah satu indikator mutu yang berdampak pada penilaian akreditasi program studi. Kelulusan tepat waktu didefinisikan sebagai kondisi di mana mahasiswa menyelesaikan seluruh beban studi dalam batas waktu yang ditetapkan oleh institusi pendidikan tinggi.

Berdasarkan Peraturan BAN-PT Nomor 13 Tahun 2025 tentang IAPS 5.0, masa studi normal program sarjana adalah 8 semester atau 4 tahun (Badan Akreditasi Nasional Perguruan Tinggi, 2025). Mahasiswa yang lulus melebihi ketentuan waktu tersebut dikategorikan sebagai lulus terlambat.

Sejumlah penelitian telah dilakukan untuk memprediksi status kelulusan mahasiswa menggunakan pendekatan *machine learning*. Dwinanda dkk. (2023) menggunakan algoritma XGBoost untuk mengklasifikasi status kelulusan mahasiswa Program Studi Statistika Universitas Tanjungpura menggunakan 140 data numerik dengan IPK semester 1–4 sebagai salah satu fitur akademik, dan memperoleh akurasi sebesar 75% dengan sensitivitas 76,19%. Pawitra dkk. (2024) membandingkan sembilan algoritma *machine learning* untuk memprediksi kelulusan tepat waktu mahasiswa Prodi Pendidikan Teknik Elektronika dan Informatika di salah satu universitas negeri di Yogyakarta, menggunakan 133 data mahasiswa dengan 51 fitur yang mencakup variabel demografis dan IPK semester 1–8. Dengan menangani ketidakseimbangan kelas menggunakan SMOTE dan menjadikan AUC sebagai metrik evaluasi utama, algoritma Random Forest terpilih sebagai model terbaik dengan AUC 0.875. Rachardian dan Sediyono (2024) menggunakan metode k-Nearest Neighbors untuk memprediksi kelulusan tepat waktu mahasiswa S1 dalam rangka pemantauan program studi sesuai standar akreditasi, menggunakan data mahasiswa angkatan 2019 di Universitas PQR dengan IPK semester 1–8 sebagai salah satu fitur akademik, dan memperoleh akurasi pengujian sebesar 87,76% serta presisi 89,19%. Bakri dkk. (2025) menginvestigasi pengaruh teknik *resampling* dan *hyperparameter tuning* terhadap prediksi kelulusan tepat waktu menggunakan *dataset* 4.093 mahasiswa STIEM Bongaya dengan 15 variabel prediktor termasuk IPK semester 1–4. Penelitian tersebut menemukan bahwa kombinasi *resampling* RUS-ROS dengan algoritma Random Forest menghasilkan performa terbaik dengan *F1-score* 0.942, sekaligus menegaskan bahwa penanganan ketidakseimbangan kelas secara tepat dapat berkontribusi signifikan terhadap performa model.

Dari penelitian-penelitian tersebut, justifikasi pemilihan titik prediksi berdasarkan struktur kurikuler yang terdokumentasi belum teridentifikasi secara eksplisit: penelitian yang menggunakan IPK semester 1–4 belum secara eksplisit mengaitkan pemilihan titik prediksi tersebut dengan struktur kurikulum program studi, sementara penelitian yang menggunakan IPK semester 1–8 cenderung bersifat retrospektif karena prediksi baru dapat dilakukan setelah data akademik hingga semester 8 tersedia. Penelitian ini mengisi gap tersebut dengan menetapkan semester 6 sebagai batas fitur berdasarkan struktur kurikulum Program Studi

Informatika UII yang terdokumentasi, di mana model dibangun menggunakan fitur akademik semester 1–6.

Secara spesifik, Panduan Akademik Informatika UII Tahun 2021/2022 menunjukkan bahwa mahasiswa yang menempuh beban SKS secara optimal pada setiap semester akan menyisakan Kuliah Kerja Nyata, Tugas Akhir, dan Karya Ilmiah pada semester 7 (Program Studi Informatika UII, 2021, hlm. 31-32). Selain itu, mahasiswa yang memaksimalkan beban SKS pada setiap semester dapat memiliki kesempatan untuk menyelesaikan mata kuliah tahun terakhir lebih awal, sehingga berpeluang menyelesaikan studi pada semester 7 melalui jalur arus cepat. Kondisi ini menjadikan semester 6 sebagai titik prediksi yang relevan karena riwayat akademik semester 1–6 telah merepresentasikan capaian akademik utama mahasiswa sebelum memasuki tahap akhir studi. Dengan demikian, hasil prediksi masih dapat dimanfaatkan sebagai bahan pemantauan akademik sebelum mahasiswa memasuki tahap penyelesaian studi. Dalam penelitian ini, mahasiswa dikategorikan lulus tepat waktu apabila menyelesaikan studi dalam kurun waktu maksimal 8 semester, berlaku baik bagi mahasiswa jalur normal maupun jalur arus cepat.

2.3 Data Akademik

Dalam konteks *Educational Data Mining* (EDM), data akademik secara umum didefinisikan sebagai kumpulan informasi digital unik yang dihasilkan dari lingkungan pendidikan untuk memahami proses belajar siswa serta meningkatkan hasil pembelajaran. Berdasarkan taksonomi bidang ini, data pendidikan tersebar ke dalam berbagai kelompok, di mana jenis data yang digunakan dalam penelitian ini berfokus pada kategori data rekam akademik (*academic performance data*) yang mencakup capaian berkala seperti nilai IP, IPK, dan beban SKS dalam rentang semester tertentu, serta kategori data administrasi akademik (*administrative data*) yang mencakup catatan kumulatif selama mahasiswa aktif kuliah seperti jumlah kehadiran, izin, dan status cuti. Dalam konteks prediksi ketepatan waktu kelulusan, data akademik tersebut dapat digunakan sebagai dasar untuk mempelajari pola capaian mahasiswa dan mengidentifikasi mahasiswa yang berisiko lulus terlambat. Romero dan Ventura (2010, 2020) menyatakan bahwa seluruh sumber data akademik ini secara spesifik diperoleh dari jejak digital yang terekam pada sistem informasi akademik kampus, platform *Learning Management Systems* (LMS), sistem bimbingan cerdas (*Intelligent Tutoring Systems*), serta basis data administratif institusi pendidikan.

Pada praktiknya, data akademik sering kali memiliki nilai kosong (*missing values*). Nilai kosong dapat muncul karena berbagai faktor, seperti perbedaan lama studi mahasiswa, atribut yang tidak tercatat pada periode tertentu, atau struktur data yang berbeda antarangkatan. Penanganan nilai kosong dapat dilakukan melalui beberapa pendekatan, seperti menghapus data, melakukan imputasi, atau mempertahankan nilai kosong apabila karakteristik data dan algoritma yang digunakan memungkinkan. Oleh karena itu, strategi penanganan nilai kosong perlu disesuaikan dengan tujuan penelitian, karakteristik *dataset*, dan metode pemodelan yang digunakan.

2.3.1 *Data Collection*

Data collection merupakan tahap awal dalam penelitian berbasis *data mining*. Dalam konteks EDM, data umumnya bersumber dari sistem informasi akademik institusi yang mencatat aktivitas dan capaian akademik mahasiswa secara longitudinal (Romero & Ventura, 2010). Penggunaan data institusional semacam ini dikategorikan sebagai data sekunder, yaitu data yang dikumpulkan oleh suatu entitas untuk keperluan operasional dan kemudian dimanfaatkan kembali untuk tujuan penelitian (Saunders dkk., 2003).

Pada penelitian prediktif berbasis data akademik, pemilihan atribut perlu disesuaikan dengan tujuan prediksi dan titik waktu penggunaan model. Atribut yang digunakan sebaiknya hanya mencerminkan informasi yang tersedia sebelum prediksi dilakukan, agar model tidak memanfaatkan informasi masa depan yang belum tersedia pada kondisi nyata. Hal ini penting untuk menghindari *data leakage*, yaitu kondisi ketika fitur yang digunakan dalam pelatihan mengandung informasi yang secara tidak langsung telah merepresentasikan target prediksi.

2.3.2 *Labeling*

Labeling merupakan proses pemberian kelas atau kategori pada setiap *instance* dalam *dataset* sebagai target prediksi model (Alpaydin, 2014). Label menentukan keluaran yang akan dipelajari oleh model, sehingga kejelasan definisi label menjadi faktor penting dalam membangun model klasifikasi. Dalam penelitian prediktif di bidang pendidikan, label dapat berupa status kelulusan, risiko putus studi, capaian akademik, atau kategori performa mahasiswa.

Dalam beberapa penelitian berbasis data institusional, *dataset* yang diperoleh tidak selalu memuat label target secara eksplisit. Kondisi ini mengharuskan peneliti membentuk label berdasarkan indikator yang tersedia dan aturan akademik yang terdokumentasi. Pada konteks

prediksi ketepatan waktu kelulusan, label dapat diturunkan dari informasi akademik seperti jumlah SKS, IPK, masa studi, serta aktivitas akademik mahasiswa pada semester tertentu. Dengan demikian, proses pelabelan perlu dijelaskan secara eksplisit agar kriteria kelas target yang digunakan dalam pemodelan dapat ditelusuri, diterapkan secara konsisten, dan dapat dipertanggungjawabkan.

2.4 *Feature Engineering*

Feature engineering merupakan salah satu tahapan dalam siklus pengembangan model *machine learning*. Secara konseptual, *feature engineering* didefinisikan sebagai proses memilih, mentransformasikan, mengekstraksi, dan memanipulasi data mentah (*raw data*) menjadi variabel-variabel input baru yang representatif (Zheng & Casari, 2018). Variabel baru hasil rekayasa ini dirancang agar mampu menonjolkan struktur yang terdapat di dalam data.

Dalam penelitian ini, konsep *feature engineering* digunakan untuk mengolah data akademik mahasiswa berupa Indeks Prestasi (IP) dan Satuan Kredit Semester (SKS) dari semester satu hingga semester enam. Data tersebut memiliki karakteristik longitudinal karena merepresentasikan kondisi akademik mahasiswa secara berurutan dari waktu ke waktu. Berdasarkan karakteristik tersebut, *feature engineering* pada penelitian ini diarahkan untuk membentuk fitur yang dapat menangkap perubahan, variasi, dan ringkasan performa akademik mahasiswa selama masa studi.

Pentingnya *feature engineering* pada data pendidikan juga ditunjukkan oleh Hubbard dan Amponsah (2025), yang melakukan rekayasa fitur pada data *Learning Management System* (LMS) untuk meningkatkan prediksi performa mahasiswa. Penelitian tersebut menunjukkan bahwa data mentah, seperti *login* dan nilai pada LMS, dapat menghasilkan informasi prediktif yang lebih kuat setelah ditransformasikan menjadi fitur yang lebih representatif. Meskipun konteks data yang digunakan berbeda, yaitu data LMS, penelitian tersebut memperkuat dasar bahwa pemilihan dan pembentukan fitur yang sesuai dengan karakteristik data merupakan aspek penting dalam pemodelan prediktif di bidang pendidikan. Berdasarkan karakteristik data akademik yang digunakan, proses *feature engineering* dalam penelitian ini diarahkan pada dua konsep utama, yaitu pembentukan fitur perubahan antarperiode dan pembentukan fitur ringkasan statistik.

2.4.1 Pembentukan Fitur Perubahan Antarperiode

Pembentukan fitur perubahan antarperiode dilakukan untuk merepresentasikan perubahan nilai akademik mahasiswa dari satu semester ke semester berikutnya. Konsep ini dapat dikaitkan dengan prinsip *first-order differencing*, yaitu teknik yang menghitung perbedaan nilai suatu variabel pada periode saat ini terhadap periode sebelumnya (Baccar, 2019). Dalam konteks data akademik, konsep ini digunakan untuk merepresentasikan peningkatan atau penurunan performa mahasiswa dari waktu ke waktu.

2.4.2 Pembentukan Fitur Ringkasan Statistik

Pembentukan fitur ringkasan statistik dilakukan untuk merangkum karakteristik performa akademik mahasiswa dalam bentuk yang lebih padat. Ringkasan statistik seperti nilai minimum, maksimum, rentang, varians, dan rata-rata dapat digunakan untuk merepresentasikan pola umum performa mahasiswa selama beberapa semester. Dengan demikian, model tidak hanya mempelajari nilai mentah pada tiap semester, tetapi juga memperoleh informasi tambahan mengenai stabilitas, variasi, dan kecenderungan umum performa akademik mahasiswa yang mungkin tidak terlihat secara langsung dari fitur asli.

2.5 *Feature Selection*

Feature selection merupakan proses pemilihan himpunan bagian fitur yang paling relevan dari sekumpulan fitur yang tersedia untuk digunakan dalam proses pemodelan. Guyon dan Elisseeff (2003) menjelaskan bahwa *feature selection* bertujuan untuk meningkatkan performa prediksi, mengurangi kompleksitas komputasi, serta membantu pemahaman terhadap data yang digunakan. Dengan memilih fitur yang lebih relevan, model diharapkan dapat mempelajari pola data secara lebih efisien dan mengurangi pengaruh fitur yang tidak memberikan kontribusi signifikan terhadap target prediksi.

Salah satu pendekatan yang dapat digunakan dalam *feature selection* adalah *filter method*. Pendekatan ini menilai relevansi fitur berdasarkan ukuran statistik tertentu tanpa melibatkan proses pelatihan model secara langsung. Bommert dkk. (2019) menjelaskan bahwa *filter method* relatif cepat dihitung dan dapat dikombinasikan dengan berbagai jenis model karena proses seleksinya tidak bergantung pada algoritma prediktif tertentu. Oleh karena itu, pendekatan ini dapat digunakan sebagai tahap seleksi awal karena prosesnya relatif sederhana dan tidak terikat pada model tertentu.

Dalam penelitian ini, *feature selection* dilakukan menggunakan koefisien korelasi Pearson. Koefisien ini digunakan untuk mengukur kekuatan dan arah hubungan linear antara dua variabel numerik. Nilai korelasi berada pada rentang -1 sampai 1, dengan nilai mendekati 1 menunjukkan hubungan linear positif yang kuat, nilai mendekati -1 menunjukkan hubungan linear negatif yang kuat, dan nilai mendekati 0 menunjukkan hubungan linear yang lemah (Hastie dkk., 2009).

Penggunaan korelasi dalam *feature selection* dapat diarahkan pada dua aspek, yaitu relevansi fitur terhadap target dan redundansi antarfitur. Hall (1999) menjelaskan bahwa himpunan bagian fitur yang baik adalah himpunan fitur yang memiliki hubungan tinggi dengan kelas atau target, tetapi tidak memiliki korelasi tinggi satu sama lain. Berdasarkan konsep tersebut, fitur yang berkorelasi cukup kuat terhadap target dapat dipertahankan, sedangkan fitur yang memiliki korelasi sangat tinggi dengan fitur lain dapat dipertimbangkan sebagai fitur redundan.

Pendekatan *feature selection* berbasis matriks korelasi atau asosiasi juga telah digunakan dalam penelitian prediksi kelulusan mahasiswa. Amiruddin dan Ishak (2022) menerapkan *correlation matrix with heatmap* untuk menilai keterkaitan atribut input terhadap status kelulusan mahasiswa, sehingga dari 13 atribut awal diperoleh 9 atribut yang dianggap relevan untuk proses klasifikasi. Sementara itu, Swastikawati dan Utami (2025) menggunakan *Cramér's V correlation matrix* dalam tahap *preprocessing* untuk menganalisis hubungan antarfitur diskret dan menghapus atribut yang kurang relevan terhadap label status kelulusan. Meskipun ukuran korelasi atau asosiasi yang digunakan pada penelitian terdahulu tidak sepenuhnya sama dengan penelitian ini, keduanya menunjukkan bahwa analisis hubungan antarvariabel dapat digunakan untuk mendukung proses pemilihan fitur pada domain prediksi kelulusan mahasiswa.

Meskipun koefisien korelasi Pearson dapat mereduksi dimensi data dan mengatasi masalah redundansi antarfitur, metode ini memiliki keterbatasan utama, yaitu hanya mampu mengukur hubungan yang bersifat linear (Hall, 1999). Jika terdapat hubungan nonlinear yang kompleks antarfitur atau antara fitur dengan label (misalnya hubungan berbentuk kurva atau eksponensial), koefisien Pearson dapat menghasilkan nilai mendekati nol ($r \approx 0$). Namun, dalam konteks penelitian ini, keterbatasan tersebut diantisipasi melalui tiga strategi utama yang saling berintegrasi.

Pertama, penerapan *Pearson Correlation Matrix* diatur melalui pendekatan dua filter adaptif, yaitu korelasi fitur dengan label dan korelasi antarfitur. Pada filter korelasi fitur dengan

label, sistem menyediakan mekanisme penyesuaian batas acuan (*threshold*) secara fleksibel hingga nilai korelasi yang rendah. Fleksibilitas ini bertujuan agar fitur-fitur numerik penting yang berpotensi memiliki hubungan nonlinear kuat terhadap label tetap dapat dipertahankan. Sementara itu, filter korelasi antarfitur diterapkan dengan batas acuan yang disesuaikan secara dinamis guna mengeliminasi redundansi informasi. Kedua, fitur-fitur nonlinear yang berhasil dipertahankan oleh filter tersebut selanjutnya akan diproses menggunakan algoritma pemodelan berbasis *tree ensemble* (*Histogram-based Gradient Boosting*, *Extreme Gradient Boosting*, dan *Light Gradient Boosting Machine*) yang memiliki kemampuan inheren untuk menangkap dan mempelajari pola nonlinear secara mandiri (Chen & Guestrin, 2016; Ke dkk., 2017; Pedregosa dkk., 2011). Ketiga, rentang batas acuan optimal untuk kedua filter tersebut dieksplorasi secara otomatis menggunakan teknik pencarian acak (*Random Search*) berbasis validasi silang (*cross-validation*). Melalui kombinasi strategi ini, fitur yang terpilih diharapkan dapat meminimalkan efek redundansi sekaligus meningkatkan keselarasan karakteristik informasi dengan kebutuhan model prediktif.

2.6 *Gradient Boosting*

Gradient Boosting merupakan salah satu metode *ensemble learning* yang membangun model prediktif secara bertahap melalui kombinasi beberapa model lemah, umumnya berupa pohon keputusan (*decision tree*) berukuran kecil. Berbeda dengan pendekatan *ensemble* seperti *Bootstrap Aggregating* yang membangun beberapa model secara paralel, *gradient boosting* membangun model secara sekuens (Ayulani dkk., 2023). Pada setiap iterasi, model baru dilatih untuk memperbaiki kesalahan prediksi dari model sebelumnya, sehingga hasil akhir diperoleh dari akumulasi kontribusi beberapa model yang dibangun secara bertahap (Hastie dkk., 2009).

Secara umum, prinsip kerja *gradient boosting* didasarkan pada proses optimasi fungsi *loss*. Model pada iterasi berikutnya diarahkan untuk meminimalkan kesalahan model sebelumnya dengan mengikuti arah gradien negatif dari fungsi *loss*. Dengan mekanisme tersebut, *gradient boosting* mampu membentuk model yang adaptif terhadap pola data dan sering digunakan pada permasalahan klasifikasi maupun regresi berbasis data tabular. Pada kasus klasifikasi, proses ini membuat setiap pohon tambahan berkontribusi dalam memperbaiki prediksi kelas yang masih keliru pada tahap sebelumnya.

Dalam penelitian ini, keluarga algoritma *gradient boosting* dipilih karena memiliki karakteristik yang sesuai dengan kondisi data akademik mahasiswa. Data akademik yang digunakan dalam penelitian ini mengandung nilai kosong (*missing value*) pada beberapa

atribut. Dalam konteks data akademik mahasiswa, *missing value* tidak selalu dapat dipandang sebagai sekadar ketiadaan data, karena pada kondisi tertentu *missing value* dapat merepresentasikan informasi akademik tertentu. Sebagai contoh, nilai IP, IPK, atau SKS yang kosong pada semester tertentu dapat mengindikasikan bahwa mahasiswa tidak memiliki aktivitas akademik pada semester tersebut, misalnya karena cuti atau sebab akademik lainnya. Kondisi tersebut berpotensi berkaitan dengan keterlambatan masa studi mahasiswa.

Berdasarkan kondisi data tersebut, diperlukan model yang dapat menangani keberadaan *missing value* tanpa harus bergantung sepenuhnya pada proses imputasi manual. Proses tersebut berisiko mengubah makna nilai kosong menjadi nilai buatan yang belum tentu merepresentasikan kondisi akademik sebenarnya. Oleh karena itu, penelitian ini menggunakan beberapa implementasi *gradient boosting* berbasis pohon keputusan, seperti *Histogram-based Gradient Boosting*, *Extreme Gradient Boosting*, dan *Light Gradient Boosting Machine*, ketiganya memiliki mekanisme internal untuk menangani *missing value* selama proses pembentukan pohon.

Selain itu, hubungan antara fitur akademik dan status ketepatan waktu kelulusan tidak selalu bersifat linear. Pada tahap *feature selection*, korelasi Pearson digunakan sebagai pendekatan awal untuk menilai relevansi fitur terhadap target dan mengurangi redundansi antarfitur berdasarkan hubungan linear. Namun, penggunaan Pearson tidak dimaksudkan untuk membatasi proses pemodelan hanya pada pola hubungan linear. Fitur akademik seperti perubahan indeks prestasi, variasi capaian akademik, serta jumlah SKS yang ditempuh hingga semester enam dapat membentuk pola yang kompleks terhadap peluang mahasiswa untuk lulus tepat waktu atau terlambat. Model berbasis pohon keputusan dalam keluarga *gradient boosting* memiliki kemampuan untuk menangkap pola tersebut tanpa memerlukan asumsi hubungan linear antarvariabel. Oleh karena itu, algoritma tersebut digunakan karena model berbasis pohon mampu menangkap interaksi antarfitur secara lebih fleksibel setelah proses seleksi fitur dilakukan.

2.6.1 *Extreme Gradient Boosting*

Extreme Gradient Boosting (XGB) merupakan pengembangan dari *gradient boosting* yang dirancang sebagai sistem peningkatan pohon (*tree boosting*) yang terukur. Chen dan Guestrin (2016) menjelaskan bahwa XGB tidak hanya berfokus pada pembentukan model prediktif, tetapi juga pada optimasi sistem agar proses pelatihan dapat dilakukan secara efisien pada data berukuran besar. Beberapa komponen utama yang membedakan XGB dari *Gradient*

Boosting konvensional adalah mekanisme sadar kerenggangan data (*sparsity-aware*), penggunaan regularisasi pada fungsi objektif, dan sketsa kuantil tertimbang (*weighted quantile sketch*).

Salah satu karakteristik penting XGB adalah mekanisme *sparsity-aware* untuk menangani data *sparse*, termasuk nilai yang hilang (*missing values*). Mekanisme ini bekerja saat proses pembentukan pohon keputusan dengan mendeteksi baris data (*instance*) yang memiliki *missing value* pada suatu fitur. Seluruh *instance* tersebut akan dikelompokkan, lalu disimulasikan untuk dikirim ke cabang kiri dan cabang kanan. Algoritma XGB kemudian menghitung nilai *gain* dari kedua simulasi tersebut, yaitu metrik yang mengukur seberapa besar penurunan tingkat kesalahan (*loss*) model. Cabang yang menghasilkan nilai *gain* tertinggi akan ditetapkan sebagai jalur *default* bagi data yang kosong pada percabangan tersebut. Dampaknya, *missing value* menjadi berguna dalam proses pelatihan.

Selain itu, XGB juga menggunakan regularisasi dalam fungsi objektifnya. Regularisasi digunakan untuk memberikan penalti terhadap kompleksitas model, sehingga model tidak hanya diarahkan untuk meminimalkan kesalahan prediksi, tetapi juga dikendalikan agar tidak terlalu kompleks. Mekanisme ini membantu mengurangi risiko *overfitting*, terutama ketika algoritma memiliki banyak pohon atau struktur pohon yang dalam.

Untuk meningkatkan efisiensi pada data berukuran besar, XGB menggunakan *weighted quantile sketch* untuk mencari titik pembelahan (*split*). Teknik ini digunakan untuk menentukan kandidat *split* berdasarkan distribusi berbobot dari data, sehingga proses pencarian *split* dapat dilakukan secara lebih efisien tanpa harus mengevaluasi seluruh nilai fitur secara eksak. Dengan kombinasi mekanisme *sparsity-aware*, regularisasi, dan teknik *weighted quantile sketch*, XGB menjadi salah satu implementasi *gradient boosting* yang kuat untuk pemodelan data tabular. Di samping aspek komputasinya yang efisien, implementasi XGB di bidang pendidikan juga telah terbukti memberikan akurasi yang sangat kompetitif dalam memodelkan status penyelesaian studi mahasiswa tingkat sarjana (Nirmala dkk., 2022).

2.6.2 *Histogram-based Gradient Boosting*

Histogram-based Gradient Boosting (HGB) merupakan implementasi *gradient boosting* berbasis pohon keputusan yang menggunakan pendekatan histogram dalam proses pembelahan simpul induk (*node splitting*). Pada algoritma *gradient boosting* konvensional, proses pencarian titik pembelahan (*split*) pada fitur numerik kontinu dapat menjadi mahal secara komputasi karena algoritma perlu mengevaluasi banyak kandidat nilai *split*. HGB mengatasi

permasalahan tersebut dengan mengubah nilai fitur kontinu ke dalam sejumlah interval diskret (*bins*) sebelum proses pelatihan pohon dilakukan (Pedregosa dkk., 2011).

Pada implementasi HGB di scikit-learn, setiap fitur pada data masukan terlebih dahulu dikelompokkan ke dalam *bin* bernilai integer sebelum tahap pelatihan. Parameter max_bins digunakan untuk menentukan jumlah maksimum *bin* pada fitur yang memiliki nilai, dengan batas maksimum 255 *bin*. Sementara itu, nilai kosong (*missing value*) ditempatkan pada *bin* khusus sehingga tetap dapat diproses dalam pembentukan pohon. Selama proses pelatihan, pada setiap kandidat *split*, algoritma HGB mempelajari apakah sampel dengan *missing value* sebaiknya diarahkan ke cabang kiri atau cabang kanan berdasarkan *potential gain* yang dihasilkan. Representasi berbasis *bin* ini membuat proses pembentukan model pohon keputusan menjadi lebih ringkas karena pencarian *split* tidak lagi dilakukan terhadap seluruh nilai kontinu asli, melainkan terhadap sejumlah *bin* yang jauh lebih sedikit.

Selain pendekatan histogram, HGB juga menyediakan parameter regularisasi, yaitu l2_regularization, yang dapat digunakan untuk memberikan penalti terhadap kompleksitas model. Regularisasi ini berperan dalam membantu mengendalikan risiko keselarasan yang berlebihan (*overfitting*) selama proses pelatihan. Oleh karena itu, HGB relevan digunakan sebagai salah satu algoritma pembanding dalam penelitian ini karena mampu merepresentasikan pendekatan *gradient boosting* berbasis histogram yang efisien untuk data tabular numerik.

2.6.3 *Light Gradient Boosting Machine*

Light Gradient Boosting Machine (LGBM) merupakan kerangka kerja *gradient boosting* berbasis pohon keputusan yang dikembangkan untuk meningkatkan efisiensi pelatihan dan penggunaan memori pada data berukuran besar. Ke dkk. (2017) menjelaskan bahwa salah satu tantangan utama pada *Gradient Boosting Decision Tree* adalah tingginya biaya komputasi dalam proses pencarian titik pembelahan (*split*). Untuk mengatasi masalah tersebut, LGBM mengembangkan dua teknik utama, yaitu *Gradient-based One-Side Sampling* dan *Exclusive Feature Bundling*.

Gradient-based One-Side Sampling (GOSS) merupakan teknik untuk mengurangi jumlah baris dalam kumpulan data. Teknik ini bekerja dengan cara mempertahankan baris data (*instance*) dengan nilai gradien besar dan mengambil sebagian *instance* dengan gradien kecil secara acak. *Instance* dengan gradien besar dianggap lebih berpengaruh terhadap perhitungan perolehan informasi karena merepresentasikan data yang masih sulit diprediksi oleh algoritma.

Dengan cara ini, LGBM dapat mengurangi jumlah data yang digunakan dalam perhitungan *split* tanpa kehilangan terlalu banyak informasi penting bagi proses pembelajaran.

Exclusive Feature Bundling (EFB) adalah sebuah teknik yang bekerja dengan cara menggabungkan beberapa kolom yang jarang berisi data secara bersamaan menjadi satu kolom baru. Teknik ini bertujuan untuk mengurangi jumlah fitur yang perlu diproses dalam pembentukan histogram, sehingga beban komputasi dan penggunaan memori dapat dikurangi. EFB sangat relevan pada data berdimensi tinggi atau data *sparse*, karena banyak fitur dapat digabungkan tanpa menyebabkan konflik nilai yang besar.

Selain GOSS dan EFB, algoritma LGBM menggunakan strategi pertumbuhan pohon berbasis daun (*leaf-wise*). Berbeda dengan pendekatan berbasis kedalaman (*level-wise*) yang menumbuhkan pohon secara merata pada setiap level, strategi *leaf-wise* memilih daun dengan potensi penurunan *loss* terbesar untuk dikembangkan terlebih dahulu. Namun, strategi ini juga memerlukan pengendalian parameter seperti jumlah daun maksimum agar struktur pohon tidak terlalu kompleks dan berisiko mengalami *overfitting*. Dengan karakteristik tersebut, algoritma LGBM digunakan dalam penelitian ini sebagai salah satu algoritma pembanding dalam keluarga *gradient boosting*. Penggunaan GOSS, EFB, dan strategi *leaf-wise* menjadikan LGBM relevan untuk mengevaluasi pendekatan *gradient boosting* yang menekankan efisiensi pelatihan dan kemampuan pemodelan pola nonlinear pada data tabular.

2.7 Pemilihan *Hyperparameter* Menggunakan *Random Search*

Hyperparameter merupakan parameter yang nilainya ditentukan sebelum proses pelatihan model dilakukan dan tidak dipelajari secara langsung dari data (Probst, Boulesteix, & Bischl, 2019). Berbeda dengan parameter model yang diperoleh selama proses pelatihan, *hyperparameter* berfungsi mengatur perilaku algoritma pembelajaran, seperti kompleksitas model, laju pembelajaran, jumlah iterasi, serta mekanisme regularisasi. Oleh karena itu, pemilihan *hyperparameter* yang tepat menjadi salah satu tahapan penting dalam pengembangan model pembelajaran mesin karena dapat memengaruhi kemampuan model dalam mempelajari pola data dan melakukan generalisasi terhadap data baru.

Dalam penelitian ini, pemilihan *hyperparameter* diperlukan untuk mengontrol perilaku model *gradient boosting*, karena dapat memengaruhi performa prediksi. Nilai *hyperparameter* yang terlalu sederhana dapat menyebabkan model mengalami *underfitting*, sedangkan konfigurasi yang terlalu kompleks dapat meningkatkan risiko *overfitting*. Mengingat penelitian ini bertujuan membangun model prediksi ketepatan waktu kelulusan dengan performa yang

seimbang pada kedua kelas, proses pemilihan *hyperparameter* tidak hanya diarahkan untuk meningkatkan performa prediksi, tetapi juga untuk memperoleh konfigurasi model yang sesuai dengan karakteristik data akademik yang digunakan.

2.7.1 Mekanisme *Random Search*

Random Search merupakan metode optimasi *hyperparameter* yang bekerja dengan memilih kombinasi nilai *hyperparameter* secara acak dari ruang pencarian yang telah ditentukan. Berbeda dengan *Grid Search* yang mengevaluasi seluruh kombinasi *hyperparameter*, *Random Search* hanya mengevaluasi sejumlah kombinasi tertentu sesuai jumlah iterasi yang ditetapkan. Dengan demikian, *Random Search* dapat menjadi alternatif yang lebih efisien ketika ruang pencarian *hyperparameter* berukuran besar.

Bergstra dkk. (2012) menunjukkan bahwa *Random Search* dapat lebih efisien dibandingkan *Grid Search* karena tidak semua *hyperparameter* memiliki pengaruh yang sama terhadap performa model. Pada *Grid Search*, banyak sumber daya komputasi dapat terpakai untuk mengevaluasi kombinasi dari *hyperparameter* yang kurang berpengaruh. Sebaliknya, *Random Search* memberikan peluang untuk mengeksplorasi lebih banyak variasi nilai pada *hyperparameter* yang berpengaruh dalam jumlah evaluasi yang lebih terbatas.

Dalam penelitian ini, *Random Search* diterapkan melalui *RandomizedSearchCV* dengan validasi silang *Stratified K-Fold*. Setiap kombinasi *hyperparameter* dievaluasi berdasarkan nilai *macro F1-score*, karena metrik tersebut memperhitungkan performa model pada kedua kelas secara seimbang. Penggunaan validasi silang bertujuan agar pemilihan *hyperparameter* tidak hanya bergantung pada satu pembagian data latih, melainkan mempertimbangkan performa rata-rata model pada beberapa bagian data (*fold*). Validasi silang umum digunakan dalam proses estimasi performa dan pemilihan model karena dapat memberikan gambaran performa yang lebih stabil dibandingkan evaluasi pada satu pembagian data saja (Kohavi, 1995).

2.7.2 Parameter Kunci dalam *Pipeline Gradient Boosting*

Ruang pencarian *hyperparameter* dalam penelitian ini mencakup konfigurasi pada tahapan *feature engineering*, *feature selection*, dan model *gradient boosting*. Pada tahap *feature engineering*, *hyperparameter* yang digunakan adalah keputusan untuk mempertahankan atau menghapus fitur awal setelah fitur turunan dibentuk. Pengaturan ini digunakan untuk membandingkan apakah model memperoleh performa lebih baik ketika hanya

menggunakan fitur hasil rekayasa atau tetap mempertahankan kombinasi antara fitur awal dan fitur turunan.

Pada tahap *feature selection*, *hyperparameter* yang dipilih berkaitan dengan ambang korelasi fitur terhadap label dan ambang korelasi antarfitur. Ambang korelasi fitur terhadap label digunakan untuk memilih fitur yang memiliki hubungan kuat dengan target prediksi, sedangkan ambang korelasi antarfitur digunakan untuk mengurangi redundansi antarfitur. Dengan memasukkan ambang *feature selection* ke dalam proses *Random Search*, penelitian ini memberikan kesempatan bagi model untuk memilih konfigurasi seleksi fitur yang paling sesuai berdasarkan hasil validasi silang.

Pada model *gradient boosting*, *hyperparameter* utama yang dipilih berkaitan dengan laju pembelajaran (*learning rate*), jumlah estimator (*n_estimator/max_iter*), kompleksitas pohon, sampling, dan regularisasi. Parameter *learning rate* mengatur besarnya kontribusi setiap pelajar baru terhadap model akhir, sedangkan *n_estimator* menentukan banyaknya pembentukan pohon yang dilakukan. Keduanya saling berkaitan karena *learning rate* yang lebih kecil umumnya membutuhkan jumlah estimator yang lebih banyak agar model dapat mempelajari pola data secara memadai.

Kompleksitas pohon dikendalikan melalui *hyperparameter* seperti kedalaman maksimum pohon (*max_depth*) dan batas minimum jumlah sampel pada node daun (*min_samples_leaf*). Pengaturan ini penting karena pohon yang terlalu kompleks dapat meningkatkan risiko *overfitting*, sedangkan pohon yang terlalu sederhana dapat menyebabkan model kurang mampu menangkap pola pada data (*underfitting*). Selain itu, *hyperparameter* sampling seperti *subsample* dan *colsample_bytree* digunakan untuk mengatur proporsi baris dan fitur yang digunakan dalam pembentukan pohon, sehingga dapat membantu meningkatkan kemampuan generalisasi model.

Regularisasi juga digunakan untuk membatasi kompleksitas model. Pengaturan regularisasi membantu mengurangi kecenderungan model untuk terlalu menyesuaikan diri terhadap data latih. Meskipun setiap model *gradient boosting* memiliki nama dan karakteristik *hyperparameter* yang berbeda, prinsip umum pengaturannya tetap berkaitan dengan keseimbangan antara kemampuan model dalam mempelajari pola data dan kemampuannya dalam melakukan generalisasi.

2.7.3 Strategi Pemilihan *Hyperparameter* dalam *Pipeline*

Strategi pemilihan *hyperparameter* dalam penelitian ini dilakukan dengan terlebih dahulu menentukan ruang pencarian untuk setiap tahapan *pipeline*. Ruang pencarian tersebut mencakup konfigurasi *feature engineering*, ambang *feature selection*, serta *hyperparameter* dari model *gradient boosting*. Penentuan ruang pencarian dilakukan dengan mempertimbangkan jumlah data yang dimiliki dan fungsi masing-masing *hyperparameter*, terutama yang berkaitan dengan laju pembelajaran, jumlah proses *boosting*, kompleksitas pohon, sampling, dan regularisasi.

Setelah ruang pencarian ditentukan, proses pemilihan konfigurasi dilakukan menggunakan *Random Search*. Metode ini mengambil sejumlah kombinasi secara acak untuk dievaluasi. Oleh karena itu, konfigurasi yang diperoleh tidak diklaim sebagai konfigurasi paling optimal secara global, melainkan sebagai konfigurasi terbaik berdasarkan kombinasi yang dievaluasi dalam ruang pencarian penelitian ini.

Setiap kombinasi *hyperparameter* dievaluasi menggunakan validasi silang *Stratified K-Fold* pada data latih dengan *macro F1-score* sebagai metrik utama. Penggunaan *macro F1-score* dilakukan karena penelitian ini menekankan keseimbangan performa model dalam mengenali mahasiswa yang lulus tepat waktu dan mahasiswa yang lulus terlambat. Setelah kombinasi terbaik diperoleh, model dievaluasi pada data uji terpisah untuk melihat performa akhir terhadap data yang tidak digunakan dalam proses optimasi.

