BAB II

LANDASAN TEORI

2.1 Penelitian Terdahulu

Penelitian terdahulu, atau tinjauan pustaka, memegang peran krusial dalam fondasi sebuah studi ilmiah. Fungsi utamanya adalah untuk memetakan lanskap pengetahuan yang sudah ada, sehingga membantu dalam mengidentifikasi topik-topik yang telah mapan dan area yang masih memerlukan eksplorasi lebih lanjut. Peran ini esensial untuk membangun justifikasi yang kuat atas urgensi dilakukannya sebuah penelitian baru. Dengan menganalisis korpus literatur yang ada, celah penelitian (*research gap*) dapat diidentifikasi secara jelas. Proses ini tidak hanya memperdalam pemahaman konseptual dan teoretis terkait topik yang diteliti, tetapi juga menyediakan landasan metodologis yang kokoh untuk perancangan studi.

Selain itu, tinjauan penelitian terdahulu berfungsi untuk memposisikan studi yang sedang dilakukan secara jelas di dalam spektrum keilmuan yang lebih luas. Hal ini membantu menunjukkan keterkaitan antara studi baru dengan bangunan pengetahuan sebelumnya, baik sebagai pengembangan, pengujian, maupun sanggahan terhadap teori atau temuan yang telah ada. Temuan dan hasil dari studi-studi sebelumnya, termasuk keberhasilan, kegagalan, kelebihan, dan keterbatasan metodologi yang digunakan, turut menjadi bahan pertimbangan kritis. Analisis ini sangat penting untuk menilai orisinalitas dan signifikansi kontribusi yang diharapkan dapat diberikan oleh penelitian ini. Tabel 2.1 berikut merangkum dan menganalisis daftar penelitian terdahulu yang relevan dan mendukung penelitian ini.

Tabel 2.1 Penelitian Terdahulu

Judul Penelitian	Nama Peneliti / Jurnal	Permasalahan	Solusi yang Ditawarkan	Data yang Digunakan	Model yang digunakan	Hasil	Kekurangan
On Developing Generic Models for Predicting Student Outcomes in Educational Data Mining [16]	Gomathy Ramaswami, Teo Susnjak, dan Anuradha Mathrani / Big Data and Cognitive Computing	Model prediksi *course-specific* (spesifik per mata kuliah) tidak skalabel, mahal, dan rentan *overfitting* jika data mata kuliah sedikit.	Mengembangkan model "generik" (*course-agnostic*) yang portabel dan dapat beroperasi di semua mata kuliah.	Data LMS, SMS, dan EMS dari berbagai mata kuliah di institusi pendidikan tinggi Australia.	*CatBoost* (dibandingkan RF, NB, LR, KNN). Interpretasi menggunakan SHAP.	*CatBoost* terbukti terbaik (Akurasi ~75%, AUC 0.87)	Akurasi 75% sehingga memiliki celah untuk pengembangan ataupun peningkatan
Predicting Student Academic Success Using XGBoost and Optuna Tuning Based on Student Attendance [10]	Marcello Roy, Iwan Prasetiawan, Ririn Ikana Desanti, dan Suryasari / Journal of Applied Data Sciences	Kebutuhan akan model prediksi IPK yang akurat. Performa model *boosting* (seperti XGBoost) sangat bergantung pada *hyperparameter* yang sulit diatur.	Mengoptimasi hyperparameter XGBoost secara efisien menggunakan *framework* Optuna (Optimasi Bayesian)	961 data mahasiswa sarjana di Universitas Multimedia Nusantara (UMN)	XGBoost (Regresi).	Model yang dioptimasi Optuna (R^2 0.8456) secara signifikan mengalahkan *baseline* (R^2 0.8068)	Model regresi (prediksi IPK), bukan klasifikasi (status kelulusan). Fitur yang digunakan adalah agregat sederhana, bukan rekayasa fitur dinamis.
Multiclass Prediction Model for Student Grade Prediction Using	Siti Dianah Abdul Bujang, Ali Selamat, Roliana Ibrahim, Ondrej Krejcar, Enrique	Tantangan dalam menangani *dataset multiclass* (5 kelas) yang tidak seimbang, yang menyebabkan *overfitting*	Mengusulkan model SFS (SMOTE + *Feature Selection*). Data diseimbangkan menggunakan SMOTE,	1282 data nilai akhir mahasiswa (5 kelas: *Exceptional*, *Excellent*,	*Random Forest* (RF), J48, SVM, NB, kNN, LR	Kombinasi RF + SMOTE + Feature *Selection* (SFS-1) memberikan	Hasil 99.5% sangat tinggi dan berisiko *overfitting*. Tidak

Judul Penelitian	Nama Peneliti / Jurnal	Permasalahan	Solusi yang Ditawarkan	Data yang Digunakan	Model yang digunakan	Hasil	Kekurangan
Machine Learning [17]	Herrera-Viedma, dan Hamido Fujita / IEEE Access	dan misklasifikasi kelas minoritas (misal: '*Fail*' hanya 21 dari 1282 data).	kemudian fitur diseleksi (*Wrapper*/*Filter*).	*Distinction*, *Pass*, *Fail*)		hasil tertinggi (F-*measure* 99.5%).	menggunakan *CatBoost*/*XGBoost*.
Prediction of Students' Academic Performance... Using Long Short-Term Memory (LSTM) [18]	Luis Vives, Ivan Cabezas, Juan Carlos Vives, Nilton German Reyes, Janet Aquino, Jose Bautista Cóndor, Dan S. Francisco Segura Altamirano / IEEE Access	Tingkat *dropout* tinggi pada mata kuliah "*Programming Fundamentals*". Tantangan teknis utama adalah data tidak seimbang yang menyebabkan *overfitting*.	Menggunakan model *Deep Learning* LSTM. Membandingkan dua teknik *oversampling*: SMOTE dan *Generative Adversarial Networks* (GAN).	661 data mahasiswa (Lulus/Gagal) dari 2 universitas di Peru.	LSTM, DNN, DT, RF, LR, SVC, KNN.	LSTM + GAN (Akurasi 98.3%) mengungguli LSTM + SMOTE dan model lain	Model ML tradisional (DT, RF, dll.) berkinerja sangat buruk (<35%) saat dilatih dengan data GAN. *Training* GAN sangat lama.
Efficient hyperparameter tuning for predicting student performance with	Saleh Albahli	(1) Data tidak seimbang; (2) Proses *hyperparameter tuning* (*Grid*/*Random Search*)	(1) Menggunakan SMOTE untuk data tidak seimbang; (2) Menggunakan *Bayesian Optimization*	5000 data mahasiswa di Arab Saudi	*Random Forest* dan *Decision Tree*.	Decision Tree + SMOTE + *Bayesian Optimization*	Akurasi 100% sangat mungkin *overfitting*. Fokus pada regresi (GPA), bukan klasifikasi.

Judul Penelitian	Nama Peneliti / Jurnal	Permasalahan	Solusi yang Ditawarkan	Data yang Digunakan	Model yang digunakan	Hasil	Kekurangan
Bayesian optimization [19]	Multimedia Tools and Applications	sangat lambat dan tidak efisien.	untuk tuning yang efisien.	(memprediksi CUMGPA)		mencapai akurasi 100%	
Machine learning analysis of factors affecting college students' academic performance [20]	Jingzhao Lu, Yaju Liu, Shuo Liu, Zhuo Yan, Xiaoyu Zhao, Yi Zhang, Chongran Yang, Haoxin Zhang, Wei Su dan Peihong Zhao Frontiers in Psychology	Perlunya mengeksplorasi faktor non-akademik (metakognitif, motivasi, kesehatan mental) sebagai prediktor kinerja akademik.	Menggunakan *Chi-Square Test* untuk seleksi fitur dari data kuesioner. Membandingkan 4 model (LOG, SVC, RFC, XGBoost).	1101 data kuesioner mahasiswa (Lulus/Gagal) di Hebei Agricultural University	LOG, SVC, RFC, XGBoost.	XGBoost model terbaik (Akurasi 86.52%, Recall 89.22%)	*Dataset* relatif kecil dan spesifik pada satu universitas
Prediksi kelulusan siswa sekolah menengah pertama menggunakan machine learning [11]	Agusti Frananda Alfonsus Naibaho dan Amalia Zahra Jurnal Informatika dan Teknik Elektro Terapan	Adanya siswa yang lulus tidak tepat waktu. Tantangan utama: data sangat tidak seimbang (523 Lulus Tepat Waktu vs. 26 Terlambat)	Menggunakan *Random Over Sampling* (duplikasi data) untuk menyeimbangkan data. Membandingkan 3 model.	549 data siswa SMPN 1 Lubuk Alung (data pribadi, akademik, data orang tua).	*Decision Tree*, *Random Forest*, dan XGBoost.	*Random Forest* + *Over Sampling* mencapai akurasi 99.5%	Metodologi *oversampling* yang lemah (hanya duplikasi data), yang diakui peneliti menyebabkan *overfitting* dan akurasi yang ambigu .

Judul Penelitian	Nama Peneliti / Jurnal	Permasalahan	Solusi yang Ditawarkan	Data yang Digunakan	Model yang digunakan	Hasil	Kekurangan
Prediction of Student academic Perfomance Using a Hybrid 2D CNN Model [21]	Sujan Poudyal, Mahnas J. Mohammadi-Aragh, dan John E. Ball / electronics	Jarangnya penggunaan *Deep Learning* (CNN) untuk memprediksi kinerja akademik, karena data pendidikan biasanya berbentuk 1D (numerik), bukan 2D (gambar).	Transformasi data 1D ke gambar 2D (matriks 8x5 piksel). Menggunakan model *Hybrid* 2D CNN untuk klasifikasi.	OULAD (Open University Learning Analytics Dataset) . 32.593 mahasiswa	Hybrid 2D CNN (dibandingkan KNN, NB, DT, LR, ANN).	Model *Hybrid* 2D CNN (Akurasi 88%) mengalahkan semua model *baseline* (misal: LR 86.05%)	Tidak ada interpretasi fitur (bersifat *black box*). Hanya menggunakan akurasi sebagai metrik evaluasi
Analysis and Prediction of Influencing Factors of College Student Achievement Based on Machine Learning [22]	Dongxuan Wang, Dapeng Lian, Yazhou Xing, Shiying Dong, Xinyu Sun, dan Jia Yu / Frontiers in Psychology	Hasil penelitian dari universitas top mungkin tidak berlaku di "universitas biasa Tiongkok" (*ordinary Chinese colleges*).	Mengidentifikasi faktor-faktor dari kuesioner menggunakan *Chi-Square Test*. Membandingkan 4 model ML.	292 mahasiswa baru *Computer Science* (Lulus/Gagal)	LOG, SVC, RFC, NBC.	SVC (*Support Vector Classifier*) terbukti terbaik dan paling stabil (Akurasi 86.18%, Recall 82.83%)	Dataset sangat kecil (292). Hasil sangat spesifik dan tidak dapat digeneralisasi
Prediksi Kelulusan Mahasiswa Menggunakan Data mining Algoritma K-means [23]	Ray Mondow Sagala / TelKa Jurnal Teknologi Informasi dan Komunikasi	Kelulusan mata kuliah prasyarat tidak dapat diketahui sebelum akhir semester, yang dapat menghambat kelulusan	Menggunakan *clustering* K-Means (*k* =3) untuk mengelompokkan mahasiswa (Tidak Lulus, Cukup, Baik).	118 data mahasiswa (nilai tugas, *unit test*, *mid test*, kehadiran, status tinggal, dan sebagainya)	*K-Means* dan *Chi-Square Attribute Selection*	Akurasi 93%, Presisi 96%, Recall 92%.	Dataset sangat kecil (118). K-Means sangat bergantung pada penentuan *centroid* awal yang acak.

