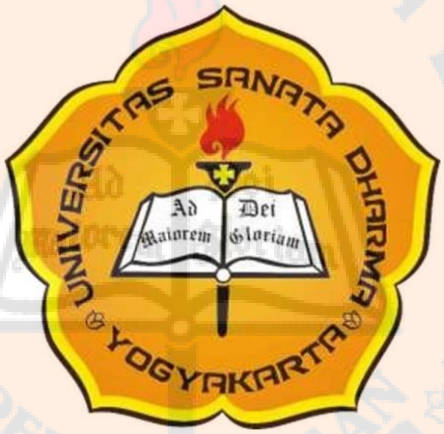
PERBANDINGAN PENGGUNAAN NORMALISASI *MIN-MAX* DAN *Z-SCORE* PADA *CATBOOST* DALAM KLASIFIKASI KEGAGALAN PEMBAYARAN DI SEKTOR PERBANKAN

SKRIPSI

Diajukan untuk memenuhi salah satu syarat
memperoleh gelar Sarjana Komputer
Program Studi Informatika



Disusun oleh :
RIAN CHRISTIAN SIDIN
205314001

PROGRAM STUDI INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS SANATA DHARMA
YOGYAKARTA
2024

COMPARISON OF THE USE OF MIN-MAX NORMALIZATION AND Z-SCORE ON CATBOOST IN THE CLASSIFICATION OF PAYMENT FAILURES IN THE BANKING SECTOR

THESIS

Present as Partial Fulfillment of The Requirement
To Obtain Sarjana Komputer Degree
in Informatics Study Program

By :

RIAN CHRISTIAN SIDIN

205314001

FACULTY OF SCIENCE AND TECHNOLOGY

SANATA DHARMA UNIVERSITY

YOGYAKARTA

2024

SKRIPSI

PERBANDINGAN PENGGUNAAN NORMALISASI *MIN-MAX* DAN *Z-SCORE* PADA *CATBOOST* DALAM KLASIFIKASI KEGAGALAN PEMBAYARAN DI SEKTOR PERBANKAN

Disusun oleh:

Rian Christian Sidin

Nim: 205314001

Dosen Pembimbing

Ir. Paulina Heruningsih Prima Rosa, S.Si., M.Sc.

Yogyakarta, 29 Juli 2024

LEMBAR PENGESAHAN SKRIPSI

PERBANDINGAN PENGGUNAAN NORMALISASI *MIN-MAX* DAN *Z-SCORE* PADA *CATBOOST* DALAM KLASIFIKASI KEGAGALAN PEMBAYARAN DI SEKTOR PERBANKAN

Dipersiapkan dan disusun oleh :

Rian Christian Sidin

NIM : 205314001

SUSUNAN DEWAN PENGUJI

JABATAN		NAMA LENGKAP	TANDA TANGAN
Ketua	:	Eko Hari Parmadi, S.Si., M.Kom.	
Sekretaris	:	Ir. Kartono Pinaryanto, S.T., M.Cs.	
Pembimbing	:	Ir. Paulina Heruningsih Prima Rosa, S.Si., M.Sc.	

Yogyakarta, 24 Juli 2024

Fakultas Sains dan Teknologi

Universitas Sanata Dharma

Dekan,

Ir. Drs. Haris Sriwindono, M.Kom., Ph.D.

PERNYATAAN KEASLIAN KARYA

Saya menyatakan dengan sesungguhnya bahwa skripsi yang saya tulis ini tidak memuat karya atau bagian karya orang lain, kecuali yang telah disebutkan dalam kutipan dan daftar pustaka dengan mengikuti ketentuan sebagaimana layaknya karya ilmiah.

Apabila dikemudian hari ditemukan indikasi plagiarisme dalam naskah ini, saya bersedia menanggung segala sangsi sesuai peraturan perundang-undangan yang berlaku

Yogyakarta, 12 Juli 2024

Penulis,

Rian Christian Sidin

LEMBAR PERNYATAAN PERSEJUTUAN PUBLIKASI KARYA ILMIAH UNTUK KEPERLUAN AKADEMIS

Yang bertanda tangan di bawah ini, saya mahasiswa Universitas Sanata Dharma:

Nama : Rian Christian Sidin

Nim : 205314001

Demi pengembangan ilmu pengetahuan saya memberikan karya ilmiah ini kepada Perpustakaan Universitas Sanata Dharma karya ilmiah saya yang berjudul:

PERBANDINGAN PENGGUNAAN NORMALISASI *MIN-MAX* DAN *Z-SCORE* PADA *CATBOOST* DALAM KLASIFIKASI KEGAGALAN PEMBAYARAN DI SEKTOR PERBANKAN

Beserta perangkat yang diperlukan (bila ada). Dengan demikian saya memberikan kepada Perpustakaan Universitas Sana Dharma hak untuk menyimpan, mengalihkan dalam bentuk media lain, untuk kepentingan akademis tanpa perlu meminta ijin dari saya maupun memberikan royalti kepada saya selama tetap mencantumkan nama saya sebagai penulis.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di Yogyakarta

Pada tanggal: 12 Juli 2024

Yang menyatakan,

Rian Christian Sidin

MOTTO

Roma 8:18

"Sebab aku yakin, bahwa penderitaan yang kita alami sekarang tidak ada bandingannya dengan kemuliaan yang kelak akan dinyatakan kepada kita".

KATA PENGANTAR

Puji syukur kehadirat Tuhan Yang Maha Esa atas segala rahmat dan karunia-Nya, sehingga penulis dapat menyelesaikan tugas akhir yang berjudul "**PERBANDINGAN PENGGUNAAN NORMALISASI *MIN-MAX* DAN *Z-SCORE* PADA *CATBOOST* DALAM KLASIFIKASI KEGAGALAN PEMBAYARAN DI SEKTOR PERBANKAN".** Penelitian ini disusun untuk memenuhi salah satu syarat memperoleh gelar Sarjana di Program Studi Informatika, Universitas Sanata Dharma.

Dalam proses penyusunan penelitian ini, penulis telah banyak mendapatkan bantuan, bimbingan, dan dukungan dari berbagai pihak. Oleh karena itu, penulis ingin menyampaikan ucapan terima kasih yang sebesar-besarnya kepada:

1. Tuhan Yesus Kristus yang selalu menyertai, memberkati, menopang dan selalu memberikan kekuatan bagi penulis sehingga penulis dapat menyelesaikan tugas akhir ini dengan baik.
2. Ibu Ir.Paulina Heruningsih Prima Rosa S.Si, M.Sc, selaku dosen pembimbing skripsi yang selalu membagi waktu untuk penulis melakukan bimbingan dan selalu memberikan arahan yang baik, sehingga penulis dapat menyelesaikan tugas akhir ini.
3. Bapak Ir. Drs. Haris Sriwindono, M.Kom., Ph.D. selaku Dekan Fakultas Sains dan Teknologi universitas Sanata Dharma.
4. Bapak Dr. Ir. Iwan Binanto, selaku Ketua Program Studi S1 Informatika Universitas Sanata Dharma.
5. Seluruh dosen Program Studi Informatika Fakultas Sains dan Teknologi Universitas Sanata Dharma yang telah memberikan ilmu pengetahuan dan pengalaman yang sangat berharga kepada penulis.
6. Bapak Imanuel D. Sidin, Ibu Meianna Sihombing dan Kakak Rina Septriani Sidin yang selalu memberikan semangat, doa dan juga motivasi untuk penulis agar penulis dapat menyelesaikan penelitian ini dengan lancar.

7. Seluruh teman-teman Informatika angkatan 2020 yang selalu menghibur dan memberi semangat kepada penulis sehingga penulis dapat menyelesaikan tugas akhir ini.
8. Keluarga besar ADIKA yang selalu memberikan dukungan serta doa kepada penulis agar dapat menyelesaikan tugas akhir ini dengan lancar.
9. Semua pihak yang tidak dapat disebutkan satu per satu, yang telah memberikan dukungan, baik secara langsung maupun tidak langsung, kepada penulis dalam menyelesaikan tugas akhir ini.

Penulis menyadari bahwa dalam penyusunan tugas akhir ini masih terdapat kekurangan dan keterbatasan. Oleh karena itu, kritik dan saran yang membangun sangat diharapkan untuk perbaikan di masa mendatang. Akhir kata, penulis berharap semoga tugas akhir ini dapat memberikan manfaat bagi perkembangan ilmu pengetahuan dan dapat menjadi referensi bagi peneliti lainnya.

Yogyakarta, 12 Juli 2024

Penulis,

Rian Christian Sidin

ABSTRAK

Kegagalan pembayaran di sektor perbankan adalah masalah kritis yang mempengaruhi stabilitas keuangan dan operasional bank. Mendeteksi kegagalan pembayaran secara akurat sangat penting untuk mengurangi risiko dan kerugian. Penelitian ini mengeksplorasi penggunaan algoritma *CATBOOST* dalam klasifikasi kegagalan pembayaran, dengan fokus pada efektivitas dua metode normalisasi, yaitu *Min-Max* dan *Z-Score*. Pengujian dilakukan menggunakan metode *k-fold cross-validation* dengan *k-fold* 3, 5, 7, dan 10 untuk menilai akurasi dan efisiensi model.

Hasil penelitian menunjukkan bahwa algoritma *CATBOOST* memberikan akurasi stabil sekitar 93% untuk kedua metode normalisasi. Akurasi tertinggi dicapai dengan metode normalisasi *Min-Max* pada *k-fold* 7 sebesar 93,32%. Selain itu, waktu pelatihan *CATBOOST* cenderung lebih cepat dengan normalisasi *Z-Score* dibandingkan *Min-Max*. Evaluasi indikator lain seperti presisi, *recall*, dan *F1 Score* juga menunjukkan hasil yang baik, dengan presisi di atas 99%, *recall* sekitar 87%, dan *F1 Score* sekitar 92%. Dengan demikian, dapat disimpulkan bahwa *CATBOOST* efektif dalam klasifikasi kegagalan pembayaran di sektor perbankan setelah normalisasi data. Kedua metode normalisasi memberikan hasil yang serupa, namun normalisasi *Z-Score* lebih efisien dalam hal waktu pelatihan. Pemilihan metode normalisasi dapat disesuaikan dengan preferensi tanpa mengorbankan akurasi dan performa model.

Kata kunci: CATBOOST, Normalisasi Min-Max, Normalisasi Z-Score, Klasifikasi Kegagalan Pembayaran, Perbankan.

ABSTRACT

Payment failures in the banking sector are a critical issue that affects the financial stability and operational efficiency of banks. Accurately detecting payment failures is essential to mitigate risks and losses. This study explores the use of the *CATBOOST* algorithm in classifying payment failures, focusing on the effectiveness of two normalization methods: *Min-Max* and *Z-Score*. The evaluation was conducted using k-fold cross-validation with k-fold values of 3, 5, 7, and 10 to assess the model's accuracy and efficiency.

The results show that the *CATBOOST* algorithm provides a stable accuracy of around 93% for both normalization methods. The highest accuracy was achieved with the *Min-Max* normalization method at k-fold 7, reaching 93.32%. Additionally, the training time for *CATBOOST* tends to be faster with *Z-Score* normalization compared to *Min-Max*. Other performance indicators such as precision, recall, and F1 Score also showed favorable results, with precision above 99%, recall around 87%, and F1 Score around 92%. Therefore, it can be concluded that *CATBOOST* is effective in classifying payment failures in the banking sector after data normalization. Both normalization methods yield similar results, but *Z-Score* normalization is more efficient in terms of training time. The choice of normalization method can be adjusted based on user preference without compromising model accuracy and performance.

Keywords: *CATBOOST*, *Min-Max* Normalization, *Z-Score* Normalization, Payment Failure Classification, Banking.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I

PENDAHULUAN

1.1 Latar Belakang

Terdapat berbagai alasan dan faktor yang menyebabkan kebutuhan akan kredit. Orang memiliki kepentingan dan motivasi yang berbeda untuk mencoba berbagai metode guna memperoleh kredit. Di sisi lain, pemerintah telah menghidupkan kembali organisasi dan lembaga yang bertanggung jawab dalam mendistribusikan kredit, memberikan kesempatan kepada masyarakat untuk mendapatkan bantuan dari sumber-sumber kredit yang sah. Selain itu, terdapat juga jalur atau sumber-sumber di luar lingkup pemerintah yang memberikan kesempatan kepada individu atau kelompok yang memiliki sumber daya keuangan yang cukup untuk memberikan kredit (Bambang Catur P, 2014).

Dunia perbankan Indonesia sedang menghadapi persaingan yang ketat saat menghadapi Masyarakat Ekonomi Asean (MEA). MEA adalah sebuah program kerja sama peningkatan ekonomi regional (keuangan, perdagangan/bisnis dan perbankan) antar negara-negara Asean. Persaingan antara bank-bank besar Asean membuat perbankan di Indonesia menghadapi ancaman yang serius mengingat aset dan modal yang dimiliki perbankan di Indonesia kurang kuat untuk bersaing dengan bank-bank besar Asean. Persaingan yang tinggi membuat bank semakin aktif dan produktif menyalurkan kredit dan produk-produk perbankan lainnya demi menguasai sektor makro dan mikro. Bank lebih fokus untuk mengejar keuntungan dan kurang memperhitungkan risiko yang mungkin akan timbul dari persaingan tersebut (Purnamandari & Badera, 2015).

Pengaruh perubahan dalam ekonomi dan kebijakan bank bisa membuat kredit jadi lebih berisiko. Ini memaksa bank untuk lebih berhati-hati dalam mengawasi risiko agar bisnis mereka tetap berjalan lancar. Risiko gagal bayar pinjaman meningkat dalam situasi ekonomi buruk dengan pendapatan menurun atau tingkat pengangguran tinggi. Kebijakan perbankan yang ketat atau perubahan suku bunga juga mempengaruhi kemampuan membayar pinjaman tepat waktu. Kredit bermasalah merujuk pada situasi di mana pengembalian kredit memiliki risiko kegagalan, bahkan dapat menunjukkan potensi kerugian bagi bank (Subroto & Arianto, 2011).

Dalam penelitian ini, masalah yang ingin diatasi adalah prediksi kegagalan pembayaran pinjaman, yang merupakan isu krusial bagi lembaga keuangan karena dapat menyebabkan kerugian signifikan, untuk itu penelitian ini menggunakan model *Categorical Boosting (CATBOOST)* dengan normalisasi data yaitu: *Min-Max normalization* dan *Z-Score normalization.* *Min-Max* dipilih karena dapat mengubah skala fitur ke rentang [0,1], yang membantu model dalam mengkonsolidasi berbagai fitur dengan skala berbeda ke dalam skala yang sama. Sementara itu, *Z-Score* dipilih karena dapat mengubah distribusi fitur ke distribusi standar dengan mean 0 dan standar deviasi 1, yang sangat berguna jika data memiliki distribusi normal. Pada tahap klasifikasi dibandingkan hasil akurasi yang diperoleh *CATBOOST* dari *Min-Max normalization*, *Z-Score normalization*, untuk mengetahui metode normalisasi data mana yang lebih optimal dan akurat dalam meningkatkan performansi klasifikasi gagal membayar pinjaman. Dengan pemahaman yang lebih baik tentang faktor-faktor ini, lembaga keuangan dapat

mengambil keputusan yang lebih baik dalam memberikan pinjaman dan mengurangi risiko gagal bayar pinjaman (Andini, 2017).

Penelitian terdahulu berpendapat bahwa penggunaan model *CATBOOST* berhasil mencapai skor tertinggi dalam hal akurasi, menunjukkan performa yang sangat baik, hasil dari pengaturan parameter dan penilaian model juga mengindikasikan bahwa model *CATBOOST* mendapatkan skor yang lebih tinggi dibandingkan dengan model *XGBoost* (Purbolingga dkk., 2023), hal ini juga diperkuat oleh (Nasution dkk., 2023) yang mengungkapkan bahwa model *CATBOOST* mempunyai kemampuan yang tangguh dalam mendeteksi risiko kredit dan berkontribusi dalam meminimalkan risiko kredit dalam P2P lending.

1.2 Rumusan Masalah

1. Berapa tingkat akurasi yang dihasilkan oleh algoritma *CATBOOST* dalam memprediksi kegagalan pembayaran pinjaman?
2. Metode normalisasi data mana yang lebih optimal antara *Min-Max normalization* dan *Z-Score normalization* dalam meningkatkan performa klasifikasi oleh *CATBOOST*?

