TESIS

Komparasi Algoritma Data Mining Untuk Memprediksi Piutang Biaya Pendidikan Mahasiswa Tak Tertagih (Studi Kasus: Kampus Institut Teknologi dan Bisnis Muhammadiyah Wakatobi)

Disusun oleh:

Nama : **Sry Fasila Hamka**

NIM : **21.55.2150**

Konsentrasi : **Business Intelligence**

PROGRAM STUDI S2 PJJ TEKNIK INFORMATIKA

PROGRAM PASCASARJANA UNIVERSITAS AMIKOM YOGYAKARTA

YOGYAKARTA

2023

TESIS

Komparasi Algoritma Data Mining Untuk Memprediksi Piutang Biaya Pendidikan Mahasiswa Tak Tertagih (Studi Kasus: Kampus Institut Teknologi dan Bisnis Muhammadiyah Wakatobi)

Data Mining Algorithm Comparison To Predicting Receivables Of Unbilled Student Education Costs (Case Study: Kampus Institut Teknologi Dan Bisnis Muhammadiyah Wakatobi)

Diajukan untuk memenuhi salah satu syarat memperoleh derajat Magister

Disusun oleh:

Nama : **Sry Faslia Hamka**
NIM : **21.55.2150**
Konsentrasi : **Business Intelligence**

PROGRAM STUDI S2 PJJ TEKNIK INFORMATIKA
PROGRAM PASCASARJANA UNIVERSITAS AMIKOM YOGYAKARTA
YOGYAKARTA
2023

HALAMAN PENGESAHAN

Komparasi Algoritma Data Mining Untuk Memprediksi Piutang Biaya Pendidikan Mahasiswa Tak Tertagih
(Studi Kasus: Kampus Institut Teknologi dan Bisnis Muhammadiyah Wakatobi)

Data Mining Algorithm Comparison To Predicting Receivables Of Unbilled Student Education Costs
(Case Study : Kampus Institut Teknologi Dan Bisnis Muhammadiyah Wakatobi)

Dipersiapkan dan Disusun oleh

Sry Faslia Hamka

21.55.2150

Telah Diujikan dan Dipertahankan dalam Sidang Ujian Tesis
Program Studi S2 Teknik Informatika
Program Pascasarjana Universitas AMIKOM Yogyakarta
pada hari Kamis, tanggal 7 Desember 2023

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

Yogyakarta, tanggal 7 Desember 2023
Rektor

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

HALAMAN PERSETUJUAN

KOMPARASI ALGORITMA DATA MINING UNTUK MEMPREDIKSI PIUTANG BIAYA PENDIDIKAN MAHASISWA TAK TERTAGIH
(Studi Kasus: Kampus Institut Teknologi dan Bisnis Muhammadiyah Wakatobi)

Dipersiapkan dan Disusun oleh

Sry Faslia Hamka

21.55.2150

Telah Diujikan dan Dipertahankan dalam Sidang Ujian Tesis
Program Studi S2 Teknik Informatika
Program Pascasarjana Universitas AMIKOM Yogyakarta
pada hari **Kamis**, tanggal **7 Desember 2023**

Pembimbing Utama

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

Anggota Tim Penguji

Hanif Al Fatta, M.Kom., Ph.D
NIK. 190302096

Pembimbing Pendamping

Kusnawi, S.Kom.,M.Eng
NIK. 19030202112

Hanafi, S.Kom., M.Eng., Ph.D
NIK. 190302024

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

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

Yogyakarta, tanggal 7 Desember 2023
Direktur Program Pascasarjana

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

HALAMAN PERNYATAAN KEASLIAN TESIS

Yang bertandatangan di bawah ini,

Nama mahasiswa : Sry Faslia Hamka
NIM : 21.55.2150
Konsentrasi : Business Intelligence

Menyatakan bahwa Tesis dengan judul berikut:
Komparasi Algoritma Data Mining Untuk Memprediksi Piutang Biaya Pendidikan Mahasiswa Tak Tertagih (Studi Kasus: Kampus Institut Teknologi dan Bisnis Muhammadiyah Wakatobi)

Dosen Pembimbing Utama : Prof. Dr. Kusrini, M.Kom
Dosen Pembimbing Pendamping : Kusnawi, S.Kom.,M.Eng

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, tanggal 7 Desember 2023

Sry Faslia Hamka

KATA PENGANTAR

Puji dan syukur penulis panjatkan kehadirat Tuhan Yang Maha Kuasa, karena dengan limpahan rezeki dan anugrah-Nya, penulisan tesis ini dapat terselesaikan. Tesis ini berjudul "Komparasi Algoritma Data Mining untuk Memprediksi Piutang Biaya Pendidikan Mahasiswa Tak Tertagih (Studi Kasus: Kampus Institut Teknologi dan Bisnis Muhammadiyah Wakatobi)" dan merupakan salah satu tugas serta persyaratan yang harus dipenuhi dalam menyelesaikan pendidikan pada jenjang Strata 2 (S2) di Program Pascasarjana Magister Teknik Informatika Universitas Amikom Yogyakarta.

Dukungan dari berbagai pihak selama penulis mengikuti perkuliahan pada Program Pascasarjana Magister Teknik Informatika Universitas Amikom Yogyakarta hingga penulisan tesis ini merupakan sumbangan yang tak ternilai harganya. Proses penulisan ini telah memberikan pengalaman yang sangat berarti bagi penulis tentang arti "Perjuangan, Tantangan, Cobaan, dan Kesabaran" yang selalu menghampiri penulis di setiap tahapan penulisan ini. Oleh karena itu, melalui kesempatan ini, sepatutnya penulis mengucapkan terima kasih kepada:

1. Rektor Universitas Amikom Yogyakarta Prof. Dr. M. Suyanto, M.M
2. Ibu Prof. Dr. Kusrini, M.Kom dan Bapak Kusnawi, S.Kom.,M.Eng selaku dosen pembimbing yang di sela-sela kesibukannya masih sempat meluangkan waktu yang dengan penuh kesabaran membimbing dan mengarahkan penulis demi menuju proses kesempurnaan sehingga penulisan Tesis ini dapat terselesaikan.

3. Pimpinan dan segenap Pengelola Program Studi PJJ Magister Teknik Informatika Universitas Amikom Yogyakarta atas pelayanan dan bantuan kepada penulis dalam menyelesaikan tesis ini.
4. Seluruh dosen pengajar yang selalu membina dan mentrasfer ilmu kepada penulis dari awal perkuliahan hingga penyelesaian studi sehingga telah memberikan kontribusi ilmu dalam menambah wawasan, cara berpikir yang dapat dijadikan bekal dalam melaksanakan tugas sekembali dari studi ini.
5. Rekan-rekan mahasiswa Program Studi PJJ Teknik Informatika atas suasana kekeluargaan dan kebersamaan meskipun hanya bertatap maya yang telah tercipta selama mengikuti perkuliahan.
6. Keluarga kecilku yang selalu mendorong dan membantu dalam penyelesaiaan studi ini.

Penulis menyadari bahwa karya ini masih jauh dari kesempurnaan, oleh karena itu penulis mengharapkan kritik dan saran yang konstruktif dari pembaca akan sangat dihargai sehingga penyempurnaan dan perbaikan tesis ini dapat dilakukan baik masa kini maupun dimasa yang akan datang.

Akhirnya dalam doa Penulis bermohon semoga semua bantuan yang telah diberikan akan mendapat imbalan yang berlipat ganda dari Tuhan Yang Maha Pengasih.

Yogyakarta, 7 Desember 2023

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

INTISARI

Penelitian ini bertujuan membandingkan kinerja tiga algoritma data mining, yaitu C4.5, Artificial Neural Network, dan Naïve Bayes, dalam memprediksi piutang biaya pendidikan mahasiswa tak tertagih. Data diperoleh dari Kampus Institut Teknologi dan Bisnis Muhammadiyah Wakatobi, menggunakan data mahasiswa dari PDDikti dan data piutang biaya pendidikan dari Biro Administrasi Keuangan. Batasan penelitian mencakup data mahasiswa dari tahun akademik 2020/2021, 2021/2022, dan 2022/2023, serta data piutang biaya pendidikan dari tahun anggaran 2020-2023. Proses evaluasi menggunakan Confusion Matrix, dengan preprocessing data di Google Colaboratory menggunakan Python.

Setelah penggabungan dataset, dilakukan proses preprocessing data. Pengujian model dilakukan dalam tiga percobaan dengan proporsi data training:data testing 80:20 dan 90:10 serta menggunakan k-fold cross-validation selanjutnya menganalisis atribut yang paling berpengaruh untuk setiap algoritma. Hasil percobaan menunjukkan bahwa algoritma C4.5 berhasil dengan atribut berpengaruh "Jumlah Piutang UKT mahasiswa", "Pendidikan Wali" dan "Pekerjaan Wali". Algoritma Artificial Neural Network tidak memberikan kinerja memuaskan, hal ini mengindikasikan adanya ketidakseimbangan kelas dalam data, dengan atribut "Program Studi" dan "Jumlah Piutang UKT mahasiswa" menunjukkan neuron kurang aktif mempengaruhi model, atribut "Perguruan Tinggi" menunjukkan neuron aktif membentuk jaringan model. Tetapi jika menggunakan k-fold cross-validation menunjukkan model memiliki performa lebih baik. Algoritma Naïve Bayes menunjukkan kinerja baik dengan atribut berpengaruh yaitu "Jumlah Piutang UKT mahasiswa", "Jumlah Piutang BPP" dan "Umur Piutang BPP".

Secara keseluruhan, penelitian ini menyimpulkan algoritma C4.5 dan Naïve Bayes memberikan hasil lebih baik dalam hal akurasi dan presisi dibandingkan dengan Artificial Neural Network dalam memprediksi piutang biaya pendidikan mahasiswa tak tertagih.

Kata kunci: komparasi algoritma, data mining, prediksi, piutang, pendidikan

ABSTRACT

This research aims to compare the performance of three data mining algorithms, namely C4.5, Artificial Neural Network, and Naïve Bayes, in predicting uncollectible student education receivables. Data was obtained from the Muhammadiyah Wakatobi Institute of Technology and Business Campus, using student data from PDDikti and education fee receivable data from the Financial Administration Bureau. Research limitations include student data from the 2020/2021, 2021/2022, and 2022/2023 academic years, as well as education fee receivable data from the 2020-2023 fiscal year. The evaluation process uses Confusion Matrix, with data preprocessing in Google Colaboratory using Python.

After merging the datasets, the data preprocessing process is carried out. Model testing was carried out in three experiments with a proportion of training data: testing data of 80:20 and 90:10 using k-fold cross-validation and then analyzing the most influential attributes for each algorithm. The experimental results show that the C4.5 algorithm is successful with the influential attributes "Amount of student UKT Receivables", "Guardian Education" and "Guardian Occupation". The Artificial Neural Network algorithm does not provide satisfactory performance, this indicates a class imbalance in the data, with the attributes "Study Program" and "Number of Student UKT Receivables" indicating that neurons are less active in influencing the model, the "University" attribute indicates that neurons are active in forming the model network. However, using k-fold cross-validation shows that the model has better performance. The Naïve Bayes algorithm shows good performance with influential attributes, namely "Amount of student UKT Receivables", "Amount of BPP Receivables" and "Age of BPP Receivables".

Overall, this research concludes that the C4.5 and Naïve Bayes algorithms provide better results in terms of accuracy and precision compared to the Artificial Neural Network in predicting uncollectible student education receivables.

Keywords: comparative algorithms, data mining, predictions, receivables, education

BAB I

PENDAHULUAN

1.1. Latar Belakang Masalah

Institut Teknologi dan Bisnis Muhammadiyah Wakatobi (ITBMW) adalah merupakan perguruan tinggi yang berada dibawah naungan persyarikatan Muhammadiyah serta didirikan sejak tahun 2020 sesuai keputusan Menteri Pendidikan dan Kebudayaan Nomor 942/M/2020 tentang izin pendirian Institut Teknologi dan Bisnis Muhammadiyah Wakatobi (ITBMW). Dalam penyelenggaraan pendidikan di kampus ITBMW tidak dikaitkan dengan tujuan komersial, dan sasarannya berpihak kepada masyarakat yang kurang mampu.

Berdasarkan laporan bidang keuangan kampus ITBMW, piutang biaya pendidikan mahasiswa per 31 Desember 2022 sebesar Rp. 984.534.100. Jumlah tersebut meningkat jika dibandingkan tahun sebelumnya yang hanya sebesar Rp. 462.850.000 atau naik sebesar 47%. Hal ini disebabkan masih banyak mahasiswa yang menunggak biaya pendidikan dan karena dengan diberlakukannya sistem mengangsur. Dengan demikian, maka semakin besar piutang mahasiswa akan semakin besar pula potensi piutang biaya pendidikan mahasiswa tak tertagih.

Resiko piutang biaya pendidikan mahasiswa tak tertagih terdiri dari beberapa macam, yaitu: 1) Tidak dibayarnya seluruh tagihan biaya pendidikan; 2) Resiko tidak dibayarnya Sebagian piutang; 3) Tidak tertanamnya modal dalam piutang. Menurut Carl S Warren (2017) terdapat dua metode untuk menilai,

mencatat atau menghapus langsung piutang usaha yang tidak dapat ditagihkan, yaitu (Gitania Aimbu, 2021):

1. Metode penghapusan langsung (*direct-write-off methods*), mencatat beban piutang tak tertagih hanya pada saat suatu piutang dianggap benar-benar tak tertagih.
2. Metode pencadangan (*allowance method*), mencatat beban piutang tak tertagih dengan mengestimasi jumlah piutang tak tertagih pada akhir periode akuntansi.

Permasalahan ini dapat mempengaruhi kinerja piutang mahasiswa karena dalam Undang-Undang No. 12 Tahun 2012 disebutkan bahwa perguruan tinggi wajib memenuhi hak mahasiswa melalui pinjaman tanpa bunga. Oleh sebab itu perguruan tinggi harus dapat mengendalikan piutangnya agar dapat menghindari resiko piutang tak tertagih.

Data yang digunakan dalam penelitian ini adalah data mahasiswa Instititut Teknologi dan Bisnis Muhammadiyah Wakatobi yang diperoleh dari Pangkalan Data Perguruan Tinggi (PDDikti) Tahun Akademik 2020/2021, 2021/2022 dan 2021/2023. Serta data berupa piutang biaya pendidikan mahasiswa tak tertagih diperoleh dari data internal Biro Administrasi Keuangan Institut Teknologi dan Bisnis Muhammadiyah Wakatobi Tahun Anggaran 2021, 2022, dan 2023.

Dalam memprediksi piutang biaya pendidikan mahasiswa tak tertagih yang ada di Kampus Institut Teknologi dan Bisnis Muhammadiyah Wakatobi, maka data yang dimiliki akan diklasifikasikan dan kemudian dibandingkan, algoritma yang dimaksud adalah C4.5, Artificial Neural Network (ANN) dan Naïve Bayes. Hal ini

dilakukan agar peneliti dapat memperoleh nilai akurasi yang terbaik dari ketiga algoritma tersebut. Mengapa pada penelitian ini menggunakan algoritma C4.5, ANN dan Naïve Bayes? C4.5 merupakan model klasifikasi yang menyediakan bagaimana cara agar dataset dapat dikelompokkan dengan cepat dan efektif (Erik Dwi Anggara, 2022). ANN Artificial Neural Network menunjukkan pendekatan yang efektif untuk tujuan yang bersifat umum untuk mengetahui pola, klasifikasi, clustering dan khususnya peramalan time series dengan tingkat keakuratan yang tinggi. Naïve Bayes merupakan algoritma yang bekerja secara independen dalam mengklasifikasikan fitur objek dengan sangat mudah, efektif dan efisien (Woro Isti Rahayu, 2021).

Berbicara tentang algoritma yang digunakan dalam penelitian ini yaitu C4.5, Artificial Neural Network dan Naïve Bayes maka ada beberapa penelitian sebelumnya terkait dengan algoritma dan metodologi yang digunakan pada penelitian ini. Wori Isti Rahayu dkk menggunakan algoritma K-Means Clustering dan Naïve Bayes dengan data sampel berjumlah 111 yaitu 20 data yang merupakan cluster prioritas (C1), dan 91 data yang merupakan cluster tidak prioritas (C2) (Woro Isti Rahayu, 2021). Kekurangan pada penelitian tersebut adalah dataset yang digunakan hanya sedikit namun algoritma Naïve bayes memiliki akurasi yang terbilang sangat tinggi yaitu 98% dibandingkan algoritma K-Means Clustering yaitu sebesar 72%.

Dalam penelitian tersebut terdapat beberapa kekurangan yang menjadi celah atau peluang untuk penelitian lebih lanjut yaitu adalah ukuran dataset yang terbatas. Dengan hanya 111 sampel, hasil analisis statistik dan pemodelan tidak dapat secara

luas digeneralisasi ke populasi yang lebih besar. Penelitian lanjutan dapat mencari dataset yang lebih besar atau mengumpulkan lebih banyak data jika memungkinkan. Penelitian tersebut juga menunjukkan bahwa algoritma K-Means Clustering memiliki akurasi yang rendah (72%) dibandingkan dengan Naïve Bayes (98%), yang menunjukkan bahwa penggunaan algoritma harus ditingkatkan atau diubah. Studi lanjutan dapat mencoba algoritma clustering tambahan atau mencari metode perbaikan. Kekurangan penelitian tersebut juga hanya dapat menilai akurasi, untuk mendapatkan gambaran yang lebih luas tentang kinerja model penelitian lanjutan dapat mempertimbangkan penggunaan metrik evaluasi lain seperti presisi dan recall. Penelitian lanjutan juga dapat dilakukan dengan studi komparatif dengan mencoba algoritma clustering atau klasifikasi lainnya untuk membandingkan kinerja dalam konteks yang sama dengan K-Means Clustering dan Naïve Bayes, hal ini akan memberikan wawasan lebih dalam tentang algoritma mana yang paling sesuai. Untuk itu peneliti mencoba memperhatikan jumlah data set yang akan digunakan agar lebih banyak lagi dan juga akan mencoba membandingkan tiga algoritma pada penelitian kali ini yang salah satunya adalah algoritma Naïve bayes yang akan dibandingkan dengan algoritma C4.5 dan Artificial Neural Network untuk mengetahui kinerja ketiga algoritma tersebut dalam memprediksi piutang biaya pendidikan mahasiswa tak tertagih pada Kampus ITBM Wakatobi.

Arno Prayogo dkk dalam penelitiannya menggunakan algoritma C4.5 untuk menguji data riwayat transaksi nasabah yang layak memperoleh biaya kredit pada PT Astra Internasional Auto Plaju 2000 tanpa melihat apakah para nasabah tersebut

memiliki masalah atau tidak dengan cicilannya. Penelitian tersebut menggunakan rapidminer sebagai solusi untuk menganalisis datasetnya, dengan menggunakan 8 atribut dalam pengklasifikasian kelayakan pemberian kredit pada nasabah yang kemudian menghasilkan nilai akurasi sebesar 87.36%. nilai akurasi ini lebih baik dibandingkan ketika peneliti menggunakan 4 atribut saja yang hanya menghasilkan nilai akurasi sebesar 79.50 % (Arno Prayogo Nawary, 2021).

Penelitian tersebut berkonsentrasi pada klasifikasi kelayakan pemberian kredit tanpa mempertimbangkan masalah cicilan nasabah. Menggabungkan elemen riwayat kredit dan cicilan nasabah ke dalam model klasifikasi adalah gap penelitian yang dapat dipelajari. Hal ini akan membantu menentukan pelanggan yang layak mendapat kredit dan mampu mengelola cicilan dengan baik. Penelitian tersebut juga menemukan bahwa penggunaan delapan atribut dalam mengklasifikasi lebih akurat daripada penggunaan empat atribut. Studi lanjutan yaitu dapat mengeksplorasi dengan menentukan atribut mana yang paling berpengaruh dalam meningkatkan akurasi. Analisis atribut yang lebih mendalam dapat membantu dalam memahami kontribusi masing-masing atribut terhadap prediksi kelayakan kredit. Pada penelitian tersebut hanya menggunakan satu algoritma saja sehingga penelitian lanjutan dapat mencoba berbagai algoritma atau dengan studi komparatif dengan membandingkan akurasi, presisi dan recal masing-masing algoritma klasifikasi, hal ini akan memberikan wawasan lebih dalam tentang algoritma mana yang paling sesuai. Oleh karena itu maka penelitian kali ini akan menggunakan algoritma C4.5 dan akan dibandingkan dengan algoritma Artificial Neural Network

dan Naïve Bayes untuk memprediksi piutang biaya pendidikan mahasiswa tak tertagih.

Penelitian selanjutnya yang dilakukan oleh Sari Dewi untuk memprediksi keberhasilan marketing dalam menentukan layak atau tidaknya calon nasabah kredit perbankan dengan membandingkan algoritma Neural Network, Naïve Bayes, dan K-Nearest Neighbor. Pada penelitian tersebut ditemukan bahwa algoritma Neural Network memiliki nilai akurasi yang lebih baik yaitu sebesar 89,91% dan disusul oleh algoritma K-Nearest Neighbor yaitu sebesar 87,79% dan selanjutnya adalah Naïve Bayes sebesar 84,60%.

Berdasarkan penelitian tersebut dapat melakukan penelitian lanjutan dengan mencoba algoritma selain tiga algoritma yang telah disebutkan di atas, terutama yang terkait dengan machine learning dan data mining. Hal ini akan memberikan pemahaman yang lebih baik tentang algoritma yang terbaik untuk tugas prediksi ini. Pada penelitian kali ini juga akan membandingkan tiga algoritma klasifikasi, dua diantaranya adalah algoritma yang sudah digunakan pada penelitian yang dilakukan oleh Sari Dewi yaitu Artificial Neural Network dan Naïve Bayes dan ditambah dengan algoritma lain yaitu C4.5 untuk mengetahui algoritma mana yang memiliki performa yang lebih baik dalam memprediksi piutang biaya pendidikan mahasiswa di kampus Institut Teknologi dan Bisnis Muhammadiyah Wakatobi.

Penelitian berikutnya adalah yang dilakukan oleh Erik Dwi Anggara dkk. Penelitian tersebut dilakukan untuk mengelompokkan data dengan bantuan machine learning sehingga mempermudah untuk memprediksi kelayakan seorang karyawan dalam memperoleh promosi jabatan berdasarkan kinerjanya. Algoritma

yang digunakan dalam penelitian tersebut adalah Decision Tree dan Regresi Logistik, dimana dari kedua algoritma tersebut ditemukan bahwa nilai akurasi algoritma Decision Tree lebih unggul yaitu sebesar 98,04% dengan menggunakan 50 data, sedangkan algoritma Regresi Logistik yaitu sebesar 86,21% sebanyak 50 data.

Berdasarkan penelitian tersebut terdapat beberapa gap penelitian yang dapat menjadi dasar penelitian berikutnya yaitu penting untuk menguji model tersebut pada dataset yang lebih besar atau dalam konteks perusahaan yang berbeda. Penelitian tersebut juga tidak menjelaskan atribut apa yang digunakan dalam pemodelan sehingga terdapat gap penelitian yang dapat dianalisis lebih mendalam terkait artribut yang mempengaruhi keputusan promosi sehingga dapat memahami atribut-atribut yang paling berpengaruh dalam kelayakan promosi karyawan. Selain decision tree dan regresi logistik, penelitian selanjutnya dapat mencoba algoritma machine learning lainnya untuk perbandingan. Dari hasil penelitian tersebut maka pada penelitian kali ini akan membandingkan tiga algoritma yang salah satunya adalah C4.5 untuk dibandingkan dengan dua algoritma lainnya yaitu Artificial Neural Network dan Naïve Bayes untuk memprediksi piutang biaya pendidikan mahasiswa tak tertagih.

Selanjutnya adalah penelitian yang dilakukan oleh Viry Puspaning Ramadhan dkk yang membandingkan algoritma Neural Network dan regresi Linear untuk memprediksi harga saham BMRI sehingga dapat diketahui algoritma mana yang memiliki performa yang lebih baik diantara keduanya. Analisis datanya menggunakan Rapid Miner yang hasil pengujiannya diperoleh bahwa algoritma

Neural Network memberikan prediksi yang lebih baik karena memiliki tingkat error yang lebih sedikit dibandingkan dengan algoritma Regresi Linear.

Terdapat beberapa aspek yang dapat menjadi dasar untuk penelitian selanjutnya dan mengisi gap penelitian dalam penelitian tersebut seperti melakukan analisis lebih mendalam tentang atribut yang paling berpengaruh dalam prediksi harga saham BMRI, dapat mencoba untuk mengurai vector kemungkinan yang dihasilkan Neural Network dan Regresi Linier untuk memahami alasan dibalik hasil yang berbeda, sehingga pada penelitian kali ini penulis akan menggunakan tiga algoritma klasifikasi yaitu C4.5, Neural Network dan Naïve Bayes untuk memperoleh hasil akurasi yang lebih baik lagi dari penelitian sebelumnya.

Penelitian sebelumnya juga dilakukan oleh Victor Saputra Ginting dkk yang menerapkan algoritma C4.5 untuk memprediksi keterlambatan pembayaran uang sekolah dengan menggunakan bahasa pemrograman python untuk mengolah datanya. Pada penelitian tersebut algoritma C4.5 mencapai nilai akurasi 73%.

Kekurangan dalam penelitian tersebut adalah tingkat akurasi yang tidak terlalu besar dan algoritma yang digunakan hanya satu saja, sehingga terdapat beberapa aspek yang dapat menjadi dasar untuk penelitian lebih lanjut seperti dapat mempertimbangkan penggunaan atribut yang lebih relevan atau mencoba mengidentifikasi atribut yang paling berpengaruh dalam memprediksi keterlambatan pembayaran uang sekolah, mengklasifikasi keterlambatan menjadi kategori yang lebih spesifik misalnya "ringan", "sedang" dan "parah" yang dapat membantu dalam pengambilan tindakan yang lebih tepat. Selain algoritma C4.5, penelitian lebih lanjut dapat mengkomparasi kinerja algoritma tersebut dengan

metode machine learning lainnya sehingga akan memberikan wawasan tentang algoritma mana yang paling efektif dalam kasus tersebut. Penelitian kali ini akan menggunakan tiga algoritma untuk membandingkan nilai akurasinya dengan harapan memperoleh nilai akurasi yang lebih besar lagi (Victor Saputra Ginting, 2020).

Elkham Adakh dkk yang melakukan penelitian dengan membandingkan tiga algoritma yaitu C4.5, Adaboost dan Naïve Bayes untuk memprediksi kinerja bank pada pasar pertukaran saham di Negara Teheran. Dari ketiga algoritma tersebut ditemukan bahwa algoritma Naïve Bayes memiliki performa yang lebih baik dengan nilai akurasi 88,89% yang kemudian disusul oleh Adaboost sebesar 83,78% dan kemudian C4.5 sebesar 77,78%.

Penelitian tersebut merupakan langkah yang sangat baik dalam menerapkan machine learning dalam konteks keuangan. Namun, terdapat gap penelitian yang dapat dijadikan dasar untuk studi lebih lanjut yaitu dapat menganalisis lebih dalam atribut atau fitur yang digunakan dalam pemodelan, atribut apa yang sangat berpengaruh dalam memprediksi kinerja bank. Penelitian lanjutan dapat juga mencoba mengkomparasi algoritma machine learning lainnya sehingga dapat diketahui algoritma yang memiliki performasi yang terbaik dalam kasus tersebut. Penelitian kali ini juga membandingkan tiga algoritma yang dua diantaranya adalah algoritma yang sama pada penelitian Elkham Adakh dkk, yaitu C4.5 dan Naïve Bayes dan mencoba dengan algoritma lainnya yaitu Neural Network untuk memprediksi piutang biaya kuliah mahasiswa tak tertagih di Kampus Institut Teknologi dan Bisnis Muhammadiyah Wakatobi.

1.2. Rumusan Masalah

Berdasarkan celah dan kekurangan penelitian-penelitian sebelumnya sehingga penelitian ini dapat menganalisis lebih dalam atribut yang paling berpengaruh dalam memprediksi piutang biaya kuliah mahasiswa dan akan melihat bagaimana hasil prediksi berupa akurasi, presisi dan recal menggunakan tiga algoritma klasifikasi yaitu C4.5, Artificial Neural Network dan Naïve Bayes untuk memprediksi piutang biaya kuliah mahasiswa Institut Teknologi dan Bisnis Muahammadiyah Wakatobi dengan harapan memperoleh performasi kinerja yang terbaik dari ketiga algoritma tersebut. Selanjutnya melakukan studi komparasi untuk membandingkan kinerja ketiga algoritma tersebut untuk memberikan wawasan lebih dalam terkait algoritma mana yang paling sesuai. Penelitian ini akan mempermudah institusi pendidikan khususnya Kampus Institut Teknologi dan Bisnis Muhammadiyah Wakatobi untuk memperoleh informasi tentang piutang biaya pendidikan mahasiswa tak tertagih sehingga perguruan tinggi dapat mengendalikan piutangnya agar dapat menghindari resiko piutang tak tertagih.

Berdasarkan latar belakang dan kekurangan penelitian-penelitian sebelumnya yang menjadi masalah utama dalam penelitian ini adalah:

a. Bagaimana hasil prediksi piutang biaya pendidikan mahasiswa tak tertagih menggunakan algoritma C4.5, Artificial Neural Network dan Naïve Bayes?
b. Bagaimana analisis atribut yang paling berpengaruh dalam memprediksi piutang biaya kuliah mahasiswa tak tertagih menggunakan algoritma C4.5, Artificial Neural Network dan Naïve Bayes?

c. Bagaimana hasil perbandingan algoritma C4.5, Artificial Neural Network dan Naïve Bayes untuk memprediksi piutang biaya pendidikan mahasiswa tak tertagih?

1.3. Batasan Masalah

Batasan masalah dalam penelitian ini, yaitu:

a. Data mahasiswa diperoleh dari Pangkalan Data Pendidikan Tinggi (PDDikti) Institut Teknologi dan Bisnis Muhammadiyah Wakatobi Tahun Akademik 2020/2021, 2021/2022, dan 2022/2023.
b. Data berupa piutang biaya pendidikan mahasiswa tak tertagih diperoleh dari data internal Biro Administrasi Keuangan Institut Teknologi dan Bisnis Muhammadiyah Wakatobi Tahun Anggaran 2020, 2021, 2022, dan 2023.
c. Dalam penelitian ini hanya menerapkan algoritma C4.5, Artificial Neural Network dan Naïve Bayes yang selanjutnya dilakukan penerapan Evaluasi Confusion Matrix.
d. Pengolahan datanya menggunakan google colaboratory dengan bahasa pemograman phyton.

1.4. Tujuan Penelitian

Tujuan utama penelitian adalah untuk:

a. Mengetahui hasil prediksi piutang biaya pendidikan mahasiswa tak tertagih menggunakan algoritma C4.5, Artificial Neural Network dan Naïve Bayes.
b. Mengetahui lebih dalam atribut yang paling berpengaruh dalam memprediksi piutang biaya kuliah mahasiswa tak tertagih menggunakan algoritma C4.5, Artificial Neural Network dan Naïve Bayes.

c. Memberikan pemahaman bagi para pembaca agar mengetahui hasil perbandingan algoritma C4.5, Artificial Neural Network dan Naïve Bayes untuk memprediksi piutang biaya pendididkan mahasiswa tak tertagih.

1.5. Manfaat Penelitian

Manfaat Penelitian adalah untuk:

a. Memberikan kontribusi dalam pengembangan ilmu pengetahuan dan menambah pengetahuan keilmuan pada bidang data mining khususnya yang berkaitan dengan penerapan algoritma C4.5, Artificial Neural Network dan Naïve Bayes
b. Hasil penelitian ini diharapkan dapat menjadi bahan masukan bagi penelitian yang berkaitan dengan komparasi metode data mining.

BAB II

TINJAUAN PUSTAKA

2.1. Tinjauan Pustaka

Beberapa penelitian yang menggunakan algoritma C4.5, Artificial Neural Network dan Naïve Bayes yang digunakan pada penelitian penelitian-penelitian sebelumnya antara lain penelitian yang dilakukan oleh Iqbal Taufik Ahmad Nur dkk. Penelitian tersebut dilakukan untuk mendeteksi kualitas kredit pada koperasi simpan pinjam dengan algoritma Support Vector Machine (SVM), Naïve Bayes dan Neural Network melalui proses evaluasi menggunakan 5-fold cross validation. Hasil yang diperoleh menyatakan bahwa algoritma Neural Network memperoleh hasil akurasi sebesar 86,81%, rerata *precision* sebesar 0,8194, rerata *recall* sebesar 0,8236, dan rerata nilai AUC sebesar 0,9158. Walaupun saat eksekusi algoritma ini membutuhkan waktu 3,058 detik dibandingkan dengan algoritma SVM dan Naïve Bayes hanya kisaran 0-1 detik (Iqbal Taufik Ahmad Nur, 2018).

Penelitian selanjutnya adalah penelitian yang dilakukan oleh Putri Armilia Prayesy dkk menggunakan algoritma Naïve Bayes, dan Neural Network untuk menganalisis klasifikasi kredit kepegawaian sesuai dengan kriteria yang menjadi standar bank. Proses pemodelan dilakukan dengan validasi split dengan menggunakan data kredit debitur eksisting tahun 2017-2020 berjumlah 1.314 dataset yang kemudian dibagi menjadi 2 bagian, yaitu 80% sebagai data pelatihan sedangkan 20% digunakan sebagai data pengujian,. Hasil penelitian menunjukkan bahwa algoritma Neural Network memiliki hasil yang lebih baik dengan nilai benar

sebesar 84,13%, dibandingkan dengan algoritma Naive Bayes hanya menghasilkan nilai sebesar 72,62% (Putri A. P., 2022).

Berikutnya adalah penelitian yang dilakukan oleh Aulia Rahmayanti dkk yang membandingkan algoritma Naïve Bayes dengan C4.5 untuk memprediksi kelulusan tepat waktu mahasiswa. Hasil dari penelitian tersebut menyebutkan bahwa algoritma C4.5 memiliki performa yang lebih baik dengan nilai akurasi sebesar 90%, nilai pada recall dan precision sebesar 92% dan 94% sedangkan algoritma Naïve Bayes memiliki nilai akurasi sebesar 85%. dengan nilai recall 86% dan precision 93% (Aulia Rahmayanti, 2022).

Fatmi Zola dkk melakukan penelitian yang bertujuan untuk membantu memecahkan masalah pemberian beasiswa terhadap siswa yang berprestasi dengan menggunakan algoritma Jaringan Syaraf Tiruan Backpropagation dalam memprediksi prestasi siswa. Data testing dan data trainingnya menggunakan nilai ujian sekolah. Hasil dari pengujian dengan pola arsitektur 4-2-1, data menjadi dua bagian yaitu 20 data pelatihan dengan persentase error 95,6 %. Dan 20 data pengujian dengan persentase error 100%. Semakin kecil tingkat ketelitian error yang digunakan maka akan semakin kecil penyimpangan hasil Jaringan Syaraf Tiruan dengan target yang diinginkan (Fatmi Zola, 2018).

Selanjutnya penelitian dilakukan oleh Rendi Irawan dkk yang mencoba membandingkan klasifikasi data mining untuk menganalisis prediksi kelayakan kredit melalui metode Naïve Bayes dan Decision Tree. Data calon debitur tersebut telah diolah melalui tahapan data mining – Naïve Bayes dan Decision Tree. Data diuji melalui validasi silang k-folds (k = 10). Hasil dari penelitian ini adalah akurasi

metode Decision Tree (J-48) lebih tinggi dibandingkan dengan metode Naïve Bayes. Hasil perbandingan kedua algoritma adalah algoritma Decision Tree (J-48) memiliki akurasi sebesar 95,24% dan algoritma Naïve Bayes memiliki akurasi sebesar 79,59% (Rendi Irawan, 2020).

Nur Hadianto dkk dalam penelitiannya menggunakan algoritma Neural Network untuk mengklasifikasikan peminjaman nasabah bank dan menggunakan tools rapid miner untuk pengolahan datanya yang kemudian dilanjutkan dengan pengukuran menggunakan confusion matrix, kurva ROC. Hasil yang diperoleh melalui pengujian confusion matrix, kurva ROC menunjukkan nilai akurasi yang sangat tinggi, dan nilai dominan AUC dan algoritma. Nilai akurasi adalah 98,24% dengan AUC sebesar 0,979 (Nur Hadianto, 2019).