Sebuah penelitian menangani permasalahan model prediksi yang spesifik per mata kuliah, yang dinilai tidak skalabel dan rentan *overfitting* saat data historis mata kuliah tersebut sedikit. Solusi yang ditawarkan adalah pengembangan model "generik" (*course-agnostic*) yang portabel dan dapat beroperasi di berbagai mata kuliah. Dengan menggunakan data LMS, SMS, dan EMS, model CatBoost diterapkan dan diinterpretasi menggunakan SHAP. Hasilnya menunjukkan bahwa CatBoost merupakan model terbaik (AUC 0.87) dibandingkan algoritma lain. Meskipun demikian, akurasi 75% yang dicapai dicatat sebagai keterbatasan, walaupun terdapat justifikasi mengapa angka tersebut wajar [16].

Penelitian lain [2] berfokus pada kebutuhan untuk memprediksi IPK secara akurat, dengan permasalahan bahwa performa model *boosting* sangat bergantung pada *hyperparameter* yang sulit diatur. Solusi yang diajukan adalah mengoptimasi *hyperparameter* XGBoost secara efisien menggunakan *framework* Optuna. Studi ini menggunakan 961 data mahasiswa sarjana di Universitas Multimedia Nusantara (UMN). Hasil evaluasi menunjukkan bahwa model yang dioptimasi Optuna (R^2 0.8456) secara signifikan mengalahkan *baseline* (R^2 0.8068). Keterbatasan studi ini adalah fokusnya pada model regresi (prediksi IPK), bukan klasifikasi status kelulusan, serta penggunaan fitur agregat yang sederhana [10].

Studi berikutnya secara spesifik mengatasi tantangan dalam menangani *dataset multiclass* (5 kelas) yang tidak seimbang yang menyebabkan misklasifikasi pada kelas minoritas (misalnya, kelas *'Fail'* hanya 21 dari 1282 data). Solusi yang diusulkan adalah model SFS, yang merupakan gabungan dari teknik *oversampling* SMOTE dengan *Feature Selection* (Seleksi Fitur). Model ini diuji pada 1282 data nilai akhir mahasiswa. Hasilnya menunjukkan bahwa kombinasi *Random Forest* (RF) dengan SFS memberikan F-*measure* tertinggi mencapai 99.5%. Keterbatasan utama dari hasil ini adalah F-*measure* 99.5% yang sangat tinggi, yang mengindikasikan adanya potensi *overfitting* [17].

Sebuah studi difokuskan pada tingginya tingkat *dropout* pada mata kuliah "*Programming Fundamentals*", dengan tantangan teknis utama berupa data yang tidak seimbang. Solusi yang diuji adalah model *Deep Learning* LSTM yang

dikombinasikan dengan dua teknik *oversampling* yaitu SMOTE dan *Generative Adversarial Networks* (GAN). Berdasarkan 661 data mahasiswa (Lulus/Gagal), ditemukan bahwa kombinasi LSTM + GAN memberikan akurasi tertinggi (98.3%). Kelemahannya, data sintetis yang dihasilkan GAN ternyata tidak cocok untuk model ML tradisional (seperti DT dan RF) yang kinerjanya turun drastis di bawah 35% saat menggunakan data tersebut [18].

Penelitian menangani dua permasalahan sekaligus yaitu data tidak seimbang dan proses *hyperparameter tuning* yang lambat. Solusi yang ditawarkan adalah penggunaan SMOTE untuk data tidak seimbang dan *Bayesian Optimization* untuk *tuning* yang efisien, menggantikan *Grid Search*. Pada 5000 data mahasiswa, kombinasi *Decision Tree*, SMOTE, dan *Bayesian Optimization* berhasil mencapai akurasi 100%. Keterbatasan yang jelas dari hasil ini adalah akurasi 100% yang sangat mungkin merupakan indikasi *overfitting*, serta fokus penelitian pada regresi (prediksi CUMGPA), bukan klasifikasi [19].

Penelitian mengeksplorasi faktor non-akademik (seperti kesadaran metakognitif, motivasi, dan kesehatan mental) sebagai prediktor kinerja. Solusi yang digunakan adalah *Chi-Square Test* untuk seleksi fitur dari data kuesioner, yang kemudian diumpankan ke empat model ML, termasuk *XGBoost*. Pada 1101 data kuesioner, *XGBoost* terbukti menjadi model terbaik dengan akurasi 86.52%. Keterbatasan penelitian ini adalah penggunaan dataset yang relatif kecil dan spesifik dari satu universitas [20].

Sebuah studi menghadapi permasalahan data yang sangat tidak seimbang (rasio 523:26) dalam memprediksi kelulusan siswa SMP. Solusi yang digunakan adalah metode *Random Over Sampling*, yang pada dasarnya hanya menduplikasi data kelas minoritas. Meskipun *Random Forest* yang dikombinasikan dengan metode ini mencapai akurasi 99.5%, penelitian ini memiliki kelemahan metodologis yang signifikan. Penggunaan duplikasi data tersebut diakui menyebabkan *overfitting* parah dan menghasilkan akurasi yang ambigu [11].

Sebuah penelitian mengatasi jarangnya penggunaan *Deep Learning* (CNN) untuk data pendidikan 1D. Solusinya adalah pendekatan inovatif dengan

mentransformasi data 1D numerik menjadi representasi gambar 2D (matriks 8x5 piksel) yang kemudian diklasifikasikan menggunakan model *Hybrid* 2D CNN. Pada dataset OULAD (32.593 mahasiswa), model CNN ini (akurasi 88%) berhasil mengalahkan model *baseline*. Keterbatasan utamanya adalah model ini bersifat *black box* (tidak ada interpretasi fitur) dan evaluasi hanya menggunakan metrik akurasi [21].

Penelitian didasarkan pada permasalahan bahwa temuan dari universitas top mungkin tidak berlaku di "universitas biasa Tiongkok". Menggunakan *Chi-Square Test* untuk seleksi fitur kuesioner dan membandingkan empat model, ditemukan bahwa SVC (*Support Vector Classifier*) adalah model terbaik dan paling stabil (akurasi 86.18%). Keterbatasan utama dari penelitian ini adalah ukuran dataset yang sangat kecil (292 mahasiswa), sehingga hasilnya sangat spesifik dan tidak dapat digeneralisasi [22].

Studi bertujuan memprediksi kegagalan mata kuliah prasyarat sebelum akhir semester. Solusi yang digunakan adalah *clustering K-Means* (k = 3) untuk mengelompokkan mahasiswa (Tidak Lulus, Cukup, Baik). Pada 118 data mahasiswa, model ini mencapai akurasi 93%. Keterbatasan penelitian ini terletak pada dataset yang sangat kecil (118) dan penggunaan K-Means yang hasilnya dapat bervariasi karena sensitif terhadap penentuan *centroid* awal yang acak [23].

Berdasarkan kesenjangan tersebut, penelitian ini dirancang untuk menjawab tantangan tersebut secara komprehensif. Untuk menjawab kebutuhan akan model yang kuat pada dataset kecil, algoritma CatBoost dipilih. CatBoost sebagai bagian dari keluarga *gradient boosting*, dikenal tangguh dalam menangani data kategorikal secara *native* dan akan dioptimalkan melalui proses *hyperparameter tuning*, sebuah langkah yang terbukti krusial. Untuk mengatasi masalah fundamental data *multiclass* yang tidak seimbang, teknik *oversampling* SMOTE akan diterapkan. Variasi SMOTETomek juga akan diuji (khususnya pada model Semester 6) untuk membersihkan *noise* pasca-*sampling*. Terakhir, untuk mengisi kesenjangan terbesar, yaitu kurangnya interpretabilitas , penelitian ini akan menerapkan SHAP (*SHapley Additive exPlanations*). Penggunaan SHAP bertujuan

untuk "membuka kotak hitam" model CatBoost, sehingga faktor-faktor yang mendorong prediksi kelulusan dapat diidentifikasi dan dijelaskan.

2.2 Teori yang berkaitan

Berikut adalah teori yang dikemukakan dalam penelitian ini. Teori tersebut yang menjadi acuan dalam menggali struktur yang lebih mendalam dalam hal yang akan dibahas.

2.2.1 Faktor-Faktor Kinerja Akademik

Kinerja akademik merupakan hasil dari proses pendidikan yang kompleks dan multifaset. Berbagai literatur di bidang psikologi pendidikan mengidentifikasi bahwa keberhasilan mahasiswa dipengaruhi oleh beragam variabel [24]. Variabel-variabel ini secara umum dapat diklasifikasikan menjadi dua kategori besar yaitu faktor internal dan faktor eksternal [25].

Faktor internal adalah atribut-atribut yang berasal dari dalam diri mahasiswa, yang mencakup aspek psikologis, kognitif, dan perilaku [26]. Beberapa faktor internal yang paling sering diteliti meliputi motivasi belajar, kebiasaan belajar (*learning behaviors*), kemampuan manajemen waktu, status kesehatan mental, dan kesadaran metakognitif [27]. Faktor-faktor ini secara langsung menentukan bagaimana seorang mahasiswa memandang proses belajar, mengelola sumber daya, dan merespons tantangan akademik [28].

