OPTIMASI DETEKSI CLICK FRAUD PADA IMBALANCE DATASET DENGAN QUAD DIVISION PROTOTYPE SELECTION BASED KNN DAN XGBOOST

SKRIPSI

Oleh:

KEVIN

20211000039

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS BUDDHI DHARMA

TANGERANG

2025

OPTIMASI DETEKSI CLICK FRAUD PADA IMBALANCE DATASET DENGAN QUAD DIVISION PROTOTYPE SELECTION BASED KNN DAN XGBOOST

SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana pada Program Studi Teknik Informatika Fakultas Sains dan Teknologi
Universitas Buddhi Dharma
Jenjang Pendidikan Strata 1

Oleh:
KEVIN
20211000039
FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS BUDDHI DHARMA
TANGERANG

2025

LEMBAR PERNYATAAN KEASLIAN SKRIPSI

Yang bertanda tangan di bawah ini,

NIM	: 20211000039
Nama	: Kevin
Jenjang Studi	: Strata 1
Program Studi	: Teknik Informatika
Peminatan	: Data Science

Dengan ini saya menyatakan bahwa:

1. Skripsi ini adalah asli dan belum pernah diajukan untuk mendapat gelar akademik Sarjana atau kelengkapan studi, baik di Universitas Buddhi Dharma maupun di Pereguruan Tinggi lainnya.
2. Skripsi ini saya buat sendiri tanpa bantuan dari pihak lain, kecuali arahan dosen pembimbing.
3. Dalam Skripsi ini tidak terdapat karya atau pendapat yang telah ditulis atau dipublikasikan orang lain, kecuali secara tertulis dengan jelas dan dicantumkan sebagai acuan dalam naskah dengan disebutkan nama pengarang dan dicantumkan daftar pustaka.
4. Dalam Skripsi ini tidak terdapat pemalsuan (kebohongan), seperti buku, artikel, jurnal, data sekunder, pengolahan data, dan pemalsuan tanda tangan dosen atau Ketua Program Studi Universitas Buddhi Dharma yang dibuktikan dengan keasliannya.
5. Lembar pernyataan ini saya buat dengan sesungguhnya, tanpa paksaan dan apabila dikemudian hari atau pada waktu lainnya terdapat penyimpangan dan ketidak benaran dalam pernyataan ini, saya bersedia menerima sanksi akademik berupa pencabutan gelar akademik yang telah saya peroleh karena Skripsi ini serta sanksi lainnya sesuai dengan peraturan dan norma yang berlaku.

Tangerang, 04-08-2025
Yang membuat pernyataan,

Kevin
20211000039

UNIVERSITAS BUDDHI DHARMA

LEMBAR PERSETUJUAN PUBLIKASI KARYA ILMIAH

Yang bertanda tangan di bawah ini,

NIM	: 20211000039
Nama	: Kevin
Jenjang Studi	: Strata 1
Program Studi	: Teknik Informatika
Peminatan	: Data Science

Dengan ini menyetujui untuk memberikan ijin kepada pihak Universitas Buddhi Dharma, Hak Bebas Royalti Non – Eksklusif (*Non-exclusive Royalty-Free Right*) atas karya ilmiah kami yang berjudul: "OPTIMASI DETEKSI CLICK FRAUD PADA IMBALANCE DATASET DENGAN QUAD DIVISION PROTOTYPE SELECTION BASED KNN DAN XGBOOST".

Dengan Hak Bebas Royalti Non – Eksklusif ini pihak Universitas Buddhi Dharma berhak menyimpan, mengalih-media atau format-kan, mengelolanya dalam pangkalan data (*database*), mendistribusikannya, dan menampilkan atau mempublikasikannya di *internet* atau media lain untuk kepentingan akademis tanpa perlu meminta ijin dari saya selama tetap mencantumkan nama saya sebagai penulis atau pencipta karya ilmiah tersebut.

Saya bersedia untuk menanggung secara pribadi, tanpa melibatkan pihak Universitas Buddhi Dharma, segala bentuk tuntutan hukum yang timbul atas pelanggaran Hak Cipta dalam karya ilmiah saya ini.

Demikian pernyataan ini saya buat dengan sebenarnya.

Tangerang, 04-08-2025

Yang membuat pernyataan,

Kevin

20211000039

LEMBAR PENGESAHAN PEMBIMBING

OPTIMASI DETEKSI CLICK FRAUD PADA IMBALANCE DATASET DENGAN QUAD DIVISION PROTOTYPE SELECTION BASED KNN DAN XGBOOST

Dibuat Oleh:

NIM : 20211000039

Nama : Kevin

Telah disetujui untuk dipertahankan di hadapan Tim Penguji Ujian Komprehensif

Program Studi Teknik Informatika

Data Science

Tahun Akademik 2024/2025

Disahkan oleh,

Tangerang, 04-08-2025

Pembimbing

Aditiya Hermawan, M.Kom., M.M.

NUPTK. 9538766667130223

LEMBAR PENGESAHAN SKRIPSI

OPTIMASI DETEKSI CLICK FRAUD PADA IMBALANCE DATASET DENGAN QUAD DIVISION PROTOTYPE SELECTION BASED KNN DAN XGBOOST

Dibuat Oleh:

NIM : 20211000039

Nama : Kevin

Telah disetujui untuk dipertahankan di hadapan Tim Penguji Ujian Komprehensif

Program Studi Teknik Informatika

Data Science

Tahun Akademik 2024/2025

Disahkan oleh,

Tangerang, 04-08-2025

Dekan,	Ketua Program Studi
Dr. Yakub, M.Kom., M.M.	**Hartana Wijaya, M.Kom.**
NUPTK. 1836747648130172	NUPTK. 3844759660131192

LEMBAR PENGESAHAN TIM PENGUJI

Nama	:	Kevin
NIM	:	20211000039
Fakultas	:	Sains dan Teknologi
Judul Skripsi	:	OPTIMASI DETEKSI CLICK FRAUD PADA IMBALANCE DATASET DENGAN QUAD DIVISION PROTOTYPE SELECTION BASED KNN DAN XGBOOST

Dinyatakan LULUS setelah mempertahankan di depan Tim Penguji pada hari Senin, 04-08-2025

		Nama penguji:	Tanda Tangan:
Ketua Sidang	:	**Edy, S.T., M.Kom.** NUPTK. 7560760661130203	.....
Penguji I	:	**Ramona Dyah Safitri, S.Si, M.Si.** NUPTK. 8652771672230312	.....
Penguji II	:	**Aditiya Hermawan, M.Kom., M.M.** NUPTK. 9538766667130223	.....

Mengetahui,

Dekan Fakultas Sains dan Teknologi

Dr. Yakub, M.Kom., M.M.

NUPTK. 1836747648130172

KATA PENGANTAR

Dengan mengucapkan Puji Syukur kepada Tuhan Yang Maha Esa, yang telah memberikan Rahmat dan karunia-Nya kepada penulis sehingga dapat menyusun dan menyelesaikan Skripsi ini dengan judul "**OPTIMASI DETEKSI CLICK FRAUD PADA IMBALANCE DATASET DENGAN QUAD DIVISION PROTOTYPE SELECTION BASED KNN DAN XGBOOST**". Tujuan utama dari pembuatan Skripsi ini adalah sebagi salah satu syarat kelengkapan dalam menyelesaikan program pendidikan Strata 1 Program Studi Teknik Informatika di Universitas Buddhi Dharma. Dalam penyusunan Skripsi ini penulis banyak menerima bantuan dan dorongan baik moril maupun materiil dari berbagai pihak, maka pada kesempatan ini penulis menyampaikan rasa terima kasih yang sebesar-besarnya kepada:

1. Ibu Dr. Limajatini, S.E., M.M., B.K.P, sebagai Rektor Universitas Buddhi Dharma
2. Bapak Dr. Yakub, S.Kom., M.Kom., M.M, Dekan Fakultas Sains dan Teknologi
3. Bapak Hartana Wijaya, S.Kom., M.Kom, sebagai Ketua Program Studi Teknik Informatika
4. Bapak Aditiya Hermawan, S.Kom., M.Kom, sebagai pembimbing yang telah membantu dan memberikan dukungan serta harapan untuk menyelesaikan penulisan Skripsi ini
5. Orang tua dan keluarga yang selalu memberikan dukungan baik moril dan materiil.
6. Teman-teman yang selalu membantu dan memberikan semangat

Serta semua pihak yang terlalu banyak untuk disebutkan satu-persatu sehingga terwujudnya penulisan ini. Penulis menyadari bahwa penulisan Skripsi ini masih belum sempurna, untuk itu penulis mohon kritik dan saran yang bersifat membangun demi kesempurnaan penulisan di masa yang akan datang.

Akhir kata semoga Skripsi ini dapat berguna bagi penulis khususnya dan bagi para pembaca yang berminat pada umumnya.

Tangerang, 04-08-2025

Penulis

Optimasi Deteksi *Click Fraud* Pada *Imbalance Dataset* Dengan *Quad Division Prototype Selection Based* KNN dan XGBoost
97 Halaman + xvii / 51 Tabel / 39 Gambar / 1 Lampiran

ABSTRAK

Pertumbuhan pesat teknologi informasi dan ekonomi digital mendorong peningkatan penggunaan iklan digital berbasis internet, seperti model *Pay-Per-Click* (PPC), yang menjadi strategi pemasaran efektif bagi banyak perusahaan. Namun, di balik efektivitasnya, model ini rentan terhadap ancaman *click fraud*, yaitu aktivitas klik tidak sah yang dilakukan untuk tujuan merugikan pengiklan, sehingga dapat menyebabkan kerugian finansial yang signifikan. Berbagai penelitian sebelumnya telah menggunakan model *machine learning* sebagai pendekatan untuk mendeteksi *click fraud*. Namun, ketidakseimbangan data menjadi tantangan utama yang dapat menghambat kinerja optimal model tersebut. Untuk mengatasi permasalahan ini, penelitian ini bertujuan mengoptimalkan deteksi *click fraud* dengan menggabungkan metode *Quad Division Prototype Selection Based K-Nearest Neighbor* (QDPSKNN), yang mampu menangani ketidakseimbangan data secara efektif, serta model *Extreme Gradient Boosting* (XGBoost) sebagai klasifikasi akhir. Dataset yang digunakan adalah *TalkingData AdTracking Fraud Detection Challenge*, yang terdiri dari 100.000 sampel dengan 8 atribut, termasuk alamat IP, ID aplikasi, jenis perangkat, sistem operasi, saluran distribusi iklan, serta label yang menunjukkan apakah klik tersebut valid atau merupakan palsu. Metode penelitian ini melibatkan *preprocessing data*, teknik *balancing data*, *hyperparameter tuning* menggunakan *Random Search*, serta evaluasi model dengan metrik *accuracy*, *precision*, *recall*, dan *F1-score*. Hasil penelitian menunjukkan bahwa kombinasi QDPSKNN dengan model XGBoost memberikan performa terbaik dengan *accuracy* sebesar 96,77%, *recall* 99,30%, *precision* 96,91%, dan *F1-score* 98,09%, mengungguli metode lainnya dalam mendeteksi *click fraud*. Manfaat dari penelitian ini adalah menyediakan pendekatan yang mampu meningkatkan akurasi deteksi klik tidak sah pada data yang tidak seimbang, dengan memanfaatkan teknik seleksi *instance* yang informatif untuk memperkuat keandalan sistem deteksi untuk platform periklanan digital.

Kata kunci: ***Data Balancing, Machine Learning, Pay-Per-Click, Random Search, TalkingData***

Optimazion of Click Fraud Detection on Imbalance Dataset using Quad Division Prototype Selection Based KNN and XGBoost
97 *Pages* + xvii / 51 *Tables* / 39 *Figures* / 1 *Attachments*

ABSTRACT

The rapid growth of information technology and the digital economy has driven an increase in the use of internet-based digital advertising, such as the Pay-Per-Click (PPC) model, which has become an effective marketing strategy for many companies. However, despite its effectiveness, this model is vulnerable to click fraud threats, which involve invalid click activities intended to harm advertisers, potentially causing significant financial losses. Various previous studies have employed machine learning models as an approach to detect click fraud. However, data imbalance remains a major challenge that can hinder the optimal performance of these models. To address this issue, this study aims to optimize click fraud detection by combining the Quad Division Prototype Selection Based K-Nearest Neighbor (QDPSKNN) method, which effectively handles data imbalance, with the Extreme Gradient Boosting (XGBoost) model as the final classifier. The dataset used is the TalkingData AdTracking Fraud Detection Challenge, consisting of 100,000 samples with 8 attributes, including IP address, application ID, device type, operating system, ad distribution channel, and a label indicating whether a click is valid or fraudulent. The research methodology involves data preprocessing, data balancing techniques, hyperparameter tuning using Random Search, and model evaluation using accuracy, precision, recall, and F1-score metrics. The results show that the combination of QDPSKNN with the XGBoost model achieves the best performance with an accuracy of 96.77%, recall of 99.30%, precision of 96.91%, and an F1-score of 98.09%, outperforming other methods in detecting click fraud. The benefit of this study is to provide an approach that enhances the accuracy of detecting invalid clicks on imbalanced data by utilizing informative instance selection techniques to improve the reliability of detection systems for digital advertising platforms.

Keywords: ***Data Balancing, Machine Learning, Pay-Per-Click, Random Search, TalkingData***

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

BAB I

PENDAHULUAN

1.1 Latar Belakang

Kemajuan teknologi informasi berbasis internet berdampak pada pertumbuhan ekonomi digital (Soemarwoto, 2022). Ekonomi digital merupakan proyeksi talenta baru di era revolusioner kita saat ini yang dapat menginspirasi daya saing. Ekonomi digital saat ini berkembang pesat di seluruh dunia, termasuk di Indonesia, seperti yang ditunjukkan oleh peningkatan jumlah pengguna internet (Rahayu et al., 2022). Bisnis harus mengubah strategi mereka di pasar yang semakin kompetitif, salah satunya adalah manajemen pemasaran *online*.

