Pemodelan *Machine Learning* untuk Memprediksi Kelulusan Mata Kuliah dengan Metode *Random Forest*

Skripsi

Disusun untuk memenuhi salah satu syarat memperoleh gelar Sarjana Komputer

FARID WUJDI MUBAROK
1313617006

PROGRAM STUDI ILMU KOMPUTER
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS NEGERI JAKARTA
2023

LEMBAR PERSETUJUAN HASIL SIDANG SKRIPSI

Pemodelan *Machine Learning* untuk Memprediksi Kelulusan Mata Kuliah dengan Metode *Random Forest*

Nama : Farid Wujdi Mubarok

No.Registrasi : 1313617006

	Nama	Tanda Tangan	Tanggal
Penanggung Jawab			
Dekan	: Prof. Dr. Muktiningsih N, M.Si.		
	NIP. 196405111989032001	.....	.....
Wakil Penanggung Jawab			
Wakil Dekan I	: Dr. Esmar Budi, S.Si., MT.		
	NIP. 197207281999031002	.....	.....
Ketua	: Ir. Fariani Hermin Indiyah, MT.		
	NIP.196002111987032001	.....	14/2/2023
Sekretaris	: Muhammad Eka Suryana, M.Kom.		
	NIP.198512232012121002	.....	16/2/2023
Penguji	: Drs. Mulyono, M.Kom.		
	NIP.196605171994031003	.....	14/2/2023
Pembimbing I	: Dr. Ria Arafiyah, M.Si.		
	NIP.197511212005012004	.....	16/2/2023
Pembimbing II	: Med Irzal, M.Kom.		
	NIP.197706152003121001	.....	16/2/2023

Dinyatakan lulus ujian skripsi tanggal: 10 Februari 2023

LEMBAR PERNYATAAN

Saya menyatakan dengan sesungguhnya bahwa skripsi dengan judul "**Pemodelan *Machine Learning* untuk Memprediksi Kelulusan Mata Kuliah dengan Metode *Random Forest***" yang disusun sebagai syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Ilmu Komputer Universitas Negeri Jakarta adalah karya ilmiah saya dengan arahan dari dosen pembimbing.

Sumber informasi yang diperoleh dari penulis lain yang telah dipublikasikan yang disebutkan dalam teks skripsi ini, telah dicantumkan dalam Daftar Pustaka sesuai dengan norma, kaidah dan etika penulisan ilmiah.

Jika dikemudian hari ditemukan sebagian besar skripsi ini bukan hasil karya saya sendiri dalam bagian-bagian tertentu, saya bersedia menerima sanksi pencabutan gelar akademik yang saya sanding dan sanksi-sanksi lainnya sesuai dengan peraturan perundang-undangan yang berlaku.

Jakarta, 27 Januari 2023

Farid Wujdi Mubarok

LEMBAR PERSEMBAHAN

Dipersembahkan untuk orang-orang yang kusayangi

Untuk Papa dan Mama

Untuk semua Kakakku

Dan untuk diriku sendiri

KATA PENGANTAR

Alhamdulillahirabbil'alamiin, segala puji bagi Allah *Subhanahu wa ta'ala* atas berkat Rahmat, Hidayah dan Karunia-Nya yang telah memberikan segala bentuk pertolongan pada penyusunan skripsi sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer di Program Studi Ilmu Komputer Universitas Negeri Jakarta dengan judul "Pemodelan *Machine Learning* untuk Memprediksi Kelulusan Mata Kuliah dengan Metode *Random Forest*".

Selama penulis melakukan penelitian ini terdapat banyak rintangan dan tantangan yang dihadapi. Namun dengan adanya pihak-pihak yang telah membantu penulis untuk menghadapi hambatan-hambatan yang ada, hal tersebut menjadi faktor pendorong penulis untuk tidak menyerah. Pada kesempatan ini, penulis ingin mengucapkan terima kasih yang sebesar-besarnya kepada semua pihak yang telah membantu penulis dalam penyelesaian penelitian ini, khususnya kepada:

1. Ibu Ir. Fariani Hermin, M.T., selaku Koordinator Program Studi Ilmu Komputer, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Negeri Jakarta.
2. Ibu Ria Arafiyah, M.Si., selaku dosen pembimbing yang telah membimbing, mengarahkan, serta memberikan saran yang baik untuk penulis dalam menyelesaikan skripsi ini.
3. Bapak Med Irzal, M.Kom., selaku dosen pembimbing yang telah membimbing, mengarahkan, serta memberikan saran yang baik untuk penulis dalam menyelesaikan skripsi ini.
4. Ibu Dra. Ellis Salsabila, M.Si., selaku dosen Program Studi Pendidikan Matematika FMIPA UNJ yang telah mengizinkan penulis untuk mengolah data nilai mahasiswa.
5. Seluruh Bapak-Ibu Dosen Program Studi Ilmu komputer FMIPA UNJ yang telah memberikan ilmu serta bimbingan selama penulis berkuliah di Ilmu Komputer FMIPA UNJ.
6. Terima kasih untuk Dyan, Okta, Hafiz, Rara, Nabila, Ka Deby dan Mas Harry yang telah men-*support* dan mendengarkan keluh kesah penulis selama menyusun skripsi ini.

7. Dan Terima kasih kepada teman-teman Mahasiswa Ilmu Komputer FMIPA UNJ angkatan 2017 atas dukungan dan kebersamaannya selama kurang lebih 5 tahun terakhir.

Semoga segala bantuan yang telah diberikan kepada penulis mendapat balasan dari Allah *Subhanahu wa ta'ala*. Penulis menyadari masih banyaknya kekurangan yang ada pada skripsi ini, baik dari materi, tata bahasa, maupun cara penulisan. Oleh karena itu penulis sangat terbuka atas kritik dan saran untuk evaluasi bagi penulis kedepannya. Akhir kata penulis berharap agar skripsi ini dapat berguna dan bermanfaat bagi para pembaca. Atas perhatiannya penulis mengucapkan terima kasih.

Jakarta, 27 Januari 2023

Farid Wujdi Mubarok

ABSTRAK

FARID WUJDI MUBAROK. Pemodelan *Machine Learning* untuk Memprediksi Kelulusan Mata Kuliah dengan Metode *Random Forest*. Skripsi, Program Studi Ilmu Komputer, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Negeri Jakarta. Januari 2023.

Selama menjalani perkuliahan, terdapat mahasiswa yang gagal dalam suatu mata kuliah sehingga diharuskan mengulang mata kuliah tersebut pada semester atau tahun berikutnya. Fenomena ini menyebabkan masa studi mahasiswa berpeluang lebih lama, yang dapat berujung putus kuliah atau *drop out*. Untuk mengatasi masalah tersebut, maka dikembangkan suatu pemodelan *machine learning* untuk mengklasifikasikan kelulusan mata kuliah. Penelitian ini bertujuan untuk membuat model *machine learning* dengan algoritma *random forest* dalam mengklasifikasikan kelulusan mata kuliah mahasiswa serta menguji validitas dari model yang telah dibuat. *Datasets* yang digunakan dalam melakukan penelitian ini bersumber dari data nilai mahasiswa dan data absensi mata kuliah kalkulus integral serta data publik pada PDDIKTI. Sebelum dilakukan pemodelan, *datasets* terlebih dahulu diolah, sehingga diperoleh *datasets* yang bersih. Model *random forest* terbaik diperoleh dengan melakukan pencarian nilai OOB dengan *error* terkecil dari berbagai kombinasi jumlah pohon dan jumlah fitur maksimal yang digunakan dalam membangun model. Model *random forest* terbaik menghasilkan akurasi sebesar 82%. Pengujian validitas model *random forest* menggunakan *k-fold cross validation* dengan k sebanyak 10. Hasil uji validitas dari model terbaik mendapatkan hasil rata-rata akurasi sebesar 90% dan standar deviasi sebesar 0.07.

Kata kunci. *K-fold cross validation, Machine Learning, Random Forest*

ABSTRACT

FARID WUJDI MUBAROK. Machine Learning Modeling for Predicting Course Graduation with the Random Forest Method. Thesis, Computer Science Study Program, Faculty of Mathematics and Natural Science, State University of Jakarta. January 2023.

During the study, there are students who fail a course so that they are required to repeat the course in the following semester or year. This phenomenon causes the student's study period to be longer, which can lead to dropping out of college. To overcome this problem, a machine learning model was developed to classify course graduation. This research aims to create a machine learning model with random forest algorithm in classifying student course graduation and test the validity of the model that has been created. Datasets used in conducting this research are sourced from student grade data and attendance data for integral calculus courses and also public data on PDDIKTI. Before modeling, the datasets are first processed so that clean datasets are obtained. The best random forest model is obtained by finding the OOB value with the smallest error from various combinations of the number of trees and the maximum number of features used in building the model. The best random forest model produces an accuracy of 82%. The validity test of the random forest model uses k-fold cross validation with k as much as 10. The validity test results of the best model get an average accuracy of 90% and a standard deviation of 0.07.

Keywords. *K-fold cross validation, Machine Learning, Random Forest*

DAFTAR ISI

LEMBAR PERSETUJUAN HASIL SIDANG SKRIPSI i

LEMBAR PERNYATAAN ii

LEMBAR PERSEMBAHAN iii

KATA PENGANTAR iv

ABSTRAK vi

ABSTRACT vii

DAFTAR ISI viii

DAFTAR GAMBAR x

DAFTAR TABEL xii

DAFTAR LAMPIRAN xiii

BAB I PENDAHULUAN 1

A. Latar Belakang 1

B. Rumusan Masalah 3

C. Tujuan Penelitian 4

D. Batasan Masalah 4

E. Manfaat Penelitian 4

BAB II KAJIAN PUSTAKA 5

A. Exploratory Data Analysis 5

B. Data *Pre-Processing* 7

C. Struktur Data Pohon 12

D. *Random Forest* 16

E. Evaluasi Machine Learning 25

F. Uji Validitas *Machine Learning* 27

G. Kelulusan Mata Kuliah 28

BAB III METODOLOGI PENELITIAN 31
A. Alat Penelitian 31
B. Tahapan Penelitian 31
BAB IV HASIL DAN PEMBAHASAN 45
A. Program CART 45
B. Program *Random Forest* 49
C. Hasil Tahapan *Pre-processing* Data 53
D. Hasil Tahapan Pemodelan *Random Forest* 62
E. Hasil Analisis Model *Random Forest* 68
BAB V KESIMPULAN DAN SARAN 70
A. Kesimpulan 70
B. Saran 70
DAFTAR PUSTAKA 72
LAMPIRAN 76
DAFTAR RIWAYAT HIDUP 96

DAFTAR GAMBAR

Gambar 2.1 Noise 8
Gambar 2.2 Ilustrasi struktur data pohon 12
Gambar 2.3 Simpul pohon dalam kode program 15
Gambar 2.4 Simpul pohon yang telah dimodifikasi 15
Gambar 2.5 Struktur pohon pada metode CART 18
Gambar 2.6 Ilustrasi dari *bagging* 21
Gambar 2.7 Ilustrasi dari random forest 22
Gambar 2.8 Ilustrasi K-*Fold Cross Validation* 28
Gambar 3.1 Tahapan penelitian 32
Gambar 3.2 Pratinjau *Datasets* 35
Gambar 3.3 *Datasets info* 36
Gambar 3.4 Korelasi Antar Kolom 37
Gambar 3.5 Alur *pre-processing* data 38
Gambar 3.6 Alur pemodelan *Random Forest* 41
Gambar 3.7 Ilustrasi *random forest* 42
Gambar 4.1 *Pseudocode method gini_index* 45
Gambar 4.2 *Pseudocode method impurity_node* 45
Gambar 4.3 *Pseudocode method split_data* 46
Gambar 4.4 *Pseudocode method make_tree* 47
Gambar 4.5 *Pseudocode method class_value* 48
Gambar 4.6 *Pseudocode method predict* 48
Gambar 4.7 *Pseudocode method visit_tree* 48
Gambar 4.8 *Pseudocode method bootstrapping* 49
Gambar 4.9 *Pseudocode method m_feature_selection* 50
Gambar 4.10 *Pseudocode make_forest* 51
Gambar 4.11 *Pseudocode method predict* 51
Gambar 4.12 *Pseudocode method visit_forest* 52
Gambar 4.13 *Pseudocode method voting_forest* 52
Gambar 4.14 Sebaran nilai akhir mahasiswa 54
Gambar 4.15 Persentase kelulusan mahasiswa 55

Gambar 4.16 Jumlah nilai *null* tiap kolom 55
Gambar 4.17 *Boxplot* yang menunjukkan *outliers* pada *datasets*..... 56
Gambar 4.18 Perbedaan sebelum dan sesudah kolom dihilangkan..... 56
Gambar 4.19 Predikat mahasiswa..... 57
Gambar 4.20 Ilustrasi pengisian nilai *null*..... 58
Gambar 4.21 Nilai *outliers* 59
Gambar 4.22 *Outliers* sesudah ditangani..... 59
Gambar 4.23 Jumlah fitur dan sampel pada data latih dan data tes..... 60
Gambar 4.24 Data latih sebelum *balancing* data..... 61
Gambar 4.25 Data latih sesudah *balancing* data 61
Gambar 4.26 *Hasil confusion matrix* terhadap data latih 63
Gambar 4.27 *Hasil confusion matrix* terhadap data tes..... 63
Gambar 4.28 Galat OOB 65
Gambar 4.29 *Hasil confusion matrix* model baru terhadap data latih..... 66
Gambar 4.30 Hasil *confusion matrix* model baru terhadap data tes..... 66

DAFTAR TABEL

Tabel 2.1 Contoh data yang tidak lengkap 8
Tabel 2.2 Contoh data yang tidak konsisten 9
Tabel 2.3 Tabel nama program studi 10
Tabel 2.4 Tabel nama program studi setelah di-*encode* 10
Tabel 2.5 Terminologi struktur data pohon 13
Tabel 2.6 Perbedaan ID3, C4.5 dan CART 17
Tabel 2.7 *Confusion matrix* 26
Tabel 2.8 Kriteria nilai Universitas Negeri Jakarta 29
Tabel 3.1 Data Nilai Mahasiswa 33
Tabel 3.2 Data Presensi Mahasiswa 34
Tabel 3.3 Data PDDIKTI 34
Tabel 3.4 *Datasets* 35
Tabel 4.1 Jumlah Mahasiswa 54
Tabel 4.2 Tabel hasil *encoding* label 60
Tabel 4.3 Nilai *precision*, *recall*, dan *F-Measure* 64
Tabel 4.4 Hasil akurasi dengan 10-*fold cross validation* 64
Tabel 4.5 Nilai *precision*, *recall*, dan *F-Measure* model baru 67
Tabel 4.6 Perbandingan model lama dan model baru 67
Tabel 4.7 Hasil akurasi *k-fold cross validation* model optimum 68
Tabel 4.8 Hasil *feature importance* 69

DAFTAR LAMPIRAN

Lampiran 1 Surat Permintaan Data..... 76
Lampiran 2 *Datasets*..... 77
Lampiran 3 Hasil *encoding* data latih..... 81
Lampiran 4 Hasil *encoding* data tes..... 84
Lampiran 5 Hasil normalisasi data latih 86
Lampiran 6 Hasil normaliasi data tes 91
Lampiran 7 Tabel hasil uji OOB *Error* 94
Lampiran 8 *Source Code* 95

BAB I
PENDAHULUAN

A. Latar Belakang

Bagi kehidupan mahasiswa, salah satu faktor penting selama berkuliah adalah dapat lulus dan menyelesaikan seluruh beban akademik dengan baik. Beban akademik direpresentasikan dalam bentuk banyaknya mata kuliah yang harus ditempuh mahasiswa (setara 144 SKS total untuk program studi sarjana) dan semuanya harus lulus dalam kurun waktu paling lama empat belas semester (setara tujuh tahun) (Ristekdikti, 2015). Agar dapat mencapai target di atas, seorang mahasiswa setidaknya dituntut lulus beberapa mata kuliah tiap semesternya. Kelulusan mata kuliah tiap semester tersebut kemudian dikonversi menjadi sistem penilaian Indeks Prestasi Semester (IPS) untuk mengukur kinerja akademik mahasiswa.

Kualitas dari kelulusan mata kuliah tiap semester memengaruhi durasi masa studi mahasiswa. Semakin banyak mata kuliah yang lulus, maka semakin besar peluang mahasiswa untuk lulus tepat waktu. Selama menjalani perkuliahan, ada mahasiswa yang gagal dalam suatu mata kuliah sehingga diharuskan mengulang mata kuliah tersebut pada semester atau tahun berikutnya. Fenomena seperti ini menyebabkan masa studi mahasiswa berpeluang lebih lama dan panjang, yang dapat berujung putus kuliah atau *drop out* (DO). Berdasarkan data Statistik Pendidikan Tinggi tahun 2020, tingkat putus kuliah mahasiswa di Indonesia pada tahun 2019 tercatat sebesar 7% dari total mahasiswa terdaftar 8.483.213 orang (Kemendikbud, 2020). Selain itu, setiap perguruan tinggi Indonesia distandarisasi melalui sistem akreditasi program studi dengan salah satu dimensi penilaian untuk meningkatkan akreditasi adalah nilai *output* dan *outcomes* yang di dalamnya mencakup kualitas lulusan (BAN-PT, 2019).