Faktor eksternal mencakup elemen-elemen dari lingkungan yang memengaruhi mahasiswa. Ini termasuk situasi keluarga (seperti tingkat pendidikan atau pekerjaan orang tua), kondisi sosioekonomi secara umum, dukungan sosial dari rekan sebaya, dan akses terhadap sumber belajar yang berkualitas [29]. Selain faktor-faktor tersebut, riwayat akademik historis (seperti Indeks Prestasi Kumulatif/IPK dan nilai-nilai sebelumnya) seringkali menjadi prediktor tunggal terkuat untuk kinerja di masa depan. Penelitian ini memanfaatkan data transkrip dan demografis sebagai proksi untuk merepresentasikan faktor-faktor historis dan eksternal tersebut [12].

2.2.2 Motivasi Belajar dan Keterlibatan Mahasiswa

Di antara beragam faktor internal, motivasi belajar dianggap sebagai salah satu pendorong fundamental [30]. Motivasi didefinisikan sebagai kekuatan internal yang membangkitkan, mengarahkan, dan memelihara perilaku yang berorientasi pada tujuan, dalam hal ini tujuan akademik [31]. Motivasi yang tinggi seringkali dipicu oleh ketertarikan terhadap mata kuliah atau tujuan karir yang jelas, menjadi sumber energi psikologis yang memungkinkan mahasiswa untuk bertahan dalam menghadapi kesulitan dan materi yang kompleks [32].

Keterlibatan (*engagement*) mahasiswa adalah manifestasi perilaku dari motivasi internal tersebut. Keterlibatan adalah sebuah konsep multidimensional yang mencakup keterlibatan perilaku (misalnya kehadiran, partisipasi di kelas), keterlibatan emosional (misalnya antusiasme), dan keterlibatan kognitif (misalnya pengerahan usaha mental) [33]. Dalam lingkungan pembelajaran modern, keterlibatan ini sering diukur melalui data proksi, seperti frekuensi akses ke *Learning Management Systems* (LMS), jumlah interaksi dalam forum diskusi, dan ketepatan waktu serta kualitas pengumpulan tugas [34].

Dalam penelitian ini, data transkrip mentah digunakan untuk merekayasa fitur-fitur yang secara tidak langsung mencerminkan keterlibatan dan persistensi. Fitur-fitur seperti jumlah mata kuliah yang diulang, jumlah SKS yang gagal, atau tren perolehan IPK per semester (dikenal sebagai *slope*) dapat menjadi indikator kuat dari tingkat keterlibatan atau demotivasi mahasiswa sepanjang perjalanan studi mereka.

2.2.3 Sistem Peringatan Dini (*Early Warning Systems* - EWS) dalam Pendidikan

Tujuan praktis utama dari pemodelan prediksi di bidang pendidikan adalah untuk membangun sebuah Sistem Peringatan Dini (*Early Warning System* - EWS). EWS adalah sebuah kerangka kerja proaktif, bukan reaktif, yang menggunakan analisis data untuk mengidentifikasi mahasiswa yang

menunjukkan pola-pola yang terkait dengan risiko kegagalan akademik atau *drop out* [35]. Identifikasi ini idealnya dilakukan pada tahap sedini mungkin dalam siklus studi mahasiswa untuk memaksimalkan peluang keberhasilan intervensi [1].

Fungsi EWS tidak berhenti pada identifikasi risiko. Tujuan akhirnya adalah untuk menyediakan informasi yang dapat ditindaklanjuti (*actionable insights*) kepada para pemangku kepentingan akademik, seperti dosen pembimbing akademik dan manajemen program studi. Dengan peringatan dini, institusi dapat menerapkan intervensi yang ditargetkan dan terpersonalisasi, misalnya berupa bimbingan akademik tambahan, konseling, atau lokakarya keterampilan belajar. Intervensi ini dirancang untuk membantu mahasiswa mengatasi kesulitan mereka sebelum masalah tersebut menjadi tidak dapat diperbaiki.

Penelitian ini berkontribusi langsung pada pengembangan EWS dengan merancang tiga model terpisah untuk titik waktu yang berbeda (Semester 2, 4, dan 6). Pendekatan multi-model ini selaras dengan filosofi EWS, yang memungkinkan identifikasi risiko pada tahap yang sangat dini (Semester 2) serta evaluasi yang lebih komprehensif pada tahap pertengahan dan akhir studi (Semester 4 dan 6). Hal ini memungkinkan institusi untuk menerapkan strategi intervensi yang berbeda-beda sesuai dengan tingkat urgensi dan tahap studi mahasiswa.

2.3 *Framework* dan Algoritma

Bagian ini membahas kerangka dan algoritma yang digunakan dalam penelitian untuk memprediksi performa akademik mahasiswa. Setiap pendekatan dipilih berdasarkan relevansinya dengan karakteristik data yang tersedia, terutama ukuran *dataset* yang terbatas, ketidakseimbangan antar kelas, serta kebutuhan akan interpretasi yang kuat untuk mendukung pengambilan keputusan. Pembahasan dimulai dengan penjelasan mengenai konsep dasar yang menjadi landasan metode *machine learning* yang digunakan, kemudian dilanjutkan dengan uraian algoritma

yang menjadi inti pemodelan, termasuk kelebihan dan peranannya dalam konteks sistem peringatan dini di lingkungan pendidikan tinggi.

2.3.1 Educational Data Mining (EDM)

Penelitian ini secara metodologis berada dalam domain *Educational Data Mining* (EDM). EDM didefinisikan sebagai bidang interdisipliner yang berfokus pada pengembangan dan penerapan metode komputasi (termasuk *machine learning*, statistika, dan *data mining*) untuk mengeksplorasi dan menganalisis data unik yang berasal dari lingkungan Pendidikan [36]. Tujuan utamanya adalah untuk menemukan pola-pola tersembunyi dan mengekstrak pengetahuan berharga dari data akademik mentah [37].

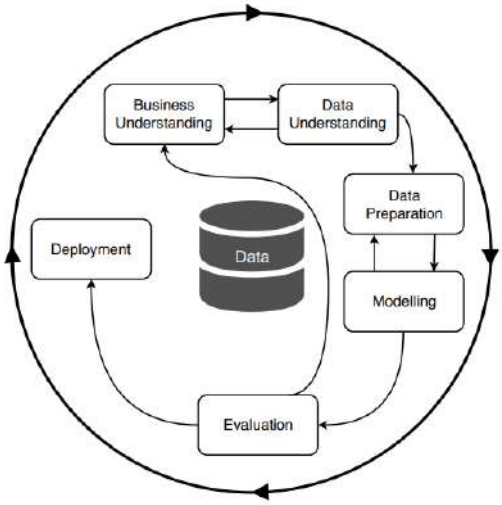
Aplikasi EDM sangat beragam, mencakup pemodelan perilaku mahasiswa, analisis efektivitas kurikulum, dan pengembangan sistem bimbingan adaptif [38],[39],[40]. Salah satu area aplikasi yang paling umum dan berdampak tinggi adalah prediksi kinerja akademik. Dalam hal ini, EDM digunakan untuk membangun model yang dapat memprediksi keberhasilan atau kegagalan mahasiswa, yang menjadi dasar bagi EWS.

Penelitian ini merupakan penerapan klasik dari EDM. Dengan menggunakan data akademik (transkrip) dan data demografis mahasiswa sebagai input, penelitian ini bertujuan untuk memecahkan masalah prediksi kinerja akademik. Hasil dari model EDM ini dirancang untuk mendukung pengambilan keputusan berbasis data di tingkat institusional, yang sejalan dengan tujuan utama dari bidang EDM itu sendiri.

2.3.2 *Cross-Industry Standard Process for Data Mining* (CRISP-DM)

CRISP-DM (*Cross-Industry Standard Process for Data Mining*) merupakan *framework* paling popular per 2024 yang digunakan untuk mengarahkan proses analisis data secara sistematis [41]. *Framework* ini membagi proses menjadi enam tahap utama, yaitu *business understanding*, *data understanding*, *data preparation*, *modeling*, *evaluation*, dan *deployment*. Keenam tahap tersebut bersifat iteratif, artinya peneliti dapat kembali ke tahap sebelumnya untuk melakukan perbaikan bila ditemukan kekurangan pada

tahap selanjutnya [42]. Pendekatan ini memastikan setiap langkah memiliki keterkaitan yang kuat, sehingga hasil akhir tidak hanya fokus pada aspek teknis, tetapi juga selaras dengan tujuan penelitian. Dalam pendidikan, CRISP-DM memberikan kerangka kerja yang jelas untuk mengelola proses prediksi kinerja akademik mahasiswa, mulai dari memahami masalah pendidikan yang dihadapi hingga penerapan hasil prediksi dalam pengambilan Keputusan [43].



Gambar 2.1 Alur CRISP-DM [44]

Kelebihan utama CRISP-DM adalah fleksibilitasnya untuk diterapkan di berbagai bidang, termasuk pendidikan, bisnis, dan industri. Struktur tahapan yang jelas memudahkan peneliti atau praktisi dalam mengatur alur kerja, meminimalkan kesalahan, serta meningkatkan replikasi penelitian seperti pada Gambar 2.1 [45]. Selain itu, sifat iteratifnya memungkinkan evaluasi dan penyempurnaan model secara berulang sehingga kualitas prediksi dapat meningkat. Namun, CRISP-DM juga memiliki keterbatasan. *Framework* ini bersifat umum sehingga memerlukan penyesuaian khusus agar benar-benar sesuai dengan karakteristik data dan domain penelitian [46]. Tahap *data preparation* yang menjadi salah satu bagian terpenting sering kali memakan waktu lebih lama dibanding tahap lainnya, terutama ketika data yang digunakan bersifat heterogen, tidak terstruktur, atau memiliki kualitas rendah. Hal ini menuntut peneliti untuk memiliki keterampilan teknis yang baik serta pemahaman mendalam terhadap data yang dianalisis [47].

2.3.3 Klasifikasi dalam Supervised Learning

Klasifikasi merupakan salah satu tugas utama dalam *supervised learning*, yaitu pendekatan pemodelan yang memanfaatkan pasangan data fitur dan label untuk mempelajari fungsi pemetaan dari ruang fitur ke ruang kelas [48]. Tujuan pembelajaran adalah memperoleh hipotesis yang mampu melakukan generalisasi yang baik terhadap sampel yang tidak terlihat pada saat pelatihan [49]. Proses ini biasanya diformulasikan sebagai minimisasi fungsi rugi yang mengukur perbedaan antara label sebenarnya dan prediksi model, dengan penambahan regularisasi untuk mengendalikan kompleksitas agar tidak terjadi *overfitting* [50]. Pada tataran teoritis, proses pembelajaran dapat dilihat sebagai upaya mendekati batas keputusan optimal yang memisahkan distribusi kelas di ruang fitur.