Manajemen pemasaran *online* merupakan pendekatan strategis dalam pemasaran yang memanfaatkan berbagai platform dan teknologi digital berbasis internet untuk mempromosikan barang dan layanan kepada audiens yang lebih tertarget (Daniati Nazara & Ginting, 2024). Internet merupakan jaringan global yang memungkinkan pertukaran informasi secara *real-time* di seluruh dunia, dan telah menjadi pendorong utama dalam perubahan cara bisnis berinteraksi dengan konsumen. Dengan kemajuan internet, bisnis atau organisasi lain menggunakan iklan digital untuk memasarkan barang mereka (Salam & Kho, 2023). Ada berbagai jenis iklan digital, termasuk pemasaran konten, pemasaran sosial media, *Search Engine Optimization* (SEO), *Search Engine Marketing* (SEM), *Pay-Per-Click* (PPC), *email marketing*, dan pesan singkat (Hananto et al., 2024). Salah satu strategi iklan digital yang paling banyak digunakan oleh perusahaan adalah PPC (Aljabri & Mohammad, 2023).

PPC merupakan jenis pemasaran digital di mana pengiklan mendapat kompensasi untuk setiap kali pengguna mengklik iklan mereka (Hananto et al., 2024). Namun, dalam model iklan ini, ada risiko keamanan yang disebut *Click fraud* megacu kepada PPC yang bertujuan untuk mengurangi atau mengganggu anggaran periklanan (Thejas et al., 2021).

Klik pada iklan yang ditampilkan di internet sebanyak 36% adalah tidak sah atau palsu (Sisodia & Sisodia, 2023a). Penelitian yang dilakukan di Universitas Baltimore menemukan bahwa aktivitas *click fraud* telah menyebabkan kerugian hingga lebih dari 35$ miliar pada tahun 2020 (Aljabri & Mohammad, 2023), sehingga menjadi salah satu isu terbesar dalam industri periklanan digital. Oleh karena itu, dibutuhkan solusi berbasis teknologi untuk dapat menangani ancaman ini. Salah satu metode yang terbukti cukup efektif dalam mengatasi masalah ini adalah metode *machine learning* (Gohil & Meniya, 2020).

Machine learning dianggap sebagai solusi utama untuk mengurangi kerugian yang disebabkan oleh *click fraud* (Gabryel et al., 2021). Beberapa penelitian sebelumnya telah befokus pada prediksi *click fraud* dengan menggunakan *machine learning* (Batool & Byun, 2022), salah satu penelitian yang menggunakan model *Random Forest* (RF) berhasil mencapai tingkat akurasi sebesar 88% (Neeraja et al., 2023). Selain RF, ada juga peneliti lainnya yang menggunakan model *Decision Tree* (DT) mendapatkan akurasi 82%, *Support Vector Machine* (SVM) mendapatkan akurasi 82%, *Naive Bayes* (NB) mendapatkan akurasi 79%, dan *Neural Network* (NN) mendapatkan akurasi 83% (Aljabri & Mohammad, 2023), model-model tersebut masih kurang optimal karena terdapat masalah dalam proses pengembangan model. Salah satu masalahnya adalah data yang tidak seimbang.

Data tidak seimbang adalah kondisi di mana distribusi kelas dalam *dataset* tidak merata, satu kelas memiliki jumlah *instance* yang jauh lebih banyak (mayoritas) atau lebih sedikit dibandingkan dengan kelas lainnya (minoritas) (Qadrini et al., 2022). Kondisi ini akan menjadi tantangan dimana model akan cenderung berpihak kepada kelas mayoritas dibandingkan dengan kelas minoritas (Sisodia & Sisodia, 2023b). Hal ini dapat menyebabkan hasil prediksi yang salah atau tidak akurat karena data kelas minoritas seringkali diprediksi sebagai kelas mayoritas atau sebaliknya.

Penelitian sebelumnya telah mencoba mengatasi ketidakseimbangan data ini dengan menggunakan strategi *undersampling*, yaitu strategi untuk mengatasi ketidakseimbangan data dengan cara mengeliminasi sebagian jumlah *instance* pada kelas mayoritas (Mohammed et al., 2020). Seperti penggunaan metode *Random Under Sampling* (RUS) sebagai salah satu teknik dari strategi *undersampling* untuk penerapan model dalam prediksi *click fraud*, yang menghasilkan akurasi dalam kisaran 80%-84% pada beberapa model (Aljabri & Mohammad, 2023), menunjukkan hasil yang cukup baik dalam memprediksi *click fraud*. Namun, metode ini dapat menyebabkan hilangnya informasi penting dari data mayoritas yang dihapus secara acak. Untuk mengatasi keterbatasan tersebut dilakukan kombinasi model *K-Nearest Neighbor* (KNN) dengan teknik *undersampling* yaitu *Quad Division Prototype Selection k-Nearest Neighbor* (QDPSKNN) yang diadaptasi dari penelitian (Sisodia & Sisodia, 2022b). QDPSKNN memilih prototipe data yang sesuai untuk menjaga keseimbangan distribusi kelas yang tidak seimbang (Sisodia & Sisodia, 2022b). Cara ini berbeda dengan metode *undersampling* konvensional lainnya yang menghapus *instance* kelas mayoritas untuk menyeimbangkan *dataset*, sehingga sering menyebabkan hilangnya informasi.

Pendekatan lainnya yang digunakan untuk menangani ketidakseimbangan data yaitu *Extreme Gradient Boosting* (XGBoost), sebuah algoritma *tree boosting* yang dirancang untuk meningkatkan akurasi prediksi secara efisien (Nugraha & Irawan, 2023). XGBoost dapat menangani ketidakseimbangan data dengan menyesuaikan bobot kelas melalui parameter *scale_pos_weight*, sehingga memberikan bobot lebih besar pada kelas minoritas untuk mengurangi bias terhadap kelas mayoritas dalam data pelatihan (Ikram et al., 2023). Penerapan model XGBoost berhasil mencapai akurasi yang optimal hingga 94.53% (Thejas et al., 2021). Untuk meningkatkan performa akurasi, XGBoost juga memiliki beberapa parameter lain yang perlu disesuaikan , dan untuk mendapatkan kombinasi parameter yang

optimal diperlukan proses *hyperparameter tuning* (Dalal et al., 2022). Salah satu metode *hyperparameter tuning* yang dapat digunakan adalah *random search*. *Random search* merupakan algoritma untuk menemukan parameter terbaik secara acak untuk digunakan pada model agar dapat memprediksi data dengan akurat (Putri et al., 2023).

Berdasarkan latar belakang yang telah diuraikan diatas, penelitian ini berfokus pada pendekatan baru melalui kombinasi QDPSKNN dan XGBoost serta menggunakan *random search* sebagai metode *hyperparameter tuning* yang belum pernah digunakan oleh penelitian sebelumnya. Selain itu, penelitian ini dilengkapi dengan analisis perbandingan performa akurasi dari beberapa strategi *undersampling* konvensional lainnya. Dengan demikian , penelitian ini berjudul "**OPTIMASI DETEKSI CLICK FRAUD MELALUI KOMBINASI QUAD DIVISION PROTOTYPE SELECTION BASED KNN DAN XGBOOST**".

1.2 Identifikasi Masalah

Berdasarkan latar belakang masalah di atas, dapat diidentifikasi masalah sebagai berikut:

a. Dibutuhkan solusi berbasis teknologi untuk mengurangi dampak *click fraud* dengan cara memprediksi klik yang tidak sah
b. *Dataset* yang tidak seimbang cenderung menyebabkan model menjadi kurang akurat dalam memprediksi *click fraud*
c. Pengaruh dalam pemilihan strategi *undersampling* terhadap kinerja model dalam memprediksi *click fraud* yang masih perlu diteliti lebih lanjut

1.3 Ruang Lingkup

Ruang lingkup penelitian ini adalah sebagai berikut:

a. *Dataset* sekunder berjudul "*TalkingData AdTracking Fraud Detection Challenge*" ini dikumpulkan oleh TalkingData, salah satu platform layanan *big data* terbesar di Tiongkok (https://www.kaggle.com/c/talkingdata-adtracking-fraud-detection)
b. *Dataset* terdiri atas 100.000 sampel dengan 8 atribut
c. Pengembangan aplikasi berbasis *web* untuk mengidentifikasi *click fraud*.

1.4 Tujuan dan Manfaat

1.4.1 Tujuan

Tujuan yang ingin dicapai dalam penelitian ini adalah sebagai berikut:

a. Mengimplementasikan model XGBoost untuk memprediksi *click fraud*
b. Melakukan perbandingan akurasi antara strategi *undersampling* QDPSKNN dengan strategi *undersampling* konvensional lainnya
c. Mengembangkan aplikasi web yang dapat membantu menentukan apakah suatu klik dianggap sah atau tidak sah

1.4.2 Manfaat

Manfaat yang dapat diperoleh dalam penelitian ini adalah sebagai berikut:

a. Membantu untuk memprediksi *click fraud* lebih akurat
b. Memperoleh strategi *undersampling* dengan akurasi yang tinggi dan efektif dalam penanganan *dataset* yang tidak seimbang
c. Membantu prediksi *click fraud* secara otomatis dengan aplikasi berbasis *web*

1.5 Sistematika Penulisan

Penulisan sistematis dimaksudkan untuk mempermudah pemahaman dan penelaahan penelitian. Dalam laporan penelitian ini, bahan yang ditulis dikelompokkan menjadi beberapa sub bab, yang secara garis besar dapat dijelaskan sebagai berikut:

BAB I PENDAHULUAN

Bab ini memuat penjelasan mengenai latar belakang masalah, identifikasi masalah, ruang lingkup penelitian, tujuan dan manfaat penelitian, serta sistematika penulisan.

BAB II LANDASAN TEORI

Bab ini memuat paparan mengenai teori-teori yang berhubungan dengan penelitian ini, serta mencakup tinjauan studi.

BAB III METODE PENELITIAN

Bab ini memuat paparan terkait metode yang digunakan dalam penelitian.

BAB IV HASIL DAN PEMBAHASAN

Bab ini memuat hasil evaluasi terhadap model yang diangkat, tampilan program, dan pengujian aplikasi.

BAB V SIMPULAN DAN SARAN

Bab ini memuat penjelasan mengenai kesimpulan penelitian berdasarkan hasil evaluasi dan penerapan yang telah dilakukan, serta rekomendasi yang dapat dikembangkan untuk penelitian di masa mendatang.

BAB II

LANDASAN TEORI

2.1 Data

Data adalah kumpulan fakta, informasi, atau observasi dalam bentuk interval, nominal, ordinal, atau persentase yang dapat digunakan untuk studi, pengambilan keputusan, dan tujuan lainnya (Umar Hussain, 2023). Data dapat dikategorikan menjadi tigas jenis, diantaranya yaitu data tidak terstruktur, data terstruktur, dan data semi-terstruktur. Data yang tidak terstruktur adalah jenis data yang tidak mengikuti peraturan tertentu sehingga tidak dapat dicari menggunakan kueri, sementara itu data terstruktur merupakan data yang bisa diakses oleh manusia dan mesin menggunakan kueri, dan di antara keduanya ada data semi-terstruktur, contohnya adalah format JSON dan XML (Breitinger & Jotterand, 2023).

2.2 Data Mining

Data mining merupakan proses mengekstraksi atau menyaring informasi dalam skala yang besar melalui beberapa tahapan tertentu untuk memperoleh informasi yang bernilai dari data tersebut (Asyuti & Setyawan, 2023). Proses pengolahan ini mencakup tahapan mulai dari pengumpulan , *preprocessing* ,dan pemisahan data latih dan data uji, hingga analisis menggunakan metode *machine learning*. Hasil dari proses ini berupa pola, hubungan, atau tren baru yang berpotensi mendukung pengambilan keputusan. Selain itu, *data mining* adalah teknologi unggul yang memungkinkan prediksi tren masa depan, membantu pengguna dalam membuat keputusan yang lebih tepat (Salem et al., 2022).

2.3 Machine Learning

Machine learning adalah salah satu cabang dari kecerdasan buatan, yang memungkinkan komputer untuk belajar melalui pengalaman serta mengindetifikasi pola dan relasi *input-output* dalam data menggunakan berbagai metode statistik dan probabilisik (Chowdhury et al., 2021). *Machine learning* terdiri dari dua kategori diantaranya adalah

supervised learning, dan *unsupervised learning* (Dogan & Birant, 2021). Letak perbedaan antara *supervised learning* dengan *unsupervised learning* ialah model *supervised learning* mempelajari fungsi yang menghubungkan *input* dengan *output* berdasarkan contoh pasangan data yang telah dilabeli, sedangkan *unsupervised learning* menganalisis data tanpa label dan bekerja secara mandiri tanpa bantuan manusia.

2.4 *Website*

Website merupakan sekumpulan halaman informasi yang isinya memusatkan pada suatu topik tertentu dan biasanya terkumpul dalam sebuah *domain* atau *subdomain* yang terletak di dalam *World Wide Web* (WWW) di internet (Agustin et al., 2021). (Widia & Asriningtias, 2021) juga berpendapat bahwa *website* merupakan kumpulan halaman informasi berbasis HTML yang tersimpan di *server* dan dapat diakses dimana saja melalui browser dengan menggunakan URL serta koneksi internet.

Berdasarkan karakteristiknya, *website* dibagi menjadi dua jenis diantaranya yaitu *website* statis, yang memerlukan modifikasi langsung pada *source code* secara berkala untuk mengubah konten , dan *website* dinamis, yang tidak memerlukan perubahan *source code* untuk memperbaharui konten secara berkala, karena kontennya telah disimpan di dalam *database* (Widia & Asriningtias, 2021).