Untuk mengurangi tingkat kegagalan dan putus kuliah di kalangan mahasiswa, sebagai pemangku kepentingan akademik, institusi seperti perguruan tinggi dapat membuat suatu solusi tepatguna seperti memprediksi tingkat kelulusan mahasiswa pada mata kuliah tertentu. Lembaga pendidikan berperan dalam menentukan kinerja akademik mahasiswa pada lingkungan yang kompetitif

sehingga mahasiswa mampu menyusun serta memilih keputusan yang tepat untuk strategi pembelajarannya di masa mendatang (Abdul Bujang et al., 2021). Sebagai suatu solusi dan peran perguruan tinggi sebelumnya, metode yang dapat dikembangkan untuk mengatasi permasalahan di atas adalah metode *machine learning*.

Machine learning atau yang dalam bahasa Indonesia adalah pembelajaran mesin merupakan suatu bagian dari *Artificial Intelligence* (AI). AI banyak digunakan untuk menggantikan maupun menirukan perilaku manusia dalam menyelesaikan suatu masalah (Ahmad, 2017). *Machine learning* terdiri dari berbagai teknik atau algoritma yang dapat digunakan untuk melakukan suatu prediksi ataupun melakukan klasifikasi. Teknik atau algoritma ini nantinya akan memperkirakan kemungkinan yang akan terjadi di masa yang akan datang dengan melihat beberapa informasi dan/atau melihat pola data yang ada pada masa lalu.

Istilah *machine learning* atau pembelajaran mesin pertama kali digunakan pada tahun 1920-an oleh matematikawan yang bernama Adrien Marie Legendre, Thomas Bayes dan Andrey Markov untuk menjelaskan dasar-dasar dan konsep dari *machine learning* (Takdirillah, 2020). Beberapa contoh penelitian yang menggunakan *machine learning* untuk mengatasi beberapa permasalahan adalah sebagai berikut: Implementasi Algoritma *Support Vector Machine* (SVM) untuk Prediksi Ketepatan Waktu Kelulusan Mahasiswa (Pratama et al., 2018), Implementasi *Naïve Bayes Classifier* dan *Confusion Matrix* pada Analisis Sentimen Berbasis Teks pada Twitter (Normawati & Prayogi, 2021), Klasifikasi Hutan-Non Hutan Data Alos Palsar Menggunakan Metode *Random Forest* (Sambodo et al., 2014), *Learning Progress Modeling for Monitoring Student* (Arafiyah et al., 2020) dan lainnya.

Berdasarkan penjelasan di atas, *machine learning* dapat dimanfaatkan untuk memprediksi sesuatu berdasarkan data masa lalu. Perguruan tinggi memiliki kewenangan dan sumber daya yang dapat digunakan secara optimal guna mengimplementasikan *pemodelan machine learning*. Pemanfaatan pemodelan ini diharapkan dapat memantau kemajuan dan/atau meningkatkan proses belajar mahasiswa pada suatu mata kuliah tertentu, sehingga mahasiswa dapat mengetahui

klasifikasi predikat kelulusan mata kuliah mereka dan dapat memilih strategi yang tepat untuk perkuliahan mendatang.

Terdapat penelitian terkait prediksi kelulusan mata kuliah sebagai contoh penelitian yang berjudul "Prediksi Kelulusan Mata Kuliah Menggunakan *Hybrid Fuzzy Inference System*" yang menghasilkan akurasi sebesar 93% (Izzah & Widyastuti, 2016). Penelitian lain yang sejenis berjudul "Penerapan Algoritma C5.0 Untuk Prediksi Kelulusan Pembelajaran Mahasiswa Pada Matakuliah Arsitektur Sistem Komputer" menghasilkan akurasi yang cukup tinggi yaitu sebesar 93,33% (Sungkar & Qurohman, 2021).

Dalam penelitian ini, untuk mengklasifikasi kelulusan pada suatu mata kuliah, penulis memandang dapat menggunakan pemodelan *machine learning* dengan algoritma *random forest*. *Random forest* merupakan algoritma yang masuk ke dalam golongan *ensemble method*. *Ensemble method* merupakan teknik *machine learning* yang mengombinasikan beberapa model dasar *machine learning* untuk menghasilkan suatu model yang optimum untuk melakukan klasifikasi. *Random forest* juga merupakan salah satu algoritma klasifikasi dalam *machine learning* yang memiliki kinerja tinggi, skalabilitas yang baik dan mudah untuk digunakan. Dengan beberapa parameter masukan, algoritma ini membangun dan mengombinasikan "pohon" untuk mendapatkan "pepohonan" atau "hutan". "Pohon-pohon" ini nantinya akan digunakan untuk mengidentifikasi dan memprediksi sampel data yang ada pada data latih (Arafiyah et al., 2020).

B. Rumusan Masalah

Berdasarkan pemaparan latar belakang terkait tingkat kegagalan dan putus kuliah mahasiswa terutama akibat tidak lulus mata kuliah dan terlalu lama berkuliah, maka penulis merumuskan masalah yang ada dalam dua pertanyaan penelitian sebagai berikut:

1. Bagaimana membuat model *machine learning* untuk mengklasifikasikan kelulusan mahasiswa pada suatu mata kuliah dengan algoritma *random forest*?

2. Bagaimana uji validitas model *machine learning* dalam mengklasifikasikan kelulusan mahasiswa pada suatu mata kuliah dengan menggunakan model yang telah dibuat?

C. Tujuan Penelitian

Tujuan penulis dalam pelaksanaan penelitian ini adalah sebagai berikut:

1. Dapat membuat model *machine learning* dengan algoritma *random forest* dalam mengklasifikasikan kelulusan mata kuliah mahasiswa
2. Dapat mengetahui uji validitas dari model *machine learning* yang telah dibuat.

D. Batasan Masalah

Penelitian yang dilakukan hanya dibatasi pada satu mata kuliah. Sementara data yang digunakan dalam penelitian ini adalah data nilai-nilai mahasiswa pada mata kuliah Kalkulus Integral semester 112 dan 114 di Rumpun Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Negeri Jakarta.

E. Manfaat Penelitian

Manfaat dari penelitian ini adalah sebagai berikut:

1. Membantu untuk menambah pengetahuan tentang pengklasifikasian *machine learning* dengan algoritma *random forest*.
2. Dapat membantu mahasiswa untuk dapat menyusun strategi agar dapat lulus dalam mata kuliah.
3. Dapat membantu dosen untuk mengevaluasi metode dan bahan pembelajaran.
4. Membantu program studi untuk meningkatkan proses pembelajaran mahasiswa.

BAB II
KAJIAN PUSTAKA

Dalam bab ini, penulis akan membahas beberapa teori yang terkait dengan penelitian yang dilakukan. Teori-teori yang dibahas antara lain adalah EDA (*Exploratory Data Analysis*), data *pre-processing*, struktur data pohon, *random forest*, evaluasi *machine learning*, uji validitas *machine learning*, dan kelulusan mata kuliah.

A. Exploratory Data Analysis

Exploratory Data Analysis (EDA) adalah salah satu bagian dari proses *data science* sebagai strategi dalam menganalisis sebuah data. EDA pertama kali diperkenalkan oleh salah satu ahli statistik pada tahun 1961 yang bernama John W Tukey. EDA didefinisikan sebagai pekerjaan detektif. Hal ini dikarenakan peneliti mengeksplorasi data tanpa memiliki ide atau bias tentang informasi apa yang akan diperoleh dari data yang akan dianalisis (Vegari & Budi, 2020).

Komponen-komponen dalam EDA meliputi berbagai penghitungan statistik dasar, visualisasi data, hipotesis atau dugaan awal, pemeriksaan asumsi, hingga pada *reporting*. Dalam melakukan EDA, tidak hanya terpaku dalam suatu teknik tertentu saja melainkan dapat melakukan eksplorasi data dengan berbagai macam teknik dan berbagai perspektif yang ada untuk mendapatkan suatu informasi dalam data tersebut. Hal ini dikarenakan, EDA memiliki karakteristik yang dapat berubah-ubah sesuai yang diperlukan untuk melakukan investigasi dan identifikasi pada suatu fenomena yang muncul saat melakukan penelitian empiris (Wahyuni et al., 2019).

Dalam melakukan *Exploratory Data Analysis* dapat menggunakan beberapa teknik yang dapat dilakukan, yaitu:

1. **Statistik deskriptif**

   yaitu mendeskripsikan atau merangkum data sehingga menghasilkan informasi secara umum tanpa bertujuan untuk menarik kesimpulan. Statistik deskriptif dapat menampilkan beberapa informasi penting seperti nilai rata-rata, median, modus, standar deviasi, variansi dan kecekungan. Statistik

deskriptif ini dapat ditampilkan dalam berbagai bentuk seperti tabel, diagram, grafik, dan lain-lain.

2. ***Univariate analysis***

   yaitu menganalisis kolom secara terpisah dan melihat distribusi datanya. *Univariate analysis* secara umum terbagi dua, yaitu numerik dan kategori. Analisis ini digunakan juga dengan tujuan untuk mengambil kesimpulan dengan menggunakan beberapa analisis inferensial yang mungkin digunakan.

3. ***Multivariate analysis***

   yaitu menggabungkan beberapa kolom dengan tujuan menemukan hubungan antara satu kolom dengan kolom lainnya. Analisis *multivariate* ini melibatkan variabel dalam jumlah lebih atau sama dengan tiga variabel.

Tujuan dari EDA adalah mencari pola dari sebuah data. Dengan adanya *big data* saat ini, eksplorasi data menjadi lebih rumit dikarenakan dimensi data yang terlalu besar, dengan EDA dapat mengurangi dimensi data atau memperkaya pemahaman terhadap *big data* tersebut dengan melakukan visualisasi data. EDA juga digunakan untuk mengoptimalkan suatu pengetahuan mengenai data, mengekstrak variabel-variabel penting, menangani *missing values*, pengelompokan, transformasi, distribusi data, hingga memahami hubungan antar variabel dan juga mendeteksi *outlier* dan anomali pada data. Semua hal yang telah disebutkan sebelumnya dapat digunakan untuk analisis data yang ada dan mengoptimalkan hasil klasifikasi dan prediksi.

Dengan melakukan EDA, maka dapat memahami kondisi *datasets* yang akan digunakan dalam penelitian. Memahami kondisi *datasets* dapat merujuk dalam hal berikut:

1. Mengekstrak variabel penting dan meninggalkan variabel yang tidak berguna (*data preprocessing*).
2. Mengidentifikasi nilai yang hilang (*missing values*) dan kesalahan manusia (*human error*).
3. Memahami hubungan, atau kekurangan, antar variabel.
4. Memaksimalkan wawasan yang dimiliki atas kondisi data dan meminimalkan potensi kesalahan di kemudian hari.

Salah satu cara untuk memahami kondisi *datasets* dengan EDA seperti yang telah dijelaskan di atas adalah dengan mengetahui hubungan korelasi antar data. Analisis korelasi adalah salah satu teknik dalam statistik yang berguna untuk menganalisis suatu hubungan dua buah variabel untuk mengetahui apakah variabel tersebut memiliki hubungan yang erat, kurang erat atau sama sekali tidak memiliki hubungan.

Dalam bentuk hubungan antar variabel akan dihasilkan bentuk hubungan yang positif atau negatif. Korelasi yang positif terjadi jika variabel X mengalami kenaikan dan diikuti dengan variabel Y. Sementara korelasi negatif terjadi jika variabel X mengalami kenaikan dan variabel Y mengalami penurunan.

Terdapat beberapa metode yang sering digunakan dalam mengukur korelasi. Metode-metode tersebut ialah sebagai berikut:

1. Koefisien korelasi *pearson*
2. Koefisien korelasi *spearman*
3. Dan koefisien data berkelompok

EDA menjadi sangat penting karena tanpa dilakukannya EDA maka banyak informasi penting pada *datasets* akan hilang. Jika pemodelan data dilakukan tanpa menerapkan EDA, ada kemungkinan model yang telah dibuat memiliki kinerja yang buruk karena pemodelan dilakukan tanpa benar-benar memahami keadaan data yang ada. Hal di atas dapat menyebabkan banyak waktu terbuang hanya untuk mencari tahu kesalahan mana yang perlu diperbaiki dan mengulang lagi proses pemodelan data.

B. Data *Pre-Processing*

Data *pre-processing* berperan penting dalam pengolahan data. Data *pre-processing* adalah sebuah konsep untuk mengubah data mentah atau data kotor menjadi data bersih yang siap untuk diolah. Data-data yang ada pada kehidupan sehari-hari umumnya memiliki beberapa masalah sebagai berikut:

1. Data yang tidak lengkap (data yang hilang)

   Merupakan data yang atributnya tidak lengkap dan tidak merepresentasikan data yang ada, adanya nilai yang hilang pada beberapa

atribut atau pada sampel tertentu. Berikut contoh data yang tidak lengkap ada pada tabel 2.1 berikut:

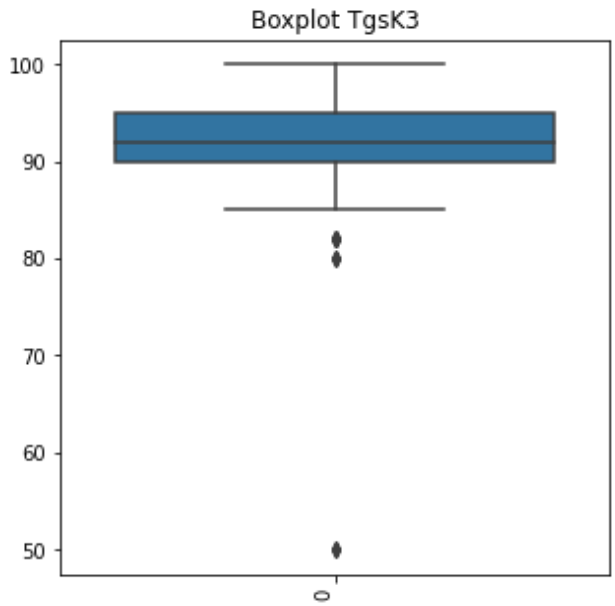
Tabel 2.1 Contoh data yang tidak lengkap

No	TgsK1	TgsK2	TgsK3	TgsK4	TgsK5	TgsK6
1	90.00	93.00	92.00	90.00	95.00	93.00
2	100.00	85.00	95.00			
3	100.00	100.00	95.00			
4	93.00	95.00	93.00	95.00	95.00	93.00

Pada tabel 2.1 di atas dapat dilihat bahwa data dengan nomor 2 dan 3 tidak memiliki data yang lengkap, hal ini dapat dilihat adanya atribut yang kosong atau tidak tercatat. Jika data sudah cukup merepresentasikan kebutuhan untuk pemodelan maka sampel data yang tidak lengkap bisa di-*drop* atau dihilangkan. Jika data terlalu sedikit dan/atau ingin tetap mempertahankan sampel tersebut, dibutuhkan suatu teknik atau algoritma untuk melengkapi kekurangan data tersebut (Adinugroho & Sari, 2018).

2. Banyaknya *noise*

Merupakan data yang mengandung kesalahan atau terdapat *outlier*, sehingga membutuhkan suatu teknik untuk menyaring data-data penting dan memisahkan dari *outlier*. Contoh data yang memiliki *outlier* bisa dilihat pada gambar berikut.



Gambar 2.1 Noise

Data *outlier* adalah suatu data yang terlalu jauh dan mempunyai perbedaan yang cukup besar dengan data lainnya. Hal ini akan menyebabkan penggunaan data menjadi bias karena tidak dapat mencerminkan data yang dibutuhkan. Tetapi, ada kalanya *outlier* berperan penting dalam beberapa penelitian atau analisis suatu permasalahan di antaranya adalah untuk mendiagnosis suatu penyakit pada tanaman, mendeteksi kerugian suatu perusahaan dan lain-lain (Adinugroho & Sari, 2018). Contoh *outlier* dapat dilihat pada gambar 2.1 dimana *plot* titik yang berada di luar *boxplot* merupakan nilai *outliers*.

3. Data yang tidak konsisten

Merupakan data yang nilai pada atributnya tidak sama atau tidak seragam sehingga menyebabkan data yang ada memiliki kesan yang ambigu atau kurang jelas. Contoh penulisan data yang tidak konsisten dapat dilihat pada tabel berikut.