1.3 Tujuan Penelitian

1) Menguji tingkat akurasi model *CATBOOST* setelah melalui proses normalisasi menggunakan *Min-Max* dan *Z-Score* untuk menentukan metode yang paling optimal.
2) Membandingkan efektivitas metode normalisasi *Min-Max* dan *Z-Score* dalam meningkatkan performa klasifikasi oleh algoritma *CATBOOST* untuk prediksi kegagalan pembayaran pinjaman bank.

1.4 Manfaat Penelitian

Manfaat penelitian ini adalah sebagai berikut:

1) Meningkatkan ketepatan dalam memprediksi gagal bayar pinjaman, membantu lembaga keuangan mengidentifikasi calon peminjam dengan memprediksi gagal pembayaran.
2) Meningkatkan efisiensi lembaga keuangan dengan menggunakan model prediksi yang lebih akurat.
3) Berkontribusi pada stabilitas sistem keuangan secara keseluruhan dengan mengurangi memprediksi gagal bayar kredit.

1.5 Batasan Masalah

Dalam penelitian ini, batasan yang ditemui adalah penggunaan data public yang diambil dari situs *Kaggle : https://www.kaggle.com/datasets/nikhil1e9/loan-default/data*.

1.6 Sistematika Penulisan

Berikut adalah gambaran singkat tentang apa yang akan diungkapkan dalam setiap bab penelitian ini:

Bab I Pendahuluan

Bagian pendahuluan mengulas topik penelitian mengenai pengaruh normalisasi data terhadap akurasi prediksi gagal bayar pinjaman dengan membandingkan metode *Min-Max dan Z-Score* menggunakan model klasifikasi *Categorical Boosting*. Latar belakang masalah, urgensi prediksi yang akurat, serta rumusan masalah dan tujuan penelitian akan diuraikan secara mendalam.

Bab II Tinjauan Pustaka

Pada bab ini, akan dibahas tinjauan pustaka yang komprehensif terkait pengaruh normalisasi data, khususnya dengan metode *Min-Max* dan *Z-Score*, terhadap akurasi model klasifikasi *Categorical Boosting*. Kelebihan dan kelemahan dari pendekatan ini juga akan dibahas.

Bab III Metodologi Penelitian

Metodologi Penelitian akan menjelaskan secara rinci mengenai data, pemrosesan data, serta menjelaskan model klasifikasi *Categorical Boosting* dengan fokus pada implementasi normalisasi data menggunakan *Min-Max* dan *Z-Score*. Metode evaluasi dan pengukuran kinerja model akan dijelaskan.

Bab IV IMPLEMENTASI DAN ANALISIS HASIL

Implementasi serta hasil dari penelitian ini. Setelah itu, akan melakukan pembahasan dan analisa terhadap hasil pengujian-pengujian yang telah dilakukan.

Bab V PENUTUP

Menyimpulkan percobaan-percobaan dalam penelitian yang telah dilakukan dan juga akan diuraikan saran dari penulis untuk pengembangan dari penelitian ini.

DAFTAR PUSTAKA

Berisi tentang referensi yang digunakan dalam penelitian ini.

BAB II

LANDASAN TEORI

2.1 Tinjauan Pustaka

Dalam penelitian yang dilakukan oleh Gde Agung Brahmana Suryanegara, dkk (2021) tentang 'Peningkatan Hasil Klasifikasi pada Algoritma *Random Forest* untuk Deteksi Pasien Penderita Diabetes Menggunakan Metode Normalisasi', permasalahan yang ditemukan adalah terdapat peningkatan jumlah penderita diabetes di Indonesia yang menjadi masalah kesehatan yang serius. Pada penelitian tersebut, algoritma *Random Forest* digunakan untuk membangun model klasifikasi yang dapat mendeteksi penyakit diabetes ke dalam dua kelas, yaitu positive diabetes dan negative diabetes. Hasil dari penelitian tersebut menunjukkan bahwa akurasi tertinggi adalah 95,45% dengan normalisasi *Min-Max* dan 95,00% dengan normalisasi *Z-Score*.

Dalam penelitian yang dilakukan oleh Westari dan Halim (2021) tentang 'Performa *Comparison* of the *K-Means Method for Classification in Diabetes Patients Using Two Normalization Methods'*, tantangan yang dihadapi adalah menemukan cara yang tepat untuk membandingkan kedua metode normalisasi data dan mengevaluasi akurasi algoritma *K-Means* dalam klasifikasi diabetes. Pada penelitian tersebut, *output* yang dihasilkan adalah perbandingan antara akurasi algoritma *K-Means* dalam klasifikasi diabetes menggunakan dua metode normalisasi data, yaitu *Min-Max* dan *Z-Score*. Hasil penelitian ini menunjukkan bahwa rata-rata akurasi tertinggi terdapat pada dataset PID yang menggunakan

metode normalisasi *Min-Max* pada data latih 30 sebesar 79%. Di sisi lain, akurasi terendah ditemukan pada dataset PID yang menggunakan metode normalisasi *Z-Score*, dengan rata-rata akurasi sebesar 67%.

Dalam penelitian yang dilakukan oleh Pandey dan Jain (2017) tentang '*Comparative Analysis* of *KNN Algorithm using Various Normalization Techniques'*, berfokus pada analisis perbandingan teknik normalisasi yang berbeda dalam algoritma *K-Nearest Neighbors* (KNN) untuk klasifikasi data. Hasil yang diperoleh dari studi ini menunjukkan bahwa teknik normalisasi *Min-Max* memberikan hasil yang lebih baik daripada teknik normalisasi *Z-Score* dalam algoritma *KNN* untuk klasifikasi data dan berdasarkan hasil penelitian, akurasi rata-rata yang diperoleh adalah 88,09% untuk normalisasi *Min-Max* dan 78,56% untuk normalisasi *Z-Score*.

Tabel 2. 1 Referensi Pilihan dalam Tinjauan Pustaka

No.	Judul Referensi	Penulis	Tahun	Hasil
1.	Peningkatan Hasil Klasifikasi pada Algoritma *Random Forest* untuk Deteksi Pasien Penderita Diabetes Menggunakan Metode Normalisasi	Gde Agung Brahmana Suryanegara, dkk	2021	Hasil penelitian ini menunjukkan bahwa model yang menggunakan *Min-Max* normalization mencapai akurasi tertinggi sebesar 95.45%. Model yang menggunakan *Z-Score* normalization juga memberikan hasil yang baik dengan akurasi sebesar 95%. Sedangkan model tanpa normalisasi data

				memiliki akurasi sebesar 92%.
2.	Performa Comparison of the *K-Means* Method for Classification in Diabetes Patients Using Two *Normalization* Methods	Westari dan Halim	2021	Hasil akurasi dari perbandingan kedua metode normalisasi, yaitu *Min-Max normalization* dan *Z-Score normalization*, menunjukkan bahwa metode normalisasi *Min-Max* menghasilkan akurasi terbaik sebesar 79% pada dataset Pima Indian Diabetes (PID), dibandingkan dengan metode normalisasi *Z-Score* yang memiliki akurasi rata-rata sebesar 67%.
3.	A Comparative Assessment of Credit Risk Model Based on *Machine Learning'*	Yuelin Wang, dkk	2019	Penelitian ini menunjukkan hasil akurasi prediksi untuk berbagai nilai K menggunakan normalisasi *Min-Max* dan normalisasi *Z-Score*. Dapat dilihat bahwa untuk setiap nilai K, teknik normalisasi yang berbeda memberikan akurasi yang berbeda pula. Misalnya, untuk nilai K=1, normalisasi *Min-Max* memberikan akurasi 100%, sedangkan normalisasi *Z-Score* memberikan akurasi 85.71%.
4.	Implementasi Corelation matrix pada klasifikasi dataset wine	Erfin Nur Rohma Khakim, dkk	2023	Normalisasi *Z-Score* menghasilkan nilai akurasi yang lebih tinggi dibandingkan normalisasi *Min-Max*, namun normalisasi

				Min-Max dapat ditingkatkan akurasinya dengan melakukan fitur seleksi. Normalisasi *Z-Score* menghasilkan nilai akurasi tertinggi pada K=1 yaitu sebesar 73,75%. Sedangkan normalisasi *Min-Max* menghasilkan nilai akurasi tertinggi pada K=1 juga yaitu sebesar 68,12%. Namun, terdapat eksperimen dengan menghilangkan atribut tertentu pada normalisasi *Min-Max* yang mampu menaikkan nilai akurasi dari 73,75% menjadi 75,62%.
5.	Performance Comparison of Grid Search and Random Search Methods for Hyperparameter Tuning in Extreme Gradient Boosting Algorithm to Predict Chronic Kidney Failure	Anggoro dan Mukti	2021	Pada penelitian tersebut algoritma yang digunakan adalah Extreme Gradient Boosting (*XGBoost*). Dalam penelitian ini, dilakukan hyperparameter tuning menggunakan metode *grid search* dan *random search*. Dari hasil penelitian, metode grid search dengan *preprocessing* data *Min-Max normalization* dan *Z-Score normalization* menghasilkan akurasi terbaik sebesar 99,28% dan nilai f-measure sebesar 0,9942. Hasil yang sama juga diperoleh dari metode *Z-Score normalization*

				dan *random oversampling* dengan nilai akurasi 99,28% dan f-measure 0,9942.
6.	A Comparison of Normalization Data Transformation Efficiency Affecting with Bank Customer Credit Data Classification using Data Mining Techniques	Chutima Suksamai, dkk	2022	Hasil akurasi dari penelitian ini menunjukkan bahwa teknik *K-Nearest Neighbor* (KNN) dengan metode normalisasi *Min-Max*, *Z-Score*, dan Mean memiliki tingkat akurasi tertinggi sebesar 80.63%. Sementara itu, teknik Decision Tree (DT) dengan metode normalisasi *Min-Max*, *Z-Score*, dan *Mean* memiliki tingkat akurasi yang sedikit lebih rendah, yaitu sekitar 80.43%. Teknik *Neural Network* (NN) dengan metode normalisasi *Min-Max*, *Z-Score*, dan *Mean* memiliki tingkat akurasi terendah sebesar 77.45%.
7.	Pengaruh Normalisasi Data Terhadap Performa Hasil Klasifikasi Algoritma Backpropagation	Permana dan Salisah	2022	Dengan akurasi 80,00%, dataset Iris tanpa normalisasi sebenarnya menghasilkan performa BP terbaik. Data dinormalisasi menggunakan Normalisasi *Z-Score* dan Normalisasi MinMax pada Algoritma Backpropagation, menghasilkan performa terendah

				dengan akurasi 96,00%. Ketika data tidak dinormalisasi, Iris Dataset memiliki kinerja terbaik, mungkin karena nilai atribut dalam dataset tidak terlalu bervariasi satu sama lain.
8.	Research on Credit Card Overdue Risk Prediction based on CatBoost Model	Zhu	(2024)	Pada penelitian ini Model SVM memiliki akurasi 81,2% dalam memprediksi kasus tunggakan kartu kredit dan akurasi 82,9% dalam memprediksi pembayaran normal. Sementara itu, model BP neural network memiliki akurasi 77,33% dalam memprediksi kasus tunggakan dan akurasi 82,9% dalam memprediksi pembayaran normal. Model CatBoost menunjukkan kinerja terbaik dengan akurasi prediksi sebesar 92,4% dan 94,0% untuk kasus tunggakan

2.2 *Loan Default*

Loan default adalah situasi ketika peminjam tidak dapat memenuhi kewajiban pembayaran pinjaman sesuai dengan perjanjian yang telah disepakati. *Loan default* dapat terjadi karena berbagai alasan, seperti ketidakmampuan peminjam untuk membayar karena masalah keuangan, kehilangan pekerjaan, atau

ketidakmampuan untuk memenuhi persyaratan pembayaran (Zhu dkk., 2023). Hal ini juga diperkuat oleh (Kurniawan & Supriyanto, 2013) yang mengungkapkan bahwa prediksi default pinjaman menjadi penting bagi lembaga keuangan untuk mengelola risiko kredit dan mengambil langkah-langkah pencegahan yang diperlukan.

2.3 *Data Mining*

Data mining adalah suatu proses analisis data yang bertujuan untuk mengidentifikasi pola, hubungan, dan wawasan berharga dalam dataset besar dan kompleks. Dengan menerapkan teknik komputasi dan statistik, *data mining* membantu mengungkap informasi tersembunyi dalam data, seperti tren pasar, perilaku pelanggan, atau anomali. Ini memberikan manfaat besar dalam pengambilan keputusan bisnis dengan memungkinkan perusahaan untuk merencanakan strategi yang lebih cerdas dan responsif terhadap perubahan lingkungan bisnis dan pasar (Zai, 2022).

2.4 Klasifikasi

Klasifikasi adalah salah satu tugas penting dalam *machine learning* yang melibatkan pengelompokan data ke dalam kategori atau kelas yang berbeda. *Data mining*, yang mencakup teknik-teknik seperti klasterisasi, regresi, asosiasi, pengelompokan, dan klasifikasi data, digunakan untuk menggali pola dan informasi yang dapat mendukung keputusan klasifikasi yang akurat (Ninditama, 2021).

2.5 *Data Balancing*

Data balancing, atau balancing data, adalah proses yang digunakan untuk mengatasi ketidakseimbangan dalam dataset yang digunakan dalam klasifikasi

pembelajaran mesin. Ketidakseimbangan dataset terjadi ketika terdapat rasio yang tidak proporsional di setiap kelas, sehingga algoritma pembelajaran mesin tidak dapat bekerja dengan baik. *Data balancing* dilakukan untuk mengurangi perbedaan rasio antara kelas mayoritas dan minoritas, sehingga algoritma dapat bekerja lebih efektif (Wicaksono dkk., 2024).

2.5.1 *SMOTE*

Dengan memilih data sampel hingga jumlah data sama dengan jumlah sampel di kelas mayoritas, pendekatan *SMOTE* menyeimbangkan distribusi data sampel di kelas minoritas (Kasanah dkk., 2019). *Overfitting* dapat terjadi saat menerapkan pendekatan *SMOTE* karena adanya duplikasi data dalam kelas minoritas menyebabkan set pelatihan menjadi identik, yang pada gilirannya dapat meningkatkan risiko overfitting. Langkah pertama dalam proses *SMOTE* adalah menghitung jarak antara data pada kelas minoritas. Selanjutnya, nilai persentase *SMOTE* ditentukan, diikuti oleh identifikasi jumlah k titik data terdekat. Terakhir, data yang disintesis dibuat dengan menggunakan persamaan berikut:

$$x_{syn} = x_{i+}(x_{knn} - x_i) \times \delta \quad (2.1)$$

Dimana:

x_{syn} adalah data sintesis yang akan diciptakan

x_i data yang akan direplikasi

x_{knn} data yang memiliki jarak terdekat dari data yang akan direplikasi

δ nilai random antara 0 dan 1

2.6 Normalisasi Data

Normalisasi data adalah proses penting dalam data mining untuk memastikan konsistensi pada record dalam suatu dataset. Normalisasi data membantu memastikan bahwa semua atribut berada pada skala yang sama, sehingga meningkatkan kualitas model saat melakukan operasi *data mining* (Pradnyana & Agustini, 2022).

2.6.1 Normalisasi *Min-Max*

Metode *Min-Max* adalah suatu teknik normalisasi yang mengubah jangkauan nilai data menjadi antara 0 dan 1. Cara untuk menghitung *Min-Max* dapat dirinci dalam persamaan 2.2 (Permana & Salisah, 2022).

$$x' = \frac{x_i - \min(x)}{\max(x) - \min(x)} \quad (2.2)$$

Dimana:

1. x_i = nilai tertentu yang akan dinormalisasi
2. x' = nilai hasil normalisasi
3. $\min(x)$ = nilai minimal dari sebuah atribut

4. $\max(x)$ = nilai maximal dari sebuah atribut

2.6.2 Normalisasi *Z-Score*

Normalisasi *Z-Score* adalah suatu teknik normalisasi yang menghasilkan nilai berdasarkan rata-rata dan standar deviasi dari data. Pendekatan ini menunjukkan ketahanan terhadap nilai yang jauh dari rata-rata atau berada di luar rentang maksimum (maksA) dan minimum (minA)(Suryanegara & Purbolaksono, 2021).