Penelitian yang dilakukan oleh Yogiek Indra Kurniawan dkk adalah untuk membuat sebuah aplikasi yang dapat memprediksi apakah calon nasabah layak atau tidak untuk diberikan kredit. Prediksi yang dilakukan menggunakan metode klasifikasi data mining, yaitu algoritma C4.5 berdasarkan data-data penunjang yang dimiliki setiap nasabah untuk mengklasifikasikan faktor manakah yang paling berpengaruh pada tingkat pembayaran kredit di koperasi. Pada aplikasi yang dibangun, algoritma C4.5 menghasilkan pohon keputusan yang mudah diinterpretasikan berdasarkan variabel-variabel yang ada. Pada aplikasi terdapat fitur yang dapat digunakan untuk mengambil keputusan terhadap nasabah yang akan mengajukan kredit di koperasi. Hasil pengujian blackbox pada aplikasi menunjukkan bahwa aplikasi telah dapat berjalan sesuai dengan yang diharapkan, sedangkan hasil pengujian algoritma juga menunjukkan bahwa aplikasi telah dapat

mengimplementasikan algoritma C4.5 dengan benar. Selain itu, hasil pengujian terhadap akurasi menunjukkan bahwa nilai rata-rata maksimal dari *Accuracy* mencapai 79,19% (Yogiek Indra Kurniawan, 2021).

Penelitian berikutnya adalah yang dilakukan oleh Anisa Nizar Ahmed yang bertujuan bertujuan untuk mengidentifikasi faktor-faktor terpenting yang menyebabkan tingginya risiko kredit dalam pinjaman usaha kecil. Ini bertujuan untuk menggunakan teknik pembelajaran mesin yang dapat memprediksi kualitas pinjaman secara akurat berdasarkan berbagai atribut yang dipertimbangkan saat mengevaluasi pinjaman. Terlihat bahwa Naïve Bayes, Decision Tree dan Random Forest bekerja dengan akurasi yang baik lebih dari 90% dan kecepatan pada dataset yang diberikan berisi 99, 699 baris dan 25 variabel. Suku bunga adalah faktor utama yang menentukan risiko kredit pinjaman dan akibatnya nilainya. Faktor lain seperti jangka waktu pelunasan dan metode pencairan juga merupakan faktor penting yang dapat dikontrol oleh UKM sebelum mengajukan pinjaman. Hal ini akan meningkatkan kemungkinan pinjaman disetujui dan juga akan meminimalkan risiko yang ditimbulkan oleh bank pemberi pinjaman (Ahmed, 2019).

Selanjutanya penelitian yang dilakukan oleh Dwi Yuni Utami dkk yang membandingkan algoritma Neural Network, Naïve Bayes dan Regresi Logistik untuk memprediksi penyakit diabetes dengan menggunakan tools rapid miner dalam mengolah datanya serta dilanjutkan dengan menerapkan Evaluasi Confussion Matrix untuk mengetahui akurasinya. Hasil yang diperoleh menyatakan bahwa Regresi Logistik memperoleh nilai akurasi 75,78% dan AUCnya sebesar 0,801 nilai tersebut lebih tinggi bila dibandingkan dengan dua algoritma

tandingannya yaitu Naïve Bayes dengan akurasi sebesar 74,87% dan AUCnya sebesar 0,799% serta Neural Network dengan nilai akurasi sebesar 69,27% dan AuCnya 0,736% (Dwi Yuni Utami, 2021).

2.2. Keaslian Penelitian

Tabel 2.1. Matriks literatur review dan posisi penelitian
Komparasi Algoritma Data Mining Untuk Memprediksi Piutang Biaya Pendidikan Mahasiswa Tak Tertagih
(Studi Kasus: Kampus Institut Teknologi dan Bisnis Muhammadiyah Wakatobi)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
1	Perbandingan Algoritma K-Means dan Naïve Bayes untuk Memprediksi Prioritas Pembayaran Tagihan Rumah sakit Berdasarkan Tingkat Kepentingan pada PT. Pertamina (Persero)	Woro Isti Rahayu, Cahyo Prianto, dan Ema Ainun Novia; Jurnal teknik Informatika ; April 2021	Membuat system prioritas tagihan pembayaran rumah sakit berdasarkan tingkat kepentingannya dengan menggunakan algoritma data mining yaitu K-Means Clustering dan Naïve Bayes Classifier agar menentukan alternatif pemecahan masalah yang ada sebagai penentuan proritas pembayaran sehingga memperoleh acuan nilai yang sesuai dengan kriteria perusahaan.	Dari hasil perhitungan perbandingan yang didapatkan pada kedua metode tersebut untuk nilai accuracy algoritma K-means sebesar 72% dan algoritma Naïve Bayes sebesar 98%. Jadi algoritma naïve bayes classifier adalah adalah yang terbaik untuk digunakan sebagai penentu alternatif pemecahan masalah yang ada dalam, penentuan prioritas pembayaran sehingga memperoleh acuan nilai yang sesuai dengan kriteria perusahaan.	Untuk penelitian selanjutnya diharapkan dapat menggunakan metode yang berbeda serta melakukan perbandingan dengan metode lainnya. Sebaiknya menentukan dahulu variable-variabel perbandingan untuk memperoleh hasil yang baik sesuai dengan tujuan penelitian.	Penelitian yang dirujuk menggunakan algoritma K-Means dan Naïve Bayes, sedangkan penelitian ini menggunakan selain algoritma Naïve Bayes ditambahkan dengan Decision Tree C4.5 dan Neural Network
2	Penerapan Data Mining dalam Memprediksi Kelancaran Kredit	Arno Prayogo Nawary, dan Kurniati;	Untuk menguji data histori dari nasabah yang menerima pembiayaan kredit	Algoritma C4.5 yang diuji dengan 8 atribut menghasilkan akurasi 87, 36% dan Ketika diuji	Peneliti selanjutnya mungkin bisa menggunakan algoritma lebih dari	Penelitian yang dirujuk hanya menggunakan algoritma C45,

Tabel 2.1. Matriks literatur review dan posisi penelitian (Lanjutan)

	Nasabah menggunakan Algoritma C.45	Bina Darma Confren on Computer Science; 2021	menggunakan algoritma C4.5, terlepas dari apakah mereka bermasalah dengan cicilan atau tidak.	dengan menggunakan 4 atribut menghasilkan nilai akurasi 79,50 %. Artinya eksperimen ini sudah mencapai tingkat baik, sehingga dapat meningkatkan tingkat ketelitian dalam proses klasifikasi dan prediksi dengan cara menambahkan beberapa atribut dan histori.	satu sebagai perbandingan	sedangkan penelitian ini menambahkan dengan algoritma Naïve bayes dan Neural Network
3	Prediksi Kinerja sebagai Rekomendasi Kenaikan Golongan dengan Decesion Tree dan Regresi Logistik.	Erik Dwi Anggara, Andreas Wijaja, dan Bernald Renaldy Suteja; Jurnal Teknik Informatika dan Sistem Informasi; 2022	Melakukan pengelompokkan data dengan menggunakan algoritma decision tree dan regresi logistic untuk membantu memprediksi kelayakan seorang karyawan agar memperoleh promosi berdasarkan kinerjanya.	Hasil yang diperoleh menyatakan bahwa nilai akurasi decision tree lebih baik dari regresi linear, untuk itu algoritma decision tree merupakan algoritma yang paling tepat untuk digunakan memprediksi kinerja pegawainya.		Penelitian yang dirujuk hanya menggunakan algoritma decision tree dan regresi logistik, sedangkan penelitian ini menggunakan selain decision tree yaitu Naïve Bayes dan Neural Network
4	Analisis Perbandingan Algoritma Forecasting dalam Prediksi Harga Saham LQ45 PT Bank Mandiri Sekuritas (BMRI)	Viry Puspaning Ramadhan dan Fandi Yulian pamuji; Jurnal Teknologi dan Manajemen Informatika; 2022	Mengetahui algoritma apa yang terbaik dalam memprediksi harga saham BMRI dengan membandingkan algoritma neural network dan regresi linier.	Hasil pengujian menggunakan algoritma neural network nilai akurasinya lebih tinggi dibandingkan regresi linier, oleh karena itu neural network memberikan prediksi lebih baik.	Peneliti berikutnya dapat mengoptimalkan algoritma peramalan atau prediksi agar dapat memperoleh nilai evaluasi yang lebih baik.	Penelitian yang dirujuk menggunakan algoritma neural network dan regresi linier, sedangkan penelitian ini menggunakan algoritma decision

Tabel 2.1. Matriks literatur review dan posisi penelitian (Lanjutan)

						tree, neural network dan naïve bayes
5	Komparasi Metode Algoritma Data Mining pada Prediksi Uji Kelayakan Credit Approval pada Calon Nasabah Kredit Perbankan.	Sari Dewi; Jurnal Khatulistiwa Informatika; 1 Juni 2019	Untuk memprediksi keberhasilan marketing dalam menentukan kelayakan calon nasabah kredit perbankan dengan membandingkan algoritma neural network,naïvee bayes dan k-nearest neighbors.	Dari hasil uji coba yang diperoleh, algoritma Neural network memiliki nilai akurasi tertinggi yaitu dibandingkan algoritma lainnya. Dengan hasil ini, menunjukkan bahwa Neural Network merupakan metode yang cukup baik dalam prediksi data sehingga dapat memberikan hasil untuk permasalahan identifikasi calon nasabah	Untuk penelitian lebih lanjut disarankan melakukan penyeleksian atribut, dikarenakan Atribut pday dalam metode algoritma tidak berpengaruh (hal ini dikarenakan nilai nya sama) sehingga bisa dianalisa lebih lanjut apakah atribut tersebut diperlukan atau tidak. Penelitian ini dapat dikembangkan dengan algoritma yang lain misalkan saja dengan metode statistik lainnya seperti Support Vector Machine	Penelitian yang dirujuk menggunakan algoritma Neural Network, naïve bayes dan k-nearest neighbors sedangkan penelitian ini menggunakan algoritma decision tree, Neural Network, dan Naïve Bayes
6	Comparison of same Data Mining Models in Forecast of Performance of Banks Accepted in	Elham Adakh, Arefeh Fadavi, Mohammd Ibrahim Muhammad Pourzandi; Iranian Journal of Finance; 2019	Menilai dan membandingkan algoritma pada data mining yaitu C4.5, Adaboost dan Naïve Bayes untuk	Klasifikasi Naïve Bayes memiliki akurasi prediksi tertinggi, diikuti oleh algoritma pohon keputusan AdaBoost dan C4.5, masing-masing.	Penelitian ini dapat bermanfaat bagi para pakar pasar saham, dan lembaga keuangan lainnya	Penelitian yang dirujuk menggunakan algoritma C.45, adaboost dan Naïve Bayes sedangkan

Tabel 2.1. Matriks literatur review dan posisi penelitian (Lanjutan)

	Tehran Stock Exchange Market		memprediksi kinerja bank pada pasar pertukaran saham di Negara Teheran.	Dengan akurasi tinggi 70%, semua model yang diterapkan memiliki daya prediksi yang tepat dengan perkiraan yang dapat diterima	yang aktif di bidang beasiswa, serta mahasiswa untuk penelitian masa depan. Indeks yang diterapkan dalam penelitian ini harus digunakan dalam metode penambangan data lainnya (misalnya, jaringan saraf, mesin vektor pendukung), sehingga hasilnya dapat dibandingkan.	penelitian ini menggunakan algoritma Neural network, decision tree dan naïve bayes
7	Penerapan Algoritma C4.5 dalam Memprediksi Keterlambatan Pembayaran Uang Sekolah menggunakan Python	Victor Saputra Ginting, Kusrini, Emha Taufiq Luthfi; Jurnal Teknologi Informasi; 1 Juni 2020	Untuk membantu pihak sekolah dalam mengambil keputusan dan mengurangi tingkat keterlambatan pembayaran uang sekolah	Pada penelitian ini penerapan algoritma C4.5 ke dalm pemrograman python dapat dilakukan. Nilai akurasi algoritma C4.5 yang dihasilkan dari penelitian ini adalah sebesar 73% dengan recall sebesar 71% dan presisi 71%.	Diharapkan penelitian selanjutnya dapat menggunakan algoritma lebih dari satu seperti KNN, CNN, Naïve Bayes, dll agar memperoleh tingkat akurasi yang lebih besar Diharapkan penelitian selanjutnya dapat dilakukan lebih	Penelitian yang dirujuk hanya menggunakan algoritma C.45 sedangkan penelitian ini menambahkan dengan algoritma neural network dan naïve bayes

Tabel 2.1. Matriks literatur review dan posisi penelitian (Lanjutan)

					mendalam lagi pada atribut yang akan digunakan agar tingkat akurasi yang diperoleh lebih optimal.	
8	Perbandingan Performa Metode Klasifikasi SVM, Neural Network dan Naïve Bayes untuk Mendeteksi Kualitas Pengajuan Kredit di Koperasi Simpan Pinjam	Iqbal Taufik Ahmad Nur, Nanang Yudi Setiawan, Fitra A. Bachtiar; Jurnal Teknologi Informasi dan Ilmu Komputer; 2019	Untuk menemukan metode yang memiliki kinerja yang lebih baik diantara ketiga metode tersebut yaitu SVM, Neural Network dan Naïve Bayes.	Proses pendeteksian kualitas kredit dengan menggunakan 185 data hasil dari pre processing data, dimana parameter yang digunakan pada penelitian ini didasarkan pada pendapat Pimpinan KSP, algoritma Neural Network mempunyai tingkat akurasi, precision, recall, dan nilai AUC terbaik dibandingkan dengan metode klasifikasi lain yang digunakan pada penelitian ini, yaitu SVM dan Naïve Bayes dengan rerata tingkat akurasi 86,81%, precision 0,8194, recall 0,8236, dan nilai AUC 0,9158, sehingga yang tergolong sebagai klasifikasi paling	Karena Neural Network menjadi algoritma yang paling lambat pada saat dieksekusi karena adanya proses iterasi pada tahap klasifikasi. Maka diharapkan pada penelitian selanjutnya mencegah terjadi hal-hal tersebut	Penelitian yang dirujuk membandingkan algoritma SVM, Neural Network, dan Naïve Bayes sedangkan penelitian ini membandingkan algoritma decision tree, Neural Network dan Naïve Bayes

Tabel 2.1. Matriks literatur review dan posisi penelitian (Lanjutan)

				baik untuk pendeteksian kualitas kredit pada proses pengajuan pinjaman di koperasi simpan pinjam adalah Neural Network. Namun, Neural Network menjadi metode dengan waktu eksekusi paling lambat apabila dibandingkan dengan dua metode lain		
9	Classification of the Fluency Multipurpose of Bank Mandiri Credit Payments Based on Debtor Preferences using Naïve Bayes and Neural Network	Putri Armilia Prayesy, Edi Surya Negara; Jurnal Online Informatika; 2022	Menganalisis klasifikasi kredit kepegawaian berdasarkan kriteria yang menjadi standar bank	Hasil perbandingan menunjukkan bahwa algoritma Neural Network memiliki hasil yang lebih baik dengan nilai akurasi sebesar 84,13%, sedangkan algoritma Naive Bayes hanya menghasilkan nilai akurasi sebesar 72,62%	Perlu ditambahkan beberapa algoritma lain sebagai pembanding. Menambahkan lebih banyak data dan atribut sehingga dapat meningkatkan nilai akurasi yang lebih baik.	Penelitian yang dirujuk menggunakan algoritma Neural network dan naïve bayes, sedangkan penelitian ini ditambahkan dengan algoritma decision tree
10	Perbandingan Metode Algoritma C4.5 dan Naïve Bayes untuk Memprediksi Kelulusan Mahasiswa	Aulia Rahmayanti, Lili Rusdiana, Suratno Suratno; Walisongo Journal of Information; 2022	Membandingkan akurasi algoritma C4.5 dengan Naïve Bayes pada sejumlah dataset.	Data diuji menggunakan tools Weka kemudian model yang diuji akan menghasilkan nilai accuracy kedua algoritma tersebut menunjukan tingkat akurasi algoritma C4.5		Penelitian yang dirujuk menggunakan algoritma C.45 dan naïve bayes, sedangkan penelitian ini menambahkan dengan algoritma neural network

Tabel 2.1. Matriks literatur review dan posisi penelitian (Lanjutan)

				sebesar 90% lebih baik dari pada algoritma Naïve Bayes sebesar 85%. Algoritma C4.5 juga memberikan nilai pada recall dan precision sebesar 92% dan 94% lebih baik dibandingkan algoritma Naïve Bayes dengan nilai 86% dan 93%.		
11	Jaringan Syaraf Tiruan menggunakan Algoritma Backpropagation untuk Memprediksi Prestasi Siswa	Fatmi Zola, Gunadi Widi Nurcahyo, Julius Santony; Jurnal Teknologi dan Open Source; 2018	Diharapkan dapat membantu memecahkan masalah pemberian beasiswa	Dengan menggunakan Jaringan Syaraf Tiruan dengan algoritma Backpropagation proses prediksi lebih cepat, akurat, meminimalisir kesalahan dan bisa menggunakan teknologi komputer. Serta mudah dalam pngembangannya. Dapat menganalisis faktor menyebabkan kemerosotan prestasi siswa Aplikasi Jaringan Syaraf Tiruan dapat memprediksi prestasi siswa pada mata		Penelitian yang dirujuk menggunakan algoritma backpropagation sedangkan penelitian ini menggunakan algoritma decision tree, naïve bayes dan neural network

Tabel 2.1. Matriks literatur review dan posisi penelitian (Lanjutan)

				pelajaran, Bahasa Indonesia, Bahasa Inggris, Matematika dan IPA dengan algoritma		
12	Comparison of Data Mining Classification Methods for Predicting Credit Appropriation through Naïve Bayes and Decision Tree Methods.	Rendi Irawan, Agustinus Eko Setiawan, Kurnia Muludi; Proceeding of 6th ICITB 2020	Untuk menganalisis prediksi kelayakan kredit melalui metode Naïve Bayes dan Decision tree.	Metode Decision Tree memiliki akurasi yang baik sebesar 95,24% dan metode Naïve Bayes memiliki akurasi sebesar 79,50%.	Perlu menggabungkan lebih banyak metode lagi dalam analisis data dan pemecahan masalah sehingga suatu sistem akan lebih efektif dan efisien ke dalam mengolah atau menyajikan informasi Mengatur waktu penelitian secara maksimal Dibutuhkan peran responden yang terlibat langsung pada penelitian ini	Penelitian yang dirujuk menggunakan algoritma decision tree dan naïve bayes sedangkan penelitian ini ditambahkan dengan algoritma neural network
13	Klasifikasi Peminjaman Nasabah Bank menggunakan Metode Neural Network	Nur Hadianto, Hafifah Bella Novitasari, Ami Rahmawati; Jurnal Pilar Nusa Mandiri; 2019	Mengklasifikasi nasabah yang layak untuk mendapatkan pinjaman dengan memperhitungkan parameter yang ada seperti usia, jumlah pendapatan, jumlah	Algoritma backpropagation dalam Neural Network menggunakan struktur 12 – 15 – 8 – 1 , training cycles sejumlah 500, learning rate 0.01 dan momentum 0.1		Penelitian yang dirujuk menggunakan algoritma backpropagation sedangkan penelitian ini ditambahkan dengan algoritma

Tabel 2.1. Matriks literatur review dan posisi penelitian (Lanjutan)

			keluarga, rata-rata pengeluaran perybulan tingkat pendidikan dan lainnya	menghasilkan nilai akurasi = 98.24%, dengan AUC sebesar =0,979 yang menunjukkan bahwa klasifikasi yang dihasilkan sangat baik, sehingga nasabah dengan parameter yang ada dapat diprediksi menggunakan pola ini untuk menentukan nasabah yang layak diberikan pinjaman dari pihak Bank		decision tree dan naïve bayes
14	Prediction for Cooperative Credit eligibility using Data Mining Classification with C4.5 Algorthm	Yogiek Indra Kurniawan, Annastalia Fatikasari, Muhammad Luthfi Hidayat, Mohamad Waluyo	Untuk membuat sebuah aplikasi yang dapat memprediksi apakah calon nasabah layak atau tidak untuk diberikan kredit	Algoritma C4.5 berhasil digunakan pada aplikasi dan dapat berjalan dengan baik. Hasil pengujian blackbox pada aplikasi menunjukkan bahwa aplikasi sudah sesuai seperti yang diinginkan serta semua menu yang ada dapat berjalan dengan baik. Hasil pengujian accuracy yang didapat menunjukkan bahwa nilai rata – rata pada semua data uji mencapai 79.19%.		Penelitian yang dirujuk menggunakan algoritma C.45 sedangkan penelitian ini selain menggunakan algoritma C.45 ditambahkan dengan algoritma naïve bayes dan neural network

Tabel 2.1. Matriks literatur review dan posisi penelitian (Lanjutan)

15	Credit-Risk Assessment of Small Business Loans using Naïve Bayes, Decision Tree and Random Forest	Anisa Nizar Ahmed; National College of Ireland; 2019	Menganalisis aplikasi pinjaman usaha kecil menggunakan algoritma pembelajaran mesin untuk mengidentifikasi faktor-faktor yang menyebabkan risiko kredit tinggi	Naïve Bayes, Decision Tree dan Random Forest bekerja dengan akurasi yang baik lebih dari 90% dan kecepatan pada dataset yang diberikan berisi 99, 699 baris dan 25 variabel		Penelitian yang dirujuk menggunakan algoritma naïve bayes, decision tree dan random forest sedangkan penelitian ini menggunakan algoritma naïve bayes, decision tree dan neural network
16	Comparison of Neural Network Algortma, Naïve Bayes and Logistic Regression to Find the Highest Accuracy in Diabetes	Dwi Yuni Utami, Elah Nurlelah, Fuad Nur Hasan; Journal of Informatics and Telecommunication Engineering; 2021	Membandingkan algoritma Neural Network, Naïve Bayes dan Logistic Regression dengan menggunakan aplikasi rapidminer dan menerapkan Evaluasi Confusion Matrix untuk memperoleh akurasi dan ROC curve.	Hasil dari penelitian ini adalah metode logistic regression menjadi metode yang memiliki akurasi yang baik yaitu 75,78% dan AUC sebesar 0,801 disusul oleh algoritma Naïve Bayes dengan akurasi sebesar 74,87% dan nilai AUCnya sebesar 0,799%, selanjutnya adalah algoritma Neural Network dengan nilai akurasi sebesar 69,27% dan nilai AUC sebesar 0,736%		Penelitian yang dirujuk menggunakan algoritma neural network, naïve bayes dan logistic regression sedangkan penelitian ini menggunakan decision tree, naïve bayes dan neural network

2.3. Landasan Teori

Pada penelitian ini, sangat diperlukan definisi dan informasi untuk memperdalam materi dan mempermudah dalam pembuatan penulisan, diantaranya adalah sebagai berikut:

2.3.1 Data Mining

Data mining merupakan perangkat lunak yang digunakan untuk menemukan pola, tren dan aturan tersembunyi yang terkandung dalam basis besar dan menghasilkan aturan yang digunakan untuk memprediksi perilaku masa depan (Rendi Irawan, 2020). Data mining juga dapat diartikan sebagai proses iterative dan interaktif untuk menemukan pola dan model baru yang lengkap, yang berguna dan dapat dipahami dalam data base yang akurat. Sebagai tren dan pola yang diperlukan dalam mendukung pengambilan keputusan dimasa depan, maka basis data besar dilibatkan dalam penambangan data. (Arno Prayogo Nawary, 2021).

Proses data mining terbagi menjadi dua kategori berdasarkan tugas dan tujuannya sesuai dengan dari adanya target variabel dan metode pembelajaran (learning), yaitu (Muhammad Arhami, 2020):

1. Belajar yang diawasi (Supervised Learning); konsep pada metode ini adalah mencoba mencari hubungan antara atribut input dan atribut target, hubungan yang dicari tersebut diwakili dalam struktur yang dinamakan model. Contoh: Regresi, Decesion Tree, K-Nearest Neighbors, Naïve Bayes Classification, Neural Network, dll

2. Belajar tanpa pengawasan (Unsupervised Learning); konsep pada metode ini adalah tidak ada pengawasan, dimana hanya ada input data yang tersedia, dan penggunanya dimungkinkan untuk mempelajari model yang lebih besar dan lebih kompleks. Contoh: Clustering dan Self Organization Map (SOM).

Menurut Daniel T. Larose dalam (Kusrini, 2009), terdapat enam fungsi pada data mining, yaitu:

1. Deskripsi

   Terkadang peneliti dan analis secara sederhana ingin mencoba mencari cara untuk menggambarkan pola dan kecenderungan yang terdapat dalam data. Deskripsi dari pola dan kecenderungan sering memberikan kemungkinan penjelasan untuk suatu pola atau kecenderungan.

2. Estimasi

   Estimasi hampir sama dengan klasifikasi , hanya saja variable target estimasi lebih kea rah numerik daroada kearah kategori. Model dibangun menggunakan *record* lengkap yang menyediakan nilai dari variable target sebagai nilai prediksi.

3. Prediksi

   Prediksi hampir sama dengan klasifikasi dan estimasi, kecuali bahwa dalam prediksi nilai dari hasil akan ada di masa mendatang.

4. Klasifikasi

   Dalam klasifikasi terdapat target variable kategori. Contoh: penggolongan pendapatan dapat dipisahkan dalam tiga kategori, yaitu: pendapatan tinggi, pendapatan sedang, dan pendapatan rendah.

5. Pengklusteran

Clustering merupakan pengelompokkan record, pengamatan, atau memperhatikan dan memebentuk kelas objek-objek yang memiliki kemiripan. Cluster adalah kumpulan record yang meiliki kemiripan satu dengan lainnya serta memiliki ketidak miripan dengan record-record dalam cluster lain.

6. Asosiasi

Asosiasi dalam data mining bertugas untuk menemukan atribut yang muncul dalam satu waktu. Dalam dunia bisnis lebih umum disebut dengan analisis keranjang belanja.

Dalam penelitian ini, fungsi yang digunakan adalah fungsi prediksi, fungsi prediksi adalah fungsi bagaimana sebuah proses nantinya akan menemukan pola tertentu dari suatu data. Pola-pola tersebut dapat diketahui dari berbagai variabel yang ada pada data. Pola yang sudah ditemukan tersebut dapat dipakai untuk memprediksi variabel lain yang belum diketahui nilai ataupun jenisnya. Oleh karena alasan tersebut, fungsi ini memudahkan dan menguntungkan bagi siapapun yang memerlukan prediksi yang akurat (Erik Dwi Anggara, 2022).

Dalam Data Mining ada banyak model dalam prosesnya. Diantaranya yaitu menggunakan Knowledge Discovery In Database (KDD) yang merupakan "salah satu metode yang dapat digunakan untuk memperoleh pengetahuan yang berasal dari database yang tersedia. Untuk lebih jelasnya, berikut gambaran dari metode KDD" (Novianti, 2019).

1. Selection

   Penentuan variable yang tujuan utamanya adalah untuk membuat dataset target dari data asli, yaitu memilih subset dari variabel atau sampel data, di mana penemuan harus dilakukan.

2. Prepocessing

   Preprocessing, yang bertujuan untuk "membersihkan" data dengan melakukan berbagai operasi, seperti pemodelan kebisingan dan penghapusan, mendefinisikan strategi yang tepat untuk menangani bidang data yang hilang, akuntansi untuk waktu-urutan informasi.

3. Transformation

   Transformasi, yang bertanggung jawab untuk mengurangi dan memproyeksikan data, dalam rangka untuk memperoleh representasi yang cocok untuk tugas tertentu yang akan dilakukan; ini biasanya dilakukan dengan melibatkan teknik transformasi atau metode yang dapat menemukan representasi invarian dari data.

4. Data mining

   Tahapan yang berkaitan dengan mengekstrak pola menarik dengan memilih (i) metode atau tugas data-mining tertentu (mis, ringkasan, klasifikasi, pengelompokan, regresi, dan sebagainya), (II) algoritma yang tepat (s) untuk melakukan tugas yang dihadapi, dan (III) representasi yang sesuai dari hasil output.

5. Evaluation/ Interpretation

Yang dimanfaatkan oleh pengguna untuk menafsirkan dan mengekstrak pengetahuan dari pola yang ditambang, dengan memvisualisasikan pola; interpretasi ini biasanya dilakukan dengan memvisualisasikan pola, model, atau data yang diberikan seperti pasir model, membungkus, secara iteratif melihat kembali pada langkah sebelumnya dari proses.

2.3.2 Algoritma C4.5

Algoritma C4.5 merupakan salah satu algoritma Decision tree (Pohon Keputusan) yang paling efektif untuk melakukan klasifikasi dan prediksi. Algoritma yang dimasukkan berupa training sample dan sample. Algoritma C4.5 merupakan algoritma yang dikembangkan dari algoritma ID3 (Iterative Dichotomiser 3) yang mudah dipahami dan dapat divisualisasikan dalam bentuk pohon keputusan yang menarik. (Arno Prayogo Nawary, 2021). C4.5 adalah algoritma pohon keputusan dan generalisasi dari algoritma ID3, yang menggunakan kriteria rasio keuntungan untuk memilih sifat tertentu. Algoritma berhenti ketika jumlah sampel di bawah nilai yang ditentukan. Di sisi lain, algoritma mengeksploitasi teknik pasca-pemangkasan, menerima data numerik yang mirip dengan algoritma sebelumnya. Selain itu, versi metode yang dimodifikasi dapat diterapkan untuk data yang tidak lengkap (Elham Adakh, 2019).

Pohon keputusan adalah struktur berhierarki yang digunakan untuk memprediksi kelas atau label dari data yang diuji berdasarkan serangkaian keputusan. Algoritma C4.5 secara iteratif membagi dataset menjadi subset yang

lebih kecil berdasarkan atribut yang paling informatif dalam hal pemisahan kelas. Atribut yang paling berpengaruh mengidentifikasikan atribut yang memiliki kontribusi signifikan terhadap prediksi model.

Secara umum Langkah-langkah C4.5 dalam membangun sebuah pohon keputusan adalah sebagai berikut (Kusrini, 2009):

1. Memilih atribut sebagai akar.
2. Membuat cabang untuk tiap-tiap nilai.
3. Membagi kasus dalam cabang.
4. Mengulangi proses untuk setiap proses cabang sampai semua kasus pada cabang memiliki kelas yang sama.

Tahapan-tahapan dalam membuat pohon keputusan dengan menggunakan algoritma C4.5 adalah sebagai berikut (Novianti, 2019):

1. Mempersiapkan data training: Data training dapat diperoleh dari data histori yang pernah terjadi sebelumnya dan telah dikelompokkan dalam kelas-kelas tertentu.
2. Menentukan akar dari pohon dengan menghitung nilai gain yang tertinggi dari masing-masing atribut atau berdasarkan nilai entropy terendah. Sebelum menghitung nilai gain terlebih dahulu mencari nilai entropy dengan rumus sebagai berikut:

$$Entropy(S) \equiv E(S) = \sum_{i=1}^{N} - p_i \log_2 (p_i)$$

Keterangan:

S = himpunan kasus berdasarkan partisinya

N = jumlah partisi

P_i = probabilitas tiap kelas

3. Kemudian menghitung nilai gain dengan rumus sebagai berikut:

$$Gain(A,S) = Entropy(S) - \sum_{j=1}^{v} \frac{|Sj|}{|S|} Entropy(Sj)$$

Atau Gain (A,S) = H(S) – H(A,S)

|Sj| = jumlah kejadian/contoh dengan nilai ke-j dari atribut A

|S| = total jumlah kejadian/contoh dalam himpunan data S

J = himpunan partisi yang berbeda dari atribut A

Entropy (Sj) = entropi dari subset kejadian untuk atribut A

H(A,S) = entropi dari suatu atribut A

4. Ulangi Langkah ke-2 hingga semua nilai record terpartisi. Dan proses partisi akan berhenti, jika:

- Semua record dalam node N memperoleh nilai yang sama.
- Tidak ada atribut dalam record yang terpartisi Kembali.
- Tidak ada record dalam cabang yang kosong.

2.3.3 Artificial Neural Network

Algoritma Neural Network atau Jaringan Syaraf Tiruan merupakan salah satu representasi buatan dari otak manusia yang digunakan untuk mensimulasikan proses dari pembelajaran otak manusia itu sendiri. Neural Network tercipta sebagai model generalisasi matematis dari *human cognition* atau pemhaman manusia berdasarkan asumsinya sebagai proses infofrmasi yang terjadi pada elemen sederhana atau biasa disebut dengan neuron. Selain itu isyarat yang mengalir dianatara neuron atau sel syaraf tersebut akan melalui suatu sambungan penghubung. Sambungan penghubung tersebut masing-masing memiliki nilai

bobot yang sesuai. Setiap sel syaraf tersebut merupakan fungsi dari aktivasi terhadap isyarat dan hasil penjumlahan berbobot yang masuk kepada sel syaraf yang digunakan untuk menentukan output dari isyaratnya (Viry Puspaning Ramadhan, 2022).

Neural network juga dapat diartikan sebagai satu set unit input/output yang terhubung dimana tiap relasinya memiliki bobot. Neural Network ini merupakan sistem adaptif yang dapat merubah strukturnya untuk memecahkan masalah berdasarkan informasi eksternal maupun internal yang mengalir melalui jaringan tersebut. Secara sederhana Neural Network adalah sebuah alat pemodelan data statistik non-linear. Neural Networ dapat digunakan untuk memodelkan hubungan yang kompleks antara input dan output untuk menemukan pola-pola pada data. Neuron juga terdiri dari satu output. Outputnya adalah terbentuk dari pengolahan berbagai input oleh neuron (Dewi, 2019).

Backpropagation merupakan salah satu bagian dari algoritma jaringan syaraf tiruan yang dalam teknik pembelajarannya dilakukan dengan cara menyesuaikan bobot-bobot jaringan syaraf tiruan dengan arah mundur berdasarkan nilai error dalam proses pembelajarannya. Metode ini bekerja melalui proses iterative dengan menggunakan kumpulan data trining (membandingkan nilai prediksi dari jaringan dengan setiap data trining). Pada setiap prosesnya, bobot relasi dalam jaringan dimodifikasi agar memngurangi nilai Mean Square Error (MSE) antara nilai prediksi dari jaringan dengan nilai realnya. Disebut sebagai metode backpropagatian dikarenakan modifikasi relasi jaringan saraf tersebut dilakukan dengan arah mundur dari lapisan output layer hingga layer pertama dari

hidden layer (Kusrini, 2009). Tiga langkah utama pada metode Backpropagation, yaitu (Zulinda, 2019):

1. Data dimasukkan dalam input jaringan (Feedforward).
2. Perhitungan dan propagasi balik dari error yang ditemukan (Backpropagation)
3. Pembaharuan bobot dan bias.

Berikut adalah Langkah-langkah algoritma pembelajaran untuk Backpropagation oleh (Fausett, 1994):

Langkah 0 : Inisialisasi bobot (set bobot pada nilai random yang kecil)

Langkah 1 : Jika kondisi berhenti tidak terpenuhi, lakukanlah langkah 2-9

Langkah 2 : Untuk setiap pasangan pelatihan, lakukan langkah 3-8

Feedforward

Langkah 3 : Setiap neuron pada lapisan input (Xi, i=1,2,…, n) menerima sinyal input xi dan menjalankan sinyal tersebut ke semua neuron pada lapisan selanjutnya (dalam lapisan tersembunyi).

Langkah 4 : Untuk setiap neuron dalam lapisan tersembunyi (Zj, j=1,2, …, p) jumlahkan bobotnya dengan sinyal input masing-masing:

$$Z_{in_j} = v_{oj} + \sum_{i=1}^{n} x_i v_j \qquad (1)$$

Terapkan fungsi aktivasi untuk menghitung nilai sinyal output:

$$Z_j = f(Z_{in_j}) \qquad (2)$$

Kemudian kirimkan sinyal tersebut ke samua lapisan neuron pada lapisan selanjutnya (lapisan output).