Selain itu, penelitian ini juga meninjau selisih antara skor data latih dan skor validasi silang. Selisih tersebut digunakan sebagai indikator tambahan untuk melihat kecenderungan *overfitting*. Dengan demikian, proses optimasi tidak hanya diarahkan untuk memperoleh nilai validasi silang yang tinggi, tetapi juga untuk menilai kemampuan generalisasi model.

2.8 Teknik Evaluasi

Evaluasi kinerja model merupakan tahapan penting dalam penelitian klasifikasi karena digunakan untuk menilai kemampuan model dalam menghasilkan prediksi yang sesuai dengan label aktual. Pada penelitian ini, evaluasi dilakukan untuk mengukur performa model *gradient boosting* dalam memprediksi ketepatan waktu kelulusan mahasiswa. Permasalahan yang digunakan merupakan klasifikasi biner, dengan kelas positif merepresentasikan mahasiswa yang lulus tepat waktu dan kelas negatif merepresentasikan mahasiswa yang lulus terlambat.

Evaluasi model dilakukan menggunakan beberapa pendekatan, yaitu *confusion matrix*, metrik evaluasi klasifikasi, evaluasi kinerja *train* dan *cross-validation*, optimasi *threshold*,

AUC-ROC, serta *permutation feature importance*. *Confusion matrix* digunakan untuk melihat distribusi prediksi benar dan salah pada masing-masing kelas. Metrik evaluasi klasifikasi digunakan untuk mengukur performa model secara kuantitatif. Evaluasi kinerja *train* dan *cross-validation* digunakan untuk meninjau indikasi *overfitting* dan *underfitting* melalui perbandingan skor data latih dan validasi silang. Optimasi *threshold* digunakan untuk menentukan batas probabilitas klasifikasi yang lebih sesuai dengan tujuan penelitian. AUC-ROC digunakan sebagai metrik tambahan untuk menilai kemampuan model dalam membedakan kedua kelas, sedangkan *permutation feature importance* digunakan sebagai analisis tambahan untuk memahami kontribusi fitur terhadap prediksi model.

2.8.1 *Confusion Matrix*

Confusion matrix merupakan alat evaluasi yang digunakan untuk membandingkan hasil prediksi model dengan label aktual pada permasalahan klasifikasi (Sokolova & Lapalme, 2009). Melalui *confusion matrix*, performa model dapat dianalisis berdasarkan jumlah prediksi yang benar dan salah pada setiap kelas. Matriks ini menjadi dasar perhitungan berbagai metrik evaluasi seperti *accuracy*, *precision*, *recall*, dan *F1-score*. Dalam klasifikasi biner, *confusion matrix* terdiri dari empat komponen utama, yaitu *true positive*, *true negative*, *false positive*, dan *false negative* seperti yang ditunjukkan pada Tabel 2.1.

Tabel 2.1 Tabel *Confusion Matrix*

Data class	**Real P**	**Real N**
Classified as P	TP	FP
Classified as N	FN	TN

Sumber: (Sokolova & Lapalme, 2009)

Pada penelitian ini, kelas positif didefinisikan sebagai mahasiswa yang lulus tepat waktu, sedangkan kelas negatif didefinisikan sebagai mahasiswa yang lulus terlambat. Berdasarkan definisi tersebut, komponen *confusion matrix* dalam penelitian ini dapat dijelaskan sebagai berikut:

a. *True Positive* (TP): merupakan kondisi ketika mahasiswa yang sebenarnya lulus tepat waktu diprediksi benar sebagai lulus tepat waktu.
b. *False Positive* (FP): merupakan kondisi ketika mahasiswa yang sebenarnya lulus terlambat diprediksi salah sebagai lulus tepat waktu.

c. *True Negative* (TN): merupakan kondisi ketika mahasiswa yang sebenarnya lulus terlambat diprediksi benar sebagai lulus terlambat.
d. *False Negative* (FN): merupakan kondisi ketika mahasiswa yang sebenarnya lulus tepat waktu diprediksi salah sebagai lulus terlambat.

Dalam konteks penelitian ini, *false positive* perlu diperhatikan karena model memprediksi mahasiswa sebagai lulus tepat waktu, padahal mahasiswa tersebut sebenarnya lulus terlambat. Kesalahan ini dapat menyebabkan mahasiswa yang berisiko terlambat tidak teridentifikasi oleh model. Sementara itu, *false negative* menunjukkan mahasiswa yang sebenarnya lulus tepat waktu tetapi diprediksi sebagai lulus terlambat. Kedua jenis kesalahan tersebut tetap perlu diperhatikan karena penelitian ini bertujuan menghasilkan model yang mampu mengenali kedua kelas secara seimbang.

2.8.2 Metrik Evaluasi

Berdasarkan nilai-nilai pada *confusion matrix*, performa model klasifikasi dapat diukur menggunakan beberapa metrik evaluasi sebagai berikut:

a. Akurasi (*Accuracy*)

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{2.1}$$

Berdasarkan Persamaan (2.1), *Accuracy* mengukur proporsi prediksi yang benar terhadap seluruh data yang dievaluasi. Metrik ini memberikan gambaran umum mengenai tingkat ketepatan model, tetapi dapat kurang representatif apabila performa pada masing-masing kelas tidak seimbang.

b. Presisi (*Precision*)

$$Precision = \frac{TP}{TP + FP} \tag{2.2}$$

Berdasarkan Persamaan (2.2), *Precision* mengukur proporsi prediksi kelas positif yang benar dari seluruh data yang diprediksi sebagai kelas positif. Dalam penelitian ini, *precision* menunjukkan seberapa andal model ketika memprediksi mahasiswa sebagai lulus tepat waktu.

c. *Recall*

$$Recall = \frac{TP}{TP + FN} \tag{2.3}$$

Berdasarkan Persamaan (2.3), *Recall* mengukur proporsi data kelas positif yang berhasil dikenali dengan benar oleh model. Dalam penelitian ini, *recall* menunjukkan kemampuan model dalam menemukan mahasiswa yang benar-benar lulus tepat waktu.

d. *F1-score*

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{2.4}$$

Berdasarkan Persamaan (2.4), *F1-score* merupakan rata-rata harmonik dari *precision* dan *recall*. Metrik ini digunakan untuk memberikan ukuran performa yang lebih seimbang ketika *precision* dan *recall* sama-sama perlu diperhatikan.

e. *Macro F1-score*

$$Macro\ F1 = \frac{1}{2}(F1_0 + F1_1) \tag{2.5}$$

Berdasarkan Persamaan (2.5), *macro F1-score* merupakan rata-rata nilai *F1-score* dari setiap kelas tanpa mempertimbangkan jumlah data pada masing-masing kelas. Dengan demikian, setiap kelas memperoleh bobot yang sama dalam perhitungan metrik. Dalam penelitian ini, *macro F1-score* digunakan sebagai metrik utama karena model diharapkan tidak hanya kuat dalam mengenali salah satu kelas, tetapi mampu memprediksi mahasiswa yang lulus tepat waktu dan mahasiswa yang lulus terlambat secara seimbang.

Penggunaan *macro F1-score* juga sesuai dengan tujuan penelitian yang menekankan keseimbangan performa pada kedua kelas. Oleh karena itu, *macro F1-score* digunakan dalam proses pemilihan *hyperparameter*, evaluasi awal dengan *threshold* bawaan, optimasi *threshold*, evaluasi akhir model, dan perhitungan *permutation feature importance*.

2.8.3 Evaluasi Kinerja dengan *Cross-validation*

Evaluasi model pada tahap *training* menggunakan validasi silang (*cross-validation*) dilakukan untuk menilai kemampuan model dalam mempertahankan performa pada data

validasi yang tidak digunakan dalam proses pelatihan (*fitting*) pada setiap *fold*. Model yang baik tidak hanya memiliki performa tinggi pada data latih, tetapi juga mampu memberikan performa yang relatif konsisten pada data validasi. Dalam pembelajaran mesin, kemampuan model untuk mempertahankan kinerja pada data yang tidak digunakan selama proses *fitting* menjadi penting karena model yang terlalu menyesuaikan diri terhadap data latih berpotensi mengalami penurunan performa ketika dihadapkan pada data lain. Penggunaan *cross-validation* dalam evaluasi model bertujuan untuk memperoleh estimasi performa yang lebih stabil dibandingkan evaluasi pada satu pembagian data saja (Kohavi, 1995).

$$Gap_{train-cv} = |S_{train} - S_{cv}| \tag{2.6}$$

Berdasarkan Persamaan (2.6), selisih antara skor data latih dan skor *cross-validation* digunakan sebagai indikator tambahan untuk mengamati konsistensi kinerja model dan kemungkinan terjadinya *overfitting*. Apabila skor data latih jauh lebih tinggi dibandingkan skor *cross-validation*, model dapat menunjukkan indikasi *overfitting* karena performanya pada data latih tidak dapat dipertahankan pada data validasi. Sementara itu, apabila skor data latih dan skor *cross-validation* sama-sama rendah, model dapat menunjukkan indikasi *underfitting* karena belum mampu mempelajari pola data secara memadai.

Dalam penelitian ini, perbandingan kinerja *training* dan *cross-validation* tidak digunakan sebagai dasar tunggal dalam pemilihan model. Konfigurasi terbaik setiap model tetap ditentukan berdasarkan nilai *macro F1-score* pada proses *cross-validation*. Sementara itu, perbandingan antara *mean train score*, *mean cross-validation score*, dan selisih keduanya digunakan sebagai analisis tambahan untuk mengamati konsistensi kinerja model serta indikasi *overfitting* setelah proses pemilihan *hyperparameter*.

2.8.4 Optimasi *Threshold*

Ambang batas (*threshold*) adalah nilai acuan yang digunakan model *machine learning* untuk mengubah nilai probabilitas menjadi sebuah keputusan klasifikasi. Secara umum, model klasifikasi biner menggunakan *threshold* bawaan sebesar 0,50. Artinya, jika suatu data memiliki probabilitas prediksi kelas positif mencapai 0,50 atau lebih, data tersebut otomatis dikelompokkan ke dalam kelas positif. Sebaliknya, jika di bawah nilai tersebut, data akan dimasukkan ke dalam kelas negatif.

Namun, *threshold* bawaan 0,50 belum tentu menghasilkan performa terbaik untuk setiap permasalahan klasifikasi. Pemilihan *threshold* dapat memengaruhi distribusi prediksi, termasuk jumlah *false positive* dan *false negative*. Oleh karena itu, *threshold* dapat disesuaikan berdasarkan tujuan evaluasi tertentu, misalnya untuk meningkatkan *F1-score*, *recall*, *precision*, atau metrik lain yang sesuai dengan kebutuhan penelitian. Dalam konteks *machine learning* modern, penyesuaian *decision threshold* juga dikenal sebagai salah satu pendekatan untuk mengoptimalkan keputusan klasifikasi berdasarkan metrik tertentu (Lipton dkk., 2014).

Proses optimasi diperlukan untuk menghasilkan nilai *threshold* yang paling sesuai dengan karakteristik distribusi data. Agar model terhindar dari bias akibat kebocoran data (*data leakage*), tahap pencarian nilai *threshold* optimal ini wajib diintegrasikan di luar data pengujian (*held-out test set*). Berdasarkan teori *K-fold cross-validation* yang dikemukakan oleh Hastie dkk. (2009), data dibagi menjadi K bagian, di mana model dilatih pada K-1 bagian dan dievaluasi pada satu bagian yang tersisa secara bergantian. Kumpulan hasil prediksi pada bagian data yang disisihkan inilah yang secara praktis disebut sebagai prediksi *out-of-fold* (OOF). Secara matematis, nilai probabilitas OOF untuk setiap sampel *i* didefinisikan pada Persamaan (2.7):

$$\hat{p}_i = Pr(\hat{y}_i = 1 | x_i; \theta_{-k(i)}) \tag{2.7}$$

di mana x_i menyatakan vektor fitur dari sampel ke-i, dan $\theta_{-k(i)}$ merepresentasikan parameter model yang dilatih tanpa melibatkan *fold k* yang memuat sampel i. Formulasi ini memastikan bahwa probabilitas prediksi dihasilkan secara independen dari proses pelatihan model. Selanjutnya, untuk mengonversi nilai probabilitas kontinu $\hat{p}_i$ menjadi label keputusan biner $\hat{y}_i(t)$ di bawah pengaruh nilai *threshold t*, digunakan fungsi indikator sebagaimana dirumuskan pada Persamaan (2.8):

$$\hat{y}_i(t) = 1[\hat{p}_i \geq t] \tag{2.8}$$

Nilai fungsi indikator 1[ ] akan mengevaluasi pernyataan menjadi label 1 (positif) jika kondisi terpenuhi, dan label 0 (negatif) jika sebaliknya.

Dalam penelitiannya, Lipton dkk. (2014) merumuskan aturan keputusan berbasis teori optimasi untuk memaksimalkan metrik evaluasi yang selaras dengan proporsi kelas, salah

satunya adalah nilai *macro-averaged F1-score*. Pencarian nilai ambang batas terbaik dilakukan dengan menguji himpunan kandidat nilai *t* dalam rentang interval tertentu, yang secara matematis dinyatakan pada Persamaan (2.9):

$$t^* = arg\max_{t \in T} F1_{macro}\left(y, \hat{y}(t)\right) \tag{2.9}$$

di mana t^* merupakan nilai ambang batas keputusan yang optimal, *T* adalah himpunan kandidat *threshold*, sedangkan *y* dan $\hat{y}_i(t)$ berturut-turut merupakan label riil (*true label*) serta label hasil prediksi model pada *threshold t*. Sedangkan operator *arg max* menunjukkan nilai ambang batas yang menghasilkan skor rata-rata F1 makro tertinggi dalam ruang pencarian. Melalui kombinasi isolasi data berbasis OOF dan aturan maksimisasi, nilai t^* yang terpilih diharapkan dapat meminimalkan risiko *overfitting*, sehingga mampu memberikan gambaran yang lebih realistis saat diaplikasikan pada data uji.

Setelah *threshold* terbaik diperoleh dari data latih, model dievaluasi kembali pada data uji terpisah menggunakan *threshold* tersebut. Evaluasi akhir ini digunakan untuk membandingkan performa model setelah penyesuaian *threshold* terhadap performa awal pada *threshold* bawaan 0,50. Dengan demikian, optimasi *threshold* dalam penelitian ini digunakan untuk menyesuaikan batas keputusan model agar lebih sesuai dengan tujuan evaluasi.

2.8.5 AUC-ROC

Receiver Operating Characteristic atau ROC merupakan kurva evaluasi yang menggambarkan hubungan antara *true positive rate* (TPR) dan *false positive rate* (FPR) pada berbagai nilai *threshold* (Fawcett, 2006). ROC digunakan untuk melihat kemampuan model dalam membedakan kelas positif dan kelas negatif berdasarkan skor probabilitas yang dihasilkan model. *Area Under the Curve* atau AUC merupakan luas area di bawah kurva ROC yang digunakan sebagai ukuran ringkas dari kemampuan separabilitas model.

Secara konseptual, TPR dan FPR dalam analisis kurva ROC dapat didefinisikan sebagai berikut:

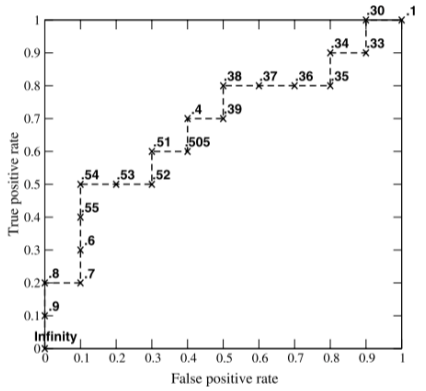
a. *true positive rate*

   TPR menunjukkan proporsi data kelas positif yang berhasil dikenali dengan benar oleh model. Dalam penelitian ini, TPR menggambarkan kemampuan model dalam mengenali mahasiswa yang benar-benar lulus tepat waktu.

b. *false positive rate*

FPR menunjukkan proporsi data kelas negatif yang salah diprediksi sebagai kelas positif. Dalam penelitian ini, FPR menggambarkan proporsi mahasiswa yang sebenarnya lulus terlambat tetapi diprediksi sebagai lulus tepat waktu.

Sebagai ilustrasi, visualisasi grafis dari kurva ROC beserta representasi nilai AUC untuk kasus klasifikasi biner ditunjukkan pada Gambar 2.1.



Gambar 2.1 Ilustrasi grafik kurva AUC-ROC

Sumber: (Fawcett, 2006)

Nilai AUC berada pada rentang 0 sampai 1. Nilai AUC yang semakin mendekati 1 menunjukkan bahwa model memiliki kemampuan yang semakin baik dalam membedakan kelas positif dan negatif. Sebaliknya, nilai AUC yang mendekati 0,5 menunjukkan bahwa kemampuan model dalam membedakan kedua kelas mendekati tebakan acak.

Dalam penelitian ini, AUC-ROC digunakan sebagai metrik tambahan setelah evaluasi berbasis *threshold* dilakukan. Hal ini karena *macro F1-score*, *precision*, *recall*, dan *confusion matrix* bergantung pada *threshold* tertentu, sedangkan AUC-ROC mengevaluasi kemampuan model pada berbagai nilai *threshold*. Dengan demikian, metrik ini digunakan untuk melengkapi evaluasi performa model.

2.8.6 *Permutation Feature Importance*

Permutation feature importance merupakan metode interpretasi model yang digunakan untuk mengukur kontribusi relatif setiap fitur terhadap performa prediksi model. Metode ini bekerja dengan mengacak nilai suatu fitur, kemudian mengamati perubahan performa model setelah fitur tersebut diacak. Apabila pengacakan suatu fitur menyebabkan penurunan performa yang besar, maka fitur tersebut dianggap memiliki kontribusi penting terhadap prediksi model.

Sebaliknya, apabila pengacakan fitur tidak banyak menurunkan performa, maka fitur tersebut dianggap memiliki kontribusi yang lebih kecil terhadap prediksi model.

Secara umum, *permutation feature importance* dapat dihitung dengan membandingkan skor awal model dengan skor model setelah nilai fitur tertentu diacak. Rumusnya dapat dituliskan sebagai berikut:

$$PFI_j = S_{baseline} - \frac{1}{R}\sum_{r=1}^{R} S_{permutedj}^{(r)} \tag{2.10}$$

Nilai Pada Persamaan (2.10), PFI_j menunjukkan nilai *permutation feature importance* pada fitur ke-*j*, $S_{baseline}$ menunjukkan skor performa model sebelum fitur diacak, $S_{permuted,j}^{(r)}$ menunjukkan skor performa model setelah fitur ke-*j* diacak pada pengulangan ke-*r*, dan *R* menunjukkan jumlah pengulangan permutasi. Nilai PFI_j yang semakin besar menunjukkan bahwa pengacakan fitur tersebut menyebabkan penurunan performa yang semakin tinggi, sehingga fitur tersebut memiliki kontribusi relatif yang lebih besar terhadap prediksi model.

Dalam penelitian ini, *permutation feature importance* diterapkan setelah model terbaik diperoleh dari proses pelatihan dan evaluasi. Perhitungan dilakukan terhadap fitur final yang telah melalui tahapan *feature engineering* dan *feature selection* pada *pipeline* terbaik. Skor performa yang digunakan dalam perhitungan *permutation feature importance* adalah *macro F1-score* agar konsisten dengan metrik utama penelitian. Dengan demikian, fitur yang dianalisis merupakan fitur yang benar-benar digunakan oleh *classifier* akhir dalam melakukan prediksi. Hasil *permutation feature importance* digunakan sebagai analisis tambahan untuk memahami kontribusi relatif fitur terhadap performa prediksi model. Namun, nilai *importance* tidak digunakan untuk menyimpulkan hubungan kausal antara fitur akademik dan ketepatan waktu kelulusan mahasiswa.

2.9 Penelitian Terdahulu

Penelitian mengenai prediksi tingkat kelulusan mahasiswa telah banyak dilakukan menggunakan berbagai pendekatan pembelajaran mesin. Tinjauan terhadap penelitian terdahulu diperlukan untuk memahami perkembangan metodologi, efektivitas algoritma yang pernah digunakan, serta karakteristik prapemrosesan data pada domain akademik. Pencarian penelitian terdahulu dalam subbab ini dilakukan melalui *Google Scholar* sebagai basis

pencarian utama. Literatur yang dipilih difokuskan pada artikel jurnal dan prosiding konferensi yang relevan dengan topik prediksi kelulusan mahasiswa, prediksi ketepatan waktu kelulusan, serta penerapan *machine learning* dalam domain pendidikan. Kata kunci yang digunakan dalam proses pencarian meliputi *student graduation prediction*, *on-time graduation prediction*, *student academic performance*, *missing data*, *gradient boosting*, serta 'status kelulusan mahasiswa'. Literatur yang digunakan difokuskan pada publikasi dalam rentang tahun 2021 hingga 2025, agar kajian yang digunakan tetap relevan dengan perkembangan metode *machine learning* terkini. Setiap literatur yang diperoleh kemudian diverifikasi melalui halaman resmi penerbit, jurnal, prosiding, atau repositori institusi untuk memastikan kesesuaian informasi bibliografis dan isi kajian. Secara umum, studi terdahulu berfokus pada optimasi metrik klasifikasi biner dan penanganan variasi kondisi *dataset*, seperti adanya data hilang (*missing value*) maupun penentuan batas waktu prediksi (*early prediction*). Rangkuman mengenai fokus, metode, *dataset*, dan temuan dari berbagai penelitian terdahulu yang relevan dengan penelitian ini disajikan pada Tabel 2.2.

Tabel 2.2 Penelitian Terdahulu

Penulis	**Fokus Penelitian**	**Metode**	***Dataset***	**Temuan**
(Dwinanda dkk., 2023)	Klasifikasi status kelulusan mahasiswa sedini mungkin untuk meningkatkan angka ketepatan waktu kelulusan mahasiswa.	*Machine learning* dengan algoritma *Extreme Gradient Boosting* (XGB).	140 baris data dan 11 kolom, dengan kolom terakhir berisi label target. 6 kolom diantaranya berisi data kategorikal, sedangkan sisanya berupa data numerik.	XGB optimal pada jumlah putaran (*round*) 3 dan jumlah lipatan (*fold*) 5. Dengan nilai *accuracy* 75%, *precision* 88.89%, *sensitivity* 76.19%, dan *specificity* 71.43%.
(Nirmala dkk., 2022)	memilih model *machine learning* yang paling akurat untuk menebak status kelulusan mahasiswa yang memiliki data hilang (*missing values*)	*Machine learning* dengan algoritma Random Forest dan XGB. Teknik imputasi MissForest diterapkan untuk menangani *missing values*.	13,377 baris data dan 17 kolom, dengan kolom terakhir berisi label target. 10 kolom diantaranya berisi data numerik, sedangkan sisanya berisi data kategorikal.	Metode imputasi MissForest pada model Random Forest dan XGB lebih superior dibanding metode imputasi mean dan median pada model yang sama. Selain itu, model XGB lebih superior dibanding Random

			Proporsi data yang hilang adalah 20.9% dari total sel.	Forest di banyak metrik, kecuali *sensitivity* dan G-mean.
(Aydin & Ozturk, 2021)	Membandingkan performa model klasifikasi XGB tanpa imputasi dengan XGB yang menggunakan metode imputasi.	Pengujian dilakukan menggunakan prosedur *10-fold cross-validation* dan dievaluasi secara statistik memakai *Friedman test*. Sedangkan metode imputasi nya meliputi K-Nearest Neighbor (KNN), MICE, Soft-Impute, mean, dan *class-mean*. Metrik yang digunakan adalah *F1-score*.	Menggunakan 10 jenis *benchmark dataset* terstruktur dari KEEL *Repository*. Data tersebut dimanipulasi menggunakan mekanisme *Missing Completely at Random* (MCAR).	Kesimpulan utama dari paper ini menyatakan bahwa metode imputasi data tidak memberikan pengaruh signifikan terhadap performa akhir XGB.
(Ayulani dkk., 2023)	Menentukan bulan paling ideal untuk mendeteksi dini risiko kegagalan mahasiswa pada suatu mata kuliah berbasis *online*, sekaligus membandingkan ketepatan tiga algoritma *tree-based ensemble* dalam memprediksi jumlah mahasiswa yang berisiko tidak lulus.	Membandingkan tiga algoritma, yaitu Random Forest, XGB, dan *Light Gradient Boosting Machine* (LGBM). Menggunakan metode SMOTE (*Synthetic Minority Over-sampling Technique*) untuk mengatasi masalah data tidak seimbang. Pengujian dilakukan dalam dua skenario, yaitu menggunakan seleksi fitur (*feature selection*) dan tanpa seleksi fitur. Data aktivitas dibagi ke dalam 3 periode waktu pemantauan, yaitu bulan pertama, dua bulan pertama, dan tiga bulan pertama masa studi.	Data log aktivitas pembelajaran *online* pada sistem LMS Universitas Indonesia yang bernama EMAS (*e-Learning Management System*). Ukuran data sangat kecil, terdiri dari 232 mahasiswa yang mengambil Mata Kuliah X (dengan rincian 219 mahasiswa lulus mata kuliah dan 13 mahasiswa tidak lulus mata kuliah).	Performa akademik mahasiswa paling optimal diprediksi pada akhir bulan kedua perkuliahan. Waktu ini dinilai terbaik karena memberikan tingkat akurasi yang tinggi sekaligus menyisakan waktu yang cukup bagi dosen untuk memberikan bantuan/intervensi sebelum semester berakhir. Model LGBM dengan seleksi fitur menghasilkan akurasi tertinggi sebesar 86.8%, mengungguli Random Forest (80%) dan XGB (75%).
(Novianto dkk., 2023)	Komparasi tiga algoritma klasifikasi yang dijalankan menggunakan *tool* RapidMiner untuk klasifikasi ketepatan waktu	Menggunakan sebuah *tool* bernama RapidMiner. Didalamnya terdapat fitur untuk menggunakan algoritma KNN, Naive Bayes, dan Decision Tree.	Menggunakan data 300 mahasiswa, dengan atribut NIM, jenis kelamin, IPS1-IPS7, dan IPK sebagai fitur,	Metode KNN menjadi algoritma terbaik dengan tingkat *accuracy* tertinggi mencapai 96.67%, disusul oleh Decision Tree (94.00%) dan Naive Bayes (77.33%).

	kelulusan mahasiswa.		serta status sebagai label.	
(Haryatmi & Pramita Hervianti, 2021)	Memprediksi ketepatan waktu kelulusan mahasiswa menggunakan *Support Vector Machine* (SVM).	Model prediksi dibangun menggunakan SVM. *Instance* yang memiliki *missing value* dihapus. Terdapat 3 skenario split, 90:10, 80:20, dan 70:30.	Data awal berjumlah 2192 baris, dengan atribut usia, jenis kelamin, IPS1-IPS4, IPK, jurusan, total SKS, dan lama masa studi sebagai fitur, serta status sebagai label.	Model SVM terbukti kuat karena berhasil mempertahankan nilai *accuracy* pada *train* dan *test set* di 3 skenario berbeda. Namun, terdapat indikasi bias performa akibat data *imbalance*, di mana model kesulitan mengenali kelas minoritas (mahasiswa tepat waktu) dengan *error* mencapai 31.5% - 42.8% pada *confusion matrix*.
(Hubbard & Amponsah, 2025)	penerapan *feature engineering* pada data *Learning Management System* (LMS) untuk meningkatkan prediksi performa mahasiswa, khususnya berdasarkan data *login* dan nilai pada LMS.	Penelitian ini membandingkan berbagai fitur hasil rekayasa dari data LMS *login* dan LMS *grades*, kemudian mengevaluasinya menggunakan beberapa model *machine learning* seperti LGBM, XGBoost, Random Forest, ExtraTrees, dan SGD.	*Dataset* berasal dari data LMS dan data akademik mahasiswa sarjana pada sebuah universitas regional di Amerika Serikat, mencakup sembilan semester dari *Fall* 2020 hingga *Fall* 2024, dengan 85.848 data *student-semester*, 19.977.140 data *login*, dan lebih dari 10 juta data nilai LMS.	Hasil penelitian menunjukkan bahwa rekayasa fitur pada data LMS dapat meningkatkan daya prediksi terhadap performa mahasiswa. Fitur LMS *grades* lebih kuat dalam memprediksi GPA, sedangkan fitur *periodic logins* lebih berkontribusi dalam prediksi *discontinuance*. Kombinasi data LMS dengan data akademik tradisional juga menghasilkan model prediksi yang lebih baik dibandingkan penggunaan data LMS secara terpisah.
(Bakri dkk., 2025)	prediksi ketepatan waktu kelulusan mahasiswa dengan membandingkan	Penelitian ini menggunakan tujuh algoritma *machine learning*, yaitu Random Forest, SVM-RBF,	*Dataset* berasal dari sistem informasi akademik STIEM Bongaya	Hasil penelitian menunjukkan bahwa kombinasi Random Forest dengan *resampling* RUS-ROS

	beberapa algoritma *machine learning* serta mengevaluasi pengaruh *resampling* dan *hyperparameter tuning* pada *dataset* yang tidak seimbang.	XGBoost, Naïve Bayes, K-Nearest Neighbors, Neural Network, dan AdaBoost. Penanganan ketidakseimbangan data dilakukan menggunakan RUS, ROS, RUS-ROS, SMOTENC, dan SMOTENC-RUS, sedangkan optimasi *hyperparameter* dilakukan menggunakan *Grid Search* dengan validasi *hold-out* dan *10-fold cross-validation.*	University, terdiri dari 4.093 data mahasiswa dengan 15 variabel prediktor. Target klasifikasi berupa status kelulusan tepat waktu, dengan distribusi kelas 3.071 mahasiswa lulus tepat waktu dan 1.022 mahasiswa lulus terlambat.	menghasilkan performa terbaik dengan *average metric score* sebesar 0,948 dan *F1-score* sebesar 0,942. Penelitian ini juga menunjukkan bahwa kombinasi *resampling* dan *hyperparameter tuning* dapat meningkatkan performa klasifikasi pada *dataset* pendidikan yang tidak seimbang.

Berdasarkan Tabel 2.2, penelitian terdahulu mengenai prediksi kelulusan, ketepatan waktu kelulusan, dan performa akademik mahasiswa dapat dianalisis melalui beberapa bingkai utama. Dari sisi metode, sebagian penelitian menggunakan algoritma klasifikasi konvensional, seperti K-Nearest Neighbor (KNN), Naïve Bayes, Decision Tree, Support Vector Machine (SVM), Random Forest, dan Neural Network. Pendekatan ini terlihat pada penelitian Novianto dkk. (2023) yang membandingkan KNN, Naïve Bayes, dan Decision Tree, serta Haryatmi dan Pramita Hervianti (2021) yang menggunakan SVM untuk memprediksi ketepatan waktu kelulusan mahasiswa. Di sisi lain, beberapa penelitian mulai menggunakan model berbasis *ensemble* dan *gradient boosting*, seperti XGBoost, LightGBM, AdaBoost, ExtraTrees, dan model sejenis. Hal ini terlihat pada penelitian Dwinanda dkk. (2023), Nirmala dkk. (2022), Aydin dan Ozturk (2021), Ayulani dkk. (2023), Hubbard dan Amponsah (2025), serta Bakri dkk. (2025). Penggunaan model *ensemble* dan *gradient boosting* menunjukkan bahwa penelitian terdahulu mulai mengarah pada model yang mampu menangkap pola nonlinear dan hubungan kompleks antarfitur pada data akademik.

Dari sisi teknik pendukung, penelitian terdahulu tidak hanya berfokus pada pemilihan algoritma, tetapi juga menerapkan beberapa strategi pemrosesan data dan optimasi model. Nirmala dkk. (2022) dan Aydin dan Ozturk (2021) secara khusus membahas pengaruh teknik imputasi data hilang terhadap performa XGBoost, sedangkan Ayulani dkk. (2023) dan Bakri dkk. (2025) menerapkan teknik *resampling* untuk menangani ketidakseimbangan kelas. Selain itu, Ayulani dkk. (2023) juga membandingkan skenario dengan dan tanpa *feature selection*,

sementara Bakri dkk. (2025) mengombinasikan *resampling* dengan *hyperparameter tuning*. Hal ini menunjukkan bahwa performa model pada domain akademik tidak hanya dipengaruhi oleh jenis algoritma, tetapi juga oleh strategi prapemrosesan, keseimbangan data, pemilihan fitur, dan konfigurasi *hyperparameter* yang digunakan.

Dari sisi *dataset* dan fitur, penelitian terdahulu menggunakan karakteristik data yang cukup beragam. Beberapa penelitian menggunakan data akademik terstruktur, seperti IPS, IPK, SKS, status kelulusan, lama studi, usia, jenis kelamin, dan jurusan, sebagaimana terlihat pada penelitian Novianto dkk. (2023), Haryatmi dan Pramita Hervianti (2021), Dwinanda dkk. (2023), Nirmala dkk. (2022), dan Bakri dkk. (2025). Sementara itu, penelitian lain menggunakan data aktivitas pembelajaran dari *Learning Management System* (LMS), seperti log aktivitas, *login*, nilai LMS, dan data performa pembelajaran *online*, sebagaimana terlihat pada penelitian Ayulani dkk. (2023) serta Hubbard dan Amponsah (2025). Perbedaan jenis data tersebut menunjukkan bahwa prediksi pada domain *Educational Data Mining* (EDM) dapat dilakukan menggunakan data akademik formal maupun data aktivitas pembelajaran, tetapi sebagian besar penelitian masih berfokus pada fitur langsung atau fitur mentah yang tersedia pada sistem akademik dan LMS.

Dari sisi performa, hasil penelitian terdahulu menunjukkan bahwa tidak ada satu algoritma yang selalu menjadi model terbaik pada seluruh dataset. Novianto dkk. (2023) memperoleh akurasi tertinggi menggunakan KNN sebesar 96,67%, sedangkan Bakri dkk. (2025) memperoleh performa tinggi melalui kombinasi Random Forest dengan *resampling* RUS-ROS, yaitu *average metric score* sebesar 0,948 dan *F1-score* sebesar 0,942. Pada penelitian Ayulani dkk. (2023), LightGBM dengan *feature selection* menghasilkan akurasi tertinggi sebesar 86,8%, mengungguli Random Forest dan XGBoost. Sementara itu, Dwinanda dkk. (2023) menunjukkan bahwa XGBoost memperoleh akurasi sebesar 75%, dan Nirmala dkk. (2022) menunjukkan bahwa XGBoost cenderung lebih unggul dibandingkan Random Forest pada banyak metrik ketika dikombinasikan dengan teknik imputasi tertentu. Perbedaan hasil tersebut memperlihatkan bahwa performa model sangat dipengaruhi oleh karakteristik *dataset*, target prediksi, teknik prapemrosesan, skenario evaluasi, serta metrik yang digunakan.

Berdasarkan sintesis tersebut, terdapat beberapa celah penelitian yang menjadi posisi utama dalam penelitian ini. Beberapa penelitian terdahulu telah memanfaatkan model *machine learning*, termasuk model berbasis *gradient boosting*, serta menerapkan teknik pendukung seperti imputasi data, *feature selection*, *resampling*, dan *hyperparameter tuning*. Namun, penerapan teknik-teknik tersebut belum banyak diarahkan secara khusus pada prediksi awal

ketepatan waktu kelulusan mahasiswa berdasarkan riwayat performa akademik hingga semester enam. Selain itu, sebagian penelitian masih menitikberatkan penggunaan fitur akademik mentah, seperti IP, IPK, SKS, status kelulusan, dan lama studi, atau menggunakan data aktivitas LMS yang tidak selalu tersedia pada seluruh konteks perguruan tinggi. Berdasarkan celah penelitian tersebut, subbab berikutnya menyajikan kerangka pemikiran yang menjelaskan alur konseptual penelitian, mulai dari pemanfaatan data akademik hingga semester enam, pembentukan dan pemilihan fitur, penerapan model *gradient boosting*, hingga evaluasi performa model.

2.10 Kerangka Pemikiran

Kerangka pemikiran dalam penelitian ini disusun untuk menggambarkan alur konseptual mulai dari pemanfaatan data akademik sampai diperolehnya model terbaik untuk prediksi ketepatan waktu kelulusan mahasiswa. Tahap pertama adalah mengidentifikasi data akademik yang tersedia dan relevan sebagai dasar prediksi. Berdasarkan data yang dimiliki, nilai Indeks Prestasi (IP), Indeks Prestasi Kumulatif (IPK), dan jumlah Satuan Kredit Semester (SKS) pada semester satu hingga semester enam merupakan atribut akademik yang menggambarkan riwayat capaian mahasiswa sebelum memasuki tahap akhir masa studi.

Dalam penelitian ini, fitur IP dan SKS dipilih sebagai fitur dasar untuk membentuk fitur turunan. IP digunakan karena dapat menunjukkan capaian akademik mahasiswa pada setiap semester, sedangkan SKS digunakan untuk merepresentasikan beban studi yang ditempuh mahasiswa. Sementara itu, IPK tidak digunakan karena merupakan nilai kumulatif yang telah merangkum capaian IP dari semester sebelumnya, sehingga berpotensi menghasilkan informasi yang tumpang tindih dengan fitur IP semester. Data akademik tersebut kemudian diolah melalui proses *feature engineering* untuk membentuk fitur turunan yang mencakup perubahan IP antarsemester, variasi IP, nilai IP minimum, nilai IP maksimum, rentang IP, serta rata-rata beban SKS per semester. Pembentukan fitur ini dilakukan karena ketepatan waktu kelulusan tidak hanya dipengaruhi oleh capaian akademik pada satu semester tertentu, tetapi juga oleh pola perkembangan akademik mahasiswa dari waktu ke waktu.