Tabel 2.2 Contoh data yang tidak konsisten

Nama Siswa	Umur
Acit	19
Bintang	12/04/96
Cinta	15 Juli 1998
Dika	20
Dika	1/07/1998

Dapat dilihat pada tabel di atas bahwa atribut umur yang seharusnya berisi satuan tahun, namun pada beberapa sampel ditulis dengan format yang berbeda. Data yang tidak konsisten juga merupakan data yang memiliki duplikasi dengan maksud yang sama tetapi dengan penulisan yang tidak sama atau homogen sehingga menjadikan data ambigu (Adinugroho & Sari, 2018).

Dalam data *pre-processing* juga dilakukan transformasi data seperti *encoding* pada data kategori, *feature scaling* data numerik dan juga *balancing data*.

1. *Encoding* data kategori

encoding data kategori adalah sebuah proses untuk mengonversi data kategori menjadi data numerik agar dapat diolah pada algoritma *machine learning* nantinya. Metode yang digunakan untuk meng-*encode* data kategori menyesuaikan dengan

skala pengukuran data kategori itu sendiri. Ada dua jenis skala pengukuran data kategori yaitu skala ordinal dan skala nominal. Data dengan skala ordinal ialah data kategori yang dapat dibedakan dan dapat diurutkan tingkatannya sebagai contoh data tingkat kepuasan pelanggan yang dinyatakan dengan sangat tidak puas, tidak puas, cukup puas, puas, dan sangat puas. Sementara untuk data skala nominal ialah jenis data kategori yang dipergunakan hanya untuk penggolongan saja dan tidak dapat diurutkan tingkatannya sebagai contoh, data jenis kelamin laki-laki dan perempuan (Ramdani & Utami, 2022).

Untuk meng-*encode* data kategori dengan jenis ordinal dilakukan dengan cara diurutkan sesuai dengan tingkatannya lalu diberi penomoran sesuai tingkatannya. Sebagai contoh, sangat tidak puas (1), tidak puas (2), cukup puas (3), puas (4), dan sangat puas (5). Sementara untuk meng-*encode* skala nominal dilakukan dengan metode *one-hot-encoding*. Metode *one-hot-encoding* ini nantinya akan mengubah data kategori menjadi sebuah biner 0 dan 1, dimana semua elemen nantinya akan bernilai 0 terkecuali elemen yang memiliki kategori tersebut akan bernilai 1. Contoh *encoding* dengan *one-hot-encoding* dapat dilihat pada tabel 2.3 dan 2.4 di bawah ini.

Tabel 2.3 Tabel nama program studi

ID	Prodi
1	Matematika
2	Pendidikan Matematika
3	Ilmu Komputer

Tabel 2.4 Tabel nama program studi setelah di-*encode*

ID	Matematika	Pendidikan Matematika	Ilmu Komputer
1	1	0	0
2	0	1	0
3	0	0	1

Setiap kategori dari kolom Prodi pada tabel 2.3 akan menjadi kolom baru yang nantinya akan bernilai 0 atau 1 seperti pada tabel 2.4. Dengan penggunaan *one-hot-encoding* maka akan terjadi penambahan jumlah kolom.

2. *Feature scaling*

Untuk *feature scaling* pada data numerik dilakukan dikarenakan pada beberapa *datasets* terdapat perbedaan rentang nilai di setiap atributnya. Perbedaan

rentang nilai pada setiap atribut menyebabkan tidak berfungsinya atribut yang memiliki nilai jauh lebih kecil dibandingkan dengan atribut-atribut lainnya. Oleh karena itu, diperlukan adanya *feature scaling* data numerik untuk menyamakan rentang nilai pada setiap atribut dengan skala tertentu (Nasution et al., 2019). Ada beberapa metode yang sering digunakan untuk *feature scaling* data numerik diantaranya adalah normalisasi *min-max* dan normalisasi *z-score*.

Normalisasi *min-max* akan mengubah data numerik menjadi skala 0 (minimal) hingga 1 (maksimal). Rumus untuk normalisasi *min-max* adalah sebagai berikut:

$$x_{norm} = \frac{x_{old} - x_{min}}{x_{max} - x_{min}} \tag{2.1}$$

Dimana:

x_{norm} : Nilai *x* baru setelah di normalisasi.

x_{old} : Nilai *x* lama.

x_{min} : Nilai minimum pada *datasets*.

x_{max} : Nilai maksimum pada *datasets*.

Sementara untuk normalisasi *z-score* adalah normalisasi berdasarkan *mean* (nilai rata-rata) dan *standard deviation* (standar deviasi). Rumus dari normalisasi *z-score* adalah sebagai berikut:

$$x_{norm} = \frac{x_{old} - \mu}{\sigma} \tag{2.2}$$

Dimana:

x_{norm} : Nilai *x* baru setelah di normalisasi.

x_{old} : Nilai *x* lama.

μ : Nilai rata-rata populasi.

σ : Nilai simpangan baku.

3. *Balancing data*

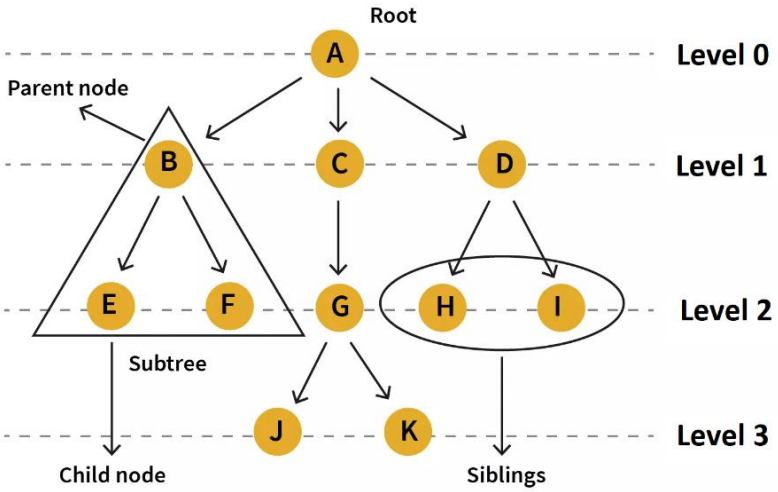
Selain *encoding* dan *feature scaling* data numerik, untuk beberapa kasus *datasets* dapat juga dilakukan *balancing data* untuk menyeimbangkan kelas atau label pada *datasets*. Menyeimbangkan kelas pada *datasets* dilakukan agar model *machine learning* tidak memiliki kinerja yang buruk karena proses klasifikasinya hanya dilakukan pada kelas atau label yang mayoritas (Ardiningtyas & Rosa, 2021).

Salah satu metode untuk menyeimbang data yang tidak seimbang (*imbalance data*) adalah SMOTE (*Synthetic Minority Oversampling Technique*). SMOTE pertama kali diperkenalkan oleh Nithes V Chawla dengan cara membuat replikasi data yang biasa dikenal dengan data sintesis (*syntetic data*). Metode SMOTE bekerja mencari KNN (*k-nearest neighbors*) untuk setiap data pada kelas minoritas, lalu dibuat data sintesis sebanyak persentase duplikasi yang diinginkan antara data minor dan KNN yang dipilih secara acak (Siringoringo, 2018).

C. Struktur Data Pohon

Struktur data pohon merupakan salah satu struktur data dalam bidang informatika yang digunakan untuk merepresentasikan hubungan data yang bersifat hierarki antar elemen-elemennya. Hierarki pada struktur data pohon dapat diumpamakan sebagai hubungan orang tua dan anak. Struktur data pohon merupakan sebuah struktur data yang non-linear dikarenakan data yang disimpan pada struktur data ini tidak disimpan secara berurutan melainkan diatur ke berbagai *level* atau tingkatan hierarki.

Struktur data pohon terdiri dari kumpulan simpul atau *node*, dimana setiap simpul digunakan untuk menyimpan nilai dan sebuah *list* rujukan ke simpul lain yang disebut sebagai simpul anak atau *child node*. Tiap-tiap simpul pada struktur data pohon dihubungkan dengan sebuah garis yang disebut dengan *edge*.



Sumber: https://www.scaler.com/

Gambar 2.2 Ilustrasi struktur data pohon

Ilustrasi dari struktur data pohon dapat dilihat dalam gambar 2.3 di atas. Ada beberapa istilah dalam struktur data pohon yang bisa dilihat dalam tabel berikut.

Tabel 2.5 Terminologi struktur data pohon

Terminologi	Deskripsi	Posisi pada ilustrasi
Root node	*Root node* atau simpul akar merupakan simpul paling atas dari suatu pohon	Simpul A
Parent node	*Parent node* merupakan simpul pendahulu dari simpul lainnya	Simpul A merupakan *parent node* dari simpul B, C dan D
Child node	*Child node* merupakan simpul turunan dari simpul di atasnya (*parent node*)	Simpul F dan E adalah simpul anak dari simpul B
Leaf node	*Leaf node* merupakan simpul yang tidak memiliki *child node* sama sekali	Simpul E, F, J, K, H, dan I adalah *leaf node*
Internal Node	*Internal node* merupakan simpul yang minimal memilik satu *child node*	Semua simpul adalah *internal node* kecuali simpul E, F, J, K, H, dan I
Edge	*Edge* merupakan garis yang menghubungkan dua buah simpul	Garis yang menghubungkan simpul A dan B, A dan C, A dan D, dan lain-lain
Siblings	*Siblings* merupakan simpul yang memiliki *parent node* yang sama	Simpul H dan I merupakan *siblings* karena memiliki *parent node* simpul D
Path	*Path* merupakan sejumlah *edge* yang berurutan dari suatu *node* ke *node* lain	AC, CG, GK merupakan *path* dari simpul A ke simpul K
Level	*Level* merupakan representasi dari generasi tiap *node* dari atas ke bawah, hitungan *level* dimulai dari 0	Simpul A berada pada *level* 0. Simpul B, C dan D berada pada *level* 1, dan seterusnya
Height of node	*Height of node* merupakan jumlah *edge* dari *leaf node* ke suatu simpul	*Height of node* dari simpul E adalah 0, sementara untuk simpul

Terminologi	Deskripsi	Posisi pada ilustrasi
		B adalah 1, dan seterusnya
Depth of node	*Depth of node* merupakan jumlah edge dari *root node* ke suatu simpul	*Depth of node* dari simpul A adalah 0, sementara untuk simpul B adalah 1, dan seterusnya
Height of tree	*Height of tree* merupakan *path* terpanjang dari *leaf node* terdalam ke *height node*	*Height of tree* dari ilustrasi adalah 3, dihitung dari simpul J/K ke *root node*
Subtree	*Subtree* merupakan sebuah simpul beserta turunannya	Simpul B, E dan F merupakan salah satu dari *subtree*

Ada beberapa jenis-jenis dari struktur data pohon yang sering ditemui, diantaranya adalah sebagai berikut.

1. *General tree*

   Merupakan struktur data pohon yang tidak memiliki batasan jumlah *node*. Dengan kata lain setiap simpul bebas memiliki berapapun jumlah *child node*.

2. *Binary tree*

   Merupakan jenis pohon yang hanya memiliki paling banyak 2 *child node* pada setiap simpulnya. Biasanya simpul-simpul tersebut disebut dengan simpul kiri (*left node*) dan simpul kanan (*right node*).

3. *Binary search tree*

   Merupakan sebuah algoritma untuk melakukan pencarian atau pengurutan. Simpul sebelah kiri pada struktur pohon ini memiliki nilai yang lebih kecil daripada nilai pada *parent node*-nya, sementara simpul sebelah kanan memiliki nilai yang lebih besar daripada nilai pada *parent node*-nya.

4. *Balanced tree*

   Merupakan struktur pohon yang memiliki ketinggian sama atau hanya berbeda 1 pada *subtree* sebelah kanan dan kirinya.

Simpul atau *node* pada struktur data pohon dalam kode program umumnya seperti pada gambar 2.3 berikut.

```
struct Node
{
  int data;
  struct Node *left_child;
  struct Node *right_child;
};
```

Sumber:https://www.geeksforgeeks.org/

Gambar 2.3 Simpul pohon dalam kode program

Dalam penelitian yang penulis buat simpul atau *node* pada struktur pohon yang digunakan akan dimodifikasi agar pada setiap simpul dapat menyimpan *'feature'* dan *'threshold'*. *'Feature'* dan *'threshold'* ini nantinya akan berperan sebagai rujukan untuk ke simpul anak. Untuk menyimpan *'feature '*dan *'threshold'* pada suatu simpul secara bersamaan penulis akan menyimpannya pada variabel yang ada di dalam *class node* atau kelas simpul di *python*.

```
struct Node {
    float impurity
    float gini
    int feature
    float threshold
    struct Node *left
    struct Node *right
    int depth
    int class_value
    string feature_type
    int n_samples = n_samples
};
```

Gambar 2.4 Simpul pohon yang telah dimodifikasi

D. *Random Forest*

Random forest adalah salah satu algoritma klasifikasi dalam *machine learning* yang memiliki keunggulan dalam skalabilitasi yang baik, kinerja yang tinggi, dan mudah untuk digunakan. *Random forest* juga merupakan salah satu dari sekian banyak metode *ensemble* yang bertujuan untuk menemukan dan meningkatkan akurasi dari klasifikasi suatu data dari sebuah pemilah tinggal yang tidak stabil dengan menggabungkan metode-metode yang sama dan menggunakan proses *majority voting* untuk menentukan hasil dari klasifikasi. Istilah *random forest* sendiri diusulkan pertama kali oleh Tin Kam Hao pada tahun 1995 dengan nama *random decision forest*. Kemudian metode ini dikembangkan lagi pada tahun 2001 oleh Leo Breiman dalam tulisannya yang berjudul *Random Forest*.

Random forest (RF) merupakan pengembangan dari metode CART dengan menggunakan metode *bootstrap aggregating* (*bagging*) dan *random feature selection* (Breiman, 2001). *Bootstrap aggregarting* (*Bagging*) adalah teknik yang dapat digunakan untuk membentuk sampel *bootstrap* dimana setiap pohon keputusan dibangun dengan menggunakan sampel *bootstrap* dari data kandidat atribut untuk dibagi di setiap *node* yang berasal dari himpunan atribut acak dari hasil data yang dipilih dan disimpan (Mambang & Byna, 2017). *Bagging* merupakan metode *ensemble* yang banyak diterapkan pada algoritma klasifikasi yang bertujuan meningkatkan akurasi pengklasifikasi dengan menggabungkan pengklasifikasi tunggal dan hasilnya lebih baik daripada *random sampling* (Widiastuti, 2018). Untuk penjelasan lebih lanjut tentang CART dan *Bagging* adalah sebagai berikut:

1. *Classification Tree and Regression Tree* (CART)

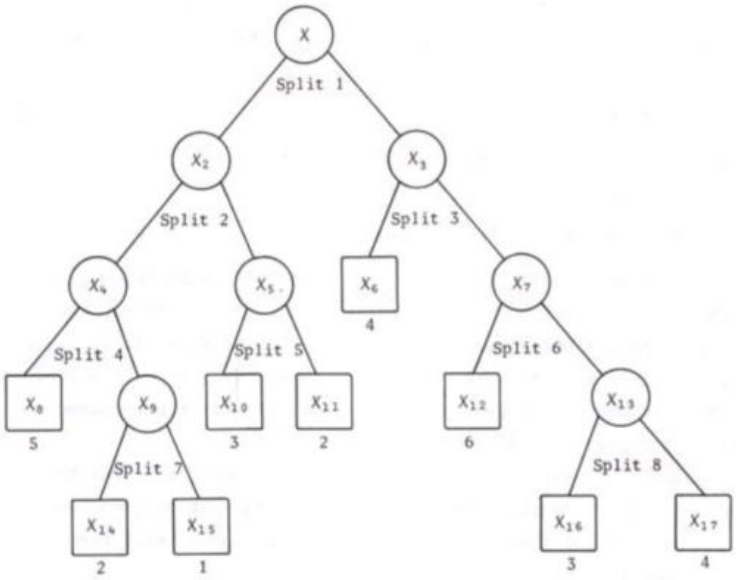
Classification Tree and Regression Tree atau bisa juga disebut dengan CART merupakan salah satu metode dalam membentuk suatu pohon keputusan atau *Decision Tree*. *Decision Tree* atau pohon keputusan merupakan suatu algoritma klasifikasi *machine learning* yang cukup populer dan banyak digunakan secara praktis dengan menggunakan struktur berhierarki. Tujuan utama menggunakan *decision tree* atau pohon keputusan adalah kemampuannya untuk memecah proses pengambilan keputusan yang kompleks menjadi proses yang lebih sederhana, sehingga pengambil keputusan dapat menginterpretasikan solusi dari masalah

dengan lebih baik. *Decision tree* juga berguna untuk mengeksplorasi data, menemukan hubungan tersembunyi antara sejumlah calon variabel *input* dengan sebuah variabel target. *Decision tree* memadukan antara eksplorasi data dan pemodelan, sehingga sangat bagus sebagai langkah awal dalam proses pemodelan bahkan ketika dijadikan sebagai model akhir dari beberapa teknik lain. Ada beberapa metode untuk membentuk suatu pohon keputusan yaitu *Iterative Dychotomizer version 3* (ID3), C4.5, dan *Classification and Regression Tree* (CART). Perbedaan antara ID3, C4.5 dan CART dapat dilihat pada tabel 2.3 berikut. Metode C4.5 cocok digunakan pada klasifikasi untuk data-data yang bersifat non-numerik, sementara metode CART cocok digunakan untuk klasifikasi data numerik (Alverina et al., 2018).