Langkah 5 : Untuk setiap neuron pada lapisan output (Yk, k=1,2, …, m), jumlahkan bobotnya dengan sinyal inputnya masing-masing:

$$Y_{in_k} = w_{oj} + \sum_{i=}^{n} Z_j W_k \quad (3)$$

Terapkan fungsi aktivasi untuk menghitung nilai sinyal output:

$$Y_k = f(Y_{in_k}) \quad (4)$$

Backpropagation of error

Langkah 6 : Setiap neuron pada lapisan output (Yk, k=1,2, …, m) menerima sebuah pola target yang berhubungan dengan pola input pelatihan kemudian dihitung kesalahannya dengan menggunakan rumus sebagai berikut:

$$\delta_{k=(t_k - y_k)\cdot f'((y_{in_k})} \quad (5)$$

Setelah itu hitung perubahan bobotnya yang nantinya digunakan untuk mengubah nilai w_{jk} dengan menggunakan rumus berikut ini:

$$\Delta w_{jk} = \alpha . \delta_k . z_k \quad (6)$$

Lalu hitunglah perubahan biasnya yang akan mengubah nilai w_{ok}

$$\Delta w_{0k} = \alpha . \delta_k \quad (7)$$

Langkah 7 : Untuk setiap neuron pada lapisan tersembuyi (Zj, j =1,2…,p): Jumlahkanlah nilai delta imputannya dari neuron pada lapisan diatasnya.

$$\delta_{in_j} = \sum_{k=1}^{m} \delta_k w_{jk} \quad (8)$$

Kalikan dengan turunan aktivasinya untuk menghitung nilai kesalahannya

$$\delta_j = \delta_{in_j} . f'(Z_{in_j}) \quad (9)$$

Hitung perubahan bobotnya (digunakan nanti untuk mengubah nilai v_{ij})

$$\Delta v_{ij} = \alpha . \delta_j . x_i \quad (10)$$

Kemudian hitung perubahan biasnya (digunakan nanti untuk mengubah nilai v_{0j})

$$\Delta v_{0j} = \alpha . \delta_j \quad (11)$$

Perbaharui BoBot dan Bias

Langkah 8 : Untuk setiap neuron pada lapisan keluaran (Yk,k=1,2,…, m) ganti nilai bobot dan biasnya (j = 0,1,2,…, n)

$$w_{ij_{(baru)}} = w_{jk_{(lama)}} + \Delta w_{jk} \quad (12)$$

Untuk setiap neuron pada lapisan tersembunyi (Zj,j=1,2, …, p) ganti nilai bobot biasnya (i=0,1,2, …, n)

Langkah 9 : menguji dan memeriksa pada saat kondisi berhenti.

2.3.4 Naïve Bayes

Algoritma Naïve Bayes adalah pengklasifikasian statistik yang dapat digunakan untuk memprediksi probabilitas keanggotaan suatu class. Naïve Bayes merupakan klasifikasi berdasarkan pada teorema Bayes, algoritma ini memiliki kemampuan klasifikasi menyerupai algoritma decision tree dan neural network (Dewi, 2019). Thomas Bayes adalah seorang ilmuwan Inggris yang menemukan klasifikasi naïve bayes dengan metode probabilitas dan statistik untuk memprediksi peluang di masa depan berdasarkan pengalaman di masa sebelumnya sehingga dikenal sebagai Teorema Bayes. Rumus Naive Bayes Classifier ditunjukkan di bawah ini (Novianti, 2019):

$$P(H \mid X) = \frac{P(H) \cdot P(H)}{P(X)} \quad (1)$$

Keterangan:

X : Data class yang belum diketahui

H : Hipotesis data class spesifik

P(H|X) : Probabilitas hipotesis H berdasar kondisi X (*posteriori probability*)

P(H) : Probabilitas hipotesis H (*prior probability*)

P(X|H) : Probabilitas X berdasarkan kondisi H (*like hood probability*)

P(X) : Probabilitas X (*evidence probability*)

Pada konteks Algoritma Naïve Bayes, analisis atribut merupakan langkah penting dalam memahami atribut-atribut atau fitur-fitur yang digunakan dalam proses klasifikasi yang mempengaruhi hasil prediksi. Atribut yang digunakan dalam Algoritma Naïve Bayes dapat dianalisis untuk menggambarkan kontribusi suatu atribut terhadap hasil prediksi. Ini melibatkan mengidentifikasi atribut-atribut yang paling berpengaruh dalam pemisahan kelas atau dalam proses klasifikasi. Atribut yang paling berpengaruh adalah yang memberikan kontribusi signifikan dalam membedakan kelas-kelas atau dalam membuat prediksi yang akurat.

2.3.5 Evaluasi dan Validasi Model

K-fold validation digunakan dalam pengembangan sebuah model, dimana dataset terbagi menjadi dua bagian yaitu data training dan data testing, data ini dibagi menjadi ukuran k-sama agar dapat meningkatkan hasil akurasi. Untuk k-sendiri ditentukan menjadi 10 partisi dan kemudian diulangi kembali sebanyak 10 kali lipat agar kinerja dan keakuratan model ditemukan dengan nilai rata-rata.

Beberapa parameter evaluasi yang sering digunakan dalam penelitian klasifikasi antara lain adalah Confusion matrix yang merupakan suatu metode yang sering digunakan dalam melakukan perhitungan akurasi pada pembelajaran data mining. Metode ini melakukan perhitungan dengan menghasilkan 4 keluaran, yaitu:

recall, precision, akurasi dan error rate. Cara kerja confusion matrix yaitu menilai kinerja model klasifikasi sesuai jumlah objek yang diprediksi dengan benar dan salah (Dewi, 2019).

Confusion matrix terdiri dari nilai-nilai seperti berikut ini (Novianti, 2019):

- True positif (TP); merupakan nilai positif, baik nilai prediksi maupun nilai aktualnya.
- False positif (FP); merupakan nilai yang prediksinya positif dan nilai aktualnya negative
- False negative (FN); merupakan nilai yang prediksinya negative dan nilai aktualnya positif
- True negative (TN) : merupakan nilai negative, baik nilai prediksinya maupun nilai aktualnya.

Berikut rumus menghitung rasio accuracy, precision, recall, dan f1-score:

$$Precision = \frac{TP}{TP+FP} \quad (1)$$

$$Recall = \frac{TP}{TP+FN} \quad (2)$$

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (3)$$

$$F1 - Score = \frac{2\,x\,recall\,x\,precision}{recall\,+precision} \quad (4)$$

2.3.6 Biaya Pendidikan dan Biaya Kuliah

Biaya pendidikan adalah sejumlah biaya berupa uang yang harus dibayarkan oleh peserta didik kepada lembaga tempat ia mengenyam pendidikan. Selain itu juga ada biaya-biaya yang dibutuhkan seperti membeli peralatan, transportasi,

makan, tempat tinggal dan lainnya. Namun, dari hal yang perlu dibayarkan dalam menempuh pendidikan tersebut adalah biaya pendidikan yang dibayarkan kepada lembaga pendidikan tersebut. Besarnya biaya pendidikan yang dibayarkan peserta didik tergantung dari ketentuan lembaga pendidikan tersebut. (Hendri, 2021).

Pada dasarnya biaya pendidikan merupakan dua hal penting yang perlu dikaji dan dianalisis, yaitu biaya pendidikan secara keseluruhan (*total cost*) dan biaya satuan peristiwa (*unit cost*). Biaya satuan di tingkat sekolah adalah merupakan *aggregate* biaya Pendidikan sekolah, baik itu yang bersumber dari pemerintah, orang tua dan masyarakat yang dikeluarkan untuk menyelenggarakan pendidikan dalam satu tahun pelajaran. Faktor-faktor yang mempengaruhi biaya pendidikan antara lain, yaitu (Putri A. , 2021):

- Kenaikan harga
- Perubahan relative dalam gaji pengajar
- Meningkatnya standar pendidikan
- Meningkatnya tuntutan terhadap pendidikan lebih tinggi
- Perubahan dalam populasi dan kenaikan persentase peserta didik di sekolah
- Meningkatnya usia anak yang meninggalkan sekolah.

Biaya kuliah perlu diketahui oleh calon mahasiswa yang akan masuk ke perguruan tinggi. Jenis-jenis biaya kuliah secara umum terdiri dari (Putri A. , 2021):

- Biaya seleksi
- Biaya sumbangan pengembangan

- Biaya penyelenggaran pendidikan (BPP), biaya ini berupa niaya penyelenggaran pendidikan wajib dan biaya penyelenggaran pendidikan tambahan (SKS)
- Biaya praktikum
- Dana kegiatan mahasiswa (DKM)

2.3.7 Piutang

Piutang merupakan klaim dari pihak yang akan memperoleh pembayaran (dalam hal ini adalah perguruan tinggi) kepada pihak yang memiliki kewajibannya untuk membayar (dalam hal ini adalah mahasiswa perguruan tinggi tersebut) yang pembayarannya secara umum dalam bentuk uang. Berdasarkan jangka waktu pembayarannya suatu piutang dapat dikelompokkan menjadi 3, yaitu: jangka pendek (kurang dari 1 tahun), jangka menengah (antara 1 sampai dengan 3 tahun), dan jangka panjang (lebih dari 3 tahun). Namun setiap institusi pendidikan mungkin memiliki kebijakan dan skema piutang yang berbeda-beda, Seperti halnya dengan Kampus Institut Teknologi dan Bisnis Muhammadiyah Wakatobi (ITBM Wakatobi) yang memiliki kebijakan skema piutang sebagai berikut:

1. Skema piutang jangka pendek: Menetapkan jangka waktu pembayaran yang singkat, 1-2 bulan setelah tagihan dikeluarkan. Skema ini diterapkan pada biaya pendidikan semesteran atau tahunan.
2. Skema piutang menengah: Memberikan jangka waktu pembayaran hingga 6-12 bulan setelah tagihan dikeluarkan. Skema ini diterapkan pada biaya pendidikan yang lebih besar seperti biaya pendaftaran, biaya kuliah.

3. Skema piutang jangka panjang: Menawarkan pembayaran secara cicilan atau dengan menggunakan skema pinjaman untuk membantu mahasiswa membayar biaya pendidikan. Jangka waktu pembayaran dapat mencapai beberapa tahun, tergantung pada jumlah piutang dan persyaratan yang ditetapkan oleh Kampus ITBM Wakatobi.

Terdapat dua metode yang digunakan untuk menilai, mencatat, atau menghapus langsung piutang usaha yang tidak dapat ditagih menurut Carl S Warren, yaitu (Gitania Aimbu, 2021) :

1. Metode Penghapusan Langsung (*direct write-off method)* , mencatat beban piutang tak tertagih hanya pada saat suatu piutang dianggap benar-benar tak tertagih.
2. Metode Pencadangan *(allowance method),* mencatat beban piutang tak tertagih dengan mengestimasi jumlah piutang tak tertagih pada akhir periode akuntansi.

Resiko piutang tak tertagih adalah suatu resiko yang timbul karena adanya transaksi penjualan secara kredit disebut sebagai resiko kerugian piutang. Resiko kerugian piutang terdiri dari beberapa macam yaitu (Gitania Aimbu, 2021):

1. Resiko tidak dibayarnya seluruh tagihan (piutang)
2. Resiko tidak dibayarnya sebagian piutang
3. Resiko keterlambatan pelunasan piutang
4. Resiko tidak tertanamnya modal dalam piutang.

BAB III

METODE PENELITIAN

3.1. Jenis, Sifat, dan Pendekatan Penelitian

3.1.1 Jenis Penelitian

Jenis penelitian yang digunakan adalah masuk dalam kategori penelitian eksperimen karena peneliti melakukan beberapa skenario percobaan untuk memperoleh nilai akurasi dari algoritma C4.5, Artificial Neural Network dan Naïve Bayes.

3.1.2 Sifat Penelitian

Penelitian ini bersifat deskriptif dengan membandingkan metode klasifikasi C4.5, Artificial Neural Network dan Naïve Bayes untuk memprediksi piutang biaya kuliah mahasiswa tak tertagih.

3.1.3 Pendekatan Penelitian

Penelitian ini menggunakan pendekatan kuantitatif yang artinya peneliti akan melakukan penelitian sesuai dengan langkah-langkah atau alur yang sudah ditentukan oleh peneliti.

3.2. Metode Pengumpulan Data

Bagian Pada penelitian ini, metode pengumpulan untuk mendapatkan sumber data yang digunakan adalah:

a. Pengumpulan data primer yaitu data mahasiswa yang diperoleh secara langsung dari Pangkalan Data Pendidikan Tinggi (PDDikti) Institut Teknologi dan Bisnis Muhammadiyah Wakatobi Tahun Akademik

2020/2021, 2021/2022, dan 2022/2023; serta data internal Biro Administrasi Keuangan Institut Teknologi dan Bisnis Muhammadiyah Wakatobi 2020, 2021, 2022, dan 2023.

b. Pengumpulan data sekunder yaitu data yang diperoleh dari literatur, buku, jurnal dan informasi lainnya yang berkaitan dengan judul pada penelitian ini.

3.3. Metode Analisis Data

Dalam menganalisis data pada penelitian ini, ada beberapa langkah yang harus dilakukan, yaitu:

a. Pengumpulan Data

Data yang digunakan adalah data mahasiswa yang diperoleh secara langsung dari Pangkalan Data Pendidikan Tinggi (PDDikti) Institut Teknologi dan Bisnis Muhammadiyah Wakatobi Tahun Akademik 2020/2021, 2021/2022, dan 2022/2023; serta data internal Biro Administrasi Keuangan Institut Teknologi dan Bisnis Muhammadiyah Wakatobi 2020, 2021, 2022, dan 2023.

b. Pembersihan Data

Proses pembersihan dilakukan terhadap data yang ganda, inkonsisten, missing value atau missing data dan outlier data. Proses ini dilakukan menggunakan `StandardScaler` memiliki tujuan untuk menormalkan atau menstandarisasi nilai-nilai fitur numerik. Hal tersebut perlu dilakukan agar tidak mempengaruhi performa proses klasifikasi yang akan dilakukan. Pembersihan data yang dilakukan juga dengan mengisi missing value dengan

nilai rata-rata, median, modus atau yang lainnya. Proses pembersihan data tersebut melalui proses imputasi menggunakan `SimpleImputer` yang bertujuan untuk mengisi nilai yang hilang (missing value) dengan nilai yang relevan. Tahapan ini dilakukan agar dataset lengkap dan dapat digunakan dalam pemodelan. Variabel input yang digunakan pada penelitian ini setelah melewati proses pembersihan data meliputi:

1. NIM
2. Status dengan parameter aktif
3. Perguruan Tinggi dengan parameter Institut Teknologi dan Bisnis Muhammadiyah Wakatobi
4. Program studi dengan parameter: Ilmu Perikanan; Kewirausahaan; Teknologi Informasi
5. Jenjang dengan parameter S1
6. Alamat kelurahan/desa
7. Alamat kecamatan
8. Pendidikan Ayah/Ibu/Wali dengan parameter: Tidak Sekolah; Putus SD; SD Sederajat; SMP Sederajat; SMA Sederajat; D1; D2; D3; D4/S1; S2; S3.
9. Pekerjaan Ayah/Ibu/Wali dengan parameter : Tidak Bekerja; Nelayan; Petani; Peternak; PNS/TNI/Polri; Karyawan Swasta; Pedagang Kecil; Pedagang besar; wiraswasta; wirausaha; buruh; pensiunan; tenaga kerja Indonesia; tidak dapat diterapkan; sudah meninggal.
10. Penghasilan dengan parameter: kurang dari Rp. 1000.000; Rp1000.000

– Rp 2.000.000; Lebih dari Rp. 2000.000; Kurang dari Rp. 500.000; Kurang dari Rp. 500.000 – Rp. 999.999; Rp. 1.000.000 - Rp.1.999.999; Rp 2.000.000 – 4.999.999; Rp. 5.000.000 – Rp. 20.000.000; Rp Lebih dari 20.000.000; Tidak berpenghasilan.

11. Keterangan
12. Jumlah piutang UKT Mahasiswa
13. Umur piutang UKT dengan parameter : lunas; jangka pendek; jangka menengah; jangka panjang.
14. Jumlah piutang Dana Pengembangan Pendidikan
15. Umur piutang Dana Pengembangan Pendidikan dengan parameter: lunas; jangka pendek; jangka menengah; jangka panjang.

Selanjutnya adalah variable output dalam penelitian ini adalah berupa keterangan status piutang dengan parameter: 1) piutang tertagih. 2) piutang tak tertagih.

c. Percobaan dan Pengujian Model

Percobaan dan pengujian model pada penelitian ini menggunakan google colaboratory dengan cara memasukkan data menggunakan file berformat csv. Selanjutnya, membagi dataset yang dimiliki menjadi 2 bagian, sebagian akan digunakan sebagai data training dan sebagian akan digunakan sebagai data testing. Pengujian dengan mengubah proporsi data training dan data testing adalah penting untuk mengevaluasi kinerja algoritma data mining yaitu:

- Dengan mengubah proprosi data training dan data testing dapat memahami bagaimana algoritma tersebut berperilaku dengan berbagai ukuran dataset training dan testing.
- Dengan mengubah proporsi data training dan data testing dapat menguji kemampuan model dalam menggeneralisasi dari data training ke data testing yang baru.
- Overfitting terjadi ketika model terlalu "terbiasa" dengan data training sehingga tidak dapat menggeneralisasi dengan baik pada data testing. Underfitting, di sisi lain, terjadi ketika model tidak mampu mempelajari pola yang ada dalam data training dengan cukup baik. Dengan mengubah proporsi data training dan data testing dapat mengidentifikasi apakah model cenderung mengalami overfitting atau underfitting.
- Pengujian dengan proporsi data training dan data testing yang berbeda memungkinkan untuk melihat bagaimana kinerja algoritma berubah ketika jumlah data training dan testing berubah. Hal ini membantu kita memahami sejauh mana algoritma dapat melakukan klasifikasi atau prediksi dengan akurasi yang konsisten.
- Dalam beberapa kasus, perubahan proporsi data training dan data testing dapat mempengaruhi kinerja model. Dengan melakukan pengujian menggunakan berbagai proporsi dapat menentukan proporsi optimal yang menghasilkan kinerja terbaik pada algoritma yang digunakan. Ini membantu dalam menyesuaikan model dengan cara yang lebih baik dan memaksimalkan hasil yang diinginkan.

- Ketidakseimbangan kelas terjadi ketika jumlah sampel dalam satu kelas jauh lebih banyak atau lebih sedikit daripada kelas lainnya. Dalam kasus ini, mengubah proporsi data training dan data testing dapat membantu dalam mengevaluasi dan mengatasi masalah ketidakseimbangan kelas dalam pengklasifikasian.

Percobaan pengujian dilakukan sebagai berikut:

- Percobaan 1: Algoritma C4.5 dengan proporsi data training dan data testing adalah 80:20.
- Percobaan 2: Algoritma C4.5 dengan proporsi data training dan data testing adalah 90:10.
- Percobaan 3: Algoritma C4.5 menggunakan validasi silang (cross-validation) yaitu k-fold cross-validation untuk membagi dataset menjadi k subset.
- Percobaan 4: Algoritma Neural Network dengan proporsi data training dan data testing adalah 80:20.
- Percobaan 5: Algoritma Neural Network dengan proporsi data training dan data testing adalah 90:10.
- Percobaan 6: Algoritma Neural Network menggunakan validasi silang (cross-validation) yaitu k-fold cross-validation untuk membagi dataset menjadi k subset.
- Percobaan 7: Algoritma Naive Bayes dengan proporsi data training dan data testing adalah 80:20.

- Percobaan 8: Algoritma Naive Bayes dengan proporsi data training dan data testing adalah 90:10.
- Percobaan 9: Algoritma Naïve Bayes menggunakan validasi silang (cross-validation) yaitu k-fold cross-validation untuk membagi dataset menjadi k subset.

Pengujian-pengujian ini memberikan variasi dalam proporsi data training dan data testing yang memungkinkan untuk memahami bagaimana performa model dapat berubah dengan perubahan proporsi tersebut. Dengan melibatkan pengujian tersebut dapat memperoleh pemahaman yang lebih komprehensif tentang kinerja algoritma dan memilih proporsi yang paling optimal untuk dataset.

d. Evaluasi dan Validasi

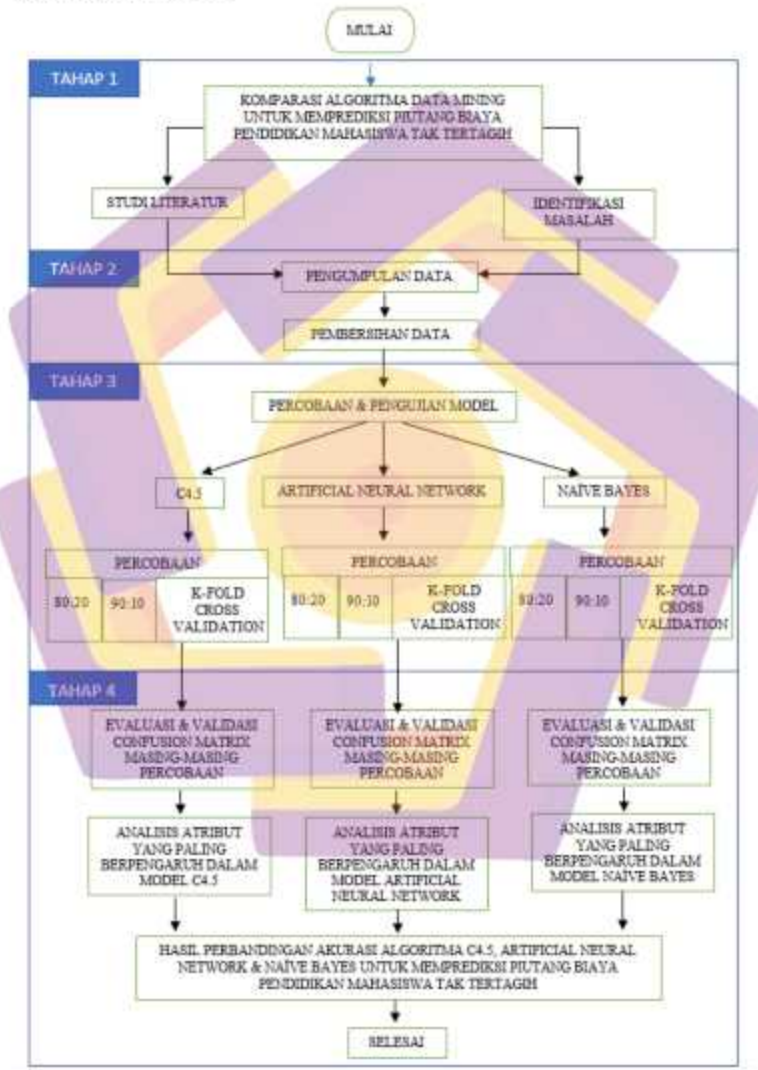
Tahap ini merupakan tahap untuk memperoleh akurasi yang terbaik dari algoritma-algoritma yang digunakan pada penelitian ini dengan menerapkan Confusion Matrix untuk melakukan perbandingan tingkat akurasi. Pada pengukuran kinerja menggunakan Confusion matrix terdapat 4 istilah sebagai representasi hasil proses klasifikasi, yaitu: True Positive (TP), True Negative (TN), False Positive (FS), dan False Negative. Perhitungan pada tahap ini akan menghasilkan 3 keluaran, yaitu: recall, precision, akurasi.

e. Analisis atribut yang paling berpengaruh dalam model.

Dalam tahapan analisis ini, akan diperiksa atribut-atribut yang digunakan dalam model prediksi untuk mengidentifikasi mana yang memiliki dampak atau kontribusi terbesar pada hasil prediksi.

3.4. Alur Penelitian

Secara garis besar langkah-langkah penelitian dapat dilihat dalam kerangka konseptual berikut ini.



Gambar 3.1. Diagram Alur Penelitian

Alur penelitian tersebut memiliki tahapan sebagai berikut:

1. Tahap satu: menentukan judul penelitian, mencari studi literatur terkait dengan penelitian yang akan dilakukan dan merumuskan permasalahan yang akan dianalisis dari gap penelitian studi literatur yang telah dikumpulkan serta memilih algoritma yang akan digunakan berdasarkan studi literatur sehingga dalam pemilihan algoritma tersebut didasarkan pada hasil ilmu yang dianalisa oleh penelitian sebelumnya.
2. Tahap dua: mengumpulkan data berupa data mahasiswa dan data piutang mahasiswa Institut Teknologi dan Bisnis Muhammadiyah Wakatobi kemudian membersihkan data yang tidak diperlukan serta menyesuaikan atribut apa yang akan digunakan dalam proses analisis data.
3. Tahap tiga: pada tahap ini akan dilakukan percobaan dan pengujian model yang digunakan pada penelitian ini yaitu algoritma C4.5, Neural Network dan Naïve Bayes untuk memprediksi piutang biaya kuliah mahasiswa tak tertagih. Caranya adalah membagi dataset menjadi dua bagian yaitu data testing dan data training dengan proporsi perbandingan pengujian 80:20, 90:10 dan menggunakan k-fold cross-validation untuk masing-masing algoritma C4.5, Artificial Neural Network dan Naïve Bayes.
4. Tahap empat: pada tahap ini akan menerapkan metode Confusion Matriks untuk memperoleh nilai akurasi masing algoritma yang digunakan dan menganalisis atribut yang berpengaruh dari masing-masing model kemudian nilai-nilai akurasi tersebut akan dibandingkan untuk mengetahui persentase nilai akurasi yang terbaik dari ketiga algoritma tersebut.

BAB IV

HASIL PENELITIAN DAN PEMBAHASAN

4.1. Pengumpulan Data

Metode klasifikasi yang digunakan dalam tahap ini adalah algoritma C4.5, Neural Network dan Naïve Bayes. Sebelum melanjutkan ke tahap pengujian, terlebih dahulu dilakukan tahap preprocessing data. Pada saat pengujian klasifikasi, hasil penelitian yang telah diuji dievaluasi, digunakan tiga kali percobaan yaitu percobaan pertama dengan proporsi data training dan data testing 80:20, percobaan kedua dengan proporsi data training dan data testing 90:10, dan percobaan ketiga dengan menggunakan k-fold cross validation atau validasi silang kemudian dilakukan tahap evaluasi menggunakan *Confusion Matrix*.

Data pertama yang digunakan adalah data mahasiswa yang diperoleh dari Pangkalan Data Pendidikan Tinggi (PDDikti) Institut Teknologi dan Bisnis Muhammadiyah Wakatobi Tahun Akademik 2020/2021, 2021/2022, dan 2022/2023. Data tersebut dapat dilihat pada Tabel 4.1 berikut ini.

Tabel 4.1 Data beberapa Mahasiswa dari PDDikti

NIM	Nama Mahasiswa	Status	Perguruan Tinggi	Program Studi	Jenjang	Alamat Keluarahan Ibu	Alamat Kecamatan	Pendidikan Wali	Pekerjaan Wali	Penghasilan Wali	Ket
220542470001	NIMIANTI	Aktif	Institut Teknologi dan Bisnis Muhammadiyah Wakatobi	Ilmu Perikanan	S1	Pada Raya Makmur	Wangi-Wangi	SD Sederajat	Petani	Kurang dari Rp 500.000	Valid
220542470002	ANITA	Aktif	Institut Teknologi dan Bisnis Muhammadiyah Wakatobi	Ilmu Perikanan	S1	Waesi	Wangi-Wangi	SD Sederajat	Nelayan	Kurang dari Rp 1.000.000	Valid
220542470003	RINTANG SUDARNO	Aktif	Institut Teknologi dan Bisnis Muhammadiyah Wakatobi	Ilmu Perikanan	S1	Mola Selatan	Wangi-Wangi Selatan	SMA Sederajat	Nelayan	Rp. 1.000.000 - Rp. 1.999.999	Valid
220542470004	MARLIF	Aktif	Institut Teknologi dan Bisnis Muhammadiyah Wakatobi	Ilmu Perikanan	S1	Pada Raya Makmur	Wangi-Wangi	Paket SD	Buruh Harian Lepas	Kurang dari Rp 500.000	Valid
220542470005	YASIR	Aktif	Institut Teknologi dan Bisnis Muhammadiyah Wakatobi	Ilmu Perikanan	S1	Mola Nelayan Bakti	Wangi-Wangi Selatan	SMA Sederajat	Nelayan	Kurang dari Rp 1.000.000	Valid
220542470006	MIFRI HANDAYANI	Aktif	Institut Teknologi dan Bisnis Muhammadiyah Wakatobi	Ilmu Perikanan	S1	Mola Samaturu	Wangi-Wangi Selatan	D4/S1	Wiraswasta	Rp. 1.000.000 - Rp. 1.999.999	Valid
220542470007	WAHYUNISA	Aktif	Institut Teknologi dan Bisnis Muhammadiyah Wakatobi	Ilmu Perikanan	S1	Mola Samaturu	Wangi-Wangi Selatan	SMA Sederajat	Nelayan	Kurang dari Rp 1.000.000	Valid
220542470008	KASMAIDIN	Aktif	Institut Teknologi dan Bisnis Muhammadiyah Wakatobi	Ilmu Perikanan	S1						Valid
220542470009	PITRIA A	Aktif	Institut Teknologi dan Bisnis Muhammadiyah Wakatobi	Ilmu Perikanan	S1	Mola Bahari	Wangi-Wangi Selatan	SMP Sederajat	Nelayan	Kurang dari Rp 1.000.000	Valid
220542470010	AHMAD AMIRUDIN	Aktif	Institut Teknologi dan Bisnis Muhammadiyah Wakatobi	Ilmu Perikanan	S1						Valid

Pada Tabel 4.1 menunjukkan beberapa data mahasiswa dari Pangkalan Data Pendidikan Tinggi melalui situs https://pddikti-admin. kemdikbud.go.id/admin/ untuk periode Tahun Akademik 2020/2021, 2021/2022 dan 2022/2023 berjumlah 260 record data.

Data kedua yang digunakan data piutang mahasiswa bersumber dari data internal Biro Administrasi Keuangan ITBMW untuk Tahun Anggaran 2021, 2022, dan 2023. Data yang telah diimport ke bentuk xls. Data Piutang Mahasiswa tersebut terdiri dari Data Piutang Biaya Sarana dan Prasarana Pembangunan terdiri dari 7 Atribut dan Data Piutang Uang Kuliah Tunggal terdiri dari 5 Atribut. Tabel 4.2 merupakan tampilan Data Piutang Biaya Sarana dan Prasarana Pembangunan.

Tabel 4.2 Data Piutang BPP

NIM	Nama Mahasiswa	Pembayaran DPP Uang Pendaftrn, Reg Ulang, Infak				
		Pendaftaran awal masuk	Registrasi Ulang	Infak	BPS	Total
220572010006	Fitriani	-	-	-	1,000,000	1,000,000
220572010003	Bambang	-	-	-	-	-
220572010008	Jamalu	-	500,000	300,000	2,000,000	2,800,000
220572010024	Wa Ewi	-	-	-	1,300,000	1,300,000
220572010026	Yuni	-	-	-	1,000,000	1,000,000
220542470003	Bintang S	-	-	300,000	2,000,000	2,300,000
220942020001	Rostiana	-	-	300,000	2,000,000	2,300,000
220942020002	Ani M	-	-	-	1,800,000	1,800,000
220572010017	Nurul Y	-	500,000	300,000	2,000,000	2,800,000
220572010021	Suhartono	-	-	300,000	2,000,000	2,300,000

Pada Tabel 4.2 menunjukkan beberapa data piutang mahasiswa untuk biaya sarana dan prasarana pembangunan untuk Tahun Anggaran 2020, 2021 dan 2022 berjumlah 260 record data.

Tabel 4.3 merupakan tampilan Data Piutang Uang Kuliah Tunggal. Tabel ini menampilkan informasi terkait piutang yang terkait dengan uang kuliah tunggal.

Tabel 4.3. Data Piutang UKT

NIM	Nama Mahasiswa	Jumlah Piutang	Penerimaan Piutang	Saldo Tahun
220572010006	Fitriani	5,000,000	4,000,000	1,000,000
220572010003	Bambang	5,000,000	4,000,000	1,000,000
220572010008	Jamalu	5,000,000	2,000,000	3,000,000
220572010024	Wa Ewi	5,000,000	5,000,000	-
220572010026	Yuni	5,000,000	4,500,000	500,000
220542470003	Bintang Sudarno	5,000,000	4,900,000	100,000
220942020001	Rostiana	5,000,000	5,000,000	-
220942020002	Ani Mazayaroh	5,000,000	5,000,000	-
220572010017	Nurul Yakin	5,000,000	1,000,000	4,000,000
220572010021	Suhartono	5,000,000	3,000,000	2,000,000

Pada Tabel 4.3, terdapat beberapa data piutang Uang Kuliah Tunggal (UKT) mahasiswa untuk Tahun Anggaran 2020, 2021, dan 2022. Jumlah total record data dalam tabel ini adalah sebanyak 260.

Hasil data setelah digabungkan dari dua sumber data utama menjadi satu dataset. Data hasil penggabungan per atribut dapat dilihat sebagai berikut ini:

```
print (dataset)
            NIM  Status  Perguruan Tinggi  Program Studi  Jenjang  \
0    220542470001       1                 1              1        1
1    220542470002       1                 1              1        1
2    220542470003       1                 1              1        1
3    220542470004       1                 1              1        1
4    220542470005       1                 1              1        1
..            ...     ...               ...            ...      ...
255  222942020023       1                 1              2        1
256  222942020024       1                 1              2        1
257  222942020025       1                 1              2        1
258  222942020026       1                 1              2        1
259  222942020029       1                 1              2        1
```

Tampilan di atas menunjukkan dataset pada Google Colaboratory dengan atribut output berupa keterangan status piutang dengan parameter 1) piutang tertagih dan 2) piutang tak tertagih.

Dataset terdiri dari 16 atribut dan 260 record data. 16 atribut tersebut terdiri dari 15 atribut input dan 1 atribut output atau target yaitu status piutang.

```
Hitung jumlah atribut
jumlah_atribut = len(dataset.columns) - 1  # Mengurangi 1
karena kolom terakhir adalah kolom target
print("Jumlah atribut:", jumlah_atribut)
Jumlah atribut: 15
Hitung jumlah record
jumlah_record = len(dataset)
print("Jumlah record:", jumlah_record)
Jumlah record: 260
Hitung jumlah itemset untuk setiap atribut
for column in dataset.columns:
    jumlah_itemset = dataset[column].nunique()
    print("Jumlah itemset untuk atribut", column, ":",
jumlah_itemset)
Jumlah itemset untuk atribut NIM : 259
Jumlah itemset untuk atribut Status : 1
Jumlah itemset untuk atribut Perguruan Tinggi : 1
Jumlah itemset untuk atribut Program Studi : 3
Jumlah itemset untuk atribut Jenjang : 1
Jumlah itemset untuk atribut Alamat Kelurahan /Desa : 50
Jumlah itemset untuk atribut Alamat Kecamatan : 7
Jumlah itemset untuk atribut Pendidikan Wali : 8
Jumlah itemset untuk atribut Pekerjaan Wali : 14
Jumlah itemset untuk atribut Penghasilan Wali : 8
Jumlah itemset untuk atribut Keterangan : 1
Jumlah itemset untuk atribut Jumlah Piutang UKT Mahasiswa : 31
Jumlah itemset untuk atribut Umur Piutang UKT Mahasiswa : 3
Jumlah itemset untuk atribut Jumlah Piutang BPP : 24
Jumlah itemset untuk atribut Umur Piutang BPP : 3
Jumlah itemset untuk atribut Status Piutang : 2
```

Tampilan ini menunjukkan jumlah atribut dataset berjumlah 16 dengan jumlah record data sebanyak 260. Jumlah itemset untuk masing-masing atribut dapat ditampilkan juga dalam bentuk tabel seperti Tabel 4.4 berikut ini.

Tabel 4.4 Jumlah Atribut yang digunakan

No	Atribut	Jumlah
1	NIM	259
2	Status	1
3	Perguruan Tinggi	1
4	Program Studi	3
5	Jenjang	1
6	Alamat Kelurahan/ Desa	50
7	Alamat Kecamatan	7
8	Pendidikan Wali	8
9	Pekerjaan Wali	14
10	Penghasilan Wali	8
11	Keterangan	1
12	Jumlah Piutang UKT Mahasiswa	31
13	Umur Piutang UKT Mahasiswa	3
14	Jumlah Piutang BPP	24
15	Umur Piutang BPP	3
16	Status Piutang	2

Berdasarkan Tabel 4.4 terdapat 16 atribut yang digunakan yaitu NIM terdapat 260 nilai, Status terdapat 1 nilai, Perguruan Tinggi terdapat 1 nilai, Program Studi 3 nilai, jenjang 1 nilai, alamat kelurahan/desa terdapat 50 nilai, Alamat Kecamatan 7 nilai, Pendidikan Wali 8 nilai, Pekerjaan Wali 14 nilai, keterangan 1 nilai, Jumlah Piutang UKT Mahasiswa 31 nilai, Umur Piutang UKT Mahasiswa 3 nilai, Jumlah Piutang BPP 24 nilai, umur piutang BPP terdapat 3 nilai dan Status piutang memiliki 2 nilai.