Setelah fitur turunan dibentuk, dilakukan proses *feature selection* berbasis korelasi Pearson untuk memilih fitur yang memiliki hubungan relevan terhadap target prediksi serta mengurangi fitur yang memiliki korelasi tinggi antarfitur. Tahapan ini bertujuan agar model tidak hanya menggunakan seluruh fitur yang tersedia, tetapi berfokus pada fitur yang lebih

informatif dalam membedakan mahasiswa yang berpotensi lulus tepat waktu dan terlambat. Fitur hasil seleksi kemudian digunakan sebagai masukan bagi model *gradient boosting*.

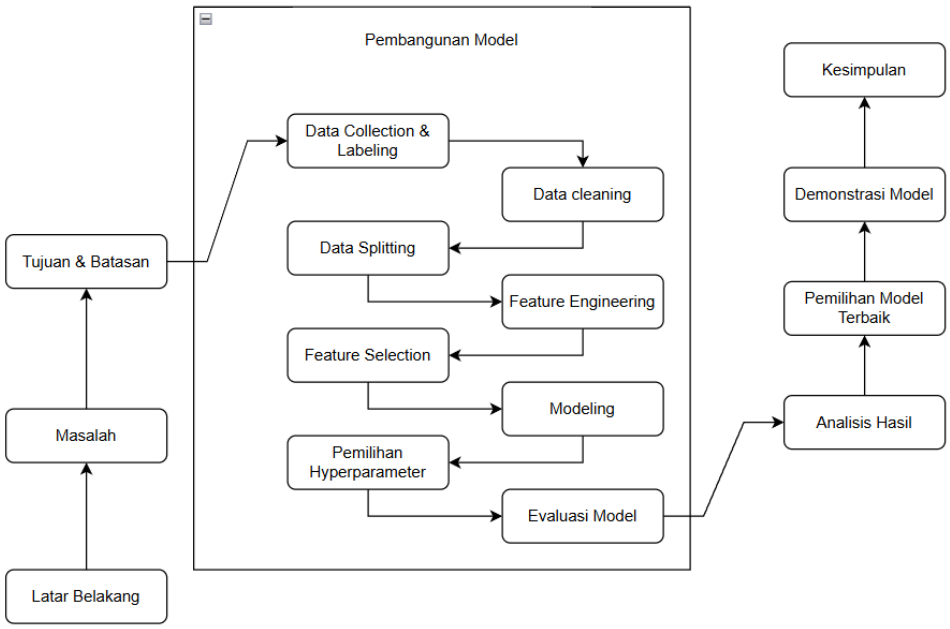
Model *gradient boosting* digunakan karena model ini mampu membangun prediksi secara bertahap melalui kombinasi beberapa model lemah sehingga dapat menangkap pola nonlinear dan interaksi antarfitur pada data akademik. Dalam proses pemodelan, dilakukan *hyperparameter tuning* untuk memperoleh konfigurasi model yang sesuai dengan karakteristik data. Setelah model dilatih, dilakukan *threshold optimization* untuk menyesuaikan batas probabilitas agar hasil prediksi lebih sesuai dengan tujuan evaluasi penelitian.

Evaluasi model dilakukan menggunakan data uji yang tidak digunakan dalam proses pelatihan dan optimasi. Metrik evaluasi digunakan untuk menilai kemampuan model dalam memprediksi mahasiswa yang lulus tepat waktu maupun terlambat. Berdasarkan hasil evaluasi tersebut, model dengan performa terbaik dipilih sebagai model akhir untuk prediksi awal ketepatan waktu kelulusan mahasiswa.

BAB III
METODOLOGI PENELITIAN

3.1 Alur Penelitian

Penelitian ini dilakukan melalui delapan tahapan utama, yaitu *data collection dan labeling*, *data cleaning*, *data splitting*, *feature engineering*, *feature selection*, *modeling*, pemilihan *hyperparameter*, dan evaluasi model. Tahapan-tahapan tersebut dirancang untuk membangun model prediksi ketepatan waktu kelulusan mahasiswa berdasarkan data akademik hingga semester enam. Alur penelitian secara utuh dirangkum pada Gambar 3.1 dalam bentuk diagram alir penelitian.



Gambar 3.1 Alur penelitian

Berdasarkan Gambar 3.1, Tahap pertama adalah pengumpulan data akademik mahasiswa melalui Direktorat Layanan Akademik UII. Tahap kedua adalah *data cleaning*, yaitu proses pembersihan data dengan membuang atribut dan label yang tidak relevan dengan tujuan penelitian. Tahap ketiga adalah *data splitting*, yaitu pembagian data menjadi data latih dan data uji. Tahap keempat adalah *feature engineering*, yaitu pembentukan fitur turunan dari data IP dan SKS mahasiswa hingga semester enam. Tahap kelima adalah *feature selection*, yaitu proses pemilihan fitur yang akan digunakan dalam pemodelan. Tahap keenam adalah *modeling*, yaitu proses pembangunan model prediksi menggunakan beberapa algoritma *gradient boosting*.

Tahap ketujuh adalah pemilihan *hyperparameter* yang dilakukan secara terstruktur menggunakan skema pencarian dan validasi silang pada data latih untuk memperoleh kombinasi konfigurasi yang paling sesuai. Tahap terakhir adalah evaluasi model, yaitu proses pengukuran performa model yang telah dibangun.

Selain tahapan pembangunan model, penelitian ini juga menyertakan demonstrasi penggunaan model melalui aplikasi sederhana berbasis Streamlit. Demonstrasi tersebut tidak menjadi bagian dari proses pelatihan maupun evaluasi model, melainkan digunakan untuk menunjukkan bagaimana model terbaik yang telah diperoleh dapat menerima input data akademik mahasiswa dan menghasilkan prediksi ketepatan waktu kelulusan. Pembahasan mengenai demonstrasi model disajikan pada Subbab 4.8 setelah hasil evaluasi model dijelaskan.

3.2 Lingkungan Penelitian

Penelitian ini dilakukan menggunakan bahasa pemrograman Python. Google Colaboratory digunakan karena menyediakan lingkungan pemrograman berbasis *cloud* yang mendukung proses eksplorasi data, pembangunan *pipeline* pemodelan, optimasi *hyperparameter*, evaluasi model, serta visualisasi hasil eksperimen. Seluruh proses pemodelan dilakukan dalam *notebook* Python, mulai dari tahap pemrosesan data, *feature engineering*, *feature selection*, pelatihan model, pemilihan *hyperparameter*, optimasi *threshold*, hingga evaluasi akhir. Sebagai tambahan, Microsoft Excel digunakan untuk melakukan proses pelabelan secara manual.

Selain digunakan untuk eksperimen pemodelan, penelitian ini juga memanfaatkan beberapa pustaka Python untuk mendukung proses analisis data dan *machine learning*. Pustaka yang digunakan meliputi pustaka untuk manipulasi data, pemodelan *machine learning*, optimasi *hyperparameter*, evaluasi model, visualisasi, serta penyimpanan model. Ringkasan perangkat lunak dan pustaka utama yang digunakan dalam penelitian ini ditunjukkan pada Tabel 3.1.

Tabel 3.1 Perangkat Lunak dan Pustaka Pendukung

No.	Perangkat Lunak/Pustaka	Versi	Fungsi
1	Microsoft Excel	2021	Perangkat untuk mengolah data tabular.
2	Google Colaboratory	-	Lingkungan komputasi berbasis *cloud* untuk menjalankan eksperimen pemodelan.
3	Python	3.12.13	Bahasa pemrograman utama yang digunakan dalam implementasi penelitian.

4	SciPy	1.16.3	*Library* untuk keperluan komputasi saintifik.
5	lightgbm	4.6.0	*Library* untuk penggunaan model LGBM.
6	xgboost	3.2.0	*Library* untuk penggunaan model XGB.
7	scikit-learn	1.6.1	*Library* pembuatan *pipeline*, *data splitting*, *cross-validation*, evaluasi model, proses optimasi, serta implementasi model HGB.
8	joblib	1.5.3	*Library* untuk menyimpan model hasil pelatihan ke dalam format file.

3.3 *Data Collection and Labeling*

3.3.1 *Data Collection*

Dalam penelitian ini, data diperoleh melalui mekanisme permohonan resmi kepada Direktorat Layanan Akademik Universitas Islam Indonesia melalui Kepala Divisi Administrasi Akademik Fakultas Teknologi Industri UII. Data yang digunakan berasal dari mahasiswa Program Sarjana pada Program Studi Informatika UII untuk tiga angkatan, yaitu angkatan 2019, 2020, dan 2021. Setiap baris data merepresentasikan satu mahasiswa, sedangkan setiap kolom merepresentasikan atribut akademik atau atribut pendukung yang berkaitan dengan riwayat studi mahasiswa. Data yang diterima hanya digunakan untuk kepentingan penelitian. Selain itu, kerahasiaan identitas mahasiswa dijaga dan tidak digunakan dalam proses pelatihan model.

Data yang diterima mencakup rekam jejak akademik mahasiswa berupa Indeks Prestasi (IP), Indeks Prestasi Kumulatif (IPK), dan jumlah Satuan Kredit Semester (SKS) yang ditempuh pada setiap semester. Selain itu, data awal juga memuat atribut identitas berupa Nomor Induk Mahasiswa (NIM) serta atribut pendukung berupa akumulasi total jumlah kehadiran (jml_hadir), jumlah izin (jmlh_izin), dan jumlah cuti (jml_cuti) mahasiswa selama masa studi. Atribut pendukung tersebut tersedia dalam bentuk agregat kumulatif dan tidak disajikan per semester.

Cakupan semester pada data mentah berbeda untuk setiap angkatan. Perbedaan tersebut terjadi karena masing-masing angkatan memiliki posisi masa studi yang berbeda pada saat data diperoleh. Rangkuman jumlah data dan cakupan semester pada data mentah dapat dilihat pada Tabel 3.2.

Tabel 3.2 Cakupan Data Akademik per Angkatan

Angkatan	Jumlah Data	Rentang Semester pada Data mentah
2019	190	Semester 1-13
2020	198	Semester 1-11
2021	179	Semester 1-9
Total	567	Bervariasi

Berdasarkan Tabel 3.2, data mentah yang diperoleh terdiri atas 567 baris data dari tiga angkatan mahasiswa. Rentang semester maksimum pada setiap angkatan tidak sama, sehingga diperlukan pembatasan cakupan atribut akademik agar fitur yang digunakan dalam pemodelan memiliki struktur yang konsisten pada seluruh angkatan. Oleh karena itu, penelitian ini membatasi atribut akademik yang digunakan hingga semester enam. Pembatasan ini juga disesuaikan dengan tujuan penelitian, yaitu melakukan prediksi awal ketepatan waktu kelulusan mahasiswa berdasarkan riwayat akademik sebelum mahasiswa memasuki tahap akhir masa studi.

Dengan pembatasan tersebut, atribut akademik setelah semester enam tidak digunakan dalam proses pemodelan. Hal ini dilakukan untuk menghindari penggunaan informasi yang terlalu dekat dengan kondisi kelulusan mahasiswa. Namun, pada tahap ini atribut identitas dan atribut pendukung yang berasal dari data mentah masih tetap dipertahankan sebelum dilakukan proses pembersihan data pada tahap berikutnya.

3.3.2 *Labeling*

Proses pelabelan pada penelitian ini dilakukan secara manual sesuai dengan batasan masalah yang telah ditetapkan, dengan mengacu pada kriteria kelulusan dalam Panduan Akademik Program Studi Informatika UII Tahun 2021/2022 dan Materi Penjelasan Akademik Program Studi Informatika UII Tahun 2022. Berdasarkan dokumen Panduan Akademik, mahasiswa dinyatakan lulus tepat waktu apabila menyelesaikan studi dalam rentang 3,5 hingga 4 tahun (Program Studi Informatika UII, 2021, hlm. 34). Selain itu, dokumen Penjelasan Akademik menetapkan aturan evaluasi tengah masa studi, di mana mahasiswa yang tidak memenuhi minimal 40 SKS dengan IPK minimal 2,00 dalam 4 semester pertama akan dikeluarkan (Program Studi Informatika UII, 2022, hlm. 64). Sebagai tambahan, masa studi maksimal yang diperkenankan adalah 14 semester atau 7 tahun, di mana mahasiswa yang melampaui batas tersebut atau tidak memenuhi evaluasi tengah masa studi dikategorikan sebagai *drop out*.

Pelabelan dilakukan berdasarkan evaluasi kondisi secara berurutan dengan prioritas sebagai berikut:

a. Label 1 (Tepat Waktu): Mahasiswa diklasifikasikan lulus tepat waktu apabila total SKS yang ditempuh ≥ 144, IPK pada semester 7 atau 8 ≥ 2,25, dan tidak terdapat aktivitas akademik pada semester 9 dan seterusnya, yang diidentifikasi dari tidak adanya nilai IP yang tercatat pada semester tersebut.
b. Label 0 (Terlambat): Apabila kondisi label 1 tidak terpenuhi, mahasiswa diklasifikasikan lulus terlambat apabila total SKS ≥ 144, IPK terakhir yang tercatat ≥ 2,25, dan terdapat aktivitas akademik pada semester 9 dan seterusnya.
c. Label 3 (Anomali): Apabila kondisi label 1 dan 0 tidak terpenuhi, mahasiswa diklasifikasikan anomali apabila tidak lulus evaluasi tengah studi namun masih terdapat rekam jejak akademik aktif pada semester 5 dan seterusnya.
d. Label 2 (*Drop Out*): Apabila kondisi label 1, 0, dan 3 tidak terpenuhi, mahasiswa diklasifikasikan *drop out* apabila tidak lulus evaluasi tengah studi dan tidak terdapat aktivitas akademik lanjut, atau telah menjalani non-aktif selama empat semester berturut-turut.
e. Label 4 (Belum Lulus): Label *default* yang diberikan apabila tidak ada kondisi di atas yang terpenuhi, termasuk kondisi data yang tidak dapat dievaluasi.

Namun, perlu dicatat bahwa pelabelan dalam penelitian ini didasarkan sepenuhnya pada data IP, IPK, dan SKS per semester yang tersedia, tanpa akses terhadap data nilai per mata kuliah, jumlah nilai C/D, maupun status resmi kelulusan yang dipegang institusi. Oleh karena itu, label yang ditetapkan merupakan hasil rekonstruksi berdasarkan indikator akademik yang tersedia dan dicocokkan dengan ketentuan masa studi yang berlaku. Keterbatasan ini perlu diakui sebagai salah satu limitasi penelitian, mengingat terdapat kemungkinan perbedaan antara label yang ditetapkan dengan status akademik resmi masing-masing mahasiswa. Selain itu, kualitas model akan sangat bergantung pada validitas *manual labeling* yang dilakukan. Dokumentasi aturan pelabelan, penjelasan kolom pendukung, serta rekap distribusi label awal disajikan pada Lampiran A. Lampiran tersebut digunakan sebagai artefak pendukung agar proses pembentukan label dapat ditelusuri tanpa menampilkan data mentah mahasiswa secara langsung.

Setelah proses *labeling* dilakukan, *dataset* kemudian disesuaikan dengan tujuan prediksi awal pada penelitian ini. Penyesuaian tersebut dilakukan dengan membatasi atribut akademik hanya sampai semester enam. Pada tahap ini, *dataset* masih memuat atribut NIM, IP semester

1 sampai 6, IPK semester 1 sampai 6, SKS semester 1 sampai 6, jumlah kehadiran, jumlah izin, jumlah cuti, dan status. Rangkuman atribut pada *dataset* setelah *labeling* dan pembatasan semester dapat dilihat pada Tabel 3.3.

Tabel 3.3 Atribut *Dataset* Setelah *Labeling* dan Pembatasan Semester

Atribut	Deskripsi	Tipe Data
NIM	Nomor Induk Mahasiswa	Integer
IP1-IP6	Indeks Prestasi	Float
IPK1-IPK6	Indeks Prestasi Kumulatif	Float
SKS1-SKS6	Satuan Kredit Semester	Integer
Jml_hadir	Jumlah hadir	Integer
Jml_izin	Jumlah izin	Integer
Jml_cuti	Jumlah cuti kuliah	Integer
status	Kategori status kelulusan	Integer

Berdasarkan Tabel 3.3, *dataset* setelah *labeling* dan pembatasan semester masih memuat atribut yang belum seluruhnya digunakan dalam proses pemodelan. Atribut NIM masih terdapat dalam *dataset* sebagai identitas mahasiswa, sedangkan atribut jumlah kehadiran, jumlah izin, dan jumlah cuti masih terdapat sebagai atribut pendukung dari data mentah. Keputusan untuk mempertahankan atau menghapus atribut tersebut dilakukan pada tahap *data cleaning*.

3.4 *Data Cleaning*

Proses *data cleaning* dilakukan dalam tiga tahap. Pertama, *label filtering* dilakukan dengan mempertahankan *instance* berlabel 0 dan 1, sedangkan *instance* dengan label 2, 3, dan 4 dikeluarkan dari *dataset* karena tidak relevan dengan tujuan klasifikasi biner yang ditetapkan. Kedua, *feature dropping* dilakukan dengan menghapus atribut NIM, jumlah kehadiran, jumlah izin, dan jumlah cuti. Atribut NIM dihapus karena hanya berfungsi sebagai identitas mahasiswa dan tidak memiliki nilai prediktif. Sementara itu, ketiga atribut lainnya dihapus karena merupakan data agregat kumulatif selama masa studi, yang berpotensi menyebabkan *data leakage* apabila digunakan dalam pemodelan prediktif. Tahap terakhir adalah pengecekan nilai hilang (*missing values*), tahap ini dilakukan untuk mengetahui proporsi *missing values* dalam data. Pengecekan dilakukan dengan cara membandingkan jumlah *missing values* pada setiap kolom fitur dengan jumlah keseluruhan data. Pada penelitian ini, teknik imputasi tidak dilakukan dengan mempertimbangkan informasi nonakademik yang dimiliki oleh *missing values*. Selain itu, model *gradient boosting* yang digunakan pada penelitian ini mampu

menangani nilai hilang secara mandiri, sebagaimana yang telah dijabarkan pada Subbab 2.6. Rekap hasil penyaringan label, distribusi data setelah proses *cleaning*, dan persentase *missing values* pada data bersih disajikan pada Lampiran A sebagai dokumentasi pendukung proses *cleaning*.

3.5 *Data Splitting*

Dalam penelitian ini, *data splitting* dilakukan setelah tahap *data cleaning* dan sebelum tahap *feature engineering* untuk menghindari potensi *data leakage*. Proses *data splitting* memiliki dua alur yang berbeda. Skenario pertama adalah pembagian data gabungan, yaitu seluruh data dari setiap angkatan digabungkan menjadi satu *dataset*, kemudian dibagi menjadi data latih dan data uji. Skenario kedua adalah pembagian data berdasarkan angkatan, yaitu setiap angkatan secara bergantian digunakan sebagai data uji, sedangkan dua angkatan lainnya digunakan sebagai data latih. Kedua skenario ini digunakan untuk tujuan yang berbeda. Skenario data gabungan digunakan sebagai skenario utama dalam pemodelan dan perbandingan performa algoritma, sedangkan skenario data berdasarkan angkatan digunakan sebagai evaluasi tambahan untuk menilai ketahanan model terhadap perbedaan karakteristik antarangkatan pada model terbaik.

3.5.1 Skenario Data Gabungan

Pada proses ini, data dibagi menjadi *training set* dan *test set* dengan rasio 80:20. Teknik *stratified sampling* diterapkan selama proses pemisahan untuk memastikan bahwa distribusi label target (status kelulusan mahasiswa) tetap seimbang dan representatif pada kedua subset data. Guna menjamin aspek reproduksibilitas riset, *random state* dikunci pada nilai 42 di seluruh tahapan komputasi. Lebih lanjut, data *test set* murni diisolasi secara eksklusif dan hanya digunakan pada tahap evaluasi akhir sehingga tidak terlibat dalam proses pelatihan maupun pemilihan konfigurasi model.

Skenario ini digunakan sebagai dasar pemilihan model terbaik dengan asumsi bahwa data latih dan data uji berasal dari distribusi yang relatif serupa. Dengan skenario tersebut, performa setiap model dapat dibandingkan pada kondisi pembagian data yang sama. Rekap jumlah data latih dan data uji beserta distribusi label pada setiap subset disajikan pada Lampiran A.

3.5.2 Skenario Data Berdasarkan Angkatan

Pada skenario data berdasarkan angkatan, data dibagi menjadi tiga kelompok dengan menjadikan satu angkatan secara bergantian sebagai *test set*, sedangkan dua angkatan lainnya digunakan sebagai *training set*. Pada *Group* 1, data angkatan 2019 digunakan sebagai *test set*, sedangkan data angkatan 2020 dan 2021 digunakan sebagai *training set*. Pada *Group* 2, data angkatan 2020 digunakan sebagai *test set*, sedangkan data angkatan 2019 dan 2021 digunakan sebagai *training set*. Selanjutnya, pada *Group* 3, data angkatan 2021 digunakan sebagai *test set*, sedangkan data angkatan 2019 dan 2020 digunakan sebagai *training set*.

Skenario ini digunakan untuk mengevaluasi ketahanan performa model ketika diterapkan pada angkatan yang tidak dilibatkan dalam proses pelatihan. Melalui pembagian tersebut, perbedaan performa antarkelompok dapat digunakan untuk mengidentifikasi indikasi adanya perbedaan karakteristik antarangkatan. Rekap jumlah data latih dan data uji beserta distribusi label pada masing-masing kelompok disajikan pada Lampiran A.

Tahapan setelah proses *data splitting*, mulai dari *feature engineering* hingga *hyperparameter tuning*, diterapkan dengan prosedur yang sama pada kedua skenario. Namun, skenario data berdasarkan angkatan hanya diterapkan pada model terbaik yang telah dipilih berdasarkan hasil evaluasi pada skenario data gabungan. Dengan demikian, skenario ini tidak digunakan untuk melakukan pemilihan model ulang, melainkan sebagai evaluasi tambahan untuk menilai ketahanan model terpilih terhadap perbedaan karakteristik antarangkatan.

3.6 *Feature Engineering*

Proses *feature engineering* dilakukan setelah tahap *data splitting* dan sebelum tahap *feature selection*. Tahap ini menggunakan data akademik mahasiswa berupa nilai Indeks Prestasi (IP) dan Satuan Kredit Semester (SKS) dari semester satu hingga semester enam. Tujuan dari tahap ini adalah membentuk fitur turunan yang mampu merepresentasikan perubahan, variasi, dan ringkasan performa akademik mahasiswa sebelum memasuki tahun keempat masa studi.

Pembentukan fitur perubahan antarperiode dilakukan untuk merepresentasikan perubahan performa akademik mahasiswa dari satu semester ke semester berikutnya. Pada penelitian ini, konsep tersebut diterapkan melalui pembentukan fitur delta IP, yaitu selisih IP antarsemester berturut-turut. Fitur ini digunakan untuk menangkap arah perubahan performa akademik mahasiswa, baik berupa peningkatan maupun penurunan nilai IP dari semester ke semester.

Selain itu, penelitian ini juga membentuk fitur ringkasan statistik dari data IP dan SKS semester satu hingga semester enam. Fitur ringkasan statistik digunakan untuk merangkum karakteristik performa akademik mahasiswa dalam bentuk yang lebih padat. Fitur yang dibentuk meliputi nilai minimum IP, maksimum IP, rentang IP, varians IP, serta rata-rata SKS per semester. Pembentukan fitur-fitur ini dilakukan agar model tidak hanya mempelajari nilai IP dan SKS mentah pada tiap semester, tetapi juga memperoleh informasi tambahan mengenai stabilitas, variasi, dan kecenderungan umum performa akademik mahasiswa selama enam semester.

Misalkan x_t menyatakan nilai IP semester pada semester ke-*t*, dan c_t menyatakan beban kredit SKS semester pada semester ke-*t*, dengan rentang waktu $t = 1, 2, \ldots, T$ di mana batasan pemotongan data akhir dalam penelitian ini adalah T = 6. Maka formula yang digunakan untuk menghitung fitur turunan dapat dijabarkan sebagai berikut:

a. Perbedaan IP (delta IP)

$$\Delta x_t = x_t - x_{t-1} \tag{3.1}$$

b. Rerata SKS per semester

$$\bar{c} = \frac{1}{T}\sum_{t=1}^{T} c_t \tag{3.2}$$

c. Varians IP

$$Var(x) = \frac{1}{T-1}\sum_{t=1}^{T}(x_t - \bar{x})^2 \tag{3.3}$$

d. Minimum IP

$$x_{min} = \min_{t \in \{1,\ldots,T\}} x_t \tag{3.4}$$

e. Maximum IP

$$x_{max} = \max_{t \in \{1,\ldots,T\}} x_t \tag{3.5}$$

f. Rentang IP (Gap IP)

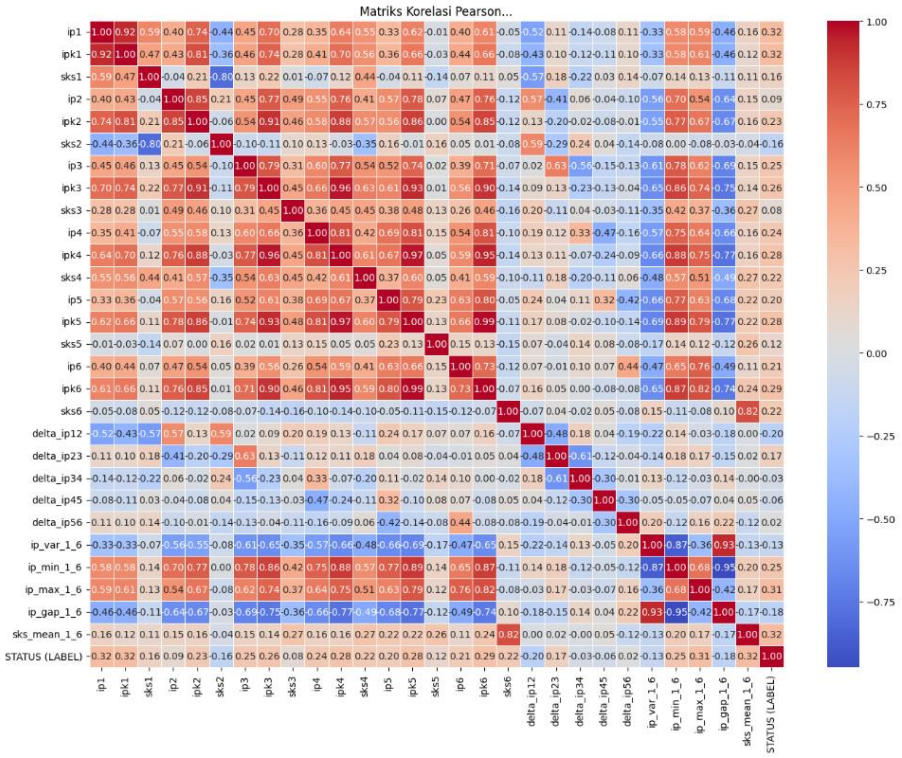
$$Range(x) = x_{max} - x_{min} \tag{3.6}$$

Berdasarkan formulasi yang telah dijelaskan, proses *feature engineering* menghasilkan sejumlah fitur turunan yang merepresentasikan perubahan, variasi, dan ringkasan performa akademik mahasiswa selama enam semester. Seluruh fitur turunan tersebut dibentuk secara otomatis melalui komponen *feature engineering* yang diintegrasikan ke dalam *pipeline* pemodelan. Integrasi ini dilakukan agar proses pembentukan fitur berlangsung secara konsisten pada data latih, data validasi pada proses *cross-validation*, dan data uji. Dengan demikian, fitur turunan yang digunakan pada tahap pelatihan dan pengujian dibentuk melalui prosedur yang sama. Pada tahap ini, penelitian juga mengevaluasi strategi penggunaan fitur dasar melalui parameter drop_raw. Parameter tersebut menentukan apakah fitur akademik awal berupa IP dan SKS semester satu hingga semester enam tetap dipertahankan atau dihapus setelah fitur turunan dibentuk. Nilai parameter ini tidak ditentukan secara manual, melainkan dimasukkan ke dalam ruang pencarian *hyperparameter* pada *Random Search* agar pemilihan strategi fitur dapat dilakukan berdasarkan performa model. Daftar lengkap fitur turunan yang dibentuk pada tahap *feature engineering* disajikan pada Lampiran B.

3.7 *Feature Selection*

3.7.1 Analisis Korelasi Awal

Untuk memahami hubungan antarfitur serta hubungan fitur terhadap label target, dilakukan perhitungan koefisien korelasi Pearson yang divisualisasikan menggunakan *correlation heatmap* sebagaimana ditunjukkan pada Gambar 3.2.



Gambar 3.2 *Correlation Heatmap* dari *dataset*

Berdasarkan Gambar 3.2, sebagian besar fitur menunjukkan korelasi terhadap variabel target pada rentang 0,20–0,32. Selain itu, ditemukan beberapa pasangan fitur yang memiliki korelasi sangat tinggi, seperti ip5 dan ipk5 (r = 0,99), ip4 dan ipk4 (r = 0,81), serta ip_min_1_6 dan ip_gap_1_6 (r = -0,95), yang mengindikasikan adanya redundansi informasi antar fitur. Hal ini dapat menjadi landasan penentuan *threshold* awal untuk pemilihan fitur.

3.7.2 Penentuan *Threshold* Korelasi Awal

Berdasarkan distribusi nilai korelasi pada *dataset*, digunakan dua *threshold* korelasi yang berbeda sesuai tujuan seleksi fitur sebagai berikut:

a. *Threshold* fitur-target digunakan untuk mengevaluasi relevansi fitur terhadap variabel target. Nilai $|r| \geq 0{,}30$ dipilih secara empiris berdasarkan distribusi korelasi pada *dataset*. Berdasarkan hasil analisis *heatmap*, fitur-fitur dengan korelasi relatif paling tinggi terhadap target terkonsentrasi pada rentang tersebut, sehingga *threshold* ini digunakan untuk mempertahankan fitur yang masih memiliki hubungan yang cukup berarti terhadap variabel target. Dengan demikian, fitur yang memiliki korelasi sangat rendah seperti $|r| < 0{,}10$ terhadap target dapat dieliminasi tanpa menghilangkan terlalu banyak informasi yang berpotensi relevan.

b. *Threshold* antar fitur digunakan untuk mendeteksi redundansi antar fitur. Berdasarkan hasil analisis *heatmap*, sebagian besar pasangan fitur memiliki korelasi di bawah 0,80. sedangkan pasangan fitur yang melebihi nilai tersebut menunjukkan hubungan yang sangat kuat dan cenderung merepresentasikan informasi yang serupa. Oleh karena itu, nilai $|r| > 0{,}80$ dipilih sebagai batas untuk mengidentifikasi dan menghilangkan fitur yang redundan.

3.7.3 Mekanisme *Feature Selection*

Mekanisme penyaringan diterapkan secara kustom melalui arsitektur filter bertingkat yang diintegrasikan ke dalam sebuah komponen *transformer* mandiri. Langkah penyaringan tersebut dijabarkan sebagai berikut yang dieksekusi melalui dua tahapan filter utama, yaitu:

1. Penyaringan Relevansi Target

   Sistem menghitung nilai korelasi Pearson absolut antara seluruh variabel prediktor terhadap label target. Atribut yang dipertahankan wajib memenuhi kriteria batas acuan awal yaitu $|r| \geq 0{,}30$. Jika tidak terdapat fitur yang memenuhi batas acuan awal, sistem menerapkan mekanisme penurunan batas acuan dengan menurunkan batas minimum korelasi menjadi $|r| \geq 0{,}10$, sedangkan atribut dengan korelasi murni di bawah $|r| < 0{,}10$ akan tetap dieleminasi secara permanen.

2. Penyaringan Redundansi Antarfitur

   Penyaringan redundansi antarfitur dilakukan dengan menghitung nilai korelasi Pearson absolut antarfitur yang telah lolos pada tahap penyaringan relevansi target. Fitur-fitur kandidat tersebut kemudian dibentuk menjadi matriks korelasi antarfitur. Selanjutnya, jika ditemukan sepasang fitur yang memiliki nilai korelasi antar-sesama sebesar $|r| > 0{,}80$, sistem akan mengeliminasi salah satu fitur secara otomatis. Penghapusan fitur redundan dilakukan secara deterministik berdasarkan urutan kandidat fitur pada matriks korelasi, sehingga mekanisme seleksi fitur dapat dijalankan secara konsisten pada setiap proses pelatihan dan validasi.

Dengan mekanisme tersebut, fitur yang dipertahankan merupakan fitur yang memenuhi dua kriteria, yaitu memiliki hubungan yang cukup kuat terhadap variabel target dan tidak memiliki redundansi yang berlebihan dengan fitur lainnya. Rekap fitur yang terpilih pada setiap model dan nilai korelasi absolut terhadap label disajikan pada Lampiran B.

3.7.4 Implementasi dalam *Pipeline*

Untuk mencegah *data leakage* selama proses validasi silang, *feature selection* diimplementasikan sebagai bagian dari *pipeline*. Dengan demikian, proses *feature selection* hanya dilakukan menggunakan data *training* pada setiap *fold* dan tidak memanfaatkan informasi dari data validasi maupun data pengujian.

3.7.5 Integrasi dengan *Random Search*

Nilai 0,30 untuk korelasi fitur-target dan 0,80 untuk korelasi antar-fitur digunakan sebagai nilai acuan awal yang diperoleh dari analisis distribusi korelasi pada *heatmap*. Namun, untuk meningkatkan fleksibilitas dan performa sistem, kedua nilai tersebut tidak ditetapkan secara permanen dan dikonfigurasikan sebagai *hyperparameter* yang dievaluasi secara otomatis menggunakan *Random Search*.

3.8 *Modeling* dengan *Gradient Boosting*

Pada penelitian ini, pemodelan dilakukan dengan pendekatan *supervised learning*, karena data yang digunakan telah memiliki label kelas berupa status kelulusan tepat waktu dan lulus terlambat. Model yang digunakan terdiri atas tiga algoritma *gradient boosting*, yaitu *Extreme Gradient Boosting* (XGB), *Histogram-Based Gradient Boosting* (HGB), dan *Light Gradient Boosting Machine* (LGBM). Penjelasan konseptual mengenai ketiga algoritma tersebut telah dijabarkan pada Bab 2, sedangkan bagian ini berfokus pada penerapan model dalam alur eksperimen penelitian.

Pemodelan dilakukan menggunakan *pipeline* agar seluruh tahapan transformasi data dapat dijalankan secara terstruktur dan konsisten. *Pipeline* digunakan untuk menggabungkan proses *feature engineering*, *feature selection*, dan *model training* ke dalam satu alur eksperimen. Dengan pendekatan ini, setiap proses transformasi hanya dipelajari dari data latih pada saat pelatihan dan validasi, sehingga dapat mengurangi risiko kebocoran data dari data uji ke dalam proses pembentukan model.

Secara umum, *pipeline* yang digunakan dalam penelitian ini terdiri atas tiga komponen utama, yaitu FeatureEngineer, PearsonCorrelationSelector, dan estimator klasifikasi. Komponen FeatureEngineer berfungsi untuk membentuk fitur turunan dari data IP dan SKS semester 1 hingga semester 6. Komponen PearsonCorrelationSelector digunakan untuk menyaring fitur berdasarkan korelasi terhadap label dan korelasi antar-fitur. Selanjutnya, fitur yang lolos seleksi digunakan sebagai masukan bagi model klasifikasi.

Dengan rancangan tersebut, tahap pemodelan pada penelitian ini tidak hanya berfokus pada pelatihan algoritma klasifikasi, tetapi juga mencakup pengaturan alur transformasi fitur yang digunakan sebelum model melakukan prediksi. Seluruh model dijalankan dan dievaluasi dalam skema eksperimen yang sama agar perbandingan performa antar-algoritma dapat dilakukan secara adil.

3.9 Pemilihan *Hyperparameter*

Proses pemilihan *hyperparameter* dilakukan menggunakan *Random Search* dengan skema 5-*fold cross validation*. Strategi ini dipilih karena model *gradient boosting* memiliki jumlah *hyperparameter* yang cukup banyak, sehingga pencarian menggunakan seluruh kombinasi parameter dapat membutuhkan biaya komputasi yang besar. Oleh karena itu, ruang pencarian *hyperparameter* dibatasi berdasarkan praktik umum pada literatur optimasi *hyperparameter* dan karakteristik model *gradient boosting*.

Pada penelitian ini, proses *tuning* dilakukan terhadap keseluruhan *pipeline* pemodelan, bukan hanya terhadap *hyperparameter* pada model klasifikasi. Dengan demikian, ruang pencarian *hyperparameter* juga mencakup parameter pada tahap *feature engineering* dan *feature selection* yang sebelumnya telah dijelaskan pada subbab terkait. Parameter pada tahap *feature engineering* mencakup opsi penggunaan atau penghapusan fitur awal melalui drop_raw, sedangkan parameter pada tahap *feature selection* mencakup batas minimum korelasi fitur terhadap label (min_label_r) dan batas maksimum korelasi antarfitur (max_inter_feature_r). Ketiga parameter tersebut disertakan dalam proses *tuning* karena keputusan penggunaan fitur awal dan ambang seleksi fitur dapat memengaruhi komposisi fitur yang digunakan oleh model.

Selain itu, *hyperparameter* model yang dioptimalkan mencakup parameter yang mengatur laju pembelajaran, jumlah iterasi *boosting*, kompleksitas pohon, regularisasi, serta mekanisme sampling data dan fitur. Pendekatan ini digunakan agar proses *tuning* tidak bergantung pada satu konfigurasi awal, tetapi tetap terkontrol secara komputasi sesuai dengan ukuran *dataset* yang digunakan dalam penelitian ini. Rangkuman terkait ruang pencarian *hyperparameter* dapat dilihat pada Tabel 3.4.

