TESIS

KOMPARASI TINGKAT AKURASI ALGORITMA RANDOM FOREST DAN K-NEAREST NEIGHBOR (KNN) UNTUK MENGKLASIFIKASI KELAYAKAN PENERIMAAN KREDIT BANK

Disusun oleh:

Nama : **Bayu Aji Santoso**

NIM : **22.55.1212**

Konsentrasi : **Business Intelligence**

PROGRAM STUDI S2 TEKNIK INFORMATIKA

PROGRAM PASCASARJANA UNIVERSITAS AMIKOM YOGYAKARTA

YOGYAKARTA

2024

TESIS

KOMPARASI TINGKAT AKURASI ALGORITMA RANDOM FOREST DAN K-NEAREST NEIGHBOR (KNN) UNTUK MENGKLASIFIKASI KELAYAKAN PENERIMAAN KREDIT BANK

COMPARISON OF ACCURACY LEVEL OF RANDOM FOREST AND K-NEAREST NEIGHBOR (KNN) ALGORITHM TO CLASSIFY ELIGIBILITY FOR ACCEPTING BANK CREDIT

Diajukan untuk memenuhi salah satu syarat memperoleh derajat Magister

Disusun oleh:

Nama : **Bayu Aji Santoso**

NIM : **22.55.1212**

Konsentrasi : **Business Intelligence**

PROGRAM STUDI S2 TEKNIK INFORMATIKA

PROGRAM PASCASARJANA UNIVERSITAS AMIKOM YOGYAKARTA

YOGYAKARTA

2024

HALAMAN PENGESAHAN

KOMPARASI TINGKAT AKURASI ALGORITMA RANDOM FOREST DAN K-NEAREST NEIGHBOR (KNN) UNTUK MENGKLASIFIKASI KELAYAKAN PENERIMAAN KREDIT BANK

COMPARISON OF ACCURACY LEVEL OF RANDOM FOREST AND K-NEAREST NEIGHBOR (KNN) ALGORITHM TO CLASSIFY ELIGIBILITY FOR ACCEPTING BANK CREDIT

Dipersiapkan dan Disusun oleh

Bayu Aji Santoso

22.55.1212

Telah Diujikan dan Dipertahankan dalam Sidang Ujian Tesis
Program Studi S2 Teknik Informatika
Program Pascasarjana Universitas AMIKOM Yogyakarta
pada Senin,08 Januari 2024

Tesis ini telah diterima sebagai salah satu syarat persyaratan
untuk memperoleh gelar Magister Komputer

Yogyakarta, 08 Januari 2024

Rektor

Prof. Dr. M. Suyanto, M.M.
NIK. 190302001

HALAMAN PERSETUJUAN

KOMPARASI TINGKAT AKURASI ALGORITMA RANDOM FOREST DAN K-NEAREST NEIGHBOR (KNN) UNTUK MENGKLASIFIKASI KELAYAKAN PENERIMAAN KREDIT BANK

COMPARISON OF ACCURACY LEVEL OF RANDOM FOREST AND K-NEAREST NEIGHBOR (KNN) ALGORITHM TO CLASSIFY ELIGIBILITY FOR ACCEPTING BANK CREDIT

Dipersiapkan dan Disusun oleh

Bayu Aji Santoso
22.55.1212

Telah Diajukan dan Dipertahankan dalam Sidang Ujian Tesis
Program Studi S2 Teknik Informatika
Program Pascasarjana Universitas AMIKOM Yogyakarta
pada Senin,08 Januari 2024

Pembimbing Utama

Prof. Dr. Kusrini, M.Kom
NIK. 190302106

Anggota Tim Penguji

Dr. Andi Sunyoto, M.Kom.
NIK. 190302052

Pembimbing Pendamping

Anggit Dwi Hartanto, M.Kom
NIK. 190302163

Alva Hendi Muhammad, S.T., M.Eng., Ph.D
NIK. 190302493

Prof. Dr. Kusrini, M.Kom
NIK. 190302106

Tesis ini telah diterima sebagai salah satu persyaratan
Untuk memperoleh gelar Magister Komputer

Yogyakarta, 08 Januari 2024
Direktur Program Pascasarjana

Prof. Dr. Kusrini, M.Kom
NIK. 190302106

HALAMAN PERNYATAAN KEASLIAN

Yang bertandatangan di bawah ini,

Nama mahasiswa : Bayu Aji Santoso
NIM : 22.55.1212
Konsentrasi : Business Intelligence

Menyatakan bahwa Tesis dengan judul berikut:
KOMPARASI TINGKAT AKURASI ALGORITMA *RANDOM FOREST* DAN K-*NEAREST NEIGHBOR* (KNN) UNTUK MENGKLASIFIKASI KELAYAKAN PENERIMAAN KREDIT BANK

Dosen Pembimbing Utama : Prof. Dr. Kusrini, M.Kom
Dosen Pembimbing Pendamping : Anggit Dwi Hartanto, M.Kom

1. Karya tulis ini adalah benar-benar ASLI dan BELUM PERNAH diajukan untuk mendapatkan gelar akademik, baik di Universitas AMIKOM Yogyakarta maupun di Perguruan Tinggi lainnya
2. Karya tulis ini merupakan gagasan, rumusan dan penelitian SAYA sendiri, tanpa bantuan pihak lain kecuali arahan dari Tim Dosen Pembimbing
3. Dalam karya tulis ini tidak terdapat karya atau pendapat orang lain, kecuali secara tertulis dengan jelas dicantumkan sebagai acuan dalam naskah dengan disebutkan nama pengarang dan disebutkan dalam Daftar Pustaka pada karya tulis ini
4. Perangkat lunak yang digunakan dalam penelitian ini sepenuhnya menjadi tanggung jawab SAYA, bukan tanggung jawab Universitas AMIKOM Yogyakarta
5. Pernyataan ini SAYA buat dengan sesungguhnya, apabila di kemudian hari terdapat penyimpangan dan ketidakbenaran dalam pernyataan ini, maka SAYA bersedia menerima SANKSI AKADEMIK dengan pencabutan gelar yang sudah diperoleh, serta sanksi lainnya sesuai dengan norma yang berlaku di Perguruan Tinggi.

Yogyakarta, 08 Januari 2024
Yang Menyatakan,

Bayu Aji Santoso

KATA PENGANTAR

Puji dan syukur saya panjatkan kepada Tuhan Yang Maha Esa yang telah memberikan rahmat, hidayah dan kekuatan sehingga saya dapat menyelesaikan tesis yang berjudul Komparasi Tingkat Akurasi Algoritma *Random Forest* Dan K-*Nearest Neighbor* (KNN) Untuk Mengklasifikasi Kelayakan Penerimaan Kredit Bank. Dengan selesainya tesis ini, Maka pada kesempatan ini saya mengucapkan terima kasih kepada :

1. Bapak Prof. Dr. M. Suyanto, M.M. selaku Direktur Program Pascasarjana Universsitas AMIKOM Yogyakarta.
2. Ibu Prof. Dr. Kusrini, M.Kom selaku dosen pembimbing utama yang selalu bijaksana memberikan bimbingan, nasehat serta waktunya selama penulisan tesis ini.
3. Bapak Anggit Dwi Hartanto, M.Kom selaku dosen pendamping yang selalu memberi arahan yang sangat bermanfaat untuk menyelesaikan tesis.
4. Kedua orang tua yang tak kenal lelah memotivasi saya untuk menyelesaikan program pasca sarjana ini.
5. Rekan rekan Mahasiswa Universitas AMIKOM Yogyakarta.
6. Kepala SMK Insan Mulia Kramat sudah memberikan dukungan serta dispensasi untuk menyelesaikan jenjang Pendidikan pascasarjana.

Semoga Allah Subhanahu wata'ala memberikan balasan yang lebih kepada semua yang telah ikut membantu saya dan menyelsaikan tesis ini. Demi perbaikan selanjutnya, saran dan kritik yang membangun akan diterima dengan senang hati dan rasa terima kasih.

Yogyakarta, 08 Januari 2024

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

INTISARI

Pemberian kredit merupakan salah satu penawaran bank yang ditawarkan kepada nasabah, namun pemberian kredit kepada nasabah yang tidak tepat dapat menimbulkan permasalahan seperti nasabah yang tidak membayar angsuran tepat waktu bahkan menunda sampai beberapa bulan pembayaran angsuran hingga terjadinya kredit macet sehingga hal ini dapat merugikan pihak bank. Oleh karena itu, dalam penelitian ini akan melakukan komparasi metode untuk mengetahui metode mana yang terbaik dalam melakukan klasifikasi kelayakan penerimaan kredit bank. Hasil penelitian diharapakan dapat dijadikan bahan pertimbangan pihak bank dalam upaya pemilihan nasabah kredit bank. Dalam penelitian ini menggunakan dataset dari UCI *Machine Learning Repository* merupakan data pembayaran kredit yang berjumlah 30000. Dataset dilakukan *split* dengan pembagian 80% data *train* dan 20% data *test* dengan jumlah masing-masing data yaitu data *train* 24000 dan data *test* 6000. Sedangkan, label yang digunakan yaitu Layak dan Tidak Layak

Dalam penelitian ini mengimplementasikan proses data mining menggunakan kerangka kerja CRISP-DM dan menggunakan bahasa pemrograman Python. Dari hasil evaluasi menggunakan *confusion matrix* mendapatkan nilai akurasi terbaik pada algoritma *random forest* yaitu 82.92%, *precision* sebesar 81.41%, *recall* sebesar 82.92% dan f1-*score* sebesar 80.83%. Sedangkan, pada algoritma KNN mendapatkan nilai *accuracy* sebesar 82.28%, *precision* sebesar 80.55%, *recall* sebesar 82.28% dan f1-*score* sebesar 79.97%.

Berdasarkan hasil evaluasi tersebut bahwa algoritma *Random Forest* mendapatkan akurasi terbaik dibandingkan dengan algoritma KNN dalam mengklasifikasi kelayakan penerimaan kredit bank.

Kata kunci: Komparasi, KNN, Kredit, Pemberian, *Random Forest*

ABSTRACT

Providing credit is one of the bank offers offered to customers, but extending credit to customers who are not appropriate can cause problems such as customers who do not pay installments on time and even delay payment of installments for several months until bad credit occurs so that this can be detrimental to the bank. Therefore, in this study a comparative method will be carried out to find out which method is the best in classifying the smoothness of bank credit payments. It is hoped that the results of the research can be used as material for consideration by the bank in the selection of bank credit customers. In this study using a dataset from the UCI Machine Learning Repository, the credit payment data totaled 30.000. The dataset is split by dividing 80% train data and 20% test data with the amount of each data, namely 24.000 train data and 6000 test data. Meanwhile, the labels used are Eligible and Ineligible

In this study, implementing the data mining process using the CRISP-DM framework and using the Python programming language. From the results of the evaluation using the confusion matrix, the best accuracy value for the random forest algorithm is 82.92%, precision is 81.41%, recall is 82.92% and f1-score is 80.83%. Meanwhile, the KNN algorithm obtains an accuracy value of 82.28%, a precision of 80.55%, a recall of 82.28% and an f1-score of 79.97%.

Based on the results of this evaluation, the Random Forest algorithm has the best accuracy compared to the KNN algorithm in classifying bank credit payments.

Keyword: Comparison, KNN, Credit, Giving, Random Forest

BAB I
PENDAHULUAN

1.1 Latar Belakang Masalah

Bank merupakan perusahaan yang memiliki data yang besar yang tersimpan di dalam *database* dan diolah menghasilkan sebuah informasi yang saling berkaitan tentang nasabah, data tersebut serta dapat digunakan untuk menjaga hubungan antar bank dengan nasabah yang valid, sehingga berguna untuk menentukan secara individual tentang penawaran produk bank (Subarkah, Pambudi, & Hidayah, 2020). Pemberian kredit merupakan salah satu penawaran bank yang ditawarkan kepada nasabah, namun pemberian kredit kepada nasabah yang tidak tepat dapat menimbulkan permasalahan seperti nasabah yang tidak membayar angsuran tepat waktu bahkan menunda sampai beberapa bulan pembayaran angsuran hingga terjadinya kredit macet sehingga hal ini dapat merugikan pihak bank. Dari permasalahan yang terjadi, diperlukan solusi yang dapat menyelesaikan permasalahan tersebut yaitu dengan menerapkan teknik data mining yaitu klasifikasi untuk menentukan kategori nasabah mana yang layak atau tidak layak untuk mengajukan pinjaman berikutnya.

Metode klasifikasi yang dapat digunakan seperti *Decision Tree, Naïve Bayes, Artificial Neural Network* (ANN), Regresi, *Support Vector Machine, Random Forest* dan K-*Nearest Neighbor* (KNN) dan metode lainnya (Maulidah, Gata, Aulianita, & Agustyaningrum, 2020). Penggunaan metode klasifikasi dalam penelitian ini yaitu metode K-*Nearest Neighbor* (KNN) dan *Random Forest*. KNN

merupakan metode yang terawasi dimana hasil dari uji data yang baru diklasifikasikan berdasarkan kelas mayoritas. Akan tetapi, terdapat permasalahan pada KNN yaitu perolehan nilai akurasi yang cenderung lebih rendah dibanding metode klasifikasi lainnya. Hal ini dibuktikan oleh penelitian terdahulu yang dilakukan oleh (Ginting, Lydia, & Zamzami, 2021) dengan mendapatkan akurasi KNN lebih rendah yaitu sebesar 66,35%. Selain itu, hal ini juga dibuktikan oleh penelitian sebelumnya yang dilakukan oleh (Erdiansyah, Lubis, & Erwansyah, 2022) dengan menghasilkan akurasi KNN lebih rendah dibanding dengan akurasi *random forest* yaitu tingkat akurasi KNN sebesar 76.78% dan *random forest* sebesar 86,56% pada klasifikasi pengobatan kutil. Feature selection merupakan salah satu teknik yang sangat penting serta sering dipakai pada pre-processing. Pada teknik ini dilakukan pengurangan jumlah fitur yang terlibat agar dapat menentukan kelas target dengan cara mengurangi fitur yang tidak sesuai dan data yang berlebihan. Metode PCA digunakan karena cukup efesien dapat mengatasi multikolineritas atau hubungan kuat antara dua variable atau lebih di berbagai kondisi dan dapat menghapuskan korelasi diantara variable bebas sehingga tidak dapat berkolerasi sama sekali dimana PCA dapat ditunjukan pada Persamaan 1 (Aini et al. 2022). Penggunaan seleksi fitur Metode PCA terbukti dapat meningkatkan hasil Akurasi pada algoritma KNN yang di buktikan oleh (Aini et al. 2022) yang menghasilkan akurasi paling tinggi sebesar 99,64% untuk Prediksi Hasil Produksi Agrikultur pada Algoritma K-Nearest Neighbor (KNN). Oleh karena itu, dalam penelitian ini akan melakukan percobaan menggunakan algoritma KNN dan *Random Forest* dengan menerapkan Feature selection PCA dalam

mengklasifikasi kelayakan penerimaan kredit bank untuk mengetahui performa dari *Random Forest* dan KNN dengan dataset yang digunakan dari UCI *Machine Learning Repository*. UCI *Machine Learning Repository* merupakan kumpulan *database*, teori domain, dan generator data yang digunakan oleh komunitas *machine learning* untuk menganalisis secara empiris algoritma *machine learning* (Ubaedi & Djaksana, 2022). Pada studi ini dataset yang digunakan dalam studi ini berasal dari bank penting (penerbit uang tunai dan kartu kredit) di Taiwan yang dikumpulkan oleh I-Cheng Yen (2016) yang terdiri dari 29.998 entri data, dibagi menjadi kelas "Dibayarkan" sebanyak 6.636 data dan "Tidak Dibayarkan" sebanyak 23.364 data, dengan 25 fitur yang mencakup berbagai jenis tipe data, termasuk numerik dan kategorik.

Dari uraian diatas, dalam penelitian ini melakukan komparasi metode *Random Forest* dan KNN dalam melakukan klasifikasi kelayakan penerimaan kredit bank. Alasan menggunakan metode *Random Forest* yaitu menghasilkan akurasi yang lebih tinggi dari metode lain, dapat mengatasi data dalam jumlah yang besar secara efisien, dan tidak terdapat pemangkasan variabel seperti pada algoritma pohon klasifikasi tunggal (Ramadhan, Susetyo, & Indahwati, 2019). Sedangkan, metode KNN yaitu dapat menghasilkan data yang lebih akurat dan efektif apabila memliki *training* data yang cukup besar (Rahardja, Juardi, & Agung, 2019). Selain itu, alasan lain pemilihan algoritma KNN dan *random forest* adalah pada penelitian-penelitian yang telah dijelaskan algoritma Random Forest dan KNN dapat mengeksplorasi tipe data dengan bentuk numerik maupun kategorik. Semetara itu alasan lainnya pada algoritma Random forest dan KNN, melewati

tahapan evaluasi. Dimana tahapan evaluasi yang digunakan menggunakan confusion matrix dan mode *K-fold cross validation. Confusion matrix* akan menghasilkan nilai *accuracy, precision, recall* dan *f1-score* dari algoritma *Random Forest* dan KNN, sedangkan model *K-Fold Cross Validation* menghasilkan memperkirakan seberapa akurat sebuah model prediktif ketika dijalankan dalam praktiknya. Algoritma *random forest* dan KNN dapat dikatakan *apple to apple* karena kedua algoritma tersebut sering digunakan pada kasus klasifikasi yang dapat digunakan untuk perbandingan (Angkasa & Pangaribuan, 2022). Oleh karena itu, dalam penelitian ini akan melakukan komparasi metode untuk mengetahui metode mana yang terbaik dalam melakukan klasifikasi kelayakan penerimaan kredit bank. Hasil penelitian diharapakan dapat dijadikan bahan pertimbangan pihak bank dalam upaya pemilihan nasabah kredit bank.

1.2 Rumusan Masalah

Berdasarkan uraian latar belakang di atas maka terdapat permasalahan yang akan di bahas pada penelitian ini dalam bentuk rumusan masalah. Rumusan masalah pada penelitian komparasi tingkat akurasi algoritma *Random Forest* dan K-*Nearest Neighbor* (KNN) untuk mengklasifikasi kelayakan penerimaan kredit bank sebagai berikut:

1. Bagaimana hasil yang diperoleh *Principal Component Analysis* (PCA) pada klasifikasi kelayakan penerimaan kredit?
2. Bagaimana tingkat akurasi algoritma *Random Forest* dan KNN dalam mengklasifikasikan kelayakan penerimaan kredit bank?

1.3 Batasan Masalah

Batasan masalah digunakan untuk memfokuskan penelitian. Dalam penelitian ini terdapat batasan masalah sebagai berikut:

1. Penelitian ini difokuskan pada penggunaan algoritma *Random Forest* dan K-*Nearest Neighbor* (KNN) dalam mengklasifikasikan kelayakan penerimaan kredit.
2. Penelitian ini hanya menggunakan data penerimaan kredit dari Dataset yang di ambil dari UCI *Machine Learning Repository* yang berasal dari bank penting (penerbit uang tunai dan kartu kredit) di Taiwan yang dikumpulkan oleh I-Cheng Yen (2016) yang terdiri dari 29.998 entri data, dibagi menjadi kelas "Dibayarkan" sebanyak 6.636 data dan "Tidak Dibayarkan" sebanyak 23.364 data, dengan 25 fitur yang mencakup berbagai jenis tipe data, termasuk numerik dan kategorik.
3. Penelitian ini tidak membahas masalah-masalah lain yang terkait dengan risiko kredit atau manajemen risiko kredit secara umum.
4. Penelitian ini menggunakan kategori nasabah pemberian kredit layak dan tidak layak.
5. Implementasi algoritma *Random Forest* dan K-*Nearest Neighbor* (KNN) menggunakan bahasa pemrograman Python dengan *tools* Google Colab.
6. Dataset yang digunakan memiliki 29.998 data dan terdiri dari 2 *class* dengan nama *class* yaitu "Y" serta terdiri dari *class* "Dibayarkan" memiliki 6.636 data, dan pada *class* "Tidak dibayarkan" memiliki 23.364 data.

7. Evaluasi yang digunakan yaitu *Confusion matrix* dan *K-flod Cross validation*

1.4 Tujuan Penelitian

Dari rumusan permasalahan yang ada, maka terdapat tujuan dalam penelitian ini yaitu:

1. Mengetahui hasil yang diperoleh *Principal Component Analysis* (PCA) pada klasifikasi kelayakan penerimaan kredit
2. Mengetahui performa dan tingkat akurasi algoritma *Random Forest* dalam mengklasifikasikan kelayakan penerimaan kredit bank.

1.5 Manfaat Penelitian

Dalam penelitian yang dilakukan, terdapat manfaat penelitian yaitu sebagai berikut:

1. Manfaat bagi industri perbankan:

   Hasil penelitian ini dapat membantu industri perbankan dalam meningkatkan efektivitas dan efisiensi pengambilan keputusan terkait penilaian risiko kredit, dengan memanfaatkan algoritma machine learning yang lebih akurat dan optimal dalam mengklasifikasikan kelayakan penerimaan kredit nasabah. Hal ini dapat membantu bank dalam mengurangi risiko kredit macet dan meningkatkan kualitas portofolio kredit mereka.

2. Manfaat bagi akademisi:

   Penelitian ini dapat memberikan kontribusi dalam pengembangan ilmu pengetahuan, khususnya dalam bidang machine learning dan aplikasinya

dalam industri perbankan. Selain itu, hasil penelitian ini juga dapat menjadi bahan referensi bagi peneliti selanjutnya yang ingin melakukan penelitian terkait penggunaan algoritma machine learning dalam industri perbankan.

3. Manfaat bagi masyarakat:

   Dengan menggunakan algoritma machine learning yang lebih akurat dalam mengklasifikasikan kelayakan penerimaan kredit nasabah, diharapkan dapat meningkatkan kepercayaan masyarakat dalam industri perbankan. Selain itu, penggunaan algoritma machine learning juga dapat mempercepat proses pengambilan keputusan dalam penilaian risiko kredit, sehingga dapat mempercepat proses persetujuan kredit bagi nasabah yang memenuhi kriteria.

4. Manfaat bagi pengembangan teknologi

   Penelitian ini dapat memberikan kontribusi dalam pengembangan teknologi machine learning, terutama dalam aplikasinya dalam industri perbankan. Hal ini dapat memperkaya pengetahuan dan pemahaman mengenai penggunaan algoritma *machine learning* dalam pemrosesan data yang kompleks, sehingga dapat diaplikasikan pada berbagai industri lainnya yang membutuhkan analisis data yang akurat dan cepat.

BAB II
TINJAUAN PUSTAKA

2.1 Tinjauan Pustaka

Tinjauan pustaka merupakan pembahasan terkait penelitian terdahulu yang dijadikan referensi dalam penelitian. Tinjauan pustaka yang digunakan sebagai berikut:

Penelitian yang dilakukan oleh Irma Handayani, Ikrimach melakukan diagnosis kanker payudara menggunakan algoritma K-*Nearest Neighbor* dan *Naïve Bayes*. Dalam bidang medis, banyak catatan penderita penyakit kanker payudara. Oleh karena itu, dalam penelitian tersebut bertujuan untuk menghasilkan algoritma terbaik dalam mengklasifikasi kanker payudara sehingga dapat diketahui kanker payudara ganas dan jinak dengan membandingkan algoritma K-*Nearest Neighbor* dan *Naïve Bayes*. Hasil penelitian menunjukkan K-*Nearest Neighbor* menghasilkan akurasi 95,79%, sedangkan *Naïve Bayes* menghasilkan akurasi 93,39% (Handayani & Ikrimach, 2020).

Penelitian berikutnya yang menggunakan metode KNN dilakukan oleh Taghfirul Azhima Yoga Siswa, Renaldi Panji Wibowo melakukan prediksi keterlambatan penerimaan kuliah yang dioptimasi dengan beberapa perbandingan algoritma seleksi fitur diantaranya *Mutual Information*, *Forward Selection*, *Backward*, dan *Recursive Elimination*. Data yang digunakan adalah data penerimaan SPP mahasiswa dari tahun 2019 - 2021 dengan teknik pembagian data menggunakan metode 5-fold cross validation. Dalam penelitian tersebut

menggunakan algoritma K-*Nearest Neighbor, Naive Bayes, C4.5, Random forest,* dan *Logistic Regression.* Hasil penelitian menunjukkan algoritma klasifikasi yang memiliki akurasi tertinggi terdapat pada *random forest* dan C4.5 dengan nilai akurasi sebesar 62,6%, *precision* 65%, *recall* 63% dan f1-*score* 61% (Siswa & Wibowo, 2023).

Berbeda dengan penelitian yang dilakukan oleh Ryan Bagus Wicaksono, Suci Aulia, Sugondo Hadiyoso, Bambang Hidayat melakukan klasifikasi tinggi dan berat manusia berdasarkan jejak kaki menggunakan metode gabor wavelet dan KNN. Mengidentifikasi tinggi dan berat badan biasanya dilakukan secara manual menggunakan timbangan dan alat ukur tinggi badan. Namun, masalah sering ditemukan ketika tubuh seseorang terpisah satu sama lain. Salah satu cara untuk memperkirakan berat dan tinggi badan adalah dengan mengukur panjang kaki. Hasil terbaik dari sistem yang diusulkan adalah akurasi 75% dengan perhitungan waktu 8,92 detik (Wicaksono, Aulia, Hadiyoso, & Hidayat, 2022).

Penerapan metode KNN juga dapat diterapkan pada identifiksi unsur hara tanaman jagung seperti yang dilakukan oleh Bain Khusnul Khotimah. Tanaman jagung memiliki ciri-ciri berupa gejala penglihatan baik dengan kerusakan batang, daun, tongkol jagung ada 17 fitur dari pengamatan untuk membangun sistem diagnostik menggunakan metode KNN. Studi ini menggunakan KNN dengan perhitungan jarak minkowski dengan menyesuaikan perubahan nilai K untuk menghasilkan yang terbaik pertunjukan. Hasil kinerja akhir dibandingkan dengan metode *Naïve Bayes* (NB). Hasil pengujian penelitian menggunakan K = 7 mendapatkan akurasi sebesar 92,40%, sedangkan pada *Naïve Bayes* menggunakan

uji k-*fold cross validation* dengan k=9 mendapatkan akurasi 92,29%. Metode klasifikasi KNN memiliki akurasi yang lebih baik daripada metode klasifikasi *Naïve Bayes* karena algoritma parametrik yang digunakan, yang mengasumsikan bahwa setiap atribut data independen, ini jarang terjadi properti di dunia nyata (Khotimah, 2022).

Penelitian lain melakukan komparasi metode *machine learning* dilakukan oleh Widya Apriliah, Ilham Kurniawan, Muhamad Baydhowi, Tri Haryati melakukan prediksi kemungkinan diabetes pada tahap awal. Tujuan dari penelitian ini adalah merancang model yang dapat memprakirakan kemungkinan terjadinya diabetes pada pasien dengan ketelitian yang maksimal. Klasifikasi adalah teknik data mining yang menetapkan kategori pada kumpulan data untuk membantu dalam memprediksi dan analisis yang lebih akurat. Algoritma klasifikasi *machine learning* yaitu *Suport Vector Machine*, *Naive Bayes* dan *Random Forest* digunakan dalam percobaan ini. Dataset yang digunakan yaitu Diabetes Hospital in Sylhet, Bangladesh yang bersumber dari UCI repository. Hasil penelitian menunjukkan *Random Forest* mengungguli dengan nilai akurasi tertinggi 97,88% dibandingkan algoritma lain. Hasil ini diverifikasi menggunakan kurva *Receiver Operating Characteristic* (ROC) secara tepat dan sistematis (Apriliah, Kurniawan, & Baydhowi, 2021).