Algoritma klasifikasi modern mencakup keluarga model linear dan non-linear. Model linear seperti *Logistic Regression* memproyeksikan fitur ke dalam peluang melalui fungsi logit, sementara *Support Vector Machine* membangun margin maksimum antara kelas dengan meminimalkan kehilangan *hinge* [51],[52]. Model berbasis pohon keputusan mempartisi ruang fitur secara rekursif untuk memaksimalkan kemurnian *node* yang diukur menggunakan *impurity* seperti *Gini* atau *entropy* [51]. Teknik *unsemble* seperti *bagging* dan *boosting* menggabungkan banyak model lemah menjadi model kuat yang lebih stabil terhadap variasi data [53]. Di luar itu, jaringan saraf melakukan komposisi non-linear berlapis dengan optimisasi berbasis *gradient*, walau pada data tabular sering kali *unsemble* berbasis pohon memberikan performa kompetitif [54].

Formulasi target pada klasifikasi dapat berupa biner, multikelas, atau multilabel [55],[56],[57]. Klasifikasi biner memisahkan dua kelas dan sering memakai metrik seperti akurasi, *precision*, *recall*, dan *F1-score* [58]. Klasifikasi multikelas memperumumkan ke lebih dari dua kelas dan memerlukan skema penggabungan metrik seperti mikro, makro, serta *weighted averaging* agar penilaian tidak bias terhadap kelas mayoritas [55]. Klasifikasi multilabel memperkenankan satu sampel memiliki lebih dari satu label yang

benar, sehingga evaluasi berbasis *subset accuracy*, *hamming loss*, dan *macro* F1 menjadi relevan [57]. Pada banyak skenario pendidikan tinggi, multikelas lazim digunakan untuk memetakan status kelulusan yang bersifat saling eksklusif.

Pemilihan fungsi rugi berpengaruh langsung terhadap sifat solusi. *Cross entropy* mendorong kalibrasi probabilitas yang baik dan memfasilitasi interpretasi skor sebagai peluang [59]. *Hinge loss* menekankan margin sehingga kuat terhadap *outlier* tertentu, tetapi tidak menghasilkan probabilitas secara langsung [55]. Pada implementasi *unsemble* berbasis pohon, varian *log loss* dan *deviance* umum dipakai karena stabil dalam optimisasi *gradient boosting* [60]. Regularisasi L1 mendorong sparsitas koefisien dan membantu seleksi fitur, sedangkan L2 menekan besaran koefisien sehingga mengurangi variansi. *Early stopping* menjadi strategi efektif untuk mencegah *overfitting* pada model beriterasi seperti *boosting* dan jaringan saraf [61].

Paradigma evaluasi memerlukan pembagian data yang ketat antara pelatihan, validasi, dan pengujian. K-*fold cross validation* memberikan estimasi performa yang lebih dapat diandalkan pada ukuran sampel terbatas karena setiap sampel bergiliran menjadi bagian data uji [62]. Untuk data yang tidak seimbang, *stratified* k-*fold* menjaga proporsi kelas pada setiap lipatan sehingga variabilitas estimasi berkurang [58]. Di luar metrik klasikal, kurva ROC dan area di bawah kurva ROC memberikan ringkasan *trade-off* antara *true positive rate* dan *false positive rate* [63]. Pada ketidakseimbangan kelas yang berat, *precision–recall curve* sering lebih informatif karena mengabaikan prediksi benar pada kelas negatif yang dominan. Kalibrasi probabilitas melalui *isotonic regression* atau *Platt scaling* penting ketika keputusan kebijakan bergantung pada ambang peluang, misalnya saat menetapkan ambang risiko pada sistem peringatan dini [58].

Kualitas data menentukan batas atas kinerja model. Kebersihan data, konsistensi skala, pengkodean fitur kategorikal, dan penanganan nilai hilang harus ditata dalam *pipeline* yang *reproducible* [55]. *Feature engineering*

mengekstraksi sinyal prediktif dari data mentah melalui konstruksi agregasi, tren, interaksi, serta transformasi non-linear. Data leakage perlu dihindari dengan memastikan informasi masa depan tidak bocor ke tahap pelatihan [59]. Proses standardisasi atau normalisasi dilakukan hanya pada data latih dan kemudian diaplikasikan ke data validasi maupun uji dengan parameter yang sama. Untuk fitur kategorikal, pendekatan yang memanfaatkan estimasi berbasis target atau statistik *out-of-fold* mencegah kebocoran yang kerap muncul pada pengkodean naif [56].

Isu bias–variansi menjadi pertimbangan sentral. Model yang sangat fleksibel bisa mencapai kesalahan pelatihan rendah namun gagal pada data baru karena variansi tinggi. Sebaliknya, model yang terlalu kaku memiliki bias tinggi dan tidak mampu menangkap pola data [57]. Teknik *unsemble*, regularisasi, serta peningkatan jumlah sampel efektif menurunkan variansi, sedangkan *feature engineering* yang bermakna dan pemilihan arsitektur yang sesuai menurunkan bias [55]. Pada data dengan dimensi tinggi dan sampel terbatas, kutukan dimensi memaksa penggunaan teknik reduksi dimensi atau seleksi fitur untuk meningkatkan rasio sinyal terhadap kebisingan [64].

Pengambilan keputusan operasional tidak selalu berhenti pada skor metrik. Penyesuaian ambang keputusan mengizinkan *trade-off* yang sadar biaya antara kesalahan tipe I dan tipe II [59]. *Cost sensitive learning* memasukkan matriks biaya langsung ke dalam proses pembelajaran agar model memprioritaskan pengurangan kesalahan yang dampaknya lebih besar. *Rejection option* atau *abstain threshold* dapat diterapkan ketika model kurang yakin sehingga kasus kritis dialihkan ke peninjauan manual [63]. Dalam ranah aplikasi pendidikan, keputusan ini berkaitan dengan alokasi sumber daya intervensi dan prioritas tindak lanjut yang tepat sasaran.

Reproducibility dan akuntabilitas menuntut praktik terbaik yang terdokumentasi. Penetapan *seed* acak, pelacakan versi data dan kode, serta pelaporan interval kepercayaan melalui *repeated cross validation* meningkatkan kepercayaan terhadap hasil [65]. Selain itu, audit jejak

eksperimen dan pelaporan konfigurasi *hyperparameter* memungkinkan replikasi dan perbandingan yang adil antar studi. Kombinasi kerangka evaluasi yang ketat, pengolahan data yang benar, serta perancangan metrik yang sesuai memastikan bahwa model klasifikasi tidak hanya unggul di atas kertas tetapi juga layak diterapkan untuk pengambilan keputusan dunia nyata [66].

2.3.4 Model *Gradient Boosting* dan *Catboost*

Model *Gradient Boosting* berkembang sebagai salah satu pendekatan paling efektif dalam menyelesaikan berbagai permasalahan prediksi, terutama ketika data memiliki pola yang kompleks. Konsep dasarnya adalah membangun model secara bertahap dengan memperbaiki kesalahan dari model sebelumnya sehingga akurasi dapat meningkat pada setiap iterasi. Di antara berbagai variasi metode ini, CatBoost menjadi salah satu algoritma yang banyak digunakan karena mampu menangani fitur kategorikal secara langsung dan memberikan performa yang stabil pada dataset berukuran kecil maupun tidak seimbang. Pendekatan ini relevan dengan kebutuhan penelitian yang memerlukan ketepatan prediksi tinggi tanpa proses prapengolahan yang berlebihan.

2.3.4.1 Model Gradient Boosting

Model *gradient boosting* merupakan salah satu pendekatan *ensemble* yang menggabungkan sejumlah *weak learner*, umumnya berupa pohon keputusan berukuran kecil, untuk menghasilkan model prediktif yang kuat dan stabil [67]. Teknik ini bekerja secara aditif dengan membangun model secara bertahap, di mana setiap model baru berusaha memperbaiki kesalahan model sebelumnya. Proses pembelajaran dilakukan dengan mengoptimalkan *loss function* menggunakan metode *gradient descent* dalam ruang fungsi, bukan pada ruang parameter seperti dalam model linear [68]. Pendekatan ini pertama kali diformulasikan secara sistematis oleh Friedman (2001) dalam *Gradient Boosting Machine* (GBM).

Pada dasarnya, *gradient boosting* berusaha meminimalkan fungsi kerugian $L(y, F(x))$ terhadap fungsi prediksi $F(x)$. Algoritma ini memulai

dari model awal $F_0(x)$, biasanya berupa nilai rata-rata dari target, lalu pada iterasi ke-m membentuk model baru yang memperbaiki residu atau gradien negatif dari fungsi kehilangan tersebut [69]. Proses matematisnya dapat dinyatakan pada Rumus 2.1.

$$F_m(x) = F_{m-1}(x) + v \, . \, h_m(x) \tag{2.1}$$

Rumus 2.1 *Boosting* [69]

dengan:

- $F_m(x)$ = model hasil *boosting* ke-m
- $F_{m-1}(x)$ = model sebelumnya
- $h_m(x)$ = pohon keputusan yang dilatih untuk memperkirakan gradien negatif dari *loss function*
- v = *learning rate* yang mengontrol besarnya pembaruan model agar tidak terlalu agresif

Proses pembaruan gradien tersebut dilakukan untuk semua sampel pelatihan $i = 1,2,\ldots,N$ pada Rumus 2.2.

$$r_{im} = \left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)} \tag{2.2}$$

Rumus 2.2 Pembaruan *Gradient* [70]

Setelah *gradient* dihitung, model baru $h_m(x)$ dilatih untuk mendekati nilai residu r_{im}, kemudian hasilnya ditambahkan ke model sebelumnya dengan bobot *learning rate* yang sesuai. Dengan mengulangi proses ini sebanyak *M* kali, *gradient boosting* mampu menghasilkan model yang sangat fleksibel dan memiliki bias rendah, walaupun risiko *overfitting* meningkat apabila iterasi dan kedalaman pohon tidak dikendalikan dengan baik [70].

Performa *gradient boosting* terbukti sangat baik untuk data tabular, tetapi terdapat beberapa kelemahan teknis [18]. Pertama, model ini sensitif terhadap pengkodean fitur kategorikal karena membutuhkan transformasi seperti *one-hot encoding* yang dapat memperbesar dimensi data secara signifikan [71]. Kedua, proses pembelajaran dapat menghasilkan *bias* akibat urutan data (*prediction shift*) ketika dilakukan secara sekuensial

[34]. Ketiga, pada *dataset* kecil, metode ini cenderung *overfit* karena tidak memiliki mekanisme regularisasi yang cukup kuat. Keterbatasan tersebut melatarbelakangi lahirnya berbagai varian baru seperti *XGBoost*, *LightGBM*, dan *CatBoost* [13].