Tabel 3.4 Ruang Pencarian *Hyperparameter*

Komponen Pipeline	**Hyperparameter**	**Ruang pencarian**	**Fungsi**
Feature Engineering	fe__drop_raw	[True, False]	Menentukan apakah fitur awal tetap digunakan atau dihapus setelah fitur turunan dibuat.
Feature Selection	fs__min_label_r	[0.1, 0.2, 0.3]	Menentukan batas minimum korelasi fitur terhadap label.
Feature Selection	fs__max_inter_feature_r	[0.75, 0.8, 0.85]	Menentukan batas maksimum korelasi antarfitur.
Model XGB	clf__learning_rate	uniform(0.01, 0.25)	Mengatur laju pembelajaran setiap iterasi *boosting*.
	clf__n_estimators	randint(30, 150)	Menentukan jumlah estimator/pohon *boosting*.
	clf__max_depth	randint(2, 6)	Mengatur kedalaman maksimum pohon.
	clf__min_child_weight	randint(1, 10)	Mengatur batas minimum bobot *instance* pada child node.
	clf__subsample	uniform(0.5, 0.5)	Mengatur proporsi sampel data yang digunakan tiap iterasi.
	clf__colsample_bytree	uniform(0.4, 0.6)	Mengatur proporsi fitur yang digunakan tiap pohon.
	clf__reg_alpha	uniform(0.0, 2.0)	Mengatur regularisasi L1.
	clf__reg_lambda	uniform(0.5, 5.0)	Mengatur regularisasi L2.
Model HGB	clf__learning_rate	uniform(0.01, 0.25)	Mengatur laju pembelajaran setiap iterasi *boosting*.
	clf__max_iter	randint(30, 150)	Menentukan jumlah iterasi *boosting*.
	clf__max_depth	randint(2, 6)	Mengatur kedalaman maksimum pohon.
	clf__max_leaf_nodes	randint(4, 16)	Menentukan jumlah maksimum daun pada pohon.
	clf__min_samples_leaf	randint(5, 30)	Menentukan jumlah minimum data pada daun.
	clf__l2_regularization	uniform(0.5, 5.0)	Mengatur regularisasi L2.

Model LGBM	clf__learning_rate	uniform(0.01, 0.25)	Mengatur laju pembelajaran setiap iterasi *boosting*.
	clf__n_estimators	randint(30, 150)	Menentukan jumlah estimator/pohon *boosting*.
	clf__max_depth	randint(2, 6)	Mengatur kedalaman maksimum pohon.
	clf__num_leaves	randint(4, 16)	Menentukan jumlah maksimum daun pada pohon.
	clf__min_data_in_leaf	randint(5, 30)	Menentukan jumlah minimum data pada daun.
	clf__subsample	uniform(0.5, 0.5)	Mengatur proporsi sampel data yang digunakan tiap iterasi.
	clf__colsample_bytree	uniform(0.4, 0.6)	Mengatur proporsi fitur yang digunakan tiap pohon.
	clf__reg_alpha	uniform(0.0, 2.0)	Mengatur regularisasi L1.
	clf__reg_lambda	uniform(0.5, 5.0)	Mengatur regularisasi L2.

Ruang pencarian pada Tabel 3.4 disusun untuk memberikan variasi konfigurasi pada setiap tahap *pipeline*, tetapi tetap mempertimbangkan efisiensi komputasi dan ukuran *dataset* yang digunakan. Parameter yang berkaitan dengan kompleksitas pohon dibatasi pada rentang rendah hingga menengah untuk mengurangi risiko *overfitting*. Sementara itu, parameter sampling dan regularisasi digunakan untuk membantu model menghasilkan performa yang lebih stabil pada data validasi. Dengan pembatasan tersebut, proses *Random Search* tetap dapat mengeksplorasi kombinasi *hyperparameter* yang beragam tanpa memperluas ruang pencarian secara berlebihan. Hasil konfigurasi terbaik dan nilai validasi silang terbaik dari proses *Random Search* disajikan pada Lampiran C, sedangkan pembahasan hasil utamanya dijelaskan pada Bab 4.

Pemilihan kombinasi *hyperparameter* terbaik pada proses optimasi dilakukan berdasarkan nilai *macro F1-score*. Metrik ini dipilih karena penelitian ini berfokus pada kemampuan model dalam mengenali kedua kelas secara seimbang, yaitu kelas mahasiswa yang lulus tepat waktu dan kelas mahasiswa yang lulus terlambat. Berbeda dengan akurasi yang dapat dipengaruhi oleh dominasi salah satu kelas, *macro F1-score* menghitung rata-rata *F1-score* dari setiap kelas dengan bobot yang sama. Dengan demikian, model tidak hanya

diarahkan untuk memiliki performa tinggi pada satu kelas tertentu, tetapi juga menjaga keseimbangan performa prediksi pada kedua kelas.

Proses pemilihan *hyperparameter* dilakukan hanya pada data latih untuk mencegah terjadinya *data leakage*. Dalam proses tersebut, data latih dibagi ke dalam lima *fold* secara terstratifikasi agar proporsi kelas pada setiap *fold* tetap seimbang. Setelah kombinasi *hyperparameter* terbaik diperoleh dari proses validasi silang, model kemudian dilatih ulang menggunakan seluruh data latih. Model hasil pelatihan ulang tersebut selanjutnya digunakan untuk melakukan evaluasi awal dan evaluasi akhir pada data uji.

3.10 Evaluasi Model

Tahap evaluasi dalam penelitian ini bertujuan untuk menetapkan kerangka penilaian kinerja model *gradient boosting* yang diusulkan. Evaluasi dirancang untuk menilai kualitas prediksi ketepatan waktu kelulusan, kemampuan generalisasi model, dan kontribusi fitur dengan mengacu pada landasan teori evaluasi yang telah dijabarkan pada Subbab 2.8. Evaluasi dilakukan melalui dua skenario, yaitu evaluasi utama pada data gabungan dan evaluasi tambahan pada data berdasarkan angkatan. Evaluasi pada data gabungan digunakan untuk membandingkan kinerja model dan menentukan model terbaik, sedangkan evaluasi pada data berdasarkan angkatan digunakan untuk menilai konsistensi kinerja model terbaik ketika diuji pada kelompok angkatan yang berbeda. Pemisahan kedua skenario tersebut dilakukan karena masing-masing memiliki tujuan evaluasi yang berbeda.

Kerangka evaluasi ini digunakan sebagai acuan dalam pelaporan dan analisis hasil eksperimen pada Bab 4. Rekap numerik hasil evaluasi pada skenario data gabungan, *confusion matrix* dalam bentuk tabel, *threshold* terpilih, dan nilai AUC-ROC setiap model disajikan pada Lampiran D. Sementara itu, hasil evaluasi pada skenario data berdasarkan angkatan disajikan secara terpisah pada Lampiran E**.** Visualisasi tambahan berupa *confusion matrix heatmap*, kurva ROC, dan *permutation feature importance* disajikan pada Lampiran F.

3.10.1 Evaluasi Utama pada Data Gabungan

Evaluasi pada skenario data gabungan merupakan evaluasi utama yang digunakan untuk membandingkan kinerja ketiga model *gradient boosting* dan menentukan model dengan kinerja terbaik. Evaluasi dilakukan secara bertahap melalui evaluasi kinerja *train* dan *cross-validation*, evaluasi awal menggunakan *threshold* 0,5, optimasi *threshold*, evaluasi akhir menggunakan *threshold* terpilih, serta evaluasi menggunakan AUC-ROC. Setelah model terbaik ditentukan,

permutation feature importance digunakan untuk menganalisis kontribusi fitur terhadap kinerja model. Tahapan tersebut dijelaskan lebih lanjut pada beberapa cucu subbab berikut.

Evaluasi Kinerja *Train* dan *Cross-validation*

Evaluasi kinerja *train* dan *cross-validation* dilakukan untuk meninjau kemampuan model dalam mempertahankan performa pada data yang tidak digunakan secara langsung dalam proses pelatihan, sebagaimana yang telah dijelaskan pada Subbab 2.8.3. Pada tahap ini, model terbaik hasil pemilihan *hyperparameter* dianalisis menggunakan nilai *mean train score*, *mean cross-validation score,* dan *gap* antara keduanya. Ketiga nilai tersebut diperoleh dari hasil validasi silang pada proses *Random Search*.

Dalam penelitian ini, evaluasi kinerja *train* dan *cross-validation* tidak digunakan sebagai dasar tunggal pemilihan model, melainkan sebagai analisis tambahan setelah proses pemilihan *hyperparameter*. Model terbaik tetap dipilih berdasarkan nilai *macro F1-score* pada validasi silang, sedangkan perbandingan antara *mean train score*, *mean cross-validation score,* dan *gap* keduanya digunakan untuk menilai apakah kemampuan generalisasi model masih dalam batas wajar atau tidak.

Evaluasi Kinerja Klasifikasi

Evaluasi kinerja klasifikasi dilakukan pada data uji yang terpisah dari data latih. Data uji tidak digunakan dalam proses pelatihan, pemilihan *hyperparameter*, maupun optimasi *threshold,* sehingga hasil evaluasi dapat digunakan untuk menilai performa model terhadap data yang belum pernah dilihat sebelumnya.

Evaluasi dilakukan menggunakan *confusion matrix* sebagai dasar pengukuran, sebagaimana dijelaskan pada Subbab 2.8.1. Dalam penelitian ini, *confusion matrix* digunakan untuk merepresentasikan kemampuan model dalam membedakan mahasiswa yang lulus tepat waktu dan mahasiswa yang lulus terlambat. Berdasarkan *confusion matrix*, kinerja model dievaluasi menggunakan metrik berikut.

a. *Accuracy*, untuk mengukur tingkat ketepatan prediksi secara keseluruhan.
b. *Precision*, untuk mengevaluasi keandalan model ketika memprediksi mahasiswa sebagai lulus tepat waktu.
c. *Recall*, untuk mengukur kemampuan model dalam mengenali mahasiswa yang benar-benar lulus tepat waktu.
d. *F1-score*, sebagai ukuran keseimbangan antara *precision* dan *recall*.

e. *Macro F1-score*, sebagai metrik utama untuk menilai keseimbangan performa model pada kelas lulus tepat waktu dan lulus terlambat.

Pada penelitian ini, *macro F1-score* digunakan sebagai metrik utama karena model diharapkan mampu mengenali kedua kelas secara seimbang. Metrik ini digunakan secara konsisten dalam proses pemilihan *hyperparameter*, evaluasi awal, optimasi *threshold*, dan evaluasi akhir.

Optimasi *Threshold*

Sebelum dilakukan penyesuaian *threshold*, model terlebih dahulu dievaluasi menggunakan *threshold* bawaan sebesar 0,50. Evaluasi awal ini digunakan sebagai dasar untuk melihat performa model setelah proses pemilihan *hyperparameter*. Pada tahap ini, probabilitas prediksi kelas positif dikonversi menjadi label kelas dengan aturan bahwa mahasiswa diprediksi lulus tepat waktu apabila nilai probabilitasnya lebih besar atau sama dengan 0,50.

Setelah evaluasi awal dilakukan, penelitian ini melakukan optimasi *threshold* menggunakan probabilitas *out-of-fold* pada data latih, sebagaimana yang telah dijabarkan pada Subbab 2.8.4. Probabilitas *out-of-fold* diperoleh melalui validasi silang, sehingga setiap data latih diprediksi oleh model yang tidak dilatih menggunakan data tersebut. Pendekatan ini digunakan agar proses pemilihan *threshold* tidak memanfaatkan data uji.

Dalam proses optimasi *threshold*, model terbaik hasil *Random Search* dikloning terlebih dahulu, kemudian diisi dengan kombinasi *hyperparameter* terbaik yang diperoleh dari tahap *tuning*. *Pipeline* hasil kloning tersebut digunakan dalam proses fungsi prediksi silang untuk menghasilkan probabilitas prediksi *out-of-fold* pada data latih. Pendekatan ini memastikan bahwa proses *feature engineering* dan *feature selection* dihitung ulang secara terpisah pada setiap *fold*, sehingga tahapan *preprocessing* tetap berada di dalam skema validasi silang dan tidak menggunakan informasi dari *fold* validasi.

Threshold kandidat ditentukan pada rentang 0,00 sampai 0,99 dengan interval 0,01. Untuk setiap kandidat *threshold*, probabilitas prediksi kelas 1 dibandingkan dengan nilai *threshold* tersebut. Apabila probabilitas lebih besar atau sama dengan *threshold*, maka *instance* diklasifikasikan sebagai kelas 1. Sebaliknya, apabila probabilitas lebih kecil dari *threshold*, maka *instance* diklasifikasikan sebagai kelas 0.

Threshold terbaik dipilih berdasarkan nilai *macro F1-score* tertinggi. Apabila terdapat lebih dari satu *threshold* yang menghasilkan nilai *macro F1-score* yang sama, maka dipilih *threshold* dengan jumlah *false positive* paling kecil. Pertimbangan ini dilakukan karena *false*

positive menunjukkan kondisi ketika mahasiswa yang sebenarnya lulus terlambat diprediksi sebagai lulus tepat waktu. Tingginya nilai *false positive* berisiko membuat program studi luput dalam memberikan intervensi dini, sehingga berdampak langsung pada penurunan rasio kelulusan tepat waktu yang krusial bagi penilaian akreditasi institusi. Setelah *threshold* terbaik diperoleh, model dievaluasi kembali pada data uji menggunakan *threshold* tersebut sebagai evaluasi akhir.

Analisis AUC-ROC

Selain evaluasi berbasis *threshold*, penelitian ini juga menghitung nilai AUC-ROC pada model terbaik. Hal ini karena *macro F1-score*, *precision*, *recall*, dan *confusion matrix* bergantung pada *threshold* tertentu, sedangkan AUC-ROC mengevaluasi kemampuan model pada berbagai nilai *threshold*. Pada tahap ini, kurva ROC divisualisasikan dan nilai AUC dihitung berdasarkan probabilitas prediksi kelas positif pada data uji, sebagaimana telah dijabarkan pada Subbab 2.8.5. Dengan demikian, AUC-ROC digunakan sebagai tambahan untuk melengkapi evaluasi utama berbasis *macro F1-score*.

Analisis *Feature Importance*

Setelah model terbaik diperoleh dan dievaluasi, penelitian ini melakukan analisis kontribusi fitur menggunakan *permutation feature importance*. Analisis ini bertujuan untuk mengetahui fitur mana yang memiliki kontribusi relatif paling besar terhadap performa prediksi model. Sebagaimana yang telah dijelaskan pada Subbab 2.8.6, *permutation feature importance* dilakukan dengan mengacak nilai setiap fitur secara bergantian, kemudian mengamati perubahan performa model setelah fitur tersebut diacak. Fitur yang menyebabkan penurunan performa lebih besar dianggap memiliki kontribusi yang lebih tinggi terhadap prediksi model. Dalam penelitian ini, hasil *permutation feature importance* digunakan sebagai analisis tambahan untuk memahami perilaku model, bukan untuk menyimpulkan hubungan kausal antara fitur akademik dan ketepatan waktu kelulusan mahasiswa.

3.10.2 Evaluasi Tambahan pada Data Berdasarkan Angkatan

Evaluasi pada skenario data berdasarkan angkatan dilakukan sebagai evaluasi tambahan terhadap model terbaik yang diperoleh dari skenario data gabungan. Berbeda dengan evaluasi utama yang digunakan untuk membandingkan dan menentukan model terbaik, evaluasi ini bertujuan untuk menilai konsistensi kinerja model ketika diuji pada kelompok angkatan yang

berbeda. Hasil evaluasi setiap kelompok kemudian dianalisis untuk melihat variasi kinerja model antarangkatan. Prosedur evaluasi dan analisis konsistensi kinerja model dijelaskan pada dua cucu subbab berikut.

Prosedur Evaluasi

Evaluasi dilakukan menggunakan pendekatan *Leave-One-Group-Out* (LOGO), dengan angkatan mahasiswa sebagai kelompok (*group*). Pendekatan LOGO membentuk iterasi evaluasi dengan menggunakan satu kelompok sebagai data uji dan kelompok lainnya sebagai data latih. Dalam penelitian ini, proses pembentukan data latih dan data uji pada setiap iterasi diimplementasikan menggunakan kelas LeaveOneGroupOut yang tersedia pada pustaka *scikit-learn* (Pedregosa et al., 2011). Metrik evaluasi yang digunakan mengacu pada metrik klasifikasi yang telah digunakan pada skenario data gabungan, dengan *macro F1-score* sebagai metrik utama. Penggunaan metrik yang sama bertujuan agar kinerja model pada masing-masing angkatan dapat dibandingkan dengan hasil evaluasi pada skenario data gabungan.

Analisis Konsistensi Kinerja Model pada Kasus Perbedaan Distribusi Data

Pada tahap ini, konsistensi kinerja model dianalisis berdasarkan nilai *macro F1-score* yang diperoleh pada setiap *fold* dalam skenario *Leave-One-Group-Out* (LOGO). Nilai *macro F1-score* dari setiap *fold* dibandingkan untuk mengidentifikasi perbedaan kinerja model ketika diuji pada angkatan yang berbeda. Selanjutnya, nilai rata-rata dan standar deviasi *macro F1-score* dari seluruh *fold* dihitung untuk memberikan gambaran mengenai kinerja model secara keseluruhan serta tingkat variasi kinerja antarangkatan. Nilai rata-rata digunakan sebagai ringkasan performa model pada skenario data berdasarkan angkatan, sedangkan standar deviasi digunakan untuk menggambarkan konsistensi kinerja model antar-*fold*. Semakin kecil standar deviasi, semakin konsisten pula kinerja model pada angkatan yang berbeda, sedangkan standar deviasi yang lebih besar menunjukkan hal sebaliknya.

Selanjutnya, rata-rata *macro F1-score* pada skenario data berdasarkan angkatan dibandingkan dengan *macro F1-score* model XGB pada skenario data gabungan. Perbandingan ini dilakukan untuk mengidentifikasi perubahan performa antara dua skenario tersebut. Selisih nilai *macro F1-score* antara kedua skenario digunakan untuk menggambarkan besarnya perubahan performa model ketika evaluasi dilakukan pada kelompok angkatan yang berbeda. Hasil perbandingan tersebut selanjutnya dianalisis pada Bab 4 untuk mengidentifikasi indikasi faktor-faktor yang berkaitan dengan perbedaan kinerja model pada kedua skenario evaluasi.

BAB IV
HASIL DAN PEMBAHASAN

4.1 Hasil *Data Collection and Labeling*

Seperti yang telah dijabarkan pada Subbab 3.3, tahap *data collection and labeling* menghasilkan data akademik mahasiswa Program Studi Informatika Universitas Islam Indonesia yang digunakan sebagai dasar dalam penelitian ini. Data yang diperoleh berasal dari tiga angkatan, yaitu angkatan 2019, 2020, dan 2021. Setiap data memuat riwayat akademik mahasiswa yang kemudian dibatasi hanya sampai semester enam. Pemangkasan atribut akademik hingga semester enam dan proses pelabelan awal dilakukan menggunakan Microsoft Excel sebelum data diproses lebih lanjut pada lingkungan Python. Pembatasan ini dilakukan karena penelitian berfokus pada prediksi awal ketepatan waktu kelulusan, sehingga informasi akademik setelah semester enam tidak digunakan dalam proses pemodelan.

Pada tahap pelabelan, data mahasiswa diklasifikasikan ke dalam lima kategori berdasarkan status akademik dan ketepatan waktu kelulusannya. Label 0 merepresentasikan mahasiswa yang lulus terlambat, label 1 merepresentasikan mahasiswa yang lulus tepat waktu, label 2 merepresentasikan mahasiswa *drop out*, label 3 merepresentasikan data anomali, dan label 4 merepresentasikan mahasiswa yang belum lulus atau belum memiliki status kelulusan akhir. Detail persebaran label yang diperoleh dari masing-masing angkatan disajikan pada Tabel 4.1.

Tabel 4.1 Persebaran Label Data Awal Berdasarkan Angkatan

Angkatan	Label 0	Label 1	Label 2	Label 3	Label 4	Jumlah Data
2019	86	57	38	3	6	190
2020	87	47	38	1	25	198
2021	12	81	32	1	53	179
Gabungan	185	185	108	5	84	567

Berdasarkan Tabel 4.1, data awal penelitian terdiri atas 567 data mahasiswa yang berasal dari tiga angkatan. Pada data gabungan, label 0 dan label 1 memiliki jumlah yang seimbang, yaitu masing-masing 185 data. Keseimbangan ini penting karena dapat mengurangi kecenderungan model untuk mempelajari salah satu kelas secara lebih dominan. Namun, distribusi label pada masing-masing angkatan menunjukkan pola yang berbeda. Angkatan 2019 dan 2020 memiliki jumlah label 0 yang lebih tinggi dibandingkan label 1, sedangkan angkatan 2021 memiliki jumlah label 1 yang lebih tinggi dibandingkan label 0. Selain itu, tiga label

sisanya ditampilkan pada tahap ini karena tabel tersebut merupakan hasil pelabelan sebelum proses *data cleaning* dilakukan. Dokumentasi aturan *labeling* dan penjelasan kolom yang digunakan dalam proses pelabelan disajikan pada Lampiran A.

Distribusi label pada penelitian ini perlu ditafsirkan secara hati-hati karena label yang digunakan merupakan hasil rekonstruksi berdasarkan aturan *labeling* yang dijelaskan pada Anak Subbab 3.3.2, bukan status kelulusan resmi yang diperoleh langsung dari pihak kampus. Selain itu, penelitian ini belum memasukkan faktor konteks eksternal seperti pandemi COVID-19 dan perubahan kurikulum sebagai variabel eksplisit dalam pemodelan. Padahal, kedua faktor tersebut berpotensi memengaruhi pola pembelajaran, proses akademik, serta penyelesaian studi mahasiswa. Penelitian oleh Aucejo dkk. (2020) menunjukkan bahwa pandemi COVID-19 dapat memengaruhi pengalaman akademik mahasiswa, termasuk keterlambatan kelulusan pada sebagian kelompok mahasiswa. Sementara itu, penelitian oleh Khoirunnisa dkk. (2021) membahas bahwa prokrastinasi dalam pengerjaan skripsi dapat berkaitan dengan keterlambatan waktu kelulusan. Faktor-faktor tersebut tidak tercakup dalam data akademik yang digunakan pada penelitian ini. Oleh karena itu, distribusi label pada penelitian ini tidak dapat diperlakukan sebagai representasi penuh dari kondisi kelulusan aktual mahasiswa, melainkan sebagai label operasional yang dibentuk berdasarkan aturan akademik dan keterbatasan data yang tersedia.

4.2 Hasil *Data Cleaning*

Seperti yang telah dijabarkan pada Subbab 3.4, tahap *data cleaning* dilakukan untuk memastikan bahwa data yang digunakan dalam proses pemodelan sesuai dengan fokus penelitian. Pada tahap ini, data hasil *data collection and labeling* dibersihkan dengan cara menyaring label yang relevan (*label filtering*) dan menghapus beberapa kolom yang tidak digunakan sebagai fitur prediksi (*feature dropping*). Kode yang digunakan untuk menjalankan proses ini disajikan pada Gambar 4.1.

```
# Filter label: Hanya menyertakan status 0 dan 1 (mengecualikan 2, 3, dan 4)
df_clean = df[df["status"].isin([0, 1])].copy()

# Memisahkan Fitur (X) dan Target/Label (y)
# Drop kolom tidak relevan: nim, jml_hadir, jmlh_izin, jml_cuti, dan target status
X = df_clean.drop(columns=['nim',
                           'jml_hadir',
                           'jmlh_izin',
                           'jml_cuti',
                           'status'])
y = df_clean['status']
```

Gambar 4.1 Kode *label filtering* dan *feature dropping* pada *Dataset* Awal

Proses *label filtering* dilakukan dengan hanya mempertahankan data mahasiswa yang memiliki label 0 dan label 1. Label 0 merepresentasikan mahasiswa yang lulus terlambat, sedangkan label 1 merepresentasikan mahasiswa yang lulus tepat waktu. Sementara itu, data dengan label 2, label 3, dan label 4 tidak digunakan karena tidak sesuai dengan fokus penelitian yang diarahkan pada klasifikasi biner ketepatan waktu kelulusan.

Tabel 4.2 Hasil Penyaringan Label pada *Dataset*

Jumlah Data	Label yang Digunakan	Keterangan
567	0,1,2,3,4	Sebelum *label filtering*
370	0,1	Setelah *label filtering*

Setelah proses *label filtering* dilakukan, jumlah data yang digunakan menjadi 370 data mahasiswa. Data tersebut terdiri dari 185 mahasiswa berlabel 0 dan 185 mahasiswa berlabel 1. Dengan demikian, *dataset* yang digunakan pada tahap berikutnya memiliki distribusi kelas yang seimbang. Hasil proses *data cleaning* disajikan pada Tabel 4.3 dan lebih lengkapnya pada Lampiran A.

Tabel 4.3 Distribusi Label Setelah *Data Cleaning*

Label	Keterangan	Jumlah Data
0	Lulus terlambat	185
1	Lulus tepat waktu	185

Proses *feature dropping* dilakukan dengan menghapus kolom yang tidak digunakan sebagai fitur prediksi. Kolom nim dihapus karena hanya berfungsi sebagai identitas mahasiswa. Kolom jml_hadir, jmlh_izin, dan jml_cuti tidak digunakan karena data tersebut bersifat kumulatif dan tidak tersedia secara spesifik dalam batas semester satu hingga semester enam. Sementara itu, kolom status dipisahkan dari fitur karena digunakan sebagai target prediksi.

Selain melakukan *label filtering* dan *feature dropping*, tahap *data cleaning* juga mencakup pengecekan nilai kosong (*missing values*) pada fitur yang digunakan. Pengecekan ini bertujuan untuk mengetahui proporsi *missing values* pada setiap fitur sebelum data digunakan pada tahap pemodelan. Persentase *missing values* dihitung pada setiap kolom fitur menggunakan jumlah *missing values* dibandingkan dengan jumlah keseluruhan data.

Tabel 4.4 Persentase *Missing Values* pada *Dataset*

Fitur	Missing Values(%)
IP1	0.000000
IPK1	0.000000
SKS1	0.000000
IP2	0.270270
IPK2	0.270270
SKS2	0.270270
IP3	0.540541
IPK3	0.540541
SKS3	0.540541
IP4	0.810811
IPK4	0.810811
SKS4	0.810811
IP5	0.270270
IPK5	0.270270
SKS5	0.270270
IP6	0.540541
IPK6	0.540541
SKS6	0.540541

Berdasarkan Tabel 4.4, nilai kosong tertinggi terdapat pada fitur semester empat, yaitu sebesar 0,811%. Nilai tersebut setara dengan tiga data kosong dari total 370 data mahasiswa. Sementara itu, fitur semester dua dan semester lima memiliki nilai kosong sebesar 0,270%, sedangkan fitur semester tiga dan semester enam memiliki nilai kosong sebesar 0,541%. Meskipun proporsi *missing values* tergolong sangat kecil, proses imputasi tidak dilakukan dengan mempertimbangkan kemampuan model *gradient boosting* dalam menangani nilai kosong pada proses pelatihan.

Hasil dari tahap *data cleaning* adalah *dataset* yang terdiri dari 370 data mahasiswa dengan 18 fitur akademik awal, yaitu IP semester 1 sampai semester 6, IPK semester 1 sampai semester 6, dan SKS semester 1 sampai semester 6. *Dataset* ini kemudian dipisahkan menjadi variabel fitur X dan target y, sehingga siap digunakan pada tahap *data splitting*. Untuk menjaga keterlacakan proses ini, rekap pendukung hasil *data cleaning* disajikan pada Lampiran A.

4.3 Hasil *Data Splitting*

Seperti yang telah dijabarkan pada Subbab 3.5, tahap *data splitting* dilakukan setelah proses *data cleaning* dan sebelum *feature engineering*. Pembagian data dilakukan melalui dua skenario, yaitu skenario data gabungan dan skenario data berdasarkan angkatan. Pada skenario

data gabungan, seluruh data dari ketiga angkatan digabungkan terlebih dahulu, kemudian dibagi menjadi data latih dan data uji secara terstratifikasi. Sementara itu, pada skenario data berdasarkan angkatan, pembagian dilakukan dengan mempertahankan kelompok angkatan sehingga setiap angkatan secara bergantian digunakan sebagai data uji dan angkatan lainnya sebagai data latih. Hasil pembagian data pada kedua skenario dijabarkan pada anak subbab berikut. Distribusi label pada kedua skenario juga dirangkum pada Lampiran A sebagai dokumentasi hasil pembagian data.

4.3.1 Hasil pada Data Gabungan

Pada skenario data gabungan, seluruh data mahasiswa angkatan 2019, 2020, dan 2021 yang telah melalui tahap *data cleaning* digabungkan menjadi satu *dataset*. *Dataset* tersebut kemudian dibagi menjadi data latih dan data uji menggunakan fungsi train_test_split dengan parameter test_size=0.2, stratify=y, dan random_state=42. Parameter test_size=0.2 menetapkan proporsi data uji sebesar 20% dari keseluruhan data, sedangkan stratify=y mempertahankan proporsi label pada kedua subset hasil pembagian. Parameter random_state=42 digunakan agar hasil pembagian data dapat direproduksi pada pengujian ulang.

Berdasarkan hasil pembagian, *dataset* yang semula terdiri atas 370 data mahasiswa terbagi menjadi 296 data latih dan 74 data uji. Distribusi kedua kelas pada *dataset* awal yang seimbang tetap dipertahankan melalui pembagian terstratifikasi, sehingga data latih terdiri atas masing-masing 148 data berlabel 0 dan 1, sedangkan data uji terdiri atas masing-masing 37 data berlabel 0 dan 1. Hasil pembagian tersebut ditampilkan pada Tabel 4.5.

Tabel 4.5 Distribusi Label pada Skenario Data Gabungan

Jenis Data	Label 0	Label 1	Jumlah Data
Data latih	148	148	296
Data uji	37	37	74

Penerapan pembagian terstratifikasi memastikan bahwa proporsi kelas pada *dataset* awal tetap dipertahankan pada data latih dan data uji. Dengan demikian, perbedaan distribusi kelas akibat proses pembagian acak dapat diminimalkan. Data latih selanjutnya digunakan dalam proses *feature engineering*, *feature selection*, pemodelan, dan optimasi *hyperparameter*, sedangkan data uji dipisahkan dari proses tersebut dan digunakan pada tahap evaluasi model.

4.3.2 Hasil pada Data Berdasarkan Angkatan

Pada skenario data berdasarkan angkatan, pembagian data dilakukan dengan mempertahankan angkatan mahasiswa sebagai kelompok. Berbeda dengan skenario data gabungan, data dari masing-masing angkatan tidak dibagi secara acak ke dalam data latih dan data uji. Sebaliknya, satu angkatan secara bergantian digunakan sebagai data uji, sedangkan data dari dua angkatan lainnya digabungkan sebagai data latih. Dengan tiga angkatan yang digunakan dalam penelitian ini, proses tersebut menghasilkan tiga konfigurasi pembagian data.

Pada konfigurasi pertama, data angkatan 2019 digunakan sebagai data uji, sedangkan data angkatan 2020 dan 2021 digunakan sebagai data latih. Pada konfigurasi kedua, data angkatan 2020 digunakan sebagai data uji dengan menggabungkan data angkatan 2019 dan 2021 sebagai data latih. Selanjutnya, pada konfigurasi ketiga, data angkatan 2021 digunakan sebagai data uji, sedangkan data angkatan 2019 dan 2020 digunakan sebagai data latih. Hasil pembagian data pada setiap konfigurasi ditampilkan pada Tabel 4.6.

Tabel 4.6 Distribusi Label pada Skenario Data Berdasarkan Angkatan

Konfigurasi	Data Latih	Data Uji	Test L0	Test L1	Train L0	Train L1	Jumlah Data
Grup 1	2020 + 2021	2019	86	57	99	128	370
Grup 2	2019 + 2021	2020	87	47	98	138	370
Grup 3	2019 + 2020	2021	12	81	173	104	370

Berbeda dengan skenario data gabungan, distribusi label pada skenario ini mengikuti distribusi alami yang terdapat pada masing-masing angkatan dan tidak diseimbangkan melalui proses stratifikasi. Oleh karena itu, jumlah dan proporsi label pada data latih maupun data uji dapat berbeda pada setiap konfigurasi. Karakteristik pembagian tersebut dipertahankan karena skenario ini digunakan untuk mengevaluasi kinerja model ketika satu kelompok angkatan dikeluarkan sepenuhnya dari data latih.

4.4 Hasil *Feature Engineering*

Seperti yang telah dijabarkan pada Subbab 3.6, tahap *feature engineering* dilakukan untuk membentuk fitur turunan dari data akademik awal. Fitur turunan ini dibangun berdasarkan nilai Indeks Prestasi (IP) dan jumlah Satuan Kredit Semester (SKS) mahasiswa dari semester satu hingga semester enam. Tujuan dari tahap ini adalah menghasilkan representasi tambahan yang dapat menggambarkan pola capaian akademik mahasiswa selama enam semester pertama.

Proses *feature engineering* diimplementasikan dalam bentuk kelas FeatureEngineer agar dapat digunakan sebagai bagian dari *pipeline* pemodelan. Dengan pendekatan ini, pembentukan fitur turunan dapat dilakukan secara otomatis pada setiap tahap pemodelan. Kode implementasi *feature engineering* disajikan pada Gambar 4.2.

```
# Buat kelas untuk melakukan feature engineering
class FeatureEngineer(BaseEstimator, TransformerMixin):
    def __init__(self, drop_raw=False):
        self.drop_raw = drop_raw

    def fit(self, X, y=None):
        return self

    def transform(self, X):
        X = X.copy()

        ip_cols  = ["ip1","ip2","ip3","ip4","ip5","ip6"]
        sks_cols = ["sks1","sks2","sks3","sks4","sks5","sks6"]

        ip = X[ip_cols].to_numpy(dtype=float)

        # Delta
        X["delta_ip12"] = X["ip2"] - X["ip1"]
        X["delta_ip23"] = X["ip3"] - X["ip2"]
        X["delta_ip34"] = X["ip4"] - X["ip3"]
        X["delta_ip45"] = X["ip5"] - X["ip4"]
        X["delta_ip56"] = X["ip6"] - X["ip5"]

        # Statistik
        X["ip_var_1_6"] = np.nanvar(ip, axis=1, ddof=0)
        X["ip_min_1_6"] = np.nanmin(ip, axis=1)
        X["ip_max_1_6"] = np.nanmax(ip, axis=1)
        X["ip_gap_1_6"] = X["ip_max_1_6"] - X["ip_min_1_6"]

        # SKS
        sks = X[sks_cols].to_numpy(dtype=float)
        X["sks_mean_1_6"] = np.nanmean(sks, axis=1)

        if self.drop_raw:
            X = X.drop(columns=ip_cols + sks_cols)

        return X
```

Gambar 4.2 Kode Implementasi *Feature Engineering* dalam Sebuah Kelas

Fitur turunan yang dihasilkan terdiri dari tiga kelompok utama, yaitu fitur perubahan IP antarsemester, fitur statistik IP selama semester satu hingga semester enam, dan fitur rata-rata SKS. Fitur perubahan IP antarsemester digunakan untuk melihat arah perubahan performa akademik mahasiswa dari satu semester ke semester berikutnya. Nilai positif pada kelompok

fitur ini mengindikasikan bahwa mahasiswa mengalami tren peningkatan performa belajar dari waktu ke waktu, sedangkan nilai negatif menunjukkan kondisi sebaliknya.

Sementara itu, fitur statistik IP digunakan untuk menggambarkan varians, nilai terendah, nilai tertinggi, dan rentang capaian IP mahasiswa. Nilai varians yang rendah dapat diartikan sebagai bentuk kestabilan performa belajar mahasiswa selama enam semester. Di sisi lain, nilai rentang capaian mahasiswa yang cukup lebar dapat menunjukkan adanya kesenjangan antara potensi maksimal dengan realisasi pada kondisi nyata. Hal ini dapat terjadi mengingat performa akademik di dunia nyata kerap dipengaruhi oleh berbagai faktor eksternal yang tidak selalu dapat terekam di dalam sistem. Selanjutnya, fitur rata-rata SKS digunakan untuk merepresentasikan konsistensi pengambilan beban studi akademik mahasiswa selama enam semester pertama. Rangkuman hasil *feature engineering* disajikan pada Tabel 4.7.

Tabel 4.7 Hasil *Feature Engineering*

Kelompok Fitur	**Fitur turunan**	**Keterangan**
Perubahan IP antarsemester	delta_ip12	Selisih IP semester 2 dan semester 1
	delta_ip23	Selisih IP semester 3 dan semester 2
	delta_ip34	Selisih IP semester 4 dan semester 3
	delta_ip45	Selisih IP semester 5 dan semester 4
	delta_ip56	Selisih IP semester 6 dan semester 5
Statistik IP	ip_var_1_6	Varians IP dari semester 1 sampai 6
	ip_min_1_6	Minimum IP dari semester 1 sampai 6
	ip_max_1_6	Maksimum IP dari semester 1 sampai 6
	ip_gap_1_6	Selisih antara nilai Maksimum IP dengan Minimum IP
Rata-rata SKS	sks_mean_1_6	Rerata SKS per semester selama 6 semester

Pada tahap ini, fitur akademik awal tetap dipertahankan karena parameter drop_raw bernilai *False* secara *default*. Dengan demikian, proses *feature engineering* menghasilkan 28 fitur yang terdiri dari 18 fitur akademik awal dan 10 fitur turunan. Fitur-fitur tersebut kemudian digunakan pada tahap *feature selection* untuk memilih fitur yang paling relevan terhadap target prediksi. Daftar fitur turunan beserta keterangan ringkasnya juga disajikan pada Lampiran B.