4.2. Pembersihan Data

Tahapan selanjutnya setelah memiliki dataset hasil penggabungan maka dilakukan proses preprocessing data dengan `StandardScaler`. Proses preprocessing yang dilakukan pada fitur-fitur numerik dengan menggunakan `StandardScaler` memiliki tujuan untuk menormalkan atau menstandarisasi nilai-

nilai fitur numerik. Ini dilakukan untuk menghilangkan perbedaan skala atau besaran yang mungkin ada di antara fitur-fitur numerik dalam dataset.

```
Melakukan preprocessing pada fitur-fitur numeric
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.transform(x_test)
Melakukan imputasi pada fitur-fitur yang memiliki missing
value
from sklearn.impute import SimpleImputer
imputer = SimpleImputer(strategy='mean')
x_train = imputer.fit_transform(x_train)
```

Proses di atas merupakan proses preprocessing data. Dengan melakukan preprocessing menggunakan `StandardScaler`, fitur-fitur numerik dalam dataset akan memiliki mean (rata-rata) sekitar 0 dan standar deviasi sekitar 1. Hal ini dapat membantu meningkatkan kinerja algoritma machine learning yang sensitif terhadap skala data.

Proses imputasi yang dilakukan pada fitur-fitur yang memiliki missing value menggunakan `SimpleImputer` bertujuan untuk mengisi nilai yang hilang (missing value) dengan nilai yang relevan. Hal ini dilakukan agar dataset lengkap dan dapat digunakan dalam analisis atau pemodelan lebih lanjut.

Dengan melakukan imputasi menggunakan `SimpleImputer`, fitur-fitur yang sebelumnya memiliki missing value akan memiliki nilai yang lengkap, sehingga dataset dapat digunakan secara keseluruhan untuk analisis dan pemodelan. Imputasi membantu mencegah kehilangan informasi yang berharga yang dapat mempengaruhi performa model machine learning atau analisis statistik yang menggunakan dataset tersebut.

4.3. Hasil Percobaan dan Pengujian Model menggunakan Algoritma C4.5.

Percobaan 1 adalah melakukan percobaan dengan menggunakan Algoritma C4.5 dengan proporsi data training dan data testing adalah 80:20. Pengujian dilakukan dengan cara membagi dataset menjadi dua bagian yaitu data testing dan data training dengan proporsi perbandingan pengujian 80:20.

```
Membagi dataset menjadi data latih dan data uji
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y,
test_size=0.20, random_state=0)
```

Proses tersebut menunjukkan proses pembagian dataset menggunakan train_test_split dengan proporsi 80% untuk data training dan 20% untuk data testing. Setelah proses pembagian dataset maka dilakukan percobaan dan pengujian model dengan menggunakan google colaboratory dapat ditampilkan sebagai berikut

```
Membangun model Decision Tree (C4.5)
from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier()
classifier.fit(x_train, y_train)
DecisionTreeClassifier
DecisionTreeClassifier()
Melakukan prediksi pada data uji
x_test = imputer.transform(x_test)
Gunakan imputer yang sama untuk data uji
y_pred = classifier.predict(x_test)
```

Proses tersebut merupakan proses menggunakan Algoritma C4.5 untuk melakukan prediksi pada data testing kemudian dilakukan proses imputasi menggunakan objek `imputer` yang sama yang telah digunakan untuk mengimputasi data training sebelumnya. Hal ini penting untuk memastikan konsistensi imputasi di antara data training dan data testing. Metode `predict()` digunakan untuk

melakukan prediksi menggunakan model Decision Tree yang telah dilatih sebelumnya. Prediksi dilakukan pada data training (`x_test`), dan hasil prediksi akan disimpan dalam variabel `y_pred`.

Hasil prediksi akan disimpan dalam variabel `y_pred` yang dapat digunakan untuk evaluasi performa model. Evaluasi model menggunakan confusion matrix seperti tampilan gambar berikut:

```
Evaluasi model menggunakan confusion matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
[[17  1]
 [ 6 28]]
```

Proses evaluasi model merupakan evaluasi model menggunakan Confusion Matrix. Hasil Confusion Matrix di atas jika dituliskan dalam bentuk tabel maka dapat ditampilkan seperti Tabel 4.5 di bawah ini:

Tabel 4.5 Tabel Confusion Matrix

	Prediksi Negatif (0)	Prediksi Positif (1)
Aktual Negatif	16	2
Aktual Positif	5	29

Dari confusion matrix di atas, kita dapat menghitung beberapa metrik evaluasi sebagai berikut:

a. Akurasi (Accuracy): Akurasi menggambarkan seberapa baik model dapat mengklasifikasikan dengan benar keseluruhan data. Rumus akurasi adalah:

```
scss
Akurasi = (TP + TN) / (TP + TN + FP + FN)
```

Dalam kasus ini, akurasi = (16 + 29) / (16 + 2 + 5 + 29) = 0.85 atau 85%.

b. Presisi (Precision): Presisi mengukur seberapa baik model dapat mengidentifikasi dengan benar positif dari semua prediksi positif. Rumus presisi adalah:

```
makefile
Presisi = TP / (TP + FP)
```

Dalam kasus ini, presisi = 16 / (16 + 5) = 0.76 atau 76%.

c. Recall (Sensitivitas atau True Positive Rate): Recall mengukur seberapa baik model dapat mendeteksi dengan benar positif dari semua data sebenarnya yang positif. Rumus recall adalah:

```
makefile
Recall = TP / (TP + FN)
```

Dalam kasus ini, recall = 16 / (16 + 2) = 0.89 atau 89%.

Percobaan 2 dilakukan dengan menggunakan Algoritma C4.5. Pengujian dilakukan dengan cara membagi dataset menjadi dua bagian yaitu data testing dan data training dengan proporsi perbandingan pengujian 90:10.

```
Membagi dataset menjadi data latih dan data uji
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y,
test_size=0.10, random_state=0)
```

Proses pembagian dataset menggunakan train_test_split dengan proporsi 90% untuk data training dan 10% untuk data testing.

Setelah proses pembagian dataset maka dilakukan percobaan dan pengujian model dengan menggunakan google colaboratory dapat ditampilkan sebagai berikut

```
Membangun model Decision Tree (C4.5)
from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier()
classifier.fit(x_train, y_train)
```

DecisionTreeClassifier

```
(Lanjutan)
DecisionTreeClassifier()
Melakukan prediksi pada data uji
x_test = imputer.transform(x_test)
Gunakan imputer yang sama untuk data uji
      y_pred = classifier.predict(x_test)
```

Proses menggunakan Algoritma C4.5 untuk melakukan prediksi pada data testing kemudian dilakukan proses imputasi menggunakan objek `imputer` yang sama yang telah digunakan untuk mengimputasi data training sebelumnya. Hal ini penting untuk memastikan konsistensi imputasi di antara data training dan data testing. Metode `predict()` digunakan untuk melakukan prediksi menggunakan model Decision Tree yang telah dilatih sebelumnya. Prediksi dilakukan pada data training (`x_test`), dan hasil prediksi akan disimpan dalam variabel `y_pred`.

Hasil prediksi akan disimpan dalam variabel `y_pred` yang dapat digunakan untuk evaluasi performa model. Evaluasi model menggunakan confusion matrix seperti tampilan gambar berikut:

```
Evaluasi model menggunakan confusion matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
[[ 7  0]
 [ 2 17]]
```

Proses evaluasi model merupakan evaluasi model menggunakan Confusion Matrix. Hasil Confusion Matrix di atas jika dituliskan dalam bentuk tabel maka dapat ditampilkan seperti Tabel 4.6 di bawah ini:

Tabel 4.6 Tabel Confusion Matrix:

	Prediksi Negatif (0)	**Prediksi Positif (1)**
Aktual Negatif	7	0
Aktual Positif	2	17

Dari confusion matrix di atas, kita dapat menghitung beberapa metrik evaluasi sebagai berikut:

a. Akurasi (Accuracy): Akurasi menggambarkan seberapa baik model dapat mengklasifikasikan dengan benar keseluruhan data. Rumus akurasi adalah:

```
scss
Akurasi = (TP + TN) / (TP + TN + FP + FN)
```

Dalam kasus ini, akurasi = (7 + 17) / (7 + 0 + 2 + 17) = 0.92 atau 92%.

b. Presisi (Precision): Presisi mengukur seberapa baik model dapat mengidentifikasi dengan benar positif dari semua prediksi positif. Rumus presisi adalah:

```
makefile
Presisi = TP / (TP + FP)
```

Dalam kasus ini, presisi = 7 / (7 + 2) = 0.78 atau 78%.

c. Recall (Sensitivitas atau True Positive Rate): Recall mengukur seberapa baik model dapat mendeteksi dengan benar positif dari semua data sebenarnya yang positif. Rumus recall adalah:

```
makefile
Recall = TP / (TP + FN)
```

Dalam kasus ini, recall = 7 / (7 + 0) = 1 atau 100%.

Percobaan 3 dilakukan dengan menggunakan Algoritma C4.5. Pengujian dilakukan dengan menggunakan validasi silang (cross-validation) yaitu k-fold cross-validation untuk membagi dataset menjadi k subset.

Inisialisasi model Decision Tree (C4.5) dan jumlah subset (fold) untuk k-fold cross-validation:

```
classifier = DecisionTreeClassifier()
k = 5  # Jumlah subset (fold)
kf = KFold(n_splits=k, shuffle=True, random_state=0)
```

Proses k-fold validation dengan k subset yaitu 5. Dataset dibagi menjadi 5 subset yang sama besar selanjutnya model dilatih dan diuji 5 kali, setiap kali menggunakan salah satu subset sebagai data testing dan k-1 subset lainnya sebagai data training.

Setelah proses pembagian dataset, dilakukan proses k-fold cross-validation yang dapat ditampilkan sebagai berikut:

Melakukan k-fold cross-validation

```
for train_index, test_index in kf.split(x):
    x_train, x_test = x[train_index], x[test_index]
    y_train, y_test = y[train_index], y[test_index]
    # Melatih model pada subset train
    classifier.fit(x_train, y_train)
    # Prediksi menggunakan subset test
    y_pred = classifier.predict(x_test)
    # Menghitung confusion matrix pada setiap iterasi
    cm = confusion_matrix(y_test, y_pred)
    # Menghitung dan menyimpan akurasi, presisi, dan recall
pada setiap iterasi
    accuracy = np.mean(y_pred == y_test)
    precision = precision_score(y_test, y_pred)
    recall = recall_score(y_test, y_pred)

    accuracies.append(accuracy)
    precisions.append(precision)
        recalls.append(recall)
```

Proses di atas merupakan proses menguji dan mengevaluasi kinerja model algoritma C4.5, pertama-tama menggunakan cross-validation dari subset k atau fold. Untuk melakukan ini, KFold digunakan untuk membagi data menjadi subset k, di mana masing-masing subset digunakan sebagai data testing dan data training. Selanjutnya, melatih model C4.5 (fit) dengan menggunakan data training (x_train, y_train) dari setiap subset, dan kemudian dilakukan prediksi (predict) pada data testing (x_test) dan menyimpan hasil prediksi dalam y_pred. Kemudian, dengan

menggunakan fungsi confusion_matrix dari scikit-learn, dapat dihitung confusion matrix untuk setiap iterasi dari hasil prediksi dan nilai sebenarnya (y_test).

Tahapan selanjutnya adalah menghitung rata-rata akurasi, presisi dan recall yang dihasilkan dari seluruh akurasi masing-masing subset. Proses tersebut dapat ditunjukkan sebagai berikut:

Menghitung rata-rata akurasi, presisi, dan recall dari semua iterasi

```
mean_accuracy = np.mean(accuracies)
mean_precision = np.mean(precisions)
mean_recall = np.mean(recalls)
print(f"Mean Accuracy: {mean_accuracy}")
print(f"Mean Precision: {mean_precision}")
print(f"Mean Recall: {mean_recall}")
Mean Accuracy: 0.8307692307692307
Mean Precision: 0.7802246418722392
Mean Recall: 0.8185635366413399
```

Proses di atas menggambarkan proses untuk menghitung dan menampilkan rata-rata akurasi, presisi dan recall dari semua iterasi. Hasil yang diperoleh yaitu rata-rata akurasi sebesar 0.83 atau 83 persen, rata-rata presisi sebesar 0.78 atau 78 persen dan rata-rata recall diperoleh sebesar 0.82 atau 82 persen.

Tahap berikutnya dari pengujian model menggunakan Algoritma C4.5 adalah menganalisis atribut yang paling berpengaruh terhadap model. Proses ini dapat ditunjukkan sebagai berikut

```
Mendapatkan feature importance
feature_importance = classifier.feature_importances_
Menampilkan feature importance untuk setiap atribut
for i, importance in enumerate(feature_importance):
    print(f'Feature {i}: {importance}')
Feature 0: 0.020561080852968094
Feature 1: 0.0
Feature 2: 0.0
Feature 3: 0.016699604743083
Feature 4: 0.0
Feature 5: 0.02236406021795304
```

```
(Lanjutan)
Feature 6: 0.026857691363765514
Feature 7: 0.057334799861973 77
Feature 8: 0.05474465148378192
Feature 9: 0.014130434782608694
Feature 10: 0.0
Feature 11: 0.6597544914645002
Feature 12: 0.0029438405797101446
Feature 13: 0.12460934464965569
Feature 14: 0.0
```

Hasil analisis atribut yang paling berpengaruh dalam algoritma C4.5 adalah berdasarkan "*feature importance*" yang telah dihitung untuk masing-masing atribut. Nilai "*feature importance*" ini mengindikasikan sejauh mana masing-masing atribut mempengaruhi pembuatan keputusan dalam model C4.5. Berikut adalah interpretasi hasil analisis atribut:

a) Feature 0 (NIM): 0.0206

Atribut NIM memiliki "*feature importance*" sekitar 2.06%. Ini menunjukkan bahwa atribut NIM memiliki kontribusi positif yang cukup rendah terhadap pembuatan keputusan dalam model C4.5.

b) Feature 1 (Status): 0.0

Atribut Status memiliki "*feature importance*" 0.0, yang berarti atribut ini tidak memberikan kontribusi apa pun dalam pembuatan keputusan dalam model. Ini mungkin karena atribut ini tidak memiliki peran signifikan dalam konteks permasalahan yang dipecahkan oleh model.

c) Feature 2 (Perguruan Tinggi): 0.0

Atribut Perguruan Tinggi memiliki "*feature importance*" 0.0, yang menunjukkan bahwa atribut ini tidak memiliki kontribusi dalam pembuatan keputusan model.

d) Feature 3 (Program Studi): 0.0167

Atribut Program Studi memiliki "*feature importance*" sekitar 1.67%. Ini menunjukkan bahwa atribut Program Studi memiliki kontribusi positif yang rendah terhadap pembuatan keputusan dalam model.

e) Feature 4 (Jenjang): 0.0

Atribut Jenjang memiliki "*feature importance*" 0.0, yang menunjukkan bahwa atribut ini tidak memengaruhi pembuatan keputusan model.

f) Feature 5 (Alamat Kelurahan/Desa): 0.0224

Atribut Alamat Kelurahan/Desa memiliki "*feature importance*" sekitar 2.24%. Ini menunjukkan bahwa atribut ini memiliki kontribusi positif yang cukup rendah terhadap pembuatan keputusan dalam model.

g) Feature 6 (Alamat Kecamatan): 0.0269

Atribut Alamat Kecamatan memiliki "*feature importance*" sekitar 2.69%. Ini menunjukkan bahwa atribut ini memiliki kontribusi positif yang cukup rendah terhadap pembuatan keputusan dalam model.

h) Feature 7 (Pendidikan Wali): 0.0573

Atribut Pendidikan Wali memiliki "*feature importance*" sekitar 5.73%. Ini menunjukkan bahwa atribut ini memiliki kontribusi positif yang cukup signifikan terhadap pembuatan keputusan dalam model.

i) Feature 8 (Pekerjaan Wali): 0.0547

Atribut Pekerjaan Wali memiliki "*feature importance*" sekitar 5.47%. Ini menunjukkan bahwa atribut ini memiliki kontribusi positif yang cukup signifikan terhadap pembuatan keputusan dalam model.

j) Feature 9 (Penghasilan Wali): 0.0141

Atribut Penghasilan Wali memiliki "*feature importance*" sekitar 1.41%. Ini menunjukkan bahwa atribut ini memiliki kontribusi positif yang rendah terhadap pembuatan keputusan dalam model.

k) Feature 10 (Keterangan): 0.0

Atribut Keterangan memiliki "*feature importance*" 0.0, yang menunjukkan bahwa atribut ini tidak memengaruhi pembuatan keputusan model.

l) Feature 11 (Jumlah Piutang UKT mahasiswa): 0.6598

Atribut Jumlah Piutang UKT mahasiswa memiliki "*feature importance*" sekitar 65.98%. Ini adalah atribut yang paling berpengaruh dalam pembuatan keputusan dalam model C4.5, dengan kontribusi yang sangat signifikan.

m) Feature 12 (Umur Piutang UKT Mahasiswa): 0.0029

Atribut Umur Piutang UKT Mahasiswa memiliki "*feature importance*" sekitar 0.29%. Kontribusinya cukup rendah.

n) Feature 13 (Jumlah Piutang BPP): 0.1246

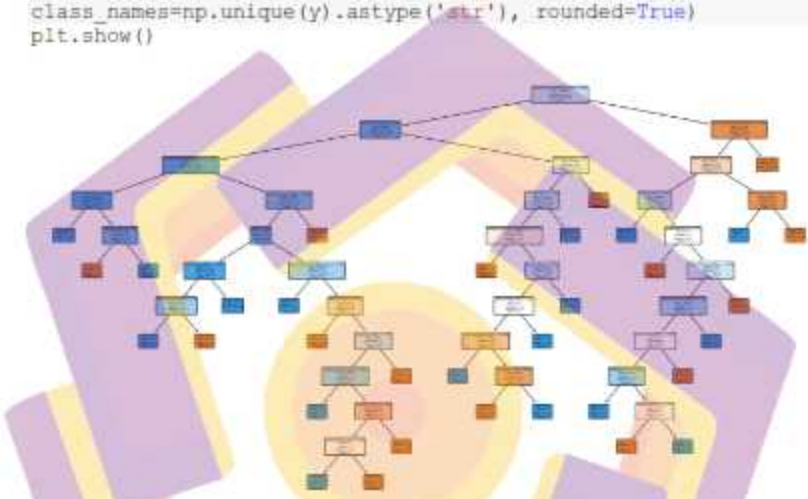
Atribut Jumlah Piutang BPP memiliki "*feature importance*" sekitar 12.46%. Ini adalah salah satu atribut yang berpengaruh dalam pembuatan keputusan dalam model.

o) Feature 14 (Umur Piutang BPP): 0.0

Atribut Umur Piutang BPP memiliki "*feature importance*" 0.0, yang menunjukkan bahwa atribut ini tidak memberikan kontribusi apa pun dalam pembuatan keputusan model.

Atribut yang paling berpengaruh sangat mudah dikenali karena muncul pada bagian atas pohon keputusan. Untuk menampilkan pohon keputusan sebagai berikut

```
Munculkan gambar pohon keputusan
plt.figure(figsize=(20, 10))
plot_tree(classifier, filled=True,
feature_names=dataset.columns[:-1],
class_names=np.unique(y).astype('str'), rounded=True)
plt.show()
```



Gambar 4.1 Pohon Keputusan

Hasil gambar pohon keputusan menunjukkan bahwa atribut "Jumlah Piutang UKT Mahasiswa" di puncak pohon keputusan serta diikuti oleh atribut "Pendidikan Wali" dan "Pekerjaan Wali". Hal ini sejalan dengan hasil *feature importance* yang telah dianalisis sebelumnya.

4.4. Hasil Percobaan dan Pengujian Model menggunakan Algoritma Artificial Neural Network.

Percobaan 4 dilakukan dengan menggunakan Algoritma Artificial Neural Network. Pengujian dilakukan dengan cara membagi dataset menjadi dua bagian yaitu data testing dan data training dengan proporsi perbandingan pengujian 80:20.

Membagi dataset menjadi data latih dan data uji

```
x_train, x_test, y_train, y_test = train_test_split(x, y,
test_size=0.20, random_state=5)
```

Proses di atas menunjukkan proses pembagian dataset menggunakan train_test_split dengan proporsi 80% untuk data training dan 20% untuk data testing.

Tahapan selanjutnya menerapkan teknik oversampling untuk menangani ketidakseimbangan kelas dalam data, proses tersebut dapat ditampilkan pada gambar berikut

Terapkan oversampling untuk menangani ketidakseimbangan kelas

```
oversample = RandomOverSampler(sampling_strategy='minority')
x, y = oversample.fit_resample(x, y)
```

Tahapan ini merupakan proses yang harus dikerjakan jika terdapat ketidakseimbangan kelas dalam data, dimana kelas positif memiliki jumlah yang jauh lebih sedikit daripada kelas negatif.

Setelah proses pembagian dataset dan oversampling maka dilakukan percobaan dan pengujian model dengan menggunakan google colaboratory dapat ditampilkan sebagai berikut:

Membangun model Neural Network dengan TensorFlow

```
model = tf.keras.models.Sequential([
    tf.keras.layers.Dense(128, activation='relu',
input_shape=(x_train.shape[1],)),
    tf.keras.layers.Dropout(0.5),  # Add dropout layer
```

```
(Lanjutan)
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])
```

Mengompilasi model

```
model.compile(optimizer='adam',      loss='binary_crossentropy',
metrics=['accuracy'])
```

Tahapan ini merupakan proses membangun arsitektur model Artificial Neural Network. Arsitektur model Sequential dibangun dengan menggunakan beberapa lapisan Dense. Lapisan pertama memiliki 128 neuron dengan fungsi aktivasi ReLU, yang sesuai dengan jumlah fitur pada input. Kemudian menambahan lapisan dropout untuk mencegah overfitting. Lapisan kedua juga memiliki 64 neuron dengan fungsi aktivasi ReLU. Lapisan terakhir memiliki 1 neuron dengan fungsi aktivasi sigmoid, yang digunakan untuk masalah klasifikasi biner. Model dikompilasi dengan menggunakan optimizer Adam, loss function binary_crossentropy (karena masalah klasifikasi biner), dan metrik evaluasi accuracy.

Proses di bawah ini adalah proses melatih model dengan menggunakan data training (`x_train` dan `y_train`). Data dipecah menjadi batch dengan ukuran 32, dan dilakukan pelatihan selama 20 epoch. Proses pelatihan akan mengupdate parameter model berdasarkan loss function yang digunakan dan optimizer yang ditentukan. Model yang telah dilatih digunakan untuk melakukan prediksi pada data uji (`x_test`).

Mengompilasi model

```
model.compile(optimizer='adam', loss='binary_crossentropy',
metrics=['accuracy'])
```

(Lanjutan)

Melatih model

```
model.fit(x_train, y_train, batch_size=32, epochs=20,
validation_split=0.1, verbose=1)
```

```
Epoch 1/20
8/8 [==============================] - 1s 29ms/step - loss:
0.6464 - accuracy: 0.1556 - val_loss: -0.1339 - val_accuracy:
0.3462
Epoch 2/20
8/8 [==============================] - 0s 9ms/step - loss: -
0.1351 - accuracy: 0.4800 - val_loss: -0.8830 - val_accuracy:
0.3462
Epoch 3/20
8/8 [==============================] - 0s 7ms/step - loss: -
0.8014 - accuracy: 0.5067 - val_loss: -1.6413 - val_accuracy:
0.3462
Epoch 4/20
8/8 [==============================] - 0s 7ms/step - loss: -
1.5192 - accuracy: 0.5067 - val_loss: -2.4762 - val_accuracy:
0.3462
Epoch 5/20
8/8 [==============================] - 0s 7ms/step - loss: -
2.1260 - accuracy: 0.5067 - val_loss: -3.4135 - val_accuracy:
0.3462
Epoch 6/20
8/8 [==============================] - 0s 7ms/step - loss: -
3.0494 - accuracy: 0.5067 - val_loss: -4.5466 - val_accuracy:
0.3462
Epoch 7/20
8/8 [==============================] - 0s 7ms/step - loss: -
3.9483 - accuracy: 0.5067 - val_loss: -5.9892 - val_accuracy:
0.3462
Epoch 8/20
8/8 [==============================] - 0s 7ms/step - loss: -
5.3067 - accuracy: 0.5067 - val_loss: -7.8136 - val_accuracy:
0.3462
Epoch 9/20
8/8 [==============================] - 0s 7ms/step - loss: -
6.8885 - accuracy: 0.5067 - val_loss: -10.0128 - val_accuracy:
0.3462
Epoch 10/20
8/8 [==============================] - 0s 7ms/step - loss: -
8.6311 - accuracy: 0.5067 - val_loss: -12.5241 - val_accuracy:
0.3462
Epoch 11/20
```

```
(Lanjutan)
8/8 [==============================] - 0s 7ms/step - loss: -
10.7383 - accuracy: 0.5067 - val_loss: -15.4577 -
val_accuracy: 0.3462
Epoch 12/20
8/8 [==============================] - 0s 7ms/step - loss: -
13.5027 - accuracy: 0.5067 - val_loss: -19.0799 -
val_accuracy: 0.3462
Epoch 13/20
8/8 [==============================] - 0s 8ms/step - loss: -
16.7328 - accuracy: 0.5067 - val_loss: -23.6218 -
val_accuracy: 0.3462
Epoch 14/20
8/8 [==============================] - 0s 8ms/step - loss: -
19.9161 - accuracy: 0.5067 - val_loss: -29.1703 -
val_accuracy: 0.3462
Epoch 15/20
8/8 [==============================] - 0s 6ms/step - loss: -
25.1071 - accuracy: 0.5067 - val_loss: -36.0063 -
val_accuracy: 0.3462
Epoch 16/20
8/8 [==============================] - 0s 6ms/step - loss: -
30.8245 - accuracy: 0.5067 - val_loss: -43.9665 -
val_accuracy: 0.3462
Epoch 17/20
8/8 [==============================] - 0s 7ms/step - loss: -
38.0286 - accuracy: 0.5067 - val_loss: -54.0498 -
val_accuracy: 0.3462
Epoch 18/20
8/8 [==============================] - 0s 7ms/step - loss: -
44.8369 - accuracy: 0.5067 - val_loss: -65.7843 -
val_accuracy: 0.3462
Epoch 19/20
8/8 [==============================] - 0s 7ms/step - loss: -
55.5955 - accuracy: 0.5067 - val_loss: -79.9842 -
val_accuracy: 0.3462
Epoch 20/20
8/8 [==============================] - 0s 7ms/step - loss: -
68.9124 - accuracy: 0.5067 - val_loss: -96.1762 -
val_accuracy: 0.3462
```

Prediksi dilakukan dengan menggunakan metode `predict()`, yang menghasilkan probabilitas prediksi kelas positif (dalam hal ini, > 0.5 dianggap

sebagai kelas positif). Hasil prediksi ini kemudian diubah menjadi bentuk biner (0 atau 1) menggunakan threshold 0.5. Selama proses pelatihan, setiap epoch akan mencetak informasi seperti loss dan accuracy untuk membantu memonitor performa model. Hasil evaluasi model, termasuk accuracy, juga dapat dievaluasi dengan menggunakan metrik evaluasi lainnya dan melalui analisis matriks konfusi.

Hasil prediksi akan disimpan dalam variabel `y_pred` yang dapat digunakan untuk evaluasi performa model. Evaluasi model menggunakan confusion matrix seperti tampilan gambar berikut:

```
Evaluasi model menggunakan confusion matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
[[34  0]
[29  0]]
```

Proses evaluasi model tersebut merupakan evaluasi model menggunakan Confusion Matrix. Hasil Confusion Matrix di atas jika dituliskan dalam bentuk tabel maka dapat ditampilkan seperti Tabel 4.7 di bawah ini:

Tabel 4.7 Tabel Confusion Matrix:

	Prediksi Negatif (0)	Prediksi Positif (1)
Aktual Negatif	34	0
Aktual Positif	29	0

Dari confusion matrix di atas, kita dapat menghitung beberapa metrik evaluasi sebagai berikut:

a. Akurasi (Accuracy): Akurasi menggambarkan seberapa baik model dapat mengklasifikasikan dengan benar keseluruhan data. Rumus akurasi adalah:

```
scss
Akurasi = (TP + TN) / (TP + TN + FP + FN)
```

Dalam kasus ini, akurasi = (34 + 0) / (34 + 0 + 29 + 0) = 0.54 atau 54%.

b. Presisi (Precision): Presisi mengukur seberapa baik model dapat mengidentifikasi dengan benar positif dari semua prediksi positif. Rumus presisi adalah:

```
makefile
Presisi = TP / (TP + FP)
```

Dalam kasus ini, presisi = 34 / (34 + 29) = 0.54 atau 54%.

c. Recall (Sensitivitas atau True Positive Rate): Recall mengukur seberapa baik model dapat mendeteksi dengan benar positif dari semua data sebenarnya yang positif. Rumus recall adalah:

```
makefile
Recall = TP / (TP + FN)
```

Dalam kasus ini, recall = 34 / (34 + 0) = 1 atau 100%.

Percobaan 5 dilakukan dengan menggunakan Algoritma Artificial Neural Network. Pengujian dilakukan dengan cara membagi dataset menjadi dua bagian yaitu data testing dan data training dengan proporsi perbandingan pengujian 90:10.

Membagi dataset menjadi data latih dan data uji

```
x_train, x_test, y_train, y_test = train_test_split(x, y,
test_size=0.10, random_state=0)
```

Tahapan ini menunjukkan proses pembagian dataset menggunakan train_test_split dengan proporsi 90% untuk data training dan 10% untuk data testing.

Tahapan selanjutnya menerapkan teknik oversampling untuk menangani ketidakseimbangan kelas dalam data, proses tersebut dapat ditampilkan pada gambar berikut

Terapkan oversampling untuk menangani ketidakseimbangan kelas

```
oversample = RandomOverSampler(sampling_strategy='minority')
x, y = oversample.fit_resample(x, y)
```

Proses ini merupakan proses yang harus dikerjakan jika terdapat ketidakseimbangan kelas dalam data, dimana kelas positif memiliki jumlah yang jauh lebih sedikit daripada kelas negatif.

Setelah proses pembagian dataset dan proses oversampling maka dilakukan percobaan dan pengujian model dengan menggunakan google colaboratory dapat ditampilkan sebagai berikut:

Membangun model Neural Network dengan TensorFlow

```
model = tf.keras.models.Sequential([
    tf.keras.layers.Dense(128, activation='relu',
input_shape=(x_train.shape[1],)),
    tf.keras.layers.Dropout(0.5),  # Add dropout layer
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])
```

Mengompilasi model

```
model.compile(optimizer='adam',      loss='binary_crossentropy',
metrics=['accuracy'])
```

Proses di atas merupakan proses membangun arsitektur model Artificial Neural Network. Arsitektur model Sequential dibangun dengan menggunakan beberapa lapisan Dense. Lapisan pertama memiliki 128 neuron dengan fungsi aktivasi ReLU, yang sesuai dengan jumlah fitur pada input. Kemudian menambahan lapisan dropout untuk mencegah overfitting. Lapisan kedua juga memiliki 64 neuron dengan fungsi aktivasi ReLU. Lapisan terakhir memiliki 1 neuron dengan fungsi aktivasi sigmoid, yang digunakan untuk masalah klasifikasi

biner. Model dikompilasi dengan menggunakan optimizer Adam, loss function binary_crossentropy (karena masalah klasifikasi biner), dan metrik evaluasi accuracy.

Tahapn di bawah ini adalah proses melatih model dengan menggunakan data training (`x_train` dan `y_train`). Data dipecah menjadi batch dengan ukuran 32, dan dilakukan pelatihan selama 20 epoch. Proses pelatihan akan mengupdate parameter model berdasarkan loss function yang digunakan dan optimizer yang ditentukan. Model yang telah dilatih digunakan untuk melakukan prediksi pada data uji (`x_test`).

Melatih model

```
model.fit(x_train, y_train, batch_size=32, epochs=20,
validation_split=0.1, verbose=1)
```

```
Epoch 1/20
8/8 [==============================] - 1s 28ms/step - loss:
0.4226 - accuracy: 0.3399 - val_loss: -0.2291 - val_accuracy:
0.4483
Epoch 2/20
8/8 [==============================] - 0s 6ms/step - loss: -
0.4311 - accuracy: 0.4980 - val_loss: -0.9836 - val_accuracy:
0.4483
Epoch 3/20
8/8 [==============================] - 0s 9ms/step - loss: -
1.1508 - accuracy: 0.4980 - val_loss: -1.7588 - val_accuracy:
0.4483
Epoch 4/20
8/8 [==============================] - 0s 6ms/step - loss: -
1.8408 - accuracy: 0.4980 - val_loss: -2.6762 - val_accuracy:
0.4483
Epoch 5/20
8/8 [==============================] - 0s 10ms/step - loss: -
2.7302 - accuracy: 0.4980 - val_loss: -3.8215 - val_accuracy:
0.4483
Epoch 6/20
```

```
(Lanjutan)
8/8 [==============================] - 0s 9ms/step - loss: -
3.9099 - accuracy: 0.4980 - val_loss: -5.2894 - val_accuracy:
0.4483
Epoch 7/20
8/8 [==============================] - 0s 7ms/step - loss: -
5.5114 - accuracy: 0.4980 - val_loss: -7.2041 - val_accuracy:
0.4483
Epoch 8/20
8/8 [==============================] - 0s 7ms/step - loss: -
7.6291 - accuracy: 0.4980 - val_loss: -9.7327 - val_accuracy:
0.4483
Epoch 9/20
8/8 [==============================] - 0s 6ms/step - loss: -
9.9724 - accuracy: 0.4980 - val_loss: -12.9868 - val_accuracy:
0.4483
Epoch 10/20
8/8 [==============================] - 0s 6ms/step - loss: -
12.9874 - accuracy: 0.4980 - val_loss: -17.0872 -
val_accuracy: 0.4483
Epoch 11/20
8/8 [==============================] - 0s 6ms/step - loss: -
17.6413 - accuracy: 0.4980 - val_loss: -22.0764 -
val_accuracy: 0.4483
Epoch 12/20
8/8 [==============================] - 0s 7ms/step - loss: -
22.4693 - accuracy: 0.4980 - val_loss: -28.4279 -
val_accuracy: 0.4483
Epoch 13/20
8/8 [==============================] - 0s 6ms/step - loss: -
29.3386 - accuracy: 0.4980 - val_loss: -36.1115 -
val_accuracy: 0.4483
Epoch 14/20
8/8 [==============================] - 0s 7ms/step - loss: -
35.3893 - accuracy: 0.4980 - val_loss: -45.5943 -
val_accuracy: 0.4483
Epoch 15/20
8/8 [==============================] - 0s 7ms/step - loss: -
45.3758 - accuracy: 0.4980 - val_loss: -56.5992 -
val_accuracy: 0.4483
Epoch 16/20
8/8 [==============================] - 0s 7ms/step - loss: -
56.8195 - accuracy: 0.4980 - val_loss: -69.9936 -
val_accuracy: 0.4483
Epoch 17/20
```

```
(Lanjutan)
8/8 [==============================] - 0s 7ms/step - loss: -
69.9285 - accuracy: 0.4980 - val_loss: -85.6022 -
val_accuracy: 0.4483
Epoch 18/20
8/8 [==============================] - 0s 9ms/step - loss: -
85.1883 - accuracy: 0.4980 - val_loss: -103.3277 -
val_accuracy: 0.4483
Epoch 19/20
8/8 [==============================] - 0s 6ms/step - loss: -
101.3752 - accuracy: 0.4980 - val_loss: -124.6510 -
val_accuracy: 0.4483
Epoch 20/20
8/8 [==============================] - 0s 6ms/step - loss: -
122.7431 - accuracy: 0.4980 - val_loss: -148.6072 -
val_accuracy: 0.4483
<keras.src.callbacks.History at 0x7e3dfeac2500>
```

Prediksi dilakukan dengan menggunakan metode `predict()`, yang menghasilkan probabilitas prediksi kelas positif (dalam hal ini, > 0.5 dianggap sebagai kelas positif). Hasil prediksi ini kemudian diubah menjadi bentuk biner (0 atau 1) menggunakan threshold 0.5. Selama proses pelatihan, setiap epoch akan mencetak informasi seperti loss dan accuracy untuk membantu memonitor performa model. Hasil evaluasi model, termasuk accuracy, juga dapat dievaluasi dengan menggunakan metrik evaluasi lainnya dan melalui analisis matriks konfusi.