2.5 *Click fraud*

Click fraud merupakan tindakan secara sengaja yang dilakukan oleh individu atau kelompok untuk menghasilkan banyak klik yang tidak sah untuk menyebabkan kerugian finansial yang besar bagi pengiklan (Mhaske et al., 2022) *click fraud* terbagi menjadi 2 jenis yaitu *publisher click inflation* adalah ketika penerbit iklan secara sengaja meningkatkan klik yang diterima audiensnya untuk mendapatkan lebih banyak uang karena pembayaran dihitung berdasarkan jumlah klik yang diterima, dan *advertiser competition clicks* adalah

praktik yang dilakukan oleh pengiklan dengan tujuan membahayakan keuangan pesaing mereka dengan mengumpulkan banyak klik yang tidak sah (Sadeghpour & Vlajic, 2021).

2.6 *Extreme Gradient Boosting* (XGBoost)

XGBoost merupakan salah satu algoritma *machine learning* berbasis *ensemble* yang menggunakan metode *Gradient Boosting Machine* (GBM) dengan menggabungkan model-model *Decision Tree* lemah (*weak learners*) menjadi model yang kuat (*strong learner*) seperti terlihat pada Gambar 2.1 (Thejas et al., 2021). Dibandingkan dengan algoritma GBM lainnya , XGBoost menunjukkan performa dan kecepatan yang lebih unggul dalam mengerjakan tugas *machine learning* yang kompleks, dan telah banyak digunakan di industri karena mampu memecahkan masalah dengan baik (F. Zhu et al., 2022). Berikut tahapan pembuatan XGBoost yang telah dijabarkan (Dhiya UI Arif, 2024):

a. Langkah pertama adalah menghitung probabilitas awal, yaitu peluang suatu data termasuk ke dalam kelas positif berdasarkan proporsi data target secara keseluruhan yang digunakan untuk menghitung peluang dasar (*prior probability*) dari kelas target (misalnya: kelas *fraud* = 0, *non-fraud* = 1). Nilai ini digunakan sebagai dasar awal sebelum proses boosting dilakukan. Nilai probabilitas diperoleh dengan menggunakan Persamaan 1:

$$\boldsymbol{Probability(p)} = \frac{\sum(\boldsymbol{Class\ value})}{\sum(\boldsymbol{Class})} \quad (1)$$

b. Residual adalah selisih antara nilai aktual dengan nilai prediksi awal. Residual ini mencerminkan seberapa besar kesalahan prediksi, dan akan digunakan untuk membentuk model pohon selanjutnya. Menghitung nilai residual dengan mengurangi nilai *class* dari seluruh *dataset* dengan nilai probabilitas awal. Nilai residual diperoleh dengan menggunakan Persamaan 2:

$$\boldsymbol{Residual(Y) = Class\ Value - Probability} \quad (2)$$

c. Membentuk *root* awal dari pohon klasifikasi menggunakan residual yang sudah ditentukan dengan menjumlahkan semua residual tersebut. Kemudian, membentuk *leaf* dengan mengklasifikasikan berdasarkan karakteristik yang ada

d. *Similarity score* digunakan untuk mengevaluasi homogenitas data dalam satu *leaf*. Nilai *similarity* yang tinggi menunjukkan bahwa data dalam *leaf* tersebut memiliki residual yang serupa dan layak untuk dijadikan satu grup. Simbol λ adalah parameter regularisasi, dan *p* merupakan probabilitas *output* dari model saat ini (sebelum di-*update*), biasanya berasal dari *sigmoid* (logit). Nilai *similarity* diperoleh dengan menggunakan Persamaan 3:

$$\textbf{Similarity score} = \frac{(\sum \textbf{Residual})^2}{\sum(\boldsymbol{p}\ \boldsymbol{x}\ (\boldsymbol{1}-\boldsymbol{p}) + \boldsymbol{\lambda}} \quad (3)$$

e. *Gain* digunakan untuk menentukan seberapa besar peningkatan kualitas pemisahan (*split*) apabila suatu *node* dibagi menjadi dua. Semakin besar *gain*, semakin optimal pemisahan dilakukan pada *node* tersebut. Perhitungan terhadap nilai *gain* pada *left similarity* dan *right similarity* diperoleh dengan menggunakan Persamaan 4:

$$\textbf{Gain} = \frac{\boldsymbol{1}}{\boldsymbol{2}}\left[\frac{\boldsymbol{G_L^2}}{\boldsymbol{H_L}+\boldsymbol{\lambda}} + \frac{\boldsymbol{G_R^2}}{\boldsymbol{H_R}+\boldsymbol{\lambda}} - \frac{(\boldsymbol{G_L}+\boldsymbol{G_R})^2}{(\boldsymbol{H_L}+\boldsymbol{H_R})+\boldsymbol{\lambda}}\right] - \boldsymbol{\gamma} \quad (4)$$

Gain digunakan untuk memilih split terbaik. G_L dan G_R adalah gradien (residual) kiri dan kanan; H_L dan H_R adalah hessian (turunan kedua). Simbol γ adalah *threshold* pemangkasan (*pruning*).

f. *Loss function* digunakan untuk menghitung total kesalahan prediksi. Nilai ini menjadi indikator utama dalam proses Setelah seluruh percabangan dilakukan, pohon dievaluasi untuk dipangkas. Jika perbedaan antara nilai gain pada node dan nilai γ (*gamma*) kurang dari nol, maka node tersebut akan dipangkas karena

kontribusinya kecil terhadap model. *Loss function* diperoleh dengan menggunakan Persamaan 5:

$$\boldsymbol{L(\theta) = \sum_{i}^{n} (y_i - \hat{y}_i)^2} \tag{5}$$

Simbol y_i merupakan nilai aktual (label) dari sampel ke-i, sedangkan $\hat{y}_i$ merupakan nilai prediksi dari sampel ke-i

g. Berikutnya, *output value* digunakan untuk memperbaiki prediksi pada *leaf* dengan mempertimbangkan regularisasi λ. Setiap *leaf* menghasilkan prediksi berdasarkan nilai residual yang ada di dalamnya. Nilai ini akan menjadi dasar penyesuaian prediksi pada iterasi selanjutnya. *Output value* diperoleh dengan menggunakan Persamaan 6:

$$\boldsymbol{Output\ Value = \frac{(\sum Residual)}{\sum (p\ x\ (1-p) + \lambda}} \tag{6}$$

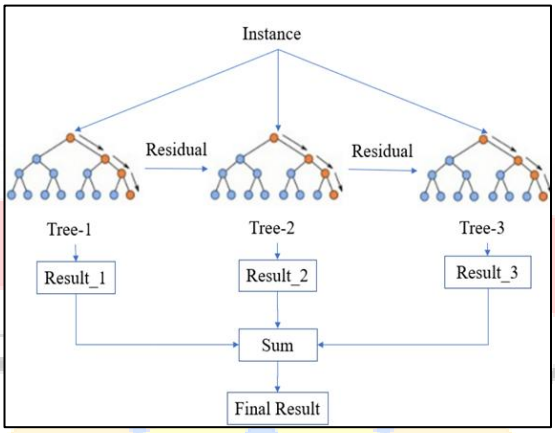
h. Langkah selanjutnya adalah penyesuaian *scale* dengan mengalikan *output value* dengan *learning rate* yang merupakan parameter untuk mengontrol seberapa besar perubahan yang diberikan oleh *tree* baru terhadap prediksi sebelumnya. Hal ini bertujuan untuk mengontrol seberapa besar perubahan prediksi yang dilakukan pada setiap iterasi. Fungsi ini mengkonversi probabilitas ke skala logit, lalu disesuaikan berdasarkan *output* value hasil *boosting* dan nilai *scale* diperoleh dengan menggunakan Persamaan 7:

$$\boldsymbol{Scale\ data(P) = \log\left(\frac{p}{1} - p\right) + (learning\ rate\ x\ output\ value)} \tag{7}$$

i. Setelah penyesuaian prediksi dilakukan, nilai logit dikonversi kembali ke probabilitas menggunakan fungsi *sigmoid* dengan menggunakan Persamaan 8:

$$\boldsymbol{New\ Probbility = \frac{e^p}{1 + e^p}} \tag{8}$$

j. Lakukan kembali langkah-langkah pada poin d dengan menggunakan berbagai kombinasi model *tree* dan kondisi *feature* yang berbeda. Tujuannya adalah untuk mengurangi nilai residual sebanyak mungkin.



Gambar 2.1 Mekanisme XGBoost
(Mhaske et al., 2022)

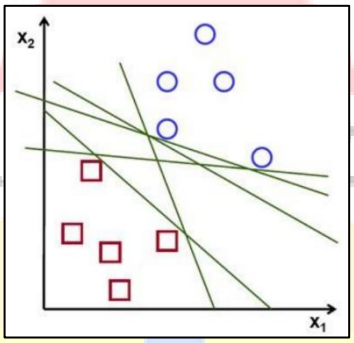
2.7 *Random Forest* (RF)

RF adalah pengembangan dari algoritma *Decision Tree* yang menggabungkan banyak pohon keputusan untuk dibentuk menjadi sekumpulan ”hutan” secara *ensemble* (Schonlau & Zou, 2020). Dalam prosesnya RF menggunakan subset data yang diambil secara acak dari *dataset* asli, kemudian menggabungkan hasil prediksi dari setiap pohon melalui rata-rata (untuk regresi) atau suara terbanyak (untuk klasifikasi), dan RF melakukan pemilihan variable secara acak pada setiap split pohon keputusan, strategi-strategi ini dilakukan untuk meningkatkan stabilitas model, mengurangi *overfitting*, dan menghasilkan akurasi yang optimal (Genuer & Poggi, 2020).

2.8 *Support Vector Machine* (SVM)

SVM merupakan algoritma *machine learning* yang memiliki kemampuan untuk memisahkan data linier dan non-linier kedalam kategori yang berbeda dengan memanfaatkan fungsi *kernel* untuk meningkatkan kinerjanya (Rosyking Lumbanraja et al., 2023). *Kernel* berfungsi untuk memetakan data input ke dalam dimensi yang lebih tinggi

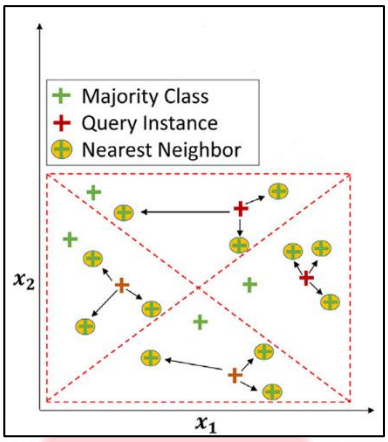
yang memungkinkan pemisahan antara dua kelas menggunakan *hyperplane* (Fajriyah Yuliati et al., 2020). *Hyperplane* adalah garis batas keputusan SVM yang digunakan untuk memisahkan dua kelas data dengan memilih batas optimal yang mendukung klasifikasi akurat pada setiap titik data (Herwinsyah & Witanti, 2022). Pada Gambar 2.2 konsep *hyperplane* dijelaskan melalui dua kelas data: positif dilambangkan dengan bentuk lingkaran , dan negatif dilambangkan dengan bentuk kotak.



Gambar 2.2 *Hyperplane* Dalam Model SVM
(Fajriyah Yuliati et al., 2020)

2.9 *Quad Division Prototype Selection Based K-Nearest Neighbor* (QDPSKNN)

QDPSKNN adalah strategi *undersampling* berbasis pemilihan prototipe yang didesain untuk menangani *dataset* yang tidak seimbang dengan memisahkan data mayoritas menjadi empat kuartil, dan digabungkan dengan algoritma *K-Nearest Neighbor* (KNN) untuk memilih sampel yang paling relevan (Sisodia & Sisodia, 2022b). KNN merupakan salah satu metode *machine learning* yang menggunakan data latih untuk mengklasifikasikan objek berdasarkan jarak yang paling dekat dengannya (Isman et al., 2021).



Gambar 2.3 Proses seleksi QDPSKNN

(Sisodia & Sisodia, 2022b)

Seperti yang terlihat pada Gambar 2.3 , KNN menghitung jarak *euclidean* dari setiap kuartil dan median sebagai titik referensi dalam memilih prototipe dengan sampel yang memiliki jarak yang paling dekat. Jarak tersebut dapat diperoleh melalui Persamaan 9:

$$\boldsymbol{dist(f_t, f_{Qi}) = \sqrt{\sum_{i=1}^{d} (f_t - f_{Q_i})^2}} \tag{9}$$

Keterangan :

d = Dimensi fitur

f_t = Median yang diperoleh menggunakan Persamaan 10:

$$\boldsymbol{Median = \begin{cases} \frac{n}{2}, & jika\ n(jumlah\ sampel)\ genap \\ \frac{n+1}{2}, & jika\ n(jumlah\ sampel)\ ganjil \end{cases}} \tag{10}$$

f_{Q_i} = Sampel dari setiap kuartil Q_i mencakup 25% dari total data mayoritas dengan menggunakan Peramaan 11:

$$\boldsymbol{Q_i = \left(\frac{i(n+1)}{4}\right) th\ value, i = 1, 2, 3} \tag{11}$$

Prototipe dari setiap kuartil digabungkan untuk membuat kumpulan data mayoritas yang lebih kecil namun masih representatif, selanjutnya kumpulan data tersebut

digabungkan dengan data minoritas yang membentuk *dataset* baru A_d dengan menggunakan Persamaan 12, dan 13:

$$\boldsymbol{d_{mJ'} = \{pp_1, pp_2, pp_3, pp_4\}} \tag{12}$$

$$\boldsymbol{A_d = d_{mJ'} \cup d_{mn}} \tag{13}$$

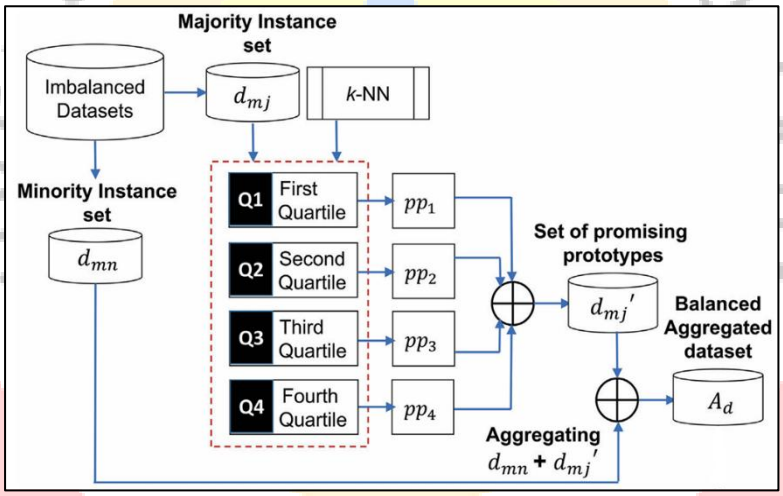
Keterangan :

pp = *Promising prototype* / sampel yang terpilih

$d_{mJ'}$ = Kumpulan data mayoritas yang diperoleh dari kuartil

d_{mn} = Kumpulan data minoritas

A_d = *Dataset* seimbang dengan gabungan dari $d_{mj'}$ dan d_{mn}



Gambar 2.4 Alur Kerja Metode QDPSKNN

(Sisodia & Sisodia, 2022b)

Gambar 2.4 menunjukkan alur kerja metode QDPSKNN yang akan digunakan pada penelitian ini.