Tabel 2.6 Perbedaan ID3, C4.5 dan CART

	ID3	C4.5	CART
Tipe atribut	Kategoris	Kategoris dan Kontinu	Kategoris dan Kontinu
Cara pemilihan	Biner atau multinomial	Biner atau multinomial	Biner
Kriteria pemilah	*Entropy* dan *information gain*	*Split entropy* dan *gain ratio*	*Indeks Gini*

Metode CART merupakan gabungan dari dua jenis pohon, yaitu *classification tree* dan *regression tree*. CART merupakan metode eksplorasi data yang didasarkan pada teknik pohon keputusan. Pohon klasifikasi dihasilkan saat peubah respons berupa data kategori sedangkan pohon regresi dihasilkan saat peubah respons berupa data numerik. Pohon terbentuk dari proses pemilahan rekursif biner pada suatu gugus data sehingga nilai peubah respons pada setiap gugus data hasil pemilahan akan lebih homogen (Breiman et al. 1984; Sartono & Syafitri 2010).



Gambar 2.5 Struktur pohon pada metode CART

Struktur dari pohon CART dapat dilihat pada gambar 2.5 berikut. Struktur pohon yang dibentuk merupakan pohon biner yang hanya memiliki dua buah *child node* pada setiap *parent node*. Pohon ini dibentuk dengan pengulangan pemilah dari *subset* X menjadi dua *subset* turunan. Dari gambar 2.4 dapat dilihat X_2 dan X_3 merupakan himpunan bagian dari X, dengan $X = X_2 \cup X_3$. Sama juga dengan X_4 dan X_5 yang merupakan himpunan bagian dari X_2, dengan $X_2 = X_4 \cup X_5$. Dan $X_3 = X_6 \cup X_7$. Untuk *subset* yang tidak terpilah seperti X_6, X_8, X_{10}, X_{11}, X_{12}, X_{14}, X_{15}, X_{16}, dan X_{17} disebut dengan *subset* terminal.

Dalam CART, pembentukan pohon diawali dengan membangun pohon sampai ukuran tertentu (secara umum sampai ukuran pohon maksimum). Pemangkasan pohon dapat dilakukan bila perlu sehingga diperoleh ukuran yang tepat sehingga dihasilkan pohon optimum. Pembangunan pohon klasifikasi CART meliputi tiga hal, yaitu:

1.1. Pemilihan pemilah (*split*)

Pemilihan pemilah atau *split* pada setiap simpul memiliki tujuan untuk menghasilkan simpul dengan tingkat kehomogenan nilai peubah respons paling tinggi. Nilai keheterogenan suatu simpul diperoleh dari nilai impuritasnya. Pada setiap simpul *t*, suatu pemilah akan membentuk dua buah pertanyaan atau pertanyaan biner sehingga

memilah simpul *t* menjadi simpul kiri t_L dan t_R (Tanjung & Kartiko, 2017).

Metode untuk pemilihan simpul pemilah menggunakan fungsi *impurity measure i* (*t*) dengan fungsi *indeks Gini* sebagai berikut:

$$i(t) = 1 - \sum_j P^2(j|t) \tag{2.3}$$

Dengan P(j|t) adalah proporsi pada kelas *j* dan simpul *t*.

Pemilah dimulai dengan cara memeriksa nilai-nilai pada setiap peubah penjelasnya untuk mendapatkan *split-point* yang diperoleh dengan mencari nilai tengah dari 2 nilai atribut yang sudah diurutkan terlebih dahulu. *Goodness of Split* merupakan suatu evaluasi pemilahan oleh pemilah *s* pada simpul *t* yang didefinisikan sebagai penurunan keheterogenan dan didefinisikan sebagai

$$\Delta i(s,t) = i(t) - P_L i(t_l) - P_R i(t_R) \tag{2.4}$$

dengan $P_L i(t_l)$ adalah proporsi pengamatan dari simpul *t* menuju kiri, $P_R i(t_R)$ adalah proporsi pengamatan dari simpul *t* menuju kanan (Tanjung & Kartiko, 2017).

Pemilah terbaik adalah pemilah yang memberikan nilai penurunan keheterogenan atau nilai penurunan impuritas tertinggi. Pohon dikembangkan dengan cara pada simpul t_1, carilah s* yang menghasilkan nilai penurunan impuritas tertinggi yaitu

$$\Delta i(s*, t_1) = \max_{s \in S} \Delta i(s, t_1) \tag{2.5}$$

dengan demikian t_1 dipilih dan kemudian dipilah menjadi t_2 dan t_3 menggunakan s*. Dengan cara yang sama dilakukan juga pencarian pemilah terbaik pada t_2 dan t_3 secara terpisah dan demikian seterusnya (Tanjung & Kartiko, 2017).

1.2. Penentuan simpul terminal

Dalam menentukan simpul terminal, ketika suatu simpul *t* memenuhi salah satu atau lebih kriteria untuk berhenti memilah. Simpul *t* tidak dipilah lagi sehingga pembentukan pohon terhenti sesuai dengan kriteria berikut (Tanjung & Kartiko, 2017):

- Tidak ada lagi peubah respons yang secara signifikan menunjukkan suatu perbedaan terhadap peubah penjelas.
- Jika pohon mencapai batas nilai maksimum dari spesifikasi yang telah ditentukan, maka proses pertumbuhan pohon berhenti.
- Jika ukuran dari *child node* kurang dari nilai ukuran *child node* dari spesifikasi yang telah ditentukan, atau berisi jumlah data pengamatan yang terlalu sedikit maka *node* tidak akan terpilah.

1.3. Penandaan label kelas

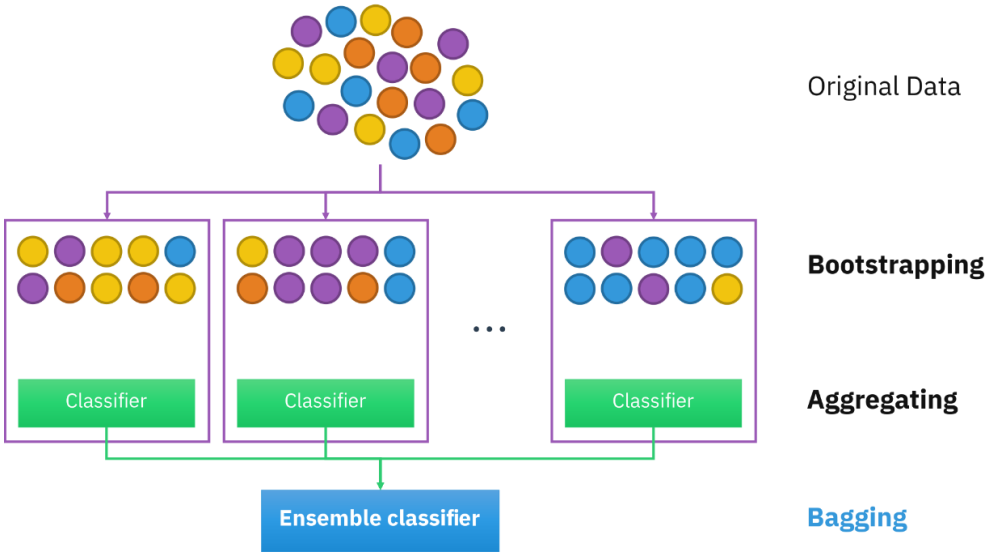
Pemilihan label kelas pada simpul terminal t ditentukan dengan aturan pluralitas atau jumlah terbanyak.

$$P(j_0|t) = \max_j P(j|t) = \max_j \frac{N_j(t)}{N(t)} \tag{2.6}$$

dengan $P(j|t)$ adalah proporsi pada simpul t, $N_j(t)$ adalah jumlah pengamatan kelas j pada simpul t, dan $N(t)$ adalah jumlah pengamatan pada simpul t. Label kelas simpul terminal t adalah j_0 yang memberikan nilai dugaan pengklasifikasian simpul t terbesar (Tanjung & Kartiko, 2017).

2. Bootstrap Aggregating

Bootstrap Aggregating atau bisa juga disebut dengan *bagging* merupakan metode yang diusulkan oleh Breiman pada tahun 1996 dengan menggabungkan metode *bootstrapping* dan *aggregating*. *Bagging* digunakan untuk meningkatkan ketepatan suatu klasifikasi pada data dengan cara mengurangi varians dan juga menjaga atau hanya meningkatkan sedikit bias. *Bagging* memiliki cara kerja dengan membuat beberapa sampel data latih baru dari data latih asli yang diambil secara acak sehingga menghasilkan data latih terbaik. Data latih baru ini kemudian digunakan untuk membuat model.

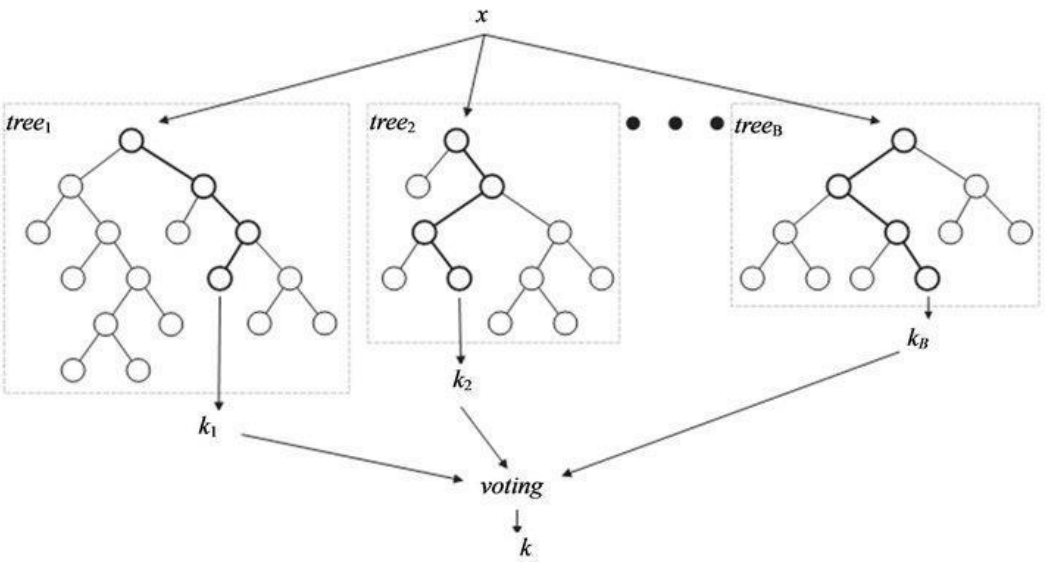


Sumber: https://commons.wikimedia.org/

Gambar 2.6 Ilustrasi dari *bagging*

Pada *bagging* dilakukan pembuatan banyak sampel *bootstrap* dari *dataset* asli. Breiman menyarankan untuk memiliki data latih sebanyak *n* pasang (X_i, Y_i) secara acak, dengan *replacement* dari setiap *bag* pasangan sampel latih $\{(X_i, Y_i), \dots, (X_n, Y_n)\}$. Dengan proses ini maka akan diperoleh data latih baru $D_n(Z)$ dengan kemungkinan pengulangan dan diberikan pengklasifikasi $g_n(X, D_n)$, yang mana dapat menghitung pengklasifikasi acak $g_n(X, D_n) = g_n(X, D_n(Z))$. Setelah itu proses ini diulang untuk setiap sampel *bootstrap* sebanyak *m* dan menghitung voting dari pengklasifikasian $g_n^{(m)}(X, Z^m, D_n)$ (Biau et al., 2008).

Dengan CART dan *Bagging* yang telah dijelaskan sebelumnya, maka selanjutnya akan dijelaskan algoritma *random forest*. Algoritma *random forest* nantinya akan membentuk banyak pohon sehingga dari pohon-pohon tersebut akan membentuk hutan (*forest*), tidak seperti *single tree* yang terdiri dari satu pohon untuk membuat klasifikasi atau prediksi. Pembentukan pohon pada RF dilakukan dengan cara melatih sampel data, sementara pemilihan variabel untuk *split* diambil secara acak. Klasifikasi atau prediksi dapat dilakukan setelah semua pohon terbentuk. Penentuan klasifikasi pada RF dilakukan berdasarkan voting dari masing-masing pohon dan hasil voting terbanyak yang dijadikan sebagai pemenang. Bentuk dari RF secara umum dapat dilihat pada gambar 2.7 berikut.



Sumber: https://www.researchgate.net/

Gambar 2.7 Ilustrasi dari random forest

Berikut ini adalah prosedur atau algoritma untuk membangun *random forest* pada gugus data yang terdiri atas *n* amatan dan *p* peubah penjelas (Breiman, 2001; Breiman & Cutler, 2003):

1. Lakukan penarikan contoh acak berukuran *n* dengan pemulihan pada gugus data. Tahap ini adalah tahap *bootstrap*.
2. Dengan menggunakan contoh *bootstrap*, pohon dibangun sampai mencapai ukuran maksimum yaitu tanpa pemangkasan (*pruning*). Pembangunan pohon dilakukan dengan menerapkan *random feature selection* yaitu *m* peubah penjelas dipilih secara acak di mana $m << p$, selanjutnya pemilah terbaik dipilih berdasarkan *m* peubah penjelas.
3. Ulangi langkah 1 dan 2 sebanyak *k* kali, sehingga terbentuk sebuah hutan yang terdiri atas *k* pohon.

Pemilihan *m* sangat berperan dalam menentukan kinerja *random forest* yang telah dibuat, hal ini dikarenakan ukuran *m* yang dipilih akan sangat mempengaruhi korelasi dan kekuatan masing-masing pohon dalam *random forest*. Dengan meningkatkan ukuran *m* maka akan meningkatkan korelasi dan kekuatan pohon-pohon, begitu juga sebaliknya. Terdapat tiga cara untuk mendapatkan nilai *m* yaitu sebagai berikut (Breiman & Cutler, 2003):

$$m = \begin{cases} m_1 = \frac{1}{2}\left|\sqrt{p}\right| \\ m_2 = \left|\sqrt{p}\right| \\ m_3 = 2 \times \left|\sqrt{p}\right| \end{cases} \tag{2.7}$$

Dimana *p* adalah total variabel.

Pada pembentukan *random forest* digunakan nilai *Gini Index* untuk menentukan split yang nantinya akan digunakan sebagai *root*/*node*. Berikut rumus untuk mencari nilai *Gini Index*:

$$Gini(S) = 1 - \sum_{i=1}^{k} p_i^2 \tag{2.8}$$

Nilai p_i adalah probabilitas dari *Gini Index* (S) milik kelas i. Adapun k adalah banyaknya nilai atribut yang termasuk ke dalam suatu kelas berdasarkan atribut data. Setelah menghitung nilai *Gini Index* (S), langkah berikutnya adalah menghitung nilai *GiniGain*. Fungsi Si adalah partisi dari S yang disebabkan oleh atribut A.

$$GiniGain(S) = Gini(S) - Gini(A,S) = Gini(S) - \sum_{i=1}^{n} \frac{|S_i|}{|S|} Gini(S_i) \tag{2.9}$$

Dalam menentukan pengklasifikasi, *random forest* melakukan sistem voting suara terbanyak dari pohon-pohon yang dibentuk. $(X_1, Y_1), \ldots, (X_n, Y_n)$ merupakan data latih dan dilambangkan dalam D_n dan variabel acak Z digunakan untuk menghitung sebuah keputusan. Maka diperoleh fungsi arbiter pengklasifikasi acak dalam bentuk $g_n(X, Z, D_n)$ yang kemudian dapat disingkat menjadi $g_n(X, Z)$. Maka dapat didefinisikan pengambilan suara mayoritas di antara pengklasifikasi acak yang diperoleh dengan mengasumsikan $Z^m = (Z_1, \ldots, Z_m)$ independen dan mengikuti kondisi pada *X*, *Y*, dan D_n, maka pemungutan suara dengan voting diperoleh sebagai berikut (Biau et al., 2008):

$$g_n^{(m)}(x, Z^m, D_n) = \begin{cases} 1 & Jika \frac{1}{m}\sum_{j=1}^{m} g_n(x, Z_j, D_n) \geq \frac{1}{2} \\ 0 & Sebaliknya \end{cases} \tag{2.10}$$

Pada setiap iterasi yang dilakukan dalam *random forest* terdapat 1/3 dari *dataset* yang digunakan tidak termuat dalam *bootstrap* atau yang disebut juga

dengan *Out-of-Bag* (OOB). OOB nantinya akan menjadi data validasi pada pohon yang bersesuaian atau yang tidak digunakan untuk membangun pohon tersebut.