Hasil prediksi akan disimpan dalam variabel `y_pred` yang dapat digunakan untuk evaluasi performa model. Evaluasi model menggunakan confusion matrix seperti tampilan gambar berikut:

Evaluasi model menggunakan confusion matrix

```
cm = confusion_matrix(y_test, y_pred)
print(cm)
[[18  0]
[14  0]]
```

Proses evaluasi model pada tahapan di atas merupakan evaluasi model menggunakan Confusion Matrix. Hasil Confusion Matrix di atas jika dituliskan dalam bentuk tabel maka dapat ditampilkan seperti Tabel 4.8 di bawah ini:

Tabel 4.8 Tabel Confusion Matrix:

	Prediksi Negatif (0)	Prediksi Positif (1)
Aktual Negatif	18	0
Aktual Positif	14	0

Dari confusion matrix di atas, kita dapat menghitung beberapa metrik evaluasi sebagai berikut:

a. Akurasi (Accuracy): Akurasi menggambarkan seberapa baik model dapat mengklasifikasikan dengan benar keseluruhan data. Rumus akurasi adalah:

```
scss
Akurasi = (TP + TN) / (TP + TN + FP + FN)
```

Dalam kasus ini, akurasi = (18 + 0) / (18 + 0 + 14 + 0) = 0.56 atau 56%.

b. Presisi (Precision): Presisi mengukur seberapa baik model dapat mengidentifikasi dengan benar positif dari semua prediksi positif. Rumus presisi adalah:

```
makefile
Presisi = TP / (TP + FP)
```

Dalam kasus ini, presisi = 18 / (18 + 14) = 0.56 atau 56%.

c. Recall (Sensitivitas atau True Positive Rate): Recall mengukur seberapa baik model dapat mendeteksi dengan benar positif dari semua data sebenarnya yang positif. Rumus recall adalah:

```
makefile
Recall = TP / (TP + FN)
```

Dalam kasus ini, recall = 18 / (18 + 0) = 1 atau 100%.

Percobaan 6 dilakukan dengan menggunakan Algoritma Artificial Neural Network. Pengujian dilakukan dengan validasi silang (cross-validation) yaitu k-fold cross-validation untuk membagi dataset menjadi k subset.

Proses di bawah ini menggunakan MLPClassifier dari scikit-learn untuk memulai model Artificial Neural Network (ANN). Model ini memiliki seratus unit neuron di bawah lapisan tersembunyi (hidden layer).

Inisialisasi model Artificial Neural Network (ANN) dan jumlah subset (fold) untuk k-fold cross-validation

```
classifier = MLPClassifier(hidden_layer_sizes=(100,),
max_iter=1000, random_state=0)
k = 5  # Jumlah subset (fold)
kf = KFold(n_splits=k, shuffle=True, random_state=0)
```

Ditetapkan batas iterasi maksimum 1000 dan mengatur state random untuk menjamin hasil yang konsisten. Selanjutnya, digunakan metode cross-validation k-fold dengan k=5, yang berarti data akan dibagi menjadi lima subset atau fold. KFold digunakan untuk membagi data menjadi k bagian yang saling bersilangan secara bergantian. Dengan menggunakan shuffle=True, data diacak sebelum dibagi menjadi subset. Ini membantu mengurangi bias yang mungkin terjadi karena pola urutan data. Oleh karena itu, setiap fold akan digunakan secara bergantian sebagai data training dan testing, dan proses ini akan diulangi lima kali.

Setelah proses pembagian data training dan data testing sesuai fold atau subset, dilakukan proses k-fold cross-validation yang dapat ditampilkan sebagai berikut:

Melakukan k-fold cross-validation

```
for train_index, test_index in kf.split(x):
    x_train, x_test = x[train_index], x[test_index]
    y_train, y_test = y[train_index], y[test_index]

    # Melatih model pada subset train
    classifier.fit(x_train, y_train)

    # Prediksi menggunakan subset test
    y_pred = classifier.predict(x_test)

    # Menghitung confusion matrix pada setiap iterasi
    cm = confusion_matrix(y_test, y_pred)

    # Menghitung dan menyimpan akurasi, presisi, dan recall
pada setiap iterasi
    accuracy = np.mean(y_pred == y_test)
    precision = precision_score(y_test, y_pred)
    recall = recall_score(y_test, y_pred)

    accuracies.append(accuracy)
    precisions.append(precision)
    recalls.append(recall)
```

Tahapan tersebut merupakan proses menguji dan mengevaluasi kinerja model algoritma Artificial Neural Network (ANN), pertama-tama menggunakan cross-validation dari subset k atau fold. Untuk melakukan ini, K Fold digunakan untuk membagi data menjadi subset k, di mana masing-masing subset digunakan sebagai data testing dan data training. Selanjutnya, melatih model Artificial Neural Network (ANN) (fit) dengan menggunakan data training (x_train, y_train) dari setiap subset, dan kemudian dilakukan prediksi (predict) pada data testing (x_test) dan menyimpan hasil prediksi dalam y_pred. Kemudian, dengan menggunakan fungsi confusion_matrix dari scikit-learn, dapat dihitung confusion matrix untuk setiap iterasi dari hasil prediksi dan nilai sebenarnya (y_test).

Tahapan selanjutnya adalah menghitung rata-rata akurasi, presisi dan recall yang dihasilkan dari seluruh akurasi masing-masing subset. Proses tersebut dapat ditunjukkan sebagai berikut:

Menghitung rata-rata akurasi, presisi, dan recall dari semua iterasi

```
mean_accuracy = np.mean(accuracies)
mean_precision = np.mean(precisions)
mean_recall = np.mean(recalls)

print(f"Mean Accuracy: {mean_accuracy}")
print(f"Mean Precision: {mean_precision}")
print(f"Mean Recall: {mean_recall}")
Mean Accuracy: 0.8576923076923076
Mean Precision: 0.8479904306220096
Mean Recall: 0.7889671477772164
```

Tahapan ini menggambarkan proses untuk menghitung dan menampilkan rata-rata akurasi, presisi dan recall dari semua iterasi. Hasil yang diperoleh yaitu rata-rata akurasi sebesar 0.86 atau 86 persen, rata-rata presisi sebesar 0.85 atau 85 persen dan rata-rata recall diperoleh sebesar 0.79 atau 79 persen.

Tahap berikutnya dari pengujian model menggunakan Algoritma Artificial Neural Network adalah menganalisis atribut yang paling berpengaruh terhadap model. Proses ini dapat ditunjukkan sebagai berikut

Membuat DataFrame dari nama atribut dan bobot pada lapisan input

```
atribut_dan_bobot = pd.DataFrame({'Nama Atribut':
dataset.columns[:-1], 'Bobot':
model.layers[0].get_weights()[0][:, 0]})
```

Menampilkan DataFrame

```
print(atribut_dan_bobot)
    Nama Atribut     Bobot
0                       NIM  0.019846
1                    Status  0.113110
2          Perguruan Tinggi  0.179026
3             Program Studi -0.265025
4                   Jenjang  0.157990
```

```
(Lanjutan)
5          Alamat Kelurahan /Desa  0.028698
6                Alamat Kecamatan  0.112230
7                 Pendidikan Wali -0.046588
8                  Pekerjaan Wali -0.016708
9                Penghasilan Wali -0.236636
10                     Keterangan  0.114750
11  Jumlah Piutang UKT Mahasiswa -0.175040
12    Umur Piutang UKT Mahasiswa -0.055177
13            Jumlah Piutang BPP -0.175849
14              Umur Piutang BPP -0.023899
```

Hasil analisis atribut yang paling berpengaruh dalam algoritma Artificial Neural Network dengan menggunakan kode `get_weights` menunjukkan bobot masing-masing atribut pada lapisan input model neural network. Dalam konteks hasil bobot yang ditunjukkan, atribut yang paling berpengaruh terhadap model diidentifikasikan berdasarkan besarnya nilai bobot (positif atau negatif).

Jika sebuah atribut memiliki bobot positif, peningkatan nilai atribut tersebut akan memberikan kontribusi positif terhadap aktivasi neuron. Hal ini berarti bahwa neuron akan lebih cenderung diaktifkan oleh nilai yang lebih tinggi dari atribut tersebut. Sebaliknya, jika sebuah atribut memiliki bobot negatif, peningkatan nilai atribut akan memberikan kontribusi negatif pada aktivasi neuron. Ini berarti neuron akan lebih cenderung diinhibisi atau kurang aktif oleh nilai yang lebih tinggi dari atribut tersebut.

Berikut adalah interpretasi bobot yang memiliki nilai absolut yang paling besar:

a) Perguruan Tinggi (Bobot: 0,179026)

Bobot positif cukup besar menunjukkan peningkatan nilai atribut ini akan memberikan kontribusi positif terhadap aktivasi neuron. Hal ini berarti bahwa neuron akan lebih cenderung diaktifkan oleh nilai yang lebih tinggi dari atribut Perguruan Tinggi.

b) Program Studi (Bobot: -0,265025)

Bobot negatif yang cukup besar menunjukkan peningkatan nilai atribut ini akan memberikan kontribusi negatif pada aktivasi neuron. Ini berarti neuron akan lebih cenderung diinhibisi atau kurang aktif oleh nilai yang lebih tinggi dari atribut Program Studi.

c) Jumlah Piutang UKT Mahasiswa (Bobot: -0,175040)

Bobot negatif yang cukup besar menunjukkan peningkatan nilai atribut ini akan memberikan kontribusi negatif pada aktivasi neuron. Ini berarti neuron akan lebih cenderung diinhibisi atau kurang aktif oleh nilai yang lebih tinggi dari atribut Jumlah Piutang UKT Mahasiswa.

4.5. Hasil Percobaan dan Pengujian Model menggunakan Algoritma Naïve Bayes.

Percobaan 7 dilakukan dengan menggunakan Algoritma Naive Bayes. Pengujian dilakukan dengan cara membagi dataset menjadi dua bagian yaitu data testing dan data training dengan proporsi perbandingan pengujian 80:20.

Membagi dataset menjadi data latih dan data uji

```
from sklearn.model_selection import train_test_split
x_train,        x_test,        y_train,        y_test        =
train_test_split(x,y,test_size=0.20, random_state=0)
```

Tahapan ini menunjukkan proses pembagian dataset menggunakan train_test_split dengan proporsi 80% untuk data training dan 20% untuk data testing.

Setelah proses pembagian dataset maka dilakukan percobaan dan pengujian model dengan menggunakan google colaboratory dapat ditampilkan sebagai berikut:

Membangun model Naive Bayes dengan GaussianNB

```
from sklearn.naive_bayes import GaussianNB
classifier = GaussianNB()
classifier.fit(x_train,y_train)
```

GaussianNB

```
GaussianNB()
```

Melakukan prediksi pada data uji

```
y_pred = classifier.predict(x_test)
```

Pada tahapan ini merupakan langkah membuat objek klasifikasi Naive Bayes menggunakan kelas GaussianNB. Objek ini akan digunakan untuk melatih model Naive Bayes menggunakan data training. Pada langkah ini, model Naive Bayes dilatih dengan memanggil metode fit() pada objek classifier yang telah dibuat sebelumnya. Metode fit() menerima dua parameter, yaitu x_train (data training) dan y_train (label atau target untuk data training). Dalam proses ini, model akan belajar untuk melakukan prediksi berdasarkan pola-pola yang ada dalam data training. Kemudian model Naive Bayes yang telah dilatih untuk melakukan prediksi pada data uji. Metode predict() digunakan untuk menghasilkan prediksi berdasarkan data uji yang diberikan.

Hasil prediksi akan disimpan dalam variabel `y_pred` yang dapat digunakan untuk evaluasi performa model. Evaluasi model menggunakan confusion matrix seperti tampilan gambar berikut:

Evaluasi model menggunakan confusion matrix

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
[[17  1]
 [ 1 33]]
```

Proses evaluasi model pada tahapan di atas merupakan evaluasi model menggunakan Confusion Matrix. Hasil Confusion Matrix di atas jika dituliskan dalam bentuk tabel maka dapat ditampilkan seperti Tabel 4.9 di bawah ini:

Tabel 4.9 Tabel Confusion Matrix:

	Prediksi Negatif (0)	Prediksi Positif (1)
Aktual Negatif	17	1
Aktual Positif	1	33

Dari confusion matrix di atas, kita dapat menghitung beberapa metrik evaluasi sebagai berikut:

a. Akurasi (Accuracy): Akurasi menggambarkan seberapa baik model dapat mengklasifikasikan secara benar keseluruhan data. Rumus akurasi adalah:

```
scss
Akurasi = (TP + TN) / (TP + TN + FP + FN)
```

Dalam kasus ini, akurasi = (17 + 33) / (17 + 1 + 1 + 33) = 0.95 atau 95%.

b. Presisi (Precision): Presisi mengukur seberapa baik model dapat mengidentifikasi dengan benar positif dari semua prediksi positif. Rumus presisi adalah:

```
makefile
Presisi = TP / (TP + FP)
```

Dalam kasus ini, presisi = 17 / (17 + 1) = 0.94 atau 94%.

c. Recall (Sensitivitas atau True Positive Rate): Recall mengukur seberapa baik model dapat mendeteksi dengan benar positif dari semua data sebenarnya yang positif. Rumus recall adalah:

```
makefile
Recall = TP / (TP + FN)
```

Dalam kasus ini, recall = 17 / (17 + 1) = 0.94 atau 94%.

Percobaan 8 dilakukan percobaan dengan menggunakan Algoritma Naïve Bayes. Pengujian dilakukan dengan cara membagi dataset menjadi dua bagian yaitu data testing dan data training dengan proporsi perbandingan pengujian 90:10.

Membagi dataset menjadi data latih dan data uji

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.10, random_state=0)
```

Proses di atas menunjukkan proses pembagian dataset menggunakan train_test_split dengan proporsi 90% untuk data training dan 10% untuk data testing.

Setelah proses pembagian dataset maka dilakukan percobaan dan pengujian model dengan menggunakan google colaboratory dapat ditampilkan sebagai berikut:

Membangun model Naive Bayes dengan GaussianNB

```
from sklearn.naive_bayes import GaussianNB
classifier = GaussianNB()
classifier.fit(x_train,y_train)
```

GaussianNB

```
GaussianNB()
```

(Lanjutan)

```
Melakukan prediksi pada data uji
y_pred = classifier.predict(x_test)
```

Pada proses ini merupakan langkah membuat objek klasifikasi Naive Bayes menggunakan kelas GaussianNB. Objek ini akan digunakan untuk melatih model Naive Bayes menggunakan data training. Pada langkah ini, model Naive Bayes dilatih dengan memanggil metode fit() pada objek classifier yang telah dibuat sebelumnya. Metode fit() menerima dua parameter, yaitu x_train (data training) dan y_train (label atau target untuk data training). Dalam proses ini, model akan belajar untuk melakukan prediksi berdasarkan pola-pola yang ada dalam data training. Kemudian model Naive Bayes yang telah dilatih untuk melakukan prediksi pada data uji. Metode predict() digunakan untuk menghasilkan prediksi berdasarkan data uji yang diberikan.

Hasil prediksi akan disimpan dalam variabel `y_pred` yang dapat digunakan untuk evaluasi performa model. Evaluasi model menggunakan confusion matrix seperti tampilan gambar berikut:

Evaluasi model menggunakan confusion matrix

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
[[ 7  0]
 [ 2 17]]
```

Proses evaluasi model pada tahapan ini merupakan evaluasi model menggunakan Confusion Matrix. Hasil Confusion Matrix di atas jika dituliskan dalam bentuk tabel maka dapat ditampilkan seperti Tabel 4.10 di bawah ini:

Tabel 4.10 Tabel Confusion Matrix:

	Prediksi Negatif (0)	Prediksi Positif (1)
Aktual Negatif	7	0
Aktual Positif	2	17

Dari confusion matrix di atas, kita dapat menghitung beberapa metrik evaluasi sebagai berikut:

a. Akurasi (Accuracy): Akurasi menggambarkan seberapa baik model dapat mengklasifikasikan secara benar keseluruhan data. Rumus akurasi adalah:

```
scss
Akurasi = (TP + TN) / (TP + TN + FP + FN)
```

Dalam kasus ini, akurasi = (7 + 17) / (7 + 0 + 2 + 17) = 0.92 atau 92%.

b. Presisi (Precision): Presisi mengukur seberapa baik model dapat mengidentifikasi dengan benar positif dari semua prediksi positif. Rumus presisi adalah:

```
makefile
Presisi = TP / (TP + FP)
```

Dalam kasus ini, presisi = 7 / (7 + 2) = 0.78 atau 78%.

c. Recall (Sensitivitas atau True Positive Rate): Recall mengukur seberapa baik model dapat mendeteksi dengan benar positif dari semua data sebenarnya yang positif. Rumus recall adalah:

```
makefile
Recall = TP / (TP + FN)
```

Dalam kasus ini, recall = 7 / (7 + 0) = 1 atau 100%.

Percobaan 9 dilakukan dengan menggunakan Algoritma Naïve Bayes. Pengujian dilakukan dengan menggunakan validasi silang (cross-validation) yaitu k-fold cross-validation untuk membagi dataset menjadi k subset.

Inisialisasi model Naive Bayes dan jumlah subset (fold) untuk k-fold cross-validation

```
classifier = GaussianNB()
k = 5  # Jumlah subset (fold)
kf = KFold(n_splits=k, shuffle=True, random_state=0)
```

Tahapan ini menunjukkan proses k-fold validation dengan k subset yaitu 5. Dataset dibagi menjadi 5 subset yang sama besar selanjutnya model dilatih dan diuji 5 kali, setiap kali menggunakan salah satu subset sebagai data testing dan k-1 subset lainnya sebagai data training.

Setelah proses pembagian dataset, dilakukan proses k-fold cross-validation yang dapat ditampilkan sebagai berikut:

Melakukan k-fold cross-validation

```
for train_index, test_index in kf.split(x):
    x_train, x_test = x[train_index], x[test_index]
    y_train, y_test = y[train_index], y[test_index]

    # Melatih model pada subset train
    classifier.fit(x_train, y_train)

    # Prediksi menggunakan subset test
    y_pred = classifier.predict(x_test)

    # Menghitung confusion matrix pada setiap iterasi
    cm = confusion_matrix(y_test, y_pred)

    # Menghitung dan menyimpan akurasi, presisi, dan recall
pada setiap iterasi
    accuracy = np.mean(y_pred == y_test)
    precision = precision_score(y_test, y_pred)
    recall = recall_score(y_test, y_pred)

    accuracies.append(accuracy)
    precisions.append(precision)
    recalls.append(recall)
```

Tahapan ini merupakan proses menguji dan mengevaluasi kinerja model algoritma Naïve Bayes, pertama-tama menggunakan cross-validation dari subset k atau fold. Untuk melakukan ini, KFold digunakan untuk membagi data menjadi subset k, di mana masing-masing subset digunakan sebagai data testing dan data training. Selanjutnya, melatih model Naïve Bayes (fit) dengan menggunakan data training (x_train, y_train) dari setiap subset, dan kemudian dilakukan prediksi (predict) pada data testing (x_test) dan menyimpan hasil prediksi dalam y_pred. Kemudian, dengan menggunakan fungsi confusion_matrix dari scikit-learn, dapat dihitung confusion matrix untuk setiap iterasi dari hasil prediksi dan nilai sebenarnya (y_test).

Tahapan selanjutnya adalah menghitung rata-rata akurasi, presisi dan recall yang dihasilkan dari seluruh akurasi masing-masing subset. Proses tersebut dapat ditunjukkan sebagai berikut:

Menghitung rata-rata akurasi, presisi, dan recall dari semua iterasi

```
mean_accuracy = np.mean(accuracies)
mean_precision = np.mean(precisions)
mean_recall = np.mean(recalls)

print(f"Mean Accuracy: {mean_accuracy}")
print(f"Mean Precision: {mean_precision}")
print(f"Mean Recall: {mean_recall}")
Mean Accuracy: 0.8653846153846153
Mean Precision: 0.8443840579710145
Mean Recall: 0.8355909536687569
```

Tahapan ini menggambarkan proses untuk menghitung dan menampilkan rata-rata akurasi, presisi dan recall dari semua iterasi. Hasil yang diperoleh yaitu

rata-rata akurasi sebesar 0.86 atau 86 persen, rata-rata presisi sebesar 0.84 atau 84 persen dan rata-rata recall diperoleh sebesar 0.83 atau 83 persen.

Tahap berikutnya dari pengujian model menggunakan Algoritma Naïve Bayes adalah menganalisis atribut yang paling berpengaruh terhadap model. Proses ini dapat ditunjukkan sebagai berikut

Probabilitas kondisional menggambarkan sejauh mana setiap atribut mempengaruhi keputusan klasifikasi. Hasil probabilitas kondisional yang dihasilkan dari model Naive Bayes Gaussian untuk dua kelas (Kelas 1 dan Kelas 2) dan 15 atribut (Atribut 0 hingga Atribut 14).

Mendapatkan probabilitas atribut dalam kelas (P(atribut|class))

```
atribut_prob_kondisional = classifier.theta_
```

Mencetak probabilitas atribut kondisional untuk setiap atribut dalam setiap kelas

```
for kelas in range(len(classifier.classes_)):
    print(f'Probabilitas kondisional untuk Kelas 
{classifier.classes_[kelas]}:')
    for atribut in 
range(len(atribut_prob_kondisional[kelas])):
        prob_atribut = 
atribut_prob_kondisional[kelas][atribut]
        print(f'Atribut {atribut}:')
        print(f'Probabilitas Kondisional: {prob_atribut}')
        print()
```

```
Probabilitas kondisional untuk Kelas 1:
Atribut 0:
Probabilitas Kondisional: -0.12317651850308532
Atribut 1:
Probabilitas Kondisional: 0.0
Atribut 2:
Probabilitas Kondisional: 0.0
Atribut 3:
Probabilitas Kondisional: 0.03541602246783174
Atribut 4:
Probabilitas Kondisional: 0.0
Atribut 5:
Probabilitas Kondisional: -0.1730774152212989
Atribut 6:
Probabilitas Kondisional: 0.10932596436181703
```