2.10 *Random Undersampling* (RUS)

RUS merupakan metode *undersampling* yang dilakukan dengan menghitung jumlah perbedaan antara kelas mayoritas dan minoritas, kemudian secara acak menghapus kelas mayoritas selama proses iterasi hingga jumlahnya seimbang dengan kelas minoritas (Bhatia

et al., 2020) . Meskipun RUS tergolong cepat dalam proses pelatihannya, metode ini dapat menurunkan kualitas data mayoritas akibat dari pola penghapusan sampel yang dilakukan secara acak (Fujiwara et al., 2020).

2.11 *Nearmiss Undersampling* (NMU)

NMU merupakan salah satu teknik dari strategi *undersampling* yang menyeimbangkan *dataset* dengan cara mengurangi sampel data mayoritas berdasarkan jarak terdekat antara sampel mayoritas dengan minoritas, dimana pendekatan yang digunakan untuk menghitung jarak tersebut adalah *K-Nearest Neighbor* (KNN) (Asih & Rianto, 2024; Oladunni et al., 2021). NMU memiliki 3 jenis versi, diantaranya adalah versi 1, versi 2, dan versi 3, di mana versi 1 memilih sampel kelas mayoritas berdasarkan rata-rata jarak terpendek ke N sampel tedekat dari kelas minoritas, versi 2 memilih sampel kelas mayoritas berdasarkan jarak terjauh ke N sampel terdekat dari kelas minoritas, dan versi 3 memilih sampel kelas mayoritas untuk setiap sampel terdekat dengan kelas minoritas (Sitorus et al., 2023). Penelitian ini akan menggunakan NMU versi 3.

2.12 *Random Search*

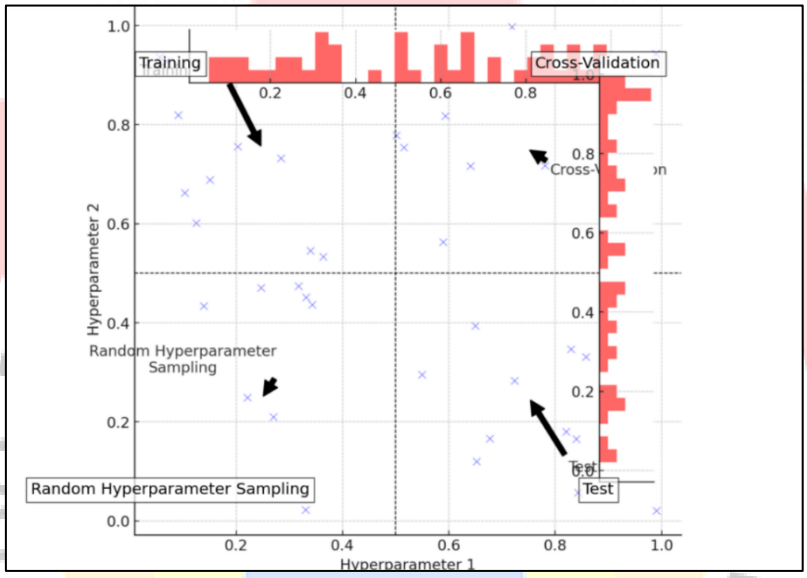
Random search merupakan metode untuk mencari kombinasi *hyperparameter* yang optimal secara acak, di mana proses ini dilakukan berulang kali hingga jumlah iterasi yang telah ditentukan tercapai (Zahedi et al., 2021). Pendekatan ini mengevaluasi model yang dihasilkan pada set validasi untuk menilai performa dari setiap kombinasi *hyperparameter* (Vo et al., 2023). Kelebihan *random search* yaitu mudah untuk implementasi, dan performa yang sering kali lebih baik dalam ruang *hyperparameter* berdimensi rendah, namun kelemahannya adalah setiap evaluasi bersifat independen, yang berarti bahwa hasil evaluasi sebelumnya tidak dipertimbangkan saat memilih kombinasi *hyperparameter* berikutnya (Ridwan & Utami, 2024). Berikut merupakan langkah-langkah dari proses *random search*:

a. Menentukan rentang nilai untuk masing-masing *hyperparameter*

b. Memilih satu set konfigurasi *hyperparameter* secara acak berdasarkan jumlah iterasi yang telah ditentukan

c. Mengevaluasi performa model menggunakan *Cross Validation* untuk setiap kombinasi *hyperparameter* dan menghitung nilai akurasi

d. Memilih kombinasi *hyperparameter* dengan performa terbaik berdasarkan nilai akurasi



Gambar 2.5 Ilustrasi *Random Search*
(Ridwan & Utami, 2024)

Gambar 2.5 menunjukkan ilustrasi strategi di mana kumpulan kombinasi *hyperparameter* acak digunakan untuk menghasilkan akurasi yang terbaik dari sebuah model. Penelitian ini akan menggunakan metode *random search* yang tersedia dalam modul *Scikit-learn* dengan nama *RandomizedSearchCV*.

2.13 *Flowchart*

Flowchart merupakan diagram atau bagan alur yang menggunakan simbol standar untuk menunjukkan alur proses atau prosedur (Chaudhuri, 2020). *Flowchart* sangat membantu untuk membuat algoritma lebih mudah dipahami oleh orang lain, terutama programmer yang mengerjakan program dengan representasi grafisnya yang ditunjukkan melalui simbol-simbol. Setiap simbol memberikan makna tertentu, dan berikut adalah

simbol-simbol yang sering digunakan dalam proses pembuatan *flowchart* sebagaimana ditampilkan pada Tabel 2.1.

Tabel 2.1 Simbol Standar *Flowchart*

Simbol	Nama	Deskripsi
	*Start/End*	Menunjukkan Dimana sebuah proses dimulai atau berakhir
	*Input/Output*	Menandakan proses input dan output data
	*Process*	Menunjukkan proses yang dilakukan oleh komputer
	*Predefined Process*	Menunjukkan pelaksanaan suatu bagian (sub-program)
	Flow Line	Digunakan untuk menghubungkan simbol
	*Decision*	Menunjukkan situasi tertentu yang akan menghasilkan dua pilihan jawaban, yaitu ya dan tidak
	*Document*	Menunjukkan bahwa input berasal dari dokumen dan output ke dokumen

2.14 *Python*

Pada tahun 1990, seorang matematikawan asal Belanda bernama Guido Van Rossum merilis *Python* untuk pertama kalinya (Silaparasetty, 2020). Nama *Python* terinspirasi dari acara komedi BBC tahun 1970-an yang terkenal yaitu "*Monty Python's Flying Circus*" (Python Sofware Foundation, 2023). Sebagai bahasa pemrograman yang bersifat *open source*, *Python* juga memiliki popularitas tinggi dalam bidang ilmiah seperti *machine learning* dan *data science* (Saitoh, 2021). *Python* adalah bahasa pemrograman yang sangat baik untuk *machine learning* karena ada banyak modul yang mendukung *machine learning*, seperti *Pandas*, *Numpy*, *Scikit-learn*, *Imbalanced-learn*, *Matplotlib*, *Seaborn*, dan yang lainnya (Silaparasetty, 2020).

2.15 *Scikit-learn*

Scikit-learn merupakan modul atau sebuah *library Python* untuk *machine learning* yang awalnya dikembangkan oleh David Cournapeau pada tahun 2007 sebagai proyek *Google Summer of Code* (GSoC). Pada tahun 2010, proyek ini diambil alih oleh Fabian

Pedregosa, Gaël Varoquaux, Alexandre Gramfort dan Vincent Michel, mereka merilis versi publik pertama *Scikit-learn* pada 1 Febuari 2010. Semenjak saat itu, *Scikit-learn* terus mengalami pembaruan secara berkala yang dibantu oleh komunitas internasional. Modul ini memanfaatkan fungsionalitas dan struktur data yang disediakan oleh SciPy untuk digunakan dalam *machine learning* (Buitinck et al., 2024).

2.16 *Imbalanced-learn*

Imbalanced-learn merupakan sebuah *library Python* yang menyediakan berbagai teknik *re-sampling* untuk mengatasi ketidakseimbangan antar kelas dalam suatu *dataset*. *Library* ini kompatibel dengan *Scikit-learn*, yang membuatnya mudah untuk diintegrasikan ke dalam pipeline *machine learning*. Proyek *Imbalanced-learn* dikembangkan pada Agustus 2014 oleh Fernando Nogueira dengan fokus awalnya pada implementasi *Synthetic Minority Oversampling Technique* (SMOTE). Guillaume Lemaitre, Dayvid Victor, dan Christos Aridas juga membantu untuk mengembangkan metode tambahan untuk *undersampling* dan *oversampling*, serta melakukan perubahan besar pada API sehingga *library* ini sepenuhnya dapat kompatibel dengan *Scikit-learn* (Lemaître et al., 2024).

2.17 *Black Box Testing*

Black Box Testing merupakan salah satu cara untuk menguji *software* yang berfokus pada pengujian fungsi yang sesuai dengan kebutuhan yang telah ditetapkan, tanpa memperhatikan struktur kode dari *software* tersebut (Yulistyanti et al., 2022). Keuntungan memakai *Black Box Testing* adalah: (1) Penguji tidak perlu memiliki kemampuan khusus tentang bahasa pemrograman yang diterapkan; (2) Pengujian dilakukan dari perspektif pengguna, sehingga dapat menemukan ambiguitas atau ketidaksesuaian dalam spesifikasi kebutuhan; (3) Ada keterkaitan antara *programmer* dan *tester*, yang mendukung kerjasama selama proses pengujian (Rahadi & Vikasari, 2020).

2.18 *Confusion Matrix*

Confusion Matrix adalah tabel yang digunakan untuk menilai kinerja algoritma klasifikasi serta merangkum hasilnya. Selain itu, *Confusion Matrix* juga dapat digunakan sebagai tabel ringkasan tentang jumlah prediksi yang tepat atau salah yang dibuat oleh model klasifikasi. *True-Positive* (TP) dan *True-Negative* (TN) menunjukkan hasil ketika model mengklasifikasikan data dengan hasil yang benar, sedangkan *False-Positive* (FP) dan *False-Negative* (FN) menunjukkan hasil ketika model mengklasifikasikan data dengan hasil yang salah (Firdaus et al., 2023).

Tabel 2.2 *Confusion Matrix*

		Actual Class	
		True	False
Prediction	True	TP	FP
	False	FN	TN

Tabel 2.2 menunjukkan ilustrasi *Confusion Matrix* dalam bentuk tabel. Setelah mendapatkan nilai TP, TN, FP, dan FN dari model, nilai-nilai tersebut digunakan untuk menghitung sejumlah metrik evaluasi yang akan dijelaskan sebagai berikut: (Borah & Sarma, 2023; Niu et al., 2023; Sisodia & Sisodia, 2022a, 2023a)

a. *Accuracy* merupakan rasio antara jumlah sampel yang diklasifikasikan dengan benar terhadap seluruh sampel yang diuji. Nilai *accuracy* dapat diperoleh menggunakan Persamaan 14:

$$\boldsymbol{Accuracy} = \frac{(\boldsymbol{TP} + \boldsymbol{TN})}{(\boldsymbol{TP} + \boldsymbol{FP} + \boldsymbol{TN} + \boldsymbol{FN})} \quad (14)$$

b. *Precision* merupakan persentase jumlah sampel yang diprediksi sebagai kelas positif dengan benar terhadap seluruh jumlah sampel yang diprediksi sebagai kelas positif. Nilai *precision* dapat diperoleh menggunakan Persamaan 15:

$$\boldsymbol{Precision} = \frac{\boldsymbol{TP}}{(\boldsymbol{TP} + \boldsymbol{FP})} \quad (15)$$

c. *Recall* merupakan persentase dari total kelas positif yang diprediksi dengan benar oleh model dibandingkan dengan total kelas positif yang sebenarnya. Nilai *recall* dapat diperoleh menggunakan Persamaan 16:

$$\boldsymbol{Recall} = \frac{\boldsymbol{TP}}{(\boldsymbol{TP} + \boldsymbol{FN})} \quad (16)$$

d. *F1-Score* merupakan metrik kinerja yang mengukur keseimbangan antara *precision* dengan *recall* dalam sebuah model. Nilai *f1-score* dapat diperoleh menggunakan Persamaan 17:

$$\boldsymbol{F1 - Score} = 2 \times \frac{\boldsymbol{Recall} \times \boldsymbol{Precision}}{\boldsymbol{Recall} + \boldsymbol{Precision}} \quad (17)$$

2.19 Kurva ROC-AUC

Kurva ROC merupakan sebuah grafik yang menggambarkan keterkaitan antara sensitivitas atau *True Positive Rate* (TPR) di sumbu Y dan *False Positive Rate* (FPR) di sumbu X untuk menghasilkan nilai AUC (Nahm, 2022). AUC adalah metrik yang menggambarkan kemampuan model membedakan kelas-kelas untuk menunjukkan tingkat separabilitas, atau sejauh mana model dapat membedakan kelas positif dan negatif (Sisodia & Sisodia, 2023a). Berikut panduan interprestasi AUC yang ditunjukkan pada Tabel 2.3.