Out of bag (OOB) *error* adalah metode untuk mengukur galat prediksi dalam random forest, *boosted decision trees* dan model *machine learning* lainnya yang menggunakan *bootstrap aggregating* (*bagging*). Dalam menghitung OOB pada *random forest* dapat dilakukan dengan langkah langkah berikut:

1. Lakukan prediksi terhadap setiap data OOB pada pohon yang bersesuaian. Data OOB (*out of bag*) adalah data yang tidak termuat dalam contoh *bootstrap*.
2. Secara rata-rata, setiap amatan gugus data asli akan menjadi data OOB sebanyak sekitar 36% dari banyak pohon. Oleh karena itu, pada langkah 1, masing-masing amatan gugus data asli mengalami prediksi sebanyak sekitar sepertiga kali dari banyaknya pohon. Jika *a* adalah sebuah amatan dari gugus data asli, maka hasil prediksi *random forest* terhadap a adalah gabungan dari hasil prediksi setiap kali a menjadi data OOB.
3. *Error* OOB dihitung dari proporsi misklasifikasi hasil prediksi *random forest* dari seluruh amatan gugus data asli.

Selain pembahasan algoritma *random forest* dan penghitungan OOB *error*, pada subbab ini akan dibahas juga berapa banyak pohon yang harus dibuat dalam memodelkan *random forest*. Pada penelitian "*How Many Trees in a Random Forest*?" dijelaskan bahwa hampir tidak adanya literatur yang menjelaskan berapa banyak pohon yang harus dibangun dalam pembuatan model *machine learning* dengan *Random Forest* untuk mendapatkan hasil yang optimal (Oshiro et al., 2012). Penelitian tersebut melakukan uji coba terhadap 29 *dataset* dengan jumlah "pohon" sebanyak 2, 4, 8, …, 4096. Kemudian hasil dari uji coba tersebut dianalisis dan disimpulkan bahwa dari "pohon" sejumlah 128 tidak ada perbedaan yang signifikan dengan penggunaan "pohon" sejumlah 256, 512, 1024, 2048 dan 4096.

Dengan demikian dapat disimpulkan berdasarkan penelitian tersebut, bahwa penambahan jumlah "pohon" yang dibangun dalam *random forest* tidak menjanjikan bertambahnya kinerja model dibandingkan dengan menggunakan jumlah "pohon" yang lebih sedikit, melainkan hanya menambah lama waktu komputasi. Oleh sebab itu berdasarkan uji coba yang telah dilakukan pada

penelitian tersebut banyaknya pohon yang dibentuk dalam membangun model *random forest* berkisar antara 64 dan 128 “pohon”. Dengan jumlah “pohon” ini maka dapat memperoleh keseimbangan pada area di bawah kurva, waktu pemrosesan dan penggunaan memori (Oshiro et al., 2012).

Dalam analisis model *random forest* yang telah dibuat, dapat dilakukan juga penghitungan *feature importance*. *Feature importance* merupakan suatu cara untuk mengukur tingkat kepentingan suatu fitur dengan menghitung berapa kali fitur tersebut muncul dalam kelompok pohon keputusan. Semakin besar nilai *feature importance* semakin penting pula variabel tersebut. Salah satu cara untuk menghitung suatu *feature importance* dapat menggunakan *Mean Decrease Impurity* (MDI) atau yang bisa juga disebut dengan *Mean Decrease Gini* (MDG). *Mean Decrease Gini* (MDG) merupakan salah satu ukuran tingkat kepentingan (*feature importance*) peubah penjelas yang dihasilkan oleh metode *random forest* (Dewi et al., 2012). Jika terdapat p peubah penjelas dengan h = (1, 2, …, p) maka MDG mengukur kepentingan peubah penjelas X_h dengan cara:

$$MDG_h = \frac{1}{k}\sum_t [d(h,t)I(h,t)] \tag{2.11}$$

Dimana: $d(h,t)$: besar penurunan indeks *Gini* untuk peubah penjelas x_h pada simpul t.

$I(h,t)$: $\begin{cases} 1; X_h\ memilih\ simpul\ t \\ 0; selainnya \end{cases}$

k : banyaknya pohon dalam *Random Forest*.

E. Evaluasi *Machine Learning*

Evaluasi *machine learning* untuk mendapatkan akurasi atau ketepatan *machine learning* dalam melakukan klasifikasi menggunakan *confusion matrix*. *Confusion matrix* adalah tabel yang menyatakan klasifikasi jumlah data uji yang benar dan jumlah data uji yang salah (Normawati & Prayogi, 2021). *Confusion matrix* merupakan alat ukur dalam bentuk matriks 2x2, yang digunakan untuk mendapatkan akurasi dari klasifikasi pada setiap data kelas lulus dan tidak lulus. Setiap kelas yang diprediksi memiliki empat kemungkinan *output* yang berbeda, yaitu *true positive* (TP) dan *true negative* (TN), yang menunjukkan klasifikasi yang benar. Jika output yang diprediksi positif sedangkan nilai asli

negatif itu disebut *false positive* (FP), dan jika *output* yang diprediksi negatif sedangkan nilai asli positif, itu disebut *false negative* (FN). Berikut ini dalam Tabel 2.7 adalah bentuk *confusion matriks* seperti yang dijelaskan di atas.

Tabel 2.7 *Confusion matrix*

	Nilai Sebenarnya		
Nilai Prediksi		*TRUE*	*FALSE*
	TRUE	*True Positive* (TP)	*False Positive* (FP)
	FALSE	*False Negative* (FN)	*True Negative* (TN)

- *True positives* adalah jumlah *record* positif yang diklasifikasikan sebagai positif.
- *False positives* adalah jumlah *record* negatif yang diklasifikasikan sebagai positif.
- *False negatives* adalah jumlah *record* positif yang diklasifikasikan sebagai negatif.
- *True negatives* adalah jumlah *record* negatif yang diklasifikasikan sebagai negatif.

Dengan *confusion matrix* maka dapat dihitung berbagai *performance metrics* untuk mengukur kinerja model yang telah dibuat. *Performance metrics* yang sering digunakan antara lain *precision*, *recall*, *accuracy* dan *F-Measure*.

- *Accuracy* menggambarkan seberapa akurat model yang telah dibuat dalam mengklasifikasikan data dengan benar.

$$Accuracy = \frac{TP+TN}{TP+TN+FN+FP} \quad (2.12)$$

- *Recall* menggambarkan keberhasilan model dalam menemukan kembali sebuah informasi.

$$Recall = \frac{TP}{TP+FN} \quad (2.13)$$

- *Precission* menggambarkan tingkat keakuratan antara data yang diminta dengan hasil prediksi yang diberikan oleh model.

$$Precission = \frac{TP}{TP+FP} \quad (2.14)$$

- F-*Measure* adalah penghitungan yang menggabungkan nilai *recall* dan nilai *precission*. Nilai *recall* dan *precision* dalam situasi tertentu memiliki prediksi yang berbeda. Ukuran yang menunjukkan timbal balik antara *recall* dan *precision* adalah *F-Measure* adalah bobot rata-rata dari harmonik dan *recall* dan *precission*

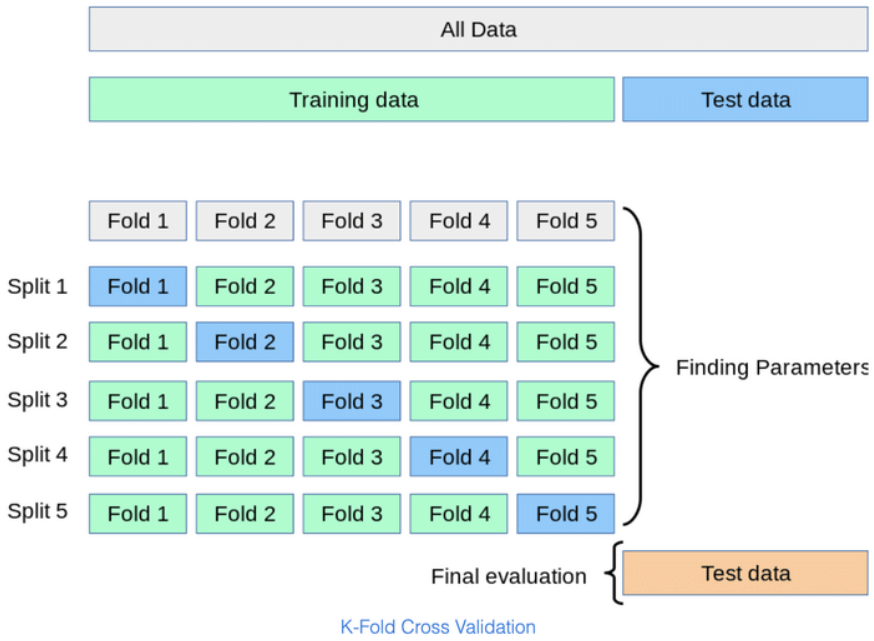
$$\text{F-}Measure = 2 \times \frac{precission \times recall}{precission + recall} \quad (2.15)$$

Keterangan: TP = *True Positives*, TN = *True Negatives*, FP = *False Positives*, FN = *False Negatives*

F. Uji Validitas *Machine Learning*

Metode yang digunakan dalam melakukan uji validitas *machine learning* adalah *Cross Validation*. *Cross-validation* adalah metode statistik untuk mengevaluasi dan membandingkan algoritma pembelajaran dengan membagi data menjadi dua segmen (Refaeilzadeh et al., 2016). Segmen data yang dibagi menjadi dua yaitu data latih dan data validasi/*test*. Model atau algoritma *machine learning* nantinya akan dilatih menggunakan data latih dan kemudian akan divalidasi oleh data validasi atau data *test*. Terdapat beberapa teknik dari *cross validation* salah satunya adalah K-*Fold Cross Validation*. Biasanya K-*Fold Cross Validation* digunakan karena dapat mengurangi waktu komputasi dengan tetap menjaga keakuratan estimasi (Permana & Effendi, 2018).

K-*Fold Cross Validation* (K-Fold CV) adalah salah satu dari jenis pengujian *cross validation* yang berfungsi untuk menilai kinerja proses sebuah metode algoritma dengan membagi sampel data secara acak dan mengelompokkan data tersebut sebanyak nilai K (Cahyanti et al., 2020). *K-Fold* CV akan membagi data menjadi sejumlah k bagian/*fold* yang mana pada setiap fold akan digunakan sebagai data latih dan data *test*. Sebagai contoh K-Fold CV dengan k=5. Datasets yang dimiliki akan dibagi menjadi 5 bagian/fold. Pada iterasi pertama, bagian/*fold* pertama akan menjadi data *test* untuk menguji model *machine learning* yang telah dibuat dan bagian/*fold* sisanya akan menjadi data latih untuk melatih model ML. Iterasi akan terus diulang sampai semua *fold* telah digunakan sebagai data *test*.



Sumber: https://www.researchgate.net/

Gambar 2.8 Ilustrasi K-*Fold Cross Validation*

G. Kelulusan Mata Kuliah

Kelulusan mata kuliah merupakan hal yang penting bagi tiap mahasiswa, terutama kelulusan pada mata kuliah wajib. Mata kuliah wajib adalah mata kuliah yang wajib diikuti oleh setiap mahasiswa. Di Universitas Negeri Jakarta (UNJ) kelulusan pada mata kuliah wajib ini akan sangat berpengaruh pada setiap mahasiswa terutama bagi mahasiswa yang akan menyelesaikan masa studinya, karena mahasiswa diwajibkan lulus mata kuliah wajib sebagaimana Peraturan Rektor UNJ Nomor 7 Tahun 2018 tentang peraturan akademik Universitas Negeri Jakarta. Selain kelulusan mata kuliah wajib terdapat juga kelulusan pada mata kuliah prasyarat, di mana mahasiswa tidak akan bisa mengambil mata kuliah lanjutan di semester selanjutnya bila mata kuliah prasyarat belum dinyatakan lulus.

Kriteria nilai yang menjadi standar kelulusan mata kuliah pada Universitas Negeri Jakarta (UNJ) sebagaimana yang diatur pada Peraturan Rektor Nomor 7 tahun 2018 adalah sebagai berikut.

Tabel 2.8 Kriteria nilai Universitas Negeri Jakarta

Tingkat Penguasaan	Nilai	Bobot	Predikat
86 – 100 %	A	4	Sangat Baik
81 – 85 %	A-	3,7	
76 – 80 %	B+	3,3	Baik
71 – 75 %	B	3	
66 – 70 %	B-	2,7	
61 – 65 %	C+	2,3	Cukup
56 – 60 %	C	2	
51 – 55 %	C-	1,7	Tidak Lulus
46 – 50 %	D	1	
< 46 %	E	0	

Sebagaimana tabel di atas apabila mahasiswa mendapatkan nilai C- atau di bawahnya, maka mahasiswa tersebut haruslah mengulang mata kuliah tersebut di semester yang akan datang nantinya. Selain dari itu mahasiswa tidak perlu mengulang mata kuliah tersebut.

Banyak faktor-faktor yang mempengaruhi dalam kelulusan mahasiswa dalam suatu mata kuliah. Salah satunya adalah faktor kehadiran dan nilai-nilai yang diperoleh selama menempuh mata kuliah tersebut seperti, nilai tugas, kuis, UTS dan UAS. Kehadiran mahasiswa dalam kegiatan tatap muka selama perkuliahan menjadi salah satu faktor yang mempengaruhi kelulusan mata kuliah. Hal ini dikarenakan apabila mahasiswa tidak hadir dalam kegiatan tatap muka selama lebih dari tiga kali maka, mahasiswa tersebut akan tidak akan diluluskan dalam mata kuliah tersebut.

Kelulusan mahasiswa dalam mata kuliah juga menjadi indikator keberhasilan belajar mahasiswa. Keberhasilan belajar adalah suatu perubahan tingkah laku yang relatif permanen yang terjadi sebagai hasil latihan dan pengalaman, dengan kata lain suatu kegiatan atau usaha sadar yang menghasilkan suatu perubahan, berupa sesuatu yang baru, yang langsung terlihat atau tersembunyi. Perubahan yang dihasilkan dari pembelajaran ini dapat berupa perubahan pengetahuan, pemahaman, sikap dan perilaku, keterampilan, kompetensi dan kemampuan, serta perubahan aspek lainnya, tergantung pada pembelajaran individu setiap mahasiswa (Santoso & Subagyo, 2017). Mahasiswa yang lulus dalam mata kuliah dianggap telah berhasil dalam proses belajar.

Teori-teori yang telah dijabarkan sebelumnya akan dijadikan acuan dalam melakukan penelitian ini, dimana sebelum dilakukan pemodelan *machine learning* akan dilakukan data *pre-processing* untuk mendapatkan *datasets* yang bersih. Struktur data pohon dibutuhkan sebagai struktur data dasar dalam membangun pohon keputusan dengan metode CART (*Classification Tree and Regression Tree*). Sebelum pembentukan pohon keputusan, terlebih dahulu dilakukan *bootsrapping* dan *random feature selection* terhadap *datasets*. Proses ini diulang sebanyak *k* kali, sehingga terbentuk model *random forest*. Dari model *random forest* tersebut, kemudian akan dievaluasi menggunakan *confusion matrix* dan diujikan validitasnya dengan *k-fold cross validation*.

BAB III
METODOLOGI PENELITIAN

Dalam melaksanakan penelitian ini, penulis menggunakan alat dan tahapan penelitian sebagai berikut ini:

A. Alat Penelitian

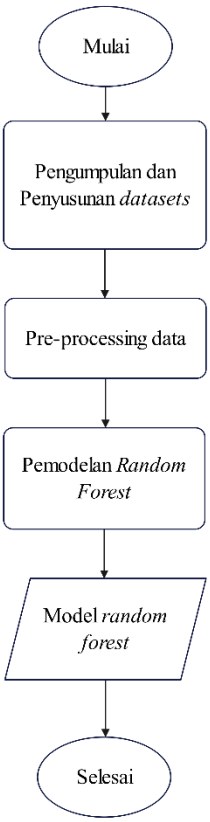
Spesifikasi perangkat lunak dan perangkat keras yang digunakan penulis dalam melakukan penelitian ini adalah sebagai berikut:

1. Perangkat Lunak:
   - Sistem Operasi : *Windows 11 Home Single Language* 64-*bit*.
   - Aplikasi Perekapan Data : *Microsoft Excel* 365.
   - Bahasa Pemrograman : *Python*.
   - Antarmuka Bahasa Pemrograman : *Google Collaboration* dan *Jupyter Notebook*.
   - *Library* Bahasa Pemrograman *: Numpy, Pandas, Seaborn, Matplotlib, Sklearn, imblearn, json, networkx*.
2. Perangkat Keras

   Perangkat keras yang digunakan berupa laptop dengan spesifikasi sebagai berikut:
   - *Processor* *: Intel Core* i7-1065G7 @1.30 GHz.
   - *Memory* : 16 GB.
   - VGA : *Intel Iris Plus Graphics* dan NVIDIA *GeForce MX250*

B. Tahapan Penelitian

Tahapan dalam penelitian ini dilakukan dengan melalui beberapa proses. Proses-proses tersebut dapat dilihat pada gambar 3.1 sebagai berikut.