```
(Lanjutan)
Atribut 7:
Probabilitas Kondisional: 0.13536822121918937
Atribut 8:
Probabilitas Kondisional: 0.0034924649181067788
Atribut 9:
Probabilitas Kondisional: -0.22242724237036393
Atribut 10:
Probabilitas Kondisional: 0.0
Atribut 11:
Probabilitas Kondisional: 0.9192599154627339
Atribut 12:
Probabilitas Kondisional: 0.8801656990923608
Atribut 13:
Probabilitas Kondisional: 0.5987308783768687
Atribut 14:
Probabilitas Kondisional: 0.47424467047004654
```

Hasil tersebut dapat dijelaskan sebagai berikut

a) Kelas 1:

1) Atribut 0 memiliki probabilitas kondisional negatif (-0.123). Ini menunjukkan bahwa nilai atribut 0 cenderung lebih rendah untuk Kelas 1.
2) Atribut 1 dan Atribut 2 memiliki probabilitas kondisional nol (0.0). Ini menunjukkan bahwa atribut 1 dan atribut 2 mungkin tidak berpengaruh dalam membedakan Kelas 1 dari kelas lainnya.
3) Atribut 3 memiliki probabilitas kondisional positif (0.035). Ini menunjukkan bahwa nilai atribut 3 cenderung lebih tinggi untuk Kelas 1.
4) Atribut 5 memiliki probabilitas kondisional negatif (-0.173). Ini menunjukkan bahwa nilai atribut 5 cenderung lebih rendah untuk Kelas 1.

5) Atribut 11 memiliki probabilitas kondisional yang sangat tinggi (0.919). Ini menunjukkan bahwa atribut 11 adalah salah satu atribut paling berpengaruh dalam membedakan Kelas 1.

6) Atribut 13 dan Atribut 14 juga memiliki probabilitas kondisional positif yang cukup tinggi (0.599 dan 0.474), yang menunjukkan bahwa atribut 13 dan atribut 14 juga berkontribusi secara signifikan dalam membedakan Kelas 1.

b) Kelas 2:

1) Atribut 0 memiliki probabilitas kondisional positif (0.086). Ini menunjukkan bahwa nilai atribut 0 cenderung lebih tinggi untuk Kelas 2.

2) Atribut 3 memiliki probabilitas kondisional negatif (-0.025). Ini menunjukkan bahwa nilai atribut 3 cenderung lebih rendah untuk Kelas 2.

3) Atribut 5 memiliki probabilitas kondisional positif (0.120). Ini menunjukkan bahwa nilai atribut 5 cenderung lebih tinggi untuk Kelas 2.

4) Atribut 6, Atribut 7, Atribut 8, dan Atribut 9 memiliki probabilitas kondisional positif, menunjukkan bahwa nilai-nilai atribut ini cenderung lebih tinggi untuk Kelas 2.

5) Atribut 11, Atribut 12, Atribut 13, dan Atribut 14 memiliki probabilitas kondisional negatif yang cukup tinggi. Ini menunjukkan bahwa atribut 11, 12, 13, dan 14 adalah atribut paling berpengaruh dalam membedakan Kelas 2.

4.6. Pembahasan

Pembahasan hasil percobaan dan pengujian model menggunakan Algoritma C4.5 melalui percobaan 1, percobaan 2 dan percobaan 3 yaitu dengan proporsi data training dan data testing 80:20 dan 90:10 serta menggunakan k-fold cross-validation. Pengujian tersebut memperoleh hasil sebagai berikut:

Tabel 4.11 Tabel Hasil Percobaan dan Pengujian Model Algoritma C4.5

Algoritma C.45	Pengujian 80:20	Pengujian 90:10	Pengujian k-fold cross-validation
Akurasi	85%	92%	83%
Presisi	76%	78%	78%
Recall	89%	100%	82%

Berdasarkan Tabel 4.11, Hal ini berarti pada pengujian dengan menggunakan proporsi data training sebesar 80% dan data testing sebesar 20%, algoritma C4.5 mampu mengklasifikasikan data dengan akurasi sebesar 85%. Akurasi mengukur sejauh mana model dapat melakukan klasifikasi yang benar dari total data yang diuji.

Selain itu, presisi sebesar 76% menunjukkan seberapa baik algoritma C4.5 dalam mengidentifikasi instance yang relevan sebagai positif (klasifikasi benar) dari total instance yang diklasifikasikan sebagai positif. Recall sebesar 89% menunjukkan seberapa baik algoritma C4.5 dalam menemukan dan mengklasifikasikan instance positif secara keseluruhan dari total instance positif yang ada.

Dalam pengujian dengan menggunakan proporsi data training sebesar 90% dan data testing sebesar 10%, algoritma C4.5 mencapai akurasi sebesar 92%.

Akurasi yang tinggi menunjukkan kemampuan algoritma untuk melakukan klasifikasi yang benar.

Presisi sebesar 78% menunjukkan bahwa algoritma C4.5 dapat mengidentifikasi sebagian besar instance yang relevan dengan benar sebagai positif, namun ada ruang untuk perbaikan dalam mengurangi kesalahan klasifikasi false positive.

Recall sebesar 100% menunjukkan bahwa algoritma C4.5 mampu menemukan dan mengklasifikasikan semua instance positif dengan benar dari total instance positif yang ada. Hal ini menunjukkan kemampuan algoritma untuk tidak mengabaikan instance positif.

Dalam pengujian menggunakan k-fold cross-validation algoritma C4.5 mampu mengklasifikasikan data dengan akurasi sebesar 85%. Akurasi mengukur sejauh mana model dapat melakukan klasifikasi yang benar dari total data yang diuji. Presisi sebesar 78% menunjukkan bahwa algoritma C4.5 dapat mengidentifikasi sebagian besar instance yang relevan dengan benar sebagai positif, namun ada ruang untuk perbaikan dalam mengurangi kesalahan klasifikasi false positive. Recall sebesar 82% menunjukkan seberapa baik algoritma C4.5 dalam menemukan dan mengklasifikasikan instance positif secara keseluruhan dari total instance positif yang ada.

Dari percobaan 1, percobaan 2 dan percobaan 3 ini, akurasi yang tinggi menunjukkan bahwa algoritma C4.5 berhasil dalam melakukan klasifikasi pada dataset yang diberikan. Dengan hasil terbaik pada proporsi data training dan data testing 90:10.

Dari analisis atribut yang paling berpengaruh, dapat dilihat bahwa beberapa atribut memiliki kontribusi yang signifikan terhadap pembuatan keputusan dalam model C4.5, sementara yang lain memiliki kontribusi yang rendah atau bahkan tidak berpengaruh sama sekali. Atribut "Jumlah Piutang UKT mahasiswa" adalah atribut yang paling berpengaruh dalam model ini, diikuti oleh "Pendidikan Wali" dan "Pekerjaan Wali". Dengan hasil ini, dapat memprioritaskan atribut-atribut yang paling penting dalam analisis dan pengambilan keputusan.

Atribut yang paling berpengaruh sangat mudah dikenali karena muncul pada bagian atas pohon keputusan. Hasil pohon keputusan pada Gambar 4.14 menunjukkan hasil yang sejalan dengan hasil *feature importance* yaitu Atribut "Jumlah Piutang UKT mahasiswa", "Pendidikan Wali" dan "Pekerjaan Wali".

Pembahasan hasil percobaan dan pengujian model menggunakan Algoritma Artificial Neural Network melalui percobaan 3, percobaan 4 dan percobaan 6 yaitu dengan proporsi data training dan data testing 80:20 dan 90:10 serta menggunakan k-fold cross-validation. Pengujian tersebut memperoleh hasil sebagai berikut:

Tabel 4.12 Tabel Hasil Percobaan dan Pengujian Model Algoritma Artificial Neural Network

Algoritma Neural Network	Pengujian 80:20	Pengujian 90:10	Pengujian k-fold cross-validation
Akurasi	54%	56%	86%
Presisi	54%	56%	85%
Recall	100%	100%	79%

Tabel 4.12 menunjukkan bahwa pada pengujian dengan menggunakan proporsi data training sebesar 80% dan data testing sebesar 20%, Algoritma Artificial Neural Network menghasilkan akurasi sebesar 54%. Akurasi ini menunjukkan sejauh mana model dapat melakukan klasifikasi yang benar dari total

data yang diuji. Dalam kasus ini, akurasi yang rendah menunjukkan bahwa model tidak mampu melakukan klasifikasi dengan tingkat keakuratan yang tinggi.

Presisi sebesar 54% menunjukkan seberapa baik model Artificial Neural Network dalam mengidentifikasi instance yang relevan sebagai positif (klasifikasi benar) dari total instance yang diklasifikasikan sebagai positif. Presisi yang rendah menunjukkan adanya banyak false positive, yaitu instance yang seharusnya diklasifikasikan sebagai negatif tetapi terklasifikasikan sebagai positif.

Recall sebesar 100% menunjukkan seberapa baik model Artificial Neural Network dalam menemukan dan mengklasifikasikan instance positif secara keseluruhan dari total instance positif yang ada. Recall yang tinggi menunjukkan bahwa model tidak mengabaikan instance positif, tetapi dalam hal ini juga menghasilkan banyak false positive.

Selanjutnya, hasil pengujian Algoritma Artificial Neural Network dengan proporsi data training dan data testing 90:10 sesuai Tabel 4.12 Pengujian ini, dengan menggunakan proporsi data training sebesar 90% dan data testing sebesar 10%, Algoritma Artificial Neural Network mencapai akurasi sebesar 56%. Akurasi yang rendah menunjukkan bahwa model tidak mampu melakukan klasifikasi dengan tingkat keakuratan yang tinggi pada dataset yang diberikan.

Presisi sebesar 56% menunjukkan seberapa baik model Artificial Neural Network dalam mengidentifikasi sebagian instance yang relevan dengan benar sebagai positif dari total instance yang diklasifikasikan sebagai positif. Presisi yang rendah menunjukkan adanya banyak false positive.

Recall sebesar 100% menunjukkan seberapa baik model Artificial Neural Network dalam menemukan dan mengklasifikasikan semua instance positif dengan benar dari total instance positif yang ada. Recall yang tinggi menunjukkan bahwa model tidak mengabaikan instance positif, tetapi dalam hal ini juga menghasilkan banyak false positive.

Dalam pengujian menggunakan k-fold cross-validation algoritma Artificial Neural Network mampu mengklasifikasikan data dengan akurasi sebesar 86%. Akurasi mengukur sejauh mana model dapat melakukan klasifikasi yang benar dari total data yang diuji. Presisi sebesar 85% menunjukkan bahwa algoritma C4.5 dapat mengidentifikasi sebagian besar instance yang relevan dengan benar sebagai positif, namun ada ruang untuk perbaikan dalam mengurangi kesalahan klasifikasi false positive. Recall sebesar 79% menunjukkan seberapa baik algoritma C4.5 dalam menemukan dan mengklasifikasikan instance positif secara keseluruhan dari total instance positif yang ada.

Dalam percobaan 4 dan percobaan 5, hasilnya menunjukkan bahwa Algoritma Artificial Neural Network yang digunakan tidak mampu memberikan kinerja yang baik dalam klasifikasi pada dataset yang diberikan. Akurasi, presisi, dan recall yang rendah menunjukkan bahwa model tersebut tidak cocok atau perlu disesuaikan lebih lanjut dengan dataset tersebut atau mungkin memerlukan modifikasi pada arsitektur atau parameter Neural Network yang digunakan.

Namun, pengujian 6 dengan metode k-fold cross-validation menunjukkan hasil yang lebih baik dengan akurasi sebesar 86%, presisi sebesar 85%, dan recall sebesar 79%. Metode k-fold cross-validation menghasilkan evaluasi yang lebih

stabil dan obyektif karena model dievaluasi pada subset data yang berbeda-beda, dan kemudian hasil rata-rata dari evaluasi tersebut digunakan untuk mengukur performa model secara keseluruhan. Hasil ini menunjukkan bahwa model memiliki tingkat keakuratan dan performa yang lebih baik ketika diuji dengan metode k-fold cross-validation dibandingkan dengan pembagian data 80:20 atau 90:10.

Dari analisis atribut yang paling berpengaruh, dapat ditunjukkan pengaruh masing-masing atribut terhadap prediksi model Artificial Neural Network. Pengaruh menunjukkan neoron akan lebih cenderung kurang aktif oleh nilai yang lebih tinggi dari beberapa atribut, sedangkan neorun akan lebih cenderung diaktifkan oleh nilai yang lebih tinggi dari atribut "Perguruan Tinggi". Atribut "Program Studi" dan "Jumlah Piutang UKT Mahasiswa" menunjukkan peningkatan nilai pada kedua atribut ini mengakibatkan neuron akan lebih cenderung kurang aktif dalam membentuk jaringan model Arificial Neural Network.

Pembahasan hasil percobaan dan pengujian model menggunakan Algoritma Naïve Bayes melalui percobaan 7, percobaan 8 dan percobaan 9 yaitu dengan proporsi data training dan data testing 80:20 dan 90:10 serta menggunakan k-fold cross-validation. Pengujian tersebut memperoleh hasil sebagai berikut:

Tabel 4.13 Tabel Hasil Percobaan dan Pengujian Model Algoritma Naïve Bayes

Algoritma Naïve Bayes	Pengujian 80:20	Pengujian 90:10	Pengujian k-fold cross-validation
Akurasi	95%	92%	86%
Presisi	94%	78%	84%
Recall	94%	100%	83%

Pada pengujian dengan menggunakan proporsi data training sebesar 80% dan data testing sebesar 20% yang ditampilkan pada Tabel 4.13, Algoritma Naive Bayes menghasilkan akurasi sebesar 95%. Akurasi ini menunjukkan sejauh mana model dapat melakukan klasifikasi yang benar dari total data yang diuji. Dalam kasus ini, akurasi yang tinggi menunjukkan bahwa model Naive Bayes mampu melakukan klasifikasi dengan tingkat keakuratan yang tinggi pada dataset yang diberikan. Presisi sebesar 94% menunjukkan seberapa baik model Naive Bayes dalam mengidentifikasi instance yang relevan sebagai positif (klasifikasi benar) dari total instance yang diklasifikasikan sebagai positif. Presisi yang tinggi menunjukkan sedikitnya false positive, yaitu instance yang seharusnya diklasifikasikan sebagai negatif tetapi terklasifikasikan sebagai positif. Recall sebesar 94% menunjukkan seberapa baik model Naive Bayes dalam menemukan dan mengklasifikasikan instance positif secara keseluruhan dari total instance positif yang ada. Recall yang tinggi menunjukkan bahwa model mampu mendeteksi sebagian besar instance positif dengan benar dan tidak mengabaikan instance positif.

Selanjutnya, hasil pengujian Algoritma Naive Bayes dengan proporsi data training dan data testing 90:10 adalah pengujian ini dengan menggunakan proporsi data training sebesar 90% dan data testing sebesar 10%, Algoritma Naive Bayes mencapai akurasi sebesar 92%. Akurasi yang tinggi menunjukkan bahwa model Naive Bayes mampu melakukan klasifikasi dengan tingkat keakuratan yang cukup tinggi pada dataset yang diberikan. Presisi sebesar 78% menunjukkan seberapa baik model Naive Bayes dalam mengidentifikasi sebagian instance yang relevan dengan

benar sebagai positif dari total instance yang diklasifikasikan sebagai positif. Presisi yang cukup tinggi menunjukkan sedikitnya false positive, tetapi masih terdapat ruang untuk perbaikan dalam mengurangi kesalahan klasifikasi false positive. Recall sebesar 100% menunjukkan seberapa baik model Naive Bayes dalam menemukan dan mengklasifikasikan semua instance positif dengan benar dari total instance positif yang ada. Recall yang tinggi menunjukkan bahwa model Naive Bayes tidak mengabaikan instance positif dan mampu mendeteksi semua instance positif.

Dalam pengujian menggunakan k-fold cross-validation algoritma Naïve Bayes mampu mengklasifikasikan data dengan akurasi sebesar 86%. Akurasi mengukur sejauh mana model dapat melakukan klasifikasi yang benar dari total data yang diuji. Presisi sebesar 84% menunjukkan bahwa algoritma C4.5 dapat mengidentifikasi sebagian besar instance yang relevan dengan benar sebagai positif, namun ada ruang untuk perbaikan dalam mengurangi kesalahan klasifikasi false positive. Recall sebesar 83% menunjukkan seberapa baik algoritma C4.5 dalam menemukan dan mengklasifikasikan instance positif secara keseluruhan dari total instance positif yang ada.

Dalam percobaan 7, percobaan 8 dan percobaan 9 Algoritma Naive Bayes menghasilkan kinerja yang baik dengan akurasi yang tinggi dan nilai presisi serta recall yang memadai.

Dengan hasil analisis atribut yang paling berpengaruh terhadap model, dapat disimpulkan bahwa atribut Jumlah Piutang UKT Mahasiswa, Jumlah Piutang BPP, dan Umur Piutang BPP adalah atribut paling berpengaruh dalam membedakan

kelas 1, sementara atribut Jumlah Piutang UKT Mahasiswa, Umur Piutang UKT Mahasiswa, Jumlah Piutang BPP, dan Umur Piutang BPP adalah atribut paling berpengaruh dalam membedakan kelas 2. Probabilitas kondisional membantu dalam memahami kontribusi relatif atribut pada klasifikasi yang dilakukan oleh model Naive Bayes.

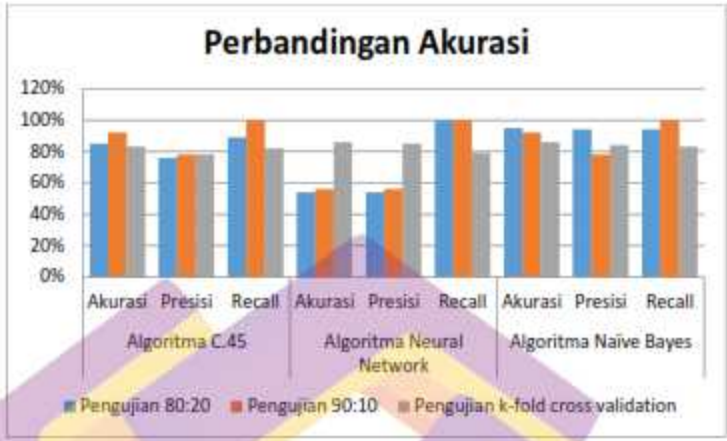
Hasil perbandingan akurasi Algoritma C4.5, Artificial Neural Network dan Naïve Bayes untuk memprediksi piutang biaya pendidikan mahasiswa tak tertagih dapat ditampilkan sesuai tabel berikut:

Tabel 4.14 Tabel Hasil Perbandingan Akurasi Algoritma C4.5, Artificial Neural Network dan Naïve Bayes

Pengujian	Algoritma C4.5			Artificial Neural Network			Naive Bayes		
	Akurasi	Presisi	Recall	Akurasi	Presisi	Recall	Akurasi	Presisi	Recall
Pengujian 80:20	85%	76%	89%	54%	54%	100%	95%	94%	94%
Pengujian 90:10	92%	78%	100%	56%	56%	100%	92%	78%	100%
Pengujian k-fold cross-validation	83%	78%	82%	86%	85%	79%	86%	84%	83%

Pada Tabel 4.14 menggambarkan secara umum bahwa Algoritma Naive Bayes dan C4.5 menunjukkan hasil yang lebih baik dalam hal akurasi dan presisi dibandingkan dengan Neural Network dalam memprediksi piutang biaya pendidikan mahasiswa tak tertagih.

Selain dalam bentuk Tabel untuk perbandingan akurasi dari Algoritma C4.5, Artificial Neural Network dan Naïve Bayes dibuat juga dalam diagram batang seperti pada Gambar berikut ini:



Gambar 4.2 Diagram Batang Hasil Perbandingan Akurasi Algoritma C4.5, Artificial Neural Network dan Naïve Bayes

Dari hasil evaluasi pada Gambar 4.32 dapat diamati beberapa hal:

- Algoritma C4.5 juga menghasilkan akurasi yang baik, tetapi memiliki variasi dalam presisi dan recall antara percobaan 90:10, 80:20 dan k-fold cross-validation.
- Algoritma Neural Network menghasilkan akurasi yang rendah dan presisi yang sama rendahnya dalam percobaan 80:20 dan 90:10, namun recall tetap tinggi. Tetapi jika menggunakan k-fold cross-validation menghasilkan akurasi yang lebih stabil dan obyektif karena model dievaluasi pada subset data yang berbeda-beda.
- Algoritma Naive Bayes memiliki performa yang baik dalam ketiga percobaan dengan akurasi yang tinggi dan presisi yang relatif stabil.

Dalam pengujian Artificial Neural Network dengan proporsi data training dan data testing sebesar 80:20 dan 90:10 yang menghasilkan nilai akurasi, presisi

yang sangat rendah dapat dijelaskan bahwa dalam kasus ini, nilai presisi dan akurasi sangat rendah, sedangkan recall memiliki nilai yang tinggi. Hal ini menunjukkan bahwa model memiliki kecenderungan untuk memprediksi kelas negatif dengan benar, tetapi tidak dapat mengidentifikasi dengan baik kelas positif. Perlu diperhatikan bahwa hasil confusion matrix ini dapat menunjukkan adanya ketidakseimbangan kelas dalam data, di mana kelas positif memiliki jumlah yang jauh lebih sedikit daripada kelas negatif.

BAB V

PENUTUP

5.1. Kesimpulan

Dalam penelitian ini, dilakukan prediksi piutang biaya pendidikan mahasiswa tak tertagih menggunakan Algoritma C4.5, Artificial Neural Network, dan Naïve Bayes dengan tiga percobaan berbeda, yaitu dengan proporsi data training dan data testing 80:20 dan 90:10 serta menggunakan k-fold cross-validation dan membandingkan ketiga algoritma tersebut serta menganalisis atribut-atribut yang paling berpengaruh dari masing-masing model. Hasil penelitian menunjukkan:

1. Percobaan menggunakan Algoritma C4.5 dengan proporsi data training dan data testing 90:10 memberikan hasil yang terbaik dan dapat menjadi alat yang sangat berguna untuk membantu institusi pendidikan dalam mengidentifikasi mahasiswa yang berisiko tinggi untuk menunggak biaya pendidikan. Hal ini menunjukkan bahwa Algoritma C4.5 berhasil dalam melakukan klasifikasi pada dataset yang diberikan dengan hasil analisis atribut yang paling penting dalam pengambilan keputusan adalah atribut "Jumlah Piutang UKT Mahasiswa" sebagai atribut yang paling berpengaruh, diikuti oleh "Pendidikan Wali" dan "Pekerjaan Wali"
2. Hasil percobaan menggunakan Artificial Neural Network menunjukkan bahwa Algoritma Artificial Neural Network yang digunakan tidak memberikan kinerja yang baik dalam klasifikasi pada dataset yang diberikan untuk

memprediksi piutang biaya pendidikan mahasiswa tak tertagih. Akurasi, presisi, dan recall yang rendah menunjukkan bahwa model memiliki kecenderungan untuk memprediksi kelas negatif dengan benar, tetapi tidak dapat mengidentifikasi dengan baik kelas positif. Perlu diperhatikan bahwa hasil confusion matrix ini dapat menunjukkan adanya ketidakseimbangan kelas dalam data, di mana kelas positif memiliki jumlah yang jauh lebih sedikit daripada kelas negatif. Tetapi jika menggunakan k-fold cross-validation menghasilkan akurasi yang tinggi, hal ini menunjukkan dengan menggunakan k-fold cross-validation berhasil memprediksi piutang biaya pendidikan mahasiswa tak tertagih dengan baik. Analisis atribut yang paling berpengaruh dari model Algoritma Neural Network menunjukkan neoron akan lebih cenderung kurang aktif oleh nilai yang lebih tinggi dari beberapa atribut yang bernilai negatif, sedangkan neorun akan lebih cenderung diaktifkan oleh nilai yang lebih tinggi dari atribut yang bernilai positif seperti atribut "Perguruan Tinggi" pada pengujian model ini. Atribut "Program Studi" dan "Jumlah Piutang UKT Mahasiswa" menunjukkan peningkatan nilai pada kedua atribut ini mengakibatkan neuron akan lebih cenderung kurang aktif dalam membentuk jaringan model Arificial Neural Network.

3. Percobaan menggunakan Algoritma Naive Bayes menunjukkan bahwa Algoritma Naïve Bayes merupakan metode yang cocok untuk masalah klasifikasi piutang biaya pendidikan mahasiswa tak tertagih. Algoritma ini mampu memberikan tingkat akurasi dan presisi yang baik yang sangat penting dalam mengidentifikasi piutang yang perlu ditagih. Selain itu algoritma ini

konsisten dalam berbagai kondisi pengujian. Analisis atribut yang paling berpengaruh terhadap algoritma Naïve bayes untuk pemisahan kelas 1 yaitu " Jumlah Piutang UKT Mahasiswa", "Jumlah Piutang BPP" dan "Umur Piutang BPP" sedangkan atribut yang paling berpengaruh untuk pemisahan kelas 2 yaitu "Jumlah Piutang UKT Mahasiswa", "Umur Piutang UKT Mahasiswa", "Jumlah Piutang BPP" dan "Umur Piutang BPP".

4. Studi komparasi terhadap ketiga Algoritma C4.5, Artificial Neural Network dan Algoritma Naïve Bayes dapat dijelaskan bahwa, Algoritma Naive Bayes dan C4.5 menunjukkan hasil yang lebih baik dalam hal akurasi dan presisi dibandingkan dengan Artificial Neural Network dalam memprediksi piutang biaya pendidikan mahasiswa tak tertagih. Hal ini disebabkan
   a. Algoritma Naïve Bayes dan C4.5 lebih baik menangani masalah ketidakseimbangan antara kelas positif dan kelas negatif karena penggunaan probabilitas dan teknik pemisahan berbasis aturan keputusan.
   b. Algoritma C4.5 untuk atribut yang paling berpengaruh sangat mudah dikenali karena muncul pada bagian atas pohon keputusan. Algoritma Naïve Bayes dapat diidentifikasi dengan mudah melalui perhitungan probabilitas kondisional. Tetapi pada Algoritma Neural Network memiliki banyak parameter yang harus disesuaikan dengan baik.

5.2. Saran

Berdasarkan kesimpulan tersebut, berikut adalah beberapa saran yang dapat diberikan untuk penelitian selanjutnya.

1. Dalam penelitian ini, Algoritma Neural Network tidak memberikan kinerja yang baik dalam memprediksi piutang biaya pendidikan mahasiswa tak tertagih. Oleh karena itu, penelitian selanjutnya dapat mempertimbangkan eksplorasi metode atau modifikasi algoritma Neural Network untuk meningkatkan kinerja seperti menggunakan arsitektur yang lebih kompleks atau mengubah parameter yang digunakan.
2. Hasil confusion matrix menunjukkan adanya ketidakseimbangan kelas dalam data, di mana kelas positif memiliki jumlah yang jauh lebih sedikit daripada kelas negatif. Penelitian selanjutnya dapat mengambil langkah-langkah untuk mengatasi masalah ini, seperti oversampling atau undersampling pada kelas minoritas, atau menggunakan teknik ensemble untuk mengatasi ketidakseimbangan kelas.
3. Selain akurasi, presisi, dan recall, penelitian selanjutnya dapat mempertimbangkan evaluasi menggunakan metrik lainnya seperti F1-score, area under the curve (AUC), atau metrik khusus untuk penanganan ketidakseimbangan kelas seperti precision-recall curve.
4. Penelitian selanjutnya dapat menggabungkan hasil dari beberapa algoritma yang berbeda, seperti C4.5 dan Naive Bayes, untuk membuat model ensemble yang lebih kuat dalam memprediksi piutang biaya pendidikan mahasiswa tak tertagih. Menggunakan teknik ensemble learning dapat menjadi pilihan untuk meningkatkan kinerja prediksi.
5. Penelitian selanjutnya dapat melibatkan penggunaan dataset yang lebih besar dan beragam.

6. Melakukan penelitian komparatif yang lebih luas dengan melibatkan algoritma-algoritma lain dalam data mining juga dapat menjadi langkah yang menarik.

DAFTAR PUSTAKA

PUSTAKA BUKU

Muhammad Arhami, M. N. (2020). Data Mining Algoritma dan Implementasi. Yogyakarta: Andi.

Kusrini, E. T. (2009). Algoritma Data Mining. Yogyakarta: Andi.

PUSTAKA MAJALAH, JURNAL ILMIAH ATAU PROSIDING

Agus Perdana Windarto, D. H. (n.d.). Model Arsitektur Backpropagation dalam Memprediksi Faktor Tunggakan Uang Kuliah (Studi Amik Tunas Bangsa).

Ahmed, A. N. (2019). Credit-Risk Asesmen of Small Business Loans using Naive Bayes, Decision Tree and Random Forest. National Collage of Ireland.

Arno Prayogo Nawary, K. (2021). Penerapan data Mining dalam Memprediksi Kelancaran Kredit Nasabah Menggunakan Algoritma C4.5 (Studi kasus pada PT. Astra Internasional (Auto 2000 Plaju). Bina Darma Confren on Computer Science.

Aulia Rahmayanti, L. R. (2022). Perbandingan Metodologi Algoritma C4.5 dan Naive Bayes untuk Memprediksi Kelulusan Mahasiswa. Walisongo Journal of Information Technology.

Dewi, S. (2019). Komparasi Metode Algoritma Data Mining pada Prediksi Uji Kelayakan Credit Approval pada Calon Nasabah Kredit Perbankan. Jurnal Khatulistiwa Informatika .

Dwi Yuni Utami, E. N. (2021). Comparison of Neural Network Algorithms, Naive Bayes and Logistic Regression to Find the Highest Accuracy in Diabetes. Journal of Informatics and Telecommunication Engineering.

Elham Adakh, A. F. (2019). Comparison of Several Deep Data Mining Models Forecast of Bank Performance Received in Tehran Shares Exchange Market. Iriana Journal of Finance.

Erik Dwi Anggara, A. W. (2022). Prediksi KInerja sebagai Rekomendasi Kenaikan Golongan dengan Decision Tree dan Regresi Logistik. Jurnal Teknik Informatika dan Sistem Informasi.

Fatmi Zola, G. W. (2018). Jaringan Saraf Tiruan Menggunakan Algoritma Backpropagation untuk Memprediksi Prestasi Siswa. Jurnal Teknologi dan Open Source.

Gitania Aimbu, H. K. (2021). Analisis pengendalian piutang untuk Meminimalkan Resiko Piutang tak Tertagih pada PT Samudra Mandiri Sentosa. Jurnal Riset Akuntansi .

Hendri, H. (2021). Implementasi Data Mining dengan Metode C4.5 untuk Prediksi Mahasiswa Penerima Beasiswa. Indonesian Journal of Computer Science, 3.

Iqbal Taufik Ahmad Nur, N. Y. (2018). Perbandingan Performa Metode Klasifikasi SVM, Neural Network dan Naive Bayes untuk Mendeteksi Kualitas Pengajuan Kredit di Koperasi Simpan Pinjam. Jurnal Teknologi Informasi dan Ilmu Komputer.

Nur Hadianto, H. B. (2019). Klasifikasi Peminjaman Nasabah Bank Menggunakan Metode Neural Network. Jurnal Pilar Nusa Mandiri.

Putri, A. (2021). Analisis Perbandingan Biaya Kuliah Program S1 Akuntansi antar Perguruan Tinggi Swasta di bekasi. Lembaga Penelitian dan Pengabdian Masyarakat Universitas Islam "45" Bekasi.

Putri, A. P. (2022). Classification of the Fluency Multipurpose of Bank Mandiri Credit Payments Based on Debtor Preferences Using Naive Bayes and Neural Network Method . Jurnal Online Informatika.

Rendi Irawan, A. E. (2020). Comparison of Data Mining Classification Methods for Predicting Credit Appropriation through Naive BAyes and Decision Tree. Proceeding of 6th ICTB.

Victor Saputra Ginting, K. E. (2020). Implementasi Algoritma C4.5 untuk Memprediksi Keterlambatan Pembayaran Sumbangan Pembangunan Pendidikan Sekolah Menggunakan Python. Jurnal Teknologi Informasi dan Teknologi.

Viry Puspaning Ramadhan, F. Y. (2022). Analisis Perbandingan Algoritma Forecasting dalam Prediksi Harga Saham LQ45 PT Bank Mandiri Sekuritas BMRI. Jurnal Teknologi dan Manajemen Informatika , 4.

Woro Isti Rahayu, C. P. (2021). Perbandingan Algoritma K-Means dan Naïve Bayes untuk Memprediksi Prioritas Pembayaran Tagihan Rumah Sakit Berdasarkan Tingkat Kepentingan pada PT. Pertamina (PERSERO). Jurnal Teknik Informatika.

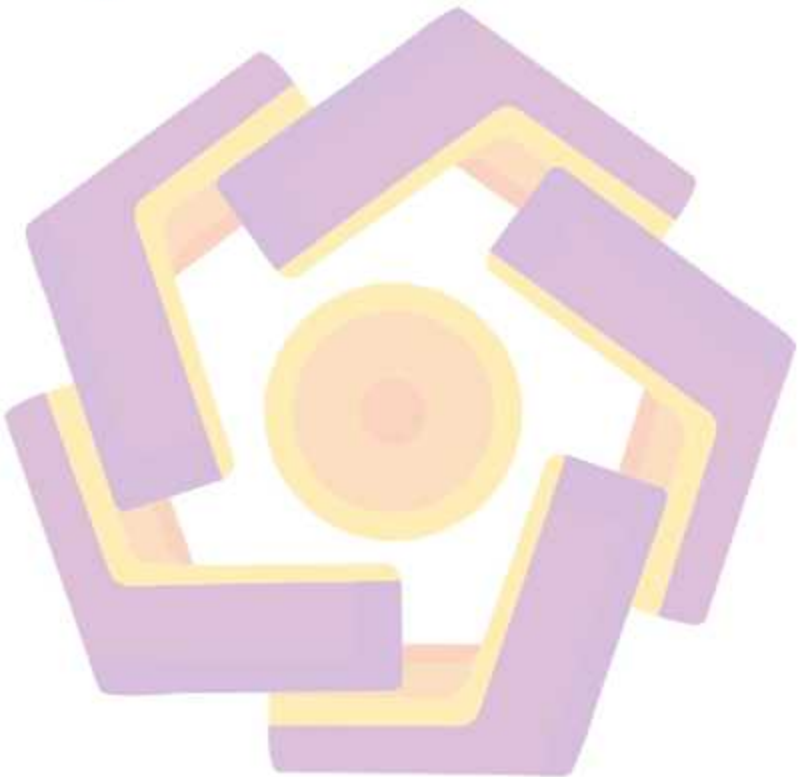
Xin Ma, Y. Z. (2019). Application of Data Mining in the Field of Human Resource Management : A Review. Atlantis Press.

Yogiek Indra Kurniawan, A. F. (2021). Prediction for Cooperative Credit Eligibility using Data Mining Classification with C4.5 Algoritma. Jurnal Teknik Informatika .

PUSTAKA LAPORAN PENELITIAN

Novianti, D. (2019). Prediksi Status Berlangganan Klien Bank menggunakan Algoritma Naive Bayes, C4.5, dan KNN Berbasis Ensemble Classifier.

Zulinda. (2019). Penerapan Jaringan Saraf Tiruan Extreme Learning Machine dan Backpropagation untuk Memprediksi Harga Saham PT Bank Mandiri (Persero) TBK.



LAMPIRAN

Percobaan 1

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
Membaca dataset
dataset = pd.read_csv('/content/drive/MyDrive/5.-Data-Finis-
Preposesing.csv')
x = dataset.iloc[:, :-1].values
y = dataset.iloc[:, -1].values
print (dataset)
```

```
           NIM  Status  Perguruan Tinggi  Program Studi
Jenjang  \
0    220542470001       1                 1              1
1
1    220542470002       1                 1              1
1
2    220542470003       1                 1              1
1
3    220542470004       1                 1              1
1
4    220542470005       1                 1              1
1
..            ...     ...               ...            ...
...
255  222942020023       1                 1              2
1
256  222942020024       1                 1              2
1
257  222942020025       1                 1              2
1
258  222942020026       1                 1              2
1
259  222942020029       1                 1              2
1

     Alamat Kelurahan /Desa  Alamat Kecamatan  Pendidikan Wali  \
0                        26                 1                3
1                        44                 1                3
2                        22                 2                5
3                        26                 1                2
4                        20                 2                5
..                      ...               ...              ...
255                      44                 1                9
256                      28                 1                5
257                      28                 1                5
258                      16                 2                5
259                      16                 2                5

     Pekerjaan Wali  Penghasilan Wali  Keterangan  \
0               3.0               2.0           1
```

```
1                2.0                4.0            1
2                2.0                5.0            1
3               11.0                2.0            1
4                2.0                4.0            1
..               ...                ...          ...
255              5.0                6.0            1
256              9.0                5.0            1
257              9.0                4.0            1
258             16.0                2.0            1
259              9.0                8.0            1

     Jumlah Piutang UKT Mahasiswa  Umur Piutang UKT Mahasiswa  \
0                         1700000                            3
1                               0                            1
2                          100000                            2
3                         3000000                            3
4                         1000000                            2
..                            ...                          ...
255                       1000000                            2
256                       1000000                            2
257                        500000                            2
258                       1000000                            2
259                       1000000                            2

     Jumlah Piutang BPP  Umur Piutang BPP  Status Piutang
0               2000000                 2               2
1                     0                 1               2
2               2300000                 3               2
3               2800000                 3               1
4               2000000                 2               2
..                  ...               ...             ...
255             3000000                 3               1
256             2000000                 2               1
257             2300000                 3               2
258             3000000                 3               2
259             2300000                 3               1

[260 rows x 16 columns]
```

```
Hitung jumlah atribut
jumlah_atribut = len(dataset.columns) - 1  # Mengurangi 1
karena kolom terakhir adalah kolom target
print("Jumlah atribut:", jumlah_atribut)
```

```
Jumlah atribut: 15
```

```
Hitung jumlah record
jumlah_record = len(dataset)
print("Jumlah record:", jumlah_record)
```

```
Jumlah record: 260
```

```
Hitung jumlah itemset untuk setiap atribut
for column in dataset.columns:
    jumlah_itemset = dataset[column].nunique()
    print("Jumlah itemset untuk atribut", column, ":",
jumlah_itemset)
```

```
Jumlah itemset untuk atribut NIM : 259
```

```
Jumlah itemset untuk atribut Status : 1
Jumlah itemset untuk atribut Perguruan Tinggi : 1
Jumlah itemset untuk atribut Program Studi : 3
Jumlah itemset untuk atribut Jenjang : 1
Jumlah itemset untuk atribut Alamat Kelurahan /Desa : 50
Jumlah itemset untuk atribut Alamat Kecamatan : 7
Jumlah itemset untuk atribut Pendidikan Wali : 8
Jumlah itemset untuk atribut Pekerjaan Wali : 14
Jumlah itemset untuk atribut Penghasilan Wali : 8
Jumlah itemset untuk atribut Keterangan : 1
Jumlah itemset untuk atribut Jumlah Piutang UKT Mahasiswa : 31
Jumlah itemset untuk atribut Umur Piutang UKT Mahasiswa : 3
Jumlah itemset untuk atribut Jumlah Piutang BPP : 24
Jumlah itemset untuk atribut Umur Piutang BPP : 3
Jumlah itemset untuk atribut Status Piutang : 2
```

```
Membagi dataset menjadi data latih dan data uji
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y,
test_size=0.20, random_state=0)
Melakukan preprocessing pada fitur-fitur numeric
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.transform(x_test)
Melakukan imputasi pada fitur-fitur yang memiliki missing
value
from sklearn.impute import SimpleImputer
imputer = SimpleImputer(strategy='mean')
x_train = imputer.fit_transform(x_train)
Membangun model Decision Tree (C4.5)
from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier()
classifier.fit(x_train, y_train)
```

DecisionTreeClassifier

```
DecisionTreeClassifier()
Melakukan prediksi pada data uji
x_test = imputer.transform(x_test)
Gunakan imputer yang sama untuk data uji
y_pred = classifier.predict(x_test)
Evaluasi model menggunakan confusion matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
[[17  1]
 [ 6 28]]
```

Percobaan 2

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
Membaca dataset
dataset = pd.read_csv('/content/drive/MyDrive/5.-Data-Finis-
Preposesing.csv')
x = dataset.iloc[:, :-1].values
y = dataset.iloc[:, -1].values
print (dataset)
```

```
           NIM  Status  Perguruan Tinggi  Program Studi
Jenjang  \
0    220542470001       1                 1                1
1
1    220542470002       1                 1                1
1
2    220542470003       1                 1                1
1
3    220542470004       1                 1                1
1
4    220542470005       1                 1                1
1
..            ...     ...               ...              ...
...
255  222942020023       1                 1                2
1
256  222942020024       1                 1                2
1
257  222942020025       1                 1                2
1
258  222942020026       1                 1                2
1
259  222942020029       1                 1                2
1

     Alamat Kelurahan /Desa  Alamat Kecamatan  Pendidikan Wali  \
0                        26                 1                3
1                        44                 1                3
2                        22                 2                5
3                        26                 1                2
4                        20                 2                5
..                      ...               ...              ...
255                      44                 1                9
256                      28                 1                5
257                      28                 1                5
258                      16                 2                5
259                      16                 2                5

     Pekerjaan Wali  Penghasilan Wali  Keterangan  \
0               3.0               2.0           1
1               2.0               4.0           1
2               2.0               5.0           1
3              11.0               2.0           1
```

```
4                2.0                4.0           1
..               ...                ...         ...
255              5.0                6.0           1
256              9.0                5.0           1
257              9.0                4.0           1
258             16.0                2.0           1
259              9.0                8.0           1

     Jumlah Piutang UKT Mahasiswa  Umur Piutang UKT Mahasiswa  \
0                         1700000                           3
1                               0                           1
2                          100000                           2
3                         3000000                           3
4                         1000000                           2
..                            ...                         ...
255                       1000000                           2
256                       1000000                           2
257                        500000                           2
258                       1000000                           2
259                       1000000                           2

     Jumlah Piutang BPP  Umur Piutang BPP  Status Piutang
0               2000000                 2               2
1                     0                 1               2
2               2300000                 3               2
3               2800000                 3               1
4               2000000                 2               2
..                  ...               ...             ...
255             3000000                 3               1
256             2000000                 2               1
257             2300000                 3               2
258             3000000                 3               2
259             2300000                 3               1

[260 rows x 16 columns]
```

Hitung jumlah atribut

```
jumlah_atribut = len(dataset.columns) - 1  # Mengurangi 1 karena kolom terakhir adalah kolom target
print("Jumlah atribut:", jumlah_atribut)
```

```
Jumlah atribut: 15
```

Hitung jumlah record

```
jumlah_record = len(dataset)
print("Jumlah record:", jumlah_record)
```

```
Jumlah record: 260
```

Hitung jumlah itemset untuk setiap atribut

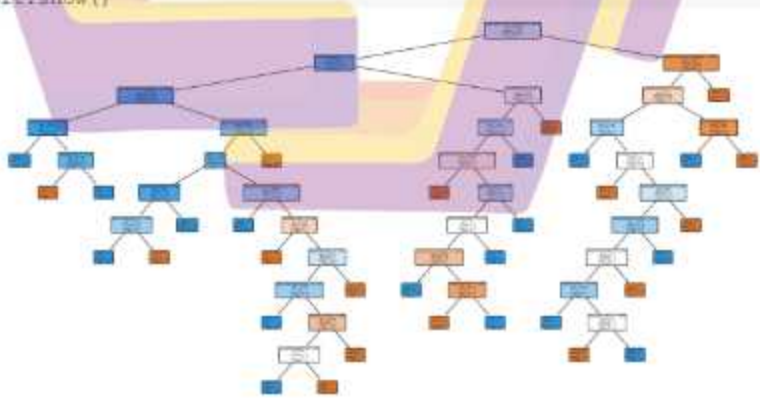
```
for column in dataset.columns:
    jumlah_itemset = dataset[column].nunique()
    print("Jumlah itemset untuk atribut", column, ":", jumlah_itemset)
```

```
Jumlah itemset untuk atribut NIM : 259
Jumlah itemset untuk atribut Status : 1
Jumlah itemset untuk atribut Perguruan Tinggi : 1
Jumlah itemset untuk atribut Program Studi : 3
```

```
Jumlah itemset untuk atribut Jenjang : 1
Jumlah itemset untuk atribut Alamat Kelurahan /Desa : 50
Jumlah itemset untuk atribut Alamat Kecamatan : 7
Jumlah itemset untuk atribut Pendidikan Wali : 8
Jumlah itemset untuk atribut Pekerjaan Wali : 14
Jumlah itemset untuk atribut Penghasilan Wali : 8
Jumlah itemset untuk atribut Keterangan : 1
Jumlah itemset untuk atribut Jumlah Piutang UKT Mahasiswa : 31
Jumlah itemset untuk atribut Umur Piutang UKT Mahasiswa : 3
Jumlah itemset untuk atribut Jumlah Piutang BPP : 24
Jumlah itemset untuk atribut Umur Piutang BPP : 3
Jumlah itemset untuk atribut Status Piutang : 2
Membagi dataset menjadi data latih dan data uji
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y,
test_size=0.10, random_state=0)
Melakukan preprocessing pada fitur-fitur numeric
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.transform(x_test)
Melakukan imputasi pada fitur-fitur yang memiliki missing value
from sklearn.impute import SimpleImputer
imputer = SimpleImputer(strategy='mean')
x_train = imputer.fit_transform(x_train)
Membangun model Decision Tree (C4.5)
from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier()
classifier.fit(x_train, y_train)
DecisionTreeClassifier
DecisionTreeClassifier()
Melakukan prediksi pada data uji
x_test = imputer.transform(x_test)
Gunakan imputer yang sama untuk data uji
y_pred = classifier.predict(x_test)
Evaluasi model menggunakan confusion matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
[[ 7  0]
 [ 2 17]]
!pip install scikit-learn
Requirement already satisfied: scikit-learn in
/usr/local/lib/python3.10/dist-packages (1.2.2)
Requirement already satisfied: numpy>=1.17.3 in
/usr/local/lib/python3.10/dist-packages (from scikit-learn)
(1.23.5)
Requirement already satisfied: scipy>=1.3.2 in
/usr/local/lib/python3.10/dist-packages (from scikit-learn)
(1.11.3)
```

```
Requirement already satisfied: joblib>=1.1.1 in
/usr/local/lib/python3.10/dist-packages (from scikit-learn)
(1.3.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in
/usr/local/lib/python3.10/dist-packages (from scikit-learn)
(3.2.0)
Mendapatkan feature importance
feature_importance = classifier.feature_importances_
Menampilkan feature importance untuk setiap atribut
for i, importance in enumerate(feature_importance):
    print(f'Feature {i}: {importance}')
Feature 0: 0.020561080852968094
Feature 1: 0.0
Feature 2: 0.0
Feature 3: 0.016689604743083
Feature 4: 0.0
Feature 5: 0.022364060217953044
Feature 6: 0.026857691363765514
Feature 7: 0.057334799861377377
Feature 8: 0.05474465148378192
Feature 9: 0.014130434782608694
Feature 10: 0.0
Feature 11: 0.6597544914645002
Feature 12: 0.0029438405797101446
Feature 13: 0.12460934464965569
Feature 14: 0.0

Munculkan gambar pohon keputusan
plt.figure(figsize=(20, 10))
plot_tree(classifier, filled=True,
feature_names=dataset.columns[:-1],
class_names=np.unique(y).astype('str'), rounded=True)
plt.show()
```



Percobaan 3

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection import cross_val_score, KFold
from sklearn.preprocessing import StandardScaler
from sklearn.impute import SimpleImputer
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix, precision_score,
recall_score
```

Membaca dataset

```
dataset = pd.read_csv('/content/drive/MyDrive/5.-Data-Finis-
Preposesing.csv')
x = dataset.iloc[:, :-1].values
y = dataset.iloc[:, -1].values
```

Preprocessing

```
sc = StandardScaler()
imputer = SimpleImputer(strategy='mean')
x = sc.fit_transform(x)
x = imputer.fit_transform(x)
```

Inisialisasi model Decision Tree (C4.5) dan jumlah subset (fold) untuk k-fold cross-validation:

```
classifier = DecisionTreeClassifier()
k = 5  # Jumlah subset (fold)
kf = KFold(n_splits=k, shuffle=True, random_state=0)
```

Inisialisasi list untuk menyimpan nilai akurasi, presisi, dan recall dari setiap iterasi

```
accuracies = []
precisions = []
recalls = []
```

Melakukan k-fold cross-validation

```
for train_index, test_index in kf.split(x):
    x_train, x_test = x[train_index], x[test_index]
    y_train, y_test = y[train_index], y[test_index]

    # Melatih model pada subset train
    classifier.fit(x_train, y_train)

    # Prediksi menggunakan subset test
    y_pred = classifier.predict(x_test)

    # Menghitung confusion matrix pada setiap iterasi
    cm = confusion_matrix(y_test, y_pred)
```

```
    # Menghitung dan menyimpan akurasi, presisi, dan recall
pada setiap iterasi
    accuracy = np.mean(y_pred == y_test)
    precision = precision_score(y_test, y_pred)
    recall = recall_score(y_test, y_pred)

    accuracies.append(accuracy)
    precisions.append(precision)
    recalls.append(recall)
```

Menghitung rata-rata akurasi, presisi, dan recall dari semua iterasi

```
mean_accuracy = np.mean(accuracies)
mean_precision = np.mean(precisions)
mean_recall = np.mean(recalls)

print(f"Mean Accuracy: {mean_accuracy}")
print(f"Mean Precision: {mean_precision}")
print(f"Mean Recall: {mean_recall}")
Mean Accuracy: 0.8307692307692307
Mean Precision: 0.7802246418722392
Mean Recall: 0.8185635366413399
```

Percobaan 4

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import tensorflow as tf
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.impute import SimpleImputer
from sklearn.metrics import confusion_matrix
from imblearn.over_sampling import RandomOverSampler
```

Membaca dataset

```
dataset = pd.read_csv('/content/drive/MyDrive/5.-Data-Finis-
Preposesing.csv')
x = dataset.iloc[:, :-1].values
y = dataset.iloc[:, -1].values
```

Terapkan oversampling untuk menangani ketidakseimbangan kelas