Perhitungan Normalisasi *Z-Score* dapat dilakukan dengan menggunakan rumus berikut:

$$n_i^1 = \frac{n_{i\,-\bar{A}}}{\sigma_A} \tag{2.3}$$

Keterangan:

1. n_i^1 = Hasil *Z-Score normalization*
2. n_i = data yang akan dinormalisasi
3. $\bar{A}$ = nilai rata-rata
4. σ_A = standar deviasi

2.7 Algoritma *CATBOOST*

CATBOOST adalah algoritma *boosting* yang inovatif, dikembangkan oleh *Yandex*, dengan fokus pada penanganan fitur kategorikal dalam data. Metode ini menonjol karena kemampuannya menangani fitur kategorikal secara langsung

tanpa memerlukan proses encoding tambahan seperti *One-Hot Encoding* atau Label *Encoding*, mengurangi kompleksitas *preprocessing* data (Nugraha & Syarif, 2023).

Dengan pendekatan *ordered boosting*, *CATBOOST* mampu mengatasi masalah data *leakage* yang sering terjadi dalam algoritma *boosting* lainnya. Lebih lanjut, keuntungan dari pohon keputusan simetris dan dukungan GPU mempercepat proses training, menjadikan *CATBOOST* pilihan utama untuk dataset besar. Dengan demikian, algoritma ini telah menjadi favorit di kalangan praktisi *machine learning* yang mengutamakan akurasi, kecepatan, dan kemudahan penggunaan.

Metode ini dirancang untuk mengatasi isu kebocoran target sambil tetap memanfaatkan secara optimal seluruh data pelatihan yang ada. Di sisi lain, *CATBOOST* mengadopsi strategi yang lebih efisien untuk mengurangi risiko *overfitting* dengan menggunakan keseluruhan data pelatihan.

Secara umum, langkah-langkah berikut ini menjelaskan cara menggunakan *CATBOOST* untuk mengkonversi fitur kategori menjadi fitur numerik:

1. Permutasi urutan acak dari objek pelatihan.
2. Kuantisasi adalah proses mengonversi nilai target dari titik desimal menjadi bilangan bulat. Pendekatan ini sangat tergantung pada jenis masalah *machine learning* yang sedang dihadapi.

   a. Dalam proses klasifikasi, nilai yang dapat diberikan pada target dapat berupa "0" (tidak termasuk dalam kelas target yang telah ditentukan) dan "1" (merupakan bagian dari kelas target yang telah ditentukan).

   b. Dalam klasifikasi multi-kelas, nilai target berupa identifikasi bilangan bulat yang mewakili kelas target (dimulai dari "0").

c. regresi, nilai label mengalami kuantisasi. Mode dan jumlah bucket diatur melalui parameter awal. Setiap nilai yang berada dalam satu bucket diberikan kelas nilai label, yaitu bilangan bulat dalam rentang yang ditentukan oleh rumus: <ID bucket - 1>.

3. Mentranskripsi nilai fitur kategori.

Mengkodekan nilai fitur kategoris. Dalam bentuk matematis, dapat direpresentasikan menggunakan persamaan di bawah ini:

Misalkan $\sigma = (\sigma 1, \ldots., \sigma n)$ adalah permutasi acak

$$x_{\sigma p,K} = \frac{\sum_{j=1}^{p=1}[x_{\sigma j,k}=x_{\sigma p,k}]y_{\sigma j}+a.p}{\sum_{j=1}^{p=1}[x_{\sigma j,k}=x_{\sigma p}]+a} \quad (2.4)$$

Keterangan :

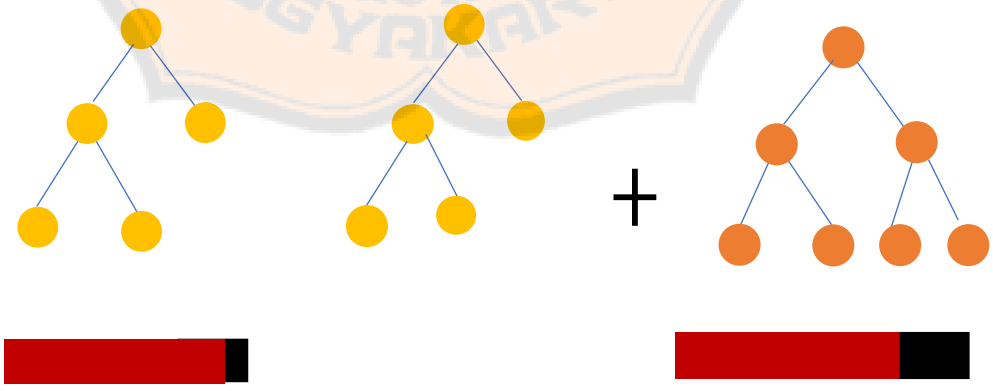
p : Prior yang biasanya dibentuk dari nilai rata-rata label dalam dataset.

a : Sebuah parameter yang lebih besar dari 0. Untuk memastikan tidak akan ada pembagian dengan 0 ketika nilai $x_{\sigma j,} \neq x_{\sigma p,.}$ Tidak ada saran yang pasti untuk menentukan nilai a.

Menambahkan prior merupakan metode untuk mengurangi gangguan dari kategori dengan frekuensi rendah. Dengan menerapkan persamaan (2.4) pada setiap contoh, rata-rata nilai label dihitung untuk contoh lain yang memiliki nilai yang sama dan ditempatkan sebelumnya.

Persamaan (2.4) tetap relevan saat melakukan konstruksi pohon keputusan dan memanfaatkan varians untuk mengevaluasi pohon keputusan. Selanjutnya, pohon keputusan yang menghasilkan nilai fungsi kerugian minimal akan dipilih. Perhitungan varians dilakukan dengan menggunakan seluruh data pelatihan. Sebagai contoh, transformasi fitur kategori menjadi fitur numerik.

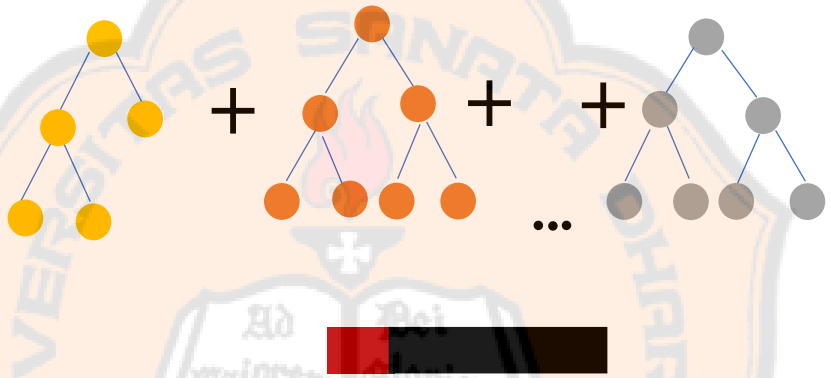
CATBOOST merupakan implementasi dari *Gradient Boosting on Decision trees* (GBDT) yang mempunyai kombinasi *gradient boosting* dengan pohon keputusan memberikan hasil yang maju di banyak aplikasi dengan data yang terstruktur. *CATBOOST* dikembangkan menggunakan teknik tersebut untuk membuat sebuah model ansambel secara berulang. Kemudian pada iterasi ke pertama, algoritma tersebut mempelajari pohon pertama untuk mengurangi kesalahan pada pelatihan yang dapat dilihat dari gambar di sebelah kiri pada gambar 2.1. Model tersebut mempunyai kesalahan yang signifikan, sehingga bukan ide yang baik untuk membangun pohon yang sangat besar untuk meningkatkannya karena terlalu terpaku pada data.



First Tree Second Tree

Gambar 2. 1 Iterasi Pohon Pertama dan Kedua pada *CATBOOST*

Gambar sebelah kanan pada gambar 2.1 menunjukkan iterasi kedua, di mana algoritma mempelajari satu pohon lagi untuk mengurangi kesalahan yang dibuat oleh pohon pertama. Algoritma mengulangi prosedur ini hingga membangun mode kualitas yang layak, seperti yang dapat dilihat pada gambar 2.2:



Gambar 2. 2 Iterasi Pohon ke-N

Pendekatan umum untuk klasifikasi pada *CATBOOST* dengan *Loglos* dan tugas pemeringkatan umumnya mengimplementasikan beberapa variasi *LambdaRank*. Setiap langkah *Gradient Boosting* menggabungkan dua langkah:

1. Menghitung gradien dari fungsi kerugian yang ingin kami optimalkan untuk setiap objek masukan
2. Mempelajari pohon keputusan yang memprediksi gradien dari fungsi kerugian.

GBDT pada *CATBOOST* membutuhkan pohon keputusan yang sesuai secara iteratif. Pohon keputusan klasifikasi yang digunakan mempelajari dengan

cara serakah, yang mengharuskan menghitung semua kemungkinan pemisahan fitur(nilai fitur kurang dari beberapa nilai yang ditentukan sebelumnya) dari semua fitur dalam data, lalu memilih fitur yang meningkatkan fungsi kerugian dengan nilai terbesar.

Setelah belahan pertama dipilih, belahan berikutnya dalam pohon akan dipilih dengan cara belahan pertama diperbaiki dan belahan berikutnya dipilih dengan memberikan belahan pertama. Operasi ini diulang sampai seluruh pohon dibangun. Skema pembelajaran *CATBOOST* pada dasarnya bersifat mendalam dengan beberapa penyederhanaan, yang diperoleh dari jenis pohon keputusan. Pemilihan *oblivious trees* memiliki beberapa keunggulan dibanding dengan pohon klasik :

1. Skema pemanasan sederhana
2. Efisien untuk diimplementasikan pada CPU
3. Kemampuan untuk membuat aplikasi model yang sangat cepat

Struktur pohon yang digunakan di *CATBOOST* juga berfungsi sebagai regularisasi, sehingga dapat memberikan manfaat yang berkualitas untuk banyak tugas. Algoritma pembelajaran pohon keputusan klasik berfisat komputasi intensif. Untuk menemukan pemisahan berikutnya, perlu mengevaluasi jumlah fitur kali jumlah pemisahan berikutnya, perlu mengevaluasi jumlah fitur kali jumlah pengamatan untuk kondisi pemisahan yang berbeda. Hal ini menyebabkan sejumlah besar kemungkinan pemisahan untuk kumpulan data besar menggunakan input berkelanjutan dan dalam banyak kasus, juga menyebabkan *overfitting*. Peningkatan memungkinkan *CATBOOST* secara signifikan mengurangi jumlah pemisahan yang

perlu dipertimbangkan. Sehingga, dapat membuat perkiraan kasar untuk fitur masukan.

Selanjutnya, terdapat fungsi cosine similarity yang digunakan untuk menilai nilai ambang (*threshold*) yang menjadi penentu awal (*root*) pada pohon keputusan. Fungsi ini juga berperan dalam memilih pohon keputusan dari berbagai kandidat yang diuji coba. Persamaan cosine similarity dituliskan sebagai berikut (Team, 2017) :

$$Cosine = \frac{\sum_{i=}^{n}{}_{1} w_i . \Delta_i . g_i}{\sqrt{\sum_{i=1}^{n} w_i \Delta_i^2} . \sqrt{\sum_{i=1}^{n} w_i g_i^2}} \quad (2.5)$$

Keterangan:

w_i : bobot objek ke-*i*

g_i : gradien pada objek ke-*i* yang sesuai dengan fungsi kerugian

Δi : nilai daun pada objek ke-*i*

2.8 *Cross-validation*

Cross-validation adalah metode statistik untuk mengevaluasi dan membandingkan algoritma pembelajaran dengan membagi data menjadi dua segmen yaitu satu digunakan untuk pembelajaran atau pelatihan model, dan yang lainnya untuk pengujian model. Ini adalah teknik validasi model yang umum digunakan untuk menilai kemampuan generalisasi model prediktif dan mencegah *overfitting*.

2.9 *Confusion Matrix*

Tabel yang digunakan untuk mengukur performa algoritma klasifikasi atau *Confusion Matrix* adalah matriks yang mencerminkan sejauh mana prediksi model klasifikasi sesuai atau tidak dengan hasil aktual. *Confusion Matrix* digunakan untuk menilai sejauh mana subjek dapat mengenali stimulus dengan akurat (Townsend, 1971). *Confusion matrix* untuk klasifikasi biner ditunjukkan pada Tabel 2.2:

Tabel 2. 2 *Confusion Matrix.*

Aktual	Prediksi	
	Positif	Negatif
Positif	TP	FN
Negatif	FP	TN

Dari Tabel 2.2, terlihat ada empat parameter yang digunakan untuk mengevaluasi performa model klasifikasi, yakni TP (*True Positive*), TN (*True Negative*), FP (*False Positive*), dan FN (*False Negative*). Untuk memahaminya lebih lanjut, berikut penjelasannya:

TP : suatu keadaan di mana model benar memprediksi data kelas positif.

TN : suatu keadaan di mana model benar memprediksi data kelas negatif.

FP : suatu keadaan di mana model salah memprediksi data kelas positif.

FN : suatu keadaan di mana model salah memprediksi data kelas negatif.

Confusion Matrix mengindikasikan jumlah TP, TN, FP, dan FN yang ada dalam data uji. Ukuran berdasarkan *confusion matrix.* mencakup akurasi, recall, presisi, dan F1-score, yang digunakan untuk mengidentifikasi model-model yang baik dalam mengklasifikasikan suatu set data tertentu. Akurasi adalah kemampuan

model untuk mengklasifikasikan data dengan akurat dan dapat diandalkan. *Recall* digunakan untuk mengukur sejauh mana model mampu mengidentifikasi hasil positif. Presisi merupakan tingkat keakuratan model dalam analisis regresi positif. F1-Score adalah rata-rata harmonis dari presisi dan recall (Dewi dkk., 2021).

Hasil uji untuk akurasi, recall, F1-score, dan presisi dapat diperoleh dengan menggunakan persamaan berikut:

$$Akurasi = \frac{TP+TN}{TP+TN+FP+FN} \quad (2.4)$$

$$Recall = \frac{TP}{TP+FN} \quad (2.5)$$

$$Presisi = \frac{TP}{TP+Fp} \quad (2.6)$$

$$F1 - score = 2\ x\ \frac{Recall \times Presisi}{Recall+presisi} \quad (2.7)$$

Keterangan :

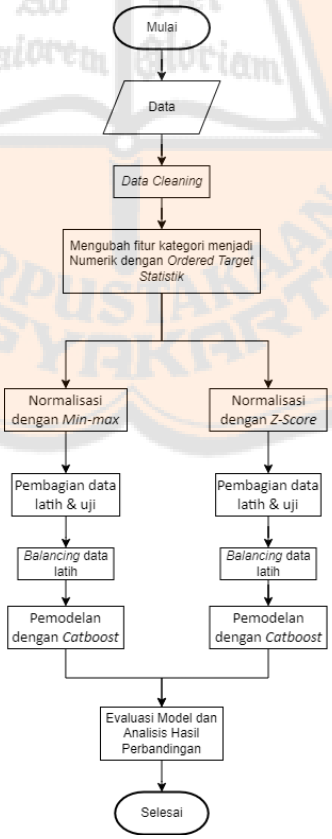
1. *Akurasi* : persentase keberhasilan model dalam mengklasifikasikan data secara benar
2. *Recall* : metrik yang mengukur kemampuan model klasifikasi untuk mengidentifikasi semua instance positif yang sebenarnya dalam dataset.
3. *Presisi* : metrik yang mengukur sejauh mana model klasifikasi benar dalam mengidentifikasi instance positif.
4. *F1-score* : rata-rata dari presisi dan recall.