2.3.4.2 CatBoost

CatBoost (*Categorical Boosting*) merupakan pengembangan dari konsep *gradient boosting* yang dirancang oleh Yandex pada tahun 2017 untuk mengatasi tiga kelemahan utama algoritma *boosting* konvensional, yaitu penanganan fitur kategorikal, *prediction shift*, dan risiko *overfitting* pada *dataset* kecil [72],[73]. CatBoost tetap berlandaskan pada prinsip *boosting* berbasis *gradient*, tetapi memperkenalkan dua inovasi utama yaitu *Ordered Boosting* dan *Efficient Handling of Categorical Features* [74].

1. *Ordered Boosting*

   Ordered Boosting merupakan mekanisme regularisasi yang menghindari *target leakage* yang muncul akibat penggunaan data pelatihan secara berurutan [75]. Dalam metode *boosting* konvensional, model pada iterasi m sering kali menggunakan informasi target dari seluruh data pelatihan, termasuk sampel yang digunakan untuk memprediksi dirinya sendiri, sehingga menyebabkan *prediction shift* [76]. *Ordered Boosting* mengatasi hal ini dengan hanya menggunakan subset data yang didahului oleh observasi saat ini untuk memperkirakan gradien [77]. Secara matematis, fungsi pembaruan dapat ditulis seperti pada Rumus 2.3.

$$F_{t+1}(x_i) = F_t(x_i) + \eta \, . \, g_t(x_i) \tag{2.3}$$

Rumus 2.3 *Ordered Boosting* [72]

   dengan $g_t(x_i)$ dihitung menggunakan urutan pengamatan $\{x_1, x_2, \ldots, x_{i-1}\}$, bukan seluruh dataset. Pendekatan ini memastikan bahwa model tidak menggunakan informasi masa depan pada saat pelatihan, sehingga hasil prediksi lebih stabil dan bebas dari bias akibat *data leakage* [78].

2. Penanganan Fitur Kategorikal Secara Otomatis

Salah satu keunggulan paling menonjol dari *CatBoost* adalah kemampuannya menangani fitur kategorikal tanpa memerlukan *one-hot encoding*. Sebagai gantinya, *CatBoost* menggunakan skema pengkodean berbasis *target statistics* yang mengonversi setiap kategori menjadi nilai numerik dengan memanfaatkan rata-rata label target pada subset data [72]. Nilai pengkodean untuk fitur kategorikal x_{cat} pada pengamatan ke-i dihitung menggunakan Rumus 2.4.

$$encoding\ (x_{cat,i}) = \frac{\sum j < i\ [x_{cat,j} = x_{cat.i}] \cdot y_j + a \cdot P}{\sum j < i [x_{cat,j} = x_{cat.i}] + a} \quad (2.4)$$

Rumus 2.4 *Encoding Catboost* [73]

dengan:

- $[x_{cat,j} = x_{cat,i}]$ = fungsi indikator yang bernilai 1 jika kategori sama,
- y_j = label target untuk observasi sebelumnya,
- a = parameter regularisasi untuk menghindari pembagian dengan nol,
- P = rata-rata target global pada seluruh data pelatihan.

Formula ini dikenal sebagai *ordered target encoding* karena menghitung nilai rata-rata target berdasarkan urutan data. Dengan cara ini, *CatBoost* dapat menghindari kebocoran informasi dan tetap memanfaatkan distribusi target untuk merepresentasikan fitur kategorikal dengan efisien [74].

3. *Loss Function* dan *Objective*

CatBoost menggunakan *loss function* yang bervariasi tergantung pada jenis tugas. Untuk klasifikasi biner, fungsi kehilangan yang umum digunakan adalah *log loss* atau *cross entropy* seperti pada Rumus 2.5.

$$L = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(p_i) + (1 - y_i)\log(1 - p_i)] \quad (2.5)$$

Rumus 2.5 *Log Loss* [77]

dengan $p_i = \sigma(F(x_i)) = \frac{1}{1+e^{-F(x_i)}}$.

Gradient dari fungsi ini dihitung sebagai dasar untuk memperbarui model pada setiap iterasi:

$$\frac{\partial L}{\partial F(x_i)} = p_i - y_i \tag{2.6}$$

Rumus 2.6 *CatBoost*

Pendekatan pada Rumus 2.6 memungkinkan *CatBoost* menyesuaikan diri terhadap probabilitas kelas dengan cara yang serupa dengan regresi logistik, namun dengan kekuatan *unsemble* pohon.

Selain dua inovasi utama di atas, CatBoost memiliki beberapa keunggulan tambahan seperti *symmetric tree structure* yang mempercepat inferensi, *ordered boosting random permutation* yang menjaga stabilitas hasil pada *dataset* kecil, serta dukungan paralelisasi penuh untuk mempercepat proses pelatihan. Struktur pohon simetris memastikan setiap level pohon menggunakan kriteria pemisahan yang sama untuk seluruh cabang, menghasilkan model yang lebih efisien dalam memori dan prediksi [76].

Secara keseluruhan, CatBoost menggabungkan keandalan *gradient boosting* dengan mekanisme regularisasi dan pengkodean canggih, menjadikannya sangat efektif untuk data tabular yang kompleks, terutama ketika mengandung banyak variabel kategorikal dan jumlah observasi terbatas. Kombinasi antara pendekatan *ordered boosting*, *target encoding*, dan optimisasi gradien menjadikan CatBoost unggul dalam stabilitas, efisiensi, dan interpretabilitas dibandingkan model *gradient boosting* konvensional.

2.3.5 Penanganan Data Tidak Seimbang

Permasalahan ketidakseimbangan kelas (*class imbalance*) merupakan tantangan umum dalam pemodelan klasifikasi, khususnya pada kasus di mana jumlah sampel pada satu kelas jauh lebih besar dibandingkan

kelas lainnya [79]. Kondisi ini sering muncul pada bidang pendidikan, kesehatan, dan keuangan, di mana kejadian "langka" justru menjadi fokus utama, seperti mahasiswa yang *drop out*, pasien dengan penyakit tertentu, atau transaksi penipuan [80],[81]. Ketidakseimbangan yang ekstrem dapat menyebabkan model *machine learning* menjadi bias terhadap kelas mayoritas karena algoritma pembelajaran cenderung meminimalkan kesalahan total tanpa mempertimbangkan distribusi kelas. Akibatnya, model dapat mencapai nilai akurasi tinggi namun gagal mengenali kasus minoritas yang lebih penting secara praktis [82].

Secara umum, permasalahan ini dapat diatasi melalui tiga pendekatan, yaitu modifikasi algoritma pembelajaran (*algorithm-level*), penyesuaian fungsi kehilangan (*cost-sensitive learning*), dan manipulasi distribusi data (*data-level*). Pendekatan terakhir, yaitu manipulasi distribusi data, menjadi strategi yang paling sering digunakan karena relatif mudah diterapkan dan bersifat independen terhadap model. Pendekatan ini dilakukan dengan dua teknik utama, yaitu *undersampling* terhadap kelas mayoritas dan *oversampling* terhadap kelas minoritas [83].

Undersampling berfungsi untuk mengurangi jumlah sampel kelas mayoritas dengan cara memilih subset data yang representatif, tetapi teknik ini berisiko menghilangkan informasi penting. Sebaliknya, *oversampling* menambah jumlah sampel kelas minoritas agar distribusi kelas menjadi seimbang. Pendekatan *oversampling* dapat dilakukan dengan cara sederhana seperti *Random Oversampling*, atau melalui metode sintetis seperti *Synthetic Minority Oversampling Technique* (SMOTE) yang memiliki landasan matematis lebih kuat [84].

2.3.5.1 Synthetic Minority Oversampling Technique (SMOTE)

SMOTE diperkenalkan oleh Chawla et al. pada 2002 sebagai metode *oversampling* berbasis sintesis data. Alih-alih menyalin data minoritas secara acak seperti pada *Random Oversampling*, SMOTE menghasilkan sampel sintetis baru dengan melakukan interpolasi antara satu

sampel minoritas dan tetangganya yang paling dekat. Ide utama SMOTE adalah memperluas ruang keputusan kelas minoritas agar lebih representatif terhadap distribusi populasi sebenarnya, sekaligus mengurangi risiko *overfitting* akibat duplikasi data [79].

Secara matematis, untuk setiap sampel minoritas x_i, SMOTE memilih salah satu tetangga terdekat x_{zi} berdasarkan jarak Euclidean dari himpunan *k-nearest neighbors (kNN)*. Kemudian, sebuah titik data sintetis x_{new}dihasilkan menggunakan persamaan seperti pada Persamaan 2.7.

$$x_{new} = x_i + \lambda \,.\, (x_{zi} - x_i) \tag{2.7}$$

Rumus 2.7 SMOTE [79]

dengan $\lambda \in [0,1]$ merupakan bilangan acak yang menentukan posisi interpolasi antara dua titik data. Proses ini dilakukan berulang kali hingga jumlah sampel minoritas meningkat sesuai rasio yang diinginkan. Dengan menghasilkan data sintetis yang berada di sepanjang garis antara dua titik minoritas, SMOTE membantu memperluas batas kelas tanpa menambah data duplikat [85].

Kelebihan utama SMOTE terletak pada kemampuannya memperkaya distribusi kelas minoritas dengan pola yang lebih alami. Namun, metode ini juga memiliki kelemahan, terutama ketika data mengandung *noise* atau kelas minoritas tidak terdistribusi secara homogen. Dalam kasus tersebut, interpolasi acak dapat menciptakan sampel sintetis di area yang sebenarnya dimiliki kelas mayoritas, sehingga memperburuk kinerja model [86]. Masalah ini memunculkan kebutuhan akan teknik lanjutan yang dapat menyeimbangkan data sekaligus membersihkan batas antar kelas, salah satunya adalah SMOTETomek.

2.3.5.2 SMOTETomek

SMOTETomek merupakan metode hibrida yang menggabungkan dua teknik utama yaitu *oversampling* dengan SMOTE dan *undersampling* menggunakan *Tomek Links*. Pendekatan ini pertama kali diusulkan oleh Batista et al. pada 2004 sebagai solusi untuk mengatasi kelemahan SMOTE

yang berpotensi menimbulkan *overlap* antar kelas. Inti dari metode ini adalah menambah data sintetis dari kelas minoritas menggunakan SMOTE, kemudian membersihkan data yang ambigu atau tumpang tindih menggunakan konsep *Tomek Links* [82].