4.5 Hasil *Feature Selection*

Untuk memilih fitur yang paling relevan terhadap target prediksi dan mengurangi fitur yang memiliki hubungan terlalu tinggi dengan fitur lain, proses *feature selection* dilakukan menggunakan pendekatan *filter method* berbasis korelasi Pearson. Metode ini dipilih karena

fitur yang digunakan pada penelitian ini berbentuk numerik, sehingga hubungan linear antarfitur maupun hubungan fitur terhadap label dapat dihitung menggunakan nilai korelasi.

Proses *feature selection* diimplementasikan dalam bentuk kelas PearsonCorrelationSelector agar dapat digabungkan ke dalam *pipeline* pemodelan. Seleksi fitur dilakukan berdasarkan dua parameter utama, yaitu min_label_r dan max_inter_feature_r. Parameter min_label_r digunakan untuk menentukan batas minimal korelasi absolut antara fitur dan label, sedangkan max_inter_feature_r digunakan untuk menentukan batas maksimal korelasi absolut antarfitur. Dengan demikian, fitur yang dipertahankan adalah fitur yang memiliki hubungan cukup terhadap target, tetapi tidak memiliki korelasi yang terlalu tinggi dengan fitur lain. Kode implementasi kelas *feature selection* disajikan pada Gambar 4.3.

```
# buat kelas untuk melakukan feature selection
class PearsonCorrelationSelector(BaseEstimator, TransformerMixin):
    def __init__(self, min_label_r=0.3, max_inter_feature_r=0.8):
        self.min_label_r = min_label_r
        self.max_inter_feature_r = max_inter_feature_r
        self.selected_features_ = None

    def fit(self, X, y):
        X_df = pd.DataFrame(X).reset_index(drop=True)
        y_series = pd.Series(y).reset_index(drop=True)

        # 1. Filter Hubungan ke Label (Korelasi Pearson absolut)
                correlations_with_target    =    X_df.apply(lambda    col:
np.abs(y_series.corr(col)))

        # Lolos aturan r >= 0.3 (otomatis r < 0.1 langsung gugur di sini)
                                                candidate_features          =
correlations_with_target[correlations_with_target                          >=
self.min_label_r].index.tolist()

        # Antisipasi darurat jika tidak ada fitur yang menyentuh r >= 0.3
        if not candidate_features:
                                                  candidate_features        =
correlations_with_target[correlations_with_target >= 0.1].index.tolist()
        if not candidate_features:
            self.selected_features_ = X_df.columns.tolist()
            return self

        # 2. Filter Hubungan Antar Fitur (Multikolinearitas > 0.8)
        X_candidates = X_df[candidate_features]
        corr_matrix = X_candidates.corr().abs()

        final_features = []
        dropped_features = set()

        for i in range(len(corr_matrix.columns)):
            col_i = corr_matrix.columns[i]
            if col_i in dropped_features:
                continue
            final_features.append(col_i)
```

```
            for j in range(i + 1, len(corr_matrix.columns)):
                col_j = corr_matrix.columns[j]
                if corr_matrix.iloc[i, j] > self.max_inter_feature_r:
                    dropped_features.add(col_j)

        self.selected_features_ = final_features
        return self

    def transform(self, X):
        X_df = pd.DataFrame(X)
        return X_df[self.selected_features_]
```

Gambar 4.3 Kode Implementasi *Feature Selection* dalam Sebuah Kelas

Kandidat nilai min_label_r dan max_inter_feature_r pada awalnya ditentukan berdasarkan hasil analisis korelasi Pearson yang telah dijelaskan pada Anak Subbab 3.7.1. Hasil analisis tersebut kemudian digunakan sebagai dasar untuk menentukan beberapa kemungkinan nilai *threshold* yang dinilai relevan untuk proses seleksi fitur. Selanjutnya, beberapa kandidat nilai *threshold* tersebut dimasukkan ke dalam ruang pencarian *hyperparameter* agar pemilihan *threshold* akhir dapat dilakukan secara empiris berdasarkan performa model pada proses validasi silang. Karena proses *feature selection* berada di dalam *pipeline* pemodelan, fitur yang terpilih ditentukan pada saat *pipeline* dilatih. Dengan demikian, hasil *feature selection* pada penelitian ini ditampilkan berdasarkan estimator terbaik yang diperoleh dari proses pemilihan *hyperparameter*.

Berdasarkan hasil model terbaik pada setiap algoritma yang digunakan, fitur yang terpilih menunjukkan adanya perbedaan. Perbedaan ini dapat terjadi karena proses *feature selection* berada di dalam *pipeline* pemodelan dan parameter seleksi fitur turut dimasukkan ke dalam ruang pencarian *hyperparameter*. Pada model XGB, HGB, dan LGBM, nilai min_label_r yang terpilih sama, yaitu 0,10. Sementara itu, nilai max_inter_feature_r yang terpilih berbeda pada setiap model. Model XGB menggunakan nilai max_inter_feature_r sebesar 0,85, HGB sebesar 0,80, dan LGBM sebesar 0,75. Dengan demikian, LGBM menggunakan batas korelasi antarfitur yang lebih ketat dibandingkan XGB dan HGB. Rangkuman fitur yang dipertahankan oleh model terbaik untuk setiap algoritma disajikan pada Tabel 4.8.

Tabel 4.8 Perbandingan Hasil *Feature Selection* pada Setiap Model

Fitur Terpilih	XGB	HGB	LGBM	Nilai Korelasi dengan Label
sks_mean_1_6	Terpilih	Terpilih	Dibuang	0.320190
ipk1	Dibuang	Terpilih	Dibuang	0.319045
ip1	Terpilih	Dibuang	Terpilih	0.317541
ip_max_1_6	Terpilih	Terpilih	Dibuang	0.309718
ipk6	Terpilih	Dibuang	Dibuang	0.289097

ipk3	Dibuang	Terpilih	Dibuang	0.261286
ip3	Terpilih	Dibuang	Terpilih	0.246584
ip4	Terpilih	Dibuang	Terpilih	0.238611
ipk2	Terpilih	Dibuang	Terpilih	0.233569
sks4	Terpilih	Dibuang	Terpilih	0.224738
sks6	Terpilih	Dibuang	Terpilih	0.220538
ip6	Terpilih	Dibuang	Terpilih	0.208614
delta_ip12	Terpilih	Terpilih	Terpilih	0.200606
ip5	Terpilih	Dibuang	Terpilih	0.198935
delta_ip23	Terpilih	Terpilih	Terpilih	0.171985
sks1	Terpilih	Dibuang	Terpilih	0.164648
sks2	Terpilih	Dibuang	Dibuang	0.161566
ip_var_1_6	Terpilih	Terpilih	Terpilih	0.127252
sks5	Terpilih	Dibuang	Terpilih	0.120044

Berdasarkan Tabel 4.8, terlihat adanya perbedaan preferensi kombinasi fitur yang dipilih oleh ketiga model *gradient boosting*. Perbedaan ini sangat masuk akal secara akademis mengingat adanya perbedaan fundamental pada mekanisme *node splitting* dan arsitektur pertumbuhan pohon (*tree growth*) dari masing-masing algoritma.

Model XGB menunjukkan karakteristik yang cenderung mempertahankan banyak fitur, termasuk fitur dengan korelasi rendah terhadap label (seperti ip_var_1_6 dan sks5). Perilaku ini didorong oleh mekanisme regularisasi L1 dan L2 internal XGBoost yang kuat, yang memungkinkan model mengeksplorasi interaksi nonlinear kompleks pada fitur-fitur berkolerasi rendah tanpa mengorbankan stabilitas varians model.

Sebaliknya, model HGB tampil sebagai model yang paling selektif dengan mengeliminasi mayoritas fitur dan lebih memilih fitur statistik makro seperti sks_mean_1_6 dan ip_max_1_6. Secara akademis, fenomena ini mengindikasikan bahwa metode *binning* histogram pada HGB menilai fitur-fitur statistik global tersebut sudah cukup representatif dalam merangkum informasi performa belajar mahasiswa, sehingga fitur semesteran individual dieksplorasi sebagai bentuk redundansi informasi.

Sementara itu, model LGBM mengambil pendekatan jalan tengah dengan memprioritaskan fitur performa semester individual sembari mempertahankan beberapa fitur turunan. Melalui strategi pertumbuhan pohon secara *leaf-wise*, LGBM mampu mengidentifikasi fluktuasi tajam pada semester tertentu (seperti fitur ip3 dan ip4) sebagai sinyal pemisah yang lebih krusial untuk memprediksi kelulusan tepat waktu, dibandingkan menggunakan nilai rata-rata akumulatif yang berisiko menyamarkan adanya penurunan performa akademik yang ekstrem di dunia nyata.

Kolom nilai korelasi dengan label menunjukkan nilai korelasi linear absolut antara masing-masing fitur dan target prediksi. Nilai ini digunakan sebagai dasar awal dalam proses seleksi fitur, tetapi tidak dapat diartikan sebagai tingkat kepentingan fitur pada model. Tingkat kepentingan fitur pada model dipengaruhi oleh proses pembelajaran algoritma, kombinasi *hyperparameter*, serta interaksi antarfitur. Pembahasan mengenai tingkat kepentingan fitur akan dijabarkan pada Anak Subbab 0. Rekap fitur terpilih pada setiap model, termasuk status fitur yang dipertahankan atau dibuang, disajikan pada Lampiran B.

4.6 Hasil *Modeling* dan Optimasi *Hyperparameter*

Untuk mencari kombinasi parameter terbaik pada setiap model yang digunakan, proses optimasi dilakukan menggunakan *RandomizedSearchCV* dengan skema validasi silang *StratifiedKFold*. Metrik yang digunakan sebagai dasar pemilihan model terbaik adalah *macro F1-score* agar performa model tidak hanya bergantung pada salah satu kelas.

Pada penelitian ini, optimasi *hyperparameter* tidak hanya dilakukan pada parameter algoritma klasifikasi, tetapi juga mencakup parameter pada tahap *feature engineering* dan *feature selection*. Dengan demikian, proses optimasi dilakukan terhadap keseluruhan *pipeline*, bukan hanya terhadap model klasifikasi. Ruang pencarian *hyperparameter* untuk masing-masing model telah dijelaskan pada Subbab 3.9. Oleh karena itu, pada subbab ini ditampilkan salah satu contoh implementasi *pipeline* dan proses optimasi *hyperparameter* pada model XGB yang disajikan pada Gambar 4.4.

```
# buat pipeline
full_pipeline_xgb = Pipeline([
    ('fe', FeatureEngineer()),
    ('fs', PearsonCorrelationSelector()),
    ("clf", XGBClassifier(random_state=42, n_jobs=1))
])

# set up hyperparameter tuning xgb
param_dist = {
    # feature engineering
    # atur keputusan drop fitur awal
    'fe__drop_raw': [True, False],

    # feature selection
    # atur batas korelasi antarfitur
    'fs__max_inter_feature_r': [0.75, 0.8, 0.85],
    # atur batas korelasi fitur dengan label
    'fs__min_label_r': [0.1, 0.2, 0.3],

    # model gradient boosting

    # Kontrol kecepatan dan iterasi
```

```
    "clf__learning_rate": uniform(0.01, 0.25),         # Nilai kontinu antara
0.01 - 0.26
    "clf__n_estimators": randint(30, 150),             # Pohon sedikit karena
data sedikit

    # Kontrol kompleksitas pohon (Kunci Utama XGB)
    "clf__max_depth": randint(2, 6),                    # Maksimal kedalaman
5 agar tidak overfit
     "clf__min_child_weight": randint(1, 10),             # Batasan minimal
bobot data di daun

    # Pengacakan baris dan fitur
    "clf__subsample": uniform(0.5, 0.5),               # Rentang rasio baris
0.5 - 0.5
    "clf__colsample_bytree": uniform(0.4, 0.6),        # Rentang rasio fitur
0.4 - 0.6

    # Regularisasi tambahan
     "clf__reg_alpha": uniform(0.0, 2.0),                 # Regularisasi L1
(Lasso)
     "clf__reg_lambda": uniform(0.5, 5.0),                # Regularisasi L2
(Ridge)

}

# CV yang umum digunakan pada kasus machine learning
cv = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)

search = RandomizedSearchCV(
    estimator=full_pipeline_xgb,
    param_distributions=param_dist,
    n_iter=150,
    scoring="f1_macro",
    cv=cv,
    n_jobs=-1,
    random_state=42,
    verbose=1,
    return_train_score=True
)

# fit modelnya
search.fit(X_train, y_train)

print("Best params:", search.best_params_)
print("Best CV score:", search.best_score_)

# dapatkan model terbaik hasil training
best_model = search.best_estimator_

print("XG Boost training is complete")
```

Gambar 4.4 Kode Implementasi *Modeling* dan *Hyperparameter Tuning*

Struktur implementasi yang sama digunakan pada model HGB dan LGBM, dengan perbedaan pada algoritma klasifikasi serta ruang pencarian *hyperparameter* yang disesuaikan

dengan karakteristik masing-masing model. Hasil konfigurasi *hyperparameter* terbaik pada masing-masing model disajikan pada Tabel 4.9.

Tabel 4.9 Hasil Optimasi *Hyperparameter* pada Setiap Model

Model	Hyperparameter	Nilai Terbaik
XGB	colsample_bytree	0,6829
	learning_rate	0,1130
	max_depth	3
	min_child_weight	1
	n_estimators	98
	reg_alpha	1,6612
	reg_lambda	5,3251
	Subsample	0,5621
	drop_raw	*False*
	max_inter_feature_r	0,85
	min_label_r	0,10
HGB	l2_regularization	1,9102
	learning_rate	0,0544
	max_depth	2
	max_iter	53
	max_leaf_nodes	6
	min_samples_leaf	21
	drop_raw	*True*
	max_inter_feature_r	0,80
	min_label_r	0,10
LGBM	colsample_bytree	0,9489
	learning_rate	0,1006
	max_depth	4
	min_data_in_leaf	15
	n_estimators	51
	num_leaves	13
	reg_alpha	0,0262
	reg_lambda	3,8177
	Subsample	0,5890
	drop_raw	*False*
	max_inter_feature_r	0,75
	min_label_r	0,10

Berdasarkan Tabel 4.9, setiap model menghasilkan konfigurasi *hyperparameter* terbaik yang berbeda. Pada model XGB, konfigurasi terbaik menggunakan max_depth = 3 dan n_estimators = 98, dengan nilai regularisasi reg_alpha = 1,6612 dan reg_lambda = 5,3251. Model HGB menghasilkan konfigurasi yang lebih sederhana dengan max_depth = 2, max_iter = 53, dan max_leaf_nodes = 6. Sementara itu, model LGBM menggunakan max_depth = 4, num_leaves = 13, dan n_estimators = 51.

Perbedaan juga terlihat pada parameter *feature engineering* dan *feature selection*. Model XGB dan LGBM menggunakan drop_raw = *False*, sehingga fitur awal tetap dipertahankan bersama fitur turunan. Sebaliknya, model HGB menggunakan drop_raw = *True*, sehingga fitur IP dan SKS awal yang digunakan dalam pembentukan fitur turunan dihapus setelah proses *feature engineering*. Pada parameter seleksi fitur, ketiga model menggunakan nilai min_label_r = 0,10, sedangkan nilai max_inter_feature_r berbeda. XGB menggunakan nilai 0,85, HGB menggunakan nilai 0,80, dan LGBM menggunakan nilai 0,75. Hal ini menunjukkan bahwa setiap model memiliki kebutuhan konfigurasi seleksi fitur yang berbeda untuk menghasilkan performa validasi terbaik.

Hasil skor validasi silang terbaik dari masing-masing model ditampilkan pada Tabel 4.10.

Tabel 4.10 Hasil *Best CV Score* pada Setiap Model

Model	**Best CV Score (Macro F1-score)**
XGB	0.7282
HGB	0,7291
LGBM	0,7286

Berdasarkan Tabel 4.10, skor validasi silang terbaik dari ketiga model berada pada rentang yang sangat berdekatan. Model HGB memperoleh skor validasi silang tertinggi sebesar 0,7291, diikuti oleh LGBM sebesar 0,7286 dan XGB sebesar 0,7282. Selisih skor yang kecil menunjukkan bahwa ketiga model memiliki kemampuan yang relatif sebanding pada data latih berdasarkan skema validasi silang yang digunakan.

Meskipun HGB memperoleh skor validasi silang tertinggi pada tahap optimasi *hyperparameter*, pemilihan model terbaik tidak hanya ditentukan berdasarkan skor validasi silang. Model tetap perlu dievaluasi lebih lanjut menggunakan data uji yang telah dipisahkan sebelumnya. Selain itu, hasil optimasi ini perlu dipahami sebagai konfigurasi terbaik berdasarkan ruang pencarian *hyperparameter*, jumlah iterasi *RandomizedSearchCV*, dan skema validasi silang yang digunakan dalam penelitian ini, bukan sebagai konfigurasi paling optimal dari seluruh kemungkinan kombinasi *hyperparameter*. Oleh karena itu, hasil optimasi *hyperparameter* ini digunakan sebagai dasar untuk melanjutkan proses evaluasi model pada tahap berikutnya. Ringkasan parameter terbaik, *best CV score*, dan perbandingan *train-CV* setiap model disajikan pada Lampiran C.

4.7 Hasil Evaluasi Model

Tahap evaluasi model dilakukan untuk mengukur performa model dalam memprediksi ketepatan waktu kelulusan mahasiswa. Pada penelitian ini, evaluasi dilakukan melalui dua skenario, yaitu evaluasi utama pada data gabungan dan evaluasi tambahan pada data berdasarkan angkatan. Evaluasi utama pada data gabungan digunakan sebagai dasar utama untuk membandingkan performa model HGB, XGB, dan LGBM, serta menentukan model dengan performa terbaik. Sementara itu, evaluasi tambahan pada data berdasarkan angkatan dilakukan setelah model terbaik diperoleh, dengan tujuan melihat konsistensi performa model ketika diuji pada skenario pembagian data yang berbeda.

Dengan demikian, evaluasi tambahan tidak digunakan sebagai dasar utama pemilihan model, melainkan sebagai analisis pendukung untuk menilai ketahanan model terhadap perbedaan karakteristik antarangkatan. Pembahasan hasil evaluasi utama pada data gabungan disajikan pada Anak Subbab 4.7.1, sedangkan hasil evaluasi tambahan pada data berdasarkan angkatan disajikan pada Anak Subbab 4.7.2.

4.7.1 Hasil Evaluasi Utama pada Data Gabungan

Evaluasi utama pada skenario data gabungan dilakukan terhadap model HGB, XGB, dan LGBM yang telah melalui proses optimasi *hyperparameter*. Evaluasi ini dilakukan secara bertahap, meliputi evaluasi kinerja *train* dan *cross-validation*, evaluasi awal menggunakan *threshold* bawaan 0,50, optimasi *threshold*, evaluasi akhir setelah optimasi *threshold*, analisis AUC-ROC, dan analisis *feature importance*. Tahapan tersebut digunakan untuk menilai performa model dari beberapa aspek, yaitu kestabilan performa model pada validasi internal data gabungan, kemampuan membedakan kedua kelas, keseimbangan prediksi terhadap kelas lulus tepat waktu dan lulus terlambat, serta kontribusi fitur terhadap hasil prediksi.

Hasil evaluasi pada skenario data gabungan digunakan sebagai dasar utama dalam pemilihan model terbaik pada penelitian ini. Rekap numerik hasil evaluasi awal, *confusion matrix* dalam bentuk tabel, optimasi *threshold*, evaluasi akhir, nilai AUC-ROC, dan tabel hasil *permutation feature importance* setiap model disajikan pada Lampiran D. Sementara itu, visualisasi tambahan evaluasi dan interpretasi model disajikan pada Lampiran F.

Evaluasi Kinerja *Train* dan *Cross-validation*

Evaluasi kinerja *train* dan *cross-validation* dilakukan untuk melihat kemampuan model dalam mempertahankan performa pada data validasi selama proses validasi silang. Evaluasi ini

dilakukan dengan membandingkan nilai rata-rata skor pada data latih dan rata-rata skor pada data validasi. Pada penelitian ini, skor yang digunakan adalah *macro F1-score* karena penelitian ini berfokus pada kemampuan model dalam mengenali kedua kelas secara seimbang.

Kode yang digunakan untuk menampilkan nilai rata-rata skor data latih, rata-rata skor validasi silang, dan selisih antara keduanya disajikan pada Gambar 4.5.

```
# uji generalisasi
best_idx = search.best_index_

train_score = search.cv_results_["mean_train_score"][best_idx]
val_score = search.cv_results_["mean_test_score"][best_idx]

print(f"Mean train score: {train_score:.4f}")
print(f"Mean CV score   : {val_score:.4f}")
print(f"Gap             : {train_score - val_score:.4f}")
```

Gambar 4.5 Kode Evaluasi Kinerja *Train* dan *Cross-validation*

Nilai *mean train score* menunjukkan rata-rata performa model pada data latih selama proses validasi silang. Sementara itu, *mean CV score* menunjukkan rata-rata performa model pada data validasi. Selisih antara keduanya digunakan untuk melihat indikasi perbedaan performa model antara data latih dan data validasi. Nilai selisih yang besar mengindikasikan kecenderungan generalisasi yang kurang stabil.

Tabel 4.11 Perbandingan *Train* dan *Cross-validation* pada Setiap Model

Model	Mean Train F1 Macro	Mean CV F1 Macro	Gap
XGB	0,8775	0,7282	0,1493
HGB	0,7954	0,7291	0,0663
LGBM	0,8961	0,7286	0,1675

Berdasarkan Tabel 4.11, ketiga model menghasilkan nilai *mean CV score* yang relatif berdekatan. Model HGB memperoleh *mean CV score* tertinggi sebesar 0,7291, diikuti oleh LGBM sebesar 0,7286 dan XGB sebesar 0,7282. Selisih nilai validasi silang antar model sangat kecil, sehingga dari sisi performa validasi, ketiga model menunjukkan kemampuan yang relatif sebanding.

Perbedaan yang lebih terlihat terdapat pada nilai gap antara *mean train score* dan *mean CV score*. Model HGB memiliki gap paling kecil, yaitu 0,0663, sehingga menunjukkan kemampuan generalisasi yang lebih stabil dibandingkan model lainnya. Sementara itu, XGB memiliki gap sebesar 0,1493 dan LGBM memiliki gap sebesar 0,1675. Nilai gap yang lebih

besar pada XGB dan LGBM menunjukkan bahwa kedua model tersebut memiliki kecenderungan lebih tinggi untuk menghafal data latih dibandingkan HGB.

Dengan demikian, hasil evaluasi kinerja *train* dan *cross-validation* menunjukkan bahwa HGB memiliki kestabilan terbaik pada proses validasi silang. Namun, pemilihan model akhir tidak hanya ditentukan berdasarkan hasil generalisasi pada data latih dan validasi. Model tetap perlu dievaluasi lebih lanjut menggunakan data uji untuk melihat performanya terhadap data yang tidak digunakan selama proses pelatihan maupun optimasi *hyperparameter*.

Evaluasi Awal dengan *Threshold* 0,50

Evaluasi awal dilakukan untuk mengukur kinerja model pada data uji menggunakan *threshold* bawaan sebesar 0,50. Pada tahap ini, probabilitas prediksi kelas 1 dikonversi menjadi label kelas dengan aturan bahwa data akan diklasifikasikan sebagai kelas 1 apabila nilai probabilitasnya lebih besar atau sama dengan 0,50. Evaluasi ini digunakan sebagai dasar sebelum dilakukan optimasi *threshold* pada tahap berikutnya.

Kode yang digunakan untuk melakukan evaluasi awal dengan threshold 0,50 disajikan pada Gambar 4.6.

```
# inisiasi data test
X_eval = X_test
y_eval = y_test

# lakukan prediksi
y_proba = best_model.predict_proba(X_eval)[:,1]
y_pred_05 = (y_proba >= 0.5).astype(int)

cm_05 = confusion_matrix(y_eval, y_pred_05)

tn, fp, fn, tp = cm_05.ravel()

# hitung nilai metrik
accuracy_05 = accuracy_score(y_eval, y_pred_05)
precision_05 = precision_score(y_eval, y_pred_05)
recall_05 = recall_score(y_eval, y_pred_05)
f1_05 = f1_score(y_eval, y_pred_05, average="macro", zero_division=0)

print("\n=== Baseline threshold 0.50 ===")

# Print nilai metrik
print(f"Accuracy: {accuracy_05:.4f}")
print(f"Precision: {precision_05:.4f}")
print(f"Recall: {recall_05:.4f}")
print(f"F1-Score: {f1_05:.4f}")

# Print Confusion Matrix
print("Confusion Matrix:\n", cm_05)
print(f"TN={tn}, FP={fp}, FN={fn}, TP={tp}")
```

```
# Print laporan klasifikasi detail
print(classification_report(y_eval, y_pred_05))
```

Gambar 4.6 Kode Evaluasi Awal dengan *Threshold* 0.50

Hasil evaluasi awal pada data uji untuk masing-masing model ditampilkan pada Tabel 4.12.

Tabel 4.12 Hasil Evaluasi Awal dengan *Threshold* 0,50

Model	Accuracy	Precision	Recall	Macro F1-score
XGB	0,7973	0,8056	0,7838	0,7973
HGB	0,7297	0,7179	0,7568	0,7295
LGBM	0,7568	0,7568	0,7568	0,7568

Berdasarkan Tabel 4.12, model XGB memperoleh performa terbaik pada evaluasi awal dengan *threshold* 0,50. Model tersebut menghasilkan nilai *accuracy* sebesar 0,7973, *precision* sebesar 0,8056, *recall* sebesar 0,7838, dan *macro F1-score* sebesar 0,7973. Model LGBM menempati posisi kedua dengan *macro F1-score* sebesar 0,7568, sedangkan model HGB memperoleh *macro F1-score* sebesar 0,7295.

Selain metrik evaluasi, hasil prediksi juga dianalisis menggunakan *confusion matrix* untuk melihat jumlah prediksi benar dan salah pada masing-masing kelas. *Confusion matrix* pada *threshold* 0,50 ditampilkan pada Tabel 4.13.

Tabel 4.13 *Confusion Matrix* Evaluasi Awal dengan *Threshold* 0,50

Model	TP	FP	FN	TN
XGB	29	7	8	30
HGB	28	11	9	26
LGBM	28	9	9	28

Berdasarkan Tabel 4.13, model XGB menghasilkan 30 prediksi benar untuk kelas 0 dan 29 prediksi benar untuk kelas 1. Model ini memiliki 7 *false positive* dan 8 *false negative*. Model HGB menghasilkan 26 *true negative*, 28 *true positive*, 11 *false positive*, dan 9 *false negative*. Sementara itu, model LGBM menghasilkan distribusi kesalahan yang seimbang, yaitu 9 *false positive* dan 9 *false negative*.

Hasil evaluasi awal menunjukkan bahwa penggunaan *threshold* bawaan 0,50 sudah memberikan performa yang cukup baik, terutama pada model XGB. Namun, *threshold* bawaan

belum tentu menghasilkan keseimbangan performa terbaik pada kedua kelas. Oleh karena itu, tahap berikutnya dilakukan optimasi *threshold* untuk mencari batas probabilitas yang lebih sesuai dengan karakteristik model dan data.

Optimasi *Threshold*

Optimasi *threshold* dilakukan untuk menentukan batas probabilitas prediksi yang lebih sesuai dengan karakteristik model dan data. Pada evaluasi awal, *threshold* yang digunakan adalah nilai bawaan 0,50. Namun, penggunaan *threshold* bawaan belum tentu menghasilkan performa terbaik, terutama ketika tujuan evaluasi diarahkan pada keseimbangan performa kedua kelas. Oleh karena itu, penelitian ini melakukan optimasi *threshold* dengan menggunakan *out-of-fold probability* pada data latih.

Proses optimasi *threshold* dilakukan hanya menggunakan data latih agar data uji tetap berfungsi sebagai data evaluasi akhir. Probabilitas *out-of-fold* diperoleh menggunakan cross_val_predict dengan metode predict_proba. Pada proses ini, *pipeline* dengan kombinasi *hyperparameter* terbaik dilatih ulang pada setiap *fold*, sehingga tahap *feature engineering* dan *feature selection* tetap dilakukan secara terpisah di masing-masing *fold* validasi. Dengan demikian, probabilitas yang digunakan untuk memilih *threshold* tidak berasal dari prediksi terhadap data yang ikut digunakan untuk melatih model pada *fold* yang sama.

Kode yang digunakan untuk melakukan optimasi *threshold* berdasarkan probabilitas *out-of-fold* disajikan pada Gambar 4.7.

```
# inisiasi kelas pencari threshold
# - Threshold dipilih dari TRAIN SAJA menggunakan OOF prob
# - Objective: max f1_macro, tie-break: FP terkecil

thresholds = np.arange(0.00, 1.00, 0.01)
def    pick_threshold_from_oof(random_search_object,    X_train,    y_train,
cv_inner, thresholds):

    # 1. Ambil blueprint pipeline awal yang belum dilatih
    # lalu isi dengan kombinasi parameter terbaik hasil tuning
    fresh_pipeline = clone(random_search_object.estimator)
    fresh_pipeline.set_params(**random_search_object.best_params_)

    # 2. Jalankan cross_val_predict menggunakan pipeline yang dinamis ini
    # Sekarang, Feature Engineering dan Feature Selection akan dihitung ulang
secara bersih di setiap fold OOF
    y_proba_oof = cross_val_predict(
        fresh_pipeline, # Menggunakan pipeline kloningan yang dinamis
        X_train,
        y_train,
        cv=cv_inner,
        method="predict_proba",
```

```
        n_jobs=-1
    )[:, 1]             # ambil proba kelas 1

    print("Prediksi Probabilitas OOF Berhasil Dihitung.")
    scan = []
    # buat list hasil f1 score pada tiap threshold
    for t in thresholds:
        y_pred_oof = (y_proba_oof >= t).astype(int)
        # gunakan f1_macro
       f1m = f1_score(y_train, y_pred_oof, average="macro", zero_division=0)
        fp = np.sum((y_train == 0) & (y_pred_oof == 1))
        # masukkan nilai threshold, skor f1 macro, dan false positivenya
        # kedalam array scan
        scan.append((t, f1m, fp))

    # ambil nilai macro f1-score tertinggi dari list
    # lalu masukkan ke dalam sebuah variabel
    best_f1 = max(r[1] for r in scan)
    # ambil list nilai-nilai threshold
    # yang memiliki nilai f1 sama dengan best_f1
    # masukkan ke variabel baru
    candidates = [r for r in scan if r[1] == best_f1]
    # pilih threshold terbaik berdasarkan variabel candidates
    # dan nilai FP terendah
    best_thr = min(candidates, key=lambda x: x[2])[0]
    return best_thr, best_f1

# Kirim objek 'search' hasil tuning
best_thr, best_f1_oof = pick_threshold_from_oof(
        random_search_object=search,
        X_train=X_train,
        y_train=y_train,
        cv_inner=cv,
        thresholds=thresholds
)
print(f"Chosen   threshold   from   TRAIN(OOF):   {best_thr:.2f}   |   (best
f1_macro={best_f1_oof:.4f})")
```

Gambar 4.7 Kode Optimasi *Threshold* Menggunakan *Out-of-Fold Probability*

Threshold dicari pada rentang 0,00 hingga 0,99 dengan interval 0,01. Pada setiap nilai *threshold*, probabilitas prediksi kelas 1 dikonversi menjadi label prediksi, kemudian dihitung nilai *macro F1-score*. *Threshold* terbaik dipilih berdasarkan nilai *macro F1-score* tertinggi. Apabila terdapat lebih dari satu *threshold* dengan nilai *macro F1-score* yang sama, maka *threshold* dengan jumlah *false positive* paling kecil dipilih sebagai *tie-breaker*.

Hasil optimasi *threshold* pada setiap model ditampilkan pada Tabel 4.14.

Tabel 4.14 Hasil Optimasi *Threshold* Menggunakan Data Latih

Model	*Threshold* Terpilih	Best OOF Macro F1-score
XGB	0,52	0,7330
HGB	0,39	0,7300
LGBM	0,51	0,7296

Berdasarkan Tabel 4.14, setiap model menghasilkan *threshold* terbaik yang berbeda. Model XGB memperoleh *threshold* terbaik sebesar 0,52 dengan nilai *macro F1-score* OOF sebesar 0,7330. Model HGB memperoleh *threshold* terbaik sebesar 0,39 dengan nilai *macro F1-score* OOF sebesar 0,7300. Sementara itu, model LGBM memperoleh *threshold* terbaik sebesar 0,51 dengan nilai *macro F1-score* OOF sebesar 0,7296.

Perbedaan *threshold* terpilih menunjukkan bahwa setiap model menghasilkan distribusi probabilitas prediksi yang berbeda. Model HGB membutuhkan *threshold* yang lebih rendah, yaitu 0,39, agar performa klasifikasi pada data latih OOF menjadi optimal. Sementara itu, XGB dan LGBM menghasilkan *threshold* yang relatif dekat dengan nilai bawaan 0,50. Hasil ini menunjukkan bahwa *threshold* bawaan tidak selalu menjadi batas keputusan terbaik untuk setiap model.

Threshold yang diperoleh pada tahap ini selanjutnya digunakan untuk melakukan evaluasi akhir pada data uji. Dengan demikian, data uji tidak digunakan dalam proses pemilihan *threshold*, melainkan hanya digunakan untuk mengukur performa akhir model setelah *threshold* terbaik ditentukan dari data latih.

Evaluasi Akhir setelah Optimasi *Threshold*

Evaluasi akhir dilakukan untuk mengukur kinerja model pada data uji setelah menggunakan *threshold* terbaik yang diperoleh dari proses optimasi *threshold* pada data latih. *Threshold* yang digunakan pada tahap ini adalah *threshold* hasil optimasi *out-of-fold* pada Subbab 0. Dengan demikian, data uji tetap digunakan hanya sebagai data evaluasi akhir dan tidak terlibat dalam proses pemilihan *threshold*. Hasil evaluasi akhir pada data uji setelah menggunakan *threshold* terbaik ditampilkan pada Tabel 4.15.

Tabel 4.15 Hasil Evaluasi Akhir dengan *Threshold* Terbaik

Model	Threshold	Accuracy	Precision	Recall	Macro F1-score
XGB	0,52	0,7973	0,8056	0,7838	0,7973
HGB	0,39	0,7568	0,6939	0,9189	0,7502
LGBM	0,51	0,7568	0,7568	0,7568	0,7568

Berdasarkan Tabel 4.15, model XGB masih memperoleh performa terbaik pada evaluasi akhir dengan nilai *accuracy* sebesar 0,7973, *precision* sebesar 0,8056, *recall* sebesar 0,7838, dan *macro F1-score* final sebesar 0,7973. Model LGBM memperoleh *macro F1-score* final sebesar 0,7568, sedangkan model HGB memperoleh *macro F1-score* final sebesar 0,7502. Hasil ini menunjukkan bahwa setelah optimasi *threshold*, XGB tetap memberikan performa paling tinggi pada data uji dibandingkan model lainnya. *Confusion matrix* hasil evaluasi akhir setelah optimasi *threshold* ditampilkan pada Tabel 4.16.

Tabel 4.16 *Confusion Matrix* Evaluasi Akhir dengan *Threshold* Terbaik

Model	TP	FP	FN	TN
XGB	29	7	8	30
HGB	34	15	3	22
LGBM	28	9	9	28

Berdasarkan Tabel 4.16, model XGB menghasilkan 30 data kelas 0 yang diprediksi benar dan 29 data kelas 1 yang diprediksi benar. Model ini memiliki 7 *false positive* dan 8 *false negative*. Model HGB menghasilkan jumlah *true positive* paling tinggi, yaitu 34 data, tetapi juga memiliki jumlah *false positive* paling tinggi, yaitu 15 data. Hal ini menunjukkan bahwa threshold 0,39 pada HGB membuat model lebih banyak memprediksi kelas 1, sehingga *recall* kelas 1 meningkat, tetapi precision kelas 1 menurun. Sementara itu, model LGBM menghasilkan 28 *true negative*, 28 *true positive*, 9 *false positive*, dan 9 *false negative*, sehingga distribusi kesalahan pada kedua kelas relatif seimbang.

Tabel 4.17 Perubahan Setelah Optimasi *Threshold*

Model	F1 Macro Threshold 0.5	F1 Macro Threshold Terbaik	Perubahan
XGB	0,7973	0,7973	0,0000
HGB	0,7295	0,7502	+0,0207
LGBM	0,7568	0,7568	0,0000

Berdasarkan Tabel 4.17, secara umum proses optimasi *threshold* pada kondisi eksperimen ini tidak memberikan dampak peningkatan skor yang besar pada keseluruhan model. Hasil *macro F1-score* untuk model XGB dan LGBM tidak mengalami perubahan, sementara model HGB mengalami peningkatan minor sebesar 0,0207. Hal ini menunjukkan bahwa dampak praktis dari pergeseran batas acuan (*decision threshold*) pada dataset ini relatif terbatas.

Secara lebih rinci, XGB dan LGBM tidak mengalami perubahan nilai *macro F1-score* setelah optimasi *threshold* karena perubahan *threshold* yang terpilih tidak mengubah distribusi prediksi pada data uji. Sementara itu, peningkatan pada HGB terjadi karena *threshold* diturunkan menjadi 0,39, sehingga model lebih banyak memprediksi data sebagai kelas 1. Perubahan ini meningkatkan *recall* kelas 1, tetapi juga disertai peningkatan *false positive*. Dengan demikian, kenaikan *macro F1-score* pada HGB perlu diinterpretasikan sebagai hasil *trade-off* antara peningkatan kemampuan mengenali kelas 1 dan bertambahnya kesalahan prediksi terhadap kelas 0.