Tabel 2.3 Kategori AUC

Nilai AUC	Kategori
0.90-1.00	*Excellent Classification*
0.80-0.90	*Good Classification*
0.70-0.80	*Fair Classification*
0.60-0.70	*Poor Classification*
<0.60	*Failure*

2.20 Tinjauan Studi

Deteksi *click fraud* dalam sistem periklanan digital berbasis *Pay-Per-Click* (PPC) telah menjadi perhatian utama dalam penelitian *Machine Learning* (ML). Seiring meningkatnya ancaman *click fraud*, berbagai pendekatan berbasis ML dan teknik pengolahan data telah dikembangkan untuk meningkatkan akurasi deteksi. Literatur yang tersedia mencakup pendekatan dari *supervised learning*, *deep learning* (DL), *ensemble*, serta teknik *balancing data* untuk mengatasi ketidakseimbangan dataset yang sering menjadi tantangan utama dalam penelitian ini.

Salah satu penelitian melakukan deteksi *click fraud* melalui penggunaan model ML, dengan membandingkan beberapa model menggunakan dataset dari Beacon, di mana *Random Forest* (RF) menunjukkan akurasi terbaik sebesar 84% dibandingkan model lain seperti *Decision Tree* (DT), *Support Vector Machine* (SVM), *Naive Bayes* (NB), dan *Neural Network* (NN) (Aljabri & Mohammad, 2023). Selain itu model RF juga diuji pada dataset dari Kaggle menghasilkan akurasi tertinggi sebesar 88% (Neeraja et al., 2023).

Pendekatan berbasis *ensemble learning* juga telah diterapkan dengan menguji *Adaptive Boosting* (Adaboost) dan *Extreme Gradient Boosting* (XGBoost), di mana XGBoost menunjukkan performa superior dengan akurasi 96.2% (Mhaske et al., 2022). Pendekatan *ensemble* lain seperti arsitektur *Convolutional Neural Network, Bidirectional Long Short-Term Memory, and Random Forest* (CNN-BiLSTM-RF) yang menggabungkan DL dengan metode *ensemble* juga digunakan dalam mendeteksi *click fraud* yang berhasil mencapai *accuracy* 99.58%, *precision* 99.60%, dan *f1-score* 99.60% (Batool & Byun, 2022). CNN juga dimanfaat sebagai sebagai model klasifikasi akhir berdasarkan *data tensor* yang telah diproses dalam penelitian yang berjudul "*Click fraud Detection of Online Advertising-LSH Based Tensor Recovery Mechanism*" (F. Zhu et al., 2022), yang membahas tentang pengembangan mekanisme deteksi click fraud menggunakan *Tensor Transformation with*

Locality-Sensitive Hasing (TT-LSH). *Dataset* yang digunakan dalam penelitian ini mencakup 18 juta klik yang dikumpulkan selama 4 hari, dan terdiri dari 7 fitur yaitu *IP address*, *time*, *device*, *os*, *app*, *publisher*, dan *click frequency*. Hasil penelitian menunjukkan kombinasi TT-LSH dengan CNN memberikan kinerja yang cukup baik menghasilkan *precision* sebesar 90%, *recall* sebesar 80%,dan ROC AUC sebesar 96.6%.

Teknik DL lainnya juga telah diterapkan dalam studi mengeksplorasi penggunaan *Convolutional Neural Network* (CNN) dan *Deep Convolution Neural Network* (DCNN) untuk mendeteksi aktivitas bot melalui pola gerakan mouse dan melalui penerapan transformasi fitur data klik menjadi representasi gambar 2 dimensi. Model CNN menggunakan 2 *dataset*, mencakup 1 *dataset* gerakan mouse dari 150 subjek manusia dengan jumlah sebanyak 10.230 sampel, dan 1 *dataset bot* yang meniru perilaku manusia dengan jumlah sebanyak 11.000 sampel. Sedangkan DCNN menggunakan dataset *Fraud Detection in Mobile Advertising* (FDMA) yang terdiri dari 1.048.575 klik dan mencakup 3 label yaitu *fraud*, *observation*, dan ok. Kedua pendekatan ini berhasil mencapai tingkat akurasi yang sangat tinggi, yaitu CNN 99.34% untuk *True Positive Rate*, dan 99.20% untuk *True Negative Rate* dan DCNN sebesar 79.8% untuk *precision* (Niu et al., 2023) (Sisodia & Sisodia, 2022a).

Selain pendekatan DL berbasis CNN dan DCNN, penelitian lain mengusulkan tiga model *Neural Network* berbasis *Multilayer Perceptron* (MLP) dengan arsitektur yang berbeda: model pertama menggunakan MLP *one-hot encoding*, model kedua menggunakan MLP *embedding layer*, dan model ketiga menggunakan MLP *embedding layer* yang terpisah. Studi ini memanfaatkan dua *dataset* yang dikumpulkan secara mandiri, yaitu *dataset monitoring click* dengan 300.000 kunjungan pengguna situs *e-commerce* dan *dataset monitoring lead* dengan 263.000 kunjungan pengguna situs lembaga keuangan. Hasil penelitian menunjukkan bahwa model ketiga memberikan performa terbaik, dengan

accuracy, *precision*, *recall*, dan *f1-score* mencapai 99.5% untuk *dataset monitoring lead* dan 98.7% untuk *dataset monitoring click*. Hal ini menegaskan bahwa pendekatan berbasis DL, baik melalui CNN, DCNN, maupun MLP, dapat secara efektif meningkatkan akurasi deteksi aktivitas *bot* dan *click fraud* (Gabryel et al., 2021).

Selain pendekatan berbasis DL dan *ensemble learning*, metode *graph mining* juga mulai diterapkan dalam mendeteksi *click fraud* dan aktivitas *botnet* menggunakan *graph-based clustering* untuk mendeteksi *botnet* dalam lalu lintas jaringan. Sebanyak 3 algoritma diusulkan dalam penelitian ini: *Markov Cluster* (MCL), *Infomap*, dan kombinasi *PageRank* dengan HDBSCAN. Penelitian ini menggunakan *dataset* sintetik dari *testbed* sebagai data latih, dan *dataset* dari CTU-13 untuk data uji. evaluasi kinerja melalui *dataset* CTU-13 menunjukkan hasil *accuracy* 99.91%, *recall* 100%, namun *precision* yang rendah 3.33% (Borah & Sarma, 2023).

Selain itu penerapan pendekatan *transfer learning* dengan pengembangan model *Feature Interaction Network* (FINet) berbasis jaringan saraf *convolutional* satu dimensi (1-D CNN) juga dilakukan. Model ini dilatih pada *dataset Talkingdata AdTracking Fraud Detection* (TDA) dan selanjutnya diuji pada *dataset Fraud Detection in Mobile Advertising* (FDMA). Hasil pendekatan ini menghasilkan akurasi terbaik dengan rata-rata *precision* 85.6%, *recall* 84.4%, *f-measure* 83.7%, dan AUC 93.7% (Sisodia & Sisodia, 2023a).

Beberapa penelitian telah mengusulkan pendekatan *hybrid* untuk meningkatkan akurasi deteksi *click fraud* seperti *Cascaded Forest and XGBoost* (CFXGB), yang mengombinasikan Random Forest dengan teknik feature transformation untuk meningkatkan akurasi dibandingkan model tunggal. Penelitian ini menggunakan 3 *dataset* dari Kaggle yaitu TalkingData, Avazu, dan Kad. *Dataset* TalkingData terdiri dari 8 atribut, dataset Avazu terdiri dari 24 atribut, dan *dataset* Kad terdiri dari 10 atribut. Hasil penelitian menunjukkan bahwa kombinasi CFXGB berhasil memberikan akurasi yang lebih besar

dibandingkan dengan menggunakan model tunggal yaitu *Cascaded Forest* (Thejas et al., 2021). Sejalan dengan itu, pengembangan pendekatan *hybrid learning* yang mengintegrasikan *rules-based techniques* dengan *K-Means Clustering* dan DT. Peneliti menggunakan *dataset* yang diperoleh dari *log* aktivitas *web* pada halaman *web* milik *publisher* iklan, yang mencakup informasi seperti *timestamp*, *HTTP referer*, *user agent*, *IP address*, serta berbagai jenis aktivitas yang dilakukan oleh pengguna. Hasil akhir pada penelitian ini menunjukkan bahwa model *hybrid learning* yaitu *clustering* dengan *classification* yang dikembangkan dapat mencapai tingkat akurasi yang optimal sebesar 93.75% (Gajewski et al., 2024).

Salah satu tantangan utama dalam mendeteksi *click fraud* adalah distribusi data yang tidak seimbang antara klik sah dan klik tidak sah. Beberapa penelitian telah mengusulkan metode *balancing data* untuk meningkatkan akurasi model. Metode *undersampling* seperti *Random Under Sampling* (RUS) telah digunakan untuk mengatasi ketidakseimbangan ini. RUS dengan kombinasi XGBoost yang diuji pada dataset Beacon dengan distribusi kelas yang tidak seimbang terdiri 5 fitur utama, dan memiliki 2.417 sampel diantaranya adalah 83 sampel *data bot*, dan 2.334 sampel data manusia berhasil menghasilkan menghasilkan akurasi dalam kisaran 80%-84% (Aljabri & Mohammad, 2023). Namun, metode ini sering kali menghilangkan informasi penting dari kelas mayoritas.

Sebagai alternatif, *Quad Division Prototype Selection Based K-Nearest Neighbor* (QDPSKNN) dikembangkan oleh (Sisodia & Sisodia, 2022b). Metode ini membagi kelas mayoritas menjadi empat kuartil sebelum memilih sampel yang paling relevan untuk pelatihan model KNN. Dalam studi ini, pendekatan QDPSKNN diuji pada 15 *dataset* dengan distribusi kelas yang tidak seimbang dan terbukti lebih efektif dibandingkan metode *undersampling* konvensional seperti *NearMiss* dan *Condensed Nearest Neighbor*. Pada dataset FDMA, metode ini berhasil mencapai tingkat *reduction rate* sebesar 93.9%, dan

mampu memberikan hasil yang lebih baik dari metrik *precision*, *recall*, *f-measure*, dan *g-mean*.

Selain *undersampling*, beberapa penelitian telah mengeksplorasi teknik *oversampling*, seperti penggunaan *Synthetic Minority Over-sampling Technique* (SMOTE) dalam mendeteksi *click fraud* (Batool & Byun, 2022) (Firdaus et al., 2023) . Penelitian berjudul "Model Deteksi *Botnet* Menggunakan Algoritma *Decision Tree* Dengan Untuk Mengidentifikasi Serangan *Click fraud*" membahas tentang penggunaan algoritma *Classification and Regression Tree* (CART) untuk mendeteksi *click fraud* oleh *botnet* dengan pendekatan *data sampling* dengan 2 metode yaitu SMOTE dan RUS, dengan memanfaatkan *dataset* CTU-13 skenario 9 yang terdiri dari 1.054.119 baris data lalu lintas jaringan dan mencakup 3 label *botnet*, normal, dan *background*. Hasil penelitian menunjukkan bahwa kombinasi SMOTE dan RUS menunjukkan akurasi yang optimal dengan tingkat *accuracy* mencapai 99.97%, dan 99.96% untuk *precision* dan *recall* (Firdaus et al., 2023).

Penelitian dalam studi berjudul "*Stacked Generalization Architecture for Predicting Publisher Behaviour from Highly Imbalanced User-Click Data Set for Click Fraud Detection*" juga menggunakan teknik *oversampling* dalam menangani *dataset* FDMA yang tidak seimbang dalam deteksi *click fraud*. Studi ini mengusulkan arsitektur *stacked generalization*, yang terdiri dari dua tahap utama, yaitu *Stacked Generalization for Resampling* (SGR) dan *Stacked Generalization for Classification* (SGC). Dalam tahap SGR, penelitian ini menggunakan 6 teknik *oversampling*, yakni SMOTE, SMOTETomek, BorderlineSMOTE-1, BorderlineSMOTE-2, SVMSMOTE, dan ADASYN, untuk meningkatkan representasi kelas minoritas. Hasil penelitian menunjukkan pendekatan *stacked generalization architecture* yang diusulkan menghasilkan akurasi yang cukup baik,

dengan *precision* sebesar 66.3%, *recall* sebesar 62.6%, dan *f1-score* sebesar 63.9% (Sisodia & Sisodia, 2023b).

2.21 Kesimpulan Tinjauan Studi

Penelitian terkait deteksi *click fraud* telah berkembang pesat dengan mengadopsi berbagai pendekatan, termasuk metode ML, DL, dan teknik berbasis *graph*. Penggunaan model tradisional ML sering kali mengalami keterbatasan dalam menangani data yang tidak seimbang. Untuk mengatasi tantangan ini, berbagai teknik telah diusulkan, seperti penggunaan model *ensemble* untuk ekstraksi fitur otomatis dan klasifikasi yang lebih akurat, hingga penerapan *stacked generalization*. Selain itu, pendekatan berbasis *tensor* telah memperkenalkan mekanisme transformasi data untuk menangkap hubungan intrinsik dalam data multi-dimensi, dan teknik *oversampling* berbasis SMOTE hingga *undersampling* berbasis RUS telah meningkatkan performa model pada *dataset* yang tidak seimbang. Teknik berbasis *graph* seperti untuk mendeteksi *botnet Peer-to-Peer* juga memberikan kontribusi signifikan dalam mengindetifikasi pola *click fraud*. Secara keseluruhan, pendekatan-pendekatan tersebut terbukti efektif untuk meningkatkan akurasi dalam deteksi *click fraud* pada iklan *online*.