Tomek Link didefinisikan sebagai pasangan titik (x_i, x_j) dari dua kelas berbeda yang merupakan tetangga terdekat satu sama lain. Pasangan ini menunjukkan bahwa kedua titik berada di area batas keputusan (*decision boundary*) yang kabur [83]. Secara formal, suatu pasangan (x_i, x_j) dikatakan sebagai *Tomek Link* jika berlaku seperti pada Rumus 2.8.

$$d(x_i, x_j) = \min_{x_k \in D} d(x_i, x_k) \tag{2.8}$$

Rumus 2.8 *Tomek Link* [83]

dengan $d(x_i, x_j)$menyatakan jarak *Euclidean* antara dua titik tersebut, dan Dadalah himpunan seluruh data pelatihan. Ketika *Tomek Link* ditemukan, paling tidak satu dari dua titik tersebut dihapus dari dataset, biasanya yang berasal dari kelas mayoritas. Langkah ini berfungsi untuk menghilangkan sampel yang berada di perbatasan kelas dan memperjelas pemisahan antar distribusi.

Proses SMOTETomek dilakukan dalam dua tahap:

1. Tahap SMOTE – menambah sampel sintetis untuk memperbesar jumlah data kelas minoritas sesuai rasio yang diinginkan.
2. Tahap *Tomek Cleaning* – menghapus pasangan data yang membentuk *Tomek Links* untuk menghilangkan area tumpang tindih antara kelas minoritas dan mayoritas.

Pendekatan ini menghasilkan dataset yang tidak hanya seimbang, tetapi juga lebih "bersih" dan terstruktur. Kombinasi *oversampling* dan *undersampling* membuat model yang dihasilkan memiliki batas keputusan yang lebih jelas dan performa generalisasi yang lebih baik [80].

2.3.5.3 SMOTENC

SMOTENC (*Synthetic Minority Oversampling Technique for Nominal and Continuous Features*) merupakan perluasan dari metode SMOTE yang dirancang khusus untuk menangani *dataset* dengan kombinasi fitur numerik dan kategorikal. Metode ini relevan digunakan ketika distribusi kelas tidak seimbang, tetapi data tidak dapat diproses menggunakan SMOTE biasa karena keberadaan fitur kategorikal yang tidak dapat diinterpolasi secara langsung [87].

SMOTENC membedakan perlakuan terhadap setiap jenis fitur. Fitur numerik tetap diproses dengan prinsip interpolasi linear sebagaimana pada SMOTE, sedangkan fitur kategorikal dihasilkan melalui mekanisme *voting* pada k-*nearest neighbors* dari sampel minoritas. Pendekatan ini mencegah terbentuknya nilai kategorikal yang tidak valid, sekaligus tetap memungkinkan penciptaan sampel sintetik yang konsisten dengan pola data asli [88].

Secara garis besar, proses SMOTENC dilakukan dalam dua tahap. Pertama, algoritma mengidentifikasi tetangga terdekat dari setiap sampel minoritas, dengan jarak antar titik dihitung menggunakan modifikasi jarak *Euclidean* yang memberi penalti khusus pada perbedaan nilai kategorikal. Kedua, sampel sintetik dibentuk melalui interpolasi pada fitur numerik dan pemilihan mayoritas pada fitur kategorikal. Dengan cara ini, SMOTENC mempertahankan sifat asli data sekaligus memperbaiki ketidakseimbangan distribusi [86].

Penggunaan SMOTENC menjadi penting dalam domain pendidikan, termasuk pada penelitian ini karena sebagian variabel menjelaskan karakteristik demografis atau latar belakang mahasiswa yang dinyatakan dalam bentuk kategorikal, seperti asal daerah atau asal sekolah. Jika dilakukan interpolasi linear terhadap fitur tersebut, sampel sintetis yang tercipta akan kehilangan makna. Dengan memanfaatkan SMOTENC, proses

oversampling dapat tetap dilakukan tanpa mengorbankan validitas nilai kategorikal [89].

Meskipun menawarkan pendekatan yang lebih realistis untuk data campuran, SMOTENC tetap memiliki keterbatasan. Algoritma ini sensitif terhadap *noise* dan dapat menghasilkan sampel sintetis yang tidak sepenuhnya mencerminkan kompleksitas hubungan antar fitur kategorikal dan numerik. Namun, pada kondisi distribusi data yang tidak seimbang, metode ini tetap menjadi salah satu pilihan yang efektif sebelum model klasifikasi dibangun [90].

2.3.5.4 Penerapan dalam Pemodelan Klasifikasi

Dalam pemodelan prediksi kinerja akademik, ketidakseimbangan kelas sering muncul karena proporsi mahasiswa yang lulus tepat waktu jauh lebih besar dibandingkan dengan yang tidak lulus tepat waktu atau *drop out*. Tanpa penanganan khusus, model klasifikasi akan cenderung memprioritaskan prediksi kelas mayoritas karena kontribusi kesalahan dari kelas minoritas sangat kecil terhadap fungsi rugi total.

Dengan menerapkan SMOTE atau SMOTETomek, distribusi kelas dapat diseimbangkan sehingga algoritma seperti CatBoost mampu belajar pola dari seluruh kelas secara proporsional. Penggunaan SMOTETomek sangat direkomendasikan ketika data memiliki *boundary noise* atau tumpang tindih yang kuat antar kelas, karena tahap pembersihan Tomek Links membantu mengurangi distorsi pada daerah batas.

Kombinasi metode ini terbukti efektif dalam meningkatkan *recall* dan *F1-score* pada kelas minoritas tanpa menurunkan performa keseluruhan model secara signifikan. Dengan demikian, SMOTE dan SMOTETomek berperan penting dalam menghasilkan model prediksi yang tidak hanya akurat secara statistik, tetapi juga adil (*fair*) dalam mengenali setiap kategori hasil akademik mahasiswa.

2.3.6 Optimasi Model (Hyperparameter Tuning dan Optuna)

Setiap algoritma *machine learning* memiliki seperangkat parameter yang mengontrol perilaku proses pelatihan dan memengaruhi hasil akhir model. Parameter tersebut dikenal sebagai *hyperparameter*, yaitu nilai yang tidak dipelajari secara langsung dari data, melainkan harus ditentukan sebelum proses pelatihan dimulai [91]. *Hyperparameter* berbeda dari parameter model seperti bobot pada regresi linear atau *node* pada pohon keputusan, karena nilainya mengatur bagaimana model belajar, bukan apa yang dipelajari [92]. Contoh *hyperparameter* pada algoritma *gradient boosting* seperti CatBoost mencakup *learning rate*, kedalaman pohon (*depth*), jumlah iterasi (*iterations*), serta kekuatan regularisasi.

Pemilihan nilai *hyperparameter* yang tepat sangat penting untuk mencapai kinerja optimal. Jika nilainya terlalu besar atau kecil, model dapat mengalami *underfitting* atau *overfitting* [93]. Fungsi objektif dalam *machine learning* jarang memiliki bentuk analitik yang dapat diselesaikan secara langsung maka pencarian kombinasi *hyperparameter* terbaik biasanya dilakukan secara empiris melalui proses yang disebut *hyperparameter tuning* [94]. Tujuan utama dari proses ini adalah menemukan konfigurasi parameter yang meminimalkan kesalahan prediksi atau memaksimalkan metrik evaluasi tertentu, seperti *F1-score* atau AUC [94].

2.3.6.1 Pendekatan Umum dalam Hyperparameter Tuning

Secara umum terdapat beberapa pendekatan utama untuk melakukan *tuning hyperparameter*, yaitu *Grid Search*, *Random Search*, dan *Bayesian Optimization*.

1. *Grid Search*

   Grid Search merupakan metode pencarian menyeluruh yang mencoba seluruh kombinasi *hyperparameter* dari ruang pencarian yang telah ditentukan. Pendekatan ini sederhana dan sistematis, tetapi memiliki kelemahan utama berupa kompleksitas komputasi yang tinggi karena jumlah kombinasi meningkat secara eksponensial seiring

bertambahnya jumlah parameter [93]. Meskipun demikian, *Grid Search* sering digunakan untuk studi awal karena memberikan pemahaman menyeluruh tentang sensitivitas model terhadap perubahan parameter [94].

2. *Random Search*

   Random Search mengatasi keterbatasan *Grid Search* dengan memilih kombinasi parameter secara acak dari ruang pencarian. Penelitian oleh Bergstra dan Bengio pada 2012 menunjukkan bahwa *Random Search* sering kali lebih efisien, terutama ketika hanya sebagian kecil *hyperparameter* yang benar-benar berpengaruh terhadap performa model [95]. Meskipun hasilnya tidak menjamin konfigurasi terbaik secara global, metode ini jauh lebih cepat dan dapat menemukan parameter mendekati optimal dengan upaya komputasi yang lebih kecil.

3. *Bayesian Optimazation*

   Bayesian Optimization memperkenalkan pendekatan probabilistik untuk memperkirakan hubungan antara *hyperparameter* dan metrik evaluasi. Prinsip dasarnya adalah menggunakan model *surrogate*, seperti *Gaussian Process* atau *Tree-structured Parzen Estimator* (TPE) untuk memprediksi performa model berdasarkan hasil percobaan sebelumnya [91]. Pendekatan ini secara iteratif menyeimbangkan antara eksplorasi (menguji area baru dari ruang parameter) dan eksploitasi (memfokuskan pencarian pada area dengan performa tinggi). Dengan demikian, *Bayesian Optimization* mampu menemukan kombinasi parameter optimal dengan jumlah iterasi yang jauh lebih sedikit dibandingkan *Grid* atau *Random Search* [96].

2.3.6.2 Konsep dan Mekanisme Optuna

Optuna merupakan pustaka Python modern untuk optimasi *hyperparameter* yang mengimplementasikan pendekatan *Bayesian Optimization* secara efisien dan fleksibel [97]. *Framework* ini dikembangkan oleh Takuya Akiba dkk. pada 2019 dan telah digunakan luas

dalam berbagai penelitian karena kemampuannya melakukan optimasi otomatis pada model *machine learning* berskala besar [98].

Optuna menggunakan pendekatan *sequential model-based optimization* (SMBO) dengan *Tree-structured Parzen Estimator* (TPE) sebagai model *surrogate*. Alih-alih mencoba kombinasi parameter secara acak, TPE memperkirakan distribusi probabilitas performa model berdasarkan percobaan yang telah dilakukan sebelumnya [99]. Dengan menggunakan dua distribusi, yaitu distribusi *good trials* (percobaan dengan hasil baik) dan *bad trials* (percobaan dengan hasil buruk), Optuna memilih *hyperparameter* baru dari area yang memiliki peluang tertinggi untuk menghasilkan performa yang lebih baik [100].