Optimasi *threshold* pada dasarnya merupakan tahap pascapelatihan (*post-processing*) yang tidak mengubah bobot internal model, tetapi mengubah batas probabilitas yang digunakan untuk menentukan kelas prediksi. Oleh karena itu, optimasi ini tidak selalu menghasilkan peningkatan performa yang besar. Meskipun demikian, proses ini tetap penting sebagai tahap evaluatif untuk memastikan bahwa *threshold* yang digunakan telah diperiksa berdasarkan metrik utama penelitian, bukan semata-mata mengikuti nilai bawaan 0,50.

Analisis AUC-ROC

Analisis AUC-ROC dilakukan untuk mengevaluasi kemampuan model dalam membedakan kelas 0 dan kelas 1 berdasarkan probabilitas prediksi. Berbeda dengan metrik seperti *accuracy*, *precision*, *recall*, dan *F1-score* yang bergantung pada *threshold* tertentu, AUC-ROC mengevaluasi performa model pada berbagai kemungkinan *threshold*. Oleh karena itu, metrik ini digunakan sebagai analisis tambahan untuk melihat kemampuan diskriminatif model terhadap kedua kelas. Kode yang digunakan untuk menghitung dan menampilkan kurva ROC disajikan pada Gambar 4.8.

```
# plot roc-auc

# ambil proba dari hasil fitting
fpr, tpr, thresholds = roc_curve(y_eval, y_proba)
auc_score = roc_auc_score(y_eval, y_proba)
```

```
# Plot the ROC curve
plt.figure(figsize=(7, 6))
plt.plot(fpr, tpr, label=f"AUC-ROC = {auc_score:.4f}")
plt.plot([0, 1], [0, 1], linestyle="--", label="Random Classifier")

plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("ROC Curve")
plt.legend(loc="lower right")
plt.grid(True)
plt.show()

print(f"AUC-ROC: {auc_score:.4f}")
```

Gambar 4.8 Kode Analisis AUC-ROC

Pada analisis ini, nilai *False Positive Rate* (FPR) dan *True Positive Rate* (TPR) dihitung berdasarkan probabilitas prediksi kelas 1. Nilai AUC-ROC yang semakin mendekati 1 menunjukkan bahwa model memiliki kemampuan yang semakin baik dalam membedakan mahasiswa yang lulus tepat waktu dan mahasiswa yang lulus terlambat. Sebaliknya, nilai AUC-ROC yang mendekati 0,5 menunjukkan bahwa kemampuan model mendekati klasifikasi acak.

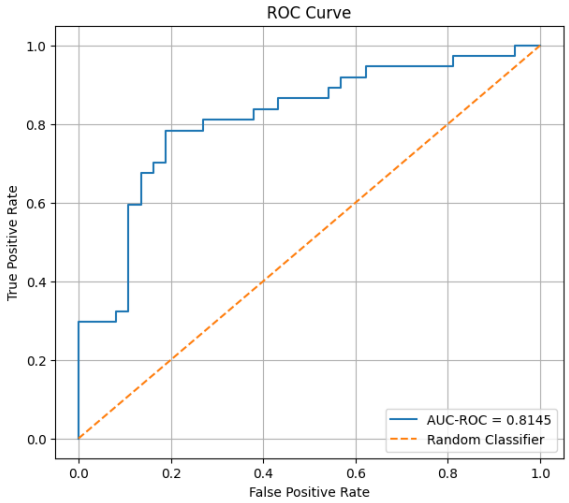
Tabel 4.18 Hasil AUC-ROC pada Setiap Model

Model	AUC-ROC
XGB	0,8145
HGB	0,8108
LGBM	0,7955

Berdasarkan Tabel 4.18, model XGB memperoleh nilai AUC-ROC tertinggi, yaitu sebesar 0,8145. Nilai tersebut menunjukkan bahwa XGB memiliki kemampuan diskriminatif terbaik dibandingkan dua model lainnya dalam membedakan kelas 0 dan kelas 1 berdasarkan probabilitas prediksi. Model HGB memperoleh nilai AUC-ROC sebesar 0,8108, yang hanya sedikit lebih rendah dibandingkan XGB. Sementara itu, model LGBM memperoleh nilai AUC-ROC sebesar 0,7955.

Hasil ini menunjukkan bahwa ketiga model memiliki kemampuan yang cukup baik dalam membedakan kedua kelas, karena seluruh nilai AUC-ROC berada di atas 0,79. Namun, XGB tetap menunjukkan performa paling baik pada analisis ini. Hasil AUC-ROC ini juga sejalan dengan evaluasi akhir pada Subbab 0, di mana XGB memperoleh nilai macro F1-score tertinggi pada data uji.

Karena XGB memperoleh nilai AUC-ROC tertinggi, kurva ROC model XGB ditampilkan pada Gambar 4.9.



Gambar 4.9 Kurva ROC pada Model XGB

Berdasarkan Gambar 4.9, kurva ROC model XGB berada di atas garis diagonal *random classifier*. Hal ini menunjukkan bahwa model memiliki kemampuan klasifikasi yang lebih baik dibandingkan prediksi acak dalam membedakan mahasiswa yang lulus tepat waktu dan lulus terlambat. Kurva ROC untuk seluruh model disajikan pada Lampiran F.

Analisis *Feature Importance*

Analisis *feature importance* dilakukan untuk mengetahui fitur yang paling berpengaruh terhadap performa model pada data uji. Pada penelitian ini, analisis dilakukan menggunakan metode *permutation importance*. Metode ini bekerja dengan cara mengacak nilai pada satu fitur, kemudian mengukur perubahan performa model setelah fitur tersebut diacak. Apabila pengacakan suatu fitur menyebabkan penurunan performa yang besar, maka fitur tersebut dianggap memiliki kontribusi yang lebih tinggi terhadap prediksi model.

```
# permutation importance
# 1. Ekstrak nama fitur final yang lolos seleksi dari komponen 'fs'
fitur_final_terpilih = best_model.named_steps['fs'].selected_features_

# 2. Ambil komponen feature engineering dan feature selection
best_fe = best_model.named_steps['fe']
best_fs = best_model.named_steps['fs']

# 3. Transformasikan X_test mentah melewati FE dan FS
```

```
X_test_engineered = best_fe.transform(X_test)
X_test_final = best_fs.transform(X_test_engineered)

# 4. Ambil classifier murni dari pipeline terbaik
final_classifier = best_model.named_steps['clf']

# 5. Hitung permutation importance pada data uji
result = permutation_importance(
    final_classifier,
    X_test_final,
    y_test,
    n_repeats=10,
    scoring="f1_macro",
    random_state=42,
    n_jobs=-1
)

# 6. Buat tabel importance
features_df = pd.DataFrame({
    'Feature': fitur_final_terpilih,
    'Importance': result.importances_mean
})

# Urutkan berdasarkan nilai kepentingan tertinggi
features_df = features_df.sort_values(by='Importance', ascending=False)

# 6. Cetak tabel dan tampilkan grafik barplot
print("--- TABEL PERMUTATION FEATURE IMPORTANCE PADA DATA UJI ---")
display(features_df)

plt.figure(figsize=(12, 8))
sns.barplot(x='Importance',
     y='Feature',
     data=features_df,
     palette='viridis')
plt.title('Permutation Feature Importance pada Data Uji\n(Berdasarkan Fitur
Terpilih Hasil Seleksi Pearson)', fontsize=14, fontweight='bold')
plt.xlabel('Penurunan F1-score Macro (Importance Mean)')
plt.ylabel('Nama Fitur Final')
plt.tight_layout()
plt.show()
```

Gambar 4.10 Kode Analisis *Feature Importance*

Berdasarkan Gambar 4.10, *permutation importance* dihitung menggunakan data uji dan metrik *macro F1-score*. Fitur yang dianalisis merupakan fitur akhir yang telah melewati tahap *feature engineering* dan *feature selection*. Dengan demikian, nilai *importance* yang diperoleh hanya dihitung terhadap fitur yang benar-benar digunakan oleh model dalam proses prediksi. Hasil *permutation importance* pada setiap model ditampilkan pada Tabel 4.19.

Tabel 4.19 Hasil *Permutation Feature Importance* pada Setiap Model

Model	Fitur dengan *Importance* Tertinggi	Nilai *Importance*
XGB	sks_mean_1_6	0,1046
	sks6	0,0798
	ip5	0,0704
	delta_ip12	0,0541
	ipk6	0,0446
HGB	sks_mean_1_6	0,1312
	delta_ip12	0,0096
	delta_ip23	0,0094
	ip_max_1_6	0,0015
LGBM	sks6	0,1517
	delta_ip12	0,0421
	ip1	0,0412
	delta_ip23	0,0338
	ip5	0,0326

Berdasarkan Tabel 4.19, hasil *permutation feature importance* menunjukkan bahwa terdapat beberapa fitur yang memberikan kontribusi relatif besar terhadap performa prediksi pada masing-masing model. Fitur-fitur tersebut dapat dikaitkan dengan konteks akademik mahasiswa, khususnya dalam menggambarkan pola capaian akademik dan progres studi hingga semester enam.

Fitur rata-rata pengambilan SKS dari semester satu hingga semester enam (sks_mean_1_6) menjadi fitur dengan nilai *importance* tertinggi pada model XGB dan HGB. Secara akademis, hal ini dapat dipahami karena rata-rata SKS mencerminkan konsistensi mahasiswa dalam mengambil beban studi pada setiap semester. Mahasiswa dengan rata-rata pengambilan SKS yang relatif stabil dan tinggi cenderung memiliki progres akademik yang lebih baik karena mampu menempuh beban studi secara berkelanjutan. Namun, fitur ini tidak dapat berdiri sendiri untuk menjelaskan kemampuan mahasiswa dalam menyelesaikan mata kuliah yang diambil, karena jumlah SKS hanya menunjukkan beban studi, bukan capaian akademik yang diperoleh. Oleh karena itu, kontribusi fitur ini perlu diinterpretasikan bersama dengan fitur lain yang merepresentasikan capaian akademik, seperti IP, IPK, delta IP, maupun fitur ringkasan IP. Hal tersebut terlihat dari hasil *importance* pada setiap model, di mana XGB juga menempatkan fitur ip5, ipk6, dan delta_ip12 sebagai fitur penting, sedangkan HGB menempatkan delta_ip12, delta_ip23, dan ip_max_1_6 sebagai fitur dengan kontribusi yang relatif tinggi.

Sementara itu, fitur jumlah SKS pada semester enam (sks6) juga memiliki nilai *importance* yang tinggi, khususnya pada model XGB dan LGBM. Fitur ini dapat dipahami sebagai indikator progres akademik mahasiswa pada fase akhir sebelum memasuki tahun keempat. Pada tahap ini, jumlah SKS yang ditempuh atau berhasil diselesaikan dapat memberikan gambaran mengenai kesiapan mahasiswa untuk melanjutkan ke tahap akhir studi, seperti pengambilan mata kuliah penjaluran, KKN, atau tugas akhir. Apabila jumlah SKS pada semester enam relatif rendah, kondisi tersebut dapat mengindikasikan adanya keterlambatan akumulasi beban studi, pengulangan mata kuliah, atau hambatan akademik lain yang berpotensi memengaruhi ketepatan waktu kelulusan. Dengan demikian, tingginya kontribusi sks6 pada model XGB dan LGBM menunjukkan bahwa informasi beban studi di semester enam membantu model dalam membedakan mahasiswa yang cenderung lulus tepat waktu dan mahasiswa yang berisiko lulus terlambat.

Selain fitur berbasis SKS, fitur perubahan IP dari semester satu ke semester dua (delta_ip12) juga muncul secara konsisten sebagai fitur penting pada ketiga model. Hal ini menunjukkan bahwa perubahan capaian akademik pada fase awal perkuliahan mengandung informasi yang membantu model dalam membedakan pola mahasiswa yang lulus tepat waktu dan terlambat. Secara akademis, semester satu dan dua merupakan masa adaptasi awal mahasiswa terhadap sistem perkuliahan, beban akademik, serta pola belajar di perguruan tinggi. Perubahan IP pada periode ini dapat mencerminkan apakah mahasiswa mengalami peningkatan, penurunan, atau kestabilan performa setelah melewati semester pertama. Dengan demikian, delta_ip12 dapat dipahami sebagai indikator dinamika awal performa akademik, bukan hanya sebagai selisih nilai antarsemester. Konsistensi kemunculan fitur ini pada ketiga model menunjukkan bahwa dinamika performa akademik di awal masa studi memiliki kontribusi yang relevan terhadap prediksi ketepatan waktu kelulusan mahasiswa.

Namun demikian, hasil *permutation feature importance* tetap perlu diinterpretasikan secara hati-hati. Nilai *importance* pada metode ini dihitung berdasarkan perubahan performa model ketika nilai suatu fitur diacak pada data uji. Oleh karena itu, nilai *importance* tidak menunjukkan hubungan sebab-akibat antara fitur dan ketepatan waktu kelulusan, melainkan hanya menunjukkan seberapa besar suatu fitur berpengaruh terhadap performa prediksi model dalam skenario evaluasi yang digunakan. Selain itu, karena perhitungan dilakukan pada data uji dengan jumlah terbatas, hasil *importance* dapat dipengaruhi oleh karakteristik dan distribusi data uji. Dengan demikian, analisis ini digunakan sebagai pendukung interpretasi model, bukan

sebagai satu-satunya dasar untuk menentukan relevansi akademik suatu fitur. Grafik *permutation feature importance* untuk masing-masing model disajikan pada Lampiran F.

Perbandingan Performa Model

Perbandingan performa model dilakukan untuk menentukan model terbaik dari tiga algoritma *gradient boosting* yang digunakan, yaitu *Extreme Gradient Boosting* (XGB), *Histogram-Based Gradient Boosting* (HGB), dan *Light Gradient Boosting Machine* (LGBM). Perbandingan ini dilakukan berdasarkan beberapa hasil evaluasi yang telah dijabarkan pada beberapa cucu subbab sebelumnya, meliputi evaluasi kinerja *train* dan *cross-validation*, evaluasi awal dengan *threshold* 0,50, optimasi *threshold*, evaluasi akhir setelah optimasi *threshold*, serta analisis AUC-ROC.

Tabel 4.20 Ringkasan Perbandingan Performa Model

Model	Mean CV F1 Macro	Gap	*Threshold* Akhir	Test F1 Macro	AUC-ROC
XGB	0,7282	0,1493	0,52	0,7973	0,8145
HGB	0,7291	0,0663	0,39	0,7502	0,8108
LGBM	0,7286	0,1675	0,51	0,7568	0,7955

Berdasarkan Tabel 4.20, XGB dipilih sebagai model terbaik pada skenario penelitian ini. Pemilihan tersebut didasarkan pada performa akhir di data uji, terutama karena XGB memperoleh nilai *macro F1-score* dan AUC-ROC yang lebih tinggi dibandingkan model lainnya. Selain itu, *confusion matrix* XGB juga menunjukkan distribusi kesalahan yang relatif seimbang, yaitu 7 *false positive* dan 8 *false negative*. Kondisi ini menunjukkan bahwa XGB tidak hanya menghasilkan performa metrik tertinggi, tetapi juga mampu menjaga keseimbangan prediksi pada kedua kelas. Dengan demikian, XGB dinilai sebagai model yang paling sesuai untuk digunakan sebagai model akhir dalam prediksi ketepatan waktu kelulusan mahasiswa pada penelitian ini.

4.7.2 Hasil Evaluasi Tambahan pada Data Berdasarkan Angkatan

Setelah XGB dipilih sebagai model terbaik pada skenario evaluasi utama menggunakan data gabungan, dilakukan evaluasi tambahan untuk melihat konsistensi performa model pada skenario data berdasarkan angkatan. Seperti yang telah dijelaskan pada Anak Subbab 3.10.2, evaluasi ini dilakukan menggunakan pendekatan *Leave-One-Group-Out*, dengan angkatan

mahasiswa sebagai kelompok data. Pada setiap grup, satu angkatan digunakan sebagai data uji, sedangkan dua angkatan lainnya digunakan sebagai data latih.

Evaluasi tambahan ini tidak digunakan sebagai dasar utama dalam pemilihan model terbaik, melainkan sebagai analisis pendukung untuk melihat ketahanan performa XGB ketika diuji pada angkatan yang tidak digunakan dalam proses pelatihan. Dengan demikian, hasil pada anak subbab ini digunakan untuk melengkapi interpretasi terhadap hasil evaluasi utama pada data gabungan. Ringkasan hasil evaluasi XGB pada setiap grup data berdasarkan angkatan ditampilkan pada Tabel 4.21.

Tabel 4.21 Hasil Evaluasi Tambahan XGB pada Data Berdasarkan Angkatan

Data Uji	Accuracy	Precision	Recall	Macro F1-score
Group 1	0,6993	0,6061	0,7018	0,6933
Group 2	0,4179	0,3697	0,9362	0,3827
Group 3	0,6667	0,8906	0,7037	0,5151
Rata-rata	0,5946	0,6221	0,7805	0,5304
Standar Deviasi	0,1539	0,2608	0,1348	0,1559

Berdasarkan Tabel 4.21, performa XGB pada skenario data berdasarkan angkatan menunjukkan hasil yang lebih rendah dibandingkan evaluasi utama pada data gabungan. Nilai rata-rata *macro F1-score* yang diperoleh adalah 0,5304, dengan nilai terendah sebesar 0,3827 pada Group 2 dan nilai tertinggi sebesar 0,6933 pada Group 1. Selain itu, standar deviasi *macro F1-score* sebesar 0,1559 menunjukkan adanya variasi performa yang cukup besar antarangkatan.

Pada tabel tersebut, nilai *precision* dan *recall* yang ditampilkan merupakan nilai untuk kelas 1, yaitu kelas lulus tepat waktu. Oleh karena itu, nilai *precision* atau *recall* yang tampak cukup tinggi belum tentu menghasilkan *macro F1-score* yang tinggi. *Macro F1-score* menghitung rata-rata F1-score dari kedua kelas, yaitu kelas 0 dan kelas 1. Dengan demikian, apabila performa model pada salah satu kelas rendah, maka nilai *macro F1-score* juga akan ikut menurun. Hal ini terlihat terutama pada Group 2 dan Group 3, di mana nilai *recall* atau *precision* kelas 1 cukup tinggi, tetapi *macro F1-score* tetap rendah karena performa model pada kelas lainnya tidak seimbang.

Untuk melihat distribusi kesalahan prediksi pada setiap grup, *confusion matrix* hasil evaluasi tambahan ditampilkan pada Tabel 4.22.

Tabel 4.22 *Confusion Matrix* Evaluasi Tambahan XGB pada Data Berdasarkan Angkatan

Data Uji	TP	FP	FN	TN
Group 1	40	26	17	60
Group 2	44	75	3	12
Group 3	57	7	24	5

Berdasarkan Tabel 4.22, pola kesalahan model berbeda pada setiap angkatan. Pada Group 1, model menghasilkan 40 *true positive* dan 60 *true negative*, dengan 26 *false positive* dan 17 *false negative*. Hasil ini menunjukkan bahwa model masih relatif lebih seimbang dalam mengenali kedua kelas dibandingkan skenario lainnya.

Pada Group 2, model menghasilkan 44 *true positive* dan hanya 3 *false negative*, sehingga *recall* kelas 1 menjadi sangat tinggi. Namun, model juga menghasilkan 75 *false positive*, yaitu mahasiswa yang sebenarnya termasuk kelas 0 tetapi diprediksi sebagai kelas 1. Jumlah *false positive* yang tinggi tersebut menunjukkan bahwa model cenderung terlalu banyak memprediksi mahasiswa sebagai lulus tepat waktu pada skenario ini. Akibatnya, performa pada kelas 0 menjadi rendah dan *macro F1-score* Group 2 turun menjadi 0,3827.

Pada Group 3, model menghasilkan 57 *true positive* dan 7 *false positive*, sehingga *precision* kelas 1 menjadi tinggi. Namun, model hanya menghasilkan 5 *true negative* dan 24 *false negative*. Kondisi ini menunjukkan bahwa model masih kesulitan mengenali kelas 0 pada skenario pengujian Group 3. Oleh karena itu, meskipun *precision* kelas 1 tinggi, nilai *macro F1-score* tetap rendah karena performa model pada kedua kelas tidak seimbang.

Secara keseluruhan, hasil evaluasi tambahan berdasarkan angkatan menunjukkan bahwa model XGB belum memiliki performa yang konsisten pada pengujian per angkatan. Rata-rata *macro F1-score* sebesar 0,5304, nilai minimum sebesar 0,3827, dan standar deviasi sebesar 0,1559 menunjukkan bahwa performa model menurun ketika diuji pada skenario pembagian data berdasarkan angkatan. Dengan demikian, meskipun XGB merupakan model terbaik pada evaluasi utama menggunakan data gabungan, model tersebut belum dapat dikatakan *robust* terhadap perbedaan karakteristik antarangkatan. Hasil ini menjadi catatan penting bahwa penerapan model pada angkatan baru perlu dilakukan secara hati-hati dan memerlukan pengujian lanjutan.

4.8 Demonstrasi Model berbasis Streamlit

Setelah model terbaik diperoleh dari rangkaian proses pemodelan dan evaluasi, penelitian ini melakukan demonstrasi model menggunakan Streamlit. Demonstrasi ini dibuat untuk menampilkan bagaimana model dapat digunakan dalam bentuk antarmuka web sederhana. Implementasi dilakukan secara lokal menggunakan Visual Studio Code dengan memuat model terbaik yang telah disimpan dalam format pkl.

Model yang digunakan pada demonstrasi ini adalah model XGB karena model tersebut memperoleh performa terbaik pada evaluasi akhir. Model disimpan menggunakan pustaka joblib, kemudian dimuat kembali pada aplikasi Streamlit. Karena model disimpan dalam bentuk *pipeline*, proses *feature engineering*, *feature selection*, dan klasifikasi dapat dijalankan secara otomatis ketika pengguna memasukkan data akademik. Dengan demikian, aplikasi tidak perlu menjalankan ulang proses pelatihan model, melainkan hanya melakukan proses prediksi berdasarkan model yang telah dilatih sebelumnya.

```
# ========================================
# 3. LOAD MODEL XGBOOST
# ========================================
@st.cache_resource
def load_model():
    # panggil model yang ada di dalam folder yang sama
    return joblib.load('model_xgb.pkl')

try:
    model = load_model()
except Exception as e:
     st.error(f"Gagal memuat model. Pastikan file model .pkl ada di folder
yang sama. Error: {e}")
    model = None
```

Gambar 4.11 Kode Pemanggilan Model pada Streamlit

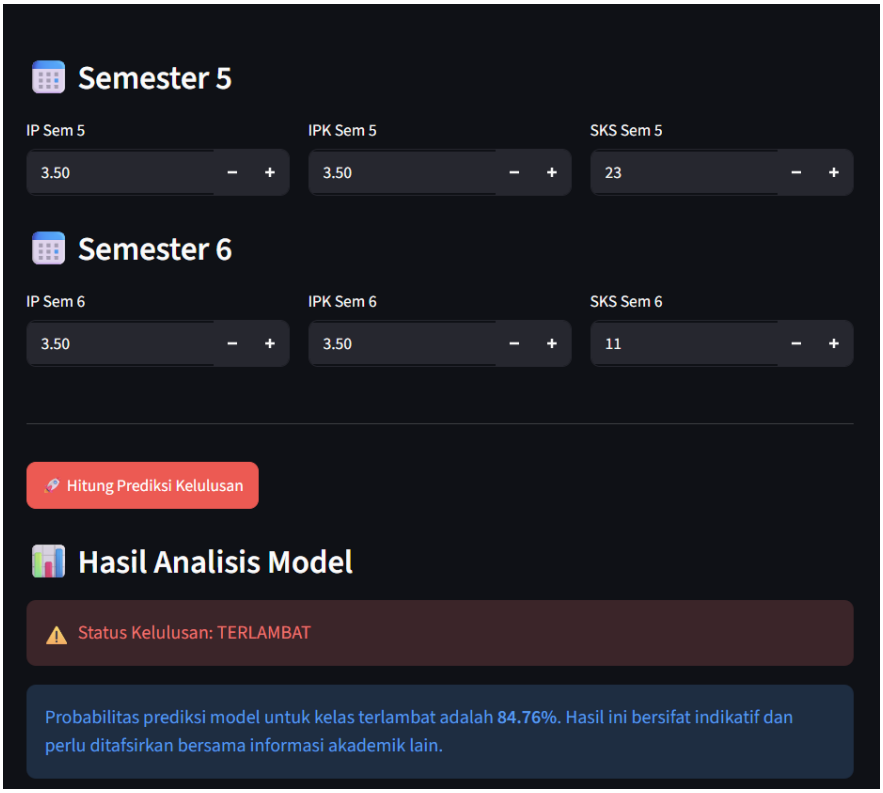
Pada aplikasi Streamlit, pengguna diminta untuk memasukkan data akademik mahasiswa dari semester satu hingga semester enam. Data yang dimasukkan meliputi nilai IP, IPK, dan jumlah SKS pada setiap semester. Input tersebut disusun dalam bentuk form sederhana agar pengguna dapat memasukkan data akademik secara bertahap berdasarkan semester.



Gambar 4.12 Tampilan Form Input Data

Setelah seluruh data akademik dimasukkan, pengguna dapat menekan tombol prediksi untuk menjalankan model. Data input yang diberikan oleh pengguna kemudian diubah menjadi DataFrame dengan struktur kolom yang sama seperti data yang digunakan pada tahap pemodelan. Data tersebut selanjutnya dimasukkan ke dalam *pipeline* model untuk menghasilkan probabilitas prediksi dan label kelas.

Hasil prediksi ditampilkan dalam dua bentuk, yaitu label status kelulusan dan tingkat keyakinan model. Apabila model memprediksi label 1, maka hasil yang ditampilkan adalah mahasiswa diprediksi lulus tepat waktu. Sebaliknya, apabila model memprediksi label 0, maka hasil yang ditampilkan adalah mahasiswa diprediksi lulus terlambat. Selain itu, aplikasi juga menampilkan nilai probabilitas dari kelas yang diprediksi sebagai bentuk informasi tambahan kepada pengguna.



Gambar 4.13 Tampilan Hasil Prediksi

Demonstrasi ini menunjukkan bahwa model yang telah dibangun dapat digunakan untuk melakukan prediksi terhadap data baru melalui antarmuka web sederhana. Namun, aplikasi Streamlit pada penelitian ini hanya digunakan sebagai media demonstrasi model, bukan sebagai sistem akademik yang siap digunakan secara operasional. Oleh karena itu, implementasi ini belum mencakup fitur seperti autentikasi pengguna, integrasi dengan basis data akademik, penyimpanan riwayat prediksi, maupun mekanisme validasi input yang lebih kompleks. Dokumentasi tambahan berupa struktur implementasi, potongan kode inti prediksi, *link* kode program, dan tampilan penuh aplikasi Streamlit disajikan pada Lampiran G. Selain itu, terdapat temuan yang akan dibahas pada subbab selanjutnya.

4.9 Pembahasan Temuan

Berdasarkan rangkaian evaluasi yang telah dilakukan, terdapat beberapa temuan utama dalam penelitian ini. Temuan-temuan tersebut mencakup perbedaan performa model pada validasi silang dan data uji, konsistensi performa model pada skenario data berdasarkan angkatan, pengaruh optimasi *threshold* terhadap distribusi prediksi, peran fitur akademik dalam proses prediksi, serta keterbatasan interpretasi hasil pada tahap demonstrasi model berbasis

Streamlit. Pembahasan temuan ini digunakan untuk menjelaskan hasil penelitian secara lebih kritis, sekaligus menempatkan kontribusi dan keterbatasan penelitian dalam konteks prediksi awal ketepatan waktu kelulusan mahasiswa.

Temuan pertama berkaitan dengan perbedaan performa model pada validasi silang dan data uji. Pada tahap validasi silang, HGB menunjukkan gap *train-CV* paling kecil dan nilai validasi silang yang sedikit lebih tinggi dibandingkan model lainnya. Namun, pada evaluasi akhir menggunakan data uji, XGB menghasilkan performa yang lebih tinggi dalam skenario evaluasi penelitian ini. Perbedaan tersebut menunjukkan bahwa performa model pada validasi silang tidak selalu sepenuhnya sejalan dengan performa pada data uji. Kondisi ini dapat dipengaruhi oleh ukuran dataset, variasi distribusi data, serta sensitivitas model terhadap subset data yang digunakan selama proses pelatihan dan pengujian.

Hal ini memperlihatkan bahwa pemilihan model terbaik tidak cukup hanya didasarkan pada satu tahap evaluasi. Validasi silang penting untuk memperoleh gambaran performa model selama proses optimasi, tetapi evaluasi pada data uji tetap diperlukan untuk menilai kemampuan model terhadap data yang tidak digunakan dalam proses pelatihan dan pemilihan *hyperparameter*. Hasil ini juga sejalan dengan pembahasan penelitian terdahulu pada Bab 2, bahwa tidak ada satu algoritma yang selalu menjadi model terbaik pada seluruh dataset. Performa model sangat dipengaruhi oleh karakteristik data, target prediksi, teknik prapemrosesan, skenario evaluasi, serta metrik yang digunakan. Oleh karena itu, XGB dipilih sebagai model terbaik dalam penelitian ini berdasarkan pertimbangan melalui performa akhir pada data uji dan performa pada data latih, bukan karena XGB secara mutlak selalu lebih unggul dibandingkan HGB dan LGBM.

Temuan kedua berkaitan dengan penurunan performa XGB pada pengujian menggunakan data yang dibagi per angkatan. Pembagian ini dapat merepresentasikan keadaan pada dunia nyata, dimana distribusi mahasiswa yang lulus tepat/terlambat tidak selalu sama di setiap angkatan. Pada penelitian ini, ditemukan bahwa perbedaan nilai *macro F1-score* antara model yang dilatih dengan dua skenario pengujian ada pada angka 0,2669 (~27%). Dengan demikian, hasil yang baik pada pengujian data gabungan tidak menjamin hasil yang sama pada pengujian data per angkatan.

Selain perbedaan distribusi label, periode angkatan 2019, 2020, dan 2021 juga berada pada konteks akademik yang tidak sepenuhnya sama. Periode tersebut berdekatan dengan pandemi COVID-19 dan perubahan kurikulum pada Program Studi Informatika UII. Kedua kondisi tersebut berpotensi memengaruhi proses pembelajaran, pola pengambilan SKS, serta

penyelesaian studi mahasiswa. Namun, penelitian ini tidak memasukkan variabel eksplisit yang merepresentasikan dampak pandemi maupun perubahan kurikulum mengingat keterbatasan data yang dimiliki. Oleh karena itu, pengaruh kedua faktor tersebut tidak dapat disimpulkan secara langsung, tetapi dapat dipahami sebagai konteks yang berpotensi menyebabkan perbedaan karakteristik antarangkatan.

Temuan ketiga berkaitan dengan optimasi *threshold*. Hasil penelitian menunjukkan bahwa perubahan *threshold* tidak selalu memberikan peningkatan performa pada seluruh model. Pada XGB dan LGBM, *threshold* hasil optimasi tidak mengubah hasil prediksi pada data uji dibandingkan *threshold* bawaan. Sementara itu, pada HGB, perubahan *threshold* memengaruhi distribusi prediksi dengan meningkatkan *recall* kelas 1, tetapi disertai peningkatan jumlah *false positive*. Kondisi ini menunjukkan adanya *trade-off* dalam penyesuaian *threshold*, karena model menjadi lebih mampu mengenali mahasiswa yang lulus tepat waktu, tetapi juga lebih sering memprediksi mahasiswa yang sebenarnya lulus terlambat sebagai lulus tepat waktu.

Dalam konteks prediksi ketepatan waktu kelulusan, *trade-off* tersebut perlu diperhatikan karena setiap jenis kesalahan prediksi memiliki konsekuensi yang berbeda. *False positive* menunjukkan kondisi ketika mahasiswa yang sebenarnya lulus terlambat diprediksi sebagai lulus tepat waktu. Kesalahan ini berisiko menyebabkan mahasiswa yang membutuhkan pemantauan akademik tidak teridentifikasi sejak awal. Sebaliknya, *false negative* menunjukkan kondisi ketika mahasiswa yang sebenarnya lulus tepat waktu diprediksi sebagai lulus terlambat, sehingga model memberikan sinyal risiko yang berlebihan terhadap mahasiswa yang sebenarnya tidak mengalami keterlambatan kelulusan. Oleh karena itu, optimasi *threshold* dalam penelitian ini tidak hanya dipahami sebagai upaya meningkatkan nilai metrik evaluasi, tetapi juga sebagai upaya untuk meninjau dampak perubahan batas keputusan terhadap distribusi kesalahan prediksi.

Hal ini memperluas pembahasan pada penelitian terdahulu yang umumnya berfokus pada pemilihan algoritma, *feature selection*, *resampling*, imputasi, atau *hyperparameter tuning*. Dengan menambahkan optimasi *threshold*, penelitian ini menunjukkan bahwa penyesuaian batas probabilitas juga perlu dipertimbangkan dalam evaluasi model klasifikasi, terutama ketika konsekuensi kesalahan prediksi pada masing-masing kelas tidak sama. Selain itu, keputusan akhir dalam menggunakan *threshold* tertentu perlu mempertimbangkan dampak positif dan negatif yang dihasilkan, terutama jika model akan diterapkan pada konteks nyata.

Temuan keempat berkaitan dengan peran fitur akademik dalam proses prediksi. Berdasarkan hasil *feature selection* dan *permutation feature importance*, fitur yang berkaitan dengan beban SKS dan dinamika IP muncul sebagai fitur yang cukup berpengaruh dalam proses prediksi. Beberapa fitur seperti rata-rata SKS, SKS semester enam, perubahan IP awal, dan variasi IP menunjukkan bahwa ketepatan waktu kelulusan tidak hanya berkaitan dengan capaian akademik pada satu semester tertentu, tetapi juga dapat direpresentasikan melalui pola beban studi dan perubahan performa akademik mahasiswa selama beberapa semester.

Temuan tersebut selaras dengan kerangka pemikiran pada Bab 2, yang menempatkan IP dan SKS sebagai fitur dasar karena keduanya merepresentasikan dua aspek akademik yang berbeda. IP mencerminkan capaian akademik mahasiswa, sedangkan SKS mencerminkan beban studi yang ditempuh. Melalui *feature engineering*, kedua jenis data tersebut tidak hanya digunakan sebagai fitur mentah, tetapi juga diolah menjadi fitur turunan yang dapat menggambarkan dinamika akademik mahasiswa dari waktu ke waktu. Dengan demikian, hasil *feature importance* memperkuat bahwa pembentukan fitur turunan memiliki peran dalam membantu model menangkap pola akademik mahasiswa selama enam semester pertama.

Temuan kelima berkaitan dengan hasil prediksi pada demonstrasi model berbasis Streamlit. Berdasarkan gambar, ditemukan bahwa model dapat menghasilkan prediksi yang tidak sepenuhnya selaras dengan interpretasi manusia. Pada salah satu skenario input, mahasiswa dengan nilai IP dan IPK yang konsisten berada pada kisaran 3,50 serta jumlah SKS per semester yang menunjukkan pola pengambilan beban studi relatif tinggi, yaitu 20, 20, 24, 24, 23, dan 11 SKS, diklasifikasikan sebagai mahasiswa yang lulus terlambat. Secara akademik, kombinasi nilai IP, IPK, dan SKS tersebut dapat dianggap sebagai pola mahasiswa dengan capaian akademik baik dan progres studi yang cukup memadai. Namun, model tetap menghasilkan prediksi lulus terlambat berdasarkan pola yang dipelajari dari data latih.

Kondisi tersebut dapat terjadi karena beberapa kemungkinan. Pertama, terdapat kemungkinan bahwa pada data latih, sebagian mahasiswa dengan karakteristik nilai IP dan IPK sekitar 3,50 memiliki label lulus terlambat, sehingga model mempelajari pola tersebut sebagai indikasi kelas lulus terlambat. Kedua, variasi data dengan kombinasi nilai IP, IPK, dan SKS seperti pada skenario demonstrasi tersebut kemungkinan belum cukup terwakili dalam data latih. Akibatnya, model dapat menghasilkan prediksi yang kurang sesuai ketika menerima input manual yang berada di luar pola dominan data latih. Temuan ini menunjukkan bahwa hasil prediksi model perlu ditafsirkan secara hati-hati, terutama pada input manual yang belum tentu selaras dengan distribusi data pelatihan.

Berdasarkan temuan tersebut, demonstrasi Streamlit pada penelitian ini tidak dapat diposisikan sebagai sistem pengambilan keputusan akademik, melainkan hanya sebagai media untuk menunjukkan mekanisme teknis penggunaan model. Nilai probabilitas yang ditampilkan oleh model juga tidak dapat dimaknai sebagai tingkat kepastian absolut, tetapi sebagai probabilitas prediksi yang dihasilkan berdasarkan pola historis data latih. Oleh karena itu, apabila model dikembangkan lebih lanjut, diperlukan pengujian terhadap skenario input yang lebih beragam, validasi terhadap distribusi data pelatihan, serta penambahan mekanisme interpretasi hasil prediksi agar output model tidak disalahartikan oleh pengguna.