BAB III

METODOLOGI PENELITIAN

3.1 Gambaran Umum Penelitian

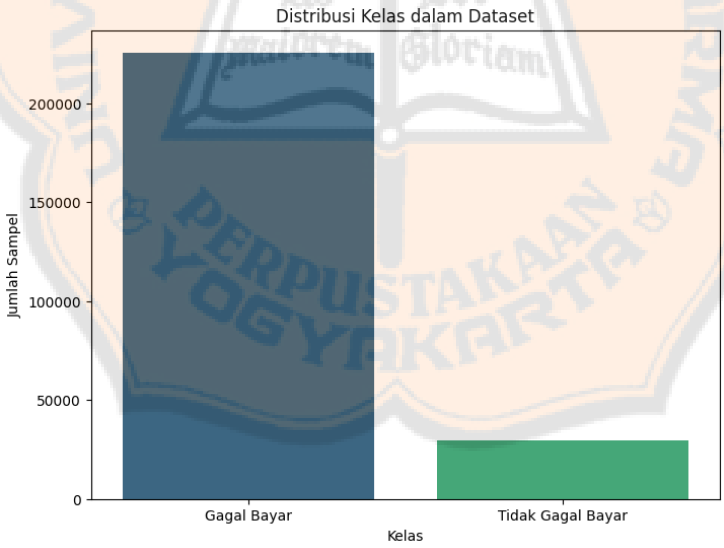
Data yang akan digunakan dalam penelitian ini adalah data tentang kegagalan pembayaran(*Loan Default*) di sector perbankan. Data tersebut diambil dari situs kaggle: https://www.kaggle.com/datasets/nikhil1e9/loan-default/data yang diakses pada tanggal 3 oktober 2023. Data yang telah melewati serangkaian proses kemudian dilakukan klasifikasi dengan menggunakan algoritma *CATBOOST*. Gambaran umum dalam penelitian ini dapat dilihat pada gambar 3.1 berikut:



Gambar 3. 1 *Flowchart* Perancangan Sistem Secara Umum

3.2 Data

Data yang digunakan pada studi ini yaitu *loan default* yang didapatkan dari kaggle. *Dataset* ini memiliki 18 atribut. Salah satu atribut dengan judul default digunakan sebagai label atau target. Namun data tersebut tidak seimbang karena jumlah orang yang gagal membayar pinjaman hanya 225,694 dari 255,348, dan yang tidak gagal membayar pinjaman 29,653, sehingga perlu dikenai balancing terlebih dahulu. Pendekatan analisis yang digunakan dalam penelitian ini menggunakan metode KDD (*Knowledge Discovery in Database*), yang merupakan prosedur *data mining* yang digunakan untuk menemukan informasi dan pola yang berguna dalam data. Berikut merupakan grafik batang dari dataset kredit bank, dapat dilihat pada gambar 3.2.



Gambar 3. 2 Grafik Batang Data Kredit Bank

Atribut dari data yang digunakan untuk penelitian ini dipaparkan pada tabel 3.1 dibawah ini:

Tabel 3. 1 Penjelasan Atribut Data

No	Nama Kolom	Tipe Kolom	Tipe Data	Contoh Nilai	Penjelasan
1	LoanID	Identifier	string	I38PQUQS96, HPSK72WA7R	Kolom yang berisi identifier unik untuk setiap pinjaman.
2	Age	Feature	integer	56 69 46	Berisi usia peminjam, yaitu usia orang yang mengajukan pinjaman.
3	Income	Feature	integer	8594 50432 84208	Kolom yang berisi pendapatan tahunan dari peminjam, yaitu jumlah uang yang mereka hasilkan dalam satu tahun.
4	LoanAmount	Feature	integer	50587 124440 129188	Kolom yang berisi jumlah uang yang sedang dipinjam oleh peminjam dalam mata uang tertentu.
5	CreditScore	Feature	integer	520 458 451	Berisi nilai kredit peminjam, yang mengindikasikan seberapa kreditnya mereka dalam hal membayar pinjaman.
6	MonthsEmployed	Feature	integer	80 15 26	Kolom yang mencantumkan jumlah bulan peminjam telah bekerja.
7	NumCreditLines	Feature	integer	1 2 3 4	Berisi jumlah garis kredit yang dimiliki oleh peminjam, yaitu jumlah pinjaman

					atau kartu kredit yang mereka miliki.
8	InterestRate	Feature	float	15.23 4.81 21.17	mendeskripsikan suku bunga yang diterapkan pada pinjaman, dalam bentuk desimal (misalnya, 0,05 untuk 5%).
9	LoanTerm	Feature	integer	36 60 24	Merupakan durasi pinjaman dalam bulan, yaitu berapa lama pinjaman akan berlangsung.
10	DTIRatio	Feature	float	0.44 0.68 0.31	Berisi tentang rasio Utang-ke-Pendapatan (Debt-to-Income Ratio) peminjam, yang mengindikasikan seberapa besar utang mereka dibandingkan dengan pendapatan mereka.
11	Education	Feature	string	Bachelor's Master's High School	Berisi tentang tingkat pendidikan tertinggi yang dicapai oleh peminjam (misalnya, PhD, Master's, Bachelor's, High School).
12	EmployementType	Feature	string	Full-time Unemployed	Ini adalah kolom yang mencantumkan jenis status pekerjaan peminjam

					(misalnya, Full-time, Part-time, Self-employed, Unemployed).
13	MaritalStatus	Feature	string	Divorced Married	kolom yang mencantumkan status pernikahan peminjam (misalnya, Single, Married, Divorced).
14	HasMortgage	Feature	string	Yes No	Merupakan kolom yang mencantumkan apakah peminjam memiliki hipotek (Yes atau No).
15	HasDependents	Feature	string	Yes No	Berisi kolom yang mencantumkan apakah peminjam memiliki tanggungan (Yes atau No).
16	LoanPurpose	Feature	string	Other Auto Business	kolom yang berisi tujuan dari pinjaman (misalnya, Home, Auto, Education, Business, Other).
17	HasCoSigner	Feature	string	Yes No	kolom yang mencantumkan apakah pinjaman memiliki co-signer (Yes atau No).
18	Default	Target	integer	0 1	Merupakan kolom target yang berisi nilai biner, yang menunjukkan apakah pinjaman mengalami default atau tidak. Nilai 1 mewakili default, sementara nilai 0 mewakili

					tidak default. Ini adalah apa yang ingin diprediksi oleh model analisis.

3.3 *Preprocessing*

Pada tahapan ini, data akan melalui beberapa tahapan *preprocessing*. Tahapan yang akan dilalui adalah data *cleaning*, *encoding* dan normalisasi data. Tahapan ini bertujuan untuk membuat kualitas dataset yang akan digunakan menjadi lebih optimal dan lebih terstruktur agar dapat menunjang performa model *CATBOOST* yang akan digunakan.

3.3.1 *Data Cleaning*

Data cleaning adalah suatu tahap dalam analisis data di mana kita membersihkan dataset dari berbagai masalah yang dapat mengganggu analisis yang akurat. Proses ini mencakup identifikasi dan penanganan nilai yang hilang, *outlier*, atau kesalahan dalam format data yang dapat mempengaruhi hasil analisis (Kasanah dkk., 2019).

Sebelum memulai proses pertambangan data (*data mining*), langkah penting yang perlu dilakukan adalah pembersihan data (*data cleaning*) yang mencakup identifikasi dan penanganan nilai yang hilang, data berlebihan, dan format data yang tidak sesuai dengan sistem, karena *preprocessing* merupakan upaya untuk mengatasi masalah yang dapat mempengaruhi hasil dari proses klasifikasi data.

3.3.2 Mengubah Fitur Kategori Menjadi Fitur Numerik Dengan *Ordered Target Statistic*

Sebelum melakukan teknik *data balancing*, fitur kategori diubah menjadi fitur numerik terlebih dahulu. Tabel 3.2 di bawah ini menampilkan output dari langkah ini.

Tabel 3. 2 Contoh Data Input

Age	Income	LoanAmount	CreditScore	MonthsEmployed	HasDependents	Default
32	31713	44799	743	0	No	0
40	132784	228510	480	114	No	0
28	140466	163781	652	94	No	0
61	62519	29676	462	16	No	0
23	17142	110469	802	56	Yes	0
54	74954	242196	635	59	Yes	0
45	54095	34835	770	94	Yes	0
32	125786	239618	338	73	No	1
34	117943	231285	337	81	Yes	1
23	74090	205698	571	39	Yes	1

Tabel 3.2 menampilkan contoh data yang terdiri dari tujuh objek dengan 6 fitur, yaitu Age, Income, LoanAmount, CreditScore, MonthsEmployed, dan HasDependents. Di sini fitur Age, Income, LoanAmount, CreditScore, MonthsEmployed, adalah fitur numerik, sedangkan HasDependents merupakan fitur kategori dengan 2 kategori, yakni Yes, dan No. Fitur kategori ini akan diubah menjadi fitur numerik.

4. Beberapa kali, baris dalam data input diacak dan menghasilkan beberapa permutasi acak. Tabel di bawah menampilkan hasil dari langkah ini.

Tabel 3. 3 Data Input Setelah dilakukan Pengacakan

Age	Income	LoanAmount	CreditScore	MonthsEmployed	HasDependents	Default
40	132784	228510	480	114	No	0
28	140466	163781	652	94	No	0
23	74090	205698	571	39	Yes	1
32	31713	44799	743	0	No	0
32	125786	239618	338	73	No	1
34	117943	231285	337	81	Yes	1
61	62519	29676	462	16	No	0
23	17142	110469	802	56	Yes	0
54	74954	242196	635	59	Yes	0
45	54095	34835	770	94	Yes	0

Pada Tabel 3.3, baris dalam data diacak sehingga urutan objek menjadi berbeda dengan urutan pada tabel sebelumnya, yaitu Tabel 3.2. Sebagai contoh, terlihat bahwa objek pertama di Tabel 3.3 memiliki Age = 40, Income = 132784, LoanAmount = 228510, CreditScore = 480 MonthsEmployed = 114 HasDependents = No dan Default dengan nilai 0, sedangkan objek pertama di Tabel 3.3 memiliki Age = 32, Income = 31713, LoanAmount = 44799, CreditScore = 743 MonthsEmployed = 0 HasDependents = No dan Default dengan nilai 0.

5. Semua nilai fitur kategori diubah menjadi bentuk numerik dengan menerapkan persamaan (3.3) sebagaimana telah dijelaskan sebelumnya:

$$x_{\sigma p,k} = \frac{\sum_{j-1}^{p-1}[x_{\sigma j,k} = x_{\sigma p,k}]Y_{\sigma j} + a \,.p}{\sum_{j=1}^{p-1}[x_{\sigma j,k} = x_{\sigma p}] + a} \quad (3.2)$$

Dengan memilih nilai a = 1, persamaan di atas dapat juga dirumuskan sebagai berikut:

$$avg\ target = \frac{countInClass + Prior}{totalCount + 1} \quad (3.3)$$

Di mana:

$countInClass$: Berapa kali nilai label sama dengan "1" untuk objek dengan nilai fitur kategori saat ini (perhitungan dibuat berdasarkan urutan objek setelah pengacakan).

$Prior$: Angka (konstanta) yang ditentukan oleh parameter awal

$totalCount$: Jumlah total objek (hingga objek sekarang) yang memiliki nilai fitur kategori yang sama dengan yang sekarang.

Nilai-nilai tersebut dihitung secara terpisah untuk setiap objek dengan memanfaatkan informasi dari objek sebelumnya.

Dalam kasus Gagal bayar contoh, terdapat 2 kateogori nilai, yaitu "Yes," dan “No”, dengan prior = 0,05. Perhitungan *ordered target statistic* dapat dijelaskan sebagai berikut:

$$Object\ 1 = avg\ target = \frac{0+0{,}05}{0+1} = 0{,}05$$

$$Object\ 2 = avg\ target = \frac{0+0{,}05}{0+1} = 0{,}05$$

$$Object\ 3 = avg\ target = \frac{1+0,05}{1+1} = 0.525$$

$$Object\ 4 = avg\ target = \frac{0+0,05}{2+1} = 0,01667$$

$$Object\ 5 = avg\ target = \frac{1+0,05}{3+1} = 0,2625$$

$$Object\ 6 = avg\ target = \frac{1+0,05}{4+1} = 0,21$$

$$Object\ 7 = avg\ target = \frac{0+0,05}{5+1} = 0.00833$$

$$Object\ 8 = avg\ target = \frac{0+0,05}{6+1} = 0.00714$$

$$Object\ 9 = avg\ target = \frac{0+0,05}{7+1} = 0.00625$$

$$Object\ 10 = avg\ target = \frac{0+0,05}{8+1} = 0.00556$$

Tabel 3. 4 Data Setelah diubah Fitur Kategeori Menjadi Numerik

Age	Income	LoanAmount	CreditScore	MonthsEmployed	HasDependents	Default
40	132784	228510	480	114	0,05	0
28	140466	163781	652	94	0.05	0
23	74090	205698	571	39	0.525	1
32	31713	44799	743	0	0.01667	0
32	125786	239618	338	73	0.2625	1
34	117943	231285	337	81	0.21	1
61	62519	29676	462	16	0.00833	0
23	17142	110469	802	56	0.00714	0
54	74954	242196	635	59	0.00625	0
45	54095	34835	770	94	0.00556	0

Pada table 3.4 menunjukkan bahwa fitur HasDependents telah di ubah menjadi fitur numerik.

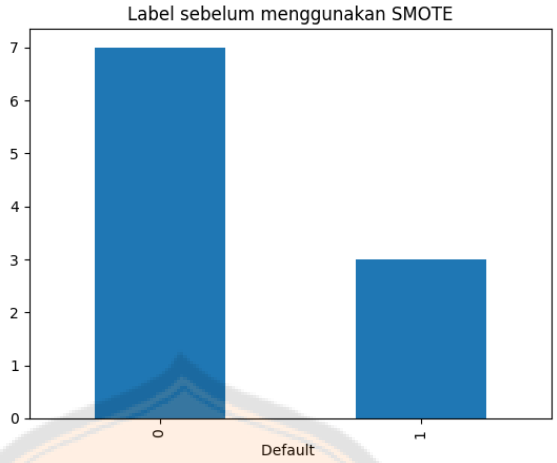
3.3.3 *Synthetic Minority Oversampling Technique*

Pada tahap ini data akan diseimbangkan menggunakan teknik *Synthetic Minority Oversampling Technique* (*SMOTE*). Pada kasus prediksi kegagalan pembayaran pinjaman, data yang digunakan memiliki ketidakseimbangan antara kelas pelanggan yang gagal membayar (kelas minoritas) dan yang tidak gagal membayar (kelas mayoritas). *SMOTE* bekerja dengan cara mensintesis sampel baru dari kelas minoritas dengan membuat kombinasi linear dari sampel yang ada. Berikut merupakan penerapan *SMOTE* dengan Data Sampel, dapat dilihat pada tabel 3.5.

Tabel 3. 5 Contoh *Data Imbalance*

Age	Income	LoanAmount	CreditScore	MonthsEmployed	HasDependents	Default
40	132784	228510	480	114	0,05	0
28	140466	163781	652	94	0.05	0
23	74090	205698	571	39	0.525	1
32	31713	44799	743	0	0.01667	0
32	125786	239618	338	73	0.2625	1
34	117943	231285	337	81	0.21	1
61	62519	29676	462	16	0.00833	0
23	17142	110469	802	56	0.00714	0
54	74954	242196	635	59	0.00625	0
45	54095	34835	770	94	0.00556	0

Tabel 3.5 di atas menunjukkan bahwa ada 10 item dengan total 6 atribut: Age, Income, LoanAmount, CreditScore, MonthsEmployed, dan HasDependents yang merupakan fitur numerik. Jumlah item di setiap kelas dalam data tidak sama banyak yaitu terdapat perbandingan 7:3, seperti pada gambar berikut:



Gambar 3. 3 Distribusi kelas menggunakan data sampel

Ketidakseimbangan dalam jumlah kedua kelas ditunjukkan pada Gambar 3.3. Terlihat bahwa kelas yang mempunyai tujuh objek berada di Kelas 1 sedangkan hanya tiga objek di Kelas 2, Kelas Satu mengungguli Kelas Dua. Dengan menggunakan teknik *SMOTE*, data kelas dua diperbarui agar sesuai dengan data kelas satu untuk mencapai keseimbangan data di antara kedua kelas.