Penelitian selanjutnya yang menggunakan *Random Forest* dilakukan oleh Budi Prasojoa, Emy Haryatmi melakukan analisa kelayakan pemberian kredit dengan menggunakan algoritma *Random Forest*. Dataset yang digunakan dalam penelitian didapat dari *machine learning repository* yaitu *german credit* data yang

disediakan dari Professor Dr. Hans Hofmann Institut f"ur Statistik und "Okonometrie Universit"at Hamburg. Variabel yang dianalisa yaitu V1 sampai dengan V20 dilakukan menggunakan perangkat lunak R. Tahapan metode penelitian menggunakan CRIPS-DM. Tahap pelatihan menggunakan 80% data dan Pengujian menggunakan 20% data secara acak dari 1000 data. Hasil performa dari algoritma *random forest* tersebut yaitu memiliki tingkat akurasi sebesar 0,83 atau 83% sehingga termasuk pada kategori klasifikasi modelnya sangat bagus (Prasojo & Haryatmi, 2021).

Penelitian yang melakukan komparasi metode KNN dan *Random Forest* dilakukan oleh Umri Erdiansyah, Ahmadi Irmansyah Lubis, Kamil Erwansyah melakukan komparasi metode KNN dan *Random Forest* pada klasifikasi pengobatan penyakit kutil. Penelitian ini akan berfokus pada komparasi metode klasifikasi K-*Nearest Neighbor* dengan *Random Forest* untuk melihat tingkat akurasi pada prediksi keberhasilan dari pengobatan penyakit kutil. Data untuk Immunotheraphy diperoleh dari UCI *Machine Learning Repository* dengan jumlah sebanyak 90 *record* data, 7 atribut dan 1 kelas atribut. Hasil penelitian yaitu metode KNN memperoleh tingkat akurasi sebesar 76.78%, kemudian metode *Random Forest* memperoleh tingkat akurasi sebesar 86.56%. Metode *Random Forest* memperoleh hasil yang lebih baik direkomendasikan jika dibandingkan dengan metode KNN (Erdiansyah, Lubis, & Erwansyah, 2022).

Penelitian yang dilakukan oleh Auliya Rahman Isnain, Jepi Supriyanto, Muhammad Pajar Kharisma melakukan implementasi KNN pada analisis sentimen pembelajaran daring. Data yang digunakan yaitu data Tweet sebanyak 1825 data

tweet Bahasa Indonesia data dikumpulkan sejak tanggal 1 Februari 2020 sampai dengan 30 September 2020 menggunakan *library python Tweepy*. Pembobotan kata dilakukan menggunakan TF-IDF, kemudian data diklasifikasi kedalam kelas positif dan negatif. Pengujian dilakukan dengan K sebanyak 20 didapatkan hasil akurasi tertinggi pada saat K = 10 dengan nilai akurasi 84,65% dengan presisi mencapai 87%, *recall* 86% f *measure* 87% serta *error rate* mencapai 0,12% dan di dapatkan pula kecenderungan opini publik terhadap Pembelajaran Daring Cenderung Positif (Isnain, Supriyanto, & Kharisma, 2021).

Penelitian yang dilakukan oleh Fanka Angelina Larasati, Dian Eka Ratnawati, Buce Trias Hanggara melakukan analisis sentimen pada ulasan aplikasi Dana menggunakan metode *Random Forest*. Aplikasi Dana memiliki banyak pengguna, sehingga sering kali terdapat ulasan positif, negatif dan netral yang tidak relevan dengan *rating* yang diberikan di *Google Play Store*. Data ulasan aplikasi Dana akan diperoleh menggunakan teknik *Web Scraping* menggunakan *API Google-Play-Scraper*. Data tersebut dilakukan *preprocessing* kemudian melakukan klasifikasi menggunakan metode *Random Forest*. Hasil pengujian dan analisis yang sudah dilakukan, dengan perbandingan data latih dan data uji 80%:20% diperoleh nilai *precision* 84%, *recall* 84%, F1-*Score* 84% dan *accuracy* sebesar 84% dengan kedalaman *tree* 65 dan jumlah *tree* 400 (Larasati, Ratnawati, & Hanggara, 2022).

Penelitian yang dilakukan oleh Febri Aldi, Irohito Nozomi, Soeheri melakukan klasifikasi jenis obat menggunakan algoritma KNN. Klasifikasi jenis obat sangat memengaruhi kesehatan pasien. Data pasien dibagi mnejadi data latih dan data uji dengan perbandingan 90:10, 80:20, 70:30 dan menggunakan model

Cross Validation. Nilai k yang digunakan yaitu k=3 dengan perhitungan *Euclidean Distance,* kemudian dilakukan evaluasi menggunakan *confusion matrix*. Performa KNN menghasilkan akurasi 98,33%, nilai *Precision* 98,8%, nilai *Recall* 96,2%, dan nilai F-*measure* 97,48% pada data 90:10. Semakin rendah nilai k, maka semakin tinggi kinerja yang dihasilkan (Aldi, Nozomi, & Soeheri, 2022).

Penelitian yang dilakukan oleh Manish Suyal, Parul Goyal melakukan analisis *review* metode KNN. Tujuan penelitian yaitu mengetahui bagaimana algoritma K-*Nearest Neighbor* (KNN) pada penerapan klasifikasi pembelajaran mesin pada dataset model dan bagaimana data yang diberikan diprediksi oleh model kelas mana data yang diberikan ini akan ada. Dalam penelitian ini mencoba menggunakan dataset yang sangat kecil berupa dataset siswa beserta nilainya untuk menentukan kelulusan siswa. Hasil penelitian berupa siswa baru lulus atau gagal dengan nilai k=3 (Suyal & Goyal, 2022).

Penelitian yang dilakukan oleh Agus Eko Minarno, Fauzi Dwi Setiawan Sumadi, Hardianto Wibowo, Yuda Munarko melakukan klasifikasi pola batik. Metode yang digunakan yaitu K-*Nearest Neighbor* (KNN) dan *Support Vector Machine* serta menggunakan fitur minimum GLCM. Fitur klasifikasi berjumlah 16 dengan hasil klasifikasi pada metode KNN sebesar 78,3% dan untuk SVM sebesar 92,3%. Oleh karena itu, metode yang digunakan sudah baik dalam mengklasifikasi pola batik (Minarno, Sumadi, Wibowo, & Munarko, 2020).

Penelitian yang dilakukan oleh S. Ponlatha, Mathisalini B, Deepthisri K. A, Kalaiyarasi. M, Kowshika.V melakukan klasifikasi genre musik. Data musik mencakup banyak genre seperti Rock, Pop, folk, Klasik dan banyak genre yang

diambil dari libROSA untuk analisis musik dan audio. *Deep learning* digunakan untuk melatih dan mengklasifikasikan sistem menggunakan KNN. Data yang berisi lebih banyak genre dan trek, maka model mengklasifikasi genre musik lebih akurat dan memberikan manfaat lebih agar proses pembelajaran (Ponlatha, B, A, M, & V, 2021).

Penelitian yang dilakukan oleh Arie Nugroho, Abdullah Husin melakukan implementasi metode *Random Forest* menggunakan normalisasi atribut. Tujuan penelitian yaitu menerapkan proses normalisasi atribut dan menggunakan algoritma random forest untuk mengatasi imbalanced dataset dan mengukur akurasi. Dataset yang digunakan yaitu dataset publik dari UCI Repository "car evaluation". Akurasi tertinggi yaitu 99%. Besarnya akurasi dapat dijadikan acuan bahwa model yang telah dibuat cocok dengan dataset yang digunakan dan dapat digunakan untuk dataset lain yang mempunyai karakteristik yang sama (Nugroho & Husin, 2022).

Penelitian yang dilakukan oleh Rashmi Agrawal melakukan klasifikasi spam menggunakan metode *Random Forest*. Dataset yang digunakan yaitu data berita diambil dari github dan disimpan dalam format csv. Hasil penelitian mendapatkan akurasi 97.4% (Agrawal, 2020).

Penelitian yang dilakukan oleh Eko Martantoh dan Intan Rosiyana dalam menerapkan algoritma KNN pada penentuan kelayakan kredit pada PT. BPR "Para Sahabat" di Bekasi. Dalam penelitian tersbut membuat sistem berasis *website* menggunakan bahasa pemrograman PHP dan database MySQL. Hasil penelitian yaitu sistem berbasis *website* yang dapat membantu pihak BPR dalam menentukan

kelayakan penerima kredit berdasarkan nilai yang diberikan dan analisis dari algortima KNN (Martantoh & Rosiyana, 2022).

2.2 Keaslian Penelitian

Pada penelitian yang dilakukan akan melakukan perbandingan tingkat akurasi dari algoritma *Random Forest* dan K-*Nearest Neighbor* (KNN) untuk mengklasifikasi kelayakan penerimaan kredit bank. Untuk mengetahui posisi penelitian yang akan dilakukan dengan beberapa penelitian yang dilakukan oleh para peneliti yang dijadikan sebagai acuan dalam melakukan penelitian ini, perbandingan dapat dilihat pada Tabel 2.1.

Tabel 2.1 Matriks literatur review dan posisi penelitian Komparasi Tingkat Akurasi Algoritma *Random Forest* Dan K-*Nearest Neighbor* (KNN) Untuk Mengklasifikasi Kelayakan Penerimaan Kredit Bank

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
1	*Accuracy Analysis of K-Nearest Neighbor and Naïve Bayes Algorithm in the Diagnosis of Breast Cancer*	Peneliti: Irma Handayani, Ikrimach Publikasi: JURNAL INFOTEL Tahun: 2020	Tujuan penelitian yaitu menghasilkan algoritma terbaik dalam mengklasifikasikan kanker payudara sehingga pasien	Hasil penelitian yaitu mendapatkan akurasi 95,79% untuk K *Nearest Neighbor* dan 93,39% untuk *Naïve Bayes*.	Penelitian berikutnya dapat meningkatkan hasil akurasi pada metode *Naïve Bayes*, karena metode ini mengasumsikan setiap atribut adalah independent.	Penelitian tersebut melakukan klasifikasi kanker payudara menggunakan metode KNN dan *Naïve Bayes*. Penelitian saat ini yaitu melakukan klasifikasi kelayakan penerimaan kredit bank dengan

Table 2.1 (lanjutan)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
			eksisting parameter dapat diprediksi mana kanker payudara ganas dan jinak.			melakukan komparasi metode *Random Forest* dan KNN.
2	Komparasi Metode Seleksi Fitur Dalam Prediksi Keterlambatan Penerimaan Biaya Kuliah	Peneliti: Taghfirul Azhima Yoga Siswa dan Renaldi Panji Wibowo Publikasi: JURNAL INFOTEL Tahun: 2022	Tujuan penelitian yaitu melakukan perbandingan performa dari metode klasifikasi K-*Nearest Neighbor, Naive Bayes,* C4.5, *Random forest,* dan *Logistic Regression* dengan seleksi fitur untuk prediksi keterlambatan	Hasil penelitian menunjukkan algoritma klasifikasi yang memiliki akurasi tertinggi terdapat pada random forest dan C4.5 dengan nilai akurasi sebesar 62,6%, precision 65%, recall 63% dan f1-score 61%	Penelitian berikutnya dapat meningkatkan akurasi pada metode seleksi fitur yang lain.	Penelitian tersebut melakukan perbandingan performa dengan beberapa algoritma serta seleksi fitut untuk klasifikasi keterlambatan pembayaran biaya kuliah. Penelitian saat ini yaitu melakukan klasifikasi kelayakan penerimaan kredit bank dengan melakukan komparasi metode *Random Forest* dan KNN.

Table 2.1 (lanjutan)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
			pembayaran kuliah.			
3.	*Human height and weight classification based on footprint using gabor wavelet and KNN methods*	Peneliti: Ryan Bagus Wicaksono, Suci Aulia, Sugondo Hadiyoso dan Bambang Hidayat Publikasi: JURNAL INFOTEL Tahun: 2022	Tujuan penelitian yaitu melakukan klasifikasi tinggi dan berat manusia berdasarkan jejak kaki menggunakan metode gabor wavelet dan KNN.	Hasil terbaik dari sistem yang diusulkan adalah akurasi 75% dengan perhitungan waktu 8,92 detik.	Penelitian berikutnya yaitu mengeksplorasi metode pemrosesan gambar lainnya untuk meningkatkan deteksi akurasi dengan sampel yang lebih besar.	Penelitian tersebut melakukan klasifikasi tinggi dan berat badan manusia berdasarkan jejak kaki menggunakan metode KNN. Penelitian saat ini yaitu melakukan klasifikasi kelayakan penerimaan kredit bank dengan melakukan komparasi metode *Random Forest* dan KNN.
4.	*Performance of the K-Nearest Neighbors method on*	Peneliti: Bain Khusnul Khotimah	Tujuan penelitian yaitu menerapkan metode KNN untuk identifikasi unsur	Hasil kinerja akhir dibandingkan dengan metode *Naïve Bayes* (NB). Hasil pengujian penelitian	-Penelitian selanjutnya dapat menggunakan dataset yang berbeda.	Penelitian tersebut melakukan identifikasi unsur hara tanaman jagung menggunakan metode KNN.

Table 2.1 (lanjutan)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
	identification of maize plant nutrients	Publikasi: JURNAL INFOTEL Tahun: 2022	hara tanaman jagung.	menggunakan K = 7 mendapatkan akurasi sebesar 92,40%, sedangkan pada *Naïve Bayes* menggunakan uji k-*fold crossvalidation* dengan k=9 mendapatkan akurasi 92,29%.		Penelitian saat ini yaitu melakukan klasifikasi kelayakan penerimaan kredit bank dengan melakukan komparasi metode *Random Forest* dan KNN.
5.	Prediksi Kemungkinan Diabetes pada Tahap Awal Menggunakan Algoritma Klasifikasi *Random Forest*	Peneliti: Widya Apriliah, Ilham Kurniawan, Muhamad Baydhowi, Tri Haryati Publikasi: SISTEMASI: Jurnal Sistem Informasi	Tujuan penelitian yaitu merancang model yang dapat memprakirakan kemungkinan terjadinya diabetes pada pasien dengan ketelitian yang maksimal.	Hasil penelitian menunjukkan *Random Forest* mengungguli dengan nilai akurasi tertinggi 97,88% dibandingkan algoritma lain. Hasil ini diverifikasi menggunakan kurva *Receiver Operating Characteristic*	Sistem yang dirancang dengan algoritma klasifikasi *machine learning* dapat digunakan untuk memprediksi atau mendiagnosis penyakit lain. Penelitian dapat diperpanjang dan ditingkatkan untuk otomatisasi analisis diabetes termasuk	Penelitian tersebut merancang model klasifikasi diabetes menggunakan metode *Suport Vector Machine*, *Naive Bayes* dan *Random Forest*. Penelitian saat ini yaitu melakukan klasifikasi kelayakan penerimaan kredit bank dengan melakukan komparasi

Table 2.1 (lanjutan)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
		Tahun: 2021		(ROC) secara tepat dan sistematis	beberapa algoritma *machine learning* lainnya.	metode *Random Forest* dan KNN.
6.	Analisa Prediksi Kelayakan Pemberian Kredit Pinjaman dengan Metode *Random Forest*	Peneliti: Budi Prasojo dan Emy Haryatmi Publikasi: Jurnal Nasional Teknologi dan Sistem Informasi Tahun: 2021	Tujuan penelitian yaitu mengetahui penerapan metode klasifikasi dengan algoritma *random forest* serta analisa hasil terbaik dari algoritma *random forest* pada setiap kreditur.	Hasil performa dari algoritma *random forest* tersebut yaitu memiliki tingkat akurasi sebesar 0,83 atau 83% sehingga termasuk pada kategori klasifikasi modelnya sangat bagus.	Penelitian selanjutnya dapat menggunakan skenario dataset yang berbeda selain 80: 20 untuk mengetahui performa model pada skenario yang lain.	Penelitian tersebut merancang model klasifikasi kelayakan pemberian kredit menggunakan metode *Random Forest*. Penelitian saat ini yaitu melakukan klasifikasi kelayakan penerimaan kredit bank dengan melakukan komparasi metode *Random Forest* dan KNN.
7.	Komparasi Metode K-*Nearest Neighbor* dan *Random Forest* Dalam	Peneliti: Umri Erdiansyah, Ahmadi Irmansyah	Tujuan penelitian yaitu melakukan komparasi metode KNN dan	Hasil penelitian yaitu metode KNN memperoleh tingkat akurasi sebesar 76.78%, kemudian	Penelitian selanjutnya dapat menerapkan pada studi kasus berbeda dan meningkatkan	Penelitian tersebut melakukan komparasi metode KNN dan *Random Forest* dalam

Table 2.1 (lanjutan)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
	Prediksi Akurasi Klasifikasi Pengobatan Penyakit Kutil	Lubis, Kamil Erwansyah Publikasi: JURNAL MEDIA INFORMATIKA BUDIDARMA Tahun: 2022	*Random Forest* pada klasifikasi pengobatan penyakit kutil.	metode *Random Forest* memperoleh tingkat akurasi sebesar 86.56%.	akurasi metode KNN.	klasifikasi pengobatan penyakit kutil. Penelitian saat ini yaitu melakukan klasifikasi kelayakan penerimaan kredit bank dengan melakukan komparasi metode *Random Forest* dan KNN.
8.	*Implementation of K-Nearest Neighbor (KNN) Algorithm For Public Sentiment Analysis of Online Learning*	Peneliti: Auliya Rahman Isnain, Jepi Supriyanto, Muhammad Pajar Kharisma Publikasi: IJCCS (Indonesian	Tujuan penelitian yaitu melakukan analisis sentimen pembelajaran daring menggunakan metode KNN.	Pengujian dilakukan dengan K sebanyak 20 didapatkan hasil akurasi tertinggi pada saat K = 10 dengan nilai akurasi 84,65% dengan presisi mencapai 87%, *recall* 86% f *measure* 87% serta *error rate* mencapai	Penelitian selanjutnya yaitu dapat meningkatkan jumlah data training yang digunakan pada proses klasifikasi, implementasi model data yang lebih besar dan diharapkan untuk menggunakan dua atau lebih	Penelitian tersebut melakukan analisis sentimen pada pembelajaran daring menggunakan metode KNN. Penelitian saat ini yaitu melakukan klasifikasi kelayakan penerimaan kredit bank dengan melakukan komparasi

Table 2.1 (lanjutan)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
		Journal of Computing and Cybernetics Systems) Tahun: 2021		0,12% dan di dapatkan pula kecenderungan opini publik terhadap Pembelajaran Daring Cenderung Positif	metode untuk meningkatkan akurasi sistem.	metode *Random Forest* dan KNN.
9.	Analisis Sentimen Ulasan Aplikasi Dana dengan Metode *Random Forest*	Peneliti: Fanka Angelina Larasati, Dian Eka Ratnawati, Buce Trias Hanggara Publikasi: Jurnal Pengembangan Teknologi Informasi	Tujuan penelitian yaitu melakukan analisis sentimen pada ulasan aplikasi Dana menggunakan metode *Random Forest.*	Hasil penelitian yaitu perbandingan data latih dan data uji 80%:20% diperoleh nilai *precision* 84%, *recall* 84%, F1-*Score* 84% dan *accuracy* sebesar 84% dengan kedalaman *tree* 65 dan jumlah *tree* 400	Penelitian selanjutnya yaitu menggunakan dataset yang lebih banyak agar hasil penelitian maksimal.	Penelitian tersebut melakukan analisis sentimen ulasan aplikasi Dana menggunakan metode *Random Forest.* Dataset yang digunakan dari *google play store.* Penelitian saat ini melakukan komparasi akurasi metode *Random Forest* dan KNN untuk mengklasifikasi kelayakan penerimaan kredit, sehingga dapat diketahui metode yang

Table 2.1 (lanjutan)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
		dan Ilmu Komputer Tahun: 2022				terbaik dalam klasifikasi kelayakan penerimaan kredit.
10.	*Comparison of Drug Type Classification Performance Using KNN Algorithm*	Peneliti: Febri Aldi, Irohito Nozomi, Soeheri Publikasi: Sinkron: Jurnal dan Penelitian Teknik Informatika Tahun: 2022	Tujuan penelitian yaitu melakukan klasifikasi jenis obat menggunakan KNN.	Hasil penelitian dengan menggunakan nilai k=3 yaitu menghasilkan akurasi 98,33%, nilai *Precision* 98,8%, nilai *Recall* 96,2%, dan nilai F-*measure* 97,48% pada data 90:10. Semakin rendah nilai k, maka semakin tinggi kinerja yang dihasilkan	Penelitian selanjutnya yaitu mengembangkan model *cross validation* menggunakan algoritma Electre.	Penelitian tersebut melakukan klasifikasi jenis obat menggunakan KNN dengan nilai k=3 yang melakukan perhitungan Euclidean Distance. Penelitian saat ini mengklasifikasi kelayakan penerimaan kredit menggunakan metode *Random Forest* dan KNN untuk dilakukan komparasi performa dengan perhitungan *confusion matrix*.

Table 2.1 (lanjutan)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
11.	*A Review on Analysis of K-Nearest Neighbor Classification Machine Learning Algorithms based on Supervised Learning*	Peneliti: Manish Suyal, Parul Goyal Publikasi: International Journal of Engineering Trends and Technology Tahun: 2022	Tujuan penelitian yaitu mengetahui bagaimana algoritma K-*Nearest Neighbor* (KNN) pada penerapan klasifikasi pembelajaran mesin pada dataset model dan bagaimana data yang diberikan diprediksi oleh model kelas mana data yang diberikan ini akan ada.	Dalam penelitian mencoba menggunakan dataset yang sangat kecil berupa dataset siswa beserta nilainya untuk menentukan kelulusan siswa. Hasil penelitian berupa siswa baru lulus atau gagal dengan nilai k=3	Penelitian selanjutnya, melakukan pembelajaran tanpa pengawasan dan sekarang kerjakan data yang tidak berlabel alih-alih mengerjakan data yang berlabel.	Penelitian tersebut melakukan analisis metode KNN pada klasifikasi kelulusan siswa dengan k=3, penelitian tersbut berhasil menentukan siswa yang lulus dan gagal, namun tidak dilakukan evaluasi untuk mengetahui keakuratan metode. Penelitian saat ini melakukan komparasi akurasi metode *Random Forest* dan KNN untuk mengklasifikasi kelayakan penerimaan kredit.
12.	*Classification of batik patterns using K-Nearest*	Peneliti: Agus Eko Minarno,	Tujuan penelitian yaitu melakukan	Hasil klasifikasi pada metode KNN sebesar 78,3% dan	Penelitian selanjutnya yaitu menggunakan	Penelitian tersesbut menerapkan fitur GLCM untuk mengenali pola

Table 2.1 (lanjutan)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
	neighbor and support vector machine	Fauzi Dwi Setiawan Sumadi, Hardianto Wibowo, Yuda Munarko Publikasi: Bulletin of Electrical Engineering and Informatics Tahun: 2020	klasifikasi pola batik menggunakan KNN dan SVM dengan fitur minimum GLCM.	untuk SVM sebesar 92,3%.	skenario pembagian dataset yang berbeda.	pada citra batik. Sedangkan, dalam penelitian saat ini tidak menggunakan metode GLCM. Penelitian saat ini akan menggunakan metode *Random Forest* dan KNN untuk melakukan komparasi metode.
13.	*Music Genre Classification Using Deep Learning with KNN*	Peneliti: S. Ponlatha, Mathisalini B, Deepthisri K. A, Kalaiyarasi. M,	Tujuan penelitian yaitu mengklasifikasikan jenis genre musik sistem menggunakan KNN.	Hasil penelitian yaitu klasifikasi jenis genre musik berdasarkan metode KNN.	Penelitian selanjutnya yaitu mengklasifikasikan musik yang didukung suasana hati, alur, komposer, penampilan, lirik, meteran, progresi	Penelitian tersebut mengklasifikasi jenis genre musik menggunakan dataset libROSA. Metode *Neural Network* terlatih sebagai ekstraktor fitur dan meningkatkan

Table 2.1 (lanjutan)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
		Kowshika. V Publikasi: International Journal of Advanced Research in Science, Communication and Technology (IJARSCT) Tahun: 2021			akor, instrument hadir, dll.	kinerja dan kecepatan algoritma klasik. Sedangkan, penelitian saat ini yaitu melakukan klasifikasi kelayakan penerimaan kredit bank dengan dataset yang digunakan secara publik dari UCI *machine learning repository* berupa data penerimaan kredit.
14.	Analisis Performa *Random Forest* Menggunakan Normalisasi Atribut	Peneliti: Arie Nugroho, Abdullah Husin Publikasi: SISTEMASI: Jurnal	Tujuan penelitian yaitu menerapkan proses normalisasi atribut dan menggunakan algoritma *random forest*	Hasil penelitian yaitu Akurasi tertinggi yaitu 99%. Besarnya akurasi dapat dijadikan acuan bahwa model yang telah dibuat cocok dengan dataset yang	Penelitian selanjutnya dapat menggunakan dataset yang lain, ditambahkan dengan pelatihan dan pengujian untuk dataset publik lain dalam bidang	Penelitian tersebut melakukan klasifikasi untuk menentukan pola dari dataset menggunakan data evaluasi mobil yang diambil dari data publik dari web UCI *Repository*.

Table 2.1 (lanjutan)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
		Sistem Informasi Tahun: 2022	untuk mengatasi imbalanced dataset dan mengukur akurasi.	digunakan dan dapat digunakan untuk dataset lain yang mempunyai karakteristik yang sama	klasifikasi yang mempunyai missing values dan inkonsistensi data, membandingkan beberapa algoritma klasifikasi atau ensemble learning method dan teknik preprocessing yang lain untuk mendapatkan variasi hasil atau untuk menghasilkan akurasi yang lebih baik dan sebagai kontribusi ke pengetahuan.	Penelitian saat ini akan menggunakan dataset UCI *machine learning repository* berupa data pembayaran kredit untuk mengklasifikasi kelayakan penerimaan kredit bank serta melakukan komparasi algoritma *Random Forest* dan KNN.
15.	*Spam Classification Using Random Forest Algorithm*	Peneliti: Rashmi Agrawal Publikasi: Palarch's	Tujuan penelitian yaitu melakukan klasifikasi spam pada berita	Hasil penelitian mendapatkan akurasi 97.4%	Penelitian selanjutnya dapat melakukan *preprocessing* pada dataset yang	Penelitian tersbut melakukan klasifikasi spam menggunakan metode *Random Forest* dengan dataset berita dari

Table 2.1 (lanjutan)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
		Journal Of Archaeology Of Egypt/Egyptology Tahun: 2020	menggunakan metode *Random Forest*.		digunakan sebelum melakukan klasifikasi untuk mendapatkan akurasi yang lebih baik.	Github. Sedangkan, penelitian saat ini akan melakukan *preprocessing* pada dataset yang digunakan, kemudian dilanjutkan dengan klasifikasi menggunakan metode *Random Forest* dan KNN.
16.	Penerapan Algoritma K-Nearest Neighbor Dalam Penentuan Kelayakan Penerima Kredit Pada PT. BPT "Para Sahabat" Di Bekasi	Peneliti: Eko Martantoh, Intan Rosiyana Publikasi: Jurnal Informatika SIMANTIK Tahun: 2022	Tujuan penelitian ini membuat sistem yang dapat membantu pihak bank dalam menentukan siapa yang layak menerima kredit dengan menerapkan algoritma KNN.	Penerapan Algoritma K-*Nearest Neighbor* sesuai diimplementasikan untuk menentukan ke efektifan dalam penentuan pemberian kredit bagi Nasabah yang baru berdasarkan pembobotan nilai yang diberikan	Penelitian selanjutnya dapat meluas area cakupan pengguna sistem tersebut.	Penelitian tersebut membuat sistem berbasis *website* menggunakan bahasa pemrograman PHP dan database MySQL yang digunakan pihak bank BPR dalam menentukan kelayakan kredit pada nasabah berdasarkan metode KNN. Penelitian saat ini melakukan implementasi

Table 2.1 (lanjutan)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
				dalamanalisa pemberian kredit.		menggunakan bahasa pemrograman Python untuk komparasi metode *Random Forest* dan KNN sehingga dapat diketahui metode terbaik.