Selain keempat temuan utama tersebut, penelitian ini juga memberikan kontribusi dalam konteks prediksi awal ketepatan waktu kelulusan mahasiswa. Sebagaimana dijelaskan pada Bab 2, sebagian penelitian terdahulu masih berfokus pada penggunaan fitur akademik mentah, seperti IP, IPK, SKS, status kelulusan, lama studi, atau fitur aktivitas dari *Learning Management System*. Sementara itu, penelitian ini menggunakan data akademik hingga semester enam dan membentuk fitur turunan yang merepresentasikan perubahan IP antarsemester, variasi IP, nilai IP minimum, nilai IP maksimum, rentang IP, serta rata-rata beban SKS per semester. Dengan demikian, kontribusi penelitian ini terletak pada pemanfaatan riwayat akademik awal mahasiswa secara terstruktur untuk membangun model prediksi awal ketepatan waktu kelulusan.

Kontribusi lain dari penelitian ini adalah perbandingan tiga varian algoritma *gradient boosting*, yaitu HGB, XGB, dan LGBM, dalam skenario evaluasi yang sama. Hasil penelitian menunjukkan bahwa ketiga model dapat digunakan untuk mempelajari pola pada data akademik mahasiswa hingga semester enam, tetapi menghasilkan performa yang berbeda pada tahap validasi silang, evaluasi data uji, optimasi *threshold*, dan analisis AUC-ROC. Perbedaan tersebut menunjukkan bahwa pemilihan model tetap perlu dilakukan secara empiris sesuai dengan karakteristik dataset dan skenario evaluasi yang digunakan. Dengan kata lain, penggunaan model yang lebih sederhana ataupun lebih kompleks tidak secara otomatis menjamin performa terbaik, sehingga proses evaluasi empiris tetap diperlukan.

Secara praktis, penelitian ini menunjukkan bahwa data rekam akademik yang umumnya tersedia dalam sistem akademik, seperti IP, IPK, dan SKS, memiliki potensi untuk digunakan sebagai dasar prediksi awal ketepatan waktu kelulusan mahasiswa. Hal ini relevan bagi perguruan tinggi yang belum tentu memiliki data aktivitas pembelajaran digital yang lengkap, seperti data *Learning Management System*. Namun, kontribusi praktis penelitian ini masih

berada pada tahap pengembangan model dan demonstrasi teknis, bukan pada pembuktian efektivitas sistem operasional dalam proses pemantauan akademik.

Secara umum, hasil penelitian ini menunjukkan bahwa pada *dataset* dan skenario evaluasi yang digunakan, data akademik hingga semester enam dapat digunakan sebagai dasar prediksi awal ketepatan waktu kelulusan mahasiswa. Model dapat menghasilkan prediksi yang cukup baik terhadap kedua kelas kelulusan dengan menggunakan kombinasi fitur akademik awal, fitur turunan, seleksi fitur, optimasi *hyperparameter*, dan optimasi *threshold*. Namun, hasil ini tetap perlu dipahami sebagai prediksi berbasis pola historis data akademik, bukan sebagai penentu mutlak status kelulusan mahasiswa. Faktor lain di luar data akademik, seperti proses pengerjaan tugas akhir, komunikasi dengan dosen pembimbing, kondisi pribadi mahasiswa, perubahan kurikulum, kondisi pembelajaran pada masa pandemi, serta faktor administratif, tetap dapat memengaruhi ketepatan waktu kelulusan tetapi belum tercakup dalam ruang lingkup penelitian ini.

4.10 Keterbatasan Penelitian

Penelitian ini memiliki beberapa keterbatasan yang perlu diperhatikan. Pertama, data yang digunakan berjumlah 370 data mahasiswa setelah proses pembersihan. Jumlah sampel data tersebut hanya berasal dari Program Studi Informatika Program Sarjana Universitas Islam Indonesia. Oleh karena itu, hasil model belum tentu dapat digeneralisasi secara langsung pada program studi atau institusi lain yang memiliki karakteristik kurikulum, aturan akademik, dan pola kelulusan yang berbeda.

Kedua, data historis yang digunakan dalam penelitian ini terbatas pada data akademik hingga semester enam. Penelitian ini berfokus pada prediksi awal, sehingga informasi yang berkaitan langsung dengan proses tugas akhir belum digunakan. Padahal, tahap tugas akhir dapat menjadi salah satu faktor penting yang memengaruhi ketepatan waktu kelulusan mahasiswa. Selain itu, fitur akademik yang digunakan juga terbatas pada fitur capaian akademik seperti IP, IPK, dan SKS. Faktor lain yang juga berpotensi memengaruhi ketepatan waktu kelulusan, seperti kondisi ekonomi, aktivitas nonakademik, motivasi belajar, proses pengerjaan tugas akhir, maupun faktor administratif, belum digunakan dalam penelitian ini.

Ketiga, label target yang digunakan pada penelitian ini merupakan hasil rekonstruksi operasional, bukan status resmi institusi. Artinya, kualitas hasil prediksi akan sangat bergantung pada kualitas pelabelan dan label itu sendiri. Perlu diakui bahwa hasil evaluasi dan

kualitas model pada penelitian ini memiliki kemungkinan untuk berubah secara drastis ketika data resmi dipakai untuk uji coba.

Keempat, proses seleksi fitur menggunakan korelasi Pearson, sehingga fitur dipilih berdasarkan hubungan linear terhadap label dan hubungan linear antarfitur. Pendekatan ini belum tentu menangkap fitur yang memiliki hubungan nonlinear terhadap target, meskipun hubungan tersebut mungkin dapat dimanfaatkan oleh model berbasis *gradient boosting*. Sehingga, metode seleksi fitur lain yang lebih relevan dapat dipertimbangkan pada penelitian selanjutnya.

Kelima, proses optimasi *hyperparameter* pada penelitian ini dilakukan menggunakan *RandomizedSearchCV* dengan jumlah iterasi dan ruang pencarian yang telah ditentukan. Oleh karena itu, konfigurasi *hyperparameter* yang diperoleh merupakan konfigurasi terbaik dalam batas ruang pencarian dan skema validasi yang digunakan, tetapi belum dapat dipastikan sebagai konfigurasi paling optimal secara global. Pengujian dengan ruang pencarian yang lebih luas, jumlah iterasi yang lebih besar, atau metode optimasi lain seperti *GridSearchCV* maupun Bayesian Optimization dapat dipertimbangkan pada penelitian selanjutnya.

Keenam, pada tahap demonstrasi model berbasis Streamlit, ditemukan bahwa model dapat menghasilkan prediksi yang tidak sepenuhnya selaras dengan interpretasi manusia. Pada salah satu skenario input manual, model menghasilkan prediksi yang tidak sepenuhnya selaras dengan interpretasi akademik secara umum, yaitu mengklasifikasikan mahasiswa dengan nilai IP dan IPK yang relatif tinggi serta beban SKS yang memadai sebagai lulus terlambat. Kondisi ini menunjukkan bahwa model masih bergantung pada pola yang dipelajari dari data latih dan dapat menghasilkan prediksi yang kurang intuitif apabila menerima kombinasi input yang tidak cukup terwakili dalam data pelatihan. Oleh karena itu, hasil prediksi model tidak dapat diposisikan sebagai keputusan akademik final, melainkan hanya sebagai indikasi awal yang perlu ditafsirkan bersama informasi akademik lain dan pertimbangan pihak program studi.

Ketujuh, penelitian ini belum membahas mekanisme mitigasi risiko terhadap kesalahan prediksi model. Penelitian ini telah membahas bahwa *false positive* dan *false negative* memiliki konsekuensi yang berbeda dalam konteks pemantauan akademik. Namun, penelitian ini belum merancang prosedur operasional untuk menentukan bagaimana hasil prediksi model sebaiknya diverifikasi, kapan hasil prediksi dapat dipercaya, serta bagaimana langkah tindak lanjut yang perlu dilakukan apabila model menghasilkan prediksi yang berisiko keliru. Dengan demikian, hasil prediksi pada penelitian ini masih terbatas sebagai indikasi awal dan belum dapat digunakan sebagai dasar pengambilan keputusan akademik tanpa validasi tambahan.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan, dapat disimpulkan beberapa hal sebagai berikut:

a. Penelitian ini berhasil membangun model prediksi awal ketepatan waktu kelulusan mahasiswa berdasarkan data akademik hingga semester enam menggunakan algoritma klasifikasi berbasis *gradient boosting*. Model yang dihasilkan juga didemonstrasikan secara lokal menggunakan Streamlit sebagai contoh penerapan model dalam bentuk antarmuka web sederhana. Demonstrasi tersebut ditujukan sebagai bentuk implementasi terbatas dan belum menunjukkan kesiapan model untuk diterapkan secara langsung pada kondisi nyata, karena hasil evaluasi menunjukkan adanya perbedaan performa antara skenario data gabungan dan skenario data berdasarkan angkatan.
b. Pada skenario utama menggunakan data gabungan, perbandingan HGB, XGB, dan LGBM menunjukkan bahwa ketiga model memiliki performa validasi silang yang relatif berdekatan. HGB memperoleh nilai validasi silang tertinggi dan *gap* antara skor data latih dan validasi silang paling kecil, sedangkan XGB menghasilkan performa terbaik pada evaluasi akhir menggunakan data uji. XGB memperoleh *accuracy* sebesar 0,7973, *precision* sebesar 0,8056, *recall* sebesar 0,7838, *macro F1-score* sebesar 0,7973, dan AUC-ROC sebesar 0,8145. Berdasarkan hasil tersebut, XGB dipilih sebagai model terbaik dalam skenario utama penelitian ini. Pemilihan tersebut berlaku dalam konteks data dan prosedur evaluasi yang digunakan pada penelitian ini.
c. Hasil tersebut menunjukkan bahwa performa yang baik pada pembagian data gabungan belum secara langsung menunjukkan kemampuan model untuk mempertahankan performanya ketika diterapkan pada kelompok angkatan dengan karakteristik yang berbeda. Dengan demikian, kemampuan generalisasi model terhadap angkatan lain masih perlu dievaluasi lebih lanjut sebelum model dapat dipertimbangkan untuk digunakan pada penerapan nyata.
d. Analisis kontribusi fitur menggunakan *permutation feature importance* menunjukkan bahwa fitur yang berkaitan dengan beban SKS dan dinamika capaian IP memiliki kontribusi penting terhadap hasil prediksi. Pada model XGB sebagai model terbaik, fitur dengan nilai *importance* tertinggi adalah sks_mean_1_6, diikuti oleh sks6, ip5, delta_ip12,

dan ipk6. Hasil ini menunjukkan bahwa rata-rata beban SKS, beban SKS semester enam, capaian IP semester lima, perubahan IP pada semester awal, dan IPK semester enam menjadi fitur yang cukup berkontribusi dalam proses prediksi ketepatan waktu kelulusan mahasiswa. Meskipun demikian, hasil *permutation feature importance* tidak menunjukkan hubungan sebab-akibat antara fitur dan ketepatan waktu kelulusan, melainkan hanya menunjukkan seberapa besar suatu fitur berpengaruh terhadap performa prediksi model dalam skenario evaluasi yang digunakan.

5.2 Saran

Subbab ini membahas saran yang dapat dipertimbangkan pada penelitian selanjutnya berdasarkan keterbatasan, temuan, dan hasil penelitian yang telah diperoleh. Saran dikelompokkan ke dalam aspek data akademik, metode penelitian, dan implementasi sebagai berikut:

a. Dari sisi data akademik
   1. Penelitian selanjutnya dapat mempertimbangkan penggunaan data yang telah memiliki label status kelulusan yang ditetapkan secara resmi oleh institusi. Pada penelitian berbasis *supervised learning*, kualitas dan validitas label memiliki peran penting karena label digunakan sebagai acuan dalam proses pembelajaran dan evaluasi model. Penggunaan label yang berasal langsung dari sistem atau pencatatan institusi diharapkan dapat mengurangi ketergantungan terhadap proses rekonstruksi label berdasarkan aturan yang ditentukan dalam penelitian.
   2. Penelitian selanjutnya dapat memperluas jumlah dan cakupan data, baik dengan menambahkan lebih banyak angkatan maupun melibatkan program studi, fakultas, atau institusi lain. Penambahan tersebut dapat meningkatkan keragaman karakteristik akademik yang dipelajari oleh model serta memungkinkan pengujian terhadap perbedaan kurikulum, kebijakan akademik, dan pola kelulusan pada populasi yang lebih beragam.
   3. Fitur yang digunakan dapat dikembangkan dengan memasukkan informasi lain yang berpotensi berkaitan dengan ketepatan waktu kelulusan mahasiswa. Beberapa fitur yang dapat dipertimbangkan antara lain progres tugas akhir, riwayat pengulangan mata kuliah, status pengambilan mata kuliah wajib, aktivitas akademik tambahan, atau faktor administratif yang berkaitan dengan proses kelulusan. Penambahan fitur

tersebut berpotensi membantu model menangkap aspek yang belum direpresentasikan oleh data capaian akademik semester satu hingga enam.

b. Dari sisi metode penelitian

1. Penelitian selanjutnya dapat melakukan validasi berbasis waktu atau angkatan secara lebih luas untuk mengevaluasi kemampuan model dalam mempertahankan performanya pada data mahasiswa dari periode yang lebih baru. Pendekatan seperti *forward chaining* dapat dipertimbangkan dengan melatih model menggunakan angkatan yang lebih lama dan mengujinya pada angkatan berikutnya. Evaluasi tersebut penting untuk mengetahui kemampuan generalisasi model ketika menghadapi perubahan karakteristik data antarperiode.
2. Penelitian selanjutnya dapat membandingkan metode *feature selection* lain yang mampu menangkap hubungan nonlinear antara fitur dan target. Pendekatan seperti *mutual information, model-based feature selection*, atau metode interpretasi lain dapat digunakan sebagai pembanding terhadap seleksi fitur berbasis korelasi Pearson pada penelitian ini.
3. Proses optimasi *hyperparameter* dapat dikembangkan menggunakan metode pencarian yang lebih terarah, seperti Bayesian Optimization atau Optuna. Pendekatan tersebut dapat digunakan untuk mengeksplorasi ruang *hyperparameter* secara lebih terarah dibandingkan pencarian acak, terutama apabila penelitian selanjutnya menggunakan jumlah data dan ruang parameter yang lebih besar.
4. Penelitian selanjutnya dapat mengevaluasi strategi yang lebih sesuai ketika distribusi kelas berbeda antarangkatan. Pendekatan seperti pemberian bobot kelas, penyesuaian fungsi objektif, atau strategi penentuan *threshold* yang mempertimbangkan distribusi masing-masing kelompok dapat diuji untuk mengetahui apakah kestabilan performa model pada data lintas angkatan dapat ditingkatkan.
5. Penelitian selanjutnya dapat mengembangkan metode interpretasi dan audit model untuk membantu pengguna memahami alasan di balik hasil prediksi. Pendekatan seperti analisis kontribusi fitur, pemeriksaan distribusi input terhadap data latih, kalibrasi probabilitas, serta evaluasi kasus-kasus prediksi yang salah dapat digunakan untuk menilai tingkat keandalan output model. Pengembangan ini penting agar model prediksi tidak hanya mampu menghasilkan label kelas, tetapi juga memberikan informasi pendukung yang membantu pihak akademik dalam menilai kelayakan hasil prediksi.

c. Dari sisi implementasi

1. Demonstrasi aplikasi yang dikembangkan pada penelitian ini dapat dilanjutkan menjadi sistem pendukung yang terintegrasi dengan lingkungan akademik. Namun, pengembangan tersebut sebaiknya dilakukan setelah model memperoleh validasi yang lebih luas terhadap data lintas angkatan dan populasi yang berbeda.
2. Apabila model dikembangkan untuk penggunaan nyata, hasil prediksi sebaiknya ditempatkan sebagai informasi pendukung bagi pihak akademik dan bukan sebagai dasar tunggal dalam menentukan kondisi atau tindakan terhadap mahasiswa. Mekanisme penggunaan sistem juga perlu mempertimbangkan risiko kesalahan prediksi, khususnya *false positive* dan *false negative*, serta menyediakan ruang bagi evaluasi oleh pihak yang memiliki pengetahuan mengenai kondisi mahasiswa secara lebih menyeluruh.
3. Sistem yang dikembangkan selanjutnya dapat dilengkapi dengan mekanisme pemantauan performa model setelah diterapkan. Evaluasi berkala diperlukan untuk mengetahui apakah distribusi data dan performa model mengalami perubahan pada angkatan baru sehingga model dapat dievaluasi, dikalibrasi ulang, atau dilatih kembali apabila diperlukan.

DAFTAR PUSTAKA

Alpaydin, E. (2014). *Introduction to Machine Learning* (3rd ed.). Cambridge: MIT Press. Retrieved from https://archive.org/details/introductiontoma00alpa_0

Amiruddin, & Ishak, R. (2022). Implementasi Seleksi Fitur Klasifikasi Waktu Kelulusan Mahasiswa Menggunakan Correlation Matrix with Heatmap. *Jambura Journal of Electrical and Electronics Engineering*, *4*(2), 169–174. doi:10.37905/jjeee.v4i2.14403

Aucejo, E. M., French, J., Ugalde Araya, M. P., & Zafar, B. (2020). The Impact of COVID-19 on Student Experiences and Expectations: Evidence from a Survey. *Journal of Public Economics*, *191*, 104271. doi:10.1016/J.JPUBECO.2020.104271

Aydin, Z. E., & Ozturk, Z. K. (2021). *Performance Analysis of XGBoost Classifier with Missing Data*. In *Proceedings of the 1st International Conference on Computing and Machine Intelligence (ICMI 2021)* (pp. 39–43). Istanbul Sabahattin Zaim University. Retrieved from https://www.izu.edu.tr/docs/default-source/kitaplar/iz%C3%BC-yay%C4%B1n-no-57--conference-proceedings-icmi-2021.pdf?sfvrsn=7989fc52_0

Ayulani, I. D., Yunawan, A. M., Prihutaminingsih, T., Sarwinda, D., Ardaneswari, G., & Handari, B. D. (2023). Tree-Based Ensemble Methods and Their Applications for Predicting Students' Academic Performance. *International Journal on Advanced Science, Engineering and Information Technology*, *13*(3), 919–927. doi:10.18517/ijaseit.13.3.16880

Baccar, Y. Ben. (2019). *Comparative Study on Time Series Forecasting* (Master of Science (Data Science)). Institut Polytechnique de Paris, Telecom Paris, Paris. https://doi.org/10.13140/RG.2.2.32241.02408

Badan Akreditasi Nasional Perguruan Tinggi. (2025). *Peraturan BAN-PT Nomor 13 Tahun 2025 tentang Instrumen Akreditasi Program Studi*. Jakarta. Retrieved from https://www.banpt.or.id/wp-content/uploads/2025/06/Peraturan-BAN-PT-Nomor-13-Tahun-2025-ttg-IAPS-5.0.pdf

Bakri, R., Alam, S., Astuti, N. P., & Bakhtiar, M. I. (2025). Optimizing Machine Learning Models for Graduation on Time Prediction: A Comparative Study with Resampling and Hyperparameter Tuning. *Jurnal Online Informatika*, *10*(2), 270–285. doi:10.15575/join.v10i2.1590

Bergstra, J., & Bengio, Y. (2012). Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research*, *13*, 281–305. Retrieved from https://www.jmlr.org/papers/volume13/bergstra12a/bergstra12a.pdf

Bommert, A., Sun, X., Bischl, B., Rahnenführer, J., & Lang, M. (2019). Benchmark for Filter Methods for Feature Selection in High-Dimensional Classification Data. *Computational Statistics & Data Analysis*, *143*, 106839. doi:10.1016/j.csda.2019.106839

Chen, T., & Guestrin, C. (2016). *XGBoost: A Scalable Tree Boosting System*. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 785–794). New York, NY, USA: Association for Computing Machinery. doi:10.1145/2939672.2939785

Diekuu, J.-B., Mekala, M. S., Abonie, U. S., Isaacs, J., & Elyan, E. (2025). Predicting Student Next-Term Performance in Degree Programs Using AI-Based Approach: A Case Study from Ghana. *Cogent Education*, *12*(1). doi:10.1080/2331186X.2025.2481000

Dwinanda, M. W., Satyahadewi, N., & Andani, W. (2023). Classification of Student Graduation Status Using XGboost Algorithm. *Barekeng*, *17*(3), 1785–1794. doi:10.30598/barekengvol17iss3pp1785-1794

Fawcett, T. (2006). An Introduction to ROC analysis. *Pattern Recognition Letters*, *27*, 861–874. doi:10.1016/j.patrec.2005.10.010

Fernandez-Garcia, A. J., Preciado, J. C., Melchor, F., Rodriguez-Echeverria, R., Conejero, J. M., & Sanchez-Figueroa, F. (2021). A Real-Life Machine Learning Experience for Predicting University Dropout at Different Stages Using Academic Data. *IEEE Access*, *9*, 133076–133090. doi:10.1109/ACCESS.2021.3115851

Guyon, I., & Elisseeff, A. (2003). An Introduction of Variable and Feature Selection. *Journal of Machine Learning Research*, *3*, 1157–1182. doi:10.1162/153244303322753616

Hall, M. (1999). *Correlation-Based Feature Selection for Machine Learning*. *Department of Computer Science*. The University of Waikato, Hamilton. Retrieved from https://ml.cms.waikato.ac.nz/publications/1999/99MH-Thesis.pdf

Haryatmi, E., & Pramita Hervianti, S. (2021). Penerapan Algoritma Support Vector Machine Untuk Model Prediksi Kelulusan Mahasiswa Tepat Waktu. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, *5*(2), 386–392. doi:10.29207/resti.v5i2.3007

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition (Springer Series in Statistics)* (2nd ed.). NY: Springer New York. doi:10.1007/978-0-387-84858-7

Hubbard, K., & Amponsah, S. (2025). *Feature Engineering on LMS Data to Optimize Student Performance Prediction*. In J. Han Henry and Stamey (Ed.), *New Frontiers in Data Science* (pp. 125–142). Springer Nature Switzerland. doi:10.1007/978-3-031-99879-9_8

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., … Liu, T.-Y. (2017). *LightGBM: a highly efficient gradient boosting decision tree*. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (pp. 3149–3157). Red Hook, NY, USA: Curran Associates Inc. doi:10.5555/3294996.3295074

Khoirunnisa, R. N., Jannah, M., Dewi, D. K., & Satiningsih, S. (2021). Prokrastinasi Akademik Mahasiswa Tingkat Akhir pada Masa Pandemi COVID-19. *Jurnal Psikologi Teori Dan Terapan*, *11*(3), 278. doi:10.26740/jptt.v11n3.p278-292

Kohavi, R. (1995). *A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection*. In *International Joint Conference on Artificial Intelligence* (Vol. 14, pp. 1–7). Stanford University. Retrieved from https://www.researchgate.net/publication/2352264_A_Study_of_Cross-Validation_and_Bootstrap_for_Accuracy_Estimation_and_Model_Selection

LAM INFOKOM. (2024). *Sertifikat Akreditasi Program Studi Informatika Program Sarjana Universitas Islam Indonesia*. Retrieved from https://informatics.uii.ac.id/wp-content/uploads/2025/02/sertifikat-akreditasi-LAM-INFOKOM-IF-S1.pdf

Lipton, Z. C., Elkan, C., & Naryanaswamy, B. (2014). Optimal Thresholding of Classifiers to Maximize F1 Measure. In F. and H. E. and M. R. Calders Toon and Esposito (Ed.), *Machine Learning and Knowledge Discovery in Databases* (pp. 225–239). Berlin, Heidelberg: Springer Berlin Heidelberg. doi:10.1007/978-3-662-44851-9_15

Nirmala, I., Wijayanto, H., & Notodiputro, K. A. (2022). Prediction of Undergraduate Student's Study Completion Status Using MissForest Imputation in Random Forest and XGBoost Models. *ComTech: Computer, Mathematics and Engineering Applications*, *13*(1), 53–62. doi:10.21512/comtech.v13i1.7388

Novianto, E., Hermawan, A., & Avianto, D. (2023). Klasifikasi Algoritma K-Nearest Neighbor, Naive Bayes, dan Decision Tree untuk Prediksi Status Kelulusan Mahasiswa S1. *Rabit : Jurnal Teknologi Dan Sistem Informasi Univrab*, *8*(2), 146–154. doi:10.36341/rabit.v8i2.3434

Pawitra, M. A. S., Hung, H.-C., & Jati, H. (2024). A Machine Learning Approach to Predicting On-Time Graduation in Indonesian Higher Education. *Elinvo (Electronics, Informatics, and Vocational Education)*, *9*(2), 294–308. doi:10.21831/elinvo.v9i2.77052

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., … Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830. doi:10.48550/arXiv.1201.0490

Probst, P., Boulesteix, A.-L., & Bischl, B. (2019). Tunability: Importance of Hyperparameters of Machine Learning Algorithms. *Journal of Machine Learning Research*, *20*, 1–32. doi:10.48550/arXiv.1802.09596

Program Studi Informatika UII. (2021). *Panduan Akademik Program Studi Informatika UII 2021/2022*. Yogyakarta. Retrieved from https://informatics.uii.ac.id/wp-content/uploads/2021/09/Panduan-Akademik-Informatika-20212022.docx-compressed.pdf

Program Studi Informatika UII. (2022). *Materi Penjelasan Akademik Program Studi Informatika UII 2022*. Yogyakarta. Retrieved from https://informatics.uii.ac.id/wp-content/uploads/2022/08/Penjelasan-Akademik-2022-1.pdf

Rachardian, S., & Sediyono, E. (2024). Prediksi Kelulusan Tepat Waktu Mahasiswa Untuk Pemantauan Program Studi Menggunakan Metode Data Mining. *AITI: Jurnal Teknologi Informasi*, *21*(2), 168–182. doi:10.24246/aiti.v21i2.168-182

Romero, C., & Ventura, S. (2010). Educational Data Mining: A Review of the State-of-the-Art. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions On*, *40*, 601–618. doi:10.1109/TSMCC.2010.2053532

Romero, C., & Ventura, S. (2020). Educational Data Mining and Learning Analytics: An Updated Survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*. doi:10.1002/widm.1355

Saunders, Mark., Lewis, Philip., & Thornhill, Adrian. (2003). *Research Methods for Business Students* (3rd ed.). Harlow: Prentice Hall. Retrieved from https://archive.org/details/researchmethodsf0000saun_3rd/page/188/mode/2up

Sokolova, M., & Lapalme, G. (2009). A Systematic Analysis of Performance Measures for Classification Tasks. *Information Processing & Management*, *45*, 427–437. doi:10.1016/j.ipm.2009.03.002

Swastikawati, C., & Utami, E. (2025). Prediksi Kelulusan Calon Mahasiswa dengan Stacking Ensemble Learning. *JURNAL SISTEMASI (Sistem Informasi)*, *14*(6), 2707–2723. Retrieved from http://sistemasi.ftik.unisi.ac.id

Zheng, A., & Casari, A. (2018). *Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists* (1st ed.). Sebastopol: O'Reilly Media, Inc. Retrieved from https://www.oreilly.com/library/view/feature-engineering-for/9781491953235/

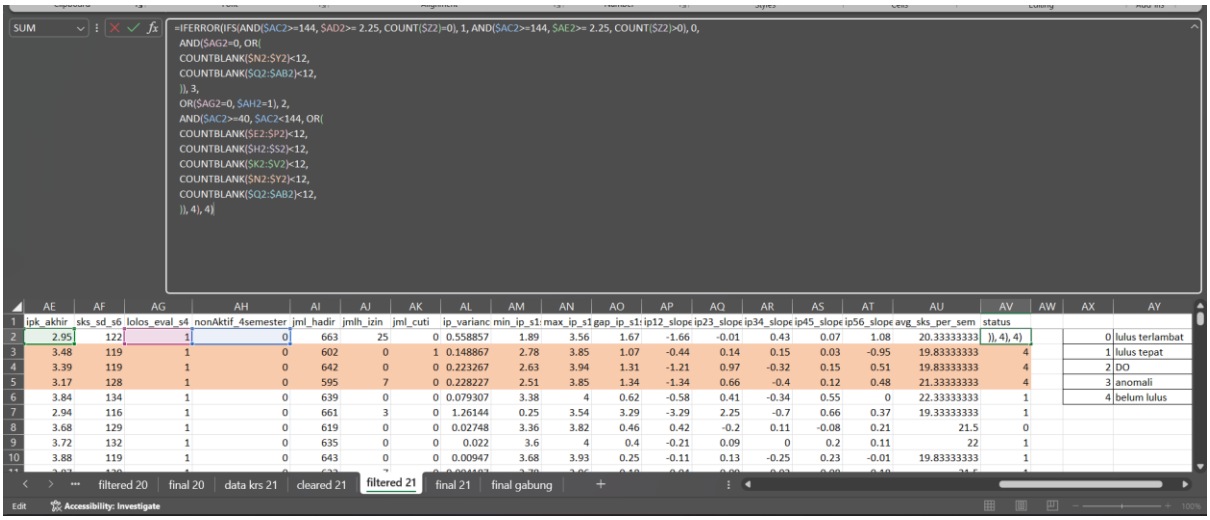
LAMPIRAN

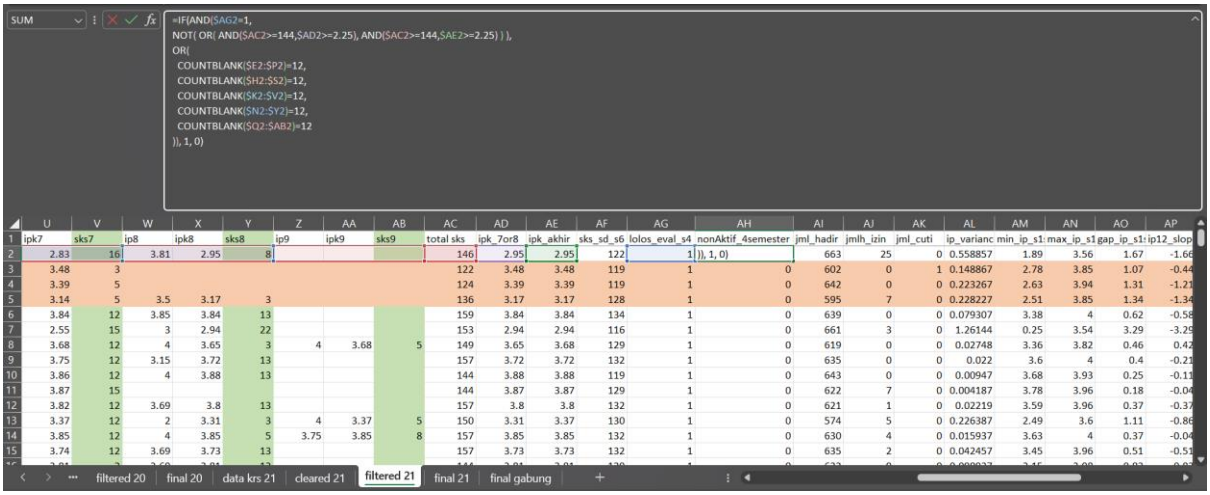
Lampiran pada penelitian ini disusun sebagai dokumen pendukung yang memuat artefak penting dari proses penelitian. Lampiran tidak dimaksudkan untuk mengulang seluruh uraian pada Bab 3 dan Bab 4, melainkan untuk menyediakan rekapitulasi data, hasil evaluasi, dokumentasi visual, serta tautan kode program yang mendukung transparansi dan keterlacakan proses penelitian. Oleh karena itu, lampiran dikelompokkan berdasarkan tahapan utama penelitian, yaitu dokumentasi data dan *labeling*, *feature engineering* dan *feature selection*, *modeling* dan *hyperparameter tuning*, evaluasi model, visualisasi evaluasi, serta demonstrasi model berbasis Streamlit.

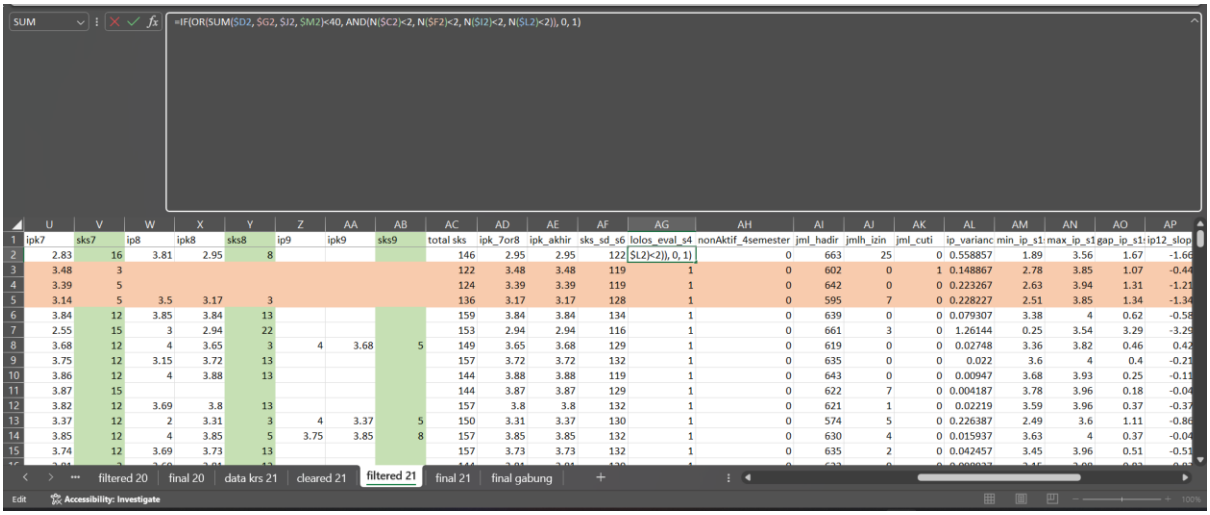
A. Lampiran A. Dokumentasi Data, *Labeling*, *Cleaning*, dan Pembagian Data

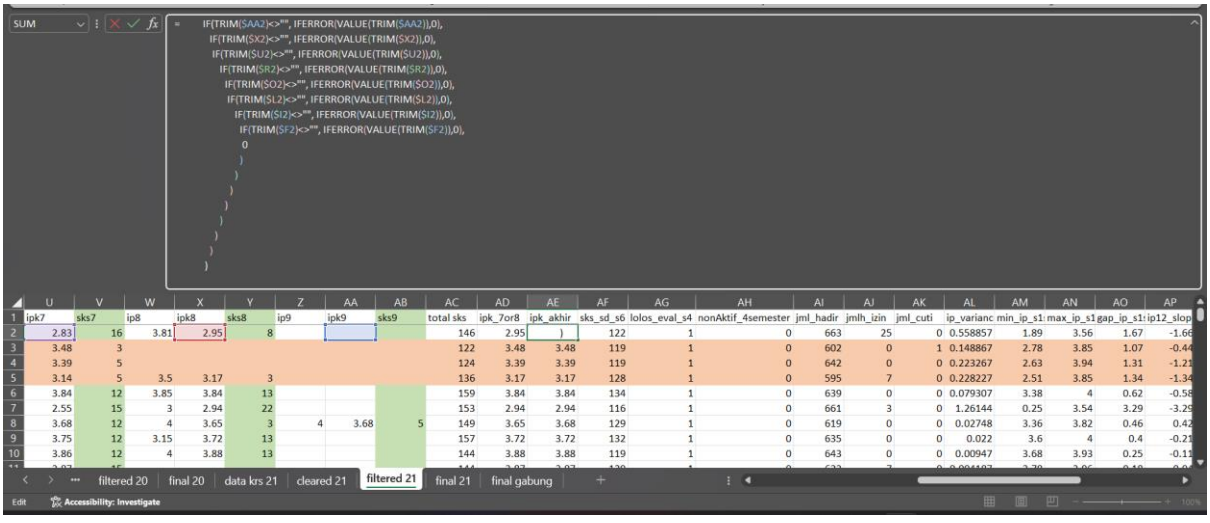
A.1 Aturan *Labeling* Data

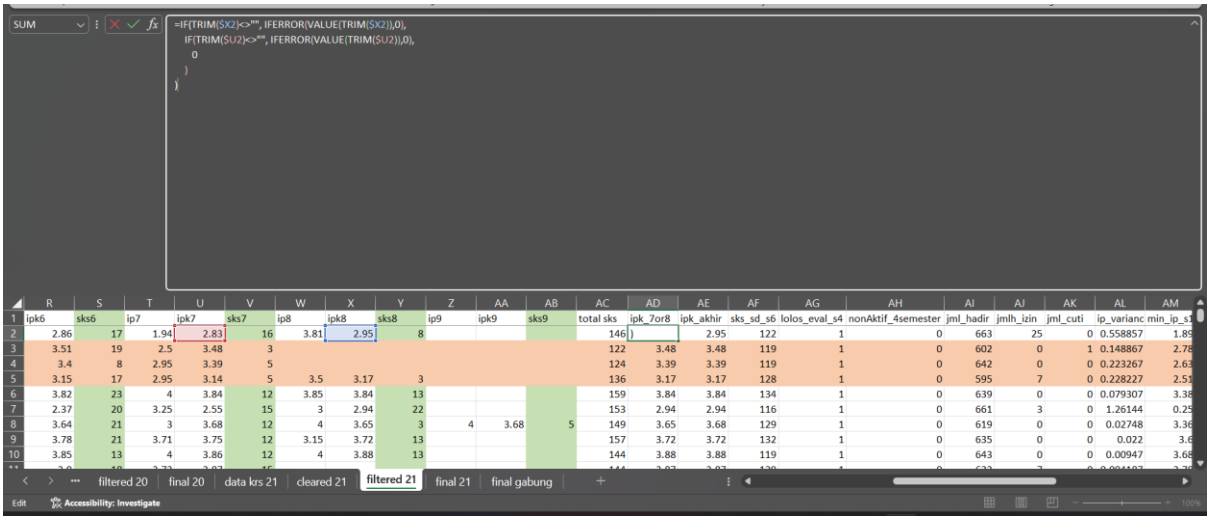
Label	**Arti**	**Kriteria umum**
0	Lulus terlambat	Memenuhi syarat kelulusan, tetapi melebihi batas waktu 8 semester
1	Lulus tepat waktu	Memenuhi syarat kelulusan dalam batas waktu 8 semester
2	Dropout	Tidak memenuhi kriteria evaluasi akademik tertentu
3	Anomali	Data tidak konsisten dengan aturan yang berlaku
4	Belum lulus	Belum memenuhi syarat kelulusan pada titik data yang tersedia