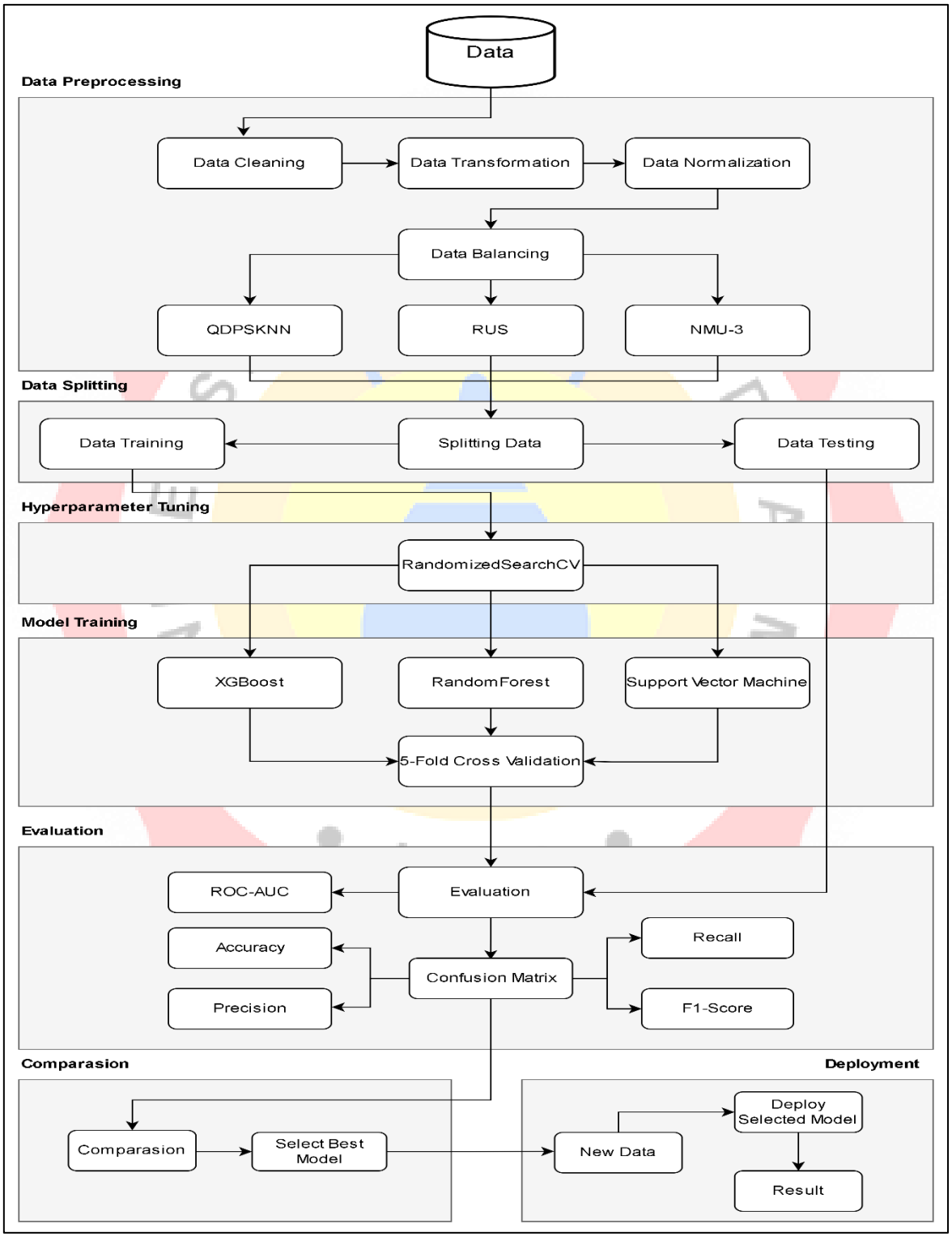
Berdasarkan studi sebelumnya, berbagai algoritma seperti XGBoost, RF, SVM dan lainnya telah digunakan untuk memprediksi *click fraud*, serta menerapkan beberapa teknik *data sampling* seperti SMOTE, RUS, QDPSKNN, dan metode serupa lainnya. Pada penelitian ini, peneliti akan melakukan perbandingan kinerja model menggunakan algoritma XGBoost dan beberapa teknik *data sampling* yaitu QDPSKNN, RUS, dan *Nearmiss* yang kemudian akan digunakan untuk memprediksi *click fraud*. Pemilihan XGBoost ini dilandasi oleh kemampuannya dalam mengatasi *dataset* yang tidak seimbang (Ikram et al., 2023) serta tinjauan studi sebelumnya yang telah membuktikan efektivitasnya dalam deteksi *click fraud* (Aljabri & Mohammad, 2023; Mhaske et al., 2022; Thejas et al., 2021). Penelitian ini

bertujuan untuk mengoptimalkan model deteksi *click fraud* dengan menggunakan fungsi *RandomizedSearchCV* sebagai metode *hyperparameter tuning* yang belum pernah dilakukan oleh studi sebelumnya sehingga penelitian ini diharapkan dapat meningkatkan akurasi model menjadi lebih optimal.

BAB III

METODE PENELITIAN

3.1 Kerangka Pemikiran



Gambar 3.1 Kerangka Pemikiran

Gambar 3.1 menunjukkan alur sistematis pengolahan data hingga *deployment* model untuk memprediksi *click fraud*. Proses dimulai dengan tahap *preprocessing*, dimana *dataset* melalui proses *data cleaning* untuk menghilangkan atribut dan nilai yang tidak penting, data *transformation* untuk memastikan format data dapat dibaca oleh model, dan *data normalization* untuk meningkatkan performa pelatihan model. Pada tahap ini juga dilakukan *data balancing* dengan menggunakan 3 metode yaitu QDPSKNN, RUS, dan NMU-3 untuk mengatasi ketidakseimbangan kelas dalam *dataset*.

Data yang telah melalui tahap *preprocessing* dilanjutkan ke proses *data splitting*, dimana data dibagi menjadi 2 bagian yaitu *data training* dan *data testing*. Kemudian, *data training* digunakan untuk proses *hyperparameter tuning* menggunakan *RandomizedSearchCV* untuk menemukan kombinasi parameter terbaik untuk 3 model yang akan diuji, yaitu XGBoost, RF, dan SVM. Sedangkan *data testing* digunakan untuk mengevaluasi performa model. Hasil evaluasi dari setiap model dibandingkan dalam tahap *comparasion*, lalu model dengan performa terbaik dipilih untuk kemudian diimplementasikan pada aplikasi berbasis web untuk mendeteksi *click fraud*.

3.2 *Data Understanding*

Peneliti menggunakan *dataset* berjudul "*TalkingData AdTracking Fraud Detection Challenge*" (TDA) yang berisi 100.000 sampel klik dan 8 attribute yaitu :

a. *IP* : Alamat *IP* dari klik dengan rentang nilai (9 – 364.757)

b. *App* : Id aplikasi pemasaran dengan rentang nilai (1 - 551)

c. *Device* : Id tipe perangkat dengan rentang nilai (0 – 3.867)

d. *Os* : Id versi sistem operasi dengan rentang nilai (0 - 866)

e. *Channel* : Id saluran penerit iklan dengan rentang nilai (3 - 498)

f. *Click_time* : Waktu terjadinya klik (UTC) dengan rentang nilai (06/11/2017 16:00 – 09/11/2017 15:59)

g. *Attributed_time* : Waktu terjadinya pengunduhan aplikasi dengan rentang nilai (06/11/2017 17:19 – 09/11/2017 15:28)

h. *Is_attributed* : Label yang menunjukkan apakah aplikasi diunduh dengan rentang nilai (0 - 1), nilai 0 sebagai *click fraud* dengan jumlah 99.773 dan nilai 1 sebagai interaksi manusia dengan jumlah 227

Dataset yang digunakan adalah *dataset* sekunder yang diperoleh dari TalkingData, salah satu platform layanan *big data* terbesar di Tiongkok https://www.kaggle.com/c/talkingdata-adtracking-fraud-detection. Tabel 3.1 menunjukkan informasi lebih detail tentang *dataset* TDA.

Tabel 3.1 Informasi *Dataset*

No	*Column*	*Non-Null Count*	*Data Type*
1	Ip	100.000	Int64
2	App	100.000	Int64
3	Device	100.000	Int64
4	Os	100.000	Int64
5	Channel	100.000	Int64
6	Click_time	100.000	Object
7	Attributed_time	227	Object
8	Is_attributed	100.000	Int64

3.3 *Data Preprocessing*

3.3.1 *Data Cleaning*

Berdasarkan hasil pemeriksaan pada *dataset* TDA , tidak ditemukan adanya nilai yang hilang, dan inkonsistensi data, tetapi ada fitur yang tidak diperlukan yaitu *attributed_time* karena tidak memiliki pengaruh penting dalam pembuatan model (Batool & Byun, 2022). Dengan demikian, fitur tersebut akan dihapus.

Tabel 3.2 Informasi *Dataset* Sebelum Proses *Data Cleaning*

No	*Column*	*Data Type*
1	Ip	Int64
2	App	Int64
3	Device	Int64
4	Os	Int64
5	Channel	Int64
6	Click_time	Object
7	Attributed_time	Object
8	Is_attributed	Int64

Tabel 3.3 Informasi *Dataset* Sesudah Proses *Data Cleaning*

No	*Column*	*Data Type*
1	Ip	Int64
2	App	Int64
3	Device	Int64
4	Os	Int64
5	Channel	Int64
6	Click_time	Object
7	Is_attributed	Int64

Tabel 3.2, dan Tabel 3.3 menunjukkan informasi perbedaan *dataset* sebelum dan sesudah *dataset* melalui proses *data cleaning*.

3.3.2 *Data Transformation*

Pembuatan model *click fraud* menggunakan *machine learning* memerlukan data input berupa angka (numerik). Tabel 3.3 menunjukkan hampir seluruh atribut telah bertipe numerik, menyisakan 1 atribut yaitu *click_time* yang masih berupa *object*. Atribut *click_time* akan dikonversi menjadi tipe data *DateTime* yang selanjutnya akan dipecah menjadi 4 komponen yang lebih detail yaitu *day*, *hour*, *minute*, dan *second* dimana masing-masing atribut tersebut bertipe data numerik. Selanjutnya atribut

click_time akan dihapus karena telah direpresentasikan oleh 4 atribut baru untuk menghindari redudansi pada *dataset*. Tabel 3.4, dan Tabel 3.5 menunjukkan informasi perbedaan *dataset* sebelum dan sesudah *dataset* melalui proses *data transformation*.

Tabel 3.4 Informasi *Dataset* Sebelum Proses *Data Transformation*

No	*Ip*	*App*	*Device*	*Os*	*Channel*	*Click_time*	*Is_attributed*
0	87540	12	1	13	497	2017-11-07 09:30:38	0
1	105560	25	1	17	259	2017-11-07 13:40:27	0
2	101424	12	1	19	212	2017-11-07 18:05:24	0
3	94584	13	1	13	477	2017-11-07 04:58:08	0
4	68413	12	1	1	178	2017-11-09 09:00:09	0
…	…	…	…	…	…	…	…
99995	124883	11	1	19	122	2017-11-09 13:25:41	0
99996	85150	9	1	13	244	2017-11-07 11:25:43	0
99997	18839	3	1	13	19	2017-11-08 11:38:42	0
99998	114276	15	1	12	245	2017-11-08 17:55:21	0
99999	119349	14	1	15	401	2017-11-07 14:32:27	0

Tabel 3.5 Informasi *Dataset* Sesudah Proses *Data Transformation*

No	*Ip*	*App*	*Device*	*Os*	*Channel*	*Day*	*Hour*	*Minute*	*Second*	*Is_attributed*
0	87540	12	1	13	497	7	9	30	38	0
1	105560	25	1	17	259	7	13	40	27	0
2	101424	12	1	19	212	7	18	5	24	0
3	94584	13	1	13	477	7	4	58	8	0
4	68413	12	1	1	178	9	9	0	9	0
…	…	…	…	…	…	…	…	…	…	…

99995	124883	11	1	19	122	9	13	25	41	0
99996	85150	9	1	13	244	7	11	25	43	0
99997	18839	3	1	13	19	8	11	38	42	0
99998	114276	15	1	12	245	8	17	55	21	0
99999	119349	14	1	15	401	7	14	32	27	0

3.3.3 *Data Normalization*

Berdasarkan hasil pemeriksaan pada *dataset* TDA, *dataset* ini memiliki rentang nilai yang cukup tinggi pada setiap atribut. Oleh karena itu, diperlukan proses *data normalization* pada *dataset* untuk meningkatkan performa prediksi model menggunakan *Min-max Normalization* dengan mengkonversi nilai *dataset* menjadi rentang 0 hingga 1 (Mqadi et al., 2021; Prateek et al., 2021; X. Zhu et al., 2021). Tabel 3.6 dan 3.7 menunjukkan perbedaan hasil proses dari data sebelum dan sesudah dilakukan proses normalisasi.