```
oversample = RandomOverSampler(sampling_strategy='minority')
x, y = oversample.fit_resample(x, y)
```

Membagi dataset menjadi data latih dan data uji

```
x_train, x_test, y_train, y_test = train_test_split(x, y,
test_size=0.20, random_state=5)
```

Melakukan preprocessing pada fitur-fitur numeric

```
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.transform(x_test)
```

Melakukan imputasi pada fitur-fitur yang memiliki missing value

```
imputer = SimpleImputer(strategy='mean')
x_train = imputer.fit_transform(x_train)
x_test = imputer.transform(x_test)
```

Membangun model Neural Network dengan TensorFlow

```
model = tf.keras.models.Sequential([
    tf.keras.layers.Dense(128, activation='relu',
input_shape=(x_train.shape[1],)),
    tf.keras.layers.Dropout(0.5),  # Add dropout layer
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])
```

Mengompilasi model

```
model.compile(optimizer='adam', loss='binary_crossentropy',
metrics=['accuracy'])
```

Melatih model

```
model.fit(x_train, y_train, batch_size=32, epochs=20,
validation_split=0.1, verbose=1)
```

```
Epoch 1/20
8/8 [==============================] - 1s 29ms/step - loss: 0.6464
- accuracy: 0.1556 - val_loss: -0.1339 - val_accuracy: 0.3462
Epoch 2/20
8/8 [==============================] - 0s 9ms/step - loss: -0.1351
- accuracy: 0.4800 - val_loss: -0.8830 - val_accuracy: 0.3462
Epoch 3/20
8/8 [==============================] - 0s 7ms/step - loss: -0.8014
- accuracy: 0.5067 - val_loss: -1.6413 - val_accuracy: 0.3462
Epoch 4/20
8/8 [==============================] - 0s 7ms/step - loss: -1.5192
- accuracy: 0.5067 - val_loss: -2.4762 - val_accuracy: 0.3462
Epoch 5/20
8/8 [==============================] - 0s 7ms/step - loss: -2.1260
- accuracy: 0.5067 - val_loss: -3.4135 - val_accuracy: 0.3462
Epoch 6/20
8/8 [==============================] - 0s 7ms/step - loss: -3.0494
- accuracy: 0.5067 - val_loss: -4.5466 - val_accuracy: 0.3462
Epoch 7/20
8/8 [==============================] - 0s 7ms/step - loss: -3.9483
- accuracy: 0.5067 - val_loss: -5.9892 - val_accuracy: 0.3462
Epoch 8/20
8/8 [==============================] - 0s 7ms/step - loss: -5.3067
- accuracy: 0.5067 - val_loss: -7.8136 - val_accuracy: 0.3462
Epoch 9/20
8/8 [==============================] - 0s 7ms/step - loss: -6.8885
- accuracy: 0.5067 - val_loss: -10.0128 - val_accuracy: 0.3462
Epoch 10/20
8/8 [==============================] - 0s 7ms/step - loss: -8.6311
- accuracy: 0.5067 - val_loss: -12.5241 - val_accuracy: 0.3462
Epoch 11/20
8/8 [==============================] - 0s 7ms/step - loss: -
10.7383 - accuracy: 0.5067 - val_loss: -15.4577 - val_accuracy:
0.3462
Epoch 12/20
8/8 [==============================] - 0s 7ms/step - loss: -
13.5027 - accuracy: 0.5067 - val_loss: -19.0799 - val_accuracy:
0.3462
Epoch 13/20
8/8 [==============================] - 0s 8ms/step - loss: -
16.7328 - accuracy: 0.5067 - val_loss: -23.6218 - val_accuracy:
0.3462
Epoch 14/20
8/8 [==============================] - 0s 8ms/step - loss: -
19.9161 - accuracy: 0.5067 - val_loss: -29.1703 - val_accuracy:
0.3462
```

```
Epoch 15/20
8/8 [==============================] - 0s 6ms/step - loss: -
25.1071 - accuracy: 0.5067 - val_loss: -36.0063 - val_accuracy:
0.3462
Epoch 16/20
8/8 [==============================] - 0s 6ms/step - loss: -
30.8245 - accuracy: 0.5067 - val_loss: -43.9665 - val_accuracy:
0.3462
Epoch 17/20
8/8 [==============================] - 0s 7ms/step - loss: -
38.0286 - accuracy: 0.5067 - val_loss: -54.0498 - val_accuracy:
0.3462
Epoch 18/20
8/8 [==============================] - 0s 7ms/step - loss: -
44.8369 - accuracy: 0.5067 - val_loss: -65.7843 - val_accuracy:
0.3462
Epoch 19/20
8/8 [==============================] - 0s 7ms/step - loss: -
55.5955 - accuracy: 0.5067 - val_loss: -79.9842 - val_accuracy:
0.3462
Epoch 20/20
8/8 [==============================] - 0s 7ms/step - loss: -
68.9124 - accuracy: 0.5067 - val_loss: -96.1762 - val_accuracy:
0.3462
<keras.src.callbacks.History at 0x7e3dfed18a90>
```

Melakukan prediksi pada data uji

```
y_pred_probs = model.predict(x_test)
y_pred = (y_pred_probs > 0.5).astype(int)
2/2 [==============================] - 0s 4ms/step
```

Evaluasi model menggunakan confusion matrix

```
cm = confusion_matrix(y_test, y_pred)
print(cm)
[[34  0]
 [29  0]]
```

Membuat DataFrame dari nama atribut dan bobot pada lapisan input

```
atribut_dan_bobot = pd.DataFrame({'Nama Atribut':
dataset.columns[:-1], 'Bobot':
model.layers[0].get_weights()[0][:, 0]})
```

Menampilkan DataFrame

```
print(atribut_dan_bobot)
    Nama Atribut     Bobot
0                       NIM  0.019846
1                    Status  0.113110
2          Perguruan Tinggi  0.179026
3             Program Studi -0.265025
4                   Jenjang  0.157990
```

```
5          Alamat Kelurahan /Desa  0.028698
6                Alamat Kecamatan  0.112230
7                 Pendidikan Wali -0.046588
8                  Pekerjaan Wali -0.016708
9                Penghasilan Wali -0.236636
10                     Keterangan  0.114750
11  Jumlah Piutang UKT Mahasiswa -0.175040
12    Umur Piutang UKT Mahasiswa -0.055177
13            Jumlah Piutang BPP -0.175849
14              Umur Piutang BPP -0.023899
```

Percobaan 5

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import tensorflow as tf
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.impute import SimpleImputer
from sklearn.metrics import confusion_matrix
from imblearn.over_sampling import RandomOverSampler
```

Membaca dataset

```
dataset = pd.read_csv('/content/drive/MyDrive/5.-Data-Finis-
Preposesing.csv')
x = dataset.iloc[:, :-1].values
y = dataset.iloc[:, -1].values
```

Terapkan oversampling untuk menangani ketidakseimbangan kelas

```
oversample = RandomOverSampler(sampling_strategy='minority')
x, y = oversample.fit_resample(x, y)
```

Membagi dataset menjadi data latih dan data uji

```
x_train, x_test, y_train, y_test = train_test_split(x, y,
test_size=0.10, random_state=0)
```

Melakukan preprocessing pada fitur-fitur numeric

```
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.transform(x_test)
```

Melakukan imputasi pada fitur-fitur yang memiliki missing value

```
imputer = SimpleImputer(strategy='mean')
x_train = imputer.fit_transform(x_train)
```

```
x_test = imputer.transform(x_test)
```

Membangun model Neural Network dengan TensorFlow

```
model = tf.keras.models.Sequential([
    tf.keras.layers.Dense(128, activation='relu',
input_shape=(x_train.shape[1],)),
    tf.keras.layers.Dropout(0.5),  # Add dropout layer
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])
```

Mengompilasi model

```
model.compile(optimizer='adam', loss='binary_crossentropy',
metrics=['accuracy'])
```

Melatih model

```
model.fit(x_train, y_train, batch_size=32, epochs=20,
validation_split=0.1, verbose=1)
```

```
Epoch 1/20
8/8 [==============================] - 1s 28ms/step - loss: 0.4226
- accuracy: 0.3399 - val_loss: -0.2291 - val_accuracy: 0.4483
Epoch 2/20
8/8 [==============================] - 0s 6ms/step - loss: -0.4311
- accuracy: 0.4980 - val_loss: -0.9836 - val_accuracy: 0.4483
Epoch 3/20
8/8 [==============================] - 0s 9ms/step - loss: -1.1508
- accuracy: 0.4980 - val_loss: -1.7588 - val_accuracy: 0.4483
Epoch 4/20
8/8 [==============================] - 0s 6ms/step - loss: -1.8408
- accuracy: 0.4980 - val_loss: -2.6762 - val_accuracy: 0.4483
Epoch 5/20
8/8 [==============================] - 0s 10ms/step - loss: -
2.7302 - accuracy: 0.4980 - val_loss: -3.8215 - val_accuracy:
0.4483
Epoch 6/20
8/8 [==============================] - 0s 9ms/step - loss: -3.9099
- accuracy: 0.4980 - val_loss: -5.2894 - val_accuracy: 0.4483
Epoch 7/20
8/8 [==============================] - 0s 7ms/step - loss: -5.5114
- accuracy: 0.4980 - val_loss: -7.2041 - val_accuracy: 0.4483
Epoch 8/20
8/8 [==============================] - 0s 7ms/step - loss: -7.6291
- accuracy: 0.4980 - val_loss: -9.7327 - val_accuracy: 0.4483
Epoch 9/20
8/8 [==============================] - 0s 6ms/step - loss: -9.9724
- accuracy: 0.4980 - val_loss: -12.9868 - val_accuracy: 0.4483
Epoch 10/20
```

```
8/8 [==============================] - 0s 6ms/step - loss: -
12.9874 - accuracy: 0.4980 - val_loss: -17.0872 - val_accuracy:
0.4483
Epoch 11/20
8/8 [==============================] - 0s 6ms/step - loss: -
17.6413 - accuracy: 0.4980 - val_loss: -22.0764 - val_accuracy:
0.4483
Epoch 12/20
8/8 [==============================] - 0s 7ms/step - loss: -
22.4693 - accuracy: 0.4980 - val_loss: -28.4279 - val_accuracy:
0.4483
Epoch 13/20
8/8 [==============================] - 0s 6ms/step - loss: -
29.3386 - accuracy: 0.4980 - val_loss: -36.1115 - val_accuracy:
0.4483
Epoch 14/20
8/8 [==============================] - 0s 7ms/step - loss: -
35.3893 - accuracy: 0.4980 - val_loss: -45.5943 - val_accuracy:
0.4483
Epoch 15/20
8/8 [==============================] - 0s 7ms/step - loss: -
45.3758 - accuracy: 0.4980 - val_loss: -56.5992 - val_accuracy:
0.4483
Epoch 16/20
8/8 [==============================] - 0s 7ms/step - loss: -
56.8135 - accuracy: 0.4980 - val_loss: -69.9936 - val_accuracy:
0.4483
Epoch 17/20
8/8 [==============================] - 0s 7ms/step - loss: -
69.9285 - accuracy: 0.4980 - val_loss: -85.6022 - val_accuracy:
0.4483
Epoch 18/20
8/8 [==============================] - 0s 9ms/step - loss: -
85.1883 - accuracy: 0.4980 - val_loss: -103.3277 - val_accuracy:
0.4483
Epoch 19/20
8/8 [==============================] - 0s 6ms/step - loss: -
101.3752 - accuracy: 0.4980 - val_loss: -124.6510 - val_accuracy:
0.4483
Epoch 20/20
8/8 [==============================] - 0s 6ms/step - loss: -
122.7431 - accuracy: 0.4980 - val_loss: -148.6072 - val_accuracy:
0.4483
<keras.src.callbacks.History at 0x7e3dfeac2500>
```

Melakukan prediksi pada data uji

```
y_pred_probs = model.predict(x_test)
y_pred = (y_pred_probs > 0.5).astype(int)
1/1 [==============================] - 0s 54ms/step
```

Evaluasi model menggunakan confusion matrix

```
cm = confusion_matrix(y_test, y_pred)
```

```
print(cm)
[[18  0]
 [14  0]]
```

Percobaan 6

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection import cross_val_score, KFold
from sklearn.preprocessing import StandardScaler
from sklearn.impute import SimpleImputer
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import confusion_matrix, precision_score,
recall_score
```

Membaca dataset

```
dataset = pd.read_csv('/content/drive/MyDrive/5.-Data-Finis-
Preposesing.csv')
x = dataset.iloc[:, :-1].values
y = dataset.iloc[:, -1].values
```

Preprocessing data

```
sc = StandardScaler()
imputer = SimpleImputer(strategy='mean')
x = sc.fit_transform(x)
x = imputer.fit_transform(x)
```

Inisialisasi model Artificial Neural Network (ANN) dan jumlah subset (fold) untuk k-fold cross-validation

```
classifier = MLPClassifier(hidden_layer_sizes=(100,),
max_iter=1000, random_state=0)
k = 5  # Jumlah subset (fold)
kf = KFold(n_splits=k, shuffle=True, random_state=0)
```

Inisialisasi list untuk menyimpan nilai akurasi, presisi, dan recall dari setiap iterasi

```
accuracies = []
precisions = []
recalls = []
```

Melakukan k-fold cross-validation

```
for train_index, test_index in kf.split(x):
    x_train, x_test = x[train_index], x[test_index]
```

```
    y_train, y_test = y[train_index], y[test_index]

    # Melatih model pada subset train
    classifier.fit(x_train, y_train)

    # Prediksi menggunakan subset test
    y_pred = classifier.predict(x_test)

    # Menghitung confusion matrix pada setiap iterasi
    cm = confusion_matrix(y_test, y_pred)

    # Menghitung dan menyimpan akurasi, presisi, dan recall
pada setiap iterasi
    accuracy = np.mean(y_pred == y_test)
    precision = precision_score(y_test, y_pred)
    recall = recall_score(y_test, y_pred)

    accuracies.append(accuracy)
    precisions.append(precision)
    recalls.append(recall)
```

Menghitung rata-rata akurasi, presisi, dan recall dari semua iterasi

```
mean_accuracy = np.mean(accuracies)
mean_precision = np.mean(precisions)
mean_recall = np.mean(recalls)

print(f"Mean Accuracy: {mean_accuracy}")
print(f"Mean Precision: {mean_precision}")
print(f"Mean Recall: {mean_recall}")
```

```
Mean Accuracy: 0.8576923076923076
Mean Precision: 0.8479904306220096
Mean Recall: 0.7889671477772164
```

Percobaan 7

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
dataset = pd.read_csv('/content/drive/MyDrive/5.-Data-Finis-
Preposesing.csv')
x = dataset.iloc[:,:-1].values
y = dataset.iloc[:,-1].values
print(x)
```

```
[[2.2054247e+11 1.0000000e+00 1.0000000e+00 ... 3.0000000e+00
  2.0000000e+06 2.0000000e+00]
 [2.2054247e+11 1.0000000e+00 1.0000000e+00 ... 1.0000000e+00
  0.0000000e+00 1.0000000e+00]
 [2.2054247e+11 1.0000000e+00 1.0000000e+00 ... 2.0000000e+00
  2.3000000e+06 3.0000000e+00]
 ...
 [2.2294202e+11 1.0000000e+00 1.0000000e+00 ... 2.0000000e+00
  2.3000000e+06 3.0000000e+00]
 [2.2294202e+11 1.0000000e+00 1.0000000e+00 ... 2.0000000e+00
  3.0000000e+06 3.0000000e+00]
 [2.2294202e+11 1.0000000e+00 1.0000000e+00 ... 2.0000000e+00
  2.3000000e+06 3.0000000e+00]]
```

```
print(y)
```

```
[2 2 2 1 2 2 2 2 2 2 2 2 2 2 1 1 2 2 2 2 1 1 1 2 2 1 1 1 1 2 1 2 2
2 1 2 1
 2 1 2 2 2 2 1 1 2 1 2 1 2 2 2 2 1 1 2 2 2 2 1 2 1 1 2 2 1 1 2 2 2
2 1 1 2
 1 1 1 1 2 1 2 1 2 2 1 2 1 1 2 2 1 1 1 2 1 2 2 2 2 1 1 1 1 1 1 1 2
1 1 1 1
 1 1 2 2 2 1 1 1 1 1 1 1 1 1 2 2 2 2 1 2 1 1 2 2 1 2 2 2 1 2 1 2
1 2 2 1
 2 2 1 1 2 2 2 1 2 2 2 1 1 1 2 1 2 2 2 1 2 1 1 2 2 2 2 2 2 2 1 2 2 1
2 2 2 1
 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 1 1 1 2 1 2
2 1 2 2
 2 1 2 1 2 1 2 1 2 2 2 2 2 2 1 1 2 2 2 2 1 2 2 1 1 1 2 2 2 2 2 2 2
1 1 2 2
 1]
```

Membagi dataset menjadi data latih dan data uji

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test =
train_test_split(x,y,test_size=0.20, random_state=0)
print(x_train)
```

```
[[2.2157201e+11 1.0000000e+00 1.0000000e+00 ... 3.0000000e+00
  2.3000000e+06 3.0000000e+00]
 [2.2254247e+11 1.0000000e+00 1.0000000e+00 ... 2.0000000e+00
  2.3000000e+06 3.0000000e+00]
 [2.2094202e+11 1.0000000e+00 1.0000000e+00 ... 3.0000000e+00
  2.8000000e+06 3.0000000e+00]
 ...
```

```
 [2.2157201e+11 1.0000000e+00 1.0000000e+00 ... 3.0000000e+00
  3.0000000e+06 3.0000000e+00]
 [2.2057201e+11 1.0000000e+00 1.0000000e+00 ... 1.0000000e+00
  8.0000000e+05 2.0000000e+00]
 [2.2254247e+11 1.0000000e+00 1.0000000e+00 ... 1.0000000e+00
  2.3000000e+06 3.0000000e+00]]
print(x_test)
[[2.2257201e+11 1.0000000e+00 1.0000000e+00 3.0000000e+00
1.0000000e+00
  2.8000000e+01 1.0000000e+00 5.0000000e+00 9.0000000e+00
5.0000000e+00
  1.0000000e+00 3.0000000e+05 2.0000000e+00 2.0000000e+06
2.0000000e+00]
 [2.2154247e+11 1.0000000e+00 1.0000000e+00 1.0000000e+00
1.0000000e+00
  2.4000000e+01 2.0000000e+00 3.0000000e+00 1.0000000e+00
4.0000000e+00
  1.0000000e+00 1.0000000e+05 2.0000000e+00 2.3000000e+06
3.0000000e+00]
 [2.2294202e+11 1.0000000e+00 1.0000000e+00 2.0000000e+00
1.0000000e+00
  1.6000000e+01 2.0000000e+00 3.0000000e+00 3.0000000e+00
4.0000000e+00
  1.0000000e+00 1.0000000e+06 2.0000000e+00 2.3000000e+06
3.0000000e+00]
 [2.2094202e+11 1.0000000e+00 1.0000000e+00 2.0000000e+00
1.0000000e+00
  3.6000000e+01 4.0000000e+00 5.0000000e+00 9.0000000e+00
6.0000000e+00
  1.0000000e+00 3.0000000e+06 3.0000000e+00 2.3000000e+06
3.0000000e+00]
 [2.2254247e+11 1.0000000e+00 1.0000000e+00 1.0000000e+00
1.0000000e+00
  2.2000000e+01 2.0000000e+00 3.0000000e+00 2.0000000e+00
4.0000000e+00
  1.0000000e+00 7.5000000e+05 2.0000000e+00 2.3000000e+06
3.0000000e+00]
 [2.2254247e+11 1.0000000e+00 1.0000000e+00 1.0000000e+00
1.0000000e+00
  1.8000000e+01 2.0000000e+00 3.0000000e+00 3.0000000e+00
4.0000000e+00
  1.0000000e+00 0.0000000e+00 1.0000000e+00 1.6000000e+06
2.0000000e+00]
 [2.2054247e+11 1.0000000e+00 1.0000000e+00 1.0000000e+00
1.0000000e+00
  3.0000000e+00 3.0000000e+00 3.0000000e+00 2.0000000e+00
2.0000000e+00
  1.0000000e+00 0.0000000e+00 1.0000000e+00 2.5000000e+06
3.0000000e+00]
 [2.2094202e+11 1.0000000e+00 1.0000000e+00 2.0000000e+00
1.0000000e+00
  1.6000000e+01 2.0000000e+00 5.0000000e+00 5.0000000e+00
6.0000000e+00
  1.0000000e+00 4.0000000e+06 3.0000000e+00 2.8000000e+06
3.0000000e+00]
```

```
 [2.2057201e+11 1.0000000e+00 1.0000000e+00 3.0000000e+00
1.0000000e+00
  2.7000000e+01 1.0000000e+00 4.0000000e+00 1.0000000e+00
1.0000000e+00
  1.0000000e+00 0.0000000e+00 1.0000000e+00 1.3000000e+06
2.0000000e+00]
 [2.2194202e+11 1.0000000e+00 1.0000000e+00 2.0000000e+00
1.0000000e+00
  5.0000000e+01 2.0000000e+00 3.0000000e+00 3.0000000e+00
4.0000000e+00
  1.0000000e+00 0.0000000e+00 1.0000000e+00 1.0000000e+06
2.0000000e+00]
 [2.2254247e+11 1.0000000e+00 1.0000000e+00 1.0000000e+00
1.0000000e+00
  2.8000000e+01 1.0000000e+00 5.0000000e+00 1.2000000e+01
6.0000000e+00
  1.0000000e+00 1.0000000e+06 2.0000000e+00 3.0000000e+06
3.0000000e+00]
 [2.2057201e+11 1.0000000e+00 1.0000000e+00 3.0000000e+00
1.0000000e+00
  2.9000000e+01 1.0000000e+00 4.0000000e+00 3.0000000e+00
4.0000000e+00
  1.0000000e+00 0.0000000e+00 1.0000000e+00 1.0000000e+06
2.0000000e+00]
 [2.2257201e+11 1.0000000e+00 1.0000000e+00 3.0000000e+00
1.0000000e+00
  2.9000000e+01 1.0000000e+00 5.0000000e+00 5.0000000e+00
6.0000000e+00
  1.0000000e+00 0.0000000e+00 1.0000000e+00 2.0000000e+06
2.0000000e+00]
 [2.2154247e+11 1.0000000e+00 1.0000000e+00 1.0000000e+00
1.0000000e+00
  1.7000000e+01 3.0000000e+00 1.0000000e+00 2.0000000e+00
2.0000000e+00
  1.0000000e+00 3.0000000e+06 3.0000000e+00 3.0000000e+06
3.0000000e+00]
 [2.2257201e+11 1.0000000e+00 1.0000000e+00 3.0000000e+00
1.0000000e+00
  1.1000000e+01 2.0000000e+00 9.0000000e+00 5.0000000e+00
6.0000000e+00
  1.0000000e+00 0.0000000e+00 1.0000000e+00 2.3000000e+06
3.0000000e+00]
 [2.2057201e+11 1.0000000e+00 1.0000000e+00 3.0000000e+00
1.0000000e+00
  2.8000000e+01 1.0000000e+00 5.0000000e+00 9.0000000e+00
4.0000000e+00
  1.0000000e+00 0.0000000e+00 1.0000000e+00 2.8000000e+06
3.0000000e+00]
 [2.2294202e+11 1.0000000e+00 1.0000000e+00 2.0000000e+00
1.0000000e+00
  4.1000000e+01 1.0000000e+00 2.0000000e+00 3.0000000e+00
2.0000000e+00
  1.0000000e+00 0.0000000e+00 1.0000000e+00 2.3000000e+06
3.0000000e+00]
 [2.2254247e+11 1.0000000e+00 1.0000000e+00 1.0000000e+00
1.0000000e+00
```

```
  1.9000000e+01 2.0000000e+00 5.0000000e+00 5.0000000e+00
6.0000000e+00
  1.0000000e+00 0.0000000e+00 1.0000000e+00 2.3000000e+06
3.0000000e+00]
 [2.2157201e+11 1.0000000e+00 1.0000000e+00 3.0000000e+00
1.0000000e+00
  1.0000000e+01 2.0000000e+00 5.0000000e+00 2.0000000e+00
5.0000000e+00
  1.0000000e+00 0.0000000e+00 1.0000000e+00 2.0000000e+06
2.0000000e+00]
 [2.2194202e+11 1.0000000e+00 1.0000000e+00 2.0000000e+00
1.0000000e+00
```

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.fit_transform(x_test)
```

```
print(x_train)
```

```
[[-0.10408483  0.          0.          ...  1.28144697  0.22171482
   0.72551234]
 [ 1.03137965  0.          0.          ...  0.06436492  0.22171482
   0.72551234]
 [-0.84119018  0.          0.          ...  1.28144697  0.85956659
   0.72551234]
 ...
 [-0.10408484  0.          0.          ...  1.28144697  1.1147073
   0.72551234]
 [-1.27411187  0.          0.          ... -1.15271714 -1.69184049
  -0.91477643]
 [ 1.03137964  0.          0.          ... -1.15271714  0.22171482
   0.72551234]]
```

```
print(y_train)
```

```
[1 1 1 2 2 1 2 1 1 2 2 1 1 2 1 2 1 2 1 1 2 2 2 2 2 1 1 2 2 2 2 2 1
1 2 1 1
 2 2 2 2 2 2 1 1 2 1 2 2 2 2 2 2 2 1 2 2 2 2 2 1 1 1 2 2 2 2 2 1
1 2 2 1
 2 1 1 1 1 1 2 1 2 2 1 2 2 2 2 1 1 2 2 2 2 1 2 2 1 2 2 2 2 1 2 2 2 2 2
2 2 1 2
 2 2 2 2 1 1 2 2 2 1 1 1 2 1 2 1 1 2 2 1 1 2 1 2 2 2 1 1 2 2 1 1 1 1
2 2 2 1
 1 2 2 1 1 2 2 1 2 2 1 2 2 1 1 1 1 2 1 2 1 2 1 2 1 2 1 2 2 2 1 1 1
2 1 2 2
 1 2 2 2 2 1 2 1 2 1 1 1 1 2 2 1 2 2 2 2 1 2 2]
```

```
from sklearn.impute import SimpleImputer

imputer = SimpleImputer(strategy='mean')
x_train = imputer.fit_transform(x_train)
```

Membangun model Naive Bayes dengan GaussianNB

```
from sklearn.naive_bayes import GaussianNB
classifier = GaussianNB()
classifier.fit(x_train,y_train)
```

GaussianNB

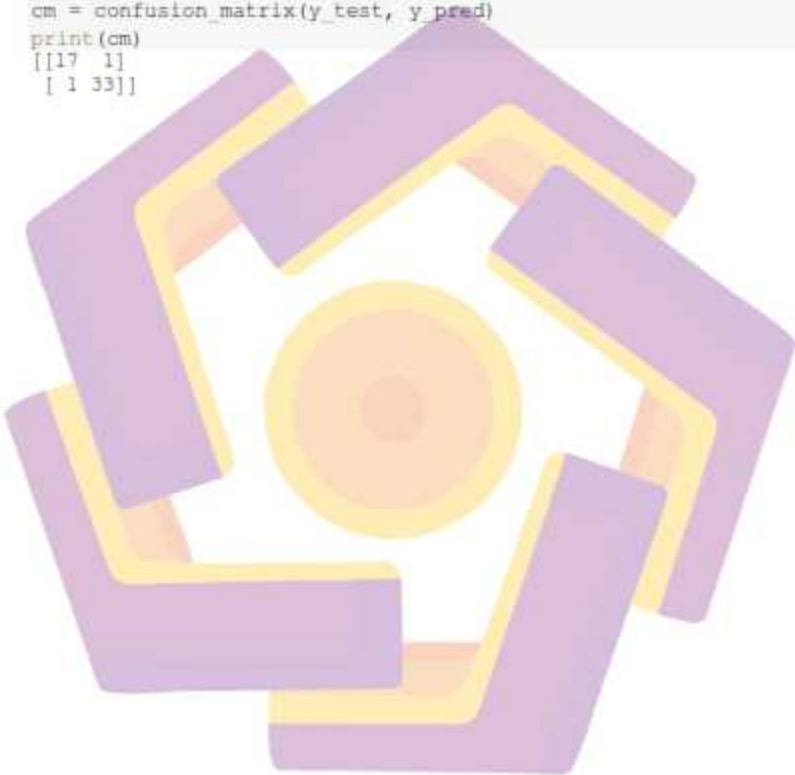
```
GaussianNB()
```

Melakukan prediksi pada data uji

```
y_pred = classifier.predict(x_test)
```

Evaluasi model menggunakan confusion matrix

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
[[17  1]
 [ 1 33]]
```



Percobaan 8

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
dataset = pd.read_csv('/content/drive/MyDrive/5.-Data-Finis-
Preposesing.csv')
x = dataset.iloc[:,:-1].values
y = dataset.iloc[:,-1].values
print(x)
```

```
[[2.2054247e+11 1.0000000e+00 1.0000000e+00 ... 3.0000000e+00
  2.0000000e+06 2.0000000e+00]
 [2.2054247e+11 1.0000000e+00 1.0000000e+00 ... 1.0000000e+00
  0.0000000e+00 1.0000000e+00]
 [2.2054247e+11 1.0000000e+00 1.0000000e+00 ... 2.0000000e+00
  2.3000000e+06 3.0000000e+00]
 ...
 [2.2294202e+11 1.0000000e+00 1.0000000e+00 ... 2.0000000e+00
  2.3000000e+06 3.0000000e+00]
 [2.2294202e+11 1.0000000e+00 1.0000000e+00 ... 2.0000000e+00
  3.0000000e+06 3.0000000e+00]
 [2.2294202e+11 1.0000000e+00 1.0000000e+00 ... 2.0000000e+00
  2.3000000e+06 3.0000000e+00]]
```

```
print(y)
```

```
[2 2 2 1 2 2 2 2 2 2 2 2 2 2 1 1 2 2 2 2 1 1 1 2 2 1 1 1 1 2 1 2 2
2 1 2 1
 2 1 2 2 2 2 1 1 2 1 2 1 2 2 2 2 1 1 2 2 2 2 1 2 1 1 2 2 1 1 2 2 2
2 1 1 2
 1 1 1 1 2 1 2 1 2 2 1 2 1 1 2 2 1 1 1 2 1 2 2 2 2 1 1 1 1 1 1 1 2
1 1 1 1
 1 1 2 2 2 1 1 1 1 1 1 1 1 1 2 2 2 2 1 2 1 1 2 2 1 2 2 2 1 2 1 2
1 2 2 1
 2 2 1 1 2 2 2 1 2 2 2 1 1 1 2 1 2 2 2 1 2 1 1 2 2 2 2 2 2 2 1 2 2 1
2 2 2 1
 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 1 1 1 2 1 2
2 1 2 2
 2 1 2 1 2 1 2 1 2 2 2 2 2 2 1 1 2 2 2 2 1 2 2 1 1 1 2 2 2 2 2 2 2
1 1 2 2
 1]
```

Membagi dataset menjadi data latih dan data uji

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test =
train_test_split(x,y,test_size=0.10, random_state=0)
print(x_train)
```

```
[[2.2094202e+11 1.0000000e+00 1.0000000e+00 ... 2.0000000e+00
  1.0000000e+06 2.0000000e+00]
 [2.2257201e+11 1.0000000e+00 1.0000000e+00 ... 2.0000000e+00
  2.8000000e+06 3.0000000e+00]
 [2.2094202e+11 1.0000000e+00 1.0000000e+00 ... 2.0000000e+00
  8.0000000e+05 2.0000000e+00]
 ...
```

```
 [2.2157201e+11 1.0000000e+00 1.0000000e+00 ... 3.0000000e+00
  3.0000000e+06 3.0000000e+00]
 [2.2057201e+11 1.0000000e+00 1.0000000e+00 ... 1.0000000e+00
  8.0000000e+05 2.0000000e+00]
 [2.2254247e+11 1.0000000e+00 1.0000000e+00 ... 1.0000000e+00
  2.3000000e+06 3.0000000e+00]]
```

```
print(x_test)
```

```
[[2.2257201e+11 1.0000000e+00 1.0000000e+00 3.0000000e+00
1.0000000e+00
  2.8000000e+01 1.0000000e+00 5.0000000e+00 9.0000000e+00
5.0000000e+00
  1.0000000e+00 3.0000000e+05 2.0000000e+00 2.0000000e+06
2.0000000e+00]
 [2.2154247e+11 1.0000000e+00 1.0000000e+00 1.0000000e+00
1.0000000e+00
  2.4000000e+01 2.0000000e+00 3.0000000e+00 1.0000000e+00
4.0000000e+00
  1.0000000e+00 1.0000000e+05 2.0000000e+00 2.3000000e+06
3.0000000e+00]
 [2.2294202e+11 1.0000000e+00 1.0000000e+00 2.0000000e+00
1.0000000e+00
  1.6000000e+01 2.0000000e+00 3.0000000e+00 3.0000000e+00
4.0000000e+00
  1.0000000e+00 1.0000000e+06 2.0000000e+00 2.3000000e+06
3.0000000e+00]
 [2.2094202e+11 1.0000000e+00 1.0000000e+00 2.0000000e+00
1.0000000e+00
  3.6000000e+01 4.0000000e+00 5.0000000e+00 9.0000000e+00
6.0000000e+00
  1.0000000e+00 3.0000000e+06 3.0000000e+00 2.3000000e+06
3.0000000e+00]
 [2.2254247e+11 1.0000000e+00 1.0000000e+00 1.0000000e+00
1.0000000e+00
  2.2000000e+01 2.0000000e+00 3.0000000e+00 2.0000000e+00
4.0000000e+00
  1.0000000e+00 7.5000000e+05 2.0000000e+00 2.3000000e+06
3.0000000e+00]
 [2.2254247e+11 1.0000000e+00 1.0000000e+00 1.0000000e+00
1.0000000e+00
  1.8000000e+01 2.0000000e+00 3.0000000e+00 3.0000000e+00
4.0000000e+00
  1.0000000e+00 0.0000000e+00 1.0000000e+00 1.6000000e+06
2.0000000e+00]
 [2.2054247e+11 1.0000000e+00 1.0000000e+00 1.0000000e+00
1.0000000e+00
  3.0000000e+00 3.0000000e+00 3.0000000e+00 2.0000000e+00
2.0000000e+00
  1.0000000e+00 0.0000000e+00 1.0000000e+00 2.5000000e+06
3.0000000e+00]
 [2.2094202e+11 1.0000000e+00 1.0000000e+00 2.0000000e+00
1.0000000e+00
  1.6000000e+01 2.0000000e+00 5.0000000e+00 5.0000000e+00
6.0000000e+00
  1.0000000e+00 4.0000000e+06 3.0000000e+00 2.8000000e+06
3.0000000e+00]
```

```
 [2.2057201e+11 1.0000000e+00 1.0000000e+00 3.0000000e+00
1.0000000e+00
  2.7000000e+01 1.0000000e+00 4.0000000e+00 1.0000000e+00
1.0000000e+00
  1.0000000e+00 0.0000000e+00 1.0000000e+00 1.3000000e+06
2.0000000e+00]
 [2.2194202e+11 1.0000000e+00 1.0000000e+00 2.0000000e+00
1.0000000e+00
  5.0000000e+01 2.0000000e+00 3.0000000e+00 3.0000000e+00
4.0000000e+00
  1.0000000e+00 0.0000000e+00 1.0000000e+00 1.0000000e+06
2.0000000e+00]
 [2.2254247e+11 1.0000000e+00 1.0000000e+00 1.0000000e+00
1.0000000e+00
  2.8000000e+01 1.0000000e+00 5.0000000e+00 1.2000000e+01
6.0000000e+00
  1.0000000e+00 1.0000000e+06 2.0000000e+00 3.0000000e+06
3.0000000e+00]
 [2.2057201e+11 1.0000000e+00 1.0000000e+00 3.0000000e+00
1.0000000e+00
  2.9000000e+01 1.0000000e+00 4.0000000e+00 3.0000000e+00
4.0000000e+00
  1.0000000e+00 0.0000000e+00 1.0000000e+00 1.0000000e+06
2.0000000e+00]
 [2.2257201e+11 1.0000000e+00 1.0000000e+00 3.0000000e+00
1.0000000e+00
  2.9000000e+01 1.0000000e+00 5.0000000e+00 5.0000000e+00
6.0000000e+00
  1.0000000e+00 0.0000000e+00 1.0000000e+00 2.0000000e+06
2.0000000e+00]
 [2.2154247e+11 1.0000000e+00 1.0000000e+00 1.0000000e+00
1.0000000e+00
  1.7000000e+01 3.0000000e+00 1.0000000e+00 2.0000000e+00
2.0000000e+00
  1.0000000e+00 3.0000000e+06 3.0000000e+00 3.0000000e+06
3.0000000e+00]
 [2.2257201e+11 1.0000000e+00 1.0000000e+00 3.0000000e+00
1.0000000e+00
  1.1000000e+01 2.0000000e+00 9.0000000e+00 5.0000000e+00
6.0000000e+00
  1.0000000e+00 0.0000000e+00 1.0000000e+00 2.3000000e+06
3.0000000e+00]
 [2.2057201e+11 1.0000000e+00 1.0000000e+00 3.0000000e+00
1.0000000e+00
  2.8000000e+01 1.0000000e+00 5.0000000e+00 9.0000000e+00
4.0000000e+00
  1.0000000e+00 0.0000000e+00 1.0000000e+00 2.8000000e+06
3.0000000e+00]
 [2.2294202e+11 1.0000000e+00 1.0000000e+00 2.0000000e+00
1.0000000e+00
  4.1000000e+01 1.0000000e+00 2.0000000e+00 3.0000000e+00
2.0000000e+00
  1.0000000e+00 0.0000000e+00 1.0000000e+00 2.3000000e+06
3.0000000e+00]
 [2.2254247e+11 1.0000000e+00 1.0000000e+00 1.0000000e+00
1.0000000e+00
```