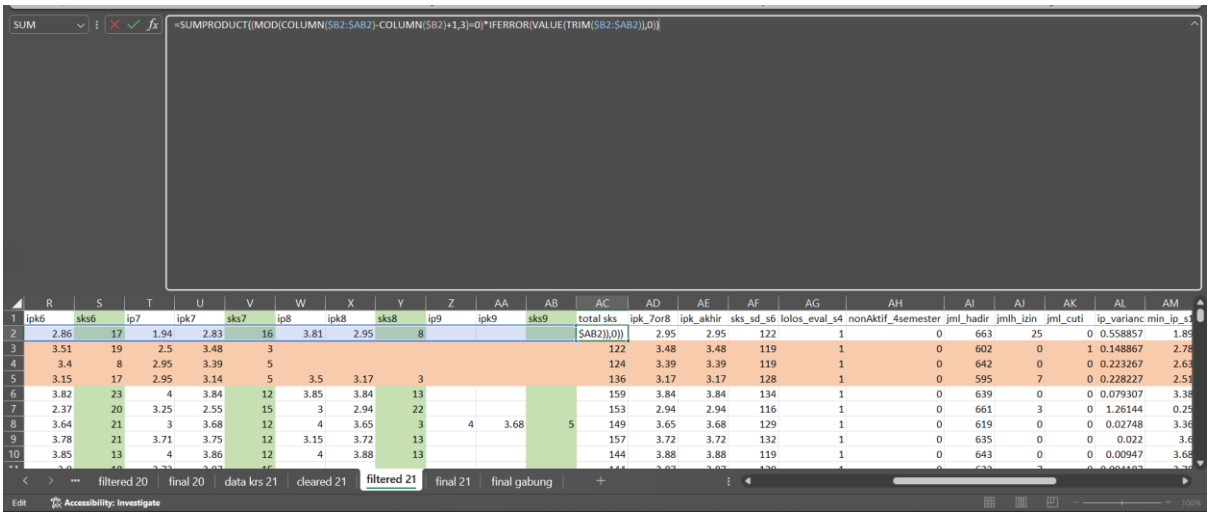



SUM
=IF(AND($AG2=1,
NOT(OR(AND($AC2>=144,$AD2>=2.25), AND($AC2>=144,$AE2>=2.25))),
OR(
COUNTBLANK($E2:$P2)=12,
COUNTBLANK($H2:$S2)=12,
COUNTBLANK($K2:$V2)=12,
COUNTBLANK($N2:$Y2)=12,
COUNTBLANK($Q2:$AB2)=12
)), 1, 0)
filtered 20
final 20
data krs 21
cleared 21
filtered 21
final 21
final gabung


SUM
=IF(OR(SUM($D2, $G2, $J2, $M2)<40, AND(N($C2)<2, N($F2)<2, N($I2)<2, N($L2)<2)), 0, 1)
filtered 20
final 20
data krs 21
cleared 21
filtered 21
final 21
final gabung
Edit
Accessibility: Investigate


SUM
=
IF(TRIM($AA2)<>"", IFERROR(VALUE(TRIM($AA2)),0),
IF(TRIM($X2)<>"", IFERROR(VALUE(TRIM($X2)),0),
IF(TRIM($U2)<>"", IFERROR(VALUE(TRIM($U2)),0),
IF(TRIM($R2)<>"", IFERROR(VALUE(TRIM($R2)),0),
IF(TRIM($O2)<>"", IFERROR(VALUE(TRIM($O2)),0),
IF(TRIM($L2)<>"", IFERROR(VALUE(TRIM($L2)),0),
IF(TRIM($I2)<>"", IFERROR(VALUE(TRIM($I2)),0),
IF(TRIM($F2)<>"", IFERROR(VALUE(TRIM($F2)),0),
0
)
)
)
)
)
)
)
)
filtered 20
final 20
data krs 21
cleared 21
filtered 21
final 21
final gabung
Edit
Accessibility: Investigate





A.2 Penjelasan Kolom Pendukung pada Proses *Labeling*

Kolom	Keterangan
AV	Kolom yang melakukan pelabelan berdasarkan aturan kelulusan Prodi Informatika UII. Kolom ini melibatkan beberapa kolom lain seperti AH, AG, AE, AD, AC, serta rentang kolom E-AB.
AH	Kolom untuk mengecek status mahasiswa yang non aktif berturut-turut selama 4 semester
AG	Kolom untuk mengecek mahasiswa yang lolos evaluasi 4 semester pertama
AE	Kolom untuk mengambil nilai IPK terakhir selama kuliah
AD	Kolom untuk mengambil nilai IPK pada semester 7 atau 8. Logika dibuat agar mengambil nilai pada semester 8 terlebih dulu, jika tidak tersedia, maka akan mengambil nilai dari semester 7
AC	Kolom untuk menghitung total SKS selama kuliah
E-AB	Kolom ini memuat data IP, IPK, dan SKS secara berurutan

A.3 Rekap Distribusi Label Awal

Angkatan	Label 0	Label 1	Label 2	Label 3	Label 4	Jumlah Data
2019	86	57	38	3	6	190
2020	87	47	38	1	25	198
2021	12	81	32	1	53	179
Gabungan	185	185	108	5	84	567

A.4 Rekap Distribusi Label Setelah *Cleaning*

Angkatan	Label 0	Label 1	Jumlah Data
2019	86	57	143
2020	87	47	134
2021	12	81	93
Gabungan	185	185	370

A.5 Persentase *Missing Values*

Fitur	Missing Values(%)
IP1	0.000000
IPK1	0.000000
SKS1	0.000000
IP2	0.270270
IPK2	0.270270
SKS2	0.270270
IP3	0.540541
IPK3	0.540541
SKS3	0.540541
IP4	0.810811
IPK4	0.810811
SKS4	0.810811
IP5	0.270270
IPK5	0.270270
SKS5	0.270270
IP6	0.540541
IPK6	0.540541
SKS6	0.540541

A.6 Distribusi Data pada Skenario Data Gabungan

Jenis Data	Label 0	Label 1	Jumlah Data
Data latih	148	148	296
Data uji	37	37	74

A.7 Distribusi Data pada Skenario Data Berdasarkan Angkatan

Konfigurasi	Data Latih	Data Uji	Test L0	Test L1	Train L0	Train L1	Jumlah Data
Grup 1	2020 + 2021	2019	86	57	99	128	370
Grup 2	2019 + 2021	2020	87	47	98	138	370
Grup 3	2019 + 2020	2021	12	81	173	104	370

B. Lampiran B. Dokumentasi *Feature Engineering* dan *Feature Selection*

B.1 Daftar Fitur Dasar

Nama Fitur	Tipe Data	Keterangan	Jumlah Data
ip1-ip6	float	nilai indeks prestasi mahasiswa	6
ipk1-ipk6	float	Nilai indeks prestasi kumulatif	6
sks1-sks6	integer	Nilai satuan kredit semester	6
Total			18

B.2 Daftar Fitur Turunan

Kelompok Fitur	Fitur turunan	Keterangan
Perubahan IP antarsemester	delta_ip12	Selisih IP semester 2 dan semester 1
	delta_ip23	Selisih IP semester 3 dan semester 2
	delta_ip34	Selisih IP semester 4 dan semester 3
	delta_ip45	Selisih IP semester 5 dan semester 4
	delta_ip56	Selisih IP semester 6 dan semester 5
Statistik IP	ip_var_1_6	Varians IP dari semester 1 sampai 6
	ip_min_1_6	Minimum IP dari semester 1 sampai 6
	ip_max_1_6	Maksimum IP dari semester 1 sampai 6
	ip_gap_1_6	Selisih antara nilai Maksimum IP dengan Minimum IP
Rata-rata SKS	sks_mean_1_6	Rerata SKS per semester selama 6 semester

B.3 Rekap Fitur Terpilih

Fitur Terpilih	XGB	HGB	LGBM	Nilai Korelasi dengan Label
sks_mean_1_6	Terpilih	Terpilih	Dibuang	0.320190
ipk1	Dibuang	Terpilih	Dibuang	0.319045
ip1	Terpilih	Dibuang	Terpilih	0.317541
ip_max_1_6	Terpilih	Terpilih	Dibuang	0.309718
ipk6	Terpilih	Dibuang	Dibuang	0.289097
ipk5	Dibuang	Dibuang	Dibuang	0.279306

ipk4	Dibuang	Dibuang	Dibuang	0.275743
ipk3	Dibuang	Terpilih	Dibuang	0.261286
ip_min_1_6	Dibuang	Dibuang	Dibuang	0.248522
ip3	Terpilih	Dibuang	Terpilih	0.246584
ip4	Terpilih	Dibuang	Terpilih	0.238611
ipk2	Terpilih	Dibuang	Terpilih	0.233569
sks4	Terpilih	Dibuang	Terpilih	0.224738
sks6	Terpilih	Dibuang	Terpilih	0.220538
ip6	Terpilih	Dibuang	Terpilih	0.208614
delta_ip12	Terpilih	Terpilih	Terpilih	0.200606
ip5	Terpilih	Dibuang	Terpilih	0.198935
ip_gap_1_6	Dibuang	Dibuang	Dibuang	0.175567
delta_ip23	Terpilih	Terpilih	Terpilih	0.171985
sks1	Terpilih	Dibuang	Terpilih	0.164648
sks2	Terpilih	Dibuang	Dibuang	0.161566
ip_var_1_6	Terpilih	Terpilih	Terpilih	0.127252
sks5	Terpilih	Dibuang	Terpilih	0.120044
ip2	Dibuang	Dibuang	Dibuang	0.089964
sks3	Dibuang	Dibuang	Dibuang	0.084765
delta_ip45	Dibuang	Dibuang	Dibuang	0.062501
delta_ip34	Dibuang	Dibuang	Dibuang	0.034302
delta_ip56	Dibuang	Dibuang	Dibuang	0.018227

B.4 Ringkasan *Threshold Feature Selection* Terbaik

Model	min_label_r	max_inter_feature_r	drop_raw
XGB	0.1	0.85	False
HGB	0.1	0.8	True
LGBM	0.1	0.75	False

C. Lampiran C. Dokumentasi *Modeling* dan Optimasi *Hyperparameter*

C.1 Ringkasan *Pipeline* Model

Pipeline terdiri dari komponen FeatureEngineer, PearsonCorrelationSelector, dan *gradient boosting classifier*. *Pipeline* digunakan agar proses *feature engineering* dan *feature selection* dilakukan secara konsisten di dalam *cross-validation*.

C.2 Parameter Terbaik Setiap Model

Model	Hyperparameter	Nilai Terbaik
XGB	colsample_bytree	0,6829
	learning_rate	0,1130
	max_depth	3
	min_child_weight	1
	n_estimators	98
	reg_alpha	1,6612
	reg_lambda	5,3251
	subsample	0,5621
	drop_raw	False
	max_inter_feature_r	0,85
	min_label_r	0,10
HGB	l2_regularization	1,9102
	learning_rate	0,0544
	max_depth	2
	max_iter	53
	max_leaf_nodes	6
	min_samples_leaf	21
	drop_raw	True
	max_inter_feature_r	0,80
	min_label_r	0,10
LGBM	colsample_bytree	0,9489
	learning_rate	0,1006
	max_depth	4
	min_data_in_leaf	15
	n_estimators	51
	num_leaves	13

	reg_alpha	0,0262
	reg_lambda	3,8177
	subsample	0,5890
	drop_raw	False
	max_inter_feature_r	0,75
	min_label_r	0,10

C.3 *Best CV Score* Setiap Model

Model	Best CV Score (Macro F1-score)
XGB	0.7282
HGB	0,7291
LGBM	0,7286

C.4 Hasil Evaluasi Kinerja *Train* dan *Cross-validation*

Model	Mean Train F1 Macro	Mean CV F1 Macro	Gap
XGB	0,8775	0,7282	0,1493
HGB	0,7954	0,7291	0,0663
LGBM	0,8961	0,7286	0,1675

D. Lampiran D. Dokumentasi Hasil pada Data Gabungan

D.1 Evaluasi Awal dengan *Threshold* 0,50

Model	Accuracy	Precision	Recall	Macro F1-score
XGB	0,7973	0,8056	0,7838	0,7973
HGB	0,7297	0,7179	0,7568	0,7295
LGBM	0,7568	0,7568	0,7568	0,7568

D.2 *Confusion Matrix* Awal dalam Bentuk Tabel

Model	TP	FP	FN	TN
XGB	29	7	8	30
HGB	28	11	9	26
LGBM	28	9	9	28

D.3 Hasil Optimasi *Threshold* pada Data Latih

Model	*Threshold* Terpilih	Best OOF Macro F1-score

XGB	0,52	0,7330
HGB	0,39	0,7300
LGBM	0,51	0,7296

D.4 Evaluasi Akhir dengan *Threshold* Terbaik

Model	**Threshold**	**Accuracy**	**Precision**	**Recall**	**Macro F1-score**
XGB	0,52	0,7973	0,8056	0,7838	0,7973
HGB	0,39	0,7568	0,6939	0,9189	0,7502
LGBM	0,51	0,7568	0,7568	0,7568	0,7568

D.5 *Confusion Matrix* Akhir dalam Bentuk Tabel

Model	**TP**	**FP**	**FN**	**TN**
XGB	29	7	8	30
HGB	34	15	3	22
LGBM	28	9	9	28

D.6 Perubahan Performa Setelah Optimasi *Threshold*

Model	**F1 Macro Threshold 0.5**	**F1 Macro Threshold Terbaik**	**Perubahan**	**Model**
XGB	0,7973	0,7973	0,0000	XGB
HGB	0,7295	0,7502	+0,0207	HGB
LGBM	0,7568	0,7568	0,0000	LGBM

D.7 Rekap AUC-ROC Setiap Model

Model	**AUC-ROC**
XGB	0,8145
HGB	0,8108
LGBM	0,7955

D.8 Rekap *Permutation Feature Importance*

Model	**Fitur dengan *Importance* Tertinggi**	**Nilai *Importance***
XGB	sks_mean_1_6	0,1046
	sks6	0,0798
	ip5	0,0704

	delta_ip12	0,0541
	ipk6	0,0446
HGB	sks_mean_1_6	0,1312
	delta_ip12	0,0096
	delta_ip23	0,0094
	ip_max_1_6	0,0015
LGBM	sks6	0,1517
	delta_ip12	0,0421
	ip1	0,0412
	delta_ip23	0,0338
	ip5	0,0326

E. Lampiran E. Tabel Evaluasi pada Skenario Data per Angkatan

E.1 Nilai Metrik Klasifikasi

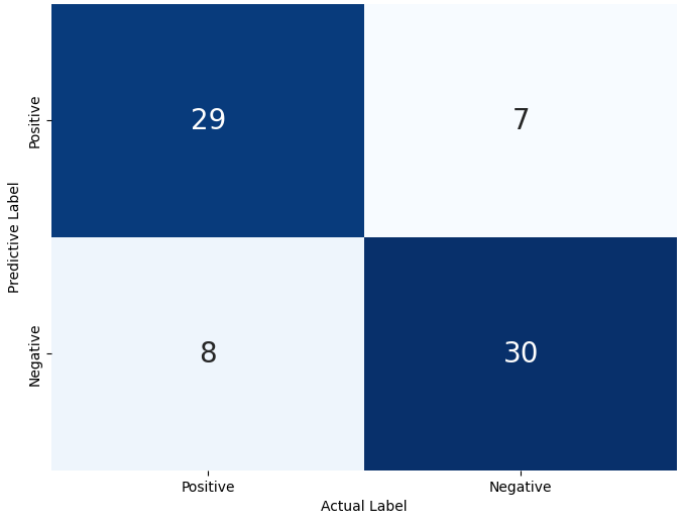
Data Uji	**Accuracy**	**Precision**	**Recall**	**Macro F1-score**
Group 1	0,6993	0,6061	0,7018	0,6933
Group 2	0,4179	0,3697	0,9362	0,3827
Group 3	0,6667	0,8906	0,7037	0,5151
Rata-rata	0,5946	0,6221	0,7805	0,5304
Standar Deviasi	0,1539	0,2608	0,1348	0,1559

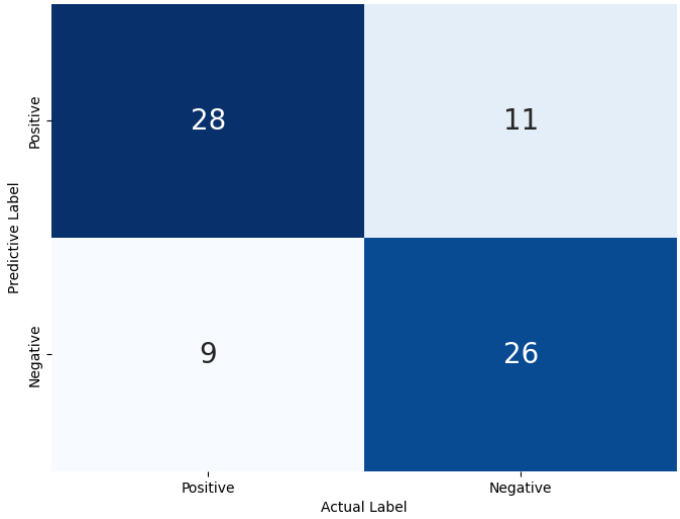
E.2 Nilai *Confusion Matrix*

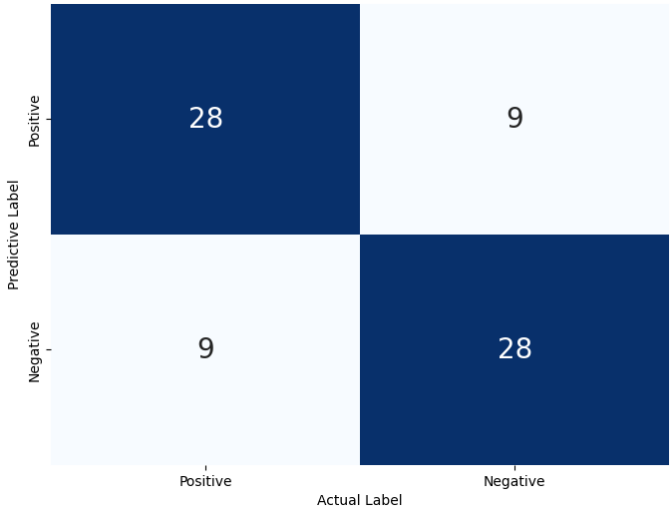
Model	**TP**	**FP**	**FN**	**TN**
XGB	29	7	8	30
HGB	34	15	3	22
LGBM	28	9	9	28

F. Lampiran F. Dokumentasi Visual Evaluasi dan Interpretasi Model

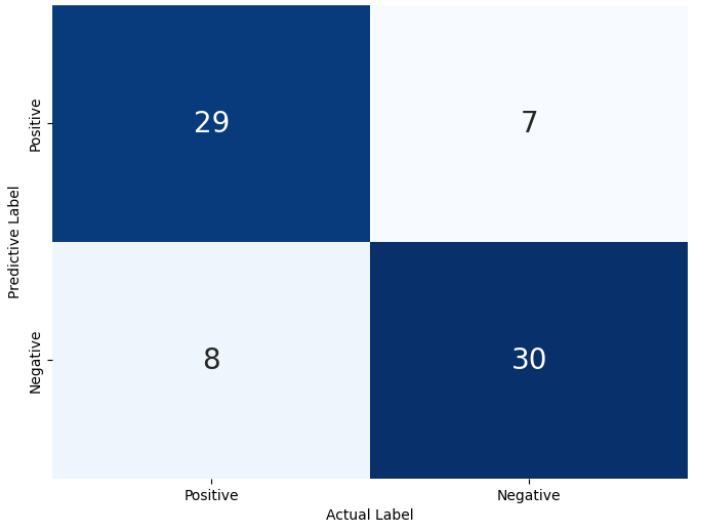
F.1 *Confusion Matrix* Awal pada Skenario Data Gabungan dalam Bentuk *Heatmap*

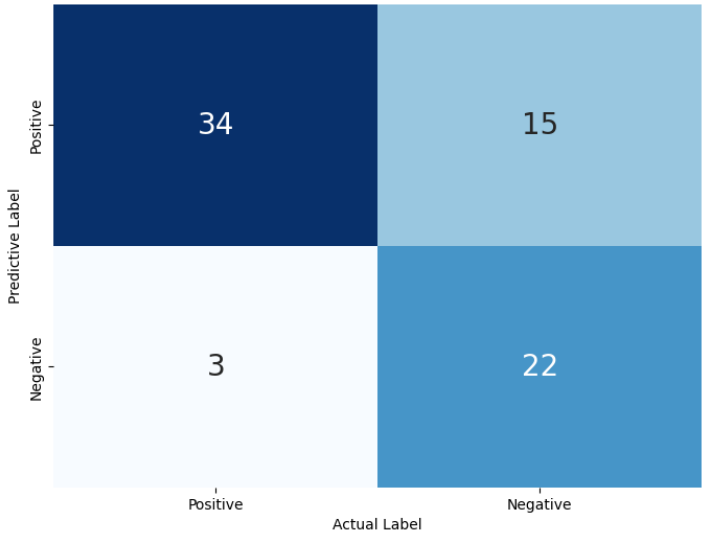


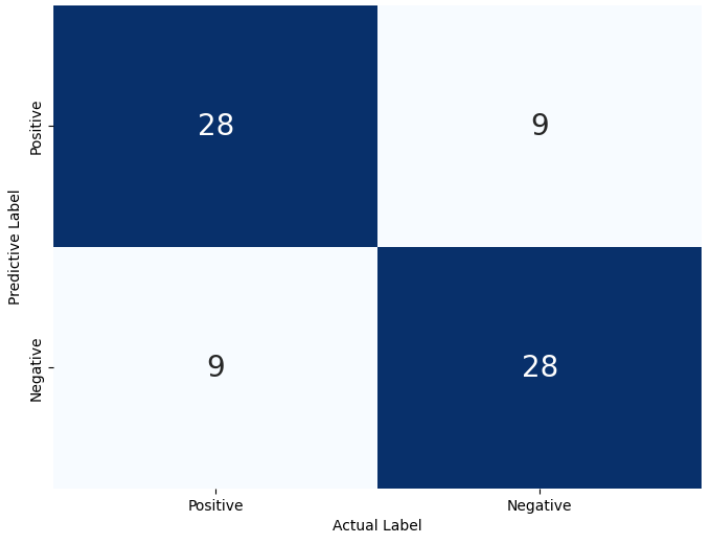




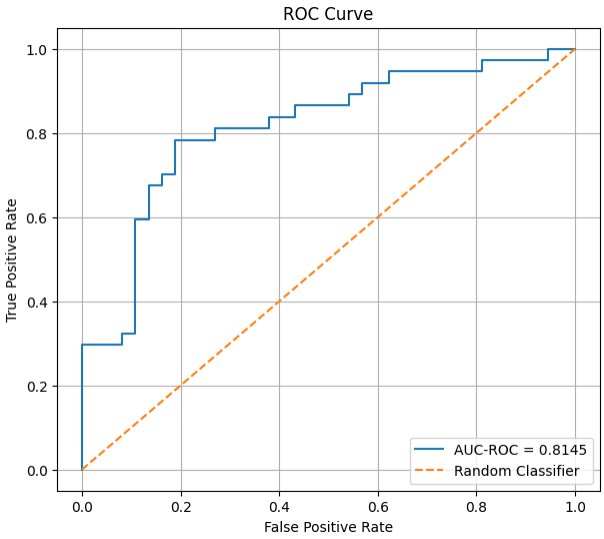
F.2 *Confusion Matrix* Akhir pada Skenario Data Gabungan dalam Bentuk *Heatmap*

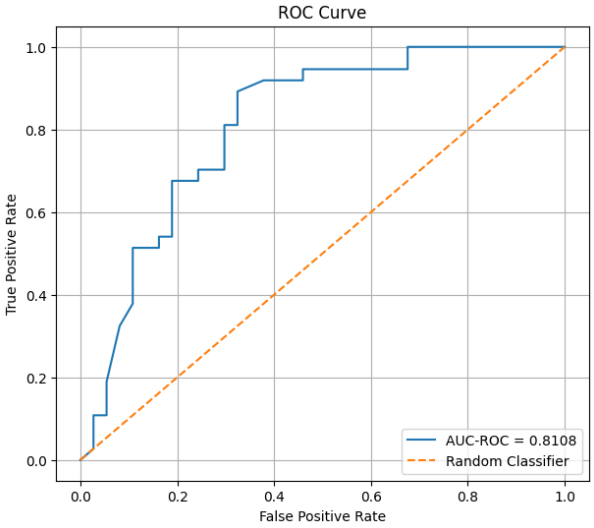


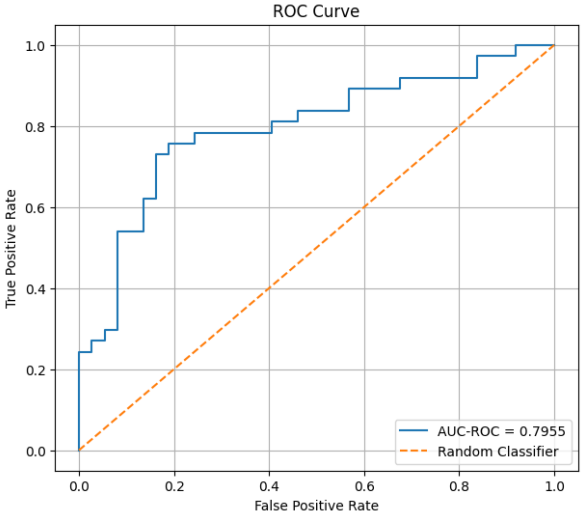




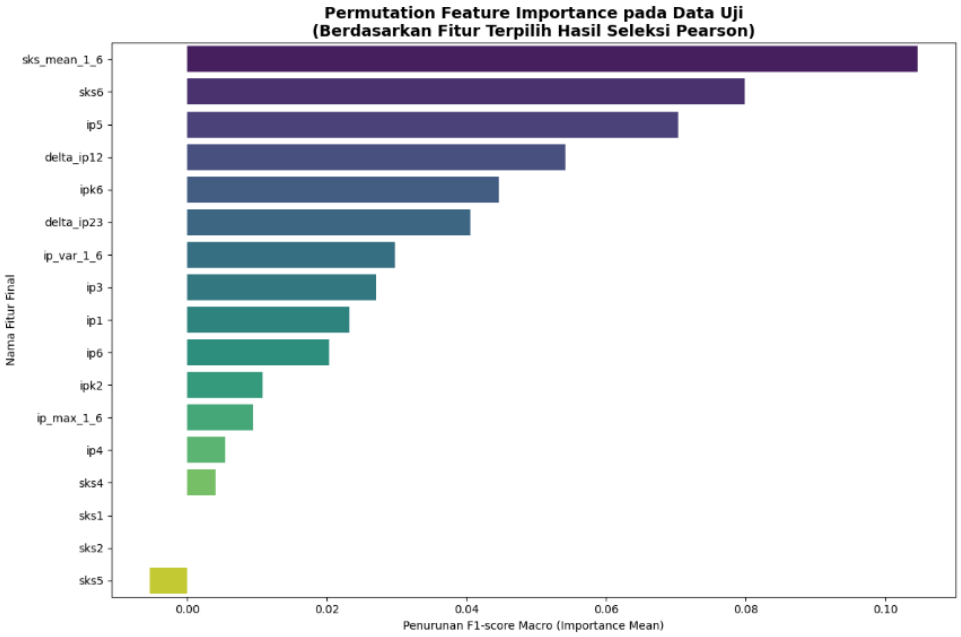
F.3 Kurva ROC pada Skenario Data Gabungan untuk Seluruh Model



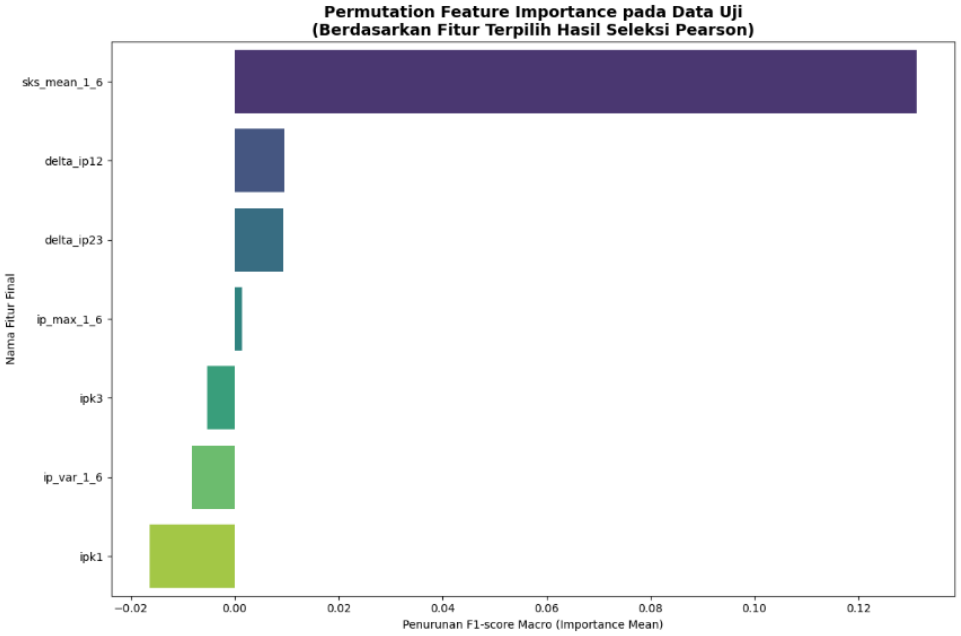




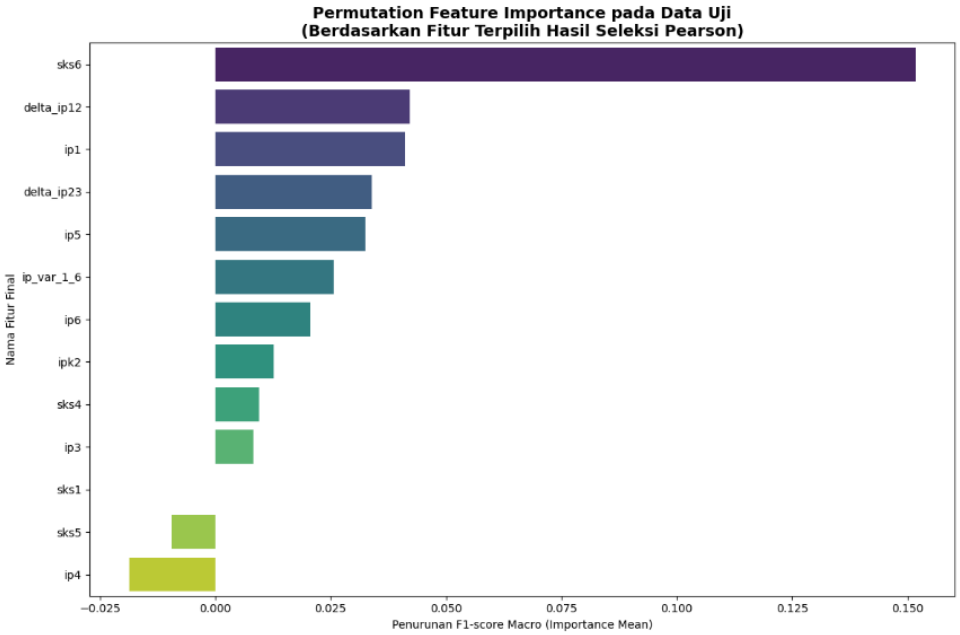
F.4 Grafik *Permutation Feature Importance* XGB pada Skenario Data Gabungan



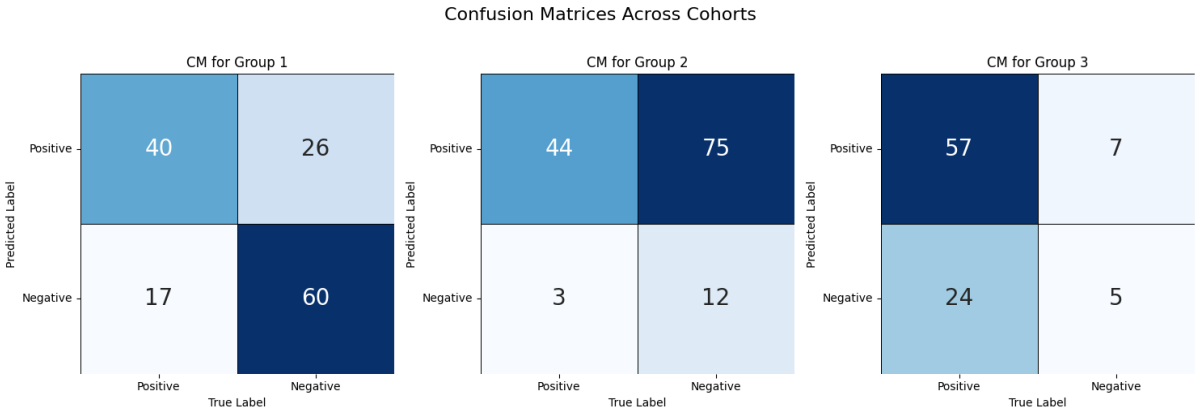
E.5 Grafik *Permutation Feature Importance* HGB pada Skenario Data Gabungan



F.6 Grafik *Permutation Feature Importance* LGBM pada Skenario Data Gabungan



F.7 *Confusion Matrix* XGB pada Skenario Data Berdasarkan Angkatan



G. Lampiran G. Dokumentasi Demonstrasi Model Berbasis Streamlit dan Kode Program

G.1 Struktur File Demonstrasi

```
streamlit_app/
├── app.py
└── model_xgb.pkl
```

G.2 Link Kode Lengkap untuk Skenario Data Gabungan

https://colab.research.google.com/drive/1dkaTun8mCknKVKnA2BQNAkxY6Nz8ZSNj?usp=sharing

G.3 Link Kode Lengkap untuk Skenario Data Berdasarkan Angkatan

https://colab.research.google.com/drive/1zg0CcVB_fDypnO81QnEhMca-ls34GLNa?usp=sharing

G.4 Potongan Kode Inti Prediksi

```
 # Dictionary untuk menyimpan input data mahasiswa
user_inputs = {}

# Buat form input per semester
for sem in range(1, 7):
    st.subheader(f"Semester {sem}")
    col1, col2, col3 = st.columns(3)

    with col1:
        user_inputs[f"ip{sem}"] = st.number_input(
                f"IP Sem {sem}", min_value=0.0, max_value=4.0, value=3.0,
step=0.01, key=f"ip_input_{sem}"
        )
    with col2:
        user_inputs[f"ipk{sem}"] = st.number_input(
               f"IPK Sem {sem}", min_value=0.0, max_value=4.0, value=3.0,
step=0.01, key=f"ipk_input_{sem}"
        )
    with col3:
        user_inputs[f"sks{sem}"] = st.number_input(
             f"SKS Sem {sem}", min_value=0, max_value=24, value=20, step=1,
key=f"sks_input_{sem}"
        )

st.markdown("---")

# ==========================================
# 5. PROSES TOMBOL PREDIKSI
# ==========================================
if st.button("Hitung Prediksi Kelulusan", type="primary"):
    if model is not None:
         # A. Satukan input dasar menjadi dataframe mentah sesuai urutan
pasca-cleaning
        # Urutan kolom: ip1, ipk1, sks1, ip2, ipk2, sks2, dst...
        data_awal_dict = {}
        for sem in range(1, 7):
            data_awal_dict[f"ip{sem}"] = [user_inputs[f"ip{sem}"]]
            data_awal_dict[f"ipk{sem}"] = [user_inputs[f"ipk{sem}"]]
            data_awal_dict[f"sks{sem}"] = [user_inputs[f"sks{sem}"]]

        df_mahasiswa = pd.DataFrame(data_awal_dict)

        # B. LANGSUNG MASUKKAN KE MODEL PIPELINE KAMU
        # (Tidak perlu panggil fe.transform atau df_final lagi karena sudah
otomatis di dalam .pkl)
        try:
            probabilitas = model.predict_proba(df_mahasiswa)

            best_threshold = 0.52
                        prediksi_label = (probabilitas[:, 1] >=
best_threshold).astype(int)
```

```
            # C. Tampilkan Hasil Ke Layar Web
            st.subheader("Hasil Analisis Model XGBoost:")

            prob_terlambat = probabilitas[0][0] * 100
            prob_tepat = probabilitas[0][1] * 100

            if prediksi_label[0] == 1:
                st.success("Status Kelulusan: TEPAT WAKTU")
                  st.info(f"Tingkat keyakinan model: **{prob_tepat:.2f}%**
kemungkinan kamu lulus tepat waktu.")
            else:
                st.error("Status Kelulusan: TERLAMBAT")
               st.info(f"Tingkat keyakinan model: **{prob_terlambat:.2f}%**
kemungkinan kamu lulus terlambat.")

        except Exception as predict_error:
                  st.error(f"Terjadi  kesalahan  saat  memproses  prediksi:
{predict_error}")

    else:
         st.warning("Model belum siap dijalankan. Harap periksa file .pkl
kamu.")
```

G.5 Tampilan Penuh Aplikasi Streamlit