Gambar 3.1 Tahapan penelitian

1. Pengumpulan dan Penyusunan *datasets*

Datasets yang digunakan dalam penelitian ini merupakan data sekunder yang berasal dari tiga sumber berbeda. Sumber pertama merupakan dari salah satu dosen program studi pendidikan matematika Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Negeri Jakarta. Kemudian sumber yang kedua berasal dari data presensi mahasiswa selama mengikuti mata kuliah tersebut. dan sumber terakhir berasal dari data biodata umum mahasiswa yang dari PDDIKTI.

Data yang diperoleh dari dosen sebagai sumber data pertama berupa data hasil belajar mahasiswa atau data nilai-nilai mahasiswa yang diperoleh dari mata kuliah Kalkulus Integral yang diampu oleh salah satu dosen program studi Pendidikan Matematika FMIPA Universitas Negeri Jakarta pada semester 112 dan 114. Data yang diperoleh berupa *file* dengan ekstensi *xlsx* sebanyak lima buah berkas (*file*): dua *file* merupakan *file* nilai kalkulus integral pada semester 112 dan tiga *file* lainnya nilai kalkulus integral pada

semester 114, yang kemudian direkapitulasi dalam satu *file*. Setelah direkapitulasi dan dibuat dalam satu *file* maka dapat diketahui, data yang diperoleh berjumlah 188 baris dan 18 kolom. Adapun kolom-kolom dari data yang telah di satukan pada satu *file* sebagai berikut:

Tabel 3.1 Data Nilai Mahasiswa

No	Nama Kolom	Penjelasan	Tipe Data
1	No	Nomor urut	Numerik
2	No Reg	Nomor registrasi mahasiswa	Numerik
3	Nama	Inisial nama mahasiswa	Kategori
4	Prodi	Program studi mahasiswa	Kategori
5	Angkatan	Angkatan masuk mahasiswa	Kategori
6	TgsK1	Nilai tugas kelompok mahasiswa pertama	Numerik
7	TgsK2	Nilai tugas kelompok mahasiswa kedua	Numerik
8	TgsK3	Nilai tugas kelompok mahasiswa ketiga	Numerik
9	TgsK4	Nilai tugas kelompok mahasiswa keempat	Numerik
10	TgsK5	Nilai tugas kelompok mahasiswa kelima	Numerik
11	TgsK6	Nilai tugas kelompok mahasiswa keenam	Numerik
12	NRK	Nilai rata-rata kelompok mahasiswa	Numerik
13	Tgs Individu	Nilai Keaktifan Mahasiswa	Numerik
14	Tgs1	Tugas Pertama	Numerik
15	UTS	Nilai ujian tengah semester mahasiswa	Numerik
16	UAS	Nilai ujian akhir semester mahasiswa	Numerik
17	NA	Nilai akhir mahasiswa	Numerik
18	NL.HRF	Nilai akhir mahasiswa dalam kategori huruf	Kategori

Untuk data kedua merupakan data presensi mahasiswa yang diperoleh dari admin program studi. Data presensi yang diambil merupakan presensi mahasiswa selama mengikuti perkuliahan mata kuliah kalkulus integral pada kelas yang sama seperti pada data nilai yang telah diperoleh sebelumnya. Sama seperti data nilai yang telah diperoleh sebelumnya, data presensi pada awalnya juga terbagi menjadi lima buah *file* dengan ekstensi *xlsx* yang kemudian penulis rekap sehingga diperoleh jumlah kehadiran dan persentase kehadiran mahasiswa selama masa perkuliahan tersebut. Adapun data yang penulis dapatkan berjumlah sama dengan data nilai yang diperoleh sebelumnya, dengan kolom-kolom sebagai berikut:

Tabel 3.2 Data Presensi Mahasiswa

No	Nama Kolom	Penjelasan	Tipe Data
1	No	Nomor urut	Numerik
2	No Reg	Nomor registrasi mahasiswa	Numerik
3	Kehadiran	Jumlah kehadiran mahasiswa	Numerik
4	Kehadiran (persentase)	Persentase kehadiran mahasiswa	Numerik

Pada data yang ketiga, diperoleh dari peladen Pangkalan Data Pendidikan Tinggi (PDDIKTI) yang diakses secara *online* dengan memasukkan kata kunci nomor registrasi mahasiswa yang telah diperoleh pada data-data sebelumnya. Data yang berhasil dihimpun merupakan data yang bersifat terbuka dan bisa diakses oleh siapa saja. Berikut kolom-kolom dari data yang telah penulis dapatkan dari peladen PDDIKTI.

Tabel 3.3 Data PDDIKTI

No	Nama Kolom	Penjelasan	Tipe Data
1	No	Nomor urut	Numerik
2	No Reg	Nomor registrasi mahasiswa	Numerik
3	Jenis Kelamin	Jenis kelamin mahasiswa	Kategori
4	SKS yang telah ditempuh	Jumlah seluruh SKS yang telah ditempuh pada semester sebelumnya	Numerik
5	Mengulang matkul kalin	Pernah atau tidak sebelumnya mengambil mata kuliah kalkulus integral	Kategori

Dari ketiga data yang telah diperoleh sebelumnya, maka dilakukan penggabungan data dengan nomor registrasi sebagai kolom acuannya. Setelah semua data telah digabungkan dilakukan perubahan nama pada kolom “Nama” menjadi hanya inisial nama mahasiswa saja dan pada kolom “No Reg” dilakukan penyensoran pada tiga angka paling belakang. Setelah dilakukan penggabungan data, dihasilkan sebuah *datasets* padu yang terdiri dari 188 baris dan 23 kolom. Adapun kolom-kolom pada *datasets* yang telah diperoleh sebagai berikut:

Tabel 3.4 *Datasets*

No	Nama Kolom	Penjelasan	Tipe Data
1	No	Nomor urut	Numerik
2	No Reg	Nomor registrasi mahasiswa	Numerik
3	Nama	Inisial nama mahasiswa	Kategori
4	Jenis Kelamin	Jenis kelamin mahasiswa	Kategori
5	SKS yang telah ditempuh	Jumlah seluruh SKS yang telah ditempuh pada semester sebelumnya	Numerik
6	Prodi	Program studi mahasiswa	Kategori
7	Angkatan	Angkatan masuk mahasiswa	Kategori
8	Mengulang matkul kalin	Pernah atau tidak sebelumnya mengambil mata kuliah kalkulus intgeral	Kategori
9	Kehadiran	Jumlah kehadiran mahasiswa	Numerik
10	Kehadiran (presentase)	Persentase kehadiran mahasiswa	Numerik
11	TgsK1	Nilai tugas kelompok mahasiswa pertama	Numerik
12	TgsK2	Nilai tugas kelompok mahasiswa kedua	Numerik
13	TgsK3	Nilai tugas kelompok mahasiswa ketiga	Numerik
14	TgsK4	Nilai tugas kelompok mahasiswa keempat	Numerik
15	TgsK5	Nilai tugas kelompok mahasiswa kelima	Numerik
16	TgsK6	Nilai tugas kelompok mahasiswa keenam	Numerik
17	NRK	Nilai rata-rata kelompok mahasiswa	Numerik
18	Tgs Individu	Nilai Keaktifan Mahasiswa	Numerik
19	Tgs1	Tugas Pertama	Numerik
20	UTS	Nilai ujian tengah semester mahasiswa	Numerik
21	UAS	Nilai ujian akhir semester mahasiswa	Numerik
22	NA	Nialai akhir mahasiswa	Numerik
23	NL.HRF	Nilai akhir mahasiswa dalam kategori huruf	Kategori

Adapun pratinjau dari *datasets* dapat dilihat pada gambar 3.2 sementara *datasets info* dapat dilihat pada gambar 3.3 berikut.

	No	No Reg	Nama	Jenis Kelamin	SKS yang telah ditempuh	Prodi	Angkatan	Mengulang Matkul Kalin	Kehadiran	Kehadiran (presentase)	...	TgsK4	TgsK5	TgsK6	NRK	Tgs Individu	Tgs1	UTS	UAS	NA	NL.HRF
0	1	1313619***	PD	L	21	Ilmu Komputer	2019	Tidak	13	100.000000	...	NaN	NaN	NaN	89.333333	2	99.333333	95.0	76.785714	71.335714	B
1	2	1313619***	AMA	L	21	Ilmu Komputer	2019	Tidak	12	92.307692	...	NaN	NaN	NaN	93.333333	3	100.000000	87.5	81.964286	70.839286	B-
2	3	1313619***	IAK	L	21	Ilmu Komputer	2019	Tidak	13	100.000000	...	NaN	NaN	NaN	90.000000	5	100.000000	45.0	38.392857	45.017857	E
3	4	1313619***	CPS	P	21	Ilmu Komputer	2019	Tidak	13	100.000000	...	NaN	NaN	NaN	100.000000	2	100.000000	95.0	90.059524	75.517857	B
4	5	1313619***	EJR	L	21	Ilmu Komputer	2019	Tidak	13	100.000000	...	NaN	NaN	NaN	93.333333	1	98.333333	90.0	78.630952	70.255286	B-

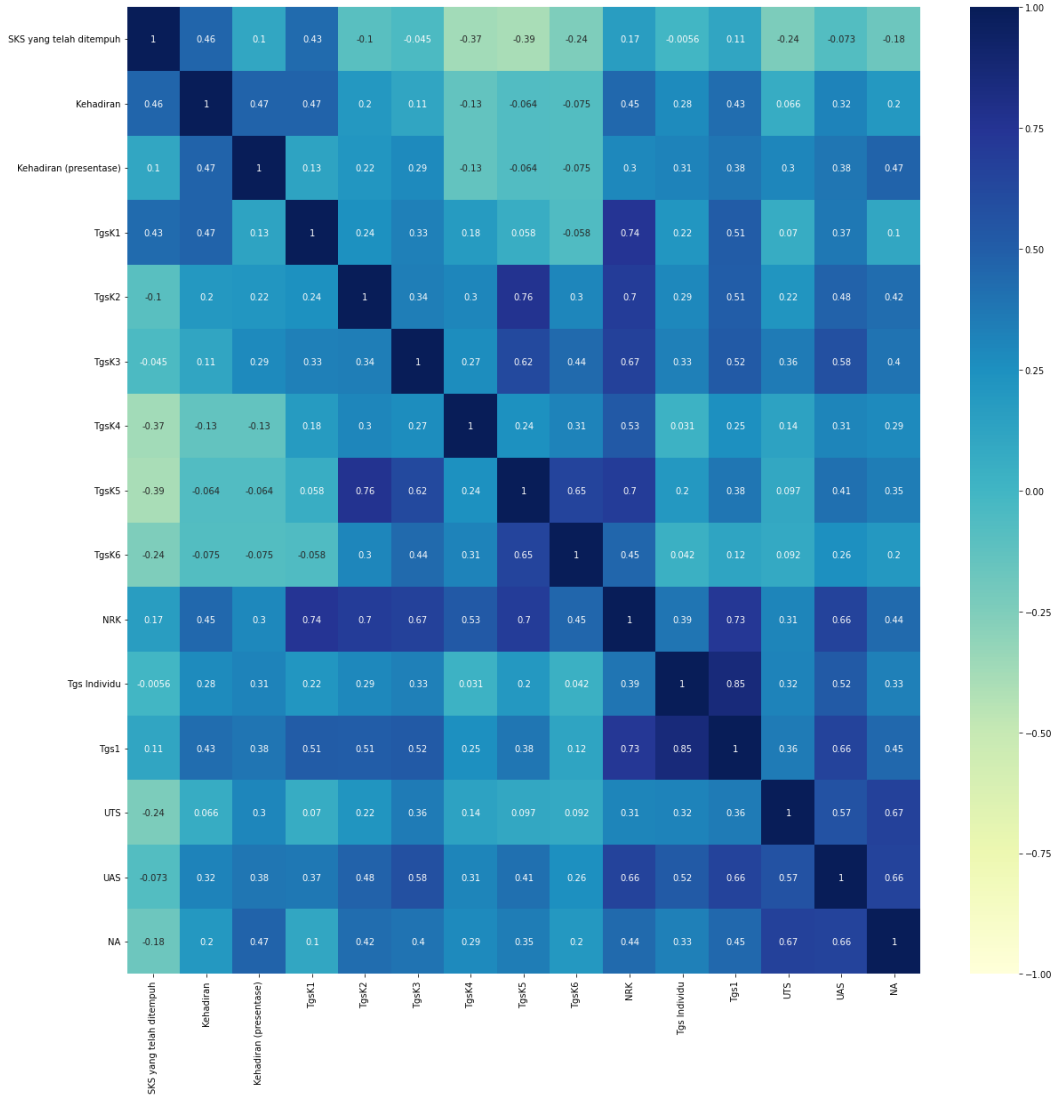
5 rows × 23 columns

Gambar 3.2 Pratinjau *Datasets*

```
RangeIndex: 188 entries, 0 to 187
Data columns (total 23 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   No                       188 non-null    int64
 1   No Reg                   188 non-null    object
 2   Nama                     188 non-null    object
 3   Jenis Kelamin            188 non-null    object
 4   SKS yang telah ditempuh  188 non-null    int64
 5   Prodi                    188 non-null    object
 6   Angkatan                 188 non-null    object
 7   Mengulang Matkul Kalin   188 non-null    object
 8   Kehadiran                188 non-null    int64
 9   Kehadiran (presentase)   188 non-null    float64
 10  TgsK1                    188 non-null    int64
 11  TgsK2                    188 non-null    int64
 12  TgsK3                    188 non-null    int64
 13  TgsK4                    82 non-null     float64
 14  TgsK5                    82 non-null     float64
 15  TgsK6                    82 non-null     float64
 16  NRK                      188 non-null    float64
 17  Tgs Individu             188 non-null    int64
 18  Tgs1                     188 non-null    float64
 19  UTS                      188 non-null    float64
 20  UAS                      188 non-null    float64
 21  NA                       188 non-null    float64
 22  NL.HRF                   188 non-null    object
dtypes: float64(9), int64(7), object(7)
memory usage: 33.9+ KB
```

Gambar 3.3 ***Datasets info***

Dari *dataset* yang telah diperoleh lalu dilakukan uji korelasi. Analisis dari uji korelasi yang telah dilakukan dengan mengecualikan variabel atau kolom “No”, “Nama”, “No Reg” dan “NL.HRF”. Untuk mengetahui korelasi antar kolom atau parameter pada *datasets* yang telah dimiliki maka, dilakukan uji korelasi semua parameter pada *datasets* terhadap parameter NA (Nilai Akhir) menggunakan pemrograman *python* yang lalu divisualisasikan pada *heatmap* dengan *library seaborn*. Dengan uji korelasi yang dilakukan maka dihasilkan sebagai gambar 3.4 sebagai berikut.