Tabel 3.6 Data Sebelum Proses Normalisasi

No	*Ip*	*App*	*Device*	*Os*	*Channel*	*Day*	*Hour*	*Minute*	*Second*	*Is_attributed*
0	87540	12	1	13	497	7	9	30	38	0
1	105560	25	1	17	259	7	13	40	27	0
2	101424	12	1	19	212	7	18	5	24	0
3	94584	13	1	13	477	7	4	58	8	0
4	68413	12	1	1	178	9	9	0	9	0
…	…	…	…	…	…	…	…	…	…	…
99995	124883	11	1	19	122	9	13	25	41	0
99996	85150	9	1	13	244	7	11	25	43	0

99997	18839	3	1	13	19	8	11	38	42	0
99998	114276	15	1	12	245	8	17	55	21	0
99999	119349	14	1	15	401	7	14	32	27	0

Tabel 3.7 Data Sesudah Proses Normalisasi

No	*Ip*	*App*	*Device*	*Os*	*Channel*	*Day*	*Hour*	*Minute*	*Second*	*Is_attributed*
0	0.239977	0.020000	0.000259	0.015012	0.997980	0.333333	0.391304	0.508475	0.644068	0
1	0.289381	0.043636	0.000259	0.019630	0.517172	0.333333	0.565217	0.677966	0.457627	0
2	0.278041	0.020000	0.000259	0.021940	0.422222	0.333333	0.782609	0.084746	0.406780	0
3	0.259289	0.021818	0.000259	0.015012	0.957576	0.333333	0.173913	0.983051	0.135593	0
4	0.187538	0.020000	0.000259	0.001155	0.353535	1.000000	0.391304	0.000000	0.152542	0
…	…	…	…	…	…	…	…	…	…	…
99995	0.342357	0.018182	0.000259	0.021940	0.240404	1.000000	0.565217	0.423729	0.694915	0
99996	0.233424	0.014545	0.000259	0.15012	0.486869	0.333333	0.478261	0.423729	0.728814	0
99997	0.051625	0.003636	0.000259	0.015012	0.032323	0.666667	0.478261	0.644068	0.711864	0
99998	0.313277	0.025455	0.000259	0.013857	0.488889	0.666667	0.739130	0.932203	0.355932	0
99999	0.327185	0.023636	0.000259	0.017321	0.804040	0.333333	0.608696	0.542373	0.457627	0

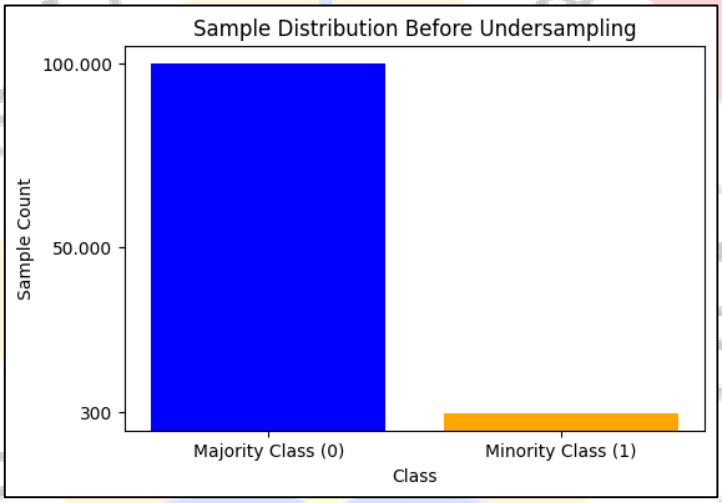
3.3.4 *Data Balancing*

Dataset TDA memiliki perbandingan jumlah sampel klik *bot* dan manusia yang cukup tinggi yaitu 99.773% (Klik = *Bot*) dan 0.227% (Klik = Manusia). Jumlah tersebut diperoleh dari perhitungan menggunakan persamaan 18 dan 19.

$$\textbf{Persentase Klik Bot} = \frac{\textbf{Jumlah Class (Bot)}}{\textbf{Jumlah Data}} \quad (18)$$

$$\textbf{Persentase Klik Manusia} = \frac{\textbf{Jumlah Class (Manusia)}}{\textbf{Jumlah Data}} \quad (19)$$

Oleh karena itu, dibutuhkan proses *data balancing* untuk menangani ketidakseimbangan data menggunakan strategi *undersampling* dari library *Python* yaitu *Imbalanced-learn*. Pada tahap ini diusulkan 3 metode *undersampling* yaitu QDPSKNN, RUS, dan NMU-3.



Gambar 3.2 Data Sebelum Proses *Undersampling*

Gambar 3.2 menunjukkan jumlah data sebelum melalui proses *undersampling* (99.773 sampel *bot*, dan 227 sampel manusia).

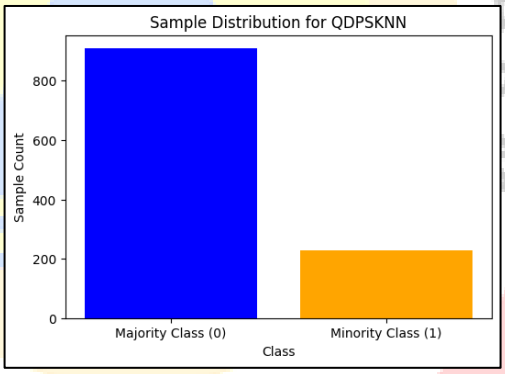
a. *Quad Division Prototype Selection Based K-Nearest Neighbor* (QDPSKNN)

Proses data balancing pada QDPSKNN diawali dengan pembagian *dataset* menjadi 2 kelompok yaitu data mayoritas (*bot*) dan data minoritas (manusia). Selanjutnya, data mayoritas dibagi menjadi 4 kuartil, lalu algoritma *K-Nearest Neighbor* (KNN) digunakan pada setiap kuartil untuk memilih sejumlah prototipe data mayoritas yang berdekatan dengan data minoritas.

Tabel 3.8 memberikan informasi lengkap tentang pengaturan parameter KNN dalam pemilihan prototipe data mayoritas dalam *Python*, yang ditentukan berdasarkan referensi dari parameter yang telah digunakan pada penelitian sebelumnya (Sisodia & Sisodia, 2022b).

Tabel 3.8 Konfigurasi Parameter KNN

Parameter	**Nilai**
Algorithm	Auto
Leaf_size	30
Metric	Minkowski
Metric_params	None
N_jobs	None
N_neighbors	5
P	2
Weights	Uniform



Gambar 3.3 Data Sesudah Proses QDPSKNN

Gambar 3.3 menunjukkan jumlah data sesudah melalui proses QDPSKNN (908 sampel *bot*, dan 227 sampel manusia).

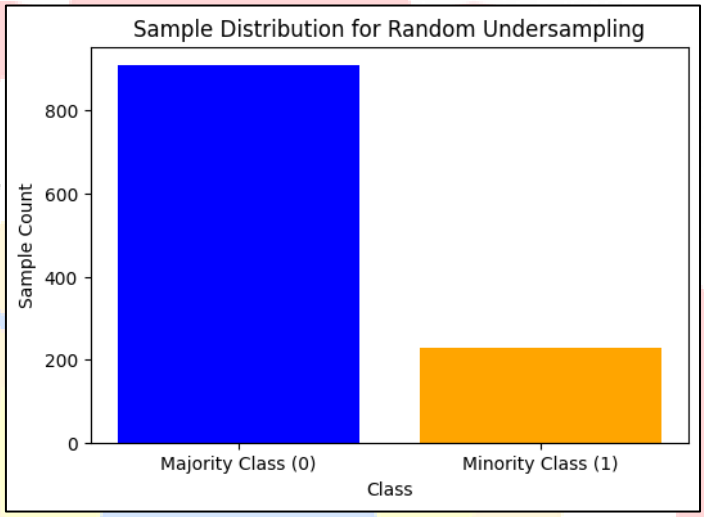
b. Random Undersampling (RUS)

Proses data balancing pada RUS dilakukan dengan cara menghapus data mayoritas secara acak sehingga jumlahnya seimbang dengan data minoritas. Metode RUS ini tersedia dalam library *Python* yaitu *Imbalanced-learn*, selanjutnya Tabel 3.9 memberikan informasi lengkap mengenai konfigurasi

parameter RUS yang akan digunakan untuk menangani ketidakseimbangan data dalam *Python*.

Tabel 3.9 Konfigurasi Parameter RUS

Parameter	Nilai
Random_state	42
Replacement	False
Sampling_strategy	0.25



Gambar 3.4 Data Sesudah Proses RUS

Gambar 3.4 menunjukkan jumlah data sesudah melalui proses RUS (908 sampel *bot*, dan 227 sampel manusia).

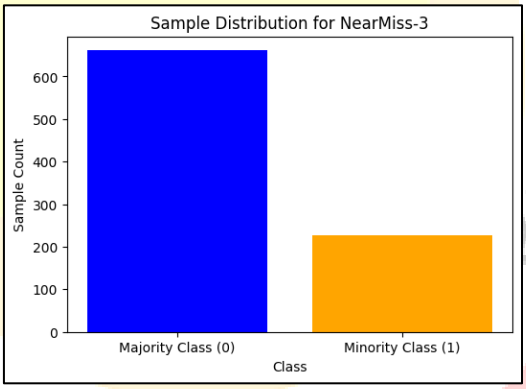
c. Nearmiss Undersampling Version 3 (NMU-3)

NMU merupakan salah satu teknik dari strategi *undersampling* yang mengurangi sampel data mayoritas berdasarkan jarak ke sampel minoritas menggunakan KNN. Peneliti menggunakan NMU versi 3 untuk menghindari masalah kehilangan informasi penting, NMU-3 ini memilih sampel dari kelas mayoritas untuk setiap sampel dalam kelas minoritas yang paling dekat, dan hanya menyimpan sampel dari kelas minoritas yang berada pada batas

keputusan menjadikannya lebih tahan terhadap *noise*. Dibandingkan dengan versi 1 dan 2 yang berfokus pada sampel mayoritas dengan nilai rata-rata jarak terdekat, dan terjauh ke semua sampel minoritas (Mqadi et al., 2021; Prateek et al., 2021). Tabel 3.10 menunjukkan konfigurasi parameter yang digunakan pada NMU-3 dalam menangani ketidakseimbangan data dalam *Python*.

Tabel 3.10 Konfigurasi Parameter NMU-3

Parameter	**Nilai**
N_jobs	None
N_neighbors	3
N_neighbors_ver3	3
Sampling_strategy	0.25
Version	3



Gambar 3.5 Data Sesudah Proses NMU-3

Gambar 3.5 menunjukkan jumlah data sesudah melalui proses NMU-3 (661 sampel *bot*, dan 227 sampel manusia).

Tabel 3.11 menunjukkan bahwa hasil dari proses *data balancing* menggunakan 3 metode yang diusulkan berhasil secara signifikan mengurangi jumlah data pada kelas mayoritas. Dengan demikian *dataset* menjadi lebih seimbang, dan dapat mendukung pelatihan model menjadi lebih akurat (Sisodia & Sisodia, 2023b).

Tabel 3.11 Ringkasan Hasil Proses *Data Balancing*

	Before Undersampling	QDPSKNN	RUS	NMU-3
Majority Class (0)	99,773	908	908	661
Minority Class (1)	227	227	227	227
Total	**100,000**	**1,135**	**1,135**	**888**

3.4 *Data Splitting*

Pada tahap ini, *dataset* dibagi menjadi 2 bagian yaitu *data training*, dan *data testing*. Pembagian data dilakukan dengan rasio 70:30 yang merupakan salah satu pendekatan yang sering digunakan dalam praktik (Joseph, 2022). *Data training* digunakan untuk melatih model agar dapat mempelajari pola-pola dalam data sehingga dapat memprediksi *click fraud* secara akurat, sedangkan *data testing* digunakan untuk mengevaluasi performa model setelah model dilatih. Distribusi data dari setiap metode *undersampling* dapat dilihat pada Tabel 3.12.

Tabel 3.12 Distribusi Pembagian Data

	Data Training (70%)	*Data Testing* (30%)	Total
QDPSKNN	794	341	**1,135**
RUS	794	341	**1,135**
NMU-3	621	267	**888**

3.5 *Hyperparameter Tuning*

Untuk mendapatkan performa optimal dari pelatihan model dilakukan proses pemilihan kombinasi parameter terbaik melalui *hyperparameter tuning*. Gambar 3.6 menunjukkan alur proses *hyperparameter tuning* dengan menggunakan fungsi *RandomizedSearchCV* yang tersedia di pustaka *Python* yaitu *Scikit-learn*, dimana parameter terbaik yang diperoleh akan digunakan dalam proses pelatihan masing-masing model, Tabel 3.13 menunjukkan konfigurasi parameter *RandomizedSearchCV* yang akan digunakan. Selanjutnya, Tabel 3.14, Tabel 3.15, Tabel 3.16 menampilkan informasi terkait *param_grid* (ruang pencarian parameter) untuk setiap model yang akan dicari kombinasi terbaik.