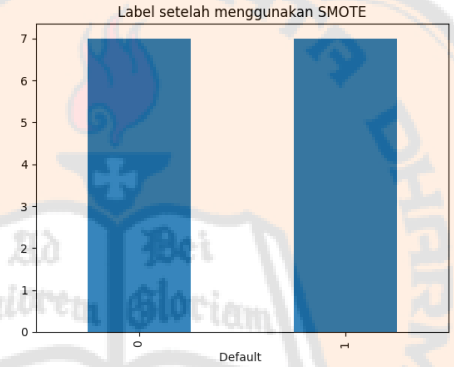
Berikut merupakan tabel hasil perhitungan *SMOTE* pada data *imbalance* penelitian ini:

Tabel 3. 6 Hasil Perhitungan *SMOTE* terhadap *data imbalance*

Age	Income	LoanAmount	CreditScore	MonthsEmployed	HasDependents	Default
40	132784	228510	480	114	0,05	0
28	140466	163781	652	94	0,05	0
23	74090	205698	571	39	0,525	1
32	31713	44799	743	0	0,01667	0
32	125786	239618	338	73	0,2625	1
31	129707,5	243784,5	338,5	69	0,28875	1
29,5	135589,75	250034,25	339,25	63	0,328125	1
27,2	144413,12	259408,875	340,37	54	0,3871875	1
23,8	157648,18	273470,812	342,06	40,5	0,47578	1
34	117943	231285	337	81	0,21	1

61	62519	29676	462	16	0,00833	0
23	17142	110469	802	56	0,00714	0
54	74954	242196	635	59	0,00625	0
45	54095	34835	770	94	0,00556	0

Tabel 3.6 di atas menunjukkan bahwa data telah seimbang dengan perbandingan 7:7, yaitu terlihat pada tabel 3.6 telah ditambahkan 4 sampel sintetis baru pada data imbalance sebelumnya, berikut merupakan gambar plot hasil penggunaan *SMOTE*:



Gambar 3. 4 Plot Hasil Sesudah Data Seimbang

Pada gambar 3.4 terlihat bahwa kelas yang mempunyai tujuh objek berada di kelas 1 sedangkan 7 objek lainya berada di kelas 2 yang berarti data tersebut telah seimbang dan berbanding 7:7, Dengan menggunakan teknik *SMOTE*, data nilai 0 diperbarui agar sesuai dengan data nilai 1 untuk mencapai keseimbangan data di antara kedua kelas.

3.3.4 Normalisasi

Normalisasi data adalah proses mengubah nilai-nilai dalam suatu dataset sehingga data tersebut memiliki skala atau rentang yang seragam. Hal ini juga diperkuat oleh (Kurniawan & Supriyanto, 2013) yang mengungkapkan bahwa mengubah skala nilai set data untuk konsistensi, kemudahan analisis, dan mengurangi dampak outlier adalah proses normalisasi data. Berikut merupakan tabel data sebelum dinormalisasi dapat dilihat pada tabel 3.7.

Tabel 3. 7 Data Sebelum dinormalisasi

Age	Income	LoanAmount	CreditScore	Months Employed	Has Dependents	Default
40	132784	228510	480	114	0,05	0
28	140466	163781	652	94	0,05	0
23	74090	205698	571	39	0,525	1
32	31713	44799	743	0	0,01667	0
32	125786	239618	338	73	0,2625	1
31	129707,5	243784,5	338,5	69	0,28875	1
29,5	135589,75	250034,25	339,25	63	0,32812	1
27,2	144413,12	259408,875	340,375	54	0,38718	1
23,8	157648,18	273470,812	342,0625	40,5	0,47578	1
34	117943	231285	337	81	0,21	1
61	62519	29676	462	16	0,00833	0
23	17142	110469	802	56	0,00714	0
54	74954	242196	635	59	0,00625	0
45	54095	34835	770	94	0,00556	0

Tabel 3.7 merupakan data awal sebelum dinormalisasi. Atribut-atribut yang digunakan untuk perhitungan menggunakan metode *Min-Max* yaitu Age, Income, LoanAmount, CreditScore, MonthsEmployee, HasDependents dan labelnya atau targetnya yaitu atribut Default. Berikut merupakan hasil data *Min-Max* normalization dapat dilihat pada tabel 3.8.

3.3.4.1 Normalisasi *Min-Max*

Normalisasi *Min-Max* adalah teknik transformasi linier yang menciptakan keseimbangan antara data dalam rentang yang sama dengan menggunakan nilai minimum dan maksimum (Suryanegara & Purbolaksono, 2021). Berikut merupakan hasil perhitungan normalisasi *Min-Max* dapat dilihat pada tabel 3.8.

Tabel 3. 8 Hasil Perhitungan Normalisasi *Min-Max*

Age	Income	LoanAmount	Credit Score	Months Employed	Has Dependents	Default
0,447368421	0,823038487	0,81557929	0,30752	1	0,08555367	0
0,131578947	0,877712236	0,55007323	0,67741	0,82456140	0,08555367	0
0	0,405305994	0,7220088	0,50322	0,34210526	1	1
0,236842105	0,103703618	0,062031673	0,87311	0	0,02138841	0
0,236842105	0,773232851	0,861142195	0,00215	0,64035087	0,49464808	1
0,210526316	0,801142654	0,878232387	0,00322	0,60526315	0,54518327	1
0,171052632	0,843007359	0,903867674	0,00483	0,55263157	0,62098606	1
0,111842105	0,905804415	0,942320604	0,00725	0,47368421	0,73469024	1
0,023026316	1	1	0,01088	0,35526315	0,90524651	1
0,289473684	0,717413246	0,826961812	0	0,71052631	0,39357769	1
1	0,322953749	0	0,26881	0,14035087	0,00533266	0
0	0	0,331397535	1	0,49122807	0,00304173	0
0,815789474	0,411455189	0,871716661	0,64086	0,51754386	0,00132835	0
0,578947368	0,262999094	0,021161238	0,93118	0,82456140	0	0

Pada table 3.8 terlihat bahwa data yang telah berhasil dinormalisasi menggunakan metode *Min-Max* yang menunjukkan variasi umur pelanggan dari muda hingga menengah. Terdapat variasi yang signifikan dalam pendapatan dan jumlah pinjaman, dengan sebagian besar pelanggan memiliki tingkat pendapatan dan jumlah pinjaman yang tinggi. Skor kredit cenderung baik, dan mayoritas pelanggan memiliki rekam jejak pekerjaan

yang cukup lama. Sebagian besar pelanggan tidak memiliki tanggungan, tetapi beberapa di antaranya mengalami gagal bayar (Default).

3.3.4.2 Normalisasi *Z-Score*

Normalisasi *Z-Score* adalah jenis normalisasi di mana rata-rata dan standar deviasi data digunakan untuk menentukan hasil. Berikut merupakan hasil sampel data *Z-Score* normalisasi dapat dilihat pada tabel 3.9.

Tabel 3. 9 Hasil Perhitungan Normalisasi *Z-Score*

Age	Income	LoanAmount	CreditScore	MonthsEmployed	HasDependents	Default
0,470842161	0,714800772	0,512288406	-0,169961229	1,724623671	-0,721273103	-0,963624112
-0,564856472	0,881875632	-0,211304539	0,781406812	1,075135267	-0,721273103	-0,963624112
-0,996397569	-0,56172772	0,25727748	0,333378839	-0,710957841	1,775204146	0,963624112
-0,219623594	-1,483379813	-1,541381257	1,284746881	-1,977460228	-0,89644697	-0,963624112
-0,219623594	0,562602142	0,636462586	-0,955392984	0,393172444	0,395571982	0,963624112
-0,305931813	0,647890356	0,683039082	-0,952627379	0,263274763	0,533535198	0,963624112
-0,435394142	0,775822679	0,752903826	-0,948478972	0,068428242	0,740480023	0,963624112
-0,629587636	0,967721162	0,857700942	-0,942256361	-0,223841539	1,05089726	0,963624112
-0,920877877	1,255568887	1,014896615	-0,932922445	-0,662246211	1,516523115	0,963624112
-0,047007155	0,392025712	0,543309594	-0,960924194	0,652967805	0,119645549	0,963624112
2,283314768	-0,813383957	-1,710438343	-0,269523001	-1,457869505	-0,940279855	-0,963624112
-0,996397569	-1,800282676	-0,807269055	1,611088244	-0,158892699	-0,946534187	-0,963624112
1,679157233	-0,542936692	0,665281548	0,68737625	-0,061469438	-0,951211797	-0,963624112
0,902383258	-0,996596484	-1,652766882	1,434089539	1,075135267	-0,954838259	-0,963624112

Dari table 3.9 dapat diambil kesimpulan bahwa data yang telah dinormalisasi menggunakan *Z-Score* menunjukkan bahwa sebagian besar pelanggan memiliki skor *Z-Score* yang mendekati nol untuk atribut umur, pendapatan, jumlah pinjaman, dan skor kredit. Meskipun demikian, beberapa pelanggan memiliki skor *Z-Score* yang cukup tinggi untuk lama bekerja dan tanggungan, menunjukkan bahwa mereka berada di atas rata-rata dalam hal variabilitas. Hasil ini menunjukkan distribusi data yang lebih merata dan normal, dengan mayoritas pelanggan berada dalam kisaran nilai yang serupa. Terdapat potensi risiko gagal bayar yang dapat diperhatikan pada pelanggan dengan skor *Z-Score* yang tinggi untuk lama bekerja dan tanggungan.

3.4 Data Latih dan Data Uji

Pada tahapan ini dataset akan dibagi menjadi set pelatihan dan set pengujian, yang sangat membantu model machine learning dalam pelatihan menggunakan data untuk memprediksi kredit macet. Efektivitas model *CATBOOST* dalam mengidentifikasi data baru di luar set pelatihan dinilai dengan menggunakan rasio 80/20 atau 80% data latih dan 20% data uji. Memperluas set data pelatihan akan meningkatkan kemampuan model untuk mengidentifikasi variasi data baru.

3.5 Pemodelan *CATBOOST*

Pada tahap ini, disajikan contoh menggunakan sampel data yang sederhana untuk menunjukkan cara algoritma *CATBOOST* membangun pohon keputusan hingga mampu melakukan klasifikasi. Berikut merupakan langkah-langkah dari *CATBOOST* untuk membangun pohon:

1. Inisialisasi Prediksi dan *Residual*

Mulai dengan prediksi awal 0 untuk semua data. Hitung *residual* sebagai perbedaan antara nilai aktual (‘*Default’*) dan prediksi awal.

2. Hitung *Residual* Awal

Residual adalah selisih antara nilai aktual (‘*Default’*) dan prediksi awal:

$$Residual = Default - Prediksi \quad (3.4)$$

Berikut merupakan tabel awal dengan prediksi dan *residual*:

Tabel 3. 10 Data awal dengan prediksi dan *residual*

Age	Default	Prediksi	Residual
0.447368	0	0	0
0.131578	0	0	0
0	1	0	1
0.236842	0	0	0
0.236842	1	0	1
0.210526	1	0	1
0.171053	1	0	1

3. Tahap selanjutnya yaitu menemukan titik split terbaik(*Threshold*)

Urutkan berdasarkan fitur yang dipilih (*Age*) dan identifikasi titik split potensial. Titik split potensial adalah nilai tengah antara dua nilai berurutan dari *Age*. Sebelum itu urutkan data urutkan data berdasarkan fitur *Age*, dapat dilihat pada table di bawah ini :

Tabel 3. 11 Data Setelah diurutkan fitur *Age*

Age	Default	Prediksi	Residual
0	1	0	1
0.131578	0	0	0
0.171053	1	0	1
0.210526	1	0	1
0.236842	0	0	0
0.236842	1	0	1
0.171053	1	0	1

4. Menentukan *Threshold* Potensial

Threshold potensial adalah nilai tengah antara dua nilai berurutan:

$$Threshold\ 1: (0 + 0.131578) / 2 = 0.065789$$

$$Threshold\ 2: (0.131578 + 0.171053) / 2 = 0.151315$$

$$Threshold\ 3: (0.171053 + 0.210526) / 2 = 0.190789$$

$$Threshold\ 4: (0.210526 + 0.236842) / 2 = 0.223684$$

$$Threshold\ 5: (0.236842 + 0.236842) / 2 = 0.236842$$

$$Threshold\ 6: (0.236842 + 0.447368) / 2 = 0.342105$$

5. Hitung Cosine Similarity untuk Setiap *Threshold*

Untuk setiap *threshold*, bagi data ke dalam dua kelompok (daun) dan hitung cosine similarity. Berikut merupakan perhitungan *cosine similarity* untuk threshold pertama (0.065789):

- Daun Kiri (Age <= 0.065789):

  Data: [0]

  Residual: [1]

  Mean Residual: 1

- Daun Kanan (Age > 0.065789):

  Data: [0.131578, 0.171053, 0.210526, 0.236842, 0.236842, 0.447368]

  Residual: [0, 0, 1, 0, 1, 0]

  Mean Residual: 0.333

Berikutnya perbarui prediksi dengan *learning rate* 0.1:

$$Prediksi\ Baru\ untuk\ Daun\ Kiri\ = 0 + 0.1 \times 1 = 0.1$$

$$Prediksi\ Baru\ untuk\ Daun\ Kanan\ = 0 + 0.1 \times 0.333$$

$$= 0.0333$$

Selanjutnya menghitung *cosine similarity* untuk *threshold* = 0.065789:

$$Residuals: [1, 0, 0, 1, 0, 1, 0]$$

$$Prediksi\ Baru: [0.05, 0.05, 0.05, 0.05, 0.05, 0.05, 0]$$

$$Dot\ Product: (1 \times 0.05) + (0 \times 0.05) + (0 \times 0.05) + (1 \times 0.05) + (0 \times 0.05) + (1 \times 0.05) + (0 \times 0) = 0.05 + 0.05 + 0.05 = 0.15$$

$$Magnitude\ of\ Residuals: \sqrt{1^2 + 0^2 + 0^2 + 1^2 + 0^2 + 1^2 + 0^2} = \sqrt{3} = 1.732$$

$$Magnitude\ of\ Prediksi\ Baru:$$

$$\sqrt{0.1^2 + 0.0333^2 + 0.0333^2 + 0.0333^2 + 0.0333^2 + 0.0333^2 + 0.0333^2} =$$

$$\sqrt{0.01 + 6 \times 0.00111} = \sqrt{0.01666} = 0.129$$

$$Cosine\ Similarity = \frac{0.1666}{1.732 \times 0.129} = \frac{0.1666}{0.223} = 0.747$$

Cosine similarity untuk threshold ini adalah 0.747. Dari hasil perhitungan *cosine similarity* untuk setiap *threshold*, kita memperoleh nilai sebagai berikut:

- *Threshold* 0.065789: 0.747
- *Threshold* 0.151316: 0.524
- *Threshold* 0.190789: 0.67

- *Threshold* 0.223684: 0.67
- *Threshold* 0.342105: 0.711

6. Pilih *Threshold* Terbaik dan Perbarui Prediksi

Dengan demikian, *threshold* terbaik berdasarkan *cosine similarity* adalah 0.065789 dengan nilai *cosine similarity* tertinggi yaitu 0.747. Langkah selanjutnya yaitu setelah mendapatkan threshold terbaik selanjutnya yaitu memperbarui prediksi. Berikut ini adalah tabel prediksi setelah memilih *threshold* terbaik dan memperbarui prediksi:

Tabel 3. 12 Tabel Prediksi Setelah Pemilihan *Threshold* Terbaik

Age	Default	Prediksi	Residual
0.447368	0	0.0333	-0.0333
0.131578	0	0.0333	-0.0333
0	1	0.1	0.9
0.236842	0	0.0333	-0.0333
0.236842	1	0.0333	0.9667
0.210526	1	0.0333	0.9667
0.171053	1	0.0333	0.9667

Dari table 3.12, prediksi telah diperbarui setelah memilih *threshold* terbaik (0.065789) dan menghitung *residual* baru berdasarkan perbedaan antara nilai *aktual* dan prediksi yang baru.