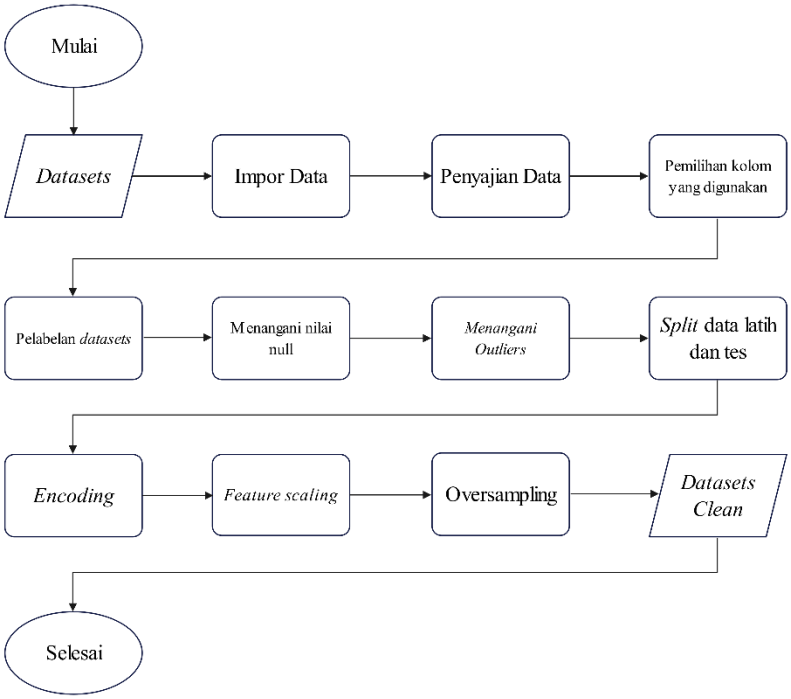
Gambar 3.4 Korelasi Antar Kolom

Dalam gambar 3.4 di atas maka, dapat diketahui bahwa korelasi parameter “NA” terhadap parameter lain memiliki nilai positif dan negatif. Dapat dilihat pada gambar korelasi parameter “NA” dengan parameter “TgsK1” sebesar 0.11, parameter “NA” dengan parameter “TgsK2” sebesar 0.32, parameter “NA” dengan parameter “TgsK3” sebesar 0.49, parameter “NA” dengan parameter “TgsK4” sebesar 0.27, parameter “NA” dengan parameter “TgsK5” sebesar 0.33, parameter “NA” dengan parameter TgsK6 sebesar 0.16, parameter “NA” dengan parameter “Tgs individu” sebesar 0.31, parameter “NA” dengan parameter “Tgs1” sebesar 0.56, parameter “NA” dengan parameter “UTS” sebesar 0.64, parameter “NA” dengan parameter

"UAS" sebesar 0.88, sementara pada parameter "NA" dengan parameter "SKS yang telah ditempuh" sebesar -0.18 .

2. Tahapan *Pre-processing* Data

Sebelum melakukan pemodelan *Random Forest*, *datasets* yang telah disusun sebelumnya terlebih dahulu akan diolah dan dilakukan *pre-processing* data. Alur *pre-processing* data yang dilakukan oleh penulis dapat dilihat pada diagram berikut.



Gambar 3.5 Alur *pre-processing* data

Tahapan *pre-processing* data yang dilakukan oleh penulis sesuai dengan gambar 3.5 antara lain sebagai berikut:

2.1. *Datasets* yang telah disusun akan diubah format *file*-nya yang sebelumnya berformat *xlxs* diubah menjadi format CSV (*Comma Separated Values*). *Datasets* tersebut kemudian akan dimuat kedalam program *python* pada Google Colab menggunakan *library pandas*.

2.2. *Datasets* kemudian akan disajikan dan dianalisis terlebih dahulu untuk mendapatkan kesimpulan dan karakteristik pada *datasets*. Dari tahapan

ini, nantinya akan diketahui apakah terdapat nilai *null* dan *outliers* pada *datasets* dan apakah *datasets* sudah *balance* atau belum.

2.3. *Datasets* yang telah dianalisis, kemudian akan diseleksi kolom-kolom mana yang dibutuhkan dan kolom mana yang tidak diikutsertakan dalam pemodelan *random forest*. Kolom-kolom yang tidak diikutsertakan dalam pemodelan *random forest* adalah sebagai berikut:

- Kolom "No." => Merepresentasikan nomor urut.
- Kolom "No. Reg" => Merepresentasikan nomor registrasi mahasiswa.
- Kolom "Nama" => Merepresentasikan inisial nama mahasiswa.
- Kolom "Kehadiran" => Merepresentasikan jumlah kehadiran mahasiswa. Kolom ini dibuang karena ada kolom yang sama yang merepresentasikan kehadiran mahasiswa tetapi dalam bentuk persentase.
- Kolom “NRK” => Merepresentasikan nilai rata-rata tugas kelompok mahasiswa. Kolom ini dibuang karena sudah ada kolom TgsK1 – TgsK6 yang digunakan dalam pembuatan model.
- Kolom "NA" => Merepresentasikan nilai akhir. Kolom ini dihilangkan karena sudah ada kolom nilai huruf sebagai representasi lain nilai akhir mahasiswa.
- Kolom "Mengulang Matkul Kalin" => Merepresentasikan apakah mahasiswa tersebut pernah mengulang atau tidak mata kuliah kalkulus integral, kolom ini buang karena isi dari kolom ini hanya satu jawaban yaitu "Tidak" semua.

2.4. Tahapan selanjutnya adalah pelabelan *datasest*. Label *datasets* dibuat berdasarkan kolom “NL.HRF”. Kolom tersebut kemudian akan dijadikan acuan untuk membuat kolom baru yang nantinya akan menjadi *label datasets*. Kolom baru tersebut dibuat dengan nama “*predicate*” dan *value* kolom akan diisi berdasarkan predikat kriteria nilai yang menjadi standar kelulusan mata kuliah pada Universitas Negeri Jakarta (UNJ) sebagaimana yang diatur pada Peraturan Rektor Nomor 7 tahun 2018 tentang Peraturan Akademik Universitas Negeri

Jakarta. Setelah kolom *predicate* dibuat, kolom "NL.HRF" akan di-*drop* atau dihapus.

2.5 Tahapan berikutnya adalah menangani nilai *null* pada *datasets*. Nilai *null* pada *datasets* tidak akan dibuang, melainkan akan ditangani dengan cara merubah nilai *null* dengan sebuah nilai baru.

2.6. Selanjutnya adalah menangani *outliers*. Untuk mendeteksi *outliers*, penulis menggunakan diagram *boxplot* dan perhitungan jangkauan interkuartil. Nilai yang termasuk ke dalam *outlier* akan diganti dengan nilai rata-rata dari kolom dimana *outlier* tersebut berada.

2.7. Tahapan berikutnya adalah membagi *datasets* atau *split datasets*. *Datasets* dibagi menjadi data X sebagai *feature datasets* dan data Y sebagai *label datasets*. Data X dan Y kemudian akan dibagi menjadi data latih dan data tes dengan data latih sebesar 70% dan data tes sebesar 30%.

2.8. Tahap selanjutnya adalah tahap *encoding* terhadap data X yang memiliki tipe kategori pada data latih dan data tes. Dalam melakukan *encoding* menggunakan *library sklearn*. Metode *encoding* yang digunakan menyesuaikan dengan tipe data kategori tiap-tiap kolomnya. Apabila kolom termasuk ke dalam kategori nominal, maka menggunakan fungsi *OneHotEncoder()*, sementara untuk kolom yang termasuk ke dalam kategori ordinal, maka menggunakan fungsi *OrdinalEncoder()*.Pada tahap ini juga dilakukan label *encoding* terhadap data Y. Sama seperti tahap sebelumnya, label *encoding* menggunakan *library sklearn* dengan fungsi *LabelEncoder()*.

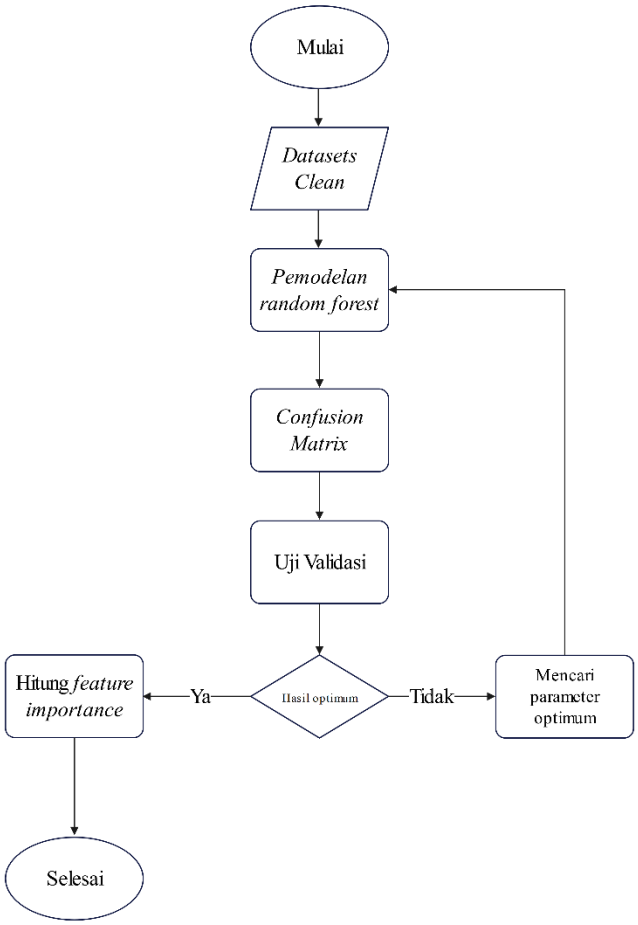
2.9. Tahap berikutnya adalah tahap *feature scaling* data numerik. Metode yang digunakan untuk melakukan *feature scaling* data numerik adalah normalisasi dengan metode *standar scaler*. Untuk melakukan *feature scaling* data numerik menggunakan *library sklearn* dengan menggunakan fungsi *StandardScaler()*.

2.10. Tahap selanjutnya adalah pemrosesan data dengan *imbalance* data menggunakan teknik *oversampling*. Hal ini dilakukan karena adanya ketidakseimbangan jumlah label pada data latih. Metode *oversampling*

yang digunakan adalah metode SMOTE (*Synthetic Minority Oversampling Technique*) dengan bantuan *library imblearn* dengan fungsi *SMOTE()*.

Setelah semua tahapan pada *pre-processing* selesai, maka akan didapatkan *datasets* bersih yang sudah siap untuk masuk ke dalam pemodelan *machine learning*.

3. Tahapan Pemodelan *Random Forest*

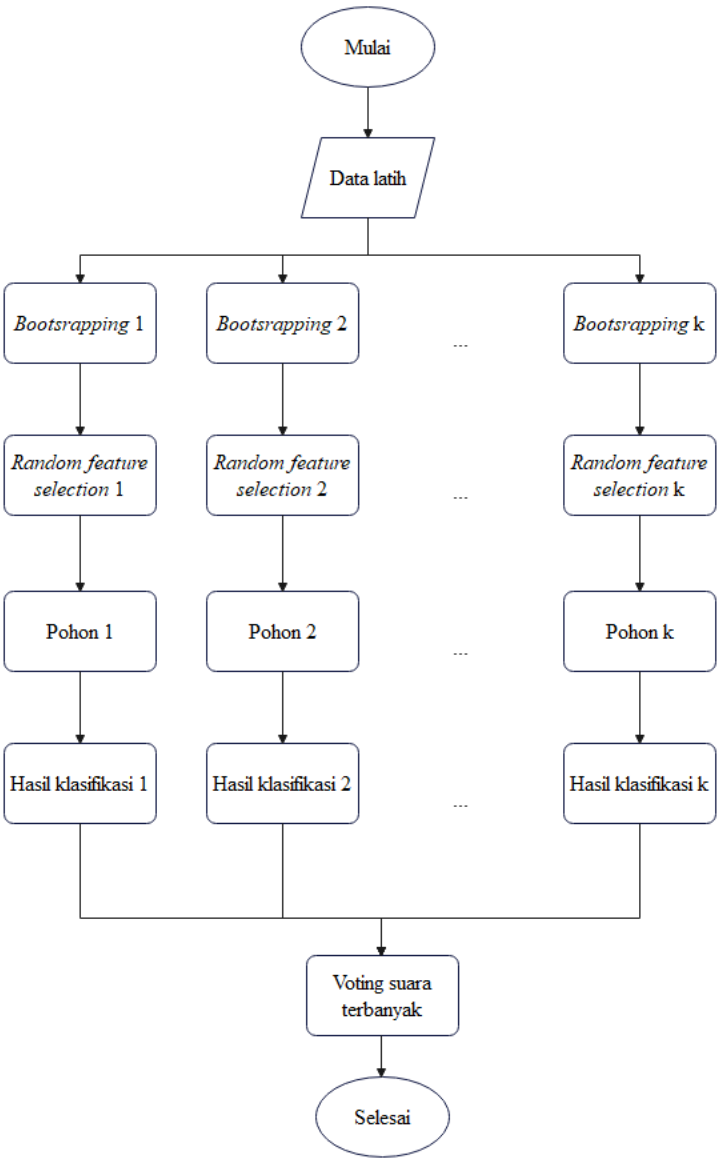


Gambar 3.6 Alur pemodelan *Random Forest*

Setelah diperoleh *datasets* yang bersih, dilakukan pemodelan *random forest* menggunakan *datasets* tersebut. Pemodelan *random forest* yang dilakukan sesuai dengan gambar 3.6 antara lain sebagai berikut:

3.1. Tahapan untuk memodelkan *random forest* dapat dilihat pada gambar 3.7. Berdasarkan gambar tersebut, tahap pertama yang dilakukan untuk memodelkan *random forest* adalah *bootsrapping*, kemudian dilanjutkan

dengan *random feature selection* pada data latih. Lalu dilanjutkan dengan membentuk pohon keputusan. Proses ini akan dilakukan sebanyak *k* kali, dimana *k* adalah banyaknya pohon yang akan dibentuk. Untuk mendapatkan hasil klasifikasi, akan dilakukan voting suara terbanyak untuk menentukan hasil dari klasifikasi.



Gambar 3.7 Ilustrasi *random forest*

3.2. Setelah model *random forest* terbentuk, selanjutnya dilakukan proses penghitungan *confusion matrix* menggunakan label data latih terhadap hasil prediksi data latih dan label data tes terhadap hasil prediksi data tes. Penghitungan *confusion matrix* dilakukan menggunakan *library*

sklearn dengan fungsi *confusion_matrix()*. Kemudian dilakukan analisis terhadap hasil perhitungan akurasi yang diperoleh dari kedua data tersebut, apakah model yang telah dibuat memiliki nilai akurasi yang baik atau malah terjadinya *overfitting* atau *underfitting*. Model mengalami *overfitting* apabila hasil akurasi terhadap data latih dan data tes memiliki selisih nilai yang besar. Sementara model mengalami *underfitting* apabila hasil akurasi terhadap data latih dan data tes memiliki nilai yang kecil. Setelah mengetahui hasil akurasi model terhadap data latih dan data tes, maka akan dilakukan penghitungan s*recall*, *precision* dan *f-measure*.

3.3. Setelah penghitungan *confusion matrix,* tahapan berikutnya adalah uji validitas model yang telah dibuat. Untuk menguji validitas model akan menggunakan *k-fold cross validation*, dengan jumlah *k* sebanyak 10. Dari hasil *k-fold cross validation* tersebut akan dihasilkan rata-rata akurasi beserta nilai dari simpangan bakunya.

3.4. Tahapan selanjutnya setelah melihat hasil validasi model adalah menilai apakah hasil dari uji validasi model sudah termasuk baik atau belum. Apabila belum, maka akan dilakukan pencarian parameter optimum dengan beberapa kombinasi parameter. Kombinasi parameter yang digunakan meliputi banyaknya pohon yang dibentuk dan jumlah fitur maksimal yang digunakan untuk membentuk suatu pohon. Banyaknya pohon yang akan diuji sebanyak 64 – 132 pohon, sementara jumlah fitur maksimal yang digunakan mengikuti rumus sebagai berikut.

$$m = \begin{cases} m_1 = \frac{1}{2}\left|\sqrt{p}\right| \\ m_2 = \left|\sqrt{p}\right| \\ m_3 = 2 \times \left|\sqrt{p}\right| \end{cases} \tag{3.1}$$

Dimana *p* adalah jumlah fitur dari data latih. Untuk menentukan kombinasi terbaik antara banyaknya pohon dengan jumlah fitur maksimal akan menggunakan nilai OOB (*Out of Bagging*) *error*. Dari seluruh nilai OOB *error* yang dihasilkan dengan berbagai kombinasi jumlah pohon dan jumlah fitur maksimal, maka selanjutnya akan dicari

nilai OOB *error* terkecil dari hasil tersebut. Kombinasi jumlah pohon dan jumlah fitur maksimal yang memiliki nilai OOB *error* terkecil ini merupakan kombinasi parameter yang optimum untuk membentuk model *random forest*.

3.5. Setelah diperoleh parameter optimum, maka akan dilakukan pemodelan ulang dari tahap awal pemodelan sampai uji validasi model. Apabila hasil dari validasi model masih kurang baik, maka akan dilakukan kembali pencarian parameter optimum dengan kombinasi yang berbeda. Sebaliknya, apabila uji validasi dinilai sudah baik, maka akan dilakukan analisis terhadap model terbaik yang telah didapatkan.

3.6. Dari hasil model yang telah didapatkan sebelumnya, dapat dilakukan penghitungan *feature importance* untuk mengetahui seberapa besar pengaruh suatu fitur terhadap hasil klasifikasi.