2.3 Landasan Teori

2.3.1 Bank

Bank merupakan perusahaan yang memiliki data yang besar yang tersimpan di dalam *database* dan diolah menghasilkan sebuah informasi yang saling berkaitan tentang nasabah, data tersebut serta dapat digunakan untuk menjaga hubungan antar bank dengan nasabah yang valid, sehingga berguna untuk menentukan secara individual tentang penawaran produk bank (Subarkah, Pambudi, & Hidayah, 2020).

Perbankan adalah lembaga intermediasi yang berperan dalam perkembangan sektor keuangan suatu negara. Intermediasi keuangan (*Financial intermediation*) merupakan suatu aktivitas penting dalam perekonomian, karena dapat menciptakan aliran dana dari pihak yang tidak produktif kepada pihak yang produktif dalam mengelola dana. hal ini akan membantu mendorong perekonomian menjadi lebih efisien dan dinamis (Prijanto, Pulung, & Sari, 2021).

2.3.2 Kredit

Menurut Undang–undang Perbankan No.10 Tahun 1998, Kredit adalah penyediaan uang atau tagihan, berdasarkan persetujuan atau kesepakatan pinjam–meminjam antara bank dengan pihak lain yang mewajibkan pihak peminjam melunasi utangnya setelah jangka waktu tertentu dengan pemberian bunga (Harlina, Suryani, & Kadang, 2022).

Kredit merupakan dana yang diberikan oleh bank kepada pihak lain berdasarkan perjanjian pinjam-meminjam, yang mewajibkan peminjam melunasi

pinjamannya setelah jangka waktu tertentu, dengan memberi bunga sebagai imbalannya (Ubaedi & Djaksana, 2022).

Dalam memilih nasabah yang layak untuk menerima kredit perlu dilakukan analisis kelayakan calon nasabah. Nasabah bisa disebut sebagai orang yang menggunakan pelayanan yang disediakan oleh bank, seseorang atau badan usahan maupun lembga yang memiliki rekening simpanan dan pinjaman, dan akan melakukan transaksi lainya, baik *online* mauoun *offline*. Jadi yang bisa disebut layak menjadi calon nasabah apabila memenuhi ketentuan yang ditetapkan oleh suatu bank, lembaga keuangan lainnya (Harlina, Suryani, & Kadang, 2022).

2.3.3 Data Mining

Data mining merupakan penggabungan dari beberapa disiplin ilmu yang dimanfaatkan untuk menggali informasi dari sekumpulan data. Informasi yang diperoleh selanjutnya dapat dimanfaatkan dalam proses penarikan kesimpulan, manajemen informasi, pengendalian proses dan sebagainya (Sartika & Saluza, 2022).

Data mining memiliki fungsi mencari pengetahuan yang bermanfaat dari sekumpulan data yang banyak. Data mining terdapat 5 peran utama yaitu (Ubaedi & Djaksana, 2022):

1. Estimasi

   Digunakan untuk memperikirakan nilai yang belum diketahui, target nilai bersifat numerik dari atribut numerik.

2. *Forecasting*

   Sama dengan estimasi, bedanya terdapat penambahan atribut *time series*.

3. Klasifikasi

   Digunakan untuk memprediksi kejadian dimasa depan. Target prediksi bersifat nominal dari atribut nominal atau numerik.

4. Klastering

   Digunakan untuk mengelompokkan objek yang serupa dalam satu klaster, tetapi antar klaster mempunyai karakter yang berbeda. Klatering tidak memiliki target, pengelompokan dibuat dari atribut yang bersifat numerik.

5. Asosiasi

   Digunakan untuk mencari hubungan yang ada pada nilai atribut dari sekumpulan data yang sering muncul secara bersamaan.

2.3.4 Klasifikasi

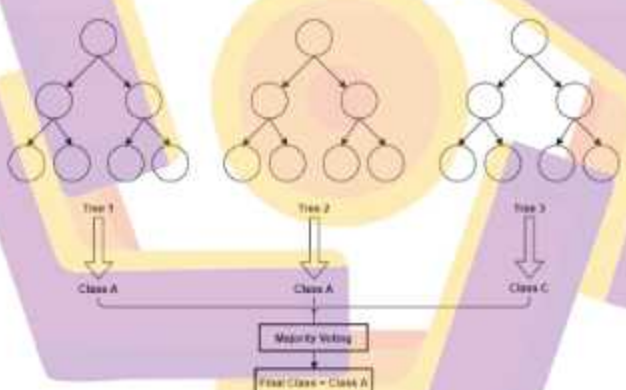
Klasifikasi merupakan teknik data mining klasik yang didasarkan pada pembelajaran mesin. Tujuan dari klasifikasi adalah memberikan label/kelas pada suatu data yang belum diketahui berdasarkan sekumpulan data yang telah diketahui label/kelasnya (Sartika & Saluza, 2022).

Klasifikasi data terdiri dari 2 (dua) langkah proses. Pertama adalah *learning* (*fase training*), dimana algoritma klasifikasi dibuat untuk menganalisa data *training* lalu di representasikan dalam bentuk *rule* klasifikasi. Proses kedua

adalah klasifikasi, dimana data tes digunakan untuk memperkirakan akurasi dari *rule* klasifikasi (Harlina, Suryani, & Kadang, 2022).

2.3.5 Random Forest

Random Forest merupakan salah satu metode CART (*Classification and Regression Tree*) dalam data mining dan tidak memerlukan asumsi apapun. Metode ini menggunakan konsep pohon keputusan (decision tree). Model ini dibentuk dari banyak pohon sehingga membentuk sebuah kumpulan pohon seperti hutan (*forest*) dengan menerapkan metode *bootstrap aggregating* (*bagging*) dan *random feature selection* (Iman & Wijayanto, 2021). Ilustrasi dari pohon keputusan dan pengambilan keputusan dengan random forest terdapat pada Gambar 2.2.



Gambar 2.2 Ilustrasi Random Forest

Sumber: (Iman & Wijayanto, 2021)

Kelebihan *random forest* (Iman & Wijayanto, 2021):

1. Akurasi sangat baik dibandingkan metode klasifikasi lainnya
2. Mampu menangani data *imbalanced*

3. Dapat menangani data dengan sampel besar
4. Dapat mengatasi data dengan *noise* dan *missing* data
5. *Error* yang dihasilkan relatif rendah

Kekurangan *random forest* (Iman & Wijayanto, 2021):

1. Akurasi yang dihasilkan cenderung tidak stabil
2. Waktu pemrosesan yang lama
3. Membutuhkan tuning model yang tepat untuk data

2.3.6 K-*Nearest Neighbor* (KNN)

K-*Nearest Neighbor* (KNN) adalah sebuah metode yang digunakan sebagai klasifikasi terhadap objek berdasarkan data pembelajaran yang jaraknya paling dekat dengan objek tersebut. Metode KNN terbagi menjadi dua fase, yaitu fase pembelajaran (*training*) dan fase klasifikasi atau pengujian (*testing*). Algoritma KNN cukup mudah untuk diimplementasikan karena bekerja menurut jarak terdekat dari *query instance* ke *sample* latih untuk menentukan tetangga terdekatnya. Jarak dekat atau jauhnya tetangga dihitung dengan menggunakan jarak *Euclidean Distance* yang dapat dilihat pada persamaan 1 (Hartono, Santoso, & Rostianingsih, 2022).

$$d = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \quad (2.1)$$

Berikut merupakan langkah – langkah untuk menghitung algoritma KNN (Hartono, Santoso, & Rostianingsih, 2022):

1. Menentukan nilai parameter k.

2. Menghitung jarak antara data training dan data testing dengan menggunakan perhitungan jarak *Euclidean Distance*.
3. Mengurutkan jarak euclid yang terbentuk dari yang paling kecil.
4. Menentukan jarak terdekat sampai urutan ke k.
5. Mengumpulkan label class Y (klasifikasi *Nearest Neighbor*).
6. Menghitung jumlah kelas dari tetangga terdekat yang terbanyak dan tetapkan kelas tersebut sebagai kelas data yang akan dievaluasi.

2.3.7 Python

Python merupakan bahasa pemrograman interpretatif multiguna dengan filosofi perancangan yang berfokus pada tingkat keterbacaan kode. *Python* dikenal sebagai bahasa yang menggabungkan kapabilitas, kemampuan, dengan sintaksis kode yang sangat jelas, dan dilengkapi dengan fungsionalitas pustaka standar yang besar serta komprehensif (Melinda, Ningrum, Suryabrata, Dwipa, & Sukoco, 2021).

Python juga merupakan salah satu bahasa pemrograman yang didukung oleh banyak komunitas yang besar. Seperti halnya pada bahasa pemrograman dinamis lainnya, python umumnya digunakan sebagai bahasa script meski pada praktiknya penggunaan bahasa ini lebih luas mencakup konteks pemanfaatan yang umumnya tidak dilakukan dengan menggunakan bahasa script. *Python* dapat digunakan untuk berbagai keperluan pengembangan perangkat lunak dan juga *Python* mampu berjalan di berbagai platform sistem operasi. Beberapa patform sistem operasi tersebut diantaranya ialah Linux atau Unix, Windows, Mac OS X,

Java Virtual Machine, Amiga, Palm, Symbian (untuk produk-produk Nokia). *Python* didistribusikan dengan beberapa lisensi yang berbeda dari beberapa versi. Namun pada prinsipnya *Python* dapat diperoleh dan dipergunakan secara bebas, bahkan untuk kepentingan komersial. Lisensi *Python* tidak bertentangan baik menurut definisi *Open Source* maupun *General Public License* (Karunia & Thanta, 2021).

2.3.8 *Google Colaboratory*

Colaboratory, atau "*Colab*" merupakan produk dari *Google Research*. Colab memungkinkan siapa saja menulis dan mengeksekusi kode python arbitrer melalui *browser*, dan sangat cocok untuk *machine learning*, analisis data, serta pendidikan. Secara lebih teknis, *Colab* merupakan layanan *notebook Jupyter* yang dihosting dan dapat digunakan tanpa penyiapan, serta menyediakan akses gratis ke resource komputasi termasuk GPU. *Resource Colab* tidak dijamin dan sifatnya terbatas, serta batas penggunaannya terkadang berfluktuasi. Hal ini diperlukan agar *Colab* dapat menyediakan *resource* secara gratis. Tujuan jangka panjang pihak *Google* adalah untuk terus menyediakan versi gratis *Colab*, dan di saat yang bersamaan berkembang secara berkelanjutan untuk memenuhi kebutuhan pengguna *Google* (Soen, Marlina, & Renny, 2022).

Google menyediakan fasilitas pemrograman online yang dapat diakses di http://colab.research.google.com dengan *Integrated Environment Development* (IDE) merujuk Jupyter Notebook dengan ekstensi *.ipynb. Keunggulan IDE ini adalah fasilitas yang disediakan oleh *Google* untuk menggunakan GPU dan TPU milik mesin pencari ini yang terkenal tangguh. Pemanfaatan GPU/TPU pengguna

dapat masuk ke menu edit lalu memilih GPU atau TPU setelah masuk ke *notebook settings* (Handayanto & Herlawati, 2020).

2.3.9 K-Fold *Cross Validation*

K-Fold Cross Validation adalah salah satu dari jenis pengujian cross validation yang berfungsi untuk menilai kinerja proses sebuah metode algoritme dengan membagi sampel data secara acak dan mengelompokkan data tersebut sebanyak nilai K k-fold. Kemudian salah satu kelompok k-fold tersebut akan dijadikan sebagai data uji sedangkan sisa kelompok yang lain akan dijadikan sebagai data latih (Cahyanti, Rahmayani, and Husniar 2020). Alur kerja cross-validation diilustrasikan pada Gambar 2.3.

Gambar 2.3 Simulasi K-Fold Cross Validation

Sumber: (Cahyanti, Rahmayani, and Husniar 2020)

2.3.10 *Confusion Matrix*

Confusion matrix merupakan perhitungan yang digunakan untuk mempresentasikan tingkat akurasi dari suatu klasifikasi. Dasar perhitungan *confusion matrix* dengan membandingkan total data yang diklasifikasikan pada

kelas yang bener dengan total seluruh data yang ada pada *matrix* (Sartika & Saluza, 2022).

Tabel 2.1 Confusion Matrix

		Prediction Class	
		Positive	*Negative*
Actual Class	*Positive*	*True Positive (TP)*	*False Negative (FN)*
	Negative	*False Positive (FP)*	*True Negative (TN)*

Dimana:

TN : Model memprediksi data ada di kelas negatif dan yang sebenarnya data memang ada di kelas negatif

TP : Model memprediksi data ada di kelas positif dan yang sebenarnya data memang ada di kelas positif

FN : Model memprediksi data ada di kelas negatif dan yang sebenarnya data memang ada di kelas positif

FP : Model memprediksi data ada di kelas positif dan yang sebenarnya data memang ada di kelas negatif.

$$Accuracy = \frac{TP+TN}{TP+TN+FN+FP} \times 100 \quad (2.2)$$

Nilai *precision, recall* dan f1-*score* digunakan untuk melihat kinerja dari model klasifikasi yang terbentuk. Precision adalah perbandingan antara jumla prediksi data kelas positif yang merupakan kelas positif dengan banyak data

yang diprediksi positif. Sedangkan *recall* adalah perbandingan antara jumlah prediksi data kelas positif yang merupakan kelas positif dengan banyaknya data yang sebenarnya positif (Sartika & Saluza, 2022).

$$precision = \frac{TP}{TP+FP} \quad (2.3)$$

$$recall = \frac{TP}{TP+FN} \quad (2.4)$$

F1-*Score* merupakan *harmonic mean* dari *precision* dan *recall*. Nilai terbaik F1-*Score* adalah 1.0 sedangkan nilai terburuknya adalah 0 (Sartika & Saluza, 2022).

$$F1 - Score = \frac{2 \times \text{Precision x Recall}}{\text{Precision+Recall}} \quad (2.5)$$

BAB III
METODE PENELITIAN

3.1 Jenis, Sifat dan Pendekatan Penelitian

Jenis penelitian yang digunakan adalah penelitian eksperimen. Penelitian eksperimen merupakan penelitian yang dilakukan dengan menerapkan serangkaian tahapan untuk membuktikan suatu konsep (Kurniasari, Kusrini, & Fatta, 2021). Dalam penelitian ini eksperimen yang dilakukan dengan mengumpulkan data yang bersifat publik dari UCI *Machine Learning Repository* kemudian dilakukan klasifikasi menggunakan algoritma *K-Nearest Neighbor* (KNN) dan *Random Forest*. Penelitian yang dilakukan bersifat deskriptif, karena untuk menggambarkan nasabah yang lancar dalam penerimaan kredit bank. Penelitian deskriptif, dimana data dijelaskan dalam bentuk angka dan tabel atau diagram. Setelah data diolah akan dilakukan analisis dengan pendekatan kuantitatif yang dijelaskan dengan hasil perhitungan angka dan tabel atau diagram.

3.2 Metode Pengumpulan Data

Metode penumpulan data merupakan cara yang digunakan untuk mengumpulkan data yang diperlukan dalam penelitian. Berikut metode penelitian yang digunakan yaitu:

1. Studi Literatur

   Studi literatur dilakukan untuk mengumpulkan data berupa informasi atau teori-teori yang berkaitan dengan penelitian ini yang diperoleh melalui buku, jurnal maupun internet. Teori yang digunakan adalah bank, kredit komparasi,

data mining, klasifikasi, *preprocessing, random forest, K-Nearest Neighbor, K-fold cross* validation, dan confusion *matrix*.

2. Pengumpulan Dataset

Pengumpulan dataset yang dilakukan dari UCI *Machine Learning Repository* bersifat terbuka dan banyak digunakan penelitian data mining. Dataset yang digunakan dalam studi ini berasal dari penelitian yang dilakukan oleh Yeh dan Lien (2016), dimana dataset ini terdiri dari 29.998 entri data, dibagi menjadi kelas "Dibayarkan" (6.636 data) dan "Tidak Dibayarkan" (23.364 data), dengan 25 fitur yang mencakup berbagai jenis tipe data, termasuk numerik dan kategorik. Dataset pembayaran kredit bank yang digunakan dalam penelitian ini dapat dilihat pada Tabel 3.1.

Tabel 3.1 Dataset Pembayaran Kredit Bank

	X1	X2	X3	X4	...	X22	X23	Y
ID	**LIMIT_BAL**	**SEX**	**EDUCATION**	**MARRIAGE**		**PAY_AMT5**	**PAY_AMT6**	**default payment next month**
1	20000	2	2	1	...	0	0	1
2	120000	2	2	2	...	0	2000	1
3	90000	2	2	2	...	1000	5000	0
4	50000	2	2	1	...	1069	1000	0
5	50000	1	2	1	...	689	679	0
...	...	...	...	...	...	...	...	...
29996	220000	1	3	1	...	5000	1000	0
29997	150000	1	3	2	...	0	0	0
29998	30000	1	2	2	...	2000	3100	1

Table 3.1 (lanjutan)

29999	80000	1	3	1	...	52964	1804	1
30000	50000	1	2	1	...	1000	1000	1

Berdasarkan Tabel 3.1 metadata yang ada pada *dataset* yang digunakan dapat dilihat pada tabel 3.2.

Tabel 3.2 Metadata Dataset

Feature		**Tipe Data**	**Keterangan**
X1	LIMIT_BAL	Numerik	Total Kredit yang diberikan
X2	SEX	Kategorik	Jenis Kelamin Customer
X3	EDUCATION	Kategorik	Jenjang Pendidikan
X4	MARRIAGE	Kategorik	Status
X5	AGE	Numerik	Umur (Tahun)
X6	PAY_0	Kategorik	Riwayat Pembayaran Semptember 2005
X7	PAY_2	Kategorik	Riwayat Pembayaran Agustus 2005
X8	PAY_3	Kategorik	Riwayat Pembayaran Juli 2005
X9	PAY_4	Kategorik	Riwayat Pembayaran Juni 2005
X10	PAY_5	Kategorik	Riwayat Pembayaran Mei 2005
X11	PAY_6	Kategorik	Riwayat Pembayaran April 2005
X12	BILL_AMT1	Numerik	Jumlah Pembayaran Semptember 2005
X13	BILL_AMT2	Numerik	Jumlah Pembayaran Agustus 2005
X14	BILL_AMT3	Numerik	Jumlah Pembayaran Juli 2005

Table 3.2 (lanjutan)

X15	BILL_AMT4	Numerik	Jumlah Pembayaran Juni 2005
X16	BILL_AMT5	Numerik	Jumlah Pembayaran Mei 2005
X17	BILL_AMT6	Numerik	Jumlah Pembayaran April 2005
X18	PAY_AMT1	Numerik	Jumlah yang sudah dibayar Semptember 2005
X19	PAY_AMT2	Numerik	Jumlah yang sudah dibayar Agustus 2005
X20	PAY_AMT3	Numerik	Jumlah yang sudah dibayar Juli 2005
X21	PAY_AMT4	Numerik	Jumlah yang sudah dibayar Juni 2005
X22	PAY_AMT5	Numerik	Jumlah yang sudah dibayar Mei 2005
X23	PAY_AMT6	Numerik	Jumlah yang sudah dibayar April 2005
Y	default payment next month	Kategorik	Keterangan Dibayar atau tidak

3.3 Metode Analisis Data

Analisis data yang dilakukan dalam penelitian ini menggunakan teknik data mining dengan menerapkan algoritma *K-Nearest Neighbor* (KNN) dan *Random Forest* untuk dilakukan komparasi akurasi dalam klasifikasi kelayakan penerimaan kredit bank. Dalam penelitian ini dataset yang digunakan data pembayaran kredit dengan rentang waktu tertentu yang diambil dari UCI *Machine Learning Repository* yang terdiri dari 29.998 entri data, dibagi menjadi kelas "Dibayarkan" (6.636 data)

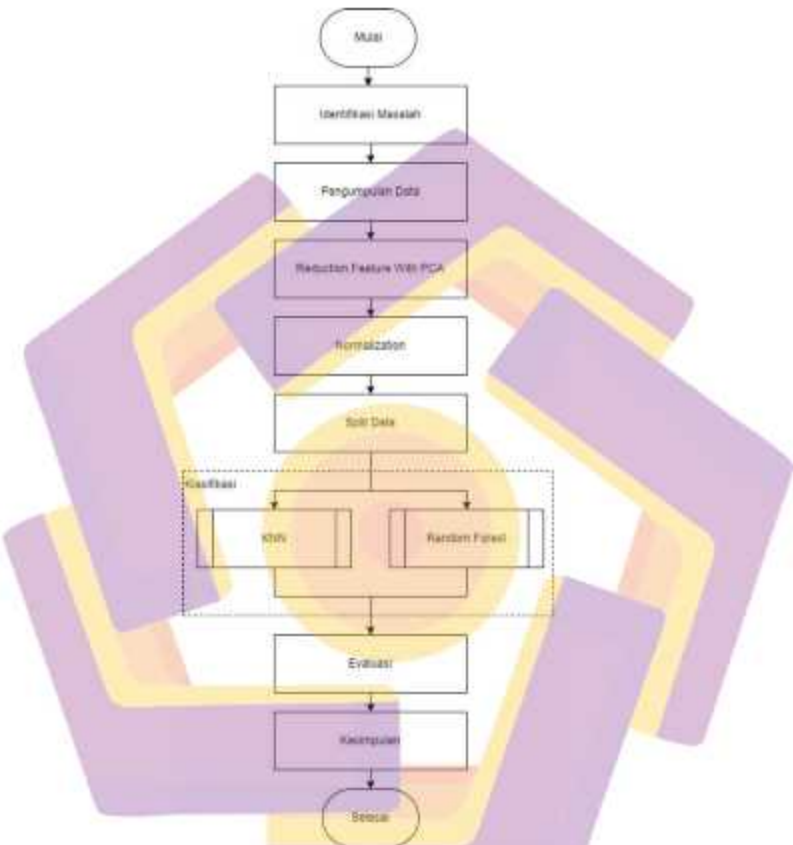
dan "Tidak Dibayarkan" (23.364 data), dengan 25 fitur yang mencakup berbagai jenis tipe data, termasuk numerik dan kategorik.

Setelah data berhasil dikumpulkan maka tahap berikutnya adalah melakukan data *reduction,* dimana pada tahap bertujuan untuk melakukan pemilihan fitur yg optimal sehingga beberapa atribut yang tidak digunakan akan dihapus dan tersisa atribut yang berpengaruh saja. Tahap berikutnya adalah melakukan proses normalisasi melibatkan pengubah-an nilai variabel agar sesuai dengan suatu rentang tertentu. Setelah itu dilakukan pembagian dataset adalah proses membagi data menjadi 2 kelompok yaitu data *training* dan data *testing.*

Pada tahap selanjutnya adalah melakukan klasifikasi menggunakan algoritma *K-Nearest Neighbor* (KNN) dan *Random Forest.* Pada tahap ini model diimplementasikan kedalam bahasa pemrograman Python dengan *tools* Google Colab untuk melakukan klasifikasi kelayakan penerimaan kredit bank. Pada tahap *evaluation* merupakan tahap pengukuran terhadap model klasifikasi dengan melakukan uji akurasi menggunakan *confusion matrix* terhadap algoritma *K-Nearest Neighbor* (KNN) dan *Random Forest.* Dari hasil *confusion matrix* dapat diketahui algoritma mana yang menghasilkan akurasi terbaik pada data pembayaran kredit.

3.4 Alur Penelitian

Alur penelitian merupakan tahapan yang digunakan untuk mencapai tujuan penelitian. Alur penelitian yang digunakan dapat dilihat pada Gambar 3.1.



Gambar 3.1 Alur Penelitian

Dari Gambar 3.1, alur penelitian dapat dijelaskan bahwa terdapat beberapa tahapan yang dilakukan dalam penelitian ini yaitu:

1. Identifikasi Masalah

Tahap identifikasi masalah merupakan tahap mengidentifikasi permasalahan yang terjadi, dalam penelitian ini permasalahan terkait metode KNN yang memperoleh akurasi yang cenderung rendah dibandingkan metode klasifikasi yang lainnya, hal ini seperti hasil penelitian yang dilakukan oleh (Ginting, Lydia, & Zamzami, 2021) dengan mendapatkan akurasi KNN lebih rendah yaitu sebesar 66,35%. Selain itu, hal ini juga dibuktikan oleh penelitian yang dilakukan oleh (Erdiansyah, Lubis, & Erwansyah, 2022) dengan menghasilkan akurasi KNN lebih rendah dibanding dengan akurasi *random forest* yaitu tingkat akurasi KNN sebesar 76.78% dan *random forest* sebesar 86,56% pada klasifikasi pengobatan kutil. Oleh karena itu, dalam penelitian ini akan melakukan percobaan menggunakan algoritma KNN dan *Random Forest* dalam mengklasifikasi kelayakan penerimaan kredit bank untuk mengetahui performa dari *Random Forest* dan KNN dengan dataset yang digunakan dari UCI *Machine Learning Repository*.

2. Pengumpulan Data

Pengumpulan data merupakan proses mengumpulkan data yang digunakan dalam penelitian dengan melakukan studi literatur untuk mendapatkan teori, penelitian terdauhulu sebagai bahan referensi dalam penelitian ini. Kemudian, pengumpulan dataset yang diambil dari UCI *Machine Learning Repository* terkait data penerimaan kredit dengan rentang waktu tertentu.

3. ***Reduction Feature***

Pada *feature reduction* akan menghasilkan *feature* yang digunakan akan berkurang dengan penggabungan. Contoh terdapat *total feature* adalah 10 menjadi 4 atau 5. Pada tahap ini *Principal Component Analysis* (PCA) digunakan untuk melakukan pemilihan *feature*. Penerapan PCA pada klasifikasi kelayakan pemberian kredit dapat membantu mengatasi masalah *overfitting*, serta mengoptimalkan model dengan mempertahankan informasi dan mereduksi data, sehingga dapat meningkatkan efisiensi dan akurasi dalam memprediksi kelayakan pemberian kredit.

4. ***Normalization***

Pada tahap ini dilakukan normalisasi, normalisasi melibatkan pengubah-an nilai variabel agar sesuai dengan suatu rentang tertentu. Salah satu metode normalisasi yang umum digunakan adalah *min-max scaling*, dimana setiap nilai variabel diubah agar berada dalam kisaran 0 hingga 1. Selain itu, normalisasi juga berperan dalam menyamakan dampak dari setiap faktor dalam proses pengambilan keputusan kredit. Dengan nilai-nilai yang telah dinormalisasi, lembaga keuangan dapat memberikan bobot yang seimbang pada setiap variabel dalam evaluasi kelayakan kredit.