Secara matematis, pendekatan ini dapat dijelaskan sebagai berikut. Misalkan x merupakan vektor *hyperparameter* dan y adalah fungsi objektif yang akan diminimalkan. Tujuan optimasi adalah menemukan x^* yang meminimalkan $y = f(x)$. Dalam pendekatan TPE, distribusi posterior $p(x \mid y)$diestimasi menggunakan dua distribusi terpisah seperti pada Rumus 2.9.

$$p(x|y) = \begin{cases} l(x)\ jika\ y < y* \\ g(x)\ jika\ y \geq y* \end{cases} \quad (2.9)$$

Rumus 2.9 Optuna [97]

di mana $l(x)$ merupakan distribusi parameter yang menghasilkan nilai fungsi objektif lebih baik dari ambang batas y^*, sedangkan $g(x)$ adalah distribusi yang menghasilkan nilai lebih buruk. Optuna kemudian memilih kombinasi *hyperparameter* yang memaksimalkan rasio $\frac{l(x)}{g(x)}$ yang berarti memilih parameter dengan peluang tertinggi untuk memperbaiki performa model [98].

Optuna juga memiliki beberapa fitur penting yang membuatnya unggul dibandingkan pustaka optimasi lain seperti *Hyperopt* atau *Scikit-Optimize*, yaitu:

1. *Automatic Pruning* (*Early Stopping*)

   Optuna dapat menghentikan percobaan yang performanya buruk sejak awal menggunakan mekanisme *median stopping rule*. Hal ini menghemat sumber daya komputasi karena model tidak perlu diselesaikan sepenuhnya jika hasil sementaranya sudah menunjukkan performa rendah.

2. *Dynamic Search Space*

   Ruang pencarian *hyperparameter* di Optuna dapat dibangun secara dinamis selama proses optimasi. Artinya, parameter baru bisa ditambahkan berdasarkan nilai parameter sebelumnya, memungkinkan pencarian yang lebih adaptif dan efisien

3. *Parallel Optimization*

   Optuna mendukung eksekusi paralel pada beberapa inti prosesor atau GPU sehingga proses *tuning* dapat dilakukan lebih cepat tanpa kehilangan konsistensi hasil.

4. Visualisasi dan *Logging*

   Framework ini menyediakan modul visualisasi interaktif untuk memantau distribusi *hyperparameter*, *optimization history*, dan hubungan antarparameter dengan metrik performa model.

2.3.6.3 Penerapan dalam Model CatBoost

Dalam penelitian yang menggunakan algoritma CatBoost, *tuning hyperparameter* berperan penting untuk meningkatkan performa model sekaligus mencegah *overfitting*. Beberapa *hyperparameter* kunci yang umum dioptimasi meliputi:

- learning_rate (η): mengontrol besar langkah dalam setiap iterasi pembaruan gradien. Nilai terlalu besar mempercepat konvergensi namun meningkatkan risiko *overfitting*, sedangkan nilai terlalu kecil memperlambat pembelajaran.
- depth: menentukan kedalaman maksimum pohon dalam setiap iterasi *boosting*. Pohon yang terlalu dalam dapat menangkap interaksi kompleks antar fitur namun mengorbankan generalisasi.

- iterations: jumlah total pohon yang dibangun. Jumlah besar memberikan akurasi lebih tinggi hingga titik tertentu, tetapi menambah biaya komputasi dan risiko *overfitting*.
- l2_leaf_reg: parameter regularisasi L2 untuk mengendalikan kompleksitas model.
- bagging_temperature: mengatur tingkat *randomness* pada proses *bootstrap* untuk meningkatkan stabilitas.

Optuna mengotomatisasi pencarian kombinasi terbaik dari parameter-parameter ini dengan meminimalkan fungsi objektif yang didefinisikan berdasarkan metrik evaluasi seperti *macro F1-score*. Contoh fungsi objektif dalam Optuna dapat diformulasikan seperti pada Rumus 2.10.

$$Objective(\theta) = 1 - F1_{macro}(f_\theta(X_{val}), Y_{val} \quad (2.10)$$

Rumus 2.10 Optuna pada CatBoost [101]

di mana $f_\theta$adalah model CatBoost dengan parameter θ, dan (X_{val}, Y_{val})merupakan data validasi. Nilai fungsi ini diminimalkan hingga diperoleh konfigurasi parameter dengan performa tertinggi.

Dengan menggunakan Optuna, proses *tuning* menjadi lebih adaptif, efisien, dan terukur dibandingkan pencarian manual. Hasil akhirnya adalah model CatBoost yang memiliki keseimbangan optimal antara bias dan variansi, mampu menangani data tidak seimbang secara lebih baik, serta memberikan hasil prediksi yang stabil dan dapat diandalkan untuk tugas klasifikasi akademik.

2.3.7 Interpretasi Model: Explainable AI (XAI) dan SHAP

Salah satu tantangan utama dalam penerapan *machine learning* modern adalah keterbatasan interpretabilitas model, terutama pada algoritma kompleks seperti *ensemble learning* dan *deep learning* [102]. Model yang memiliki performa prediktif tinggi sering kali bersifat *black box*, artinya sulit dipahami bagaimana model menghasilkan keputusan tertentu. Kondisi ini

menimbulkan masalah serius dalam konteks akademik maupun profesional, karena pengguna tidak hanya membutuhkan hasil prediksi, tetapi juga alasan yang mendasari prediksi tersebut [103]. Untuk menjawab kebutuhan tersebut, berkembanglah bidang *Explainable Artificial Intelligence* (XAI) yang bertujuan untuk membuat model kecerdasan buatan menjadi lebih transparan, dapat dipahami, dan dapat dipercaya oleh manusia.

Explainable AI mencakup seperangkat prinsip, teknik, dan metodologi yang memungkinkan pengguna memahami mekanisme pengambilan keputusan model tanpa mengorbankan performa prediktifnya [104]. Tujuan utama XAI adalah memberikan *explanation* yang dapat ditafsirkan oleh manusia atas hasil prediksi model, baik secara lokal (penjelasan terhadap satu prediksi individu) maupun global (penjelasan terhadap keseluruhan perilaku model). Pendekatan XAI juga berperan penting dalam meningkatkan keadilan (*fairness*), akuntabilitas (*accountability*), dan transparansi (*transparency*) dari sistem kecerdasan buatan [105].

2.3.7.1 Konsep dan Pendekatan Explainable AI (XAI)

Dalam literatur *machine learning*, metode interpretasi dapat diklasifikasikan berdasarkan dua dimensi utama, yaitu *model-specific* dan *model-agnostic*, serta global dan *local explanation*.

1. *Model-specific explanation* merupakan metode yang hanya dapat diterapkan pada algoritma tertentu karena bergantung pada struktur internal model. Contohnya adalah *feature importance* pada *Random Forest* dan *gain-based importance* pada XGBoost [105].
2. *Model-agnostic explanation* bersifat universal dan dapat diterapkan pada model apa pun karena beroperasi pada level *input-output* tanpa perlu mengetahui struktur internal. Contoh metode ini antara lain LIME (*Local Interpretable Model-agnostic Explanations*) dan SHAP (*SHapley Additive exPlanations*) [106].

3. *Global explanation* menjelaskan bagaimana model bekerja secara keseluruhan, dengan menyoroti fitur-fitur yang paling berpengaruh terhadap prediksi di seluruh data [107].
4. *Local explanation* berfokus pada satu observasi tertentu, menjelaskan alasan mengapa model memberikan prediksi tertentu untuk kasus tersebut [108].

Kombinasi dari pendekatan *model-agnostic* dan *local explanation* menjadi sangat populer karena memberikan fleksibilitas dalam menjelaskan hasil prediksi tanpa bergantung pada arsitektur model. Salah satu metode yang paling banyak digunakan dalam kategori ini adalah SHAP.

2.3.7.2 Konsep Dasar SHAP (SHapley Additive exPlanations)

SHAP merupakan metode interpretasi yang dikembangkan oleh Lundberg dan Lee pada 2017 berdasarkan teori permainan kooperatif (*cooperative game theory*). SHAP menghitung kontribusi setiap fitur terhadap hasil prediksi dengan menggunakan nilai Shapley yang diambil dari konsep nilai kontribusi pemain dalam permainan [109]. Dalam *machine learning*, setiap fitur dianggap sebagai "pemain" yang berkontribusi terhadap "hasil permainan", yaitu nilai prediksi model.

Nilai SHAP (ϕ_i) untuk fitur ke-i dihitung dengan mempertimbangkan semua kemungkinan subset fitur yang mungkin muncul dalam model. Secara matematis, nilai SHAP didefinisikan pada Rumus 2.11.

$$\phi_i = \sum_{S \subseteq N\{i\}} \frac{|S|!\,(|N| - |S| - 1)!}{|N|!} [f(S \cup \{i\}) - f(S)] \tag{2.11}$$

Rumus 2.11 SHAP [110]

dengan:

- N = himpunan semua fitur,
- S = subset dari fitur yang tidak mengandung fitur ke-i,
- $f(S)$ = nilai prediksi model ketika hanya fitur dalam subset Syang digunakan,

- $f(S \cup \{i\}) - f(S)$ = perubahan nilai prediksi ketika fitur $i$dimasukkan ke dalam subset S.

Rumus di atas menghitung kontribusi marjinal rata-rata fitur i terhadap seluruh kombinasi fitur lain. Setiap fitur mendapatkan bobot sesuai seberapa besar dampaknya terhadap pergeseran hasil prediksi. Proses ini memastikan bahwa interpretasi model bersifat aditif dan adil, karena kontribusi total seluruh fitur akan sama dengan selisih antara prediksi model dan rata-rata prediksi keseluruhan.

2.3.7.3 Keunggulan SHAP dibandingkan Metode Interpretasi Lain

Metode SHAP memiliki beberapa keunggulan dibandingkan pendekatan interpretasi lain seperti LIME atau *Permutation Importance*:

1. Konsistensi dan Keadilan (*Fairness*)
   SHAP memastikan bahwa jika kontribusi suatu fitur terhadap prediksi meningkat, maka nilai pentingnya tidak akan berkurang. Ini menjamin stabilitas hasil interpretasi antar eksperimen [108].
2. Additivitas (*Additivity*)
   Total kontribusi seluruh fitur selalu menjumlah pada selisih antara prediksi aktual dengan rata-rata prediksi dasar. Sifat ini memungkinkan interpretasi hasil model secara kuantitatif [109].
3. Fleksibilitas (*Model-Agnostic* dan *Model-Specific*)
   SHAP dapat diterapkan pada berbagai jenis model, baik yang berbasis pohon keputusan seperti CatBoost maupun model kompleks seperti jaringan saraf [111].
4. Kemampuan Visualisasi
   SHAP menyediakan berbagai bentuk visualisasi seperti *summary plot*, *dependence plot*, dan *force plot* yang membantu memahami hubungan antara fitur dan prediksi [112].