Tabel 3.13 Konfigurasi Parameter *RandomizedSearchCV*

Hyperparameter	**Nilai**
Estimator	[XGBClassifier(), RandomForestClassifier(), SVC()]
Param_distribution	Param_grid
N_iter	100
Scoring	F1
Cv	StratifiedKFold(n_splits=5)
Verbose	1
N_jobs	-1
Random_state	42

Tabel 3.14 *Parameter Grid* untuk XGBoost

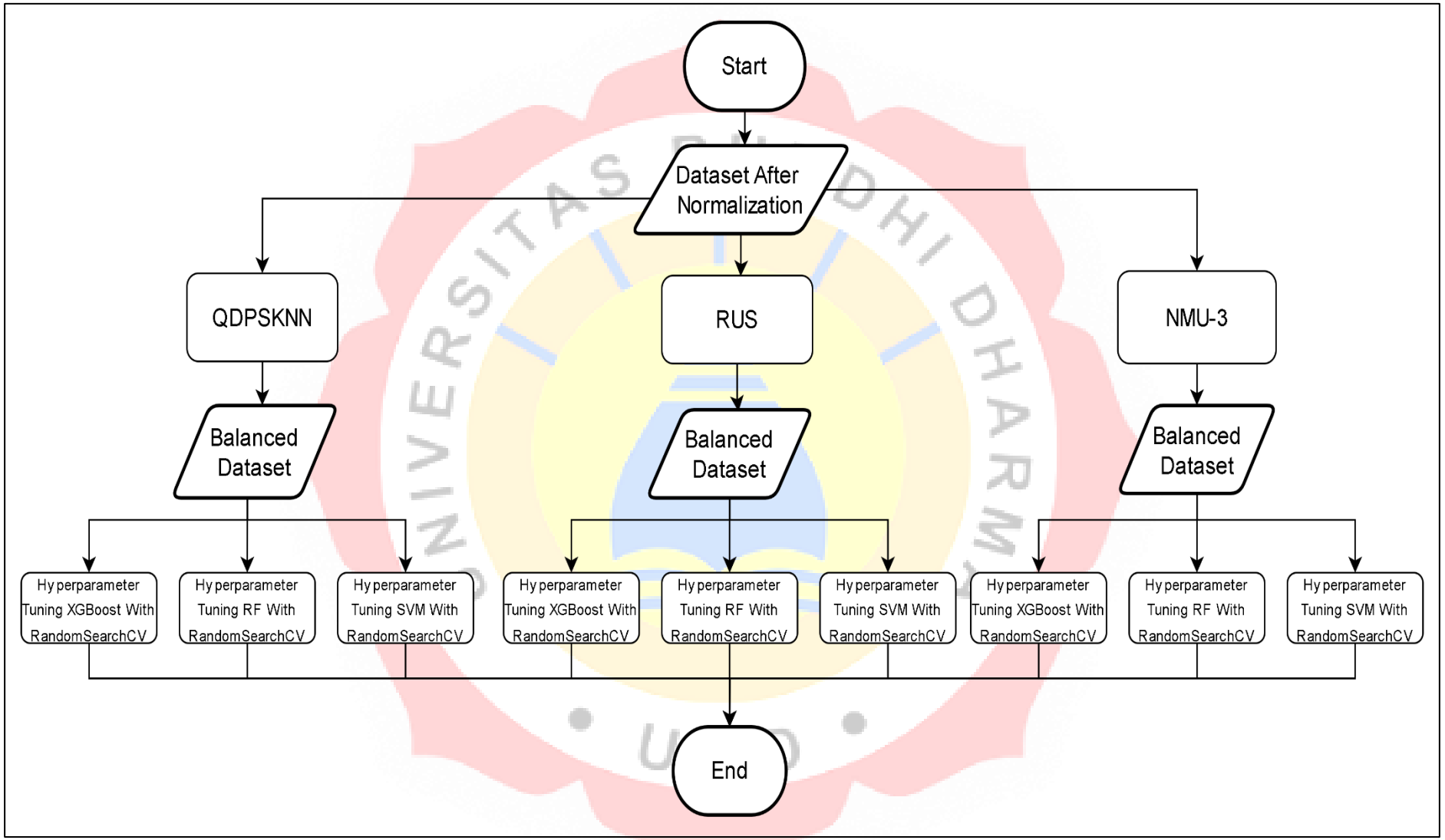
Hyperparameter	**Nilai**
N_estimators	100, 200, 300, 500
Max_depth	3, 5, 7, 10
Learning_rate	0.05, 0.1, 0.2
Subsample	0.6, 0.8, 1.0
Colsample_bytree	0.6, 0.8, 1.0
Min_child_weight	1, 5, 10
Gamma	0, 0.1, 0.3, 0.5
Reg_alpha	0, 0.1, 1, 10
Reg_lambda	1, 5, 10, 20
Scale_post_weight	Jumlah_mayoritas_data_training / Jumlah_minoritas_data_training

Tabel 3.15 *Parameter Grid* Untuk *Random Forest*

Hyperparameter	**Nilai**
N_estimators	100, 200, 300, 500
Max_depth	None, 10, 20, 30, 40
Min_samples_split	2, 5, 10
Min_samples_leaf	1, 2, 4
Max_features	‘auto’, ‘sqrt’, ‘log2’
Bootstrap	True, False

Tabel 3.16 *Parameter Grid* Untuk *Support Vector Machine*

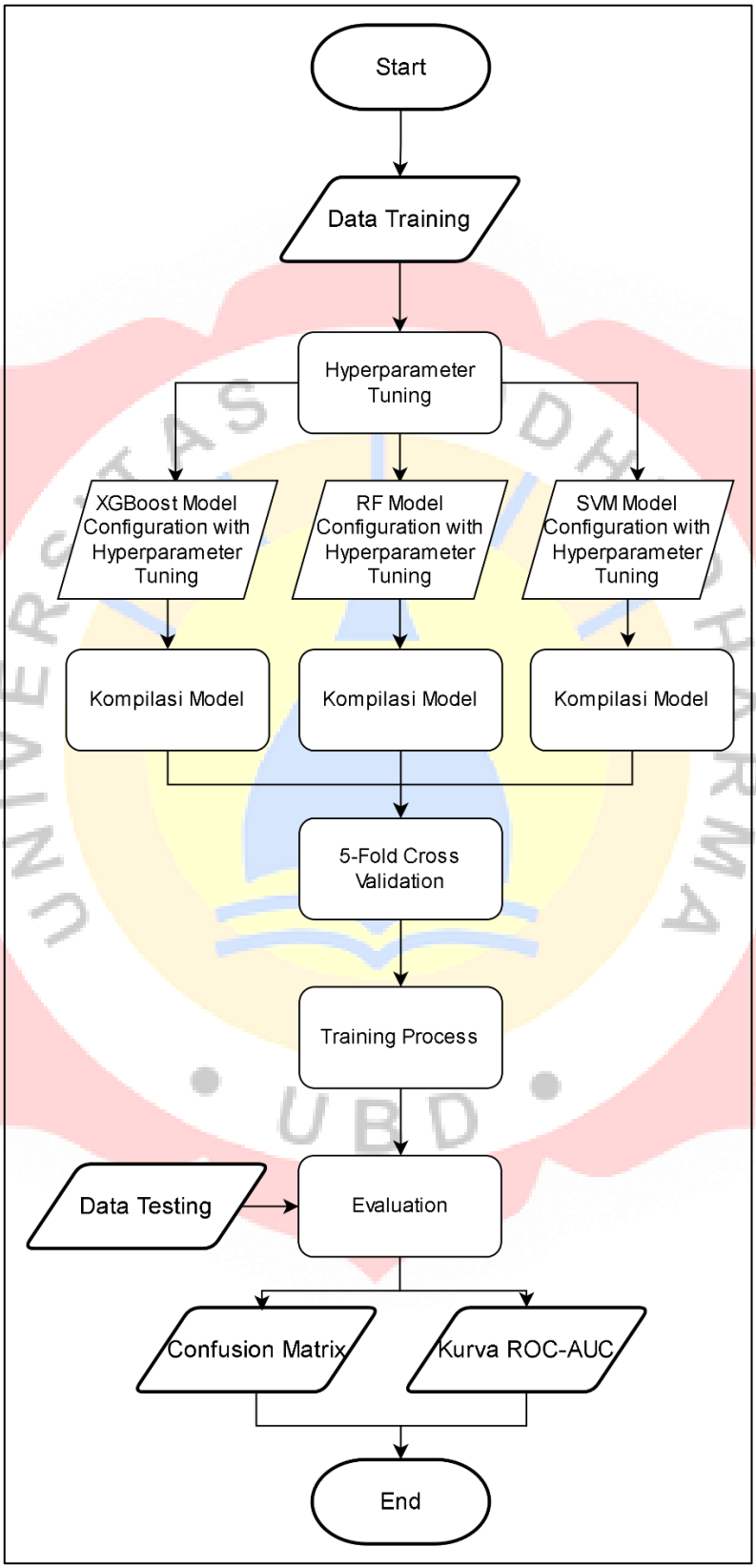
Hyperparameter	**Nilai**
C	0.1, 1, 10, 100
Gamma	1, 0.1, 0.01, 0.001
Kernel	'linear', 'rbf', 'poly', 'sigmoid'



Gambar 3.6 ***Flowchart*** **Proses** ***Hyperparameter Tuning***

3.6 Model *Training*

3.6.1 *Flowchart* Pelatihan Model



Gambar 3.7 *Flowchart Model Training*

Dalam proses pelatihan model, seperti yang ditunjukkan pada Gambar 3.7 diatas, terdapat 3 model yang digunakan yaitu XGBoost, RF, dan SVM. Setiap model menggunakan parameter yang telah ditentukan melalui proses *hyperparameter tuning* menggunakan *RandomizedSearchCV*.

Selanjutnya model dilatih menggunakan metode *Stratified K-fold Cross Validation* yang umum digunakan dalam *machine learning* untuk memastikan generalisasi model dan mengurangi risiko *overfitting* (Buitinck et al., 2024). Dalam metode ini, *data training* dibagi menjadi K *subset* (biasanya disebut "*fold*"), dimana 1 *fold* digunakan sebagai data validasi sementara sisanya digunakan untuk pelatihan. Proses ini diulangi sebanyak K kali sehingga seluruh data menjadi bagian dari pelatihan dan validasi. Dalam penelitian ini, nilai K yang digunakan adalah 5 karena dianggap sebagai pilihan *default* berdasarkan kemampuan untuk menghasilkan estimasi error yang akurat tanpa menyebabkan bias atau varians yang berlebihan (James et al., 2023).

Terakhir, model diuji menggunakan *data testing* untuk menentukan hasil akhir, dan untuk mendapatkan pemahaman yang lebih rinci evaluasi dilakukan menggunakan *Confusion Matrix* dan kurva ROC-AUC.

3.6.2 Perangkat Keras

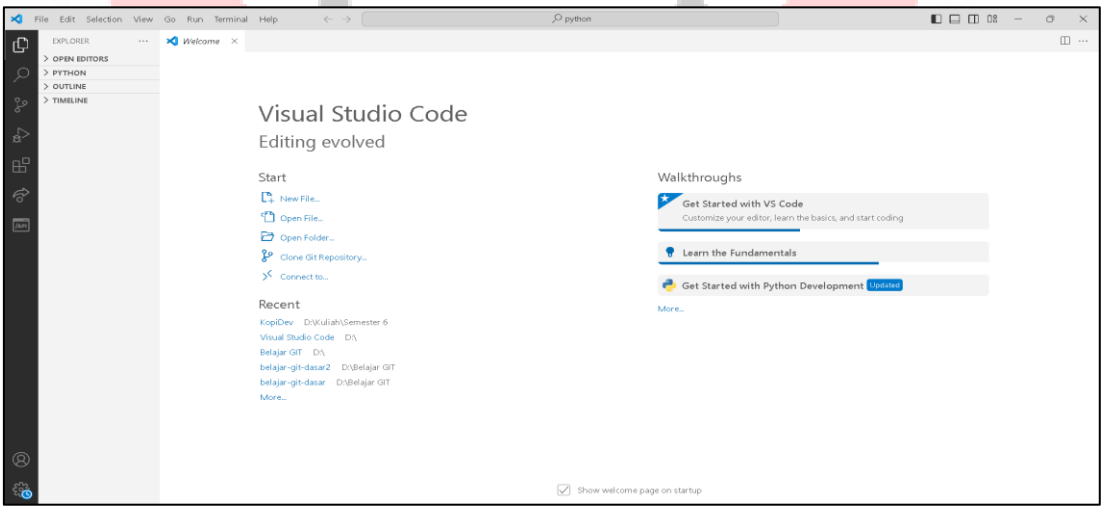
Perangkat keras atau Hardware itu sangat penting untuk melatih model *machine learning*. Spesifikasi perangkat keras yang digunakan sangat mempengaruhi kecepatan proses pelatihan model. Oleh karena itu, kecepatan proses pelatihan model berkorelasi positif dengan spesifikasi perangkat keras yang digunakan. Spesifikasi perangkat keras yang digunakan untuk melatih model dalam penelitian ini dapat dilihat pada Tabel 3.17.

Tabel 3.17 Spesifikasi Perangkat Keras

Jenis Perangkat Keras	Nama Perangkat Keras
Processor	Intel Core i7-8750H
RAM	16GB DDR4 26667MHz
Kartu Grafis	NVDIA GeForce GTX 1060
Penyimpanan	SSD 512GB
Peripheral	Keyboard Mouse

3.6.3 Lingkungan Pengembangan

Proses pelatihan model dilakukan menggunakan Visual Studio Code yang dapat dilihat pada Gambar 3.8. *Python* Version 3.12.0 digunakan untuk menjamin keamanan kinerja, fitur terbaru, dan mempertahankan kompabilitas penuh dengan *library* yang digunakan. Pustaka *Scikit-learn* Version 1.5.2 digunakan untuk menyediakan berbagai alat bantu dalam pengembangan model *machine learning*. Pustaka *Imbalanced-learn* Version 0.12.4 digunakan untuk menangani ketidakseimbangan kelas dalam *dataset* dalam proses klasifikasi, dengan menyediakan metode *resampling* seperti RUS, dan NMU.



Gambar 3.8 *Visual Studio Code*

3.7 Metode Evaluasi

Mengukur kinerja model merupakan langkah yang perlu dilakukan dalam melatih sebuah model *machine learning*. Penelitian ini akan menggunakan 2 metode evaluasi yang berbeda yaitu, *Confusion Matrix* dan kurva ROC-AUC.

3.7.1 *Confusion Matrix*

Confusion Matrix adalah tabel yang digunakan untuk menilai kinerja model klasifikasi secara menyeluruh, sehingga dapat menjelaskan sejauh mana model yang telah dilatih dapat memprediksi *click fraud* dengan tepat. *Confusion Matrix* mencakup 4 nilai yaitu Nilai *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN) yang digunakan untuk menghitung *accuracy*, *precision*, *recall*, dan *f1-score*. Nilai-nilai tersebut dihitung melalui persamaan (14)-(17). Hasil *Confusion Matrix* akan divisualisasikan menggunakan pustaka *Python* yaitu *Matplotlib*.

3.7.2 Kurva ROC-AUC

ROC-AUC adalah metrik evaluasi yang digunakan untuk mengukur sejauh mana model klasifikasi dapat membedakan kelas positif dan negatif dalam suatu *dataset*. Langkah pertama yaitu model dilatih pada *data training*, dan kemudian dilakukan prediksi pada *data testing*. Fungsi *roc_curve* dari *library Scikit-learn* digunakan untuk menghasilkan kurva ROC-AUC yang membandingkan nilai *False Positive Rate* (FPR) dan *True Positive Rate* (TPR). Parameter yang digunakan adalah *y_true*, yang merupakan target kelas *data testing*, dan *y_score* yang merupakan hasil prediksi probabilitas positif pada *data testing*. Tabel 2.3 menunjukkan semakin tinggi nilai AUC, semakin baik kemampuan model dalam melakukan klasifikasi.