5. **Klasifikasi**

Pada tahap klasifikasi dilakukan dengan menggunakan metode *Random Forest* dan KNN untuk melakukan perbandingan. Dalam penelitian ini diimplementasikan kedalam bahasa pemrograman Python dengan *tools Google Colab*.

6. **Evaluasi**

Pada tahap evaluasi dilakukan dengan menggunakan *confusion matrix* dan model K-*Fold Cross Validation*. *Confusion matrix* akan menghasilkan nilai *accuracy, precision, recall* dan f1-*score* dari algoritma *Random Forest* dan KNN, sedangkan model K-*Fold Cross Validation* membantu menghindari overfitting yatiu kondisi dimana model terlalu menghafal data pelatihan dan tidak dapat menggeneralisasi dengan baik ke dtata baru. Serta hasil dari K-*Fold Cross Validation* memberikan perkiraan yang stabil mengenai kinerja model dibandingkan dengan menggunakan data uji saja.

7. **Kesimpulan**

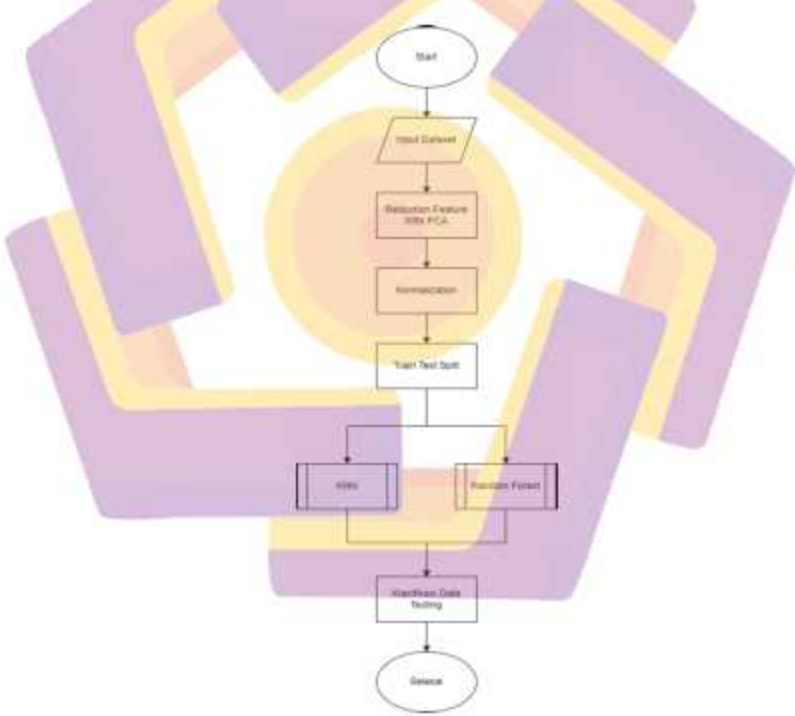
Pada tahap ini dilakukan penarikan kesimpulan dari hasil keseluruhan penelitian dapat diketahui hasil yang diperoleh dari kedua buah metode.

BAB IV

HASIL PENELITIAN DAN PEMBAHASAN

4.1 Perhitungan Manual

Perhitungan manual merupakan perhitungan secara manual yang dilakukan terhadap kelayakan penerimaan kredit bank menggunakan algoritma *Random Forest* dan K-*Nearest Neighbor* (KNN). Adapun flowchart dari perhitungan manual dapat dilihat pada Gambar 4.1.



Gambar 4.1 Flowchart Perhitungan Manual

Berdasarkan Gambar 4.1 dapat dijelaskan langkah-langkah perhitungan dalam klasifikasi pemberian kelayakan kredit bank sebagai berikut:

4.1.1 Input Dataset

Dataset yang digunakan adalah 5 sampel dengan terdiri dari 25 fitur.

Tabel 4.1 Dataset

X1	X2	X3	X4	X5	X6		X25
1	20000	2	2	1	24		1
2	120000	2	2	2	26		1
3	90000	2	2	2	34		0
4	50000	2	2	1	37		0
5	50000	1	2	1	57		0

Keterangan nama fitur pada tabel dapat dilihat di bawah ini.

Tabel 4.2 Keterangan Nama fitur

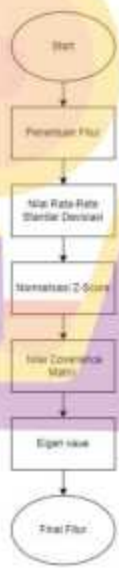
	Keterangan
X1	ID
X2	LIMIT_BAL
X3	SEX
X4	EDUCATION
X5	MARRIAGE
X6	AGE
X7	PAY_0
X8	PAY_2
X9	PAY_3
X10	PAY_4
X11	PAY_5
X12	PAY_6
X13	BILL_AMT1
X14	BILL_AMT2
X15	BILL_AMT3
X16	BILL_AMT4
X17	BILL_AMT5
X18	BILL_AMT6
X19	PAY_AMT1
X20	PAY_AMT2

Table 4.2 (lanjutan)

X21	PAY_AMT3
X22	PAY_AMT4
X23	PAY_AMT5
X24	PAY_AMT6
X25	default payment next month

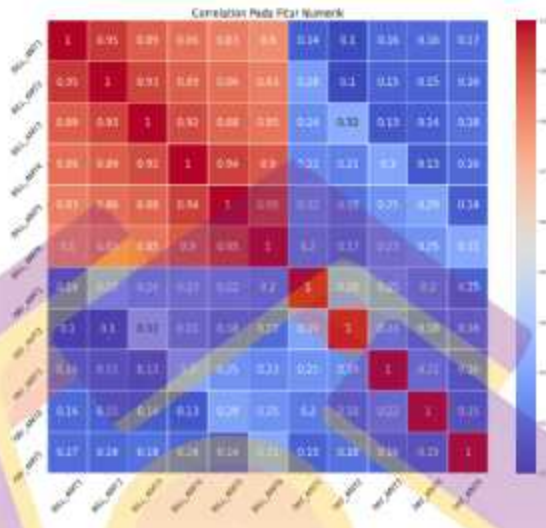
4.1.2 Reduction Feature PCA

PCA merupakan suatu teknik dalam statistika yang digunakan untuk mengurangi dimensi data dengan tetap mempertahankan sebanyak mungkin informasi yang terkandung dalam dataset. PCA dapat diaplikasikan pada kelayakan penerimaan kredit bank menggunakan algoritma Random Forest dan K-Nearest Neighbor (KNN). Untuk memahami langkah-langkah perhitungan menggunakan PCA, dapat dilihat pada Gambar 4.2.



Gambar 4.2 Flowchart PCA

Pada PCA, terlebih dahulu melihat korelasi antar feature numerik. Berikut korelasi yang dihasilkan.



Gambar 4.3 Matrik Korelasi Antar Fitur

Dapat dilihat pada warna merah, menandakan bahwasanya terjadi korelasi yang kuat antar fitur . Pada perhitungan PCA ini akan melakukan reduction dari 2 feature BILL_AMT1, BILL_AMT2 dengan 2 komponen. Artinya hasil akhir PCA terdiri dari 2 komponen vector yang dimana setiap vector adalah hasil gabungan dari kedua fitur tersebut. Untuk menghitung pertama-tama menghitung nilai rata-rata dan standard deviasi dari tiap fitur.

Tabel 4.3 Dataset kedua fitur

No	BILL_AMT1	BILL_AMT2
1	3913	3102
2	2682	1725

Table 4.3 (lanjutan)

3	29239	14027
4	46990	48233
5	8617	5670

Tabel 4.4 Nilai rata-rata dan standar deviasi tiap fitur

	BILL_AMT1	BILL_AMT2
MEAN	18288.2	14551.4
STD	17247.87121	17373.59915

Setelah itu dilakukan scalar menggunakan formula berikut (Ghoneim, 2022)

$$X_{new} = \frac{x - \mu}{\sigma}$$

Keterangan:

X = Data sebelum scalar

μ = Rat-rata

σ = Standard Deviasi

Sehingga menghasilkan sebagai berikut.

$$X_{AMT1} = \frac{3913 - 18288.2}{17247.87} = -0.833$$

$$X_{AMT2} = \frac{3102 - 14551.4}{17373.59} = -0.659$$

Tabel 4.5 Hasil Scalar

No	BILL_AMT1	BILL_AMT2
1	-0.833	-0.659
2	-0.905	-0.738
3	0.635	-0.030
4	1.664	1.939
5	-0.561	-0.511

Setelah mendapatkan nilai scalar, maka selanjut dilakukan perhitungan covariance matrix fitur yang digunakan (yaitu 2) menggunakan formula sebagai berikut (Ghoneim, 2022).

$$Cov(x,y) = \frac{\sum(xi - x^{mean}) * (yi - y^{mean})}{N}$$

Keterangan:

Xi = inputan hasil scalar

x^{mean} dan $y^{mean} = rata - rata\ dari\ hasil\ secalar\ (nilainya = 0)$

$N = Banyak\ data$

Sebelum melakukan convariance matrix, pada hasil covariance matrix berlaku jika matrix secara baris memiliki nama yang sama dengan matrix secara kolom maka formula yang digunakan adalah variance dengan formula berikut.

$$Var(x,y) = \frac{\sum(xi - x^{mean})^2}{N}$$

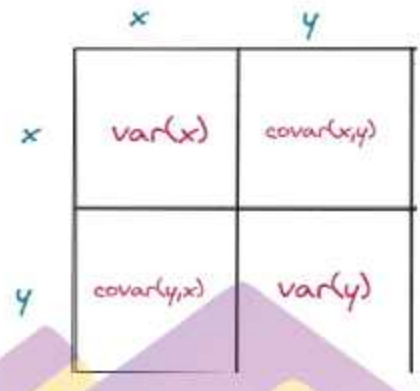
Keterangan:

Xi = inputan hasil scalar

$x^{mean} = rata - rata\ dari\ hasil\ secalar\ (nilainya = 0)$

$N = Banyak\ data$

Gambaran covariance matrix jika berjumlah n_component = 2.



Gambar 4.4 Gambaran Covariance Matrix 2 komponen

Sehingga didapatkan hasil sebagia berikut.

$$Var(BILL\_AMT1, BILL\_AMT1) = \frac{\sum(-0.833-0)^2 + (-0.905-0)^2 + \ldots + (-0.561-0)^2}{5} = 1$$

$$Cov(BILL\_AMT1, BILL\_AMT2) = \frac{\sum((-0.833-0)*(-0.659-0)) + ((-0.905-0)*(-0.738-0)) + \ldots + ((-0.561-0)*(-0.511-0))}{5} =$$

0.9

Tabel 4.6 Hasil Covariance Matrix

	BILL_AMT1	BILL_AMT2
BILL_AMT1	1	0.9
BILL_AMT2	0.9	1

Setelah mendapatkan convariance matrix, selanjutnya menghitung eigen values dan eigen vectors. Untuk mendapatkan hasil eigen values, setiap hasil

variance akan dikurangi dengan lambda (eigen values) dengan cara menghitung determinan nya terlebih dahulu (Ghoneim, 2022).

Tabel 4.7 Convariance matrix untuk determinan

	BILL_AMT1	BILL_AMT2
BILL_AMT1	1-Lambda	0.9
BILL_AMT2	0.9	1-Lambda

Formula determinan untuk ordo matrix 2x2 adalah sebagai berikut (Rodiah & Koswara, 2022).

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} = ad - bc$$

Maka hasil yang didapatkan adalah sebagai berikut.

$$= (1 - Lambda) * (1 - Lambda) - 0.9 * 0.9$$

$$= Lambda^2 - 2\,Lambda + 1 - 0.81$$

$$= Lambda^2 - 2\,Lambda + 0.2$$

Setelah mendapatkan nilai determinan, maka eigen values akan dihitung menggunakan Formula Quadratic hal ini didasari karena hasil ordo matrix memenuhi syarat perhitungan agar dapat menggunakan quadratic formula (Bouwman, Laseroms, & Lord, 2019). Berikut formula yang digunakan (Bouwman, Laseroms, & Lord, 2019).

$$Syarat\ menggunakan\ quadratic\ formula = ax^2 + bx + c$$

$$X = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Sehingga berdasarkan formula di atas, maka hasil determinan dipecah sebagai berikut.

Tabel 4.8 Hasil pemisahan nilai berdasarkan hasil determinan

a	1
b	-2
c	0.2

Sehingga didapatkan hasil sebagai berikut

$$X = \frac{-(-2) \pm \sqrt{(-2)^2 - 4 * 1 * 0.2}}{2 * 1} = 1.9 \; OR \; 0.11$$

Tabel 4.9 Hasil Eigen vaue

	Eigen value 1 (lambda 1	Eigen value 2 (lambda 2)
Result	1.9	0.11

Selanjutnya melakukan perhitungan untuk mendapatkan eigen vector dengan formula sebagai berikut (Ghoneim, 2022).

$$Convaraince\ Matrix * \begin{bmatrix} X \\ Y \end{bmatrix} = Lambda * \begin{bmatrix} X \\ Y \end{bmatrix}$$

Perhitungan pada eigen value 1 untuk mendapatkan eigen vector 1

$$\begin{bmatrix} 1 & 0.9 \\ 0.9 & 1 \end{bmatrix} * \begin{bmatrix} X \\ Y \end{bmatrix} = 1.9 * \begin{bmatrix} X \\ Y \end{bmatrix}$$

$$\begin{bmatrix} X + 0.9Y \\ 0.9X + Y \end{bmatrix} = \begin{bmatrix} 1.9X \\ 1.9Y \end{bmatrix}$$

$$\begin{bmatrix} 0.9Y \\ 0.9X \end{bmatrix} = \begin{bmatrix} 0.9X \\ 0.9Y \end{bmatrix}$$

$$\begin{bmatrix}1\\1\end{bmatrix} = \begin{bmatrix}X\\Y\end{bmatrix}$$

Sehingga nilai eigein vector 1 adalah [1,1]

Perhitungan pada eigen value 2 untuk mendapatkan eigen vector 2

$$\begin{bmatrix}1 & 0.9\\0.9 & 1\end{bmatrix} * \begin{bmatrix}X\\Y\end{bmatrix} = 0.11 * \begin{bmatrix}X\\Y\end{bmatrix}$$

$$\begin{bmatrix}X + 0.9Y\\0.9X + Y\end{bmatrix} = \begin{bmatrix}0.11X\\0.11Y\end{bmatrix}$$

$$\begin{bmatrix}0.9Y\\0.9X\end{bmatrix} = \begin{bmatrix}0.89X\\0.89Y\end{bmatrix}$$

$$\begin{bmatrix}0.1\\1\end{bmatrix} = \begin{bmatrix}X\\Y\end{bmatrix}$$

Sehingga nilai eigein vector 2 adalah [0.1,1]

Terakhir melakukan principal component dan menghitung variance ratio tertinggi pada setiap principal componen dengan formula berikut.

$$Final\ Dataset = X * Eigen\ Vector^T$$

$$Varaince\ Ratio\ (lambda) = \frac{lambda_i}{\sum lambda_i}$$

Keterangan:

X = Inputan data sebelum di scalar.

Sehingga didapatkan hasil sebagai beirkut.

$$Final\ Dataset\ (Eigen\ Vector\ 1) = \begin{matrix} 3913 & 3102 \\ 2682 & 1725 \\ 29239 & 14027 \\ 46990 & 48233 \\ 8617 & 5670 \end{matrix} * \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{matrix} 7015 \\ 4407 \\ 43266 \\ 95223 \\ 14287 \end{matrix}$$

$$Final\ Dataset\ (Eigen\ Vector\ 2) = \begin{matrix} 3913 & 3102 \\ 2682 & 1725 \\ 29239 & 14027 \\ 46990 & 48233 \\ 8617 & 5670 \end{matrix} * \begin{bmatrix} 0.1 \\ 1 \end{bmatrix}$$

$$= \begin{matrix} 3524.32 \\ 2010.45 \\ 17091.17 \\ 53414.33 \\ 6588.4 \end{matrix}$$

$$Varaince\ Ratio\ (lambda, eigen\ vector\ 1) = \frac{1.9}{\sum 1.9 + 0.11}$$

$$= \frac{1.9}{2.1} = 0.94$$

$$Varaince\ Ratio\ (lambda, eigen\ vector\ 2) = \frac{0.11}{\sum 1.9 + 0.11}$$

$$= \frac{0.11}{2.1} = 0.05$$

Tabel 4.10 Hasil PCA

No	PC1	PC2
1	7015	3524.32
2	4407	2010.45
3	43266	17091.17
4	95223	53414.33
5	14287	6588.4
Variance Ratio	0.945273632	0.054726368

Karena varaince ratio tertinggi terletak pada PC1, maka disarankan untuk menggunakan hasil reduction feature di PC1 dan artinya persentasi ratio variance hasil fitur BILL_AMT1 dan BILL_AMT2 memiliki keragaman sekitar 94.5%. Selanjutnya dikarenakan terdapat 6 BILL_AMT, maka akan dilakukan reduction feature dari BILL_AMT1 hingga BILL_AMT6 hal ini dikarenakan korelasi antar feature >= 0.9. Sehingga disini akan menggunakan bantuan python untuk mendapatkan variance ratio mana yang terbaik pada setiap komponennya.

ID	BILL_AMT1	BILL_AMT2	BILL_AMT3	BILL_AMT4	BILL_AMT5	BILL_AMT6
1	3913	3102	689	0	0	0
2	2682	1725	2682	3272	3455	3261
3	29239	14027	13559	14331	14948	15549
4	46990	48233	49291	28314	28959	29547
5	8617	5670	35835	20940	19146	19131

```
pca = PCA(n_components = 2, random_state = False)
pca_result = pca.fit_transform(feature_bill)

pca_result
```

```
array([[-34756.41310311,  -5989.89753181],
       [-31908.3542852 ,  -1500.44358414],
       [  2665.11002736,  -9094.40509898],
       [ 58872.22134829,  -5895.72794025],
       [  5127.47601266,  22450.47415517]])
```

```
pca.explained_variance_ratio_.round(3)
```

```
array([0.878, 0.101])
```

Gambar 4.2 Variance Ratio

Ternyata tingkat variance ratio tertinggi berada pada komponent satu, sehingga n_components = 1 akan digunakan sebagai reduction feature BILL_AMT. Sehingga dataset hasil reduction feature sebagai berikut.

Tabel 4.11 Dataset hasil PCA

X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11	X12	bill	X19	X20	X21	X22	X23	X24	X25
1	20000	2	2	1	24	2	2	-1	-1	-2	-2	-34756.4	0	689	0	0	0	0	1
2	120000	2	2	2	26	-1	2	0	0	0	2	-31908.4	0	1000	1000	1000	0	2000	1
3	90000	2	2	2	34	0	0	0	0	0	0	26651.11	1518	1500	1000	1000	1000	5000	0
4	50000	2	2	1	37	0	0	0	0	0	0	58872.2	2000	2019	1200	1100	1069	1000	0
5	50000	1	2	1	57	-1	0	-1	0	0	0	51274.48	2000	3668.1	1000	900	689	679	0

4.1.3 Split Data

Training dataset (merupakan satu set data yang berisi nilai dari kedua komponen di atas yang digunakan untuk menentukan kelas yang sesuai berdasarkan predictor) dan testing dataset (data baru yang akan diklasifikasikan oleh model yang telah dibuat dan akurasi klasifikasi dievaluasi) (Muningsih, 2022). Dalam membagi train dan test data tidak ada ketentuan rasionnya berapa, namun yang dapat disarankan 70:30, 80:20, 90:10. Berikut porsi data train dan test berdasarkan ratio yang disebutkan sebelumnya jika total datsetnya adalah 5.

Tabel 4.12 Proporsi train dan test

Params	Total Train	Total Test
70:30	3	2
80:20	4	1
90:10	5	0

Tabel 4.13 Keterangan Warna

Train	
Test	

Berikut gambaran split data berdasarkan rasio yang disebutkan sebelumnya.

Tabel 4.14 Split data 70:30

X1	X2	X3	X4	X5	X6		X25
1	20000	2	2	1	24		1
2	120000	2	2	2	26		1
3	90000	2	2	2	34		0
4	50000	2	2	1	37		0
5	50000	1	2	1	57		0

Tabel 4.15 Split data 80:20

X1	X2	X3	X4	X5	X6		X25
1	20000	2	2	1	24		1
2	120000	2	2	2	26		1
3	90000	2	2	2	34		0
4	50000	2	2	1	37		0
5	50000	1	2	1	57		0

Tabel 4.16 Split data 90:10

X1	X2	X3	X4	X5	X6		X25
1	20000	2	2	1	24		1
2	120000	2	2	2	26		1
3	90000	2	2	2	34		0
4	50000	2	2	1	37		0
5	50000	1	2	1	57		0

Pada perhitungan ini akan menggunakan split data dengan banding 90:10. Maka data train memiliki proposi 90% dan da testing 10%.

4.1.4 Normalization

Tahapan normalisasi digunakan untuk merubah range dari 0-1 yang salah satunya dapat menggunakan min-max normalization (Lubis & Kharisudin, 2021). Formula yang digunakan adalah sebagai berikut (Lubis & Kharisudin, 2021).

$$x^{l} = \frac{x - min_x}{max_x - min_x}$$

Keterangan:

X' = Data setelah normalisasi

X = Data Aktual yang akan dinormalisasi

x_max = Data tertinggi dari seluruh data

x_min = Data terendah dari seluruh data

Tahapan normalisasi dilakukan pada numerik yang bersifat numerik kontinus, aritnya numerik yang telah bersifat kategorikal tidak akan dilakukan dikarenakan pada numerik kategorikal memiliki maknanya tersendiri.

Tabel 4.17 Min-Max pada pada fitur numerik kontinus.

Feature	MAX	MIN
LIMIT_BAL	120000	20000
AGE	37	24
PAY_0	2	-1
PAY_2	2	0
PAY_3	0	-1
PAY_4	0	-1
PAY_5	0	-2
PAY_6	2	-2
PAY_AMT1	2000	0
PAY_AMT2	2019	689
PAY_AMT3	1200	0
PAY_AMT4	1100	0
PAY_AMT5	1069	0
PAY_AMT6	5000	0
bill	58872.22	-34756.4

$$x^{1,Limit\_Ball,Train} = \frac{2000 - 20000}{120000 - 20000} = \frac{0}{100000} = 0$$

$$x^{2,Limit\_Ball,Train} = \frac{120000 - 20000}{120000 - 20000} = \frac{100000}{100000} = 1$$

$$x^{1,Limit\_Ball,Test} = \frac{50000 - 20000}{120000 - 20000} = \frac{30000}{100000} = 0.3$$

Tabel 4.18 Hasil Normalization Data Train

X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11	X12	bil1	X19	X20	X21	X22	X23	X24	X25
1	0	2	2	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1
2	1	2	2	2	0.154	0	1	1	1	1	1	0.03	0	0.234	0.833	0.909	0	0.4	1
3	0.7	2	2	2	0.769	0.333	0	1	1	1	0.5	0.4	0.759	0.61	0.833	0.909	0.935	1	0
4	0.3	2	2	1	1	0.333	0	1	1	1	0.5	1	1	1	1	1	1	0.2	0

Tabel 4.19 Normalisasi Data Test

X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11	X12	bil1	X19	X20	X21	X22	X23	X24	X25
5	0.3	1	2	1	2.538	0	0	0	1	1	0.5	0.43	1	27.06	8.333	8.182	0.645	0.1	0

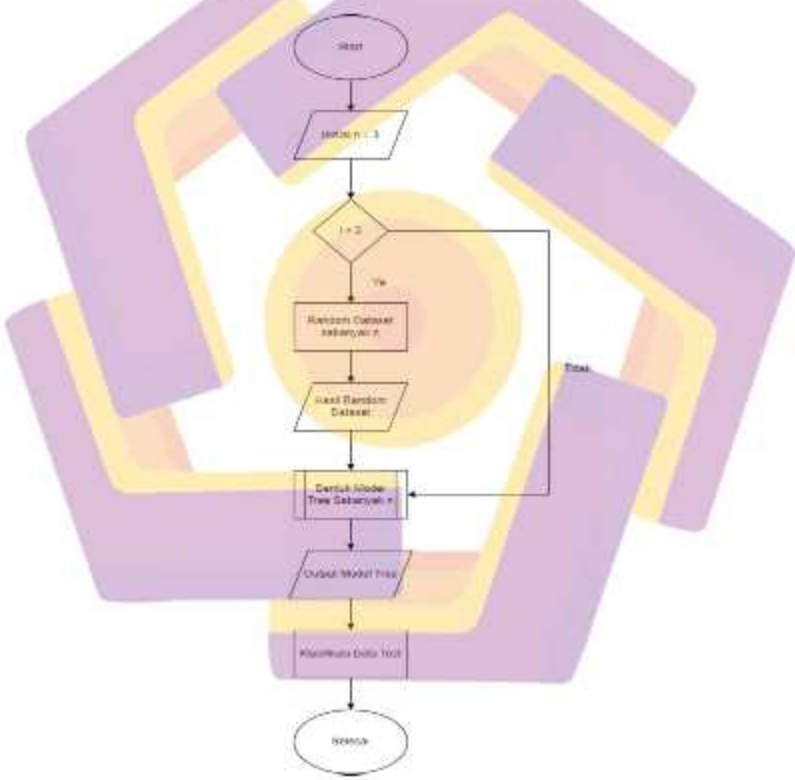
4.1.5 Klasifikasi

Pada tahapan klasifikasi akan menggunakan 2 algoritma yaitu Random Forest dan KNN.

4.1.5.1 Random Forest

Random Forest adalah algoritma ensemble yang menggabungkan beberapa pohon keputusan (decision trees) untuk meningkatkan keakuratannya. Setiap pohon dihasilkan dari subset acak data pelatihan, dan hasil dari beberapa pohon tersebut diambil melalui voting atau averaging, tergantung pada apakah ini adalah masalah klasifikasi atau regresi. Dalam Penelitian terkait pemberian kelayakan

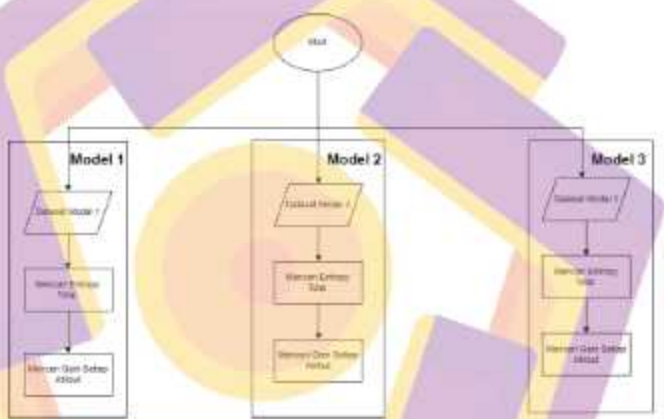
kredit, Random Forest dapat memeriksa berbagai fitur seperti riwayat kredit, pendapatan, tanggungan, dan sebagainya, serta membuat keputusan berdasarkan mayoritas voting dari pohon-pohon keputusan. Ini dapat menangani berbagai tipe data, menanggulangi overfitting, dan bekerja dengan baik untuk dataset yang besar. Pada algoritma *random forest* semakin banyak tree, maka dapat meningkat performa dari random forest tersebut (Haristu, 2019). Sehingga flowchart dari random forest sebagai berikut.



Gambar 4.3 Flowchart Random Forest

1. Inisial N

Dari Gambar 4.5, dapat dijelaskan bahwa *random forest* merupakan sekumpulan *tree*. Jika *random forest* akan membentuk 3 model *tree* dikarenakan N = 3. *(K)* dimana untuk membentuk model tree akan menggunakan formula *decision tree*, maka *random forest* akan melakukan pengacakan (*random*) dataset untuk 3 model dimana setiap model memiliki dataset nya tersendiri. Sehingga *flowchart* yang terbentuk dapat dilihat pada Gambar 4.6.