Selanjutnya menghitung metrik evaluasi seperti akurasi, presisi, *recall*, dan *F1-score*, kita perlu menggunakan informasi dari *confusion matrix* sebagai berikut.

$$Accuracy = \frac{3+3}{3+0+3+1} = \frac{6}{7} = 0.857$$

$$Precision = \frac{3}{3+10} = 1.0$$

$$Recall = \frac{3}{3+1} = \frac{3}{4} = 0.75$$

$$F1 - score = \frac{2 \times 1.0 \times 0.75}{1.0 + 0.75} = \frac{1.5}{1.75} = 0.857$$

Tabel 3. 13 Hasil Akurasi *CATBOOST*

	Nilai Sebenarnya	
Prediksi	**T**	**F**
T	3 TP	1 FN
F	0 FP	3 TN
Akurasi	**85.71%**	

3.6 Evaluasi Model dan Analisis Hasil

Setelah model dilatih, data *testing* dapat digunakan untuk menilai kinerjanya dan menentukan fitur mana yang paling penting. Fitur-fitur ini dapat diwakili oleh *performance metrics* seperti *Confusion Matrix, Accuracy, Precision & Recall, F1 Score, dan Area Under ROC Curve (AUC)*.

3.7 Skenario Pengujian

Model akan diuji dengan beberapa skenario yang digunakan. Pada pengujian ini akan dilakukan 4 kali percobaaan menggunakan nilai k-fold yaitu 3, 5, 7, dan 10, dengan data yang telah di normalisasi menggunakan *Min-Max normalization* pada *CATBOOST* dan menggunakan *Z-Score normalization* pada *CATBOOST*, dapat dilihat pada tabel 3.14.

Tabel 3. 14 Skenario Pengujian Metode

No	K-fold	Data	Metode
1	3	Normalisasi *Min-Max*	*CATBOOST*
		Normalisasi *Z-Score*	
2	5	Normalisasi *Min-Max*	
		Normalisasi *Z-Score*	
3	7	Normalisasi *Min-Max*	
		Normalisasi *Z-Score*	
4	10	Normalisasi *Min-Max*	
		Normalisasi *Z-Score*	

BAB IV

IMPLEMENTASI, HASIL DAN PEMBAHASAN

Pada bagian ini akan menjelaskan tentang implementasi dari perangkat lunak yang digunakan, hasil yang di dapatkan dan pembahasan dari hasil yang digunakan implementasi berupa *preprocessing*, pembagian data latih dan data uji, *modelling* dan hasil evaluasi.

4.1 *Preprocessing*

Preprocessing merupakan tahapan awal dalam proses persiapan dan pemrosesan data mentah sebelum melakukan analisis lebih lanjut untuk memastikan hasil analisis yang akurat dan relevan. Ada beberapa tahapan yang akan dilakukan yaitu data *cleaning*, mengubah fitur kategori menjadi numerik dengan *ordered target statistic* dan juga normalisasi data.

4.1.1 *Data Cleaning*

Pada tahapan ini, bertujuan untuk memastikan bahwa data yang digunakan dalam analisis atau pemodelan bebas dari noise, error, duplikasi, dan nilai yang hilang. Langkah-langkahnya melibatkan pemeriksaan terhadap keberadaan nilai yang hilang dalam data. Implementasi ini dapat dilakukan dengan menggunakan fungsi ‘isnull()’ yang digunakan untuk mengidentifikasi apakah terdapat nilai yang hilang dalam setiap kolom data, diikuti dengan fungsi ‘sum()’ untuk menghitung jumlah nilai yang hilang dalam setiap kolom. Implementasi dan hasilnya dapat dilihat dari gambar 4.1.

```
# langkah 3. Preprocessing
# a. Data Cleaning
# Cek apakah ada missing value
missing_values = data.isnull().sum()
print("Jumlah Missing Value per Kolom:")
print(missing_values)
```

```
Jumlah Missing Value per Kolom:
LoanID             0
Age                0
Income             0
LoanAmount         0
CreditScore        0
MonthsEmployed     0
NumCreditLines     0
InterestRate       0
LoanTerm           0
DTIRatio           0
Education          0
EmploymentType     0
MaritalStatus      0
HasMortgage        0
HasDependents      0
LoanPurpose        0
HasCoSigner        0
Default            0
dtype: int64
```

Gambar 4. 1 Implementasi *Data Cleaning* Untuk Mengecek *Missing Value*

Dari gambar 4.1 terlihat hasil menunjukkan bahwa tidak ada nilai yang hilang (missing value) dalam dataset. Setiap kolom memiliki jumlah nilai yang hilang sebanyak 0, yang menunjukkan bahwa data sudah bersih dari nilai yang hilang. Ini adalah hasil yang diinginkan karena data yang bebas dari nilai yang hilang akan memudahkan proses analisis dan pemodelan selanjutnya. Dengan demikian, kita dapat lanjut ke tahap berikutnya tanpa perlu melakukan penanganan khusus terhadap nilai yang hilang.

4.1.2 Mengubah Fitur Kategori Menjadi Fitur Numerik Dengan *Ordered Target Statistic*

Tahap ini bertujuan untuk mengubah fitur kategori menjadi fitur numerik dengan menggunakan *Ordered Target Statistic* dari *CATBOOST*. Ini dilakukan dengan menghitung rata-rata target untuk setiap nilai unik dalam setiap fitur kategori, dan kemudian menggantikan nilai-nilai kategori dengan rata-rata target yang sesuai. Proses ini memungkinkan model untuk memahami dan memproses fitur kategori sebagai bagian dari analisis atau pemodelan.

Hasil dari tahap ini adalah data yang telah mengalami transformasi, di mana fitur kategori telah diubah menjadi numerik dengan menggunakan rata-rata target yang sesuai untuk setiap nilai unik dalam fitur tersebut. Ini meningkatkan informasi yang tersedia untuk model dan dapat meningkatkan kinerja model dalam melakukan prediksi. Penggalan *code* dapat dilihat pada gambar 4.2 di bawah ini.

```
#8. Mengunbah data kategory menjadi numerik menggunakan ordered target statistik
# Hitung rata-rata target untuk setiap nilai unik dalam fitur kategori dan ganti nilai dengan rata-rata target yang sesuai
for column in ['Education', 'EmploymentType', 'MaritalStatus', 'HasMortgage', 'HasDependents', 'LoanPurpose', 'HasCoSigner']:
    target_mean = data.groupby(column)['Default'].mean().reset_index()
    target_mean.rename(columns={'Default': f'{column}_Target_Mean'}, inplace=True)
    data = data.merge(target_mean, on=column, how='left')
    data[column] = data[f'{column}_Target_Mean']
    data.drop(columns=[f'{column}_Target_Mean'], inplace=True)

# Pisahkan fitur dan variabel target setelah transformasi
X = data.drop(columns=['LoanID', 'Default'])
y = data['Default']

# Menampilkan beberapa baris pertama dan beberapa baris terakhir dari data setelah transformasi
print("Data Setelah Transformasi Fitur Kategori menjadi Numerik dengan Ordered Target Statistic:")
print(X.head())

✓ 0.8s
```

Gambar 4. 2 Penggalan *Code Ordered Target Statistic*

```
Data Sebelum Transformasi:
        LoanID  Age  Income  LoanAmount  CreditScore  MonthsEmployed  \
0  I38PQUQS96   56   85994       50587          520              80
1  HPSK72WA7R   69   50432      124440          458              15
2  C1OZ6DPJ8Y   46   84208      129188          451              26
3  V2KKSFM3UN   32   31713       44799          743               0
4  EY08JDHTZP   60   20437        9139          633               8

   NumCreditLines  InterestRate  LoanTerm  DTIRatio     Education  \
0               4         15.23        36      0.44    Bachelor's
1               1          4.81        60      0.68      Master's
2               3         21.17        24      0.31      Master's
3               3          7.07        24      0.23   High School
4               4          6.51        48      0.73    Bachelor's

  EmploymentType MaritalStatus HasMortgage HasDependents LoanPurpose  \
0      Full-time      Divorced         Yes            Yes       Other
1      Full-time       Married          No             No       Other
2     Unemployed      Divorced         Yes            Yes        Auto
3      Full-time       Married          No             No    Business
4     Unemployed      Divorced          No            Yes        Auto

  HasCoSigner  Default
0         Yes        0
1         Yes        0
2          No        1
3          No        0
4          No        0
```

Gambar 4. 3 Data Sebelum di *Ordered Target Statistic*

```
Data Setelah Transformasi Fitur Kategori menjadi Numerik dengan Ordered Target Statistic:
   Age  Income  LoanAmount  CreditScore  MonthsEmployed  NumCreditLines  \
0   56   85994       50587          520              80               4
1   69   50432      124440          458              15               1
2   46   84208      129188          451              26               3
3   32   31713       44799          743               0               3
4   60   20437        9139          633               8               4

   InterestRate  LoanTerm  DTIRatio  Education  EmploymentType  MaritalStatus  \
0         15.23        36      0.44   0.121011        0.094634       0.125328
1          4.81        60      0.68   0.108717        0.094634       0.103972
2         21.17        24      0.31   0.108717        0.135529       0.125328
3          7.07        24      0.23   0.128789        0.094634       0.103972
4          6.51        48      0.73   0.121011        0.135529       0.125328

   HasMortgage  HasDependents  LoanPurpose  HasCoSigner
0     0.108806       0.105024     0.117885     0.103601
1     0.123451       0.127244     0.117885     0.103601
2     0.108806       0.105024     0.118814     0.128661
3     0.123451       0.127244     0.123260     0.128661
4     0.123451       0.105024     0.118814     0.128661
```

Gambar 4. 4 Data Sesudah di *Ordered Target Statistic*

4.1.3 Normalisasi *Min-Max*

Pada tahap ini akan melakukan normalisasi data menggunakan *Min-Max* normalisasi yang di mana data yang sebelumnya memiliki nilai yang besar diubah menjadi nilai-nilai dari fitur ke dalam rentang tertentu, biasanya dari 0 hingga 1. Hal ini dilakukan dengan mengurangi nilai

minimum dari setiap fitur dan kemudian membagi hasilnya dengan rentang nilai fitur. Penggalan *Code* dapat dilihat pada gambar 4.5 di bawah ini.

```
# Langkah 6: Normalisasi data menggunakan MinMaxScaler
scaler_minmax = MinMaxScaler()
X_normalized = scaler_minmax.fit_transform(X)

# Menampilkan beberapa baris pertama dari data setelah normalisasi menggunakan MinMaxScaler
print("Data Setelah Normalisasi dengan MinMaxScaler:")
print(X_normalized[:5])
```

✓ 0.0s

Gambar 4. 5 Penggalan *Code* Normalisasi *Min-Max*

```
Data Sebelum Normalisasi:
   Age  Income  LoanAmount  CreditScore  MonthsEmployed  NumCreditLines  \
0   56   85994       50587          520              80               4
1   69   50432      124440          458              15               1
2   46   84208      129188          451              26               3
3   32   31713       44799          743               0               3
4   60   20437        9139          633               8               4

   InterestRate  LoanTerm  DTIRatio  Education  EmploymentType  MaritalStatus  \
0         15.23        36      0.44   0.121011        0.094634       0.125328
1          4.81        60      0.68   0.108717        0.094634       0.103972
2         21.17        24      0.31   0.108717        0.135529       0.125328
3          7.07        24      0.23   0.128789        0.094634       0.103972
4          6.51        48      0.73   0.121011        0.135529       0.125328

   HasMortgage  HasDependents  LoanPurpose  HasCoSigner
0     0.108806       0.105024     0.117885     0.103601
1     0.123451       0.127244     0.117885     0.103601
2     0.108806       0.105024     0.118814     0.128661
3     0.123451       0.127244     0.123260     0.128661
4     0.123451       0.105024     0.118814     0.128661
```

Gambar 4. 6 Data Sebelum di Normalisasi

```
Data Setelah Normalisasi dengan MinMaxScaler:
[[0.74509804 0.52588538 0.18607015 0.4007286  0.67226891 1.
  0.57521739 0.5        0.425      0.66079078 0.         1.
  0.         0.         0.74297224 0.        ]
 [1.         0.2624612  0.48751219 0.28779599 0.12605042 0.
  0.12217391 1.         0.725      0.12462744 0.         0.
  1.         1.         0.74297224 0.        ]
 [0.54901961 0.51265565 0.50689186 0.27504554 0.21848739 0.66666667
  0.83347826 0.25       0.2625     0.12462744 1.         1.
  0.         0.         0.78741215 1.        ]
 [0.2745098  0.12380092 0.16244556 0.80692168 0.         0.66666667
  0.22043478 0.25       0.1625     1.         0.         0.
  1.         1.         1.         1.        ]
 [0.82352941 0.04027437 0.01689395 0.60655738 0.06722689 1.
  0.19608696 0.75       0.7875     0.66079078 1.         1.
  1.         0.         0.78741215 1.        ]]
```

Gambar 4. 7 Data Sesudah di Normalisasi

4.1.4 Normalisasi *Z-Score*

Pada tahap ini akan melakukan tahap normalisasi menggunakan normalisasi *Z-Score* yang mana teknik *preprocessing* yang digunakan untuk mengubah nilai-nilai dalam suatu dataset sehingga memiliki mean (rata-rata) nol dan standar deviasi satu. Proses ini membuat distribusi data menjadi berpusat di sekitar nol dengan sebaran yang seragam di sepanjang sumbu. Berikut adalah Penggalan *code* untuk normalisasi *Z-Score*:

```python
# Impor modul StandardScaler
from sklearn.preprocessing import StandardScaler
# Langkah 6: Normalisasi data menggunakan StandardScaler (Z-score)
scaler_zscore = StandardScaler()
X_normalized_zscore = scaler_zscore.fit_transform(X)

# Menampilkan beberapa baris pertama dari data setelah normalisasi menggunakan StandardScaler
print("Data Setelah Normalisasi dengan StandardScaler (Z-score):")
print(X_normalized_zscore[:5])
```

✓ 0.1s

Gambar 4. 8 Penggalan *Code* Normalisasi *Z-Score*

```
Data Sebelum Normalisasi:
   Age  Income  LoanAmount  CreditScore  MonthsEmployed  NumCreditLines  \
0   56   85994       50587          520              80               4
1   69   50432      124440          458              15               1
2   46   84208      129188          451              26               3
3   32   31713       44799          743               0               3
4   60   20437        9139          633               8               4

   InterestRate  LoanTerm  DTIRatio  Education  EmploymentType  MaritalStatus  \
0         15.23        36      0.44   0.121011        0.094634       0.125328
1          4.81        60      0.68   0.108717        0.094634       0.103972
2         21.17        24      0.31   0.108717        0.135529       0.125328
3          7.07        24      0.23   0.128789        0.094634       0.103972
4          6.51        48      0.73   0.121011        0.135529       0.125328

   HasMortgage  HasDependents  LoanPurpose  HasCoSigner
0     0.108806       0.105024     0.117885     0.103601
1     0.123451       0.127244     0.117885     0.103601
2     0.108806       0.105024     0.118814     0.128661
3     0.123451       0.127244     0.123260     0.128661
4     0.123451       0.105024     0.118814     0.128661
```

Gambar 4. 9 Data Sebelum di Normalisasi

```
Data Setelah Normalisasi dengan StandardScaler (Z-score):
[[ 8.33989509e-01  8.96928115e-02 -1.08683299e+00 -3.41492335e-01
   5.90533211e-01  1.34193677e+00  2.61771239e-01 -1.52594275e-03
  -2.60752922e-01  5.26439537e-01 -1.47298267e+00  1.02509086e+00
  -9.99972587e-01 -9.99463619e-01  2.44981760e-01 -9.99784630e-01]
 [ 1.70122109e+00 -8.23020714e-01 -4.43088703e-02 -7.31666109e-01
  -1.28573104e+00 -1.34379144e+00 -1.30835003e+00  1.41279312e+00
   7.78585323e-01 -7.99015102e-01 -1.47298267e+00 -1.35457434e+00
   1.00002741e+00  1.00053667e+00  2.44981760e-01 -9.99784630e-01]
 [ 1.66888295e-01  4.38543784e-02  2.27148729e-02 -7.75717987e-01
  -9.68209399e-01  4.46694036e-01  1.15683077e+00 -7.08685475e-01
  -8.23727805e-01 -7.99015102e-01  1.32949264e+00  1.02509086e+00
  -9.99972587e-01 -9.99463619e-01  3.74576838e-01  1.00021542e+00]
 [-7.67053404e-01 -1.30345164e+00 -1.16853758e+00  1.06187463e+00
  -1.71871510e+00  4.46694036e-01 -9.67805494e-01 -7.08685475e-01
  -1.17017389e+00  1.36500195e+00 -1.47298267e+00 -1.35457434e+00
   1.00002741e+00  1.00053667e+00  9.94522764e-01  1.00021542e+00]
 [ 1.10082999e+00 -1.59285487e+00 -1.67192147e+00  3.69630835e-01
  -1.48779027e+00  1.34193677e+00 -1.05218821e+00  7.05633590e-01
   9.95114124e-01  5.26439537e-01  1.32949264e+00  1.02509086e+00
   1.00002741e+00 -9.99463619e-01  3.74576838e-01  1.00021542e+00]]
```