2.3.7.4 Implementasi SHAP pada Model Berbasis Pohon Keputusan

Pada model berbasis pohon seperti *Gradient Boosting* dan *CatBoost*, SHAP diimplementasikan secara efisien melalui algoritma TreeSHAP. Pendekatan ini memanfaatkan struktur pohon untuk menghitung nilai kontribusi fitur tanpa perlu menelusuri seluruh kombinasi subset fitur yang jumlahnya eksponensial [113]. Dengan kompleksitas waktu $O(TLD^2)$, di mana T adalah jumlah pohon, L jumlah daun per pohon, dan D kedalaman pohon, TreeSHAP mampu menghitung nilai SHAP dengan efisiensi tinggi bahkan pada dataset besar [112].

TreeSHAP bekerja dengan melacak jalur keputusan setiap sampel dalam pohon, kemudian menghitung kontribusi marjinal setiap fitur terhadap prediksi akhir berdasarkan probabilitas kemunculannya di jalur tersebut [105]. Nilai SHAP yang dihasilkan dapat diinterpretasikan baik secara global maupun lokal:

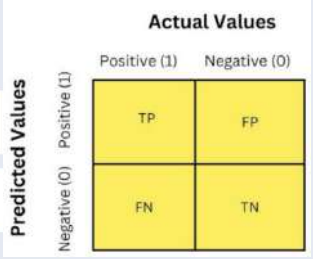
- Interpretasi global menunjukkan fitur-fitur yang secara umum paling berpengaruh terhadap prediksi model.
- Interpretasi lokal menjelaskan kontribusi fitur terhadap hasil prediksi untuk satu individu tertentu.

Dalam penelitian prediksi kinerja akademik, misalnya nilai SHAP dapat menunjukkan bahwa fitur seperti jumlah mata kuliah yang diulang, tren IPK per semester, atau nilai pada mata kuliah inti memiliki pengaruh besar terhadap kemungkinan kelulusan tepat waktu. Dengan demikian, model tidak hanya menghasilkan skor prediksi, tetapi juga memberikan penjelasan konkret yang dapat ditindaklanjuti oleh dosen pembimbing akademik.

2.3.8 Evaluation Metrics

Confusion matrix adalah alat evaluasi dalam *machine learning* yang dirancang untuk memvisualisasikan kinerja model klasifikasi melalui hubungan antara prediksi model dan data aktual dalam bentuk tabel matriks

[114]. Matriks ini terdiri dari empat elemen utama, diantaranya *True Positive* (TP) yang mencatat jumlah prediksi yang benar untuk kelas positif; *True Negative* (TN) yang menghitung prediksi benar untuk kelas negatif; *False Positive* (FP) yang menggambarkan kesalahan prediksi di mana model memprediksi positif meskipun sebenarnya negatif; dan *False Negative* (FN) yang mencatat kesalahan ketika model memprediksi negatif untuk data yang sebenarnya positif. Elemen-elemen ini membentuk dasar untuk perhitungan berbagai metrik penting yang membantu dalam mengevaluasi performa model klasifikasi secara mendalam yang dapat dilihat pada Gambar 2.2.



Gambar 2.2 Confusion Matrix [114]

Akurasi adalah metrik yang menghitung proporsi prediksi yang benar terhadap seluruh jumlah data seperti pada Rumus 2.12.

$$Akurasi = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.12}$$

Rumus *2*.12 Akurasi [115]

Metrik ini memberikan gambaran umum tentang seberapa sering model membuat prediksi yang benar, tetapi dapat menjadi kurang informatif jika *dataset* memiliki ketidakseimbangan kelas, misalnya ketika satu kelas jauh lebih dominan dibandingkan kelas lainnya.

Presisi berfokus pada keandalan prediksi positif. Presisi menunjukkan berapa proporsi prediksi positif yang relevan yang dapat dihitung dengan menggunakan Rumus 2.13.

$$Presisi = \frac{TP}{TP + FP} \tag{2.13}$$

Rumus 2.13 *Presisi* [115]

Presisi sangat penting di mana *False Positive* memiliki konsekuensi serius, seperti dalam diagnosis medis atau sistem keamanan.

Recall, atau sensitivitas, mengukur seberapa baik model mendeteksi semua kasus positif dalam *dataset*.

$$Recall = \frac{TP}{TP + FN} \tag{2.14}$$

Rumus 2.14 *Recall* [115]

Metrik pada Rumus 2.14 relevan dalam situasi di mana penting untuk meminimalkan *False Negative*, seperti mendeteksi penyakit yang memerlukan intervensi segera.

F1-*score* adalah rata-rata harmonis antara *Presisi* dan *Recall*, memberikan metrik yang lebih seimbang terutama dalam kasus ketidakseimbangan kelas yang dapat dilihat pada Rumus 2.15.

$$F1\ Score = 2\frac{Presisi\ .\ Recall}{Presisi + Recall} \tag{2.15}$$

Rumus 2.15 F1 Score [115]

F1-*score* sangat berguna untuk mengukur keseimbangan antara presisi dan *recall* dalam satu nilai yang komprehensif.

Confusion matrix memungkinkan identifikasi pola kesalahan spesifik yang dibuat oleh model, seperti kecenderungan salah memprediksi kelas tertentu. Objek yang dievaluasi meliputi hasil prediksi model dan label sebenarnya, memberikan informasi mendalam tentang kinerja model dalam hasil yang lebih kompleks. Dengan demikian, *confusion matrix* bukan hanya alat evaluasi, tetapi juga panduan strategis untuk mengarahkan perbaikan model dengan fokus yang lebih tajam pada kelemahan yang teridentifikasi [114]. Hal ini menjadikannya alat yang esensial dalam pengembangan model *machine learning*, khususnya untuk meningkatkan kualitas dan reliabilitas sistem prediksi.

2.4 Tools

Alat yang digunakan untuk merancang model sekaligus membangun *prototype* sederhana untuk penelitian ini.

2.4.1 Python

Python adalah bahasa pemrograman tingkat tinggi yang serbaguna, dikenal karena sintaksisnya yang sederhana dan mudah dipahami, menjadikannya pilihan utama untuk pengembang dari berbagai tingkat keahlian. Bahasa ini dirancang untuk meningkatkan produktivitas dengan pendekatan pemrograman yang intuitif, mendukung paradigma pemrograman berorientasi objek, fungsional, dan prosedural [116]. Python memiliki ekosistem yang kaya dengan berbagai pustaka dan *framework* seperti NumPy, Pandas, TensorFlow, dan PyTorch yang memungkinkan pengolahan data, analisis statistik, pengembangan aplikasi berbasis *web*, hingga penerapan kecerdasan buatan dan *machine learning*. Keunggulan Python terletak pada fleksibilitasnya yang memungkinkan pengguna untuk menangani data dalam jumlah besar dengan efisiensi tinggi, seperti pada analisis data berbasis grafik atau jaringan yang kompleks [117].

Bahasa ini juga didukung oleh komunitas global yang aktif, sehingga pengembang dapat dengan mudah mengakses dokumentasi, tutorial, dan solusi untuk masalah yang dihadapi. Salah satu objek utama yang sering dibahas dalam penggunaan Python adalah kemampuannya dalam memproses dan memvisualisasikan data dengan pustaka seperti Matplotlib dan Seaborn yang mempermudah pemahaman pola dalam data. Selain itu, Python dirancang untuk kompatibilitas dengan sistem operasi populer seperti Windows, macOS, dan Linux, sehingga memudahkan integrasi ke dalam berbagai lingkungan pengembangan. Python juga unggul dalam membangun aplikasi lintas platform dengan efisiensi tinggi, sekaligus mampu mendukung pengujian dan *deployment* secara cepat melalui integrasi dengan alat seperti Docker atau Jenkins. Dengan kemampuannya yang terus berkembang, Python menjadi alat yang tak tergantikan bagi berbagai kebutuhan teknologi modern, baik dalam skala kecil maupun proyek-proyek besar yang kompleks [117].

2.4.2 Google Collaboratory

Google Collaboratory adalah platform berbasis *cloud* yang memungkinkan pengguna menjalankan kode Python langsung melalui *browser*

tanpa memerlukan konfigurasi perangkat keras atau perangkat lunak tambahan [118]. Platform ini didesain untuk mendukung kebutuhan komputasi data yang intensif, seperti pembelajaran mesin, analisis data, dan pemrosesan teks, dengan akses gratis ke GPU dan TPU yang mempercepat proses perhitungan. Google Colab memanfaatkan Notebook Jupyter, sebuah format interaktif yang memungkinkan integrasi antara kode, visualisasi, dan dokumentasi dalam satu dokumen yang mudah digunakan dan dibagikan.

Google Colab mencakup data input, model pembelajaran mesin, dan hasil analisis atau prediksi yang semuanya dapat diakses dan diproses dalam lingkungan yang terintegrasi. Pengguna dapat dengan mudah mengunggah *dataset* dari berbagai sumber, seperti Google Drive, GitHub, atau sumber *online* lainnya, untuk diolah dalam proyek berbasis data. Dengan fitur kolaborasi *real-time*, Google Colab memungkinkan banyak pengguna untuk bekerja bersama secara simultan, sehingga mendukung efisiensi kerja tim. Selain itu, *platform* ini kompatibel dengan berbagai pustaka Python populer seperti TensorFlow, PyTorch, dan Scikit-learn yang sering digunakan untuk pengembangan model kecerdasan buatan dan analisis data lanjutan.

Keunggulan lainnya adalah kemampuannya untuk menyimpan dan berbagi *notebook* melalui Google Drive, menjadikan dokumen proyek dapat diakses kapan saja dan dari mana saja tanpa batasan perangkat. Dengan lingkungan yang mendukung integrasi dengan berbagai sumber daya *cloud* dan fleksibilitas penggunaan, Google Colab menjadi alat yang esensial bagi peneliti, pengembang, dan pelajar untuk mengeksplorasi potensi besar data dan teknologi secara efisien. Hal ini menjadikannya salah satu platform utama dalam dunia pemrograman modern, khususnya dalam bidang ilmu data dan kecerdasan buatan.