Gambar 4.4 Flowchart Tree

2. Random Dataset sebanyak N

Pengambilan akan dilakukan secara random dengan kemungkinan terdapat kemunculan data yang pada hasil tiap bootstrap dan bahkan tidak sama sekali. Pengambilan disini akan menggunakan index data untuk memudahakan pembacaan alur.

Tabel 4.20 Hasil Index Random Dataset

No	Dataset Random 1	Dataset Random 2	Dataset Random 3
1	1	1	2
2	2	2	3
3	3	4	4

Tabel 4.21 Dataset Random 1

	X2	X3		X25
1	0	2	...	1
2	1	2		1
3	0.7	2		0

Tabel 4.22 Dataset Random 2

	X2	X3		X25
1	0	2	...	1
2	1	2		1
4	0.3	2		0

Tabel 4.23 Dataset Random 3

	X2	X3		X25
2	1	2	...	1
3	0.7	2		0
4	0.3	2		0

Sebelum melakukan klasifikasi, dataset pada feature independent yang bernilai numerik kontinus akan dirubah menjadi interval untuk mendapatkan kategorinya. Dikarenakan data telah dilakukan min-max normalization yang memiliki range 0 hingga 1, maka interval akan berada pada <= 0.5 dan > 0.5 untuk numerik kontinus, 1 dan 2 untuk numerik kategorikal. Perhitungan Tree akan dilakukan transformasi untuk bertujuan memudahkan pembacaan hasil.

3. Mencari Entropy Total

Nilai Gain adalah *Information Gain* yang digunakan untuk mencari satu variabel/ atribut dari dataset (S) untuk dijadikan *root*/*node* dan *branch node*, yaitu satu atribut yang mempunyai nilai gain tertinggi. Formula untuk menghitung Entropy total adalah sebagai berikut (Atros, Padri, Nurdiawan, Faqih, & Anwar, 2021).

$$Entropy(S) = \sum_{i=1}^{n} - p_i * log_2 pi$$

Keterangan:

S = Jumlah Kasus atau Total Data

P = probabilitas suatu label terhadap jumlah kasus

n = Jumlah partisi

Untuk menghitung *entropy* perlu mengetahui probabilitas label terhadap total kasus. Pada dataset ini memiliki 2 label yaitu 1 dan 0. Untuk total kemunculan kedua label pada total kasus dapat dilihat pada tabel dibawah ini

Tabel 4.24 Proporsi kemunculan kelas pada setiap bootstrap

Model Ke-	1	0
1	2	1
2	2	1
3	1	2

Setelah mengetahui kemunculan setiap label pada datasetnya, maka dapat dihitung Entropy setiap model.

$$Entropy(Model\ 1) = \sum\left(-\frac{2}{3} * log_2\left(\frac{2}{3}\right)\right) + \left(-\frac{1}{3} * log_2\left(\frac{1}{3}\right)\right)$$

$$= 0.918295834$$

$$Entropy(Model\ 2) = \sum\left(-\frac{2}{3} * log_2\left(\frac{2}{3}\right)\right) + \left(-\frac{1}{3} * log_2\left(\frac{1}{3}\right)\right)$$

$$= 0.918295834$$

$$Entropy(Model\ 3) = \sum\left(-\frac{1}{3} * log_2\left(\frac{1}{3}\right)\right) + \left(-\frac{2}{3} * log_2\left(\frac{2}{3}\right)\right)$$

$$= 0.918295834$$

Tabel 4.25 Entropy total setiap model

Model Ke-	Entropy
1	0.918295834
2	0.918295834
3	0.918295834

4. Mencari Gain Setiap Atribut

Gain atau information gain merupakan kriteria yang paling populer untuk pemilihan atribut, information gain dapat dihitung dari output data atau variabel dependen y yang dikelompokkan berdasarkan atribut A, dinotasikan dengan gain (y,A). Gain (y,A) dari atribut A relatif terhadap output data y (Atros, Padri, Nurdiawan, Faqih, & Anwar, 2021). Formula dari information gain adalah sebagai berikut (Atros, Padri, Nurdiawan, Faqih, & Anwar, 2021).

$$Information\ Gain\ = Entropy\ (S) - \sum \frac{Si}{S} * Entropy\ (Si)$$

Keterangan:

S = Himpunan

Si = Probabilitas kriteria tiap fitur terhadap label

S = Jumlah kasus kriteria terkait pada fitur.

Contoh perhitungan Model ke-1.

Untuk mencari gain, diperlukan mencari entropy setiap fitur terhadap setiap kriteria pada fitur tersebut. Contoh pada fitur pada X2 terdapat 2 kriteria < 0.5 dan >= 0.5 maka perlu diketahui total kemunculan setiap kriteria terhadap terhadap label.

Tabel 4.26 Perhitungan Entropy dan Gain pada Feature X2 Pada Model 1

Atribut	Partisi	Total Data	1	0	Entropy	Gain
Total		**3**	**2**	**1**	**0.918295834**	
X2	< 0.5	1	0	1	0	0.251629167
	>= 0.5	2	1	1	1	

Pada nilai 1 pada fitur total data artinya bahwasanya kemunculan nilai < 0.5 pada fitur **X2** dan nilai >= 0.5 adalah 2. Pada label 1 kemunculan kriteria <0.5 pada fitur **X2** adalah 0 sedangkan pada label 0 adalah 1, selanjutnya pada kriteria >=0.5 pada label 1 kemunculannya adalah 1 dan pada label 0 kemunculannya adalah 1. Setelah mengetahui kemunculan tiap kriteria terhadap label, maka dapat dihitung entropy per kriteria pada fitur **X2**

$$Entropy(Model\ 1, X2, < 0.5) = \sum\left(-\frac{0}{1} * log_2\left(\frac{0}{1}\right)\right) + \left(-\frac{1}{1} * log_2\left(\frac{1}{1}\right)\right) = 0$$

$$Entropy(Model\ 1, X2, \geq 0.5) = \sum\left(-\frac{1}{1} * log_2\left(\frac{1}{1}\right)\right) + \left(-\frac{1}{1} * log_2\left(\frac{1}{1}\right)\right) = 1$$

Setelah mendapatkan nilai entropy setiap kriteria pada setiap atribut independent, maka selanjutnya dapat menghitung information gain fitur

terkait dengan formula di atas. Sebelum itu harus menghitung entropy 2 atribut, maksud dua atribut tersebut adalah entropy total dari dari kedua kriteria dengan formula sebagai berikut.

$$Entropy\ (T,x) = \sum_{c \in X} P(c) * E(c) = \sum \frac{Si}{S} * Entropy\ (Si)$$

Keterangan:

T, X = Atribut T dan Atribut X

P(C) = Probabilitas kelas atribut

E(C) = Nilai Entropy Kelas Atribut

S = Himpunan

Si = Probabilitas kriteria tiap fitur terhadap label

S = Jumlah kasus kriteria terkait pada fitur.

Jika diperhatikan formula di atas adalah formula information Gain untuk setiap kriteria pada atribut terkait

$$Entropy\ 2\ Atribut\ (Model\ 1, X2, Kriteria < 0.5\ dan \geq 0.5)$$

$$= \sum \left(-\frac{1}{3} * 0\right) + \left(\frac{2}{3} * 1\right) = 0 + 0.666666667$$

$$= 0.666666667$$

$$Information\ Gain\ (Model\ 1, A1) = \mathbf{0.918295834} - 0.666666667$$

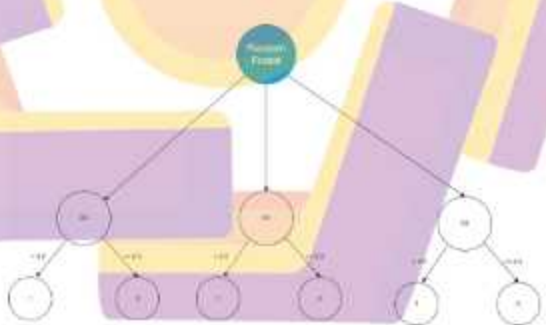
$$= 0.251629167$$

Perhitungan di atas berlaku untuk seluruh model berdasarkan dengan kriteria yang terdapat.

Feature	Gain Model Ke-1	Gain Model Ke-2	Gain Model Ke-3
X2	0.251629	0.251629	0.251629
X3	0	0	0
X4	0	0	0
X5	0.251629	0.251629	0.251629
X6	0.918296	0.918296	0.918296
X7	0.251629	0.251629	0
X8	0.918296	0.918296	0.918296
X9	0.251629	0.251629	0
X10	0.251629	0.251629	0
X11	0.251629	0.251629	0
X12	0.251629	0.251629	0
bill	0	0.918296	0.251629
X19	0.918296	0.918296	0.918296
X20	0.918296	0.918296	0.918296
X21	0.251629	0.251629	0
X22	0.251629	0.251629	0
X23	0.918296	0.918296	0.918296
X24	0.918296	0	0.251629

5. Output

Dari Information Gain di atas didapatkan model tree yang nantinya akan dilakukan penggabungan menjadi sebuah forest. Jika divisualkan menjadi sebagai berikut.



Gambar 4.5 Visual Random Forest

6. Prediksi Data Test

Pada hasil Random Forest diatas, didapatkan bahwasanya setiap model tree memiliki 1 akar dan 2 leaf (daun) pada model tree 1 dan 2 dan 3 dengan setiap model tree memiliki feature root dan leaf yang sama. Untuk menentukan kelas yang akan diprediksi akan menggunakan sistem **majortt vactory** yang dimana jika 2 model memprediksi 1 dan 1 model memprediksi 0 maka hasil prediksi data testing adalah 1 (Haristu, 2019). Maka data tersebut akan diprediksi stanting. Hasil prediksi dari ketiga model pada testing.

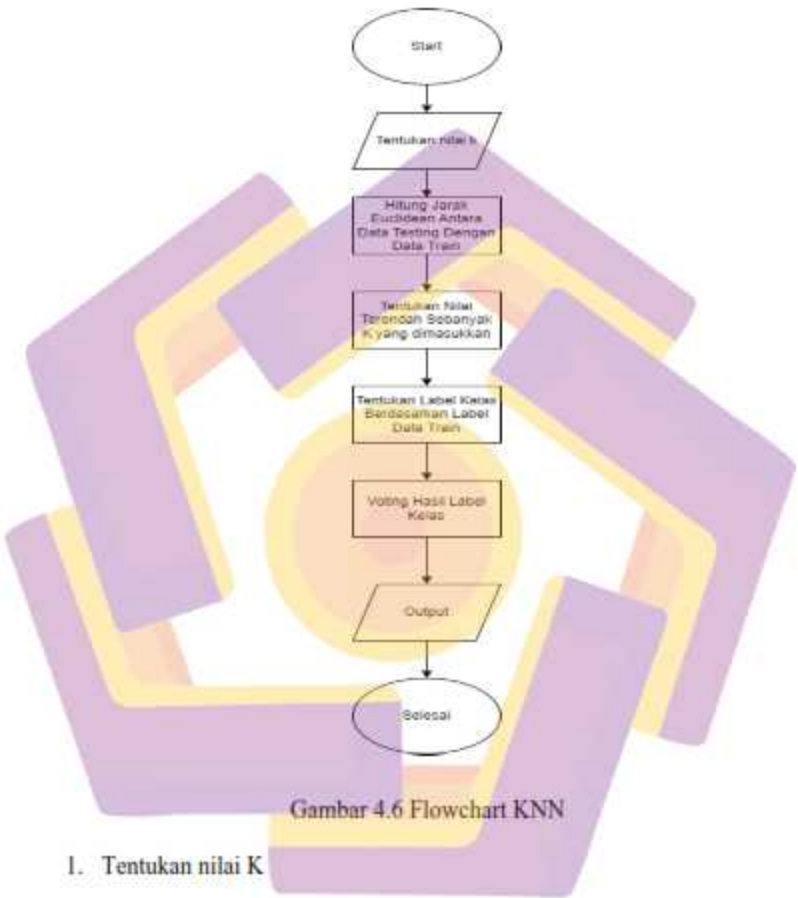
Tabel 4.27 Hasil prediksi data testing

Index Data	Model Tree 1	Model Tree 2	Model Tree 3	Prediksi	Label Sebenarnya
5	0	0	0	0	0

4.1.5.2 *K-Nearest Neighbor* (KNN)

KNN merupakan metode klasifikasi yang sangat sederhana dalam mengklasifikasikan sebuah gambar berdasarkan jarak terdekat dengan tetangganya (Farokhah 2020). Metode ini dideklarasikan sebagai metode pembelajaran yang malas merupakan contoh dari metode belajar malas (Farokhah, 2020). KNN adalah algoritma berbasis instance yang mengklasifikasikan suatu data berdasarkan mayoritas kelas dari k-nearest neighbors dalam ruang fitur. Dalam konteks kelayakan kredit, fitur-fitur seperti skor kredit, jumlah pinjaman, dan riwayat pembayaran dapat membentuk ruang fitur. Ketika data baru masuk, KNN mencari k-nearest neighbors-nya dalam ruang fitur dan menentukan kelas mayoritas dari tetangga-tetangga tersebut. Sebagai contoh, jika sebagian besar

tetangga memiliki rekam jejak pembayaran yang baik, maka data baru kemungkinan besar dianggap kelayakan untuk pemberian kredit. Sehingga flowchart dari KNN adalah sebagai berikut.



Gambar 4.6 Flowchart KNN

1. Tentukan nilai K

   Nilai pada KNN merupakan jumlah tetangga terdekat berdasarkan jaraknya (Farokhah, 2020). Misal jika K = 2, maka akan dilakukan 2 tetangga dengan jarak terdekat. Pada perhitungan ini akan menggunakan nilai K = 2 dan 3.

2. Hitung Jarak

Perhitungan jarak pada KNN dapat dilakukan dengan berbagai macam pendekatan formula salah satunya adalah euclidean distance (Dinata, Akbara, & Hasdyna, 2020).

$$d_i = \sqrt{\sum_{i=1}^{p} (X_2 - X_1)^2}$$

Keterangan

X1 = Data Training

X2 = Data Testing

i = feature

d = Jarak

p = Jumlah Data Training

Sehingga jarak total data training akan dihitung pada data testing menggunakan euclidean distance.

$$d_{train1,test} = \sqrt{\sum_{i=1}^{p} (0-0.3)^2 + (2-1)^2 + (2-2)^2 + .. + (0-0.1358)^2}$$

$$= 29.70048498$$

$$d_{train2,test} = \sqrt{\sum_{i=1}^{p} (1-0.3)^2 + (2-1)^2 + (2-2)^2 + .. + (0.4-0.1358)^2}$$

$$= 28.99900879$$

$$d_{train3,test} = \sqrt{\sum_{i=1}^{p} (0.7 - 0.3)^2 + (2 - 1)^2 + (2 - 2)^2 + .. + (1 - 0.1358)^2}$$

$$= 28.56809505$$

$$d_{train4,test} = \sqrt{\sum_{i=1}^{p} (0.3 - 0.3)^2 + (2 - 1)^2 + (2 - 2)^2 + .. + (0.2 - 0.1358)^2}$$

$$= 28.09811469$$

Tabel 4.28 Hasil Euclidean Distance Terhadap Data Testing

X1	Euclidean Distance
1	29.70048498
2	28.99900879
3	28.56809505
4	28.09811469

3. Tentukan Nilai Terendah Sebanyak K

Pada tahapan ini akan dihitung nilai terendah sebanyak K, pada tahapan sebelumnya telah ditentukan nilai K sebanyak 2 dan 3. Maka akan ditentukan nilai terendah sebanyak 2 dan 3.

Tabel 4.29 Nilai Terendah Sebanyak K

X1	Euclidean Distance	K = 2	K = 3
1	29.70048498	FALSE	FALSE
2	28.99900879	FALSE	TRUE
3	28.56809505	TRUE	TRUE
4	28.09811469	TRUE	TRUE

Nilai False menandakan bahwa nilai tersebut masuk kedalam kategori terendah, dapat dilihat perbedaan pada K = 2 dan K = 3 dimana nilai jarak terdekat memiliki total yang berbeda.

4. Tentukan label

Pada tahapan ini akan melakukan penentuan label kelas berdasarkan data train yang termasuk kedalam jarak terendah. Pada tabel 29, K = 2 yang memiliki jarak terendah adalah data ke-3 dan ke-4 yang jika dilihat pada tabel 16 data ke-3 dan data ke-4 memiliki label 0, maka penentuan label K = 2 akan diberi label 0. Pada K = 3, sedikit berbeda dikarenakan data train ke-2 termasuk kedalam jarak terendah, yang dimana jika dilihat pada tabel 16 termasuk kelas 1 yang artinya pada K = 3 terdapat kelas 1 dan 0. Sehingga menjadi sebagia berikut.

Tabel 4.30 Penetuan label

X1	Euclidean Distance	K = 2	K = 3
1	29.70048498	FALS	FALSE
2	28.99900879	FALSE	1
3	28.56809505	0	0
4	28.09811469	0	0

5. Voting label

Pada tahapan ini akan dilakukan voting berdasarkan kemunculan label pada setiap K, tahapan ini sering disebut sebagia majoritas kelas (Farokhah, 2020). Pada tabel K = 2, dikarenakan kemunculan label hanya terjadi pada label 0 maka dapat dipastikan jika K = 2 hasil prediksi data test adalah label 0. Pada K = 3, terdapat 2 kemunculan label 1 dan 0 dimana proporsi nya 1 dan 2, disinilah voting atau majority akan dilakukan dengan

mengambilkan proporsi tertingi pada label tersebut yang artinya label 0 yang akan dipilih dikarenakan memiliki proporsi tertinggi.

Tabel 4.31 Prediksi Data Testing

X1	K = 2	K = 3
5	0	0

4.1.6 Evaluasi

Untuk menguji peforma model, maka dapat dilakukan evaluasi salah satunya menggunakan confusion matrix. Confusion matrix adalah tabel yang menyatakan klasifikasi jumlah data uji yang benar dan jumlah data uji yang salah (Normawati & Prayogi, 2021).

Tabel 4.32 Confusion Matrix 2 Kelas

		Prediksi	
		0	1
Akutal	0	TP	FP
	1	FN	TN

Pada evaluasi yang digunakan akan menggunakan weighted formula dan k-cross validation. Rumus untuk perhitungan akurasi, presisi, recall dan F1-Score weighted adalah sebagai berikut (Kosasih, Fahrurozi, & Riminarsih, 2022) (Berrar, 2018).

$$Akurasi = \frac{(TP + TN)}{(TP + FP + FN + TN)} x\ 100\%$$

$$Precision = \frac{TP}{(TP+FP)} \text{ x } 100\%$$

$$Recall = \frac{TP}{(TP+FN)} \text{ x } 100\%$$

$$F1 - Score = 2\frac{Presisi * Recall}{Presisi + Recall} * 100\%$$

$$WAP = \frac{\sum p_i d_i}{\sum d_i}$$

$$WAR = \frac{\sum r_i d_i}{\sum d_i}$$

$$WAF = \frac{\sum f_i d_i}{\sum d_i}$$

$$Cross\ Validation = \frac{1}{k} \sum_{i=2\_k}^{k} Akurasi_i$$

Keterangan:

WAP = Weight Average Precision

WAR = Weight Average Recall

WAF = Weight Average F1-Score

d = Total data setiap kelas

p = presisi

r = recall

f = F1-Score

k = Nilai k yang digunakan

Sebelum itu perlu diketahui total data dari tiap kelas.

Tabel 4.33 Total Data Tiap Kelas

	Total Data
0	2360
1	640

4.1.6.1 Without PCA

Dan berikut hasil Confusion Matrix untuk KNN dan Random Forest

Tabel 4.34 Confusion Matrix KNN

		Prediksi	
		0	1
Akutal	0	2226	134
	1	430	210

Tabel 4.35 Confusion Matrix Random Forest

		Prediksi	
		0	1
Akutal	0	2247	113
	1	407	233

Berikut salah satu contoh perhitungan weight average (algoritma KNN)

$$Akurasi = \frac{2226 + 210}{2226 + 430 + 134 + 210} x\ 100\% = 81.2\ \%$$

$$Presisi = \frac{\frac{2226}{(2226 + 430)} \text{ x } 100\% * 2360 + \frac{210}{(210 + 134)} \text{ x } 100\% * 640}{2360 + 640} = 78.95\ \%$$

$$Recall = \frac{\frac{2226}{(2226 + 134)} \text{ x } 100\% * 2360 + \frac{210}{(210 + 430)} \text{ x } 100\% * 640}{2360 + 640} = 81.2\ \%$$

$$F1 - Score$$

$$= \frac{\frac{2 * \frac{2226}{(2226 + 430)} * \frac{2226}{(2226 + 134)}}{\frac{2226}{(2226 + 430)} + \frac{2226}{(2226 + 134)}} \text{ x } 100\% * 2360 + \frac{2 * \frac{210}{(210 + 134)} * \frac{210}{(210 + 430)}}{\frac{210}{(210 + 134)} + \frac{210}{(210 + 430)}} \text{ x } 100\% * 640}{2360 + 640}$$

$$= 78.93\%$$

Tabel 4.36 Hasil evaluasi dari KNN

Params	Secara Weight Avg
Akurasi	81.20%
Presisi	78.95%
Recall	81.2%
F1-Score	78.93%

Tabel 4.37 Hasil Evaluasi dari Random Forest

Params	Secara Weight Avg
Akurasi	82.67%
Presisi	80.97%
Recall	82.67%
F1-Score	80.59%

Selanjutnya menghitung dari sisis evaluasi K-Fold Cross Validation, nilai fold yang digunakan adalah 10.

Tabel 4.38 Hasil K-Fold Validation Pada Algoritma KNN dan Random Forest

Fold	Akurasi KNN	Akurasi Random Forest
2	77.023%	82.007%
3	77.093%	82.027%
4	77.277%	82.160%
5	77.273%	82.057%
6	77.327%	82.110%
7	77.333%	82.113%
8	77.350%	82.093%
9	77.303%	82.113%

Salah satu contoh perhitungan terjadi pada K = 2 di kedua algoritma, untuk mendaptkan akurasi dibutuhkan sebuah nilai confusion matrix. Sehingga didapatkan hasil sebagai berikut.

Tabel 4.39 Fold K = 2 Pada KNN

		Prediksi	
		0	1
Akutal	0	22635	729
	1	6164	472

Tabel 4.40 Fold K = 2 Pada Random Forest

		Prediksi	
		0	1
Akutal	0	22206	1158
	1	4240	2396

Selanjutnya untuk mendapatkan nilai akurasi pada fold, menggunakan formula akurasi yang dijelaskan sebelumnya. Sehingga didapatkan hasil sebagai berikut.

$$Akurasi\ (Fold = 2, KNN) = \frac{22635 + 472}{22635 + 472 + 6164 + 729}\ x\ 100\%$$

$$= 77.023\ \%$$

$$Akurasi\ (Fold = 2, Random\ Forest)$$

$$= \frac{22206 + 2396}{22206 + 2396 + 4240 + 1158}\ x\ 100\% = 82.007\ \%$$

Setelah mendapatkan nilai akurasi setiap fold, maka dihitung rata-rata nya. Sehingga didapatkan hasil sebagai berikut.

$$Cross\ Validation\ (KNN)$$

$$= \frac{1}{9} \sum_{i=2..k}^{k} 77.023 + 77.093 + .. + 77.303 = 77.2475\ \%$$

$$Cross\ Validation\ (Random\ Forest)$$

$$= \frac{1}{9} \sum_{i=2..k}^{k} 82.007 + 82.027 + .. + 82.113\% = 82.0850\%$$

Berikut tabel perbandingannya dari sisi akurasi.

Tabel 4.41 Hasil perbandingan berdasarkan 2 evaluasi (Akurasi)

Algoritma	Weighted	Cross Validation
KNN	81.20%	77.2475%
Random Forest	82.67%	82.0850%

4.1.6.2 With PCA

Pada evaluasi penggunaan PCA dilakukan dengan menggunakan formula yang sama didapatkan hasil sebagai berikut.

Tabel 4.42 Hasil Evaluasi KNN Dengan PCA

Params	**Secara Weight Avg**
Akurasi	82.10%
Presisi	80.19%
Recall	82.1%
F1-Score	79.72%

Tabel 4.43 Hasil Random Forest Dengan PCA

Params	Secara Weight Avg
Akurasi	82.63%
Presisi	80.92%
Recall	82.63%
F1-Score	80.60%

Tabel 4.44 Hasil KNN dan Random Forest pada K-Fold Dengan PCA

Fold	Akurasi KNN	Akurasi Random Forest
2	77.023%	82.00%
3	77.093%	82.04%
4	77.277%	97.66%
5	77.273%	82.03%
6	77.327%	82.12%
7	77.333%	82.03%
8	77.350%	82.16%
9	77.303%	82.09%
Average	**77.2475%**	**84.02%**

Sehingga Jika Dibandingkan Hasil Tanpa Menggunakan PCA Dan Tanpa PCA, Didapatkan Hasil Sebagai Berikut.

Tabel 4.45 Hasil Perbandingan (Akurasi)

Algoritma	*Weighted* (Akurasi) Tanpa PCA	*Cross Validation* Tanpa PCA	*Weighted* (Akurasi) Dengan PCA	*Cross Validation* Dengan PCA
KNN	81.20%	77.2475%	82.10%	77.2475%
Random Forest	82.67%	82.0850%	82.63%	84.02%

4.2 Implementasi Program

Dalam implementasi kedalam bahasa pemrograman Python terdapat *library* yang digunakan yaitu numpy, seaborn, matplotlib, PCA, Min Max Scaler, KNeighborsClassifier, RandomForestClassifier, accuracy_score, confusion_matrix, precision_score, recall_score dan f1_score. *Source code import library* dapat dilihat pada Gambar 4.1.