Gambar 4. 10 Data Sesudah di Normalisasi

4.2 Pembagian *Data Training* dan *Data Testing*

Tahap selanjutnya merupakan pembagian data sebelumnya data dibagi terlebih dahulu menggunakan *train_test_split* dengan skala 80:20, dimana 80% data *training* dan 20% data *testing*. Penggalan *Code* dapat dilihat pada gambar 4.11 di bawah ini.

```
from sklearn.model_selection import train_test_split, StratifiedKFold
# Langkah 7: Split dataset menjadi data latih dan data uji (80/20)
X_train, X_test, y_train, y_test = train_test_split(X_normalized, y, test_size=0.2, random_state=42)
✓ 0.1s
```

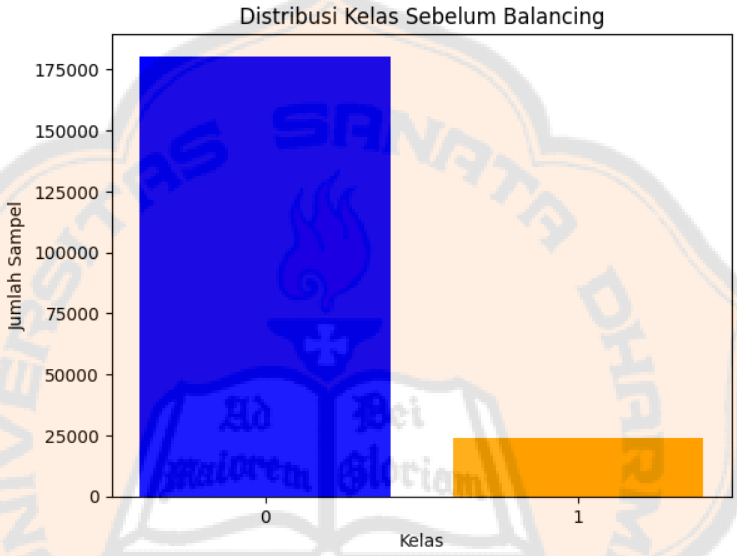
Gambar 4. 11 Penggalan *Code* Split Data dengan *train_test_split*

4.3 *Data Balancing*

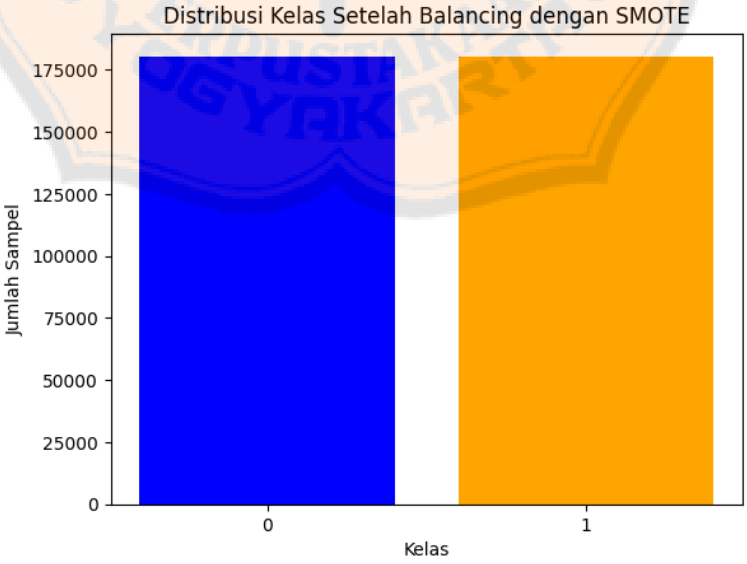
Tahap selanjutnya adalah *data balancing* yang sebelumnya sudah di split 80:20 disini akan di *balancing* menggunakan *SMOTE*, kemudian data 80% yang telah di *balancing* akan dibagi dengan *K-fold* dengan nilai split yaitu 3, 5, 7, dan 10. Penggalan Code dapat dilihat pada gambar 4.12 dibawah ini.

```python
#Langkah 8 Balancing data setelah di split
from imblearn.over_sampling import SMOTE

# Langkah 3: Balancing data menggunakan SMOTE hanya pada data latih (80%)
smote = SMOTE(random_state=42)
X_train_resampled, y_train_resampled = smote.fit_resample(X_train, y_train)

# Hitung jumlah sampel untuk setiap kelas setelah balancing
counts_resampled = y_train_resampled.value_counts()
```

Gambar 4. 12 Penggalan *Code* Data *Balancing*



Gambar 4. 13 Data Sebelum Seimbang



Gambar 4. 14 Data Sesudah Seimbang

```
# Inisialisasi KFold dengan nilai split 3,5,7, dan 10
kf = KFold(n_splits=3, shuffle=True, random_state=42)
# Gunakan kf untuk pembagian data
for train_index, test_index in kf.split(X_train_resampled, y_train_resampled):
    X_train_fold, X_val_fold = X_train_resampled[train_index], X_train_resampled[test_index]
    y_train_fold, y_val_fold = y_train_resampled.iloc[train_index], y_train_resampled.iloc[test_index]
```

Gambar 4. 15 Penggalan Code membagi data dengan *K-fold*

4.4 Pelatihan Model

Setelah tahap balancing data, kemudian masuk ke tahap selanjutnya yaitu menggunakan *CatboostClassifier* untuk pemodelan klasifikasi. Parameter yang digunakan pada algoritma *CATBOOST* mencakup *iterations* untuk menentukan jumlah iterasi atau jumlah pohon yang akan dibangun. Ada parameter *learning_rate* yang digunakan untuk mengontrol kecepatan pembelajaran. Kemudian, ada parameter *depth* yang digunakan untuk menentukan kedalaman maksimum pohon. Selanjutnya, parameter *loss_function='Logloss'* digunakan untuk menentukan fungsi kerugian yang akan dioptimalkan selama pelatihan. Lalu, ada parameter *random_seed* yang digunakan untuk memastikan hasil yang konsisten setiap kali program dijalankan. Model dilatih dengan memanggil metode fit() pada objek *CatboostClassifier* dengan data latih. Setelah dilatih, model dievaluasi menggunakan metrik seperti akurasi, presisi, recall, dan F1-score. Penggalan *code* dapat dilihat pada gambar 4.16 berikut.

```python
# Inisialisasi dan pelatihan model CatBoostClassifier
start_time = time.time()
model = CatBoostClassifier(iterations=1000, learning_rate=0.1, depth=6, loss_function='Logloss', random_seed=42)
model.fit(X_train_fold, y_train_fold, eval_set=(X_val_fold, y_val_fold), early_stopping_rounds=50, verbose=100)
training_time = time.time() - start_time
print("Training time:", training_time)

# Prediksi menggunakan model yang telah dilatih
y_pred = model.predict(X_val_fold)

# Hitung confusion matrix
conf_matrix = confusion_matrix(y_val_fold, y_pred)
print("Confusion Matrix:")
print(conf_matrix)

# Hitung metrik evaluasi
accuracy = accuracy_score(y_val_fold, y_pred)
precision = precision_score(y_val_fold, y_pred)
recall = recall_score(y_val_fold, y_pred)
f1 = f1_score(y_val_fold, y_pred)

print("Accuracy:", accuracy)
print("Precision:", precision)
print("Recall:", recall)
print("F1-score:", f1)
```

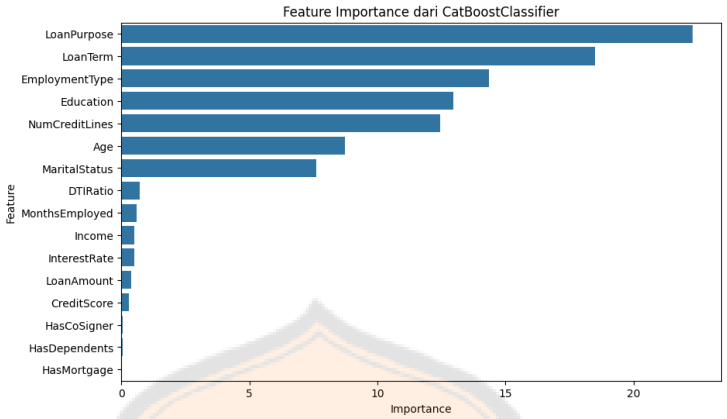
Gambar 4. 16 Penggalan *Code* Pemodelan *CATBOOST*

4.4.1 *Feature Importances*

Pada tahap ini, data akan dianalisis untuk menentukan kepentingan masing-masing fitur dalam model prediksi. Dengan menggunakan algoritma *CatboostClassifier*, akan menghitung dan memvisualisasikan pentingnya setiap fitur yang ada dalam dataset. *Feature importances* memberikan gambaran tentang seberapa besar pengaruh suatu fitur terhadap prediksi model. Fitur dengan nilai *importances* yang lebih tinggi dianggap lebih signifikan dalam menentukan hasil prediksi. Hasil visualisasi *feature importances* ditunjukkan pada gambar 4.17 berikut ini:

```python
# Feature importance
feature_importances = model.get_feature_importance()
feature_names = data.drop(columns=['LoanID', 'Default']).columns

# Plot feature importance
feature_importance_df = pd.DataFrame({'Feature': feature_names, 'Importance': feature_importances})
feature_importance_df = feature_importance_df.sort_values(by='Importance', ascending=False)

plt.figure(figsize=(10, 6))
sns.barplot(x='Importance', y='Feature', data=feature_importance_df)
plt.title('Feature Importance dari CatBoostClassifier')
plt.show()
```

Gambar 4. 17 Penggalan *Code Feature Importances*



Gambar 4. 18 *Feature Importances* dari *CatBoostClassifier*

Pada gambar 4.18, dapat dilihat bahwa fitur-fitur seperti LoanPurpose, LoanTerm, EmploymentType, Education, NumCreditLines, Age dan MaritalStatus memiliki nilai *importances* yang tinggi, menunjukkan bahwa mereka adalah faktor-faktor utama yang mempengaruhi prediksi model. Sebaliknya, fitur-fitur seperti HasMortgage, HasDependents dan HasCoSigner memiliki nilai *importances* yang rendah, sehingga bisa dipertimbangkan untuk dihapus dalam proses seleksi fitur.

4.4.2 Memilih Fitur Dengan *Importances* Di Atas *Threshold*

Pada tahap ini, dilakukan pemilihan fitur-fitur yang memiliki nilai pentingnya (*importances*) di atas ambang batas (*threshold*) tertentu. Ini bertujuan untuk menyaring hanya fitur-fitur yang memberikan kontribusi signifikan terhadap prediksi model, sehingga dapat meningkatkan performa model dan mengurangi kompleksitasnya. Berikut merupakan penggalan *code* menghitung *Importances* Di Atas *Threshold*, dapat dilihat pada gambar 4.19 dibawah ini.

```
# Pilih fitur dengan importance di atas threshold tertentu
threshold = 1.0
important_features = feature_importance_df[feature_importance_df['Importance'] > threshold]['Feature'].tolist()

# Gunakan fitur penting untuk model
X_train_important = pd.DataFrame(X_train_resampled, columns=feature_names)[important_features]
X_test_important = pd.DataFrame(X_test, columns=feature_names)[important_features]
print("Fitur yang dipakai berdasarkan importance di atas threshold:")
print(important_features)
✓ 0.0s

Fitur yang dipakai berdasarkan importance di atas threshold:
['LoanPurpose', 'LoanTerm', 'EmploymentType', 'NumCreditLines', 'Education', 'Age', 'MaritalStatus']
```

Gambar 4. 19 Penggalan *Code* Memilih Fitur Terbaik

Hasil yang ditampilkan pada gambar 4.19 menunjukkan daftar fitur yang dipilih berdasarkan nilai pentingnya (*importances*) yang melebihi ambang batas (*threshold*) tertentu, yaitu LoanPurpose, LoanTerm, EmploymentType, NumCreditLines, Education, Age, dan MaritalStatus. Fitur-fitur ini dianggap memberikan kontribusi signifikan terhadap prediksi model karena memiliki nilai penting yang lebih tinggi dari threshold yang ditetapkan. Dengan menggunakan fitur-fitur ini, diharapkan model dapat membuat prediksi yang lebih akurat dan efisien, mengingat fitur-fitur tersebut mencakup aspek-aspek penting seperti tujuan peminjaman, durasi pinjaman, jenis pekerjaan, jumlah jalur kredit, tingkat pendidikan, usia, dan status pernikahan peminjam.

4.5 Hasil Pengujian Klasifikasi Menggunakan Dataset Kredit Bank

Pada tahap ini, data kredit bank akan diuji menggunakan pembagian data *k-fold* yaitu 3,5,7, dan 10 dengan menggunakan data kredit bank setelah di normalisasi menggunakan *Min-Max* dan *Z-Score* dan telah melalui tahapan *feature*

importances. Hasil akurasi dari pengujian *k-fold* pada data kredit bank dapat dilihat pada tabel berikut.

Tabel 4. 1 Tabel Perbandingan Hasil Pengujian

K-fold	**Normalisasi *Min-Max* pada *CATBOOST***		**Normalisasi *Z-Score* pada *CATBOOST***	
	Akurasi	**Waktu Eksekusi**	**Akurasi**	**Waktu Eksekusi**
3	93.217%	36.33	93.233%	35.95
5	93.316%	45.47	93.307%	41.31
7	93.320%	45.12	93.245%	37.18
10	93.219%	48.76	93.161%	30.57

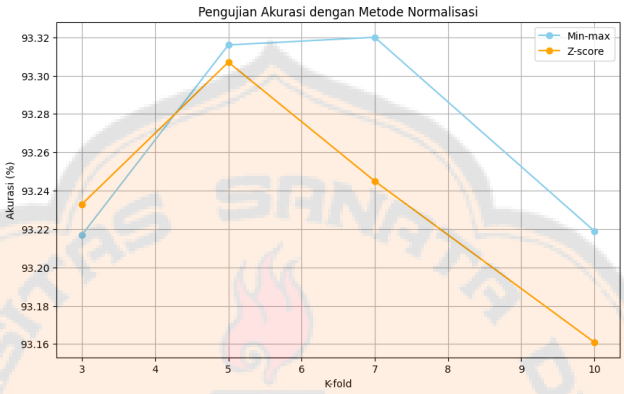
Pada tabel 4.1 menunjukkan hasil evaluasi menggunakan *K-fold cross-validation* dengan metode normalisasi *Min-Max* dan *Z-Score* pada *CatBoostClassifier* menunjukkan akurasi yang sangat mirip. Dengan normalisasi *Min-Max*, akurasi berkisar antara 93.217% hingga 93.320%, sedangkan dengan *Z-Score* berkisar antara 93.161% hingga 93.307%, menunjukkan perbedaan yang tidak signifikan. Namun, waktu eksekusi untuk normalisasi *Z-Score* lebih rendah, berkisar antara 30.57 hingga 41.31 detik dibandingkan dengan *Min-Max* yang berkisar antara 36.33 hingga 48.76 detik. Hal ini mengindikasikan bahwa meskipun kedua metode normalisasi menghasilkan akurasi yang hampir sama, normalisasi *Z-Score* lebih efisien dari segi waktu eksekusi, yang dapat menjadi pertimbangan penting dalam pemilihan metode normalisasi.