```
  1.9000000e+01 2.0000000e+00 5.0000000e+00 5.0000000e+00
6.0000000e+00
  1.0000000e+00 0.0000000e+00 1.0000000e+00 2.3000000e+06
3.0000000e+00]
 [2.2157201e+11 1.0000000e+00 1.0000000e+00 3.0000000e+00
1.0000000e+00
  1.0000000e+01 2.0000000e+00 5.0000000e+00 2.0000000e+00
5.0000000e+00
  1.0000000e+00 0.0000000e+00 1.0000000e+00 2.0000000e+06
2.0000000e+00]
 [2.2194202e+11 1.0000000e+00 1.0000000e+00 2.0000000e+00
1.0000000e+00
```

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.fit_transform(x_test)
print(x_train)
```

```
[[-0.85407796  0.          0.          ...  0.04731014 -1.43329774
  -0.92228798]
 [ 1.08180389  0.          0.          ...  0.04731014  0.84827825
   0.71267707]
 [-0.85407797  0.          0.          ...  0.04731014 -1.68680618
  -0.92228798]
 ...
 [-0.105861    0.          0.          ...  1.27737374  1.1017867
   0.71267707]
 [-1.29352582  0.          0.          ... -1.18275346 -1.68680618
  -0.92228798]
 [ 1.04672024  0.          0.          ... -1.18275346  0.21450714
   0.71267707]]
```

```
print(y_train)
```

```
[2 1 2 2 2 1 1 2 2 1 1 2 2 2 2 1 1 1 2 2 2 1 1 2 2 1 1 1 2 2 1 2
1 1 2 2
 1 1 2 1 2 1 2 1 1 2 2 2 2 2 1 1 2 2 2 2 2 1 1 2 1 1 2 2 2 2 2 1
1 2 1 2
 2 2 2 2 2 2 2 1 2 2 2 2 2 1 1 1 2 2 2 2 2 1 1 2 2 1 2 1 1 1 1 1 2
1 2 2 1
 2 2 2 1 1 2 2 2 2 1 2 2 1 2 2 2 1 2 2 2 2 2 2 2 1 2 2 2 2 2 1 1 2
2 2 1 1
 1 2 1 2 1 1 2 2 1 1 2 1 2 2 2 1 1 2 2 1 1 1 2 2 2 1 1 2 2 1 1 2 2
1 2 2 1
 2 2 1 1 1 1 2 1 2 1 2 1 2 1 2 1 2 2 2 1 1 1 2 1 2 2 1 2 2 2 2 2 1 2
1 2 1 1
 1 1 2 2 1 1 2 2 2 2 1 1 2 2]
```

```
from sklearn.impute import SimpleImputer

imputer = SimpleImputer(strategy='mean')
x_train = imputer.fit_transform(x_train)
```

Membangun model Naive Bayes dengan GaussianNB

```
from sklearn.naive_bayes import GaussianNB
classifier = GaussianNB()
```

```
classifier.fit(x_train,y_train)
```
GaussianNB
```
GaussianNB()
```
Melakukan prediksi pada data uji

```
y_pred = classifier.predict(x_test)
```
Evaluasi model menggunakan confusion matrix

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
[[ 7  0]
 [ 2 17]]
```
Mengimpor pustaka yang diperlukan

```
from sklearn.naive_bayes import GaussianNB
```
Membuat dan melatih model Naive Bayes

```
classifier = GaussianNB()
classifier.fit(x_train, y_train)
```
GaussianNB
```
GaussianNB()
```
Mendapatkan probabilitas atribut dalam kelas (P(atribut|class))

```
atribut_prob_kondisional = classifier.theta_
```
Mencetak probabilitas atribut kondisional untuk setiap atribut dalam setiap kelas

```
for kelas in range(len(classifier.classes_)):
    print(f'Probabilitas kondisional untuk Kelas
{classifier.classes_[kelas]}:')
    for atribut in
range(len(atribut_prob_kondisional[kelas])):
        prob_atribut =
atribut_prob_kondisional[kelas][atribut]
        print(f'Atribut {atribut}:')
        print(f'Probabilitas Kondisional: {prob_atribut}')
        print()
Probabilitas kondisional untuk Kelas 1:
Atribut 0:
Probabilitas Kondisional: -0.12317651850308532

Atribut 1:
Probabilitas Kondisional: 0.0

Atribut 2:
Probabilitas Kondisional: 0.0
```

```
Atribut 3:
Probabilitas Kondisional: 0.03541602246783174

Atribut 4:
Probabilitas Kondisional: 0.0

Atribut 5:
Probabilitas Kondisional: -0.17307741522129 89

Atribut 6:
Probabilitas Kondisional: 0.10932596436181703

Atribut 7:
Probabilitas Kondisional: 0.13536822121918937

Atribut 8:
Probabilitas Kondisional: 0.0034924649181067788

Atribut 9:
Probabilitas Kondisional: -0.22242724237036393

Atribut 10:
Probabilitas Kondisional: 0.0

Atribut 11:
Probabilitas Kondisional: 0.9192599154627339

Atribut 12:
Probabilitas Kondisional: 0.8801656990923608

Atribut 13:
Probabilitas Kondisional: 0.5987308783768687

Atribut 14:
Probabilitas Kondisional: 0.47424467047004654

Probabilitas kondisional untuk Kelas 2:
Atribut 0:
Probabilitas Kondisional: 0.085688012871688 44

Atribut 1:
Probabilitas Kondisional: 0.0

Atribut 2:
Probabilitas Kondisional: 0.0

Atribut 3:
Probabilitas Kondisional: -0.024637233021100442

Atribut 4:
Probabilitas Kondisional: 0.0

Atribut 5:
Probabilitas Kondisional: 0.12040168015394706

Atribut 6:
```

```
Probabilitas Kondisional: -0.07605284477343797

Atribut 7:
Probabilitas Kondisional: -0.09416919736987074

Atribut 8:
Probabilitas Kondisional: -0.002429540812596002

Atribut 9:
Probabilitas Kondisional: 0.1547319946924266

Atribut 10:
Probabilitas Kondisional: 0.0

Atribut 11:
Probabilitas Kondisional: -0.6394851585827706

Atribut 12:
Probabilitas Kondisional: -0.6122891819772922

Atribut 13:
Probabilitas Kondisional: -0.4165084371317351

Atribut 14:
Probabilitas Kondisional: -0.3299093359791643
```

Mencetak atribut yang memiliki probabilitas kondisional tertinggi untuk setiap kelas

```
for kelas in range(len(classifier.classes_)):
    atribut_terpengaruh =
np.argmax(atribut_prob_kondisional[kelas])
    prob_terpengaruh = np.max(atribut_prob_kondisional[kelas])
    print(f'Atribut paling berpengaruh untuk Kelas
{classifier.classes_[kelas]} adalah Atribut
{atribut_terpengaruh} dengan Probabilitas Kondisional:
{prob_terpengaruh}')
```

```
Atribut paling berpengaruh untuk Kelas 1 adalah Atribut 11 dengan
Probabilitas Kondisional: 0.9192599154627339
Atribut paling berpengaruh untuk Kelas 2 adalah Atribut 9 dengan
Probabilitas Kondisional: 0.1547319946924266
```

Percobaan 9

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection import cross_val_score, KFold
from sklearn.preprocessing import StandardScaler
from sklearn.impute import SimpleImputer
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import confusion_matrix, precision_score,
recall_score
```

Membaca dataset

```
dataset = pd.read_csv('/content/drive/MyDrive/5.-Data-Finis-
Preposesing.csv')
x = dataset.iloc[:, :-1].values
y = dataset.iloc[:, -1].values
```

Preprocessing

```
sc = StandardScaler()
imputer = SimpleImputer(strategy='mean')
x = sc.fit_transform(x)
x = imputer.fit_transform(x)
```

Inisialisasi model Naive Bayes dan jumlah subset (fold) untuk k-fold cross-validation

```
classifier = GaussianNB()
k = 5  # Jumlah subset (fold)
kf = KFold(n_splits=k, shuffle=True, random_state=0)
```

Inisialisasi list untuk menyimpan nilai akurasi, presisi, dan recall dari setiap iterasi

```
accuracies = []
precisions = []
recalls = []
```

Melakukan k-fold cross-validation

```
for train_index, test_index in kf.split(x):
    x_train, x_test = x[train_index], x[test_index]
    y_train, y_test = y[train_index], y[test_index]

    # Melatih model pada subset train
    classifier.fit(x_train, y_train)

    # Prediksi menggunakan subset test
```

```
    y_pred = classifier.predict(x_test)

    # Menghitung confusion matrix pada setiap iterasi
    cm = confusion_matrix(y_test, y_pred)

    # Menghitung dan menyimpan akurasi, presisi, dan recall
pada setiap iterasi
    accuracy = np.mean(y_pred == y_test)
    precision = precision_score(y_test, y_pred)
    recall = recall_score(y_test, y_pred)

    accuracies.append(accuracy)
    precisions.append(precision)
    recalls.append(recall)
```

Menghitung rata-rata akurasi, presisi, dan recall dari semua iterasi

```
mean_accuracy = np.mean(accuracies)
mean_precision = np.mean(precisions)
mean_recall = np.mean(recalls)

print(f"Mean Accuracy: {mean_accuracy}")
print(f"Mean Precision: {mean_precision}")
print(f"Mean Recall: {mean_recall}")
Mean Accuracy: 0.8653846153846153
Mean Precision: 0.8443840579710145
Mean Recall: 0.8355909536687569
```

Dataset

NIM	Status	Perguruan Tinggi	Program Studi	Jenjang	Alamat Kelurahan/Desa	Alamat Kecamatan	Pendidikan Wali	Pekerjaan Wali	Penghasilan Wali	Keterangan	Jumlah Piutang UKT Mahasiswa	Umur Piutang UKT Mahasiswa	Jumlah Piutang BPP	Umur Piutang BPP	Status Piutang
2.21E+11	1	1	1	1	26	1	3	3	2	1	1700000	3	2000000	2	2
2.21E+11	1	1	1	1	44	1	3	2	4	1	0	1	0	1	2
2.21E+11	1	1	1	1	22	2	5	2	5	1	100000	2	2300000	3	2
2.21E+11	1	1	1	1	28	1	2	11	2	1	3600000	3	2800000	3	1
2.21E+11	1	1	1	1	20	2	5	2	4	1	1000000	2	2000000	2	2
2.21E+11	1	1	1	1	21	2	9	9	5	1	0	1	0	1	2
2.21E+11	1	1	1	1	21	2	5	2	4	1	0	1	0	1	2
2.21E+11	1	1	1	1	19	2	4	2	4	1	0	1	1750000	2	2
2.21E+11	1	1	1	1	3	3	3	2	2	1	0	1	2500000	3	2
2.21E+11	1	1	1	1	20	2	5	2	2	1	1000000	2	2000000	2	2
2.21E+11	1	1	1	1	20	2	5	2	2	1	200000	2	2300000	3	2
2.21E+11	1	1	1	1	30	3	3	2	4	1	500000	2	1000000	2	2
2.21E+11	1	1	1	1	28	1	2	18	4	1	1000000	2	0	1	2
2.21E+11	1	1	1	1	9	2	3	3	4	1	400000	2	1000000	2	2
2.21E+11	1	1	3	1	24	2	3	2	2	1	3500000	3	500000	2	1
2.21E+11	1	1	3	1	8	1	4	3	2	1	2000000	3	1800000	2	1
2.21E+11	1	1	3	1	16	2	3	2	2	1	0	1	0	1	2

2.21E+11	1	1	3	1	22	2	2	1	1	1	0	1	1100000	2	2
2.21E+11	1	1	3	1	44	1	5	9	4	1	0	1	1000000	2	2
2.21E+11	1	1	3	1	28	1	5	9	4	1	0	1	1000000	2	2
2.21E+11	1	1	3	1	33	1	5	1	1	1	4000000	3	2300000	3	1
2.21E+11	1	1	3	1	37	5	3	2	2	1	3000000	3	2800000	3	1
2.21E+11	1	1	3	1	21	2	3	2	2	1	4000000	3	2800000	3	1
2.21E+11	1	1	3	1	42	1	5	5	6	1	0	1	700000	2	2
2.21E+11	1	1	3	1	20	1	3	3	2	1	2000000	3	1800000	2	2
2.21E+11	1	1	3	1	8	1	5	9	2	1	4000000	3	2000000	2	1
2.21E+11	1	1	3	1	42	1	5	6	4	1	4000000	3	2800000	3	1
2.21E+11	1	1	3	1	10	2	4	12	5	1	2500000	3	2300000	3	1
2.21E+11	1	1	3	1	50	2	5	5	6	1	4000000	3	2300000	3	1
2.21E+11	1	1	3	1	22	2	10	5	6	1	0	1	0	1	2
2.21E+11	1	1	3	1	18	2	5	2	2	1	4000000	3	2800000	3	1
2.21E+11	1	1	3	1	8	1	5	9	5	1	2000000	3	1800000	2	2
2.21E+11	1	1	3	1	44	1	3	3	2	1	0	1	800000	2	2
2.21E+11	1	1	3	1	10	2	3	1	1	1	0	1	0	1	2
2.21E+11	1	1	3	1	32	1	2	9	5	1	2000000	3	2300000	3	1
2.21E+11	1	1	3	1	44	1	5	5	6	1	0	1	0	1	2

2.21E+11	1	1	3	1	8	1	9	5	6	1	2500000	3	2800000	3	1
2.21E+11	1	1	3	1	27	1	4	1	1	1	0	1	1300000	2	2
2.21E+11	1	1	3	1	28	1	5	1	1	1	4000000	3	2800000	3	1
2.21E+11	1	1	3	1	27	1	2	3	2	1	0	1	1000000	2	2
2.21E+11	1	1	3	1	11	2	5	9	4	1	0	1	1000000	2	2
2.21E+11	1	1	3	1	24	2	3	2	4	1	700000	2	2500000	3	2
2.21E+11	1	1	3	1	8	1	5	9	5	1	0	1	2000000	2	2
2.21E+11	1	1	3	1	24	2	5	1	1	1	3000000	3	2900000	3	1
2.21E+11	1	1	3	1	24	2	2	3	2	1	4000000	3	3000000	3	1
2.21E+11	1	1	3	1	28	1	5	9	4	1	0	1	2800000	3	2
2.21E+11	1	1	3	1	25	3	5	5	6	1	500000	2	1500000	2	1
2.21E+11	1	1	3	1	44	1	5	17	4	1	0	1	800000	2	2
2.21E+11	1	1	3	1	12	2	4	3	4	1	4000000	3	3000000	3	1
2.21E+11	1	1	3	1	28	1	5	5	6	1	0	1	1000000	2	2
2.21E+11	1	1	3	1	28	1	5	5	6	1	0	1	1000000	2	2
2.21E+11	1	1	3	1	40	9	5	9	6	1	0	1	1000000	2	2
2.21E+11	1	1	3	1	1	1	3	2	4	1	700000	2	1900000	2	2
2.21E+11	1	1	3	1	16	3	5	5	6	1	1000000	2	1800000	2	1
2.21E+11	1	1	3	1	32	1	5	4	4	1	500000	2	2800000	3	1

2.21E+11	1	1	3	1	29	1	4	3	4	1	0	1	1000000	2	2
2.21E+11	1	1	3	1	24	2	5	2	4	1	0	1	0	1	2
2.21E+11	1	1	3	1	11	2	5	1	1	1	0	1	0	1	2
2.21E+11	1	1	3	1	44	1	5	1	1	1	0	1	0	1	2
2.21E+11	1	1	3	1	43	1	1	3	2	1	2000000	3	2300000	3	1
2.21E+11	1	1	3	1	16	2	5	3	2	1	300000	2	0	1	2
2.21E+11	1	1	3	1	47	1	5	5	6	1	5000000	3	3000000	3	1
2.21E+11	1	1	3	1	16	2	5	5	6	1	5000000	3	3000000	3	1
2.21E+11	1	1	2	1	20	2	5	5	6	1	0	1	2300000	3	2
2.21E+11	1	1	2	1	16	2	5	3	4	1	0	1	1800000	2	2
2.21E+11	1	1	2	1	24	2	5	9	5	1	4600000	3	2800000	3	1
2.21E+11	1	1	2	1	32	1	2	2	2	1	2300000	3	2000000	2	1
2.21E+11	1	1	2	1	16	2	5	5	6	1	0	1	2800000	3	2
2.21E+11	1	1	2	1	44	1	5	5	6	1	0	1	1000000	2	2
2.21E+11	1	1	2	1	28	1	5	5	6	1	0	1	1000000	2	2
2.21E+11	1	1	2	1	16	2	5	5	6	1	0	1	0	1	2
2.21E+11	1	1	2	1	16	2	5	5	6	1	4000000	3	2800000	3	1
2.21E+11	1	1	2	1	27	1	5	5	6	1	2500000	3	2000000	2	1
2.21E+11	1	1	2	1	26	1	5	5	6	1	500000	2	800000	2	2

2.21E+11	1	1	2	1	20	2	9	10	2	1	4000000	3	2800000	3	1
2.21E+11	1	1	2	1	19	2	5	2	2	1	1000000	2	2500000	3	1
2.21E+11	1	1	2	1	36	4	5	9	6	1	3000000	3	2300000	3	1
2.21E+11	1	1	2	1	24	2	10	12	8	1	1700000	3	2000000	2	1
2.21E+11	1	1	2	1	11	2	5	10	2	1	0	1	2000000	2	2
2.21E+11	1	1	2	1	47	1	5	5	6	1	450000	2	1800000	2	1
2.21E+11	1	1	2	1	50	2	5	7	4	1	0	1	0	1	2
2.21E+11	1	1	2	1	24	2	5	5	6	1	4000000	3	3000000	3	1
2.21E+11	1	1	2	1	50	2	3	3	4	1	0	1	1500000	2	2
2.21E+11	1	1	2	1	18	2	5	5	6	1	1000000	2	1000000	2	2
2.21E+11	1	1	2	1	18	2	5	9	8	1	3000000	3	2300000	3	1
2.21E+11	1	1	2	1	18	2	5	9	8	1	2000000	3	1000000	2	2
2.21E+11	1	1	2	1	6	5	5		1	1	4000000	3	3000000	3	1
2.21E+11	1	1	2	1	32	1	5	9	5	1	5000000	3	3000000	3	1
2.22E+11	1	1	1	1	44	1	5	9	5	1	0	1	2200000	3	2
2.22E+11	1	1	1	1	21	2	2	2	5	1	800000	2	2300000	3	2
2.22E+11	1	1	1	1	21	2	3	2	2	1	3000000	3	3000000	3	1
2.22E+11	1	1	1	1	20	3	4	9	2	1	3000000	3	3000000	3	1
2.22E+11	1	1	1	1	19	2	1	2	2	1	3000000	3	2800000	3	1

2.22E+11	1	1	1	1	19	2	3	2	6	1	500000	2	2000000	2	2
2.22E+11	1	1	1	1	19	2	4	2	2	1	2500000	3	2300000	3	1
2.22E+11	1	1	1	1	22	2	5	9	6	1	1000000	2	2600000	3	2
2.22E+11	1	1	1	1	24	2	4	2	4	1	600000	2	2000000	2	2
2.22E+11	1	1	1	1	25	1	5	9	8	1	0	1	2000000	2	2
2.22E+11	1	1	1	1	24	2	5	2	5	1	400000	2	2400000	3	2
2.22E+11	1	1	1	1	3	3	2	2	2	1	3000000	3	3000000	3	1
2.22E+11	1	1	1	1	3	3	5	2	2	1	3000000	3	3000000	3	1
2.22E+11	1	1	1	1	3	3	3	2	4	1	3000000	3	3000000	3	1
2.22E+11	1	1	1	1	3	3	4	2	2	1	3000000	3	3000000	3	1
2.22E+11	1	1	1	1	17	3	4	2	2	1	3000000	3	2800000	3	1
2.22E+11	1	1	1	1	3	3	3	2	2	1	3000000	3	2800000	3	1
2.22E+11	1	1	1	1	3	3	2	2	2	1	3000000	3	2800000	3	1
2.22E+11	1	1	1	1	24	2	3	1	4	1	100000	2	2300000	3	2
2.22E+11	1	1	1	1	16	2	5	1	2	1	3000000	3	2800000	3	1
2.22E+11	1	1	1	1	17	3	3	2	2	1	3000000	3	2800000	3	1
2.22E+11	1	1	1	1	17	3	5	10	2	1	3000000	3	2800000	3	1
2.22E+11	1	1	1	1	17	3	1	2	2	1	3000000	3	3000000	3	1
2.22E+11	1	1	1	1	17	3	5	2	2	1	3000000	3	3000000	3	1

2.22E+11	1	1	1	1	17	3	5	2	2	1	3000000	3	2800000	3	1
2.22E+11	1	1	3	1	16	2	3	2	5	1	900000	2	2300000	3	2
2.22E+11	1	1	3	1	36	4	3	3	5	1	0	1	2300000	3	2
2.22E+11	1	1	3	1	44	1	4	2	5	1	0	1	1000000	2	2
2.22E+11	1	1	3	1	43	1	5	9	4	1	3600000	3	2800000	3	1
2.22E+11	1	1	3	1	14	1	9	9	4	1	3600000	3	3000000	3	1
2.22E+11	1	1	3	1	44	1	3	3	2	1	3600000	3	3000000	3	1
2.22E+11	1	1	3	1	28	1	3	3	2	1	3600000	3	3000000	3	1
2.22E+11	1	1	3	1	8	1	5	9	5	1	2400000	3	2000000	2	1
2.22E+11	1	1	3	1	24	2	3	17	2	1	3500000	3	2100000	3	1
2.22E+11	1	1	3	1	19	2	4	9	2	1	3600000	3	3000000	3	1
2.22E+11	1	1	3	1	44	1	5	3	2	1	3600000	3	3000000	3	1
2.22E+11	1	1	3	1	24	2	5	5	0	1	3000000	3	2300000	3	1
2.22E+11	1	1	3	1	43	1	4	2	2	1	3300000	3	2350000	3	1
2.22E+11	1	1	3	1	47	1	3	3	2	1	0	1	2000000	2	2
2.22E+11	1	1	3	1	4	2	4	3	2	1	800000	2	2000000	2	2
2.22E+11	1	1	3	1	43	1	4	2	4	1	900000	2	2000000	2	2
2.22E+11	1	1	3	1	18	2	5	3	2	1	1100000	2	2000000	2	2
2.22E+11	1	1	3	1	35	4	5	1	1	1	0	1	2300000	3	1

2.22E+11	1	1	3	1	10	2	3	3	4	1	900000	2	2300000	3	2
2.22E+11	1	1	3	1	22	2	5	9	5	1	3600000	3	3000000	3	1
2.22E+11	1	1	3	1	24	2	4	9	4	1	3300000	3	2600000	3	1
2.22E+11	1	1	3	1	43	1	4	3	4	1	1000000	2	2300000	3	2
2.22E+11	1	1	3	1	32	1	2	9	5	1	1000000	2	2000000	2	2
2.22E+11	1	1	3	1	10	2	3	2	2	1	2100000	3	2100000	3	1
2.22E+11	1	1	3	1	10	2	5	2	5	1	0	1	2000000	2	2
2.22E+11	1	1	3	1	11	2	4	3	4	1	1000000	2	2300000	3	2
2.22E+11	1	1	3	1	43	1	5	12	6	1	0	1	2300000	3	2
2.22E+11	1	1	3	1	5	9	3	3	2	1	2200000	3	2000000	2	1
2.22E+11	1	1	3	1	32	1	1	2	4	1	0	1	2000000	2	2
2.22E+11	1	1	3	1	47	1	9	5	6	1	3600000	3	2800000	3	1
2.22E+11	1	1	3	1	16	2	2	3	4	1	600000	2	2300000	3	2
2.22E+11	1	1	3	1	19	2	4	2	4	1	2000000	3	3000000	3	1
2.22E+11	1	1	3	1	44	1	5	9	5	1	0	1	2000000	2	2
2.22E+11	1	1	3	1	16	2	4	2	4	1	1000000	2	2800000	3	2
2.22E+11	1	1	3	1	28	1	5	5	6	1	2400000	3	2300000	3	1
2.22E+11	1	1	3	1	11	2	3	3	6	1	0	1	0	1	2
2.22E+11	1	1	3	1	9	2	5	5	6	1	0	1	1900000	2	2

2.22E+11	1	1	2	1	7	2	1	3	2	1	3000000	3	2300000	3	1
2.22E+11	1	1	2	1	16	2	3	3	2	1	3000000	3	2800000	3	1
2.22E+11	1	1	2	1	50	2	3	3	4	1	0	1	1000000	2	2
2.22E+11	1	1	2	1	7	2	3	3	4	1	800000	2	2000000	2	2
2.22E+11	1	1	2	1	15	2	5	5	6	1	1000000	2	2300000	3	2
2.22E+11	1	1	2	1	44	1	5	9	4	1	1000000	3	2800000	3	1
2.22E+11	1	1	2	1	7	2	5	5	6	1	0	1	2300000	3	2
2.22E+11	1	1	2	1	10	2	5	9	9	1	0	1	2000000	2	2
2.22E+11	1	1	2	1	44	1	5	9	4	1	2000000	3	2800000	3	2
2.22E+11	1	1	2	1	15	1	5	9	4	1	3000000	3	3000000	3	1
2.22E+11	1	1	2	1	11	2	5	9	4	1	3000000	3	3000000	3	1
2.22E+11	1	1	2	1	11	2	8	5	6	1	3000000	3	3000000	3	1
2.22E+11	1	1	2	1	44	1	9	5	6	1	2000000	3	2000000	2	2
2.22E+11	1	1	2	1	44	1	5	9	5	1	3000000	3	3000000	3	1
2.22E+11	1	1	2	1	44	1	9	5	6	1	500000	2	2300000	3	2
2.22E+11	1	1	2	1	28	1	5	5	6	1	500000	2	2300000	3	2
2.22E+11	1	1	2	1	28	1	5	5	6	1	500000	2	2300000	3	2
2.22E+11	1	1	2	1	28	1	5	5	6	1	3000000	3	3000000	3	1
2.23E+11	1	1	1	1	18	2	3	3	4	1	0	1	2300000	3	2

2.23E+11	1	1	1	1	21	2	5	2	4	1	600000	2	2500000	3	1
2.23E+11	1	1	1	1	18	2	3	3	4	1	0	1	1600000	2	2
2.23E+11	1	1	1	1	18	2	3	3	4	1	500000	2	2300000	3	2
2.23E+11	1	1	1	1	20	2	5	6	4	1	0	1	2300000	3	2
2.23E+11	1	1	1	1	22	2	3	2	4	1	750000	2	2300000	3	2
2.23E+11	1	1	1	1	18	2	3	3	4	1	0	1	2300000	3	2
2.23E+11	1	1	1	1	22	2	4	2	4	1	0	1	2300000	3	2
2.23E+11	1	1	1	1	19	2	2	2	2	1	1000000	2	3000000	3	2
2.23E+11	1	1	1	1	22	2	3	2	5	1	650000	2	2300000	3	1
2.23E+11	1	1	1	1	28	1	5	12	6	1	1000000	2	3000000	3	2
2.23E+11	1	1	1	1	21	2	3	9	4	1	500000	2	2300000	3	2
2.23E+11	1	1	1	1	22	2	3	2	4	1	1000000	2	2800000	3	1
2.23E+11	1	1	1	1	18	2	3	2	3	1	0	1	2300000	3	2
2.23E+11	1	1	1	1	35	4	5	9	5	1	1000000	2	2800000	3	2
2.23E+11	1	1	1	1	31	2	3	2	5	1	0	1	2300000	3	2
2.23E+11	1	1	1	1	44	1	3	18	5	1	750000	2	2300000	3	1
2.23E+11	1	1	1	1	20	2	2	2	4	1	0	1	2300000	3	2
2.23E+11	1	1	1	1	20	3	5	2	5	1	1000000	2	2800000	3	2
2.23E+11	1	1	1	1	9	2	4	19	5	1	550000	2	2300000	3	2

2.23E+11	1	1	1	1	39	9	3	2	4	1	1000000	2	2800000	3	1
2.23E+11	1	1	3	1	22	2	3	7	5	1	0	1	2300000	3	2
2.23E+11	1	1	1	1	22	2	5	2	5	1	0	1	1600000	2	2
2.23E+11	1	1	1	1	14	1	2	3	4	1	0	1	2300000	3	2
2.23E+11	1	1	1	1	45	1	3	5	4	1	0	1	2000000	2	2
2.23E+11	1	1	1	1	19	2	4	2	2	1	0	1	2300000	3	2
2.23E+11	1	1	1	1	19	2	5	5	6	1	0	1	2300000	3	2
2.23E+11	1	1	3	1	9	2	5	7	5	1	0	1	2300000	3	2
2.23E+11	1	1	3	1	43	1	4	3	4	1	480000	2	2300000	3	2
2.23E+11	1	1	3	1	38	1	3	3	2	1	0	1	2300000	3	2
2.23E+11	1	1	3	1	11	2	3	3	4	1	0	1	1800000	2	2
2.23E+11	1	1	3	1	18	2	3	3	4	1	0	1	2200000	3	2
2.23E+11	1	1	3	1	50	2	3	3	4	1	0	1	2300000	3	2
2.23E+11	1	1	3	1	50	2	2	3	4	1	0	1	2300000	3	2
2.23E+11	1	1	3	1	38	1	3	9	4	1	0	1	2000000	2	2
2.23E+11	1	1	3	1	27	1	5	5	6	1	0	1	2000000	2	2
2.23E+11	1	1	3	1	11	2	9	5	6	1	0	1	2300000	3	2
2.23E+11	1	1	3	1	45	1	3	1	1	1	0	1	1800000	2	2
2.23E+11	1	1	3	1	45	1	3	4	5	1	0	1	2000000	2	2

2.23E+11	1	1	3	1	43	1	5	3	4	1	1200000	2	2300000	3	2
2.23E+11	1	1	3	1	14	1	3	2	4	1	800000	2	2000000	2	1
2.23E+11	1	1	3	1	49	1	5	3	4	1	0	1	2000000	2	2
2.23E+11	1	1	3	1	28	1	5	1	1	1	0	1	1800000	2	2
2.23E+11	1	1	3	1	15	2	3	3	4	1	500000	2	2300000	3	2
2.23E+11	1	1	3	1	24	2	3	1	1	1	1200000	2	2800000	3	1
2.23E+11	1	1	3	1	27	1	3	3	4	1	1200000	2	2300000	3	1
2.23E+11	1	1	3	1	20	2	2	2	2	1	1200000	2	2850000	3	1
2.23E+11	1	1	3	1	16	2	4	3	4	1	0	1	2000000	2	2
2.23E+11	1	1	3	1	44	1	5	6	4	1	1200000	2	3000000	3	1
2.23E+11	1	1	3	1	43	1	3	3	4	1	1200000	2	2800000	3	2
2.23E+11	1	1	3	1	27	1	3	9	4	1	0	1	2000000	2	2
2.23E+11	1	1	3	1	20	2	3	1	1	1	1200000	2	2800000	3	1
2.23E+11	1	1	3	1	28	1	5	6	5	1	0	1	2300000	3	2
2.23E+11	1	1	3	1	44	1	3	3	4	1	0	1	2000000	2	2
2.23E+11	1	1	3	1	29	1	5	5	6	1	0	1	2000000	2	2
2.23E+11	1	1	3	1	20	2	5	9	4	1	1200000	2	2100000	3	1
2.23E+11	1	1	3	1	16	2	4	9	5	1	800000	2	2300000	3	2
2.23E+11	1	1	3	1	49	1	5	5	6	1	600000	2	2000000	2	1

2.23E+11	1	1	3	1	37	5	4	9	5	1	0	1	2000000	2	2
2.23E+11	1	1	3	1	9	2	3	9	5	1	1200000	2	2800000	3	1
2.23E+11	1	1	3	1	50	2	3	3	2	1	0	1	800000	2	2
2.23E+11	1	1	3	1	11	2	5	9	4	1	1200000	2	3000000	3	1
2.23E+11	1	1	3	1	15	2	5	12	8	1	0	1	2300000	3	2
2.23E+11	1	1	3	1	50	2	3	3	2	1	0	1	2300000	3	2
2.23E+11	1	1	3	1	24	2	3	2	5	1	0	1	2300000	3	2
2.23E+11	1	1	3	1	28	1	5	9	5	1	300000	2	2000000	2	2
2.23E+11	1	1	2	1	14	1	4	9	5	1	700000	2	2300000	3	2
2.23E+11	1	1	2	1	41	1	2	3	2	1	0	1	2300000	3	2
2.23E+11	1	1	2	1	16	2	9	6	5	1	1000000	2	2800000	3	1
2.23E+11	1	1	2	1	16	2	3	3	4	1	1000000	2	2300000	3	1
2.23E+11	1	1	2	1	23	2	4	2	5	1	0	1	2300000	3	2
2.23E+11	1	1	2	1	18	2	4	1	1	1	0	1	2300000	3	2
2.23E+11	1	1	2	1	44	1	5	5	6	1	0	1	2000000	2	2
2.23E+11	1	1	2	1	13	5	3	1	1	1	1000000	2	2800000	3	2
2.23E+11	1	1	2	1	23	2	4	1	1	1	0	1	2800000	3	1
2.23E+11	1	1	2	1	47	1	5	5	6	1	500000	2	2500000	3	2
2.23E+11	1	1	2	1	48	1	3	2	2	1	700000	2	3000000	3	2

2.23E+11	1	1	2	1	19	2	5	2	4	1	1000000	2	2800000	3	1
2.23E+11	1	1	2	1	19	2	5	2	4	1	1000000	2	2300000	3	1
2.23E+11	1	1	2	1	19	2	9	9	5	1	1000000	2	2800000	3	1
2.23E+11	1	1	2	1	34	9	5	9	0	1	0	1	2300000	3	2
2.23E+11	1	1	2	1	19	2	9	5	8	1	1000000	2	2300000	3	2
2.23E+11	1	1	2	1	16	2	3	3	4	1	0	1	2300000	3	2
2.23E+11	1	1	2	1	21	2	1	2	2	1	500000	2	2300000	3	2
2.23E+11	1	1	2	1	7	2	3	3	4	1	0	1	1600000	2	2
2.23E+11	1	1	2	1	2	9	3	3	2	1	0	1	2300000	3	2
2.23E+11	1	1	2	1	44	1	3	18	5	1	1000000	2	2000000	2	2
2.23E+11	1	1	2	1	44	1	9	5	6	1	1000000	2	3000000	3	1
2.23E+11	1	1	2	1	28	1	5	9	5	1	1000000	2	2000000	2	1
2.23E+11	1	1	2	1	28	1	5	9	4	1	500000	2	2300000	3	2
2.23E+11	1	1	2	1	16	2	5	16	2	1	1000000	2	3000000	3	2
2.23E+11	1	1	2	1	16	2	5	9	8	1	1000000	2	2300000	3	1