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
from sklearn.preprocessing import MinMaxScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import KFold, cross_val_score
from sklearn.metrics import accuracy_score, confusion_matrix, precision_score, recall_score, f1_score
```

Gambar 4.7 Source Code Import Library

Berdasarkan Gambar 4.9 maka penjelasan setiap library yang digunakan adalah sebagai berikut:

1. Numpy: digunakan untuk proses aljabar linear.
2. Pandas: digunakan untuk pembersihan data, manipulasi data hingga analisis data.
3. Seaborn: digunakan untuk membuat visualisasi berupa grafik.
4. matplotlib.pyplot: digunakan untuk visualisasi berupa plot .
5. PCA: digunakan untuk mengekstraksi fitur yang penting.
6. MinMaxScaler: digunakan untuk normalisasi data.
7. KNeigborsClassifier: digunakan untuk membuat model K-Nearest Neigbors.
8. RandomForestClassifier: digunakan untuk membuat model Random Forest.
9. accuracy_score, confusion_matrix, precision_score, recall_score, f1_score: digunakan untuk menghitung nilai *accuracy, precision, recall* dan f1-*score*

4.2.1 Input Dataset

Setelah melakukan *import library*, dilakukan pembacaan dataset dengan membuat fungsi untuk pembacaan dataset. Adapun source code yang digunakan dalam melakukan pembacaan dataset dapat dilihat pada Gambar 4.10

```
data = pd.read_excel('default of credit card clients (1).xls', skiprows=[0])
data
```

Gambar 4.8 Source Code Pembacaan Datasets

Berdasarkan source code pada Gambar 4.10 maka diperoleh hasil yang dapat dilihat pada Gambar 4.11.

Gambar 4.9 Hasil Pembacaan Datasets

4.2.2 Redution Feature PCA

Reduction Feature merupakan tahapan untuk menyeleksi dataset yang digunakan dengan memilih data yang digunakan dan atribut yang digunakan dengan menggunakan PCA. PCA merupakan proses analisis dalam melakukan reduksi data dengan memilih variabel-variabel yang mampu menjelaskan sebagian besar variabel data. Proses Redution Feature PCA dapat dilihat sebagai berikut:

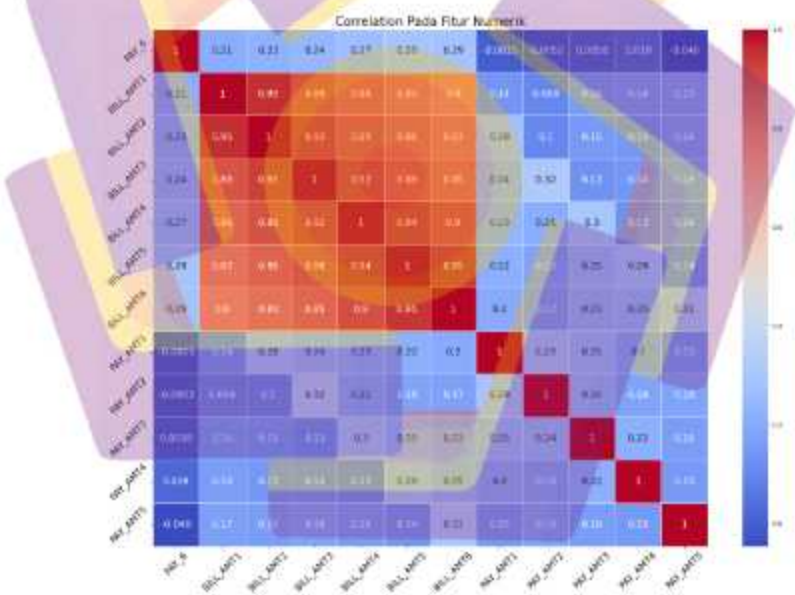
Pada PCA, terlebih dahulu melihat korelasi antar fitur numerik. Berikut source code yang digunakan untuk melihat *corelation* pada fitur numerik dapat dilihat pada Gambar 4.12.

```
correlation_matrix = data.iloc[:,10:22].corr()

plt.figure(figsize = (20,15))
sns.heatmap(correlation_matrix, annot = True, annot_kws = {"fontsize" : 15}, cmap = 'coolwarm', linewidths = 0.5)
plt.title('Correlation Pada Fitur Numerik', size = 30)
plt.xticks(fontsize = 15, rotation = 45)
plt.yticks(fontsize = 15, rotation = 45)
plt.savefig('matrik_korelasi.png')
plt.show()
```

Gambar 4.10 *Source Code Corelation* Pada fitur Numerik

Berdasarkan Gambar 4.12 maka diperoleh hasil yang dapat dilihat pada Gambar 4.13.



Gambar 4.11 Hasil *Corelation* Pada fitur Numerik

Berdasarkan Gambar 4.13 diperoleh hasil *corelation* pada fitur numerik pada warna merah, menandakan bahwa terjadi korelasi yang kuat antar fitur

BILL_AMT1, BILL_AMT2, BILL_AMT3, BILL_AMT4, BILL_AMT5, BILL_AMT6. Tedapat korelasi yang cukup tinggi di angka >80% hal ini menjadikan BILL_AMT1, BILL_AMT2, BILL_AMT3, BILL_AMT4, BILL_AMT5, BILL_AMT6 dapat di lakukan PCA. Pada perhitungan PCA ini akan melakukan reduction dari 2 fitur BILL_AMT1, BILL_AMT2 dengan 2 komponen. Artinya hasil akhir PCA terdiri dari 2 komponen vector yang dimana setiap vector adalah hasil gabungan dari kedua fitur tersebut. Selanjutnya adalah menampilkan data *feature bill* dengan menggunakan source code yang terdapat pada Gambar 4.14

```
feature_bill = data[['BILL_AMT1','BILL_AMT2','BILL_AMT3','BILL_AMT4','BILL_AMT5','BILL_AMT6']]
feature_bill
```

Gambar 4.12 *Source Code* Menampilkan *Feature Bill*

Berdasarkan Gambar 4.14 maka diperoleh hasil yang dapat dilihat pada Gambar 4.15.

	BILL_AMT1	BILL_AMT2	BILL_AMT3	BILL_AMT4	BILL_AMT5	BILL_AMT6
0	3913	3102	689	0	0	0
1	2682	1725	2682	3272	3455	3261
2	29239	14027	13559	14331	14948	15549
3	46990	48233	49291	28314	28959	29547
4	8617	5670	35835	20940	19146	19131
...	...	...	...	...	...	...
29995	188948	192815	208365	88004	31237	15980
29996	1683	1828	3502	8979	5190	0
29997	3565	3356	2758	20878	20582	19357
29998	-1645	78379	76304	52774	11855	48944
29999	47929	48905	49764	36535	32428	15313

30000 rows × 6 columns

Gambar 4.13 Hasil Tampilan *Feature* Bill

Selanjutnya melakukan perhitungan terhadap PCA dengan menggunakan *source code* yang dapat dilihat pada Gambar 4.16.

```
pca = PCA(n_components = 2, random_state = False)
pca_result = pca.fit_transform(feature_bill)

pca_result
```

Gambar 4.14 *Source Code* perhitungan PCA

Berdasarkan Gambar 4.16 maka diperoleh hasil yang dapat dilihat pada Gambar 4.17.

```
array([[-107289.37843059,    1830.69802138],
       [-103860.70264958,   -3334.48840244],
       [ -68874.7384692 ,    2603.12751175],
       ...,
       [ -83686.97948351,  -22831.58637919],
       [  -1362.84136818,  -10490.95354934],
       [ -14325.10729244,   15638.85020861]])
```

Gambar 4.15 Hasil Perhitungan PCA

Berdasarkan Gambar 4.17 diperoleh bahwa pada PCA terdapat 1 komponent utama yaiut n-component yang dimana component tersebut akan berhubungan dengan *eigen value* jika n-component = 2 maka terdapat 2 *eigen value*. Pada contoh diatas PCA akan berusaha melakukan penggabungan bill_amt 1 hingga 6 dengan n-component = 2. Artinya terdapat 2 *variance* hasil penggabungan 6 feature tersebut.

Tahap berikutnya adalah mendapatnya nilai variance dengan menggunakan *source code* yang dapat dilihat pada Gambar 4.17.

```
[ ] pca.explained_variance_ratio_.round(3)

    array([0.907, 0.05 ])
```

Gambar 4.16 *Source Code* Mendapatkan Nilai *Variance*

Pada gambar 4.18 digunakan untuk mengetahui kualitas PCA yang baik dengan nilai variance > 80%. Artinya dari 2 eigen value tersebut tersebut, yang mana variance nya > 80% dan ternyata bahwasanya eigen value ke-1 yang memiliki variance > 80% yaitu sebesar 0.907 atau 90.7%. Sehingga dapat disimpulkan bahwa PCA ke-1 yang balik baik untuk diambil hasil PCA. Setelah diperoleh hasil tersebut maka tahap berikutnya adalah menampilkan hasil perhitugan PCA yang telah dilakukan dengan menggunakan source code pada Gambar 4.19.

```
data_new = data.copy()
data_new = data_new.drop(columns=['BILL_AMT1','BILL_AMT2', 'BILL_AMT3','BILL_AMT4', 'BILL_AMT5','BILL_AMT6'])
data_new['Bill'] = bill_feature
data_new
```

Gambar 4.17 *Source Code* Menampilkan Hasil PCA

Berdasarkan Gambar 4.19 maka diperoleh hasil yang dapat dilihat pada Gambar 4.20.

Gambar 4.18 *Source Code* Hasil PCA

4.2.3 Normalisasi Data

Normalisasi data merupakan proses penyeragaman nilai yang memiliki *range* 0 dan 1 jika menggunakan *min max normalization*. *Source code* proses *min max normalization* dapat dilihat pada Gambar 4.21.

```
# Aplikasikan min max normalization
scaler = MinMaxScaler()
X_norm = scaler.fit_transform(X) # menerapkan normalisasi atribut agar memiliki skala yg sebanding
print('shape dari data x hasil normalisasi :', X_norm.shape)
```

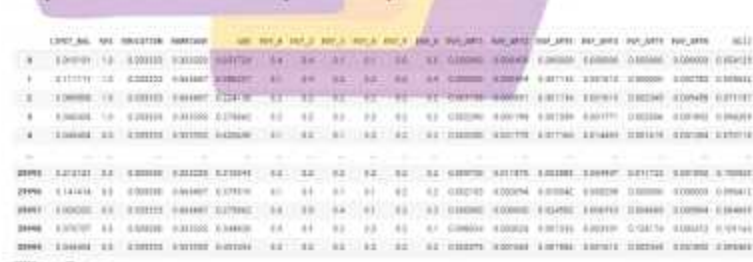
Gambar 4.19 *Source Code Min Max Normalization*

Pada Gambar 4.21 terdapat variabel scaler yang memanggil *library* MinMaxScaler untuk menggunakan *library* tersebut. Kemudian, dilakukan penerapan normalisasi atribut agar mempunyai skala yang sebanding. Hasil normalisasi yaitu terdapat 30000 baris dan 20 kolom. Hasil normalisasi data menggunakan MinMaxScaler ditampilkan dalam bentuk tabel menggunakan perintah pada Gambar 4.22.

```
# menampilkan hasil normalisasi data menggunakan MinMaxScaler
df_normalisasi = pd.DataFrame(X_norm)
df_normalisasi.columns = X.columns
df_normalisasi
```

Gambar 4.20 *Source Code* Menampilkan Hasil Normalisasi

Berdasarkan source code pada Gambar 4.22 maka diperoleh hasil tampilan data normalisasi dapat dilihat pada Gambar 4.23.



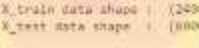
Gambar 4.21 Hasil *Min Max Scaler*

4.2.4 *Split* Data

Pada tahap ini dilakukan *split* data yaitu proses membagi data menjadi 2 kelompok yaitu data *train* dan data *test*. Data *training* digunakan untuk pelatihan model dan data *test* digunakan untuk pengujian model. Perintah melakukan *split* data dapat dilihat pada Gambar 4.24.

Gambar 4.22 *Source Code Split* Data

Pembagian dataset menggunakan perbandingan 80% data *train* dan 20% data *test* dengan jumlah masing-masing data yaitu data *train* 24000 dan data *test* 6000. Hasil *split* data dapat dilihat pada Gambar 4.25.



Gambar 4.23 Hasil *Split* Data

4.2.5 Klasifikasi

Tahap ini merupakan proses pembentukan model untuk klasifikasi menggunakan algoritma *Random Forest* da *K-Nearest Neighbor* (KNN). Pada tahap ini model diimplementasikan kedalam bahasa pemrograman Python dengan *tools* Google Colab untuk melakukan klasifikasi kelayakan penerimaan kredit bank.

4.2.5.1 Algoritma Random Forest

Algoritma *Random Forest* merupakan algoritma yang dilakukan untuk klasifikasi kelayakan penerimaan kredit bank menggunakan dataset yang telah dilakukan *preparation*. Perintah untuk melakukan pembentukan model klasifiaksi *Random Forest* dapat dilihat pada Gambar 4.26. Pada tahap ini memanggil *library* RandomForestClassifier untuk pembentukan model.

Gambar 4.24 *Source Code* Pembentukan Algoritma *Random Forest Without feature selection*

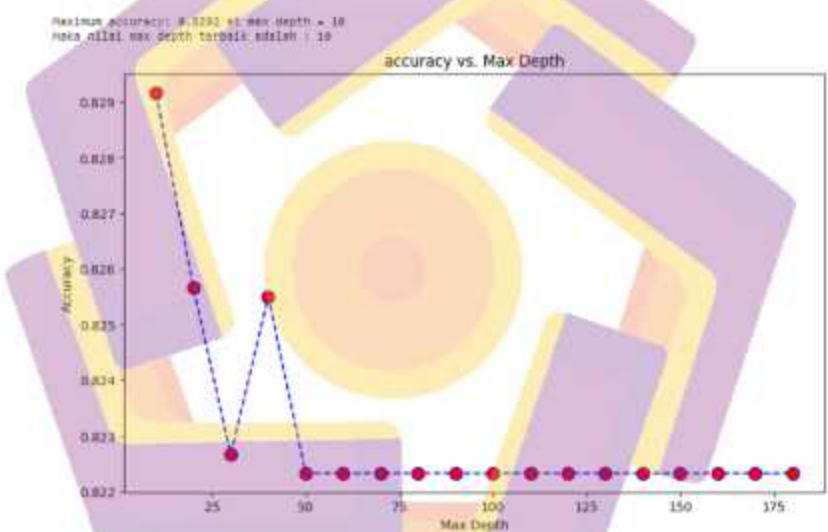
Pada Gambar 4.27 terdapat variabel untuk menyimpan nilai *error rate*, nilai *random state*, label nilai prediksi menggunakan *Random Forest*, hasil akurasi dan rentang nilai *max_depth* yang digunakan yaitu antara rentang 10-180 dengan kelipatan 10. Kemudian, melakukan perulangan untuk *max_depth* sebanyak 18. Selanjutnya, menyiapkan data untuk melatih model pada data x_train dan y_train menggunakan perintah rf.fit(x_train, y_train). Setelah model *Random Forest* dilatih, maka dilakukan prediksi menggunakan data *test* dengan perintah pred = rf.predict(X_test). Dilanjutkan dengan melakukan perhitungan nilai *error rate* dan akurasi pada data *test*.

Pada tahap selanjutnya dilakukan perhitungan nilai *error rate maximum* dan *max depth* menggunakan perintah pada Gambar 4.27.

```
plt.figure(figsize=(10,6))
plt.plot(max_depth_range, acc_rf, color = 'blue', linestyle='dashed', marker='o', markerfacecolor='red', markersize=10)
plt.title('accuracy vs. Max Depth')
plt.xlabel('Max Depth')
plt.ylabel('Accuracy')
print("Maximum accuracy:",round(max(acc_rf), 4),"at max depth =", int(max_depth_range[acc_rf.index(max(acc_rf))]))
print('Nilai nilai max depth terbaik adalah :', int(max_depth_range[acc_rf.index(max(acc_rf))]))
```

Gambar 4.25 *Source Code* Menampilkan Hasil *Maximum Error* dan Nilai *Max Depth* Pada *Random Forest*

Berdasakan source code pada gambar 4.28 maka diperoleh hasil yang dapat dilihat pada Gambar 4.21.



Gambar 4.26 Hasil *Minimum Error Rate* dan Nilai *Random State* Pada *Random Forest*

Dalam algoritma *Random Forest* memilih nilai *random state* adalah langkah awal untuk melatih model. Dari Gambar 4.28, nilai *max depth* = 10 menunjukkan tingkat kesalahan yang relatif lebih tinggi dengan nilai *error rate* yaitu 0.8292. Jadi, untuk model ini menggunakan *max depth* = 10, Selanjutnya masukkan jalur

dataset pelatihan dalam model *Random Forest* untuk melatih model dan menghitung jarak dengan sendirinya.

Sedangkan menggunakan *with feature selection* dalam klasifikasi pemberian kriedit menggunakan *Random Forest*. Perintah untuk melakukan pembentukan model klasifikasi *Random Forest with feature selection* dapat dilihat pada Gambar 4.29.

Gambar 4.27 *Source Code* Pembentukan Algoritma *Random Forest With feature Selection*

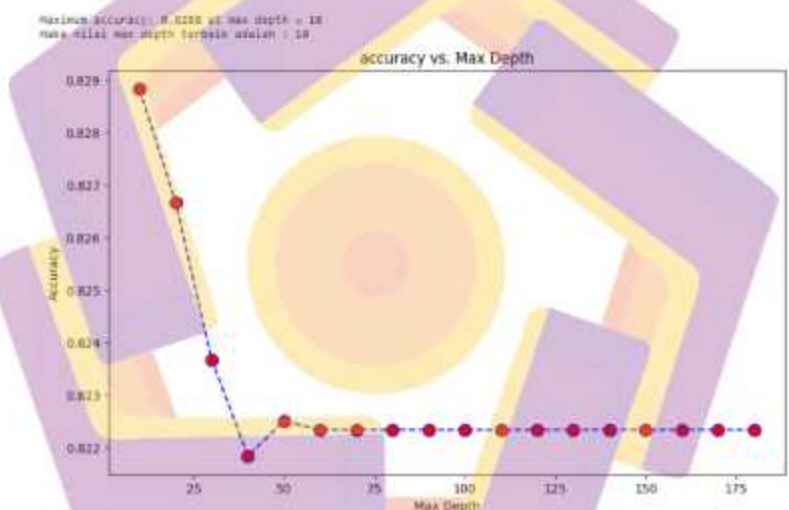
Pada Gambar 4.29 terdapat variabel untuk menyimpan nilai *error rate*, nilai *random state*, label nilai prediksi menggunakan *Random Forest*, hasil akurasi dan rentang nilai *max_depth* yang digunakan yaitu antara rentang 10-180 dengan kelipatan 10. Kemudian, melakukan perulangan untuk *max_depth* sebanyak 18. Selanjutnya, menyiapkan data untuk melatih model pada data x_train dan y_train menggunakan perintah rf.fit(x_train, y_train). Setelah model *Random Forest* dilatih, maka dilakukan prediksi menggunakan data *test* dengan perintah pred = rf.predict(X_test). Dilanjutkan dengan melakukan perhitungan nilai *error rate* dan akurasi pada data *test*.

Pada tahap selanjutnya dilakukan perhitungan nilai *error rate minimum* dan *random state* menggunakan perintah pada Gambar 4.29.

```
plt.figure(figsize=(10,6))
plt.plot(max_depth_range, acc_rf_fs, color = 'blue', linestyle='dashed', marker='o', markerfacecolor='red', markersize=10)
plt.title('accuracy vs. Max Depth')
plt.xlabel('Max Depth')
plt.ylabel('accuracy')
print("Maximum accuracy:",round(max(acc_rf_fs), 4),"at max depth =", int(max_depth_range[acc_rf_fs.index(max(acc_rf_fs))]))
print('maka nilai max depth terbaik adalah :', int(max_depth_range[acc_rf_fs.index(max(acc_rf_fs))]))
```

Gambar 4.28 *Source Code* Menampilkan Hasil *Maximum Error* dan Nilai *Max Depth* Pada *Random Forest With feature Selection*

Berdasakan source code pada gambar 4.30 maka diperoleh hasil yang dapat dilihat pada Gambar 4.26.



Gambar 4.29 Hasil *Maximum Error* dan Nilai *Max Depth* Pada *Random Forest With feature Selection*

Dari Gambar 4.31, nilai *max depth* = 10 menunjukkan tingkat kesalahan yang relatif lebih tinggi dengan nilai *error rate* yaitu 0.8288 Jadi, untuk model ini menggunakan *max depth* = 10, Selanjutnya masukkan jalur dataset pelatihan

dalam model *Random Forest* untuk melatih model dan menghitung jarak dengan sendirinya.

4.2.5.2 Algoritma K-Nearest Neighbor

Algoritma *K-Nearest Neighbor* merupakan algoritma yang dilakukan untuk klasifikasi kelayakan penerimaan kredit bank menggunakan dataset yang telah dilakukan *preparation*. Perintah untuk melakukan pembentukan model klasifiaksi KNN *Without feature selection* dapat dilihat pada Gambar 4.32.

Gambar 4.30 *Source Code* Pembentukan Algoritma *K-Nearest Neighbor without feature selection*

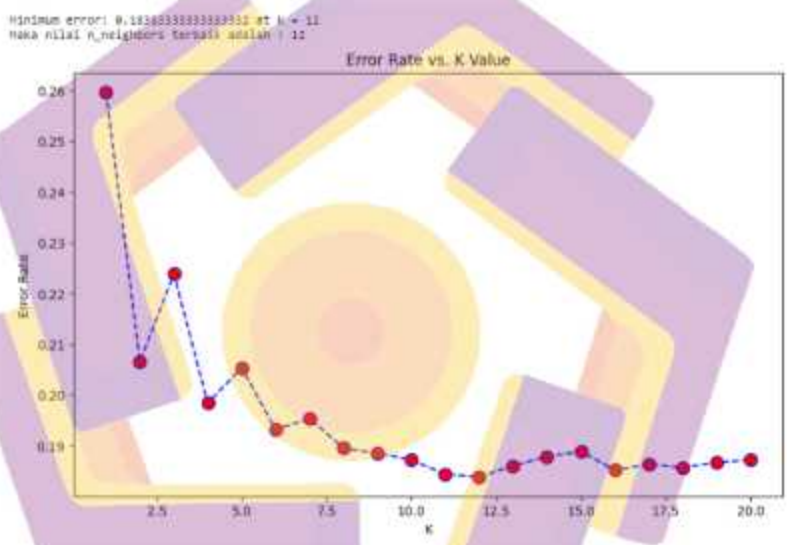
Pada Gambar 4.32 terdapat variabel untuk menyimpan nilai error rate, nilai n_neighbors, label hasil prediksi menggunakan KNN, hasil akurasi dan rentang nilai n_neighbors yang digunakan yaitu 1-20. Kemudian, melakukan perulangan untuk range neighbors sebanyak 20. Selanjutnya, menyiapkan data untuk melatih model pada data x_train dan y_train menggunakan perintah knn.fit(x_train, y_train). Setelah model KNN dilatih, maka dilakukan prediksi menggunakan data test dengan perintah pred_i = knn.predict(X_test). Dilanjutkan dengan melakukan *error rate* dan akurasi pada data *test*.

Pada tahap selanjutnya dilakukan perhitungan nilai *error rate minimum* dan k terbaik menggunakan perintah pada Gambar 4.33.

```
plt.figure(figsize=(10,6))
plt.plot(k_range, error_rate_knn, color='blue', linestyle='dashed',
         marker='o' ,markerfacecolor='red', markersize=10)
plt.title('Error Rate vs. K Value')
plt.xlabel('K')
plt.ylabel('Error Rate')
print("Minimum error:",min(error_rate_knn),"at K =",k_value[error_rate_knn.index(min(error_rate_knn))])
print('Maka nilai n_neighbors terbaik adalah :', k_value[error_rate_knn.index(min(error_rate_knn))])
```

Gambar 4.31 *Source Code* Menampilkan Hasil *Minimum Error* dan Nilai n_neighbor Terbaik Pada *K-Nearest Neighbor without feature selection*

Berdasakan source code pada gambar 4.30 maka diperoleh hasil yang dapat dilihat pada Gambar 4.34.



Gambar 4.32 Hasil *Minimum Error Rate* dan Nilai K Pada *K-Nearest Neighbor without feature selection*

Dalam algoritma KNN memilih nilai "K" adalah langkah awal untuk melatih model. Dari Gambar 4.34, nilai K=12 menunjukkan tingkat kesalahan yang relatif lebih rendah dengan nilai *error rate* yaitu 0.1838. Jadi, untuk model

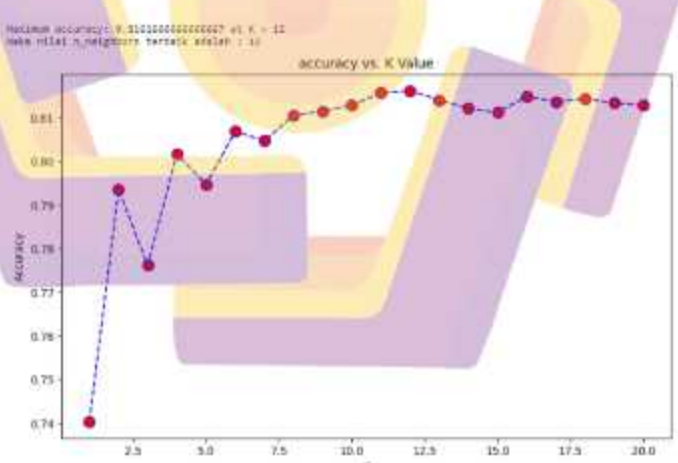
ini menggunakan K=12, Selanjutnya masukkan jalur dataset pelatihan dalam model KNN untuk melatih model dan menghitung jarak dengan sendirinya.

Selanjutnya, menentukan nilai *error rate maximum* pada model KNN menggunakan perintah pada Gambar 4.35.

```
plt.figure(figsize=(10,6))
plt.plot(k_range, acc_knn, color = 'blue', linestyle='dashed',
         marker='o', markerfacecolor='red', markersize=10)
plt.title('accuracy vs. K Value')
plt.xlabel('K')
plt.ylabel('Accuracy')
print("Maximum accuracy:",max(acc_knn),"at K =", k_value[acc_knn.index(max(acc_knn))])
print('Maka nilai n_neighbors terbaik adalah :', k_value[acc_knn.index(max(acc_knn))])
```

Gambar 4.33 *Source Code* Menampilkan Hasil *Maximum Error* dan Nilai n_neighbor Terbaik Pada *K-Nearest Neighbor without feature selection*

Berdasakan *source code* pada gambar 4.35 maka diperoleh hasil yang dapat dilihat pada Gambar 4.36.



Gambar 4.34 Hasil *Maximum Error Rate* dan Nilai K Pada *K-Nearest Neighbor without feature selection*

Dari Gambar 4.36, nilai K=12 menunjukkan tingkat kesalahan yang relatif lebih tinggi dengan nilai *error rate* yaitu 0.81616. Jadi, untuk model ini menggunakan K=12, Selanjutnya masukkan jalur dataset pelatihan dalam model KNN untuk melatih model dan menghitung jarak dengan sendirinya

Sedangkan menggunakan *with feature selection* dalam klasifikasi pemberian kriedit menggunakan KNN. Perintah untuk melakukan pembentukan model klasifikasi KNN *with feature selection* dapat dilihat pada Gambar 4.37.

Gambar 4.35 *Source Code* Pembentukan Algoritma *K-Nearest Neighbor with feature selection*

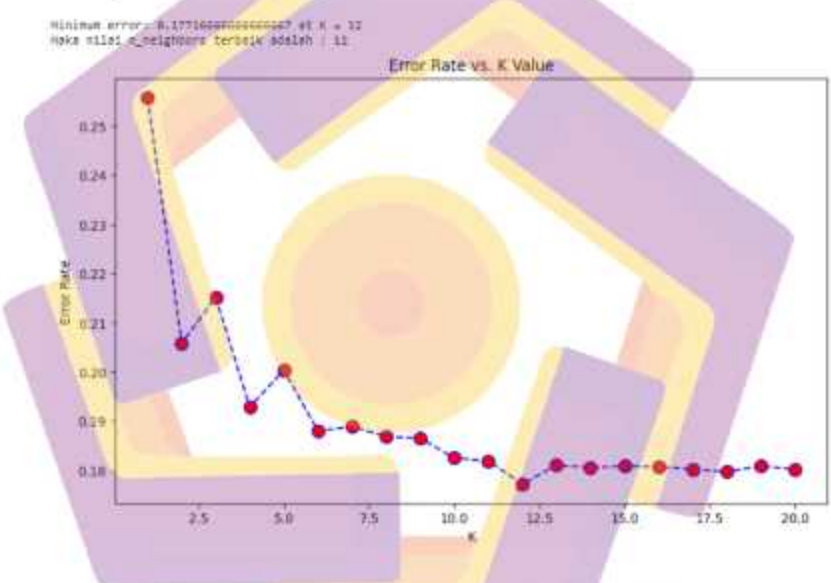
Pada Gambar 4.37 terdapat variabel untuk menyimpan nilai error rate, nilai n_neighbors, label hasil prediksi menggunakan KNN, hasil akurasi dan rentang nilai n_neighbors yang digunakan yaitu 1-20. Kemudian, melakukan perulangan untuk range neighbors sebanyak 20. Selanjutnya, menyiapkan data untuk melatih model pada data x_train dan y_train menggunakan perintah knn.fit(x_train, y_train). Setelah model KNN dilatih, maka dilakukan prediksi menggunakan data test dengan perintah pred_i = knn.predict(X_test). Dilanjutkan dengan melakukan *error rate* dan akurasi pada data *test.*

Pada tahap selanjutnya dilakukan perhitungan nilai *error rate minimum* dan k terbaik menggunakan perintah pada Gambar 4.38.