Tabel 4. 2 Tabel Hasil Akurasi, Presisi, *Recal*, dan *F1* Score

K-fold	**Pengujian**	**Akurasi**	**Presisi**	***Recall***	***F1 Score***	***Training Time* (detik)**
3	***CATBOOST Min-Max***	93.217%	99.021%	87.297%	92.790%	36.33
	CATBOOST Z-Score	93.233%	99.025%	87.326%	92.809%	35.95
5	***CATBOOST Min-Max***	93.316%	99.074%	87.450%	92.900%	45.47
	CATBOOST Z-Score	93.307%	99.135%	87.375%	92.885%	41.31
7	***CATBOOST Min-Max***	93.320%	99.120%	87.417%	92.901%	45.12
	CATBOOST Z-Score	93.245%	99.088%	87.293%	92.817%	37.18
10	***CATBOOST Min-Max***	93.219%	99.007%	87.314%	92.794%	48.76
	CATBOOST Z-Score	93.161%	99.067%	87.142%	92.723%	30.57

Pada tabel 4.2 menunjukkan hasil evaluasi menggunakan *K-fold cross-validation* untuk model *CatBoostClassifier* dengan normalisasi *Min-Max* dan *Z-Score* menunjukkan performa yang serupa dalam hal akurasi, presisi, recall, dan F1 score. Dengan normalisasi *Min-Max*, akurasi berkisar antara 93.217% hingga 93.320%, presisi antara 99.007% hingga 99.120%, recall antara 87.297% hingga 87.450%, dan F1 score antara 92.790% hingga 92.901%. Waktu pelatihan berkisar antara 36.33 hingga 48.76 detik. Sementara itu, dengan normalisasi *Z-Score*, akurasi berkisar antara 93.161% hingga 93.245%, presisi antara 99.025% hingga 99.135%, recall antara 87.142% hingga 87.375%, dan F1 score antara 92.723%

hingga 92.885%. Waktu pelatihan berkisar antara 30.57 hingga 41.31 detik. Perbedaan performa antara kedua metode normalisasi ini sangat kecil, namun normalisasi *Z-Score* menunjukkan waktu pelatihan yang lebih efisien dibandingkan dengan *Min-Max*.

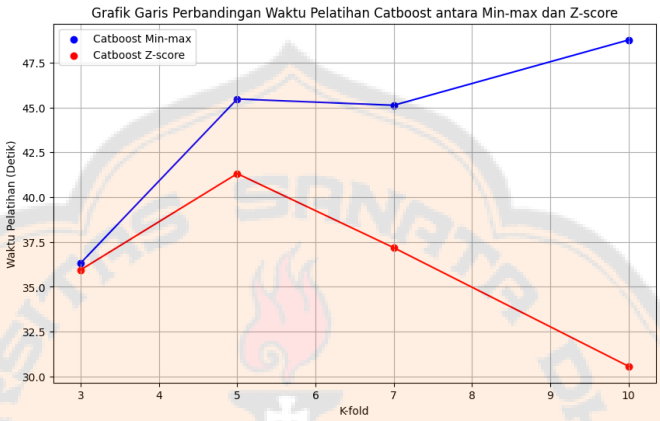


Gambar 4. 20 Plot Garis Hasil Pengujian Akurasi

Grafik pada gambar 4.20 menunjukkan perbandingan akurasi algoritma *CATBOOST* dengan dua metode normalisasi, yaitu *Min-Max* dan *Z-Score*, pada berbagai nilai *k-fold*. Sumbu horizontal (x) mewakili nilai *k-fold*, sedangkan sumbu vertikal (y) menunjukkan akurasi dalam persentase. Metode *Min-Max* (garis biru) menunjukkan variasi akurasi antara 93,217% hingga 93,32%, dengan akurasi tertinggi pada *k-fold* 7 dan terendah pada *k-fold* 3. Metode *Z-Score* (garis oranye) menunjukkan akurasi yang sedikit lebih konsisten namun sedikit lebih rendah, dengan akurasi tertinggi pada *k-fold* 5 (93,307%) dan terendah pada *k-fold* 10 (93,161%).

Secara keseluruhan, grafik ini mengindikasikan bahwa algoritma *CATBOOST* mampu memberikan akurasi tinggi dan stabil dengan kedua metode normalisasi. Meskipun terdapat variasi kecil dalam akurasi di berbagai *k-fold*,

metode *Min-Max* cenderung memberikan akurasi sedikit lebih tinggi pada beberapa *k-fold* dibandingkan dengan *Z-Score*. Namun, perbedaan ini tidak signifikan, sehingga kedua metode normalisasi dapat dianggap efektif dalam menghasilkan model yang akurat. Berikut merupakan plot garis waktu pelatihan.



Gambar 4. 21 Plot Garis Waktu Pelatihan

Grafik pada gambar 4.21 menunjukkan perbandingan waktu pelatihan (dalam detik) untuk algoritma *CATBOOST* dengan dua metode normalisasi, yaitu *Min-Max* dan *Z-Score*, pada berbagai nilai *k-fold*. Sumbu horizontal (x) mewakili nilai *k-fold*, sedangkan sumbu vertikal (y) menunjukkan waktu pelatihan dalam detik. Garis biru mewakili waktu pelatihan dengan normalisasi *Min-Max*, sementara garis merah mewakili waktu pelatihan dengan normalisasi *Z-Score*. Dari grafik ini, terlihat bahwa waktu pelatihan cenderung lebih lama untuk normalisasi *Min-Max* dibandingkan dengan *Z-Score*, dengan rentang waktu pelatihan *Min-Max* antara 36,33 detik hingga 48,76 detik, dan *Z-Score* antara 30,57 detik hingga 41,31 detik. Secara keseluruhan, normalisasi *Z-Score* menunjukkan waktu pelatihan yang lebih cepat dibandingkan *Min-Max*, terutama pada *k-fold* yang lebih tinggi.

4.6 Evaluasi Model Pada Data Uji

Pada tahap ini dilakukan pengujian terhadap data uji 20% yang telah melewati proses *preprocessing* dan *feature importances* yang di mana terdapat 7 fitur yang memiliki nilai *importances* tertinggi yang dipakai dapat dilihat pada gambar 4.18. Berikut merupakan penggalan *code* dari evaluasi model menggunakan data uji dapat dilihat pada gambar 4.22. Hasil dari pengujian pada data uji mendapatkan persentase akurasi sebesar 88,46%, yang bisa dilihat pada gambar 4.23 di bawah ini.

```
# Prediksi menggunakan model yang telah dilatih pada data uji 20%
y_pred_test = model.predict(X_test_important)

# Hitung metrik evaluasi pada data uji
accuracy_test = accuracy_score(y_test, y_pred_test)

# Tampilkan metrik evaluasi pada data uji
print("Evaluation Metrics on Test Data:")
print("Accuracy:", accuracy_test)

# Tampilkan confusion matrix pada data uji
conf_matrix_test = confusion_matrix(y_test, y_pred_test)
print("Confusion Matrix on Test Data:")
print(conf_matrix_test)

# Tampilkan classification report pada data uji
class_report_test = classification_report(y_test, y_pred_test)
print("\nClassification Report:\n", class_report_test)
```

✓ 0.0s

Gambar 4. 22 Penggalan *Code* Evaluasi Model

```
Evaluation Metrics on Test Data:
Accuracy: 0.8844135500293715
Confusion Matrix on Test Data:
[[45167     3]
 [ 5900     0]]

Classification Report:
              precision    recall  f1-score   support

           0       0.88      1.00      0.94     45170
           1       0.00      0.00      0.00      5900

    accuracy                           0.88     51070
   macro avg       0.44      0.50      0.47     51070
weighted avg       0.78      0.88      0.83     51070
```

Gambar 4. 23 Hasil Akurasi Terhadap Data Uji

4.7 Evaluasi Hasil Pengujian Akurasi Data Kredit Bank

Dari hasil pengujian tersebut, terlihat bahwa algoritma *CATBOOST* memberikan akurasi yang sangat mirip antara penggunaan normalisasi

menggunakan metode *Min-Max* dan *Z-Score* pada berbagai pembagian data *k-fold*. Akurasi tertinggi terjadi pada *k-fold* 7 menggunakan metode normalisasi *Min-Max*, dengan nilai sekitar 93,32%.

Selain akurasi, hasil pengujian juga menunjukkan nilai *presisi*, *recall*, dan *F1 score* yang tinggi untuk kedua metode normalisasi. Nilai *presisi* untuk metode *Min-Max* dan *Z-Score* berkisar antara 99% hingga 99,13%, yang menunjukkan bahwa algoritma ini sangat efektif dalam mengidentifikasi kasus positif dengan benar. *Recall*, yang mengukur kemampuan model dalam menemukan semua kasus positif, juga berada di kisaran yang tinggi antara 87,14% dan 87,45%. *F1 score*, yang merupakan harmoni dari *presisi* dan *recall*, menunjukkan performa keseluruhan yang baik dengan nilai antara 92,72% dan 92,90%.

Waktu pelatihan (*training time*) juga dicatat untuk setiap *k-fold* dan metode normalisasi. Waktu pelatihan cenderung bervariasi tergantung pada jumlah data dan kompleksitas model. Secara umum, waktu pelatihan cenderung lebih cepat pada *k-fold* yang lebih rendah dan metode normalisasi menggunakan *Z-Score*. Misalnya, pada *k-fold* 10, waktu pelatihan untuk *Z-Score* adalah 30.57 detik, lebih cepat dibandingkan dengan *Min-Max* yang membutuhkan 48.76 detik.

Model pengujian terhadap data uji 20% yang telah melewati proses *preprocessing* dan *feature importancess* yang di mana terdapat 7 fitur yang memiliki nilai *importances* tertinggi yang. Hasil dari pengujian pada data uji mendapatkan persentase akurasi yang baik yaitu sebesar 88,46%.

BAB V

PENUTUP

Pada bab ini akan berisi mengenai kesimpulan dan saran setelah melakukan penelitian mengenai prediksi kegagalan pembayaran di sektor perbankan menggunakan algoritma *CATBOOST* dengan membandingkan normalisasi *Min-Max* dan *Z-Score*.

5.1 Kesimpulan

Berdasarkan hasil pengujian pada data kredit bank yang telah dilakukan menggunakan pembagian data *k-fold* dan normalisasi menggunakan metode *Min-Max* serta *Z-Score*, berikut adalah kesimpulan yang dapat diambil:

1. Algoritma *CATBOOST* menunjukkan performa yang stabil dengan akurasi sekitar 93% pada berbagai pengujian dengan pembagian data *k-fold* yang berbeda dan metode normalisasi yang berbeda pula. Hal ini mengindikasikan bahwa *CATBOOST* cukup handal dalam memberikan prediksi yang akurat untuk data kredit bank setelah normalisasi.
2. Kedua metode normalisasi memberikan hasil performa yang optimal dan serupa, namun tidak terdapat perbedaan yang signifikan dalam akurasi antara penggunaan metode normalisasi *Min-Max* dan *Z-Score* pada *CATBOOST* yang menunjukkan bahwa kedua metode tersebut secara signifikan mempengaruhi hasil model *CATBOOST*.
3. *CATBOOST* menunjukkan tingkat presisi yang sangat baik di atas 99% untuk kedua metode normalisasi, yang berarti model ini sangat efektif dalam

mengidentifikasi kasus-kasus gagal bayar yang benar. Selain itu nilai *recall* juga konsisten di sekitar 87%, menunjukkan bahwa model mampu mendeteksi sebagian besar kasus gagal bayar yang benar, meskipun ada beberapa yang terlewat. *F1 Score* berada di sekitar 92%, mencerminkan keseimbangan yang baik antara *presisi* dan *recall*, memastikan bahwa model tidak hanya akurat tetapi juga cukup sensitif dalam mendeteksi gagal bayar.

5.2 Saran

Berdasarkan hasil penelitian tentang klasifikasi kegagalan pembayaran di sektor perbankan dengan menggunakan normalisasi *Min-Max* dan *Z-Score* serta algoritma *CATBOOST*, beberapa saran yang dapat diberikan adalah sebagai berikut:

1. Penggunaan metode lain selain *CATBOOST*, mencoba metode klasifikasi lainnya seperti *Random Forest*, *Support Vector Machine* (SVM), atau *Neural Networks* (NN) untuk membandingkan performa dengan *CATBOOST* dan memperluas pemahaman tentang algoritma mana yang paling efektif dalam konteks ini.

DAFTAR PUSTAKA

Andini, G. (2017). *faktor faktor yang menentukan keputusan pemberian pinjaman*.https://repository.uinjkt.ac.id/dspace/bitstream/123456789/40755/1/GITA%20ANDINI%20-%20FEB.pdf

Bambang Catur P. (2014). *Pengamanan Pemberian Kredit Bank Dengan Jaminan Hak Guna Bangunan*.

Dewi, N. K., Sains, F., & Teknologi, D. (2021). *DETEKSI FAKE FOLLOWER INSTAGRAM MENGGUNAKAN CATBOOST CLASSIFER SKRIPSI PROGRAM STUDI MATEMATIKA*. https://repository.uinjkt.ac.id/dspace/bitstream/123456789/56737/1/NIA%20KARUNIA%20DEWI-FST.pdf

Kasanah, A. N., Muladi, M., & Pujianto, U. (2019). Penerapan Teknik SMOTE untuk Mengatasi Imbalance Class dalam Klasifikasi Objektivitas Berita Online Menggunakan Algoritma KNN. *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, *3*(2), 196–201.

Kurniawan, D., & Supriyanto, C. (2013). Optimasi Algoritma Support Vector Machine (Svm) Menggunakan Adaboost Untuk Penilaian Risiko Kredit. *Jurnal Teknologi Informasi*, *9*(1), 1414–9999.

Nasution, F. A., Saadah, S., & Yunanto, P. E. (2023). Credit Risk Detection in Peer-to-Peer Lending Using *CATBOOST*. *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, *7*(5), 1056–1062. https://doi.org/10.29207/resti.v7i5.5139

Ninditama, I. P. (2021). Model Machine Learning untuk Klasifikasi Keluarga Sejahtera Study Kasus: Kecamatan Kota Palembang. *Jurnal Tekno Kompak*, *15*(2), 37–49.

Nugraha, W., & Syarif, M. (2023). Teknik Weighting untuk Mengatasi Ketidakseimbangan Kelas Pada Prediksi Churn Menggunakan XGBoost, LightGBM, dan *CATBOOST*. *Techno. Com*, *22*(1), 97–108.

Permana, I., & Salisah, F. N. S. (2022). Pengaruh Normalisasi Data Terhadap Performa Hasil Klasifikasi Algoritma Backpropagation: The Effect of Data Normalization on the Performance of the Classification Results of the Backpropagation Algorithm. *Indonesian Journal of Informatic Research and Software Engineering (IJIRSE)*, *2*(1), 67–72.

Pradnyana, G. A., & Agustini, K. (2018). Konsep Dasar Data Mining. vol, 1, 1-47.

Purnamandari, N., & Badera, I. D. N. (2015). Kemampuan prediksi rasio keuangan dan ukuran bank pada risiko gagal bank. *E-Jurnal Akuntansi Universitas Udayana*, *12*(2), 172–187.

Subroto, A., & Arianto, A. (2011). Penggunaan Kartu Kredit dan Perilaku Belanja Kompulsif: Dampaknya pada Risiko Gagal Bayar. *Jurnal Manajemen Pemasaran*, *6*(1), 1–7.

Suryanegara, G. A. B., & Purbolaksono, M. D. (2021a). Peningkatan Hasil Klasifikasi pada Algoritma Random Forest untuk Deteksi Pasien Penderita Diabetes Menggunakan Metode Normalisasi. *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, *5*(1), 114–122.

Suryanegara, G. A. B., & Purbolaksono, M. D. (2021b). Peningkatan Hasil Klasifikasi pada Algoritma Random Forest untuk Deteksi Pasien Penderita Diabetes Menggunakan Metode Normalisasi. *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, *5*(1), 114–122.

Team, A. G., & Yandex. (2017). *CATBOOST Document*. Team & Yandex. https://*CATBOOST*.ai/en/docs/

Townsend, J. T. (1971). Theoretical analysis of an alphabetic confusion matrix. *Perception & Psychophysics*, *9*, 40–50.

Wicaksono, D. F., Basuki, R. S., & Setiawan, D. (2024). Peningkatan Performa Model Machine Learning XGBoost Classifier melalui Teknik Oversampling dalam Prediksi Penyakit AIDS. *JURNAL MEDIA INFORMATIKA BUDIDARMA*, *8*(2), 736–747.

Zai, C. (2022). Implementasi Data Mining Sebagai Pengolahan Data. *Jurnal Portal Data*, *2*(3).

Zhu, Q., Ding, W., Xiang, M., Hu, M., & Zhang, N. (2023). Loan Default Prediction Based on Convolutional Neural Network and LightGBM. *International Journal of Data Warehousing and Mining (IJDWM)*, *19*(1), 1–16.