```
plt.figure(figsize=(10,6))
plt.plot(k_range, error_rate_knn_fs, color='blue', linestyle='dashed',
         marker='o' ,markerfacecolor='red', markersize=10)
plt.title('Error Rate vs. K Value')
plt.xlabel('K')
plt.ylabel('Error Rate')
print("Minimum error:",min(error_rate_knn_fs),"at K =",k_value[error_rate_knn_fs.index(min(error_rate_knn_fs))])
print('Maka nilai n_neighbors terbaik adalah :', k_value[error_rate_knn_fs.index(min(error_rate_knn_fs))])
```

Gambar 4.36 Source Code Menampilkan Hasil Minimum Error dan Nilai n_neighbor Terbaik Pada *K-Nearest Neighbor with feature selection*

Berdasakan source code pada gambar 4.38 maka diperoleh hasil yang dapat dilihat pada Gambar 4.39.



Gambar 4.37 Hasil *Minimum Error Rate* dan Nilai K Pada *K-Nearest Neighbor with feature selection*

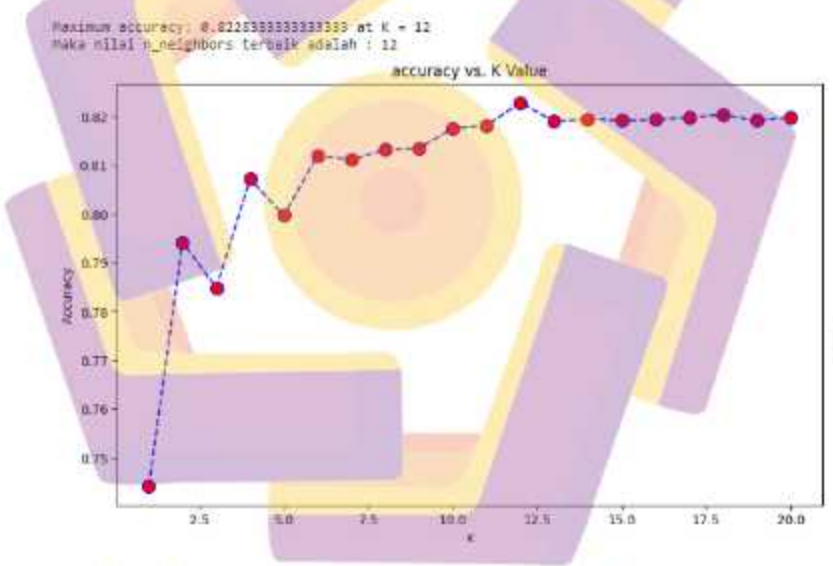
Dalam algoritma KNN memilih nilai "K" adalah langkah awal untuk melatih model. Dari Gambar 4.39, nilai K=12 menunjukkan tingkat kesalahan yang relatif lebih rendah dengan nilai *error rate* yaitu 0.17716666666666667. Jadi, untuk model ini menggunakan K=12, Selanjutnya masukkan jalur dataset

pelatihan dalam model KNN untuk melatih model dan menghitung jarak dengan sendirinya.

Selanjutnya, menentukan nilai *error rate maximum* pada model KNN menggunakan perintah pada Gambar 4.40.

```
plt.figure(figsize=(10,6))
plt.plot(k_range, acc_knn_fs, color = 'blue', linestyle='dashed',
         marker='o', markerfacecolor='red', markersize=10)
plt.title('accuracy vs. K Value')
plt.xlabel('K')
plt.ylabel('Accuracy')
print("Maximum accuracy:",max(acc_knn_fs),"at K =", k_value[acc_knn_fs.index(max(acc_knn_fs))])
print('Maka nilai n_neighbors terbaik adalah :', k_value[acc_knn_fs.index(max(acc_knn_fs))])
```

Gambar 4.38 *Source Code* Menampilkan Hasil *Maximum Error* dan Nilai n_neighbor Terbaik Pada *K-Nearest Neighbor with feature selection*



Gambar 4.39 Hasil *Maximum Error Rate* dan Nilai K Pada *K-Nearest Neighbor with feature selection*

Dari Gambar 4.41, nilai K=12 menunjukkan tingkat kesalahan yang relatif lebih tinggi dengan nilai *error rate* yaitu 0.8228333333333333. Jadi, untuk model

ini menggunakan K=12, Selanjutnya masukkan jalur dataset pelatihan dalam model KNN untuk melatih model dan menghitung jarak dengan sendirinya.

4.2.6 Evaluation

Pada tahap evaluasi merupakan tahap mengevaluasi model yang telah dibentuk untuk mendapatkan tingkat akurasi terbaik dari model yang digunakan. Dalam penelitian ini menggunakan algoritma KNN dan *Random Forest* untuk dilakukan perbandingan hasil perdorma tingkat akurasi pada klasifikasi kelayakan penerimaan kredit bank.

4.2.6.1 Evaluasi Algoritma Random Forest

Pada tahap ini dilakukan evaluasi algoritma *Random Forest* menggunakan *confusion matrix* untuk mengetahui performa tingkat akurasi dari model. Sebelum melakukan evaluasi, pada tahap ini melakukan prediksi menggunakan algoritma KNN dengan data *test* yang telah disiapkan. Kemudian, dilanjutkan proses evaluasi menggunakan *confusion matrix* yang menghasilkan nilai *accuracy, precision, recall* dan f1_*score*. Perintah melakukan prediksi data pada KNN dapat dilihat pada Gambar 4.42.

```
best_prediction_rf = pred_rf[acc_rf.index(max(acc_rf))]  # model terbaik prediksi

Confusion_matrix = confusion_matrix(y_test, best_prediction_rf, labels=[0, 1])
class_label      = [0, 1]
df_confusion     = pd.DataFrame(confusion_matrix, index = class_label, columns = c

sns.heatmap(df_confusion, annot=True, fmt = "d", cmap=plt.cm.Blues)
plt.title('Confusion matrix')
plt.xlabel('Prediction Label')
plt.ylabel('Actual Label')
plt.show()
```

Gambar 4.40 *Source Code* Prediksi Data Pada *Random Forest*

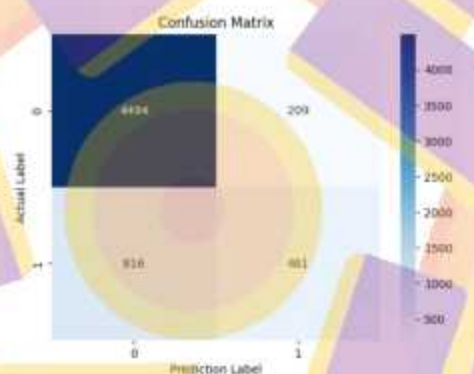
Hasil prediksi data menggunakan algoritma *Random Forest* dapat dilihat pada Gambar 4.42. Dalam penelitian pada algoritma *Random Forest* mendapatkan

nilai *accuracy* sebesar 82.67%, *precision* sebesar 81.05%, *recall* sebesar 82.67% dan f1-*score* sebesar 80.77%.

```
Accuracy  :  82.92 %
Precision :  81.41 %
Recall    :  82.92 %
F1-score  :  80.83 %
```

Gambar 4.41 Hasil Nilai *Confusion Matrix* Pada *Random Forest*

Hasil visualisasi *confusion matrix* pada algoritma *Random Forest* ditampilkan pada Gambar 4.43. Dari grafik tersebut didapatkan informasi *True Positive* (TP), *False Positive* (FP), *False Negative* (FN) dan *True Negative* (TN).



Gambar 4.42 Hasil Visualisasi Metode Random Forest

Berdasarkan Gambar 4.44 dapat dijelaskan bahwa algoritma *Random Forest* dengan diperoleh hasil prediksi pada kelas Layak dengan jumlah data 4703 data dengan label Layak yang diuji, terdapat 4494 data yang terklasifikasi dengan benar dan kesalahan prediksi sebesar 209 data yang masuk kedalam kelas Tidak Layak. Sedangkan, pada kelas Tidak Layak yang diuji dengan total 1297 dengan label Tidak Layak yang diuji, terdapat 481 data sudah terklasifikasi dengan benar

dan kesalahan prediksi sebesar 816 data yang masuk kedalam kelas Layak. Dari nilai *confusion matrix* tersebut didapatkan nilai *accuracy* sebesar 82.92%, *precision* sebesar 81.41%, *recall* sebesar 82.92% dan f1-*score* sebesar 80.83%.

Selain menggunakan evaluasi menggunakan *confusion matrix*, algoritma *Random Forest* juga melakukan evaluasi *K-fold cross validation*. Dimana *K-fold cross* validation ini digunakan untuk menguji kinerja model *Random Forest* dengan membagi data menjadi k subset (fold) yang lebih kecil. Perintah melakukan *k-fold cross validation* pada *Random Forest* dapat dilihat pada Gambar 4.42.

```
# define folds to test
folds = range(2,11)
# record mean accuracy
accuracy_, n_folds = [], []
# evaluate each k value
for k in folds:
    # define the test condition
    cv = KFold(n_splits=k, shuffle=True, random_state=42)
    # get the model
    model = RandomForestClassifier(random_state=random_range[acc_rf.index(max(acc_
    # evaluate the model
    scores = cross_val_score(model, X, y, scoring='accuracy', cv=cv)
    # evaluate k value
    acc = np.mean(scores)
    # report performance
    print('> folds=%d, accuracy=%.5f' % (k, acc))
    # store mean accuracy
    accuracy_.append(acc)
    n_folds.append(k)
```

Gambar 4.43 Source Code K-Fold Cross Validation Pada Random Forest

Hasil *K-fold cross validation Random Forest* ditampilkan pada Gambar 4.46 memproleh hasil sebagai berikut:

Tabel 4.46 Hasil *K-Fold Cross Validation Random Forest Without Feature Selection*

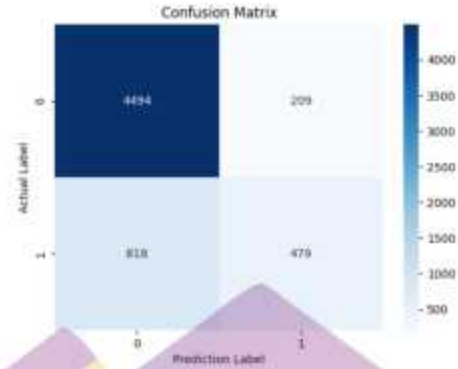
n_fold	Accuracy
2	0.82007
3	0.82027
4	0.82160
5	0.82057
6	0.82110
7	0.82113
8	0.82093
9	0.82113
10	0.82127

Berdasarkan tabel 4.46 dapat disimpulkan bahwa nilai akurasi terbaik *Random Forest* diperoleh pada n_fold = 4 sebesar 82.16%.

```
Hasil K-fold cross validation Random Forest Classifier
Akurasi= 82.16 % pada nilai n_folds = 4
```

Gambar 4.44 Hasil K-Fold Cross Validation Random Forest

Sedangkan untuk hasil visualisasi *confusion matrix* menggunakan *with feature selection* pada algoritma *Random Forest* ditampilkan pada Gambar 4.44. Dari grafik tersebut didapatkan informasi *True Positive* (TP), *False Positive* (FP), *False Negative* (FN) dan *True Negative* (TN).



Gambar 4.45 Hasil Visualisasi Menggunakan With feature selection Metode Random Forest

Berdasarkan Gambar 4.47 dapat dijelaskan bahwa algoritma *Random Forest* dengan diperoleh hasil prediksi pada kelas Layak dengan jumlah data 4703 data dengan label Layak yang diuji, terdapat 4494 data yang terklasifikasi dengan benar dan kesalahan prediksi sebesar 209 data yang masuk kedalam kelas Tidak Layak. Sedangkan, pada kelas Tidak Layak yang diuji dengan total 1297 dengan label Tidak Layak yang diuji, terdapat 479 data sudah terklasifikasi dengan benar dan kesalahan prediksi sebesar 818 data yang masuk kedalam kelas Layak. Dari nilai *confusion matrix* tersebut didapatkan nilai *accuracy* sebesar 82.88%, *precision* sebesar 81.36%, *recall* sebesar 82.88% dan f1-*score* sebesar 80.78%.

Selain menggunakan evaluasi menggunakan *confusion matrix*, algoritma *Random Forest* juga melakukan evaluasi *K-fold cross validation*. Dimana *K-fold cross* validation ini digunakan untuk menguji kinerja model *Random Forest* dengan membagi data menjadi k subset (fold) yang lebih kecil. Perintah

melakukan *k-fold cross validation* pada *Random forest* dapat dilihat pada Gambar 4.48.

```
# define folds to test
folds = range(2,11)
# record mean accuracy
accuracy_knn, n_folds_knn = [], []
# evaluate each k value
for k in folds:
    # define the test condition
    cv = KFold(n_splits=k, shuffle=True, random_state=42)
    # get the score
    model = KNeighborsClassifier(n_neighbors=k_value[acc_knn.index(max(acc_knn))])
    # evaluate the model
    scores = cross_val_score(model, X, y, scoring='accuracy', cv=cv)
    # evaluate k value
    acc = np.mean(scores)
    # report performance
    print('> folds=%d, accuracy=%.3f' % (k, acc))
    # store mean accuracy
    accuracy_knn.append(acc)
    n_folds_knn.append(k)
```

Gambar 4.46 Source Code K-Fold Cross Validation Pada Random Forest

Hasil *K-fold cross validation Random Forest* ditampilkan pada Gambar 4.48 Dari hasil evaluasi tersebut memproleh hasil sebagai berikut:

Tabel 4.47 Hasil K-Fold Cross Validation Random Forest

n_fold	Accuracy
2	0.82003
3	0.82037
4	0.82050
5	0.82033
6	0.82123
7	0.82027
8	0.82163
9	0.82087
10	0.81993

Berdasarkan Tabel 4.47 dapat disimpulkan bahwa nilai akurasi terbaik *Random Forest* diperoleh pada n_fold = 8 sebesar 82.16%.

```
Hasil K-fold cross validation Random Forest Classifier menggunakan feature selection
Akurasi= 82.16 % pada nilai n_folds = 8
```

Gambar 4.47 Hasil K-Fold Cross Validation *Random Forest With Feature Selection*

4.2.6.2 Evaluasi Algoritma K-*Nearest Neighbor*

Pada tahap ini dilakukan evaluasi algoritma KNN menggunakan *confusion matrix* untuk mengetahui performa tingkat akurasi dari model. Sebelum melakukan evaluasi, pada tahap ini melakukan prediksi menggunakan algoritma KNN dengan data *test* yang telah disiapkan. Kemudian, dilanjutkan proses evaluasi menggunakan *confusion matrix* yang menghasilkan nilai *accuracy, precision, recall* dan f1_*score*. Perintah melakukan prediksi data pada KNN dapat dilihat pada Gambar 4.51.

```
best_prediction_knn = grid_knn[acc_knn.index(max(acc_knn))].predict(X_test) # Hasil terbaik prediksi label menggunakan KNN

confusion_matrix = confusion_matrix(y_test, best_prediction_knn, labels=[0, 1])
class_label      = [0, 1]
df_confusion     = pd.DataFrame(confusion_matrix, index = class_label, columns = class_label)

sns.heatmap(df_confusion, annot=True, fmt = "d", cmap=plt.cm.Blues)
plt.title('Confusion matrix')
plt.xlabel('Prediction area')
plt.ylabel('Actual area')
plt.show()
```

Gambar 4.48 *Source Code* Prediksi Data Pada KNN

Hasil prediksi data menggunakan algoritma KNN dapat dilihat pada Gambar 4.51. Dalam penelitian pada algoritma KNN mendapatkan nilai *accuracy* sebesar 81.62%, *precision* sebesar 79.6%, *recall* sebesar 81.62% dan f1-*score* sebesar 79.05%.

```
Accuracy  :  81.62 %
Precision :  79.6 %
Recall    :  81.62 %
F1-score  :  79.05 %
```

Gambar 4.49 Hasil Nilai *Confusion Matrix* Pada KNN

Dari nilai yang diperoleh *confusion matrix* untuk algoritma KNN menggunakan *Without feature selection* dilakukan visualisasi dalam bentuk grafik menggunakan perintah pada Gambar 4.53. Dari perintah tersebut akan ditampilkan nilai prediksi dan aktual pada label Layak dan Tidak Layak.

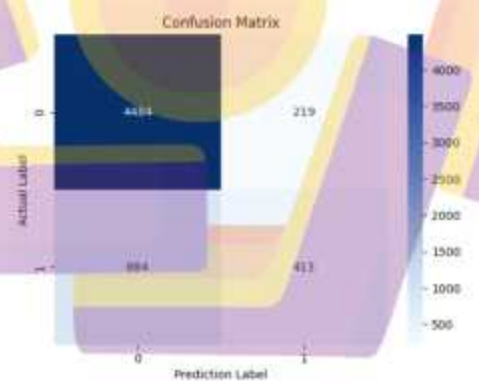
```
best_prediction_knn = pred_knn[acc_knn.index(max(acc_knn))] # untuk terbaik prediksi label menggunakan knn

Confusion_matrix = confusion_matrix(y_test, best_prediction_knn, labels=[0, 1])
class_label      = [0, 1]
DF_confusion     = pd.DataFrame(confusion_matrix, index = class_label, columns = class_label)

sns.heatmap(df_confusion, annot=True, fmt = "d", cmap=plt.cm.Blues)
plt.title('Confusion Matrix')
plt.xlabel('Prediction Label')
plt.ylabel('Actual Label')
plt.show()
```

Gambar 4.50 *Source Code* Visualisasi Hasil Menggunakan *without feature selection* Pada KNN

Hasil visualisasi *confusion matrix* pada algoritma KNN ditampilkan pada Gambar 4.54. Dari grafik tersebut didapatkan informasi *True Positive* (TP), *False Positive* (FP), *False Negative* (FN) dan *True Negative* (TN).



Gambar 4.51 Hasil Visualisasi Menggunakan Without feature selection Pada KNN

Berdasarkan Gambar 4.53 dapat dijelaskan bahwa algoritma KNN dengan diperoleh hasil prediksi pada kelas Layak dengan jumlah data 4703 data dengan label Layak yang diuji, terdapat 4484 data yang terklasifikasi dengan benar dan kesalahan prediksi sebesar 219 data yang masuk kedalam kelas Tidak Layak. Sedangkan, pada kelas Tidak Layak yang diuji dengan total 1297 dengan label Tidak Layak yang diuji, terdapat 413 data sudah terklasifikasi dengan benar dan kesalahan prediksi sebesar 884 data yang masuk kedalam kelas Layak. Dari nilai *confusion matrix* tersebut didapatkan nilai *accuracy* sebesar 81.62%, *precision* sebesar 79.6% *recall* sebesar 81.62% dan f1-*score* sebesar 79.05%.

Selain menggunakan evaluasi menggunakan *confusion matrix*, algoritma KNN juga melakukan evaluasi *K-fold cross validation*. Dimana *K-fold cross* validation ini digunakan untuk menguji kinerja model KNN dengan membagi data menjadi k subset (fold) yang lebih kecil. Perintah melakukan *k-fold cross validation* pada KNN dapat dilihat pada Gambar 4.54.

Gambar 4.52 Source Code K-Fold Cross Validation Menggunakan Without feature selection Pada KNN

Hasil *K-fold cross validation k-nearestt Neighbor classifier* ditampilkan pada Gambar 4.56. Dari hasil evaluasi tersebut memproleh hasil sebagai berikut:

Tabel 4.48 Hasil *K-Fold Cross Validation K-Nearestt Neighbor Classifier*

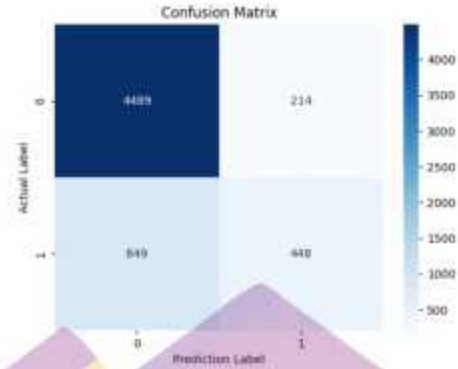
n_fold	***Accuracy***
2	0.77387
3	0.77450
4	0.77657
5	0.77637
6	0.77643
7	0.77717
8	0.77657
9	0.77593
10	0.77603

Berdasarkan Tabel 4.48 dapat disimpulkan bahwa nilai akurasi terbaik *K-Nearestt Neighbor Classifier* diperoleh pada n_fold = 7 sebesar 77.72%.

```
Hasil K-fold cross validation K-Nearest Neighbor Classifier
Akurasi= 77.72 % pada nilai n_folds = 7
```

Gambar 4.53 Hasil K-Fold Cross Validation K-Nearestt Neighbor Classifier

Sedangkan untuk hasil visualisasi *confusion matrix* menggunakan *with feature selection* pada algoritma KNN ditampilkan pada Gambar 4.57 Dari grafik tersebut didapatkan informasi *True Positive* (TP), *False Positive* (FP), *False Negative* (FN) dan *True Negative* (TN).



Gambar 4.54 Hasil Visualisasi Menggunakan With feature selection Metode KNN

Berdasarkan Gambar 4.57 dapat dijelaskan bahwa algoritma KNN dengan diperoleh hasil prediksi pada kelas Layak dengan jumlah data 4703 data dengan label Layak yang diuji, terdapat 4489 data yang terklasifikasi dengan benar dan kesalahan prediksi sebesar 214 data yang masuk kedalam kelas Tidak Layak. Sedangkan, pada kelas Tidak Layak yang diuji dengan total 1297 dengan label Tidak Layak yang diuji, terdapat 448 data sudah terklasifikasi dengan benar dan kesalahan prediksi sebesar 849 data yang masuk kedalam kelas Layak. Dari nilai *confusion matrix* tersebut didapatkan nilai *accuracy* sebesar 82.28%, *precision* sebesar 80.55%, *recall* sebesar 82.28% dan f1-*score* sebesar 79.97%.

Selain menggunakan evaluasi menggunakan *confusion matrix*, algoritma KNN juga melakukan evaluasi *K-fold cross validation*. Dimana *K-fold cross* validation ini digunakan untuk menguji kinerja model KNN dengan membagi data menjadi k subset (fold) yang lebih kecil. Perintah melakukan *k-fold cross validation* pada KNN dapat dilihat pada Gambar 4.58.

```
# record mean accuracy
accuracy_knn_fs, n_folds_knn_fs = [], []
# evaluate each k value
for k in folds:
    # define the test condition
    cv = KFold(n_splits=k, shuffle=True, random_state=42)
    # get the model
    model = KNeighborsClassifier(n_neighbors=k_value[acc_knn_fs.index(max(acc_knn_
    # evaluate the model
    scores = cross_val_score(model, X[feature], y, scoring='accuracy', cv=cv)
    # evaluate k value
    acc = np.mean(scores)
    # report performance
    print('> folds=%d, accuracy=%.5f' % (k, acc))
    # store mean accuracy
    accuracy_knn_fs.append(acc)
    n_folds_knn_fs.append(k)
```

Gambar 4.55 Source Code K-Fold Cross Validation Pada KNN

Hasil *K-fold cross validation k-nearestt Neighbor classifier* ditampilkan pada Gambar 4.52 Dari hasil evaluasi tersebut memproleh hasil sebagai berikut:

Tabel 4.49 Hasil K-Fold Cross Validation K-Nearestt Neighbor Classifier

n_fold	*Accuracy*
2	0.77407
3	0.77463
4	0.77650
5	0.77673
6	0.77653
7	0.77730
8	0.77683
9	0.77600
10	0.77630

Gambar 4.49 dapat disimpulkan bahwa nilai akurasi terbaik *K-Nearestt Neighbor Classifier* diperoleh pada n_fold = 7 sebesar 77.73%.

```
Hasil K-fold cross validation K-Nearest Neighbor Classifier menggunakan feature selection
Akurasie 77.73 % pada nilai n_folds = 7
```

Gambar 4.56 Hasil K-Fold Cross Validation K-Nearestt Neighbor Classifier

Berikut hasil yang diperoleh dari perbandingan Algoritma KNN dengan *Random Forest* baik menggunakan *without feature selection* dan *with feature selection* berdasarkan pembagian data sebagai berikut:

		KNN w/o FS	KNN with FS	Random Forest w/o FS	Random Forest with FS
0	Accuracy	0.814833	0.820083	0.826250	0.827417
1	Precision	0.794156	0.801524	0.809978	0.811598
2	Recall	0.814833	0.820083	0.826250	0.827417
3	F1-score	0.786991	0.796186	0.805023	0.806284
4	Cross validation	0.777866	0.777966	0.821600	0.821633

Gambar 4.57 Hasil Akhir Komprasi KNN dan Random Forest (60:40)

		KNN w/o FS	KNN with FS	Random Forest w/o FS	Random Forest with FS
0	Accuracy	0.813667	0.816222	0.825333	0.824889
1	Precision	0.792215	0.795852	0.808577	0.807946
2	Recall	0.813667	0.816222	0.825333	0.824889
3	F1-score	0.789296	0.792293	0.803928	0.803627
4	Cross validation	0.776400	0.778000	0.821600	0.821633

Gambar 4.58 Hasil Akhir Komprasi KNN dan Random Forest (70:30)

		KNN w/o FS	KNN with FS	Random Forest w/o FS	Random Forest with FS
0	Accuracy	0.816167	0.822833	0.829167	0.828833
1	Precision	0.796013	0.805454	0.814070	0.813630
2	Recall	0.816167	0.822833	0.829167	0.828833
3	F1-score	0.790549	0.799722	0.808251	0.807780
4	Cross validation	0.777166	0.777300	0.821600	0.821633

Gambar 4.59 Hasil Akhir Komprasi KNN dan Random Forest (80:20)

		KNN w/o FS	KNN with FS	Random Forest w/o FS	Random Forest with FS
0	Accuracy	0.812000	0.821000	0.826667	0.826333
1	Precision	0.789540	0.801869	0.809690	0.809225
2	Recall	0.812000	0.821000	0.826667	0.826333
3	F1-score	0.789271	0.797171	0.805907	0.806021
4	Cross validation	0.778500	0.777167	0.821600	0.821633

Gambar 4.60 Hasil Akhir Komprasi KNN dan Random Forest (90:10)

4.3 Perbandingan Hasil Perhitungan Yang Dilakukan dengan Penelitian sebelumnya

Hasil perhitungan yang dilakukan pada penelitian ini memperoleh hasil tingkat akurasi algoritma *Random Forest* dalam mengklasifikasikan kelayakan nilai *accuracy* 82.92%, *precision* sebesar 81.41%, *recall* sebesar 82.92% dan f1-*score* sebesar 80.83%. Sedangkan evaluasi menggunakan *K-fold cross validation* memperoleh nilai akurasi sebesar 82.16% pada nilai n_folds = 4. Sedangkan untuk tingkat akurasi algoritma KNN dalam mengklasifikasikan kelayakan penerimaan kredit bank yaitu mendapatkan nilai *accuracy* sebesar 81.77%, *precision* sebesar 79.81%, *recall* sebesar 81.77% dan f1-*score* sebesar 79.36%. Sedangkan evaluasi menggunakan *k-fold cross validation* sebesar 78.32% pada nilai n_folds = 9.

Sehingga berdasarkan hasil evaluasi tersebut dapat disimpulkan bahwa algoritma *Random Forest* mendapatkan akurasi terbaik dibandingkan dengan algoritma KNN dalam mengklasifikasi kelayakan penerimaan kredit bank.

Penelitian sebelumnya, yang menggunakan dataset yang sama dengan penelitian ini, dilakukan oleh (Islam, Eberle, and Ghafoor 2018). Penelitian tersebut membahas mengenai deteksi akun yang gagal dalam tahap awal analisis kredit sehingga terdapat potensi dalam kegagalan pembayaran ataupun kebangkrutan. Penelitian menggunakan dua pendekatan yaitu pendekatan machine learning dengan algoritma yang berbeda diantaranya yaitu Random Forest, K-Nearest Neighbor, Naïve Bayes, Gradient Boosting, serta Extremely Randomized Tress dan pendekatan heuristic untuk memprediksi resiko kegagalan kredit dataset yang di gunakan menggunakan dataset online dan ofline namun salah satu permasalahan pada penelitian ini adalah data transaksi online dan ofline tidak tersedia untuk umum. Hasil penelitian menunjukkan bahwa, pada pendekatan machine learning menggunakan algoritma Extremely Randomized mendapatkan akurasi terbaik sebesar 95.84%. Sedangkan pada pendekatan heuristic yang menggunakan algoritma yang sama dengan melakukan validasi dengan standard test dan customer spesific test mendapatkan akurasi sebesar 93.14%. Hasil penelitian tersebut menunjukkan bahwa pendekatan ini dapat memprediksi kegagalan pembayaran kredit dan sangat hemat biaya untuk organisasi pendanaan.

Perbandingan hasil penelitian ini dengan penelitian sebelumnya dapat dilihat pada Tabel 4.50.

Tabel 4.50 Perbandingan Hasil Penelitian

	Penelitian Saat Ini		**Penelitian Sebelumnya**	
	Random Forest	***K-Nearest Neighbor***	***Random Forest***	***K-Nearest Neighbor***
Accuracy	82.92%	82.28%	94.46%	82.76%
Precision	81.41%	80.55%	94.78%	71.34%
Recall	82.92%	82.28%	79.32%	36.87%
F1-Score	80.83%	79.97%	86.37%	48.62%

Berdasarkan perbandingan hasil penelitian yang terdapat pada Tabel 4.50, hasil akurasi algoritma *Random Forest* pada penelitian saat ini dan penelitian sebelumnya memiliki hasil yang lebih baik dibandingkan algoritma *K-Nearest Neighbor* pada penggunaan dataset yang sama. Namun, penelitian saat ini memiliki akurasi yang lebih rendah dibandingkan penelitian sebelumnya pada algoritma *Random Forest* dengan nilai *accuracy* 82.92%, *precision* 81.41%, *recall* 82.92%, dan *f1-score* 80.83% hal ini di sebabkan pada penelitian sebelumnya menggunakan dataset offline dan dataset online yang tidak tersedia untuk umum serta menggunakan pendekatan *Heuristic*.

4.4 Pembahasan Hasil Analisis

Dari hasil penelitian yang telah dilakukan dalam melakukan komparasi tingkat akurasi algoritma *Random Forest* dan KNN untuk mengklasifikasi kelayakan penerimaan kredit bank diimplementasikan menggunakan *tools Google Colab* dan menghasilkan tingkat akurasi dari algoritma yang digunakan. Pada

penelitian ini menggunakan *dataset* dari UCI *Machine Learning Repository* terkait data penerimaan kredit dengan rentang waktu tertentu. *Dataset* tersebut memiliki 29.998 data dan terdiri dari 2 *class* dengan nama *class* yaitu "Y" serta terdiri dari *class* "Dibayarkan" memiliki 6.636 data, dan pada *class* "Tidak Dibayarkan" memiliki 23.364 data. Dataset dilakukan pembagian dengan perbandingan 80% data *train* dan 20% data *test* dengan jumlah masing-masing data yaitu data *train* 24000 dan data *test* 6000. Sedangkan, label yang digunakan yaitu Layak dan Tidak Layak.

Dari hasil evaluasi menggunakan *confusion matrix* mendapatkan nilai akurasi terbaik pada algoritma *random forest* yaitu 82.92%, *precision* sebesar 81.41%, *recall* sebesar 82.92% dan f1-*score* sebesar 80.83%. Sedangkan, pada algoritma KNN mendapatkan nilai *accuracy* sebesar 82.28 %, *precision* sebesar 80.55%, *recall* sebesar 82.28% dan f1-*score* sebesar 79.97%.

Berdasarkan hasil evaluasi tersebut bahwa algoritma *Random Forest* mendapatkan akurasi terbaik dibandingkan dengan algoritma KNN dalam mengklasifikasi kelayakan penerimaan kredit bank.

Tabel 4.51 Komparasi Tingkat Akurasi Algoritma KNN dan Random Forest

Confusion Matrix	**Algoritma**	
	KNN	***Random Forest***
Accuracy	82.28%	82.92%
Precision	80.55%	81.41%
Recall	82.28%	82.92%
F1-Score	79.97%	80.83%

Sedangkan hasil dari perbandingan antara menggunakan seleksi fitur maupun tanpa menggunakan seleksi fitur menghasilkan nilai akurasi yang berbeda dari sebelumnya (tanpa menggunakan seleksi fitur). Berikut hasil akurasi yang menggunakan *without feature selection* dan *with feature selection* sebagai berikut:

Tabel 4.52 Komparasi Tingkat Akurasi Algoritma KNN dan Random Forest Menggunakan Without Feature Selection Dan With Feature Selection

	60,40		**70,30**		**80,20**		**90,10**	
	without FS	**with FS**	**without FS**	**with FS**	**without FS**	**with FS**	**without FS**	**with FS**
k	20	20	19	18	12	12	13	14
max depth	10	10	10	10	10	10	10	10

Hasil perbandingan yang terdapat dalam Tabel 4.52 menunjukkan bahwa penggunaan algoritma KNN dan Random Forest menghasilkan konfigurasi optimal untuk nilai k dan max depth. Konfigurasi tersebut disesuaikan dengan pembagian data dan juga penggunaan seleksi fitur atau tanpa seleksi fitur. Hasil menunjukan algoritma random forest mengalami penurunan ketika Menggunakan seleksi fitur PCA sementara pada algoritma KNN justru mengalami peningkatan performa Akurasi ketika Menggunakan seleksi fitur PCA. Hal tersebut relevan dengan penelitian sebelumnya yang lakukan oleh (Aini et al. 2022) dalam Seleksi Fitur untuk Prediksi Hasil Produksi Agrikultur pada Algoritma K-Nearest Neighbor (KNN) . Dalam penelitianya menyimpulkan bahwa penerapan teknik seleksi fitur untuk Prediksi Hasil Produksi Agrikultur menggunakan algoritma K-Nearest Neighbor yang menghasilkan teknik seleksi fitur Principal Component Analysis

(PCA) sebagai peningkatan performa algoritma K-Nearest Neigbor (KNN) dengan akurasi paling tinggi sebesar 99,64%. Pada penelitian yang di Lakukan oleh (Dwi Yulianto, Heni Hermaliani, and Kurniawati 2023) dalam Penerapan Machine Learning Dalam Analisis Stadium Penyakit Hati Untuk Proses Diagnosis dan Perawatan . hasil penelitian menghasilkan Hasil perhitungan akurasi menggunakan algoritma Random Forest dengan PCA antara sebelas dan dua fitur mengalami penurunan sebesar 0,6%.

Selain menggunakan evaluasi menggunakan *confusion matrix*, algoritma Algoritma *K-Nearestt Neighbor Classifier* dan *Random Forest* juga melakukan evaluasi *K-fold cross validation*, dimana dari hasil pengujian tersebut diperoleh bahwa nilai akurasi terbaik *K-Nearestt Neighbor Classifier* diperoleh pada n_fold = 7 sebesar 77.73%. Sedangkan untuk nilai akurasi terbaik *Random Forest* diperoleh pada n_fold = 4 sebesar 82.16 %.

Penentuan KNN dan *random forest* dalam penelitian ini baik digunakan pada dataset yang digunakan terdapat 3000 dataset dengan sifat *binary class* (kelas 0 dan kelas 1). Jumlah dataset yang dimiliki memiliki sifat imbalance dataset (tidak keseimbangan jumlah data antar kelas) (Hamami and Dahlan 2022) dan terdapat 24 (23 feature independent (bebas/tidak terikat) dan 1 feature dependent (terikat)). Terdapat 2 algoritma yang digunakan yaitu KNN dan Random Forest.

Random Forest sendiri merupakan algoritma *ensemble learning* yang merupakan kembangan dari CART. *Random Forest* sendiri cocok digunakan dalam permasalahan *feature* dimensi yang tinggi dan *imbalance* dataset (Faqih Hamami & Iqbal Ahmad Dahlan, 2022). Berdasarkan hal tersebut jika dikaitkan dengan kondisi dataset yang digunakan, maka *Random Forest* dapat digunakan dengan

kondisi dataset yang ada tanpa harus melakukan metode untuk menyeimbangkan data.

Sedangkan pada KNN sendiri, algoritma yang mengandalkan pembandingan titik point data antar satu dengan lainnya (Enggar Novianto & Arief Hermawan, 2023). Hal tersebut jika dikaitkan dengan kondisi dataset yang digunakan akan memiliki performa yang kurang baik bagi KNN itu sendiri. Hal tersebut terjadi pada imbalacen dataset yang ada, dikarenakan jika kondisi data train dan data test (90:10) dengan data train yang memiliki lebih banyak kelas 0 daripada kelas 1, maka KNN tidak mampu mencari pembanding titik point data baru terhadap data train itu sendiri. Sehingga KNN akan memberikan prediksi yang tidak sesuai dengan kondisi pada data baru tersebut.

Dari hasil perbandingan algoritma K-Nearest Neighbor Classifier dan Random Forest dalam konteks klasifikasi kelayakan pemberian kredit bank, dapat disimpulkan bahwa Random Forest menghasilkan akurasi yang lebih tinggi. Faktor-faktor seperti komposisi data training dan testing mempengaruhi hasil akurasi, dan terdapat perbedaan dalam penggunaan *feature selection*. Random Forest dianggap lebih unggul dibandingkan KNN karena mampu menangani ketidakseimbangan kelas, mengatasi kompleksitas dataset, dan efektif dalam menghindari overfitting dengan menerapkan subset acak pada setiap pohon untuk melihat fitur dan data.

BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan dalam melakukan komparasi tingkat akurasi algoritma KNN dan *Random Forest* untuk mengklasifikasi kelayakan penerimaan kredit bank didapatkan kesimpulan sebagai berikut:

1. Hasil dari penerapan *Principal Component Analysis* (PCA) pada klasifikasi kelayakan penerimaan kredit adalah melakukan *reduction* dari 2 *feature* BILL_AMT1, BILL_AMT2 dengan 2 komponen. Artinya hasil akhir PCA terdiri dari 2 komponen vector yang dimana setiap vector adalah hasil gabungan dari kedua fitur tersebut. Berdasarkan hasil perhitungan yang telah dilakukan diperoleh hasil *varaince ratio* tertinggi terletak pada PC1, maka disarankan untuk menggunakan hasil *reduction feature* di PC1 dan artinya persentasi ratio variance hasil fitur BILL_AMT1 dan BILL_AMT2 memiliki keragaman sekitar 94.5%. Selanjutnya dikarenakan terdapat 6 BILL_AMT, maka akan dilakukan *reduction feature* dari BILL_AMT1 hingga BILL_AMT6 hal ini dikarenakan korelasi antar *feature* lebih besar dari 0.8. Kualitas PCA yang baik dengan nilai *variance* > 80%. Artinya dari 2 eigen value tersebut tersebut, yang mana *variance* nya > 80% dan diperoleh hasil eigen value ke-1 yang memiliki *variance* > 80% yaitu sebesar 0.907 atau

90.7%. Sehingga dapat disimpulkan bahwa PCA ke-1 yang balik baik untuk diambil hasil PCA dan dijadikan sebuah fitur baru yaitu fitur *bill*.

2. Tingkat akurasi algoritma *Random Forest* dalam mengklasifikasikan kelayakan nilai *accuracy* 82.92%, *precision* sebesar 81.41%, *recall* sebesar 82.92% dan f1-*score* sebesar 80.83%. Sedangkan evaluasi menggunakan *K-fold cross validation* memperoleh nilai akurasi sebesar 82.16 % pada nilai n_folds = 4. Sedangkan tingkat akurasi algoritma KNN dalam mengklasifikasikan kelayakan penerimaan kredit bank yaitu mendapatkan nilai *accuracy* sebesar 82.28%, *precision* sebesar 80.55%, *recall* sebesar 82.28% dan f1-*score* sebesar 79.97%. Sedangkan evaluasi menggunakan *k-fold cross validation* sebesar 77.73 % pada nilai n_folds = 7. Sehingga berdasarkan hasil evaluasi tersebut dapat disimpulkan bahwa algoritma *Random Forest* mendapatkan akurasi terbaik dibandingkan dengan algoritma KNN dalam mengklasifikasi kelayakan penerimaan kredit bank.

5.2 Saran

Dari kesimpulan yang dibuat, dalam penelitian ini memberikan saran untuk pengembangan penelitian selanjutnya agar lebih baik. Saran yang diberikan yaitu:

1. Bagi penelitian selanjutnya, diharapkan dapat menggunakan dataset yang berbeda dengan penelitian ini untuk mendapatkan hasil yang lebih baik.
2. Bagi penelitian selanjutnya, diharapkan dapat menggunakan algoritma yang lain untuk meningkatkan performa tingkat akurasi dalam mengklasifikasi kelayakan penerimaan kredit bank.

3. Bagi penelitian selanjutnya, diharapkan dapat menggunakan parameter lain yang digunakan untuk hperparameter, selain dari yang diterapkan dalam penelitian ini seperti n_neighbor dan random_state.

DAFTAR PUSTAKA

Agrawal, R. (2020). Spam Classification Using Random Forest Algorithm. *Palarch's Journal Of Archaeology Of Egypt/Egyptology*.

Aldi, F., Nozomi, I., & Soeheri. (2022). Comparison of Drug Type Classification Performance Using KNN Algorithm. *Sinkron : Jurnal dan Penelitian Teknik Informatika*, 1028-1034.

Angkasa, V., & Pangaribuan, J. J. (2022). KOMPARASI TINGKAT AKURASI RANDOM FOREST DAN KNN UNTUK MENDIAGNOSIS PENYAKIT KANKER PAYUDARA. *Information System Development, 7*(1), 49-62.

Apriliah, W., Kurniawan, I., & Baydhowi, M. (2021). Prediksi Kemungkinan Diabetes pada Tahap Awal Menggunakan Algoritma Klasifikasi Random Forest. *SISTEMASI: Jurnal Sistem Informasi, 10*, 163-171.

Atros, K., Padri, A., Nurdiawan, O., Faqih, A., & Anwar, S. (2021). Model Klasifikasi Analisis Kepuasan Pengguna Perpustakaan Online Menggunakan K-Means dan Decission Tree. *JURIKOM (Jurnal Riset Komputer)*, 323-329.

Azis, H., Purnawansyah, & Fattah, F. (2020). Performa Klasifikasi K-NN dan Cross-validation pada Data Pasien Pengidap Penyakit Jantung. *Jurnal Ilmiah, 12*(2, -ISSN 2548-7779), .81-86.

Berrar, D. (2018). Cross-validation. *Encyclopedia of Bioinformatics and Computational Biology*, 1-8.

Bouwman, K. E., Laseroms, O. W., & Lord, R. (2019). Vereist Eigen Vermogen berekening: een simpele kwadratische formule. *SSRN*.

Dinata, R. K., Akbara, H., & Hasdyna, N. (2020). Algoritma K-Nearest Neighbor dengan Euclidean Distance dan Manhattan Distance untuk Klasifikasi Transportasi Bus. *ILKOM Jurnal Ilmiah*, 104-111.

Enggar Novianto, & Arief Hermawan. (2023). KLASIFIKASI ALGORITMA K-NEAREST NEIGHBOR, NAIVE BAYES, DECISION TREE UNTUK

PREDIKSI STATUS KELULUSAN MAHASISWA S1. *Jurnal Teknologi dan Sistem Informasi Univrab*, 146-154.

Erdiansyah, U., Lubis, A. I., & Erwansyah, K. (2022). Komparasi Metode K-Nearest Neighbor dan Random Forest Dalam Prediksi Akurasi Klasifikasi Pengobatan Penyakit Kutil. *JURNAL MEDIA INFORMATIKA BUDIDARMA, 6*, 208-214.

Faqih Hamami, & Iqbal Ahmad Dahlan. (2022). KLASIFIKASI CUACA PROVINSI DKI JAKARTA MENGGUNAKAN ALGORITMA RANDOM FOREST DENGAN TEKNIK OVERSAMPLING. *Jurnal TEKNOINFO*, 87-92.

Farokhah, L. (2020). Implementasi K-Nearest Neighbor untuk Klasifikasi Bunga dengan Ekstraksi Fitur Warna RGB. *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK*, 1129-1136.

Ghoneim, S. S. (2022). Identification of Power Transformer Insulating Paper's State Based on Principal Component Analysis. *International Journal on Electrical Engineering and Informatics*, 770-781.

Ginting, A. K., Lydia, M. S., & Zamzami, E. M. (2021). Peningkatan Akurasi Metode K-Nearest Neighbor dengan Seleksi Fitur Symmetrical Uncertainty. *JURNAL MEDIA INFORMATIKA BUDIDARMA, 5*, 1714-1719.

Handayani, I., & Ikrimach. (2020). Accuracy Analysis of K-Nearest Neighbor and Naïve Bayes Algorithm in the Diagnosis of Breast Cancer. *JURNAL INFOTEL Informatics - Telecommunication - Electronics, 12*, 151-159.

Handayanto, R. T., & Herlawati. (2020). Machine Learning Berbasis Desktop dan Web dengan Metode Jaringan Syaraf Tiruan Untuk Sistem Pendukung Keputusan. *Jurnal Komtika (Komputasi dan Informatika)*, 15-26.

Haristu, R. A. (2019). *Penerapan Metode Random Forest untuk Prediksi Win Ratio Pemain Player Unknown Battleground.* Yogyakarta: Universitas Sanata Dharma.

Harlina, S., Suryani, & Kadang, M. O. (2022). Penerapan Algoritma K-Nearest Neighbor Untuk Klasifikasi kelayakan Calon Nasabah Kredit Berbasis

Web. *Prosiding Seminar Nasional Teknik Elektro, Informatika dan Sistem Informasi (SINTaKS) , 1*.

Hartono, D., Santoso, L. W., & Rostianingsih, S. (2022). Sistem Pendukung Keputusan Pemberian Kredit berdasarkan Klasifikasi Kelancaran Pembayaran Kredit Menggunakan Metode VIKOR pada Bank XYZ. *Jurnal Infra*.

Iman, Q., & Wijayanto, A. W. (2021). Klasifikasi Rumah Tangga Penerima Beras Miskin (Raskin)/Beras Sejahtera (Rastra) di Provinsi Jawa Barat Tahun 2017 dengan Metode Random Forest dan Support Vector Machine. *JUSTIN (Jurnal Sistem dan Teknologi Informasi), 9*, 178-184.

Islam, S. R., Eberle, W., & Ghafoor, S. K. (2018). Credit Default Mining Using Combined Machine Learning and Heuristic Approach. *ArXiv*.

Isnain, A. R., Supriyanto, J., & Kharisma, M. P. (2021). Implementation of K-Nearest Neighbor (K-NN) Algorithm For Public Sentiment Analysis of Online Learning. *IJCCS (Indonesian Journal of Computing and Cybernetics Systems), 15*, 121-130.

Karunia, M., & Thanta, A. M. (2021). Rancang Bangun Aplikasi Face Recognition Pada Pendekatan CRM Menggunakan Opencv Dan Algoritma Haarcascade. *Jurnal IKRA-ITH Informatika*, 109-118.

Khotimah, B. K. (2022). Performance of the K-Nearest Neighbors method on identification of maize plant nutrients. *JURNAL INFOTEL Informatics - Telecommunication - Electronics, 14*, 8-14.

Kosasih, R., Fahrurozi, A., & Riminarsih, D. (2022). Implementasi Random Forest Pada Pengenalan Wajah Menggunakan Fitur Isomap. *CESS (Journal of Computing Engineering, System and Science)*, 459-469.

Kurniasari, I., Kusrini, & Fatta, H. A. (2021). Analisis Sentimen Opini Publik pada Instagram mengenai Covid-19 dengan SVM. *JTECS : Jurnal Sistem Telekomunikasi Elektronika Sistem Kontrol Power Sistem & Komputer, 1*, 67-74.

Larasati, F. A., Ratnawati, D. E., & Hanggara, B. T. (2022). Analisis Sentimen Ulasan Aplikasi Dana dengan Metode Random Forest. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer* , 4305-4313.

Lubis, J. K., & Kharisudin, I. (2021). Metode Long Short Term Memory dan Generalized Autoregressive Conditional Heteroscedasticity untuk Pemodelan Data Saham. *PRISMA, Prosiding Seminar Nasional Matematika*, 652-658.

Martantoh, E., & Rosiyana, I. (2022). PENERAPAN ALGORITMA K-NEAREST NEIGHBOR DALAM PENENTUAN KELAYAKAN PENERIMA KREDIT PADA PT. BPR "PARA SAHABAT"DI BEKASI. *Jurnal Informatika SIMANTIK, 7*, 28-32.

Maulidah, M., Gata, W., Aulianita, R., & Agustyaningrum, C. I. (2020). Algoritma Klasifikasi Decision Tree Untuk Rekomendasi Buku Berdasarkan Kategori Buku. *JURNAL ILMIAH EKONOMI DAN BISNIS*, 89 - 96.

Melinda, R. N., Ningrum, L. M., Suryabrata, I. B., Dwipa, G. S., & Sukoco, T. P. (2021). Program Perhitungan RAB Pekerjaan Struktur Baja (WF BEAM) Menggunakan Bahasa Python. *TIERS Information Technology Journal*, 31-38.

Minarno, A. E., Sumadi, F. D., Wibowo, H., & Munarko, Y. (2020). Classification of batik patterns using K-Nearest neighbor and support vector machine. *Bulletin of Electrical Engineering and Informatics, 9*, 1260-1267.

Muningsih, E. (2022). Kombinasi Metode K-Means dan Decision Tree dengan Perbandingan Kriteria dan Split Data. *Jurnal TEKNOINFO*, 113-118.

Normawati, D., & Prayogi, S. A. (2021). Implementasi Naïve Bayes Classifier Dan Confusion Matrix Pada Analisis Sentimen Berbasis Teks Pada Twitter. *Jurnal Sains Komputer & Informatika (J-SAKTI)* , 697-711.

Nugroho, A., & Husin, A. (2022). Analisis Performa Random Forest Menggunakan Normalisasi Atribut . *SISTEMASI: Jurnal Sistem Informasi , 11*, 186-196.

Ponlatha, S., B, M., A, D. K., M, K., & V, K. (2021). Music Genre Classification Using Deep Learning with KNN. *International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)*, 224-230.

Prasojo, B., & Haryatmi, E. (2021). Analisa Prediksi Kelayakan Pemberian Kredit Pinjaman dengan Metode Random Forest. *Jurnal Nasional Teknologi dan Sistem Informasi , 7*, 79-89.

Prijanto, B., Pulung, R. F., & Sari, A. R. (2021). ANALISIS PERBANDINGAN KUALITAS PELAYANAN BANK SYARIAH DENGAN BANK KONVENSIONAL DI KOTA DEPOK MENGGUNAKAN CARTER MODEL. *Jurnal Tabarru' : Islamic Banking and Finance*, 178 - 194.

Rahardja, C. A., Juardi, T., & Agung, H. (2019). Implementasi Algoritma K-Nearest Neighbor Pada Website Rekomendasi Laptop. *Jurnal Buana Informatika*, 75-84.

Ramadhan, A., Susetyo, B., & Indahwati. (2019). PENERAPAN METODE KLASIFIKASI RANDOM FOREST DALAM MENGIDENTIFIKASI FAKTOR PENTING PENILAIAN MUTU PENDIDIKAN. *Jurnal Pendidikan dan Kebudayaan*, 169-182.

Rodiah, R., & Koswara, U. (2022). PEMANFAATAN SOFTWARE MAXIMA VERSI 16.4.2 TERHADAP MATERI INVERS MATRIKS UNTUK MENINGKATKAN UNDERSTANDING OF CONCEPTS AND THE CORRECT COMPLETION RESULT. *Jurnal Edukasi Sebelas April (JESA)*, 46-55.

Sartika, D., & Saluza, I. (2022). Penerapan Metode Principal Component Analysis (PCA) Pada Klasifikasi Status Kredit Nasabah Bank Sumsel Babel Cabang KM 12 Palembang Menggunakan Metode Decision Tree. *GENERIC Jurnal Ilmu Komputer dan Teknologi Informasi, 14*, 45-49.

Siswa , T. A., & Wibowo, R. P. (2023). Komparasi Metode Seleksi Fitur Dalam Prediksi Keterlambatan Pembayaran Biaya Kuliah. *TEKNIKA, 12*, 73-82.

Soen, G. I., Marlina, & Renny. (2022). Implementasi Cloud Computing dengan Google Colaboratory Pada Aplikasi Pengolah Data Zoom Participants. *JITU : Journal Informatic Technology And Communication*, 24-30.

Subarkah, P., Pambudi, E. P., & Hidayah, S. O. (2020). Perbandingan Metode Klasifikasi Data Mining untuk Nasabah Bank Telemarketing. *Matrik:*

Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer, 20, 139-148.

Suyal, M., & Goyal, P. (2022). A Review on Analysis of K-Nearest Neighbor Classification Machine Learning Algorithms based on Supervised Learning. *International Journal of Engineering Trends and Technology, 70*, 43-48.

Ubaedi, I., & Djaksana, Y. M. (2022). OPTIMASI ALGORITMA C4.5 MENGGUNAKAN METODE FORWARD SELECTION DAN STRATIFIED SAMPLING UNTUK PREDIKSI KELAYAKAN KREDIT. *JSiI | Jurnal Sistem Informasi*, 17-26.

Ubaedi, I., & Djaksana, Y. M. (2022). OPTIMASI ALGORITMA C4.5 MENGGUNAKAN METODE FORWARD SELECTION DAN STRATIFIED SAMPLING UNTUK PREDIKSI KELAYAKAN KREDIT. *JSiI | Jurnal Sistem Informasi, 9*, 17-26.

Wicaksono, R. B., Aulia, S., Hadiyoso, S., & Hidayat, B. (2022). Human height and weight classification based on footprint using gabor wavelet and K-NN methods. *URNAL INFOTEL Informatics - Telecommunication - Electronics, 14*, 101-107.

Yeh, I.-C. (2016, 1 25). *default of credit card clients*. Retrieved from UC Irvine Machine Learning Repository: https://archive.ics.uci.edu/dataset/350/default+of+credit+card+clients