BAB IV
HASIL DAN PEMBAHASAN

Pada bab ini, akan dibahas sedikit penjelasan tentang program CART, program *random forest* yang telah dibuat dan hasil dari tahapan penelitian yang telah dijelaskan pada bab sebelumnya. Hasil yang diperoleh antara lain sebagai berikut:

A. Program CART

Program CART yang telah dibuat memiliki beberapa *method* atau fungsi penting yaitu sebagai berikut:

1. *Method gini_index* dan *impurity_node*

```
FUNCTION  gini_index(y):
    labels<-np.unique(y)
    gini<-0
    FOR label in labels:
        prob = len(y[y=label]) / len(y)
        gini += prob**2
    return 1 - gini
ENDFUNCTION
```

Gambar 4.1 ***Pseudocode method gini_index***

```
FUNCTION  impurity_node(y, y_left, y_right):
    prob_l<-len(y_left)  / len(y)
    prob_r<-len(y_right) / len(y)
    avg_child<-(prob_l * self.gini_index(y_left)) + (prob_r * self.gini_index(y_right))
    return self.gini_index(y) - avg_child
ENDFUNCTION
```

Gambar 4.2 ***Pseudocode method impurity_node***

Method gini_index merupakan *method* yang akan mengembalikan nilai *gini index* dari simpul yang akan dibuat. *Input* untuk *method gini_index*

merupakan *list* label atau data label pada simpul dan keluaran dari *method* ini adalah nilai *gini index*. Nilai *gini index* yang didapatkan akan digunakan untuk mendapatkan nilai impuritas simpul dengan *method impurity_node*. Nilai impuritas inilah yang akan menentukan simpul tersebut merupakan pemilah terbaik atau bukan. *Input* untuk *method impurity_node* adalah *list* label pada simpul *parent*, *list* label pada simpul *left child* dan *list* label pada simpul *right child*. Hasil keluaran dari *method* tersebut adalah nilai impuritas simpul.

2. *Method split_data*

```
FUNCTION  split_data(self, datasets, feature, threshold):
    left_data<-[]
    right_data<-[]
    FOR row in datasets:
      IF row[feature] <= threshold THEN
        left_data.append(row)
      ELSE
        right_data.append(row)
      ENDIF
    return np.array(left_data), np.array(right_data)
ENDFUNCTION
```

Gambar 4.3 *Pseudocode method split_data*

Method split_data digunakan untuk memilah data yang ada pada *parent node*. Data akan dipilah menjadi dua yang kemudian akan menjadi data pada simpul *left child* dan simpul *right child*. Pemilahan data dilakukan berdasarkan *input index* fitur dan *threshold*. Data akan masuk kedalam simpul *left child* apabila data pada fitur ke-*index* yang sesuai dengan *input index* lebih kecil atau sama dengan *threshold*, selain itu data akan masuk ke dalam simpul *right child*.

3. *Method make_tree*

```
FUNCTION  make_tree(self, datasets, depth):
    x, y<-datasets[:, :-1], datasets[:, -1]
    num_sample, num_feature<-x.shape
    current_depth<-depth
    IF num_sample >= self.min_sample_split and self.check_depth(current_depth) THEN
      best_feature<-0
      best_threshold<-0
      max_impurity<-0
      FOR features in range(num_feature):
        feature_values<-datasets[:, features]
        thresholds<-np.unique(feature_values)
        FOR threshold in thresholds:
          feature_type<-"numeric"
          left_data, right_data<-self.split_data(datasets, features, threshold, feature_type)
          IF right_data.size != 0 and left_data.size != 0 THEN
            y<-datasets[:, -1]
            y_l<-left_data[:, -1]
            y_r<-right_data[:, -1]
            impurity<-self.impurity_node(y, y_l, y_r)
            IF impurity > max_impurity THEN
              best_feature<-features
              best_threshold<-threshold
              max_impurity<-impurity
            ENDIF
          ENDIF

      IF max_impurity > 0 THEN
        gini<-self.gini_index(datasets[:, -1])
        feature_type<-"numeric"
        left_data, right_data<-self.split_data(datasets, best_feature, best_threshold, feature_type)
        left_child<-self.make_tree(left_data, current_depth+1)
        right_child<-self.make_tree(right_data, current_depth+1)
        return Node(feature<-best_feature, threshold<-best_threshold, left<-left_child, right<-right_child,
                    impurity<-max_impurity, depth<-current_depth, feature_type<-feature_type, gini<-gini, n_samples<-num_sample)
      ENDIF
    ENDIF

    gini<-self.gini_index(datasets[:, -1])
    class_vals<-self.class_value(y)
    return Node(class_vals<-class_vals, depth<-current_depth, gini<-gini, n_samples<-num_sample)
ENDFUNCTION
```

Gambar 4.4 *Pseudocode method make_tree*

Method make_tree merupakan *method* yang akan digunakan dalam pembuatan simpul yang nantinya akan berjalan secara rekursif sehingga membentuk suatu “pohon”. Untuk membuat simpul, akan dilakukan iterasi pada seluruh fitur data dan seluruh kemungkinan *threshold* yang ada. Fitur dan *threshold* sebagai pemilah terbaik akan ditentukan berdasarkan nilai impuritas tertinggi. Pembuatan simpul akan berhenti bila kriteria jumlah minimal sampel pemilah atau batas kedalaman pohon terpenuhi sehingga, simpul tersebut akan menjadi simpul *leaf*. Hasil keluaran pada *method* ini merupakan simpul *parent* yang memiliki dua simpul *child* yaitu *child left* dan *child right*. Apabila kriteria pemberhentian pembuatan simpul terpenuhi,

maka hasil keluaran *method* ini merupakan simpul *leaf* yang memiliki informasi label.

4. *Method class_value*

```
FUNCTION  class_value(data):
    unique, counts<-np.unique(data, return_counts=True)
    vote_leaf<-dict(zip(unique, counts))
    return max(vote_leaf, key<-vote_leaf.get)
ENDFUNCTION
```

Gambar 4.5 *Pseudocode method class_value*

Method class_value merupakan *method* untuk menentukan label pada simpul *leaf*. Label pada simpul *leaf* ditentukan dengan aturan pluralitas atau label dengan jumlah terbanyak. Hasil keluaran dari *method* ini adalah data label untuk simpul *leaf*.

5. *Method predict* dan *visit_tree*

```
FUNCTION  predict(X):
    prd_rst<-[]
    FOR x in X:
      rst<-self.visit_tree(self.root, x)
      prd_rst.append(rst)
    return np.array(prd_rst)
ENDFUNCTION
```

Gambar 4.6 *Pseudocode method predict*

```
FUNCTION  visit_tree(tree, x):
    IF tree.class_value != None THEN
      return tree.class_value
    ENDIF
    IF x[tree.feature] <= tree.threshold THEN
      return self.visit_tree(tree.left, x)
    ELSE
      return self.visit_tree(tree.right, x)
    ENDIF
ENDFUNCTION
```

Gambar 4.7 *Pseudocode method visit_tree*

Pada *method predict* dan *visit_tree* digunakan untuk melakukan prediksi dari data *input*. Pada *method visit_tree* akan dilakukan pengecekan

pada setiap simpul dari “pohon” yang telah dibuat dimulai dari simpul *root*. Pengecekan yang pertama dilakukan adalah mengecek apakah simpul merupakan simpul *leaf* atau bukan, apabila simpul merupakan simpul *leaf* maka akan dikembalikan nilai label. Apabila simpul yang dicek bukan simpul *leaf* maka akan dilakukan perbandingan antara data *input* dengan *rule* yang ada pada simpul. Dari perbandingan tersebut kemudian ditentukan pengecekan selanjutnya akan ke simpul *child left* atau *child right*. *Method* ini akan diulang secara rekursif sampai mendapatkan data label pada simpul *leaf*. Hasil keluaran dari *method* ini merupakan hasil klasifikasi.

B. Program *Random Forest*

Program *random forest* yang telah dibuat memiliki lima *method* atau fungsi penting yaitu sebagai berikut:

1. *Method bootstrapping*

```
FUNCTION bootsrapping(data)
  btsr_idx<-self.my_rng.choice(len(data), len(data))
  oob_idx<-np.array([i FOR i in range(len(data)) IF i not in btsr_idx])
  bootsrap_data, oob_data<-data[btsr_idx], data[oob_idx]
  return bootsrap_data, oob_data
ENDFUNCTION
```

Gambar 4.8 *Pseudocode method bootstrapping*

Pada *method* ini memiliki *input* data berupa data latih. Dari data latih tersebut, akan dilakukan pengambilan sampel secara acak dengan pengembalian. Hal tersebut dilakukan sebanyak jumlah sampel. Hasil dari pengambilan sampel tersebut akan masuk ke dalam variabel *bootsrap_data*, sementara sampel yang tidak terpilih akan masuk ke dalam variabel *oob_data*. *Method* ini akan mengembalikan data latih yang telah di-*bootstrapping* dan data yang tidak masuk ke dalam *bootstrapping* atau data OOB.

2. *Method m_feature_selection*

```
FUNCTION m_feature_selection(x, max_feature)
  sample, feature<-x.shape
  mf<-0
  IF isinstance(max_feature, (int, np.int_)) THEN
    IF int(max_feature) > feature THEN
      raise Exception()
    mf<-int(max_feature)
  ELSE
    IF max_feature = "hsqrt" THEN
      mf<-int(1/2*np.absolute(np.sqrt(feature)))
    ELSE IF max_feature = "sqrt" THEN
      mf<-int(np.sqrt(feature))
    ELSE IF max_feature = "dsqrt" THEN
      mf<-int(2*(np.sqrt(feature)))
    ELSE
      raise Exception()
  ENDIF
  m_feature_idx<-self.my_rng.choice(feature, mf, replace=False)
  features<-x[:, m_feature_idx]
  return features, m_feature_idx
ENDFUNCTION
```

Gambar 4.9 *Pseudocode method m_feature_selection*

Pada *method m_feature_selection* membutuhkan *input* data *feature* atau data X yang telah dilakukan *bootstrapping*. *Input* lain yang diperlukan dalam *method ini* adalah jumlah maksimal fitur dalam bentuk *int* atau dalam bentuk *string*. Bila *input* jumlah fitur dalam bentuk *int* maka akan langsung dimasukkan ke dalam variabel *mf* jika besar *input* lebih kecil dari jumlah fitur pada data. Sementara jika jumlah fitur dalam bentuk *string* maka akan dilakukan pengecekan terlebih dahulu kemudian akan dilakukan perhitungan sesuai dengan kondisi *string* sehingga, didapat jumlah fitur dari perhitungan yang dilakukan yang kemudian dimasukkan ke dalam variabel *mf*. Jumlah fitur dalam variabel *mf* yang didapat akan dijadikan sebagai jumlah fitur maksimal yang digunakan dalam membentuk pohon. Pemilihan fitur dalam membentuk pohon dilakukan secara acak sebanyak jumlah fitur maksimal yang terdapat pada variabel *mf*. Hasil keluaran dalam *method* ini merupakan data *feature* atau data X yang telah dipilih secara acak dan *index* fitur yang

telah dipilih. *Pseudocode method m_feature_selection* dapat dilihat pada gambar 4.2.

3. *Method make_forest*

```
FUNCTION  make_forest(self, data):
    forest<-[]
    FOR i in range(self.n_estimators):
      tree_data<-{}
      bootsrap_data, oob_data<-self.bootsrapping(data)
      x_btsr, y_btsr<-bootsrap_data[:, :-1], bootsrap_data[:, -1]
      r_features_selection, idx_feature_selection<-self m_feature_selection(x_btsr, self.max_features)
      tree_model<-MyDecisionTreeClassfier(min_sample_split=self.min_sample_split, max_depth=self.max_depth)
      tree_model.fit(r_features_selection, y_btsr)
      tree_data["tree_model"]<-tree_model
      tree_data["oob"]<-oob_data
      tree_data["idx_feature_selection"]<-idx_feature_selection
      forest.append(tree_data)
    self.oob_score<-self.calculate_oob(forest)
    return forest
ENDFUNCTION
```

Gambar 4.10 *Pseudocode make_forest*

Method make_forest membutuhkan *input* data berupa data latih yang nantinya akan digunakan untuk membentuk "pohon". Sebelum digunakan untuk membentuk pohon, data latih akan melewati *method bootstrapping* dan *method m_feature_selection* terlebih dahulu. Pohon yang telah dibentuk kemudian akan disimpan pada variabel *tree_data* beserta dengan data OOB dan data index fitur yang dipilih. Variabel *tree_data* kemudian akan disimpan ke dalam *list* bernama *forest*. Proses ini akan diulang sebanyak *k* kali sehingga, akan terbentuk banyak *tree_data* dalam variabel *forest*. *Pseudocode method make_forest* dapat dilihat pada gambar 4.3.

4. *Method predict* dan *visit_forest*

```
FUNCTION  predict(X):
    predict_result<-[]
    FOR x in X:
      result<-self.visit_forest(np.array([x]))
      predict_result.append(result)
    return np.array(predict_result)
ENDFUNCTION
```

Gambar 4.11 *Pseudocode method predict*

```
FUNCTION  visit_forest(x):
    predict_forest<-[]
    FOR tree in self.forest:
      idx_need<-tree["idx_feature_selection"]
      using_feature<-x[:, idx_need]
      predict_tree<-tree["tree_model"].predict(using_feature)
      predict_forest.append(predict_tree)
    ENDFOR
    predict_forest<-np.array(predict_forest)
    voting<-self.voting_forest(predict_forest)
    return voting
ENDFUNCTION
```

Gambar 4.12 *Pseudocode method visit_forest*

Pada *method visit_forest* digunakan untuk mendapatkan hasil klasifikasi. *Input* dari *method ini* merupakan data fitur atau data X. Kemudian akan dilakukan iterasi sebanyak jumlah *forest* yang telah dibentuk sebelumnya untuk mendapatkan hasil klasifikasi pada masing-masing pohon. Hasil klasifikasi yang didapatkan akan disimpan pada *list* dengan nama variabel *predict_forest*. Hasil klasifikasi pada variabel *predict_forest* akan dilakukan voting untuk mendapatkan hasil akhir klasifikasi dengan *method voting_forest*. *Method* ini akan mengembalikan hasil klasifikasi yang telah dilakukan voting.

5. *Method voting_forest*

```
FUNCTION  voting_forest(predict_forest_result):
  unique, counts<-np.unique(predict_forest_result, return_counts=True)
  vote<-dict(zip(unique, counts))
  return max(vote, key<-vote.get)
ENDFUNCTION
```

Gambar 4.13 *Pseudocode method voting_forest*

Method voting_forest akan digunakan untuk memvoting hasil klasifikasi yang telah didapatkan. Voting dilakukan dengan menghitung frekuensi dari hasil klasifikasi yang didapatkan, hasil klasifikasi dengan frekuensi terbesar akan dipilih sebagai hasil akhir klasifikasi. Pada *method* ini akan mengembalikan hasil voting klasifikasi.

C. Hasil Tahapan *Pre-processing* Data

1. Impor *datasets*

Langkah awal dalam tahapan penelitian ini adalah melakukan impor *datasets* ke dalam Google Colab. *Datasets* dimuat menggunakan *library drive* untuk dapat mengimpor *file csv* ke dalam Google Colab dan kemudian menggunakan *library pandas* untuk memuat *file* tersebut ke dalam program *python*. Setelah *file* berhasil dimuat, maka langkah selanjutnya adalah melakukan beberapa perbaikan pada data yang mengalami salah baca pada saat proses impor dilakukan. Selain itu, dilakukan juga mengubah beberapa nama kolom agar lebih mudah untuk digunakan pada saat tahapan selanjutnya.

2. Penyajian *datasets*

Setelah dilakukannya impor data, proses selanjutnya adalah eksplorasi data dan penyajian data. Seperti yang sudah dijelaskan pada bab sebelumnya, data yang penulis gunakan dalam pembuatan model *machine learning* berasal dari tiga sumber yang berbeda, yang kemudian direkap dan digabungkan menjadi sebuah *datasets* yang padu. *Datasets* memiliki jumlah baris sebanyak 188 baris dan jumlah kolom sebanyak 23 kolom. *Datasets* yang telah disusun kemudian dilakukan pengeksploran data untuk dilakukan analisis data dan mengetahui *insight* pada *datasets* yang telah dimiliki.

2.1. Korelasi *datasets*

Dari hasil korelasi yang telah dihasilkan pada gambar 3.4, diketahui bahwa seluruh nilai kolom memiliki nilai korelasi yang positif terhadap kolom “NA”, kecuali dengan kolom “SKS yang telah ditempuh”. Berdasarkan hal tersebut dapat disimpulkan bahwa naiknya nilai pada semua kolom kecuali kolom ‘SKS yang telah ditempuh’, akan mempengaruhi naiknya nilai “NA”.

2.2. Jumlah mahasiswa

Pada *datasets* yang telah diperoleh, maka dapat diketahui jumlah sampel sebanyak 188 sampel dari tiga program studi yang berbeda yaitu Ilmu Komputer, Pendidikan Matematika dan Matematika. Jumlah sampel juga

berasal dari dua angkatan mahasiswa yang berbeda yaitu, angkatan 2019 dan angkatan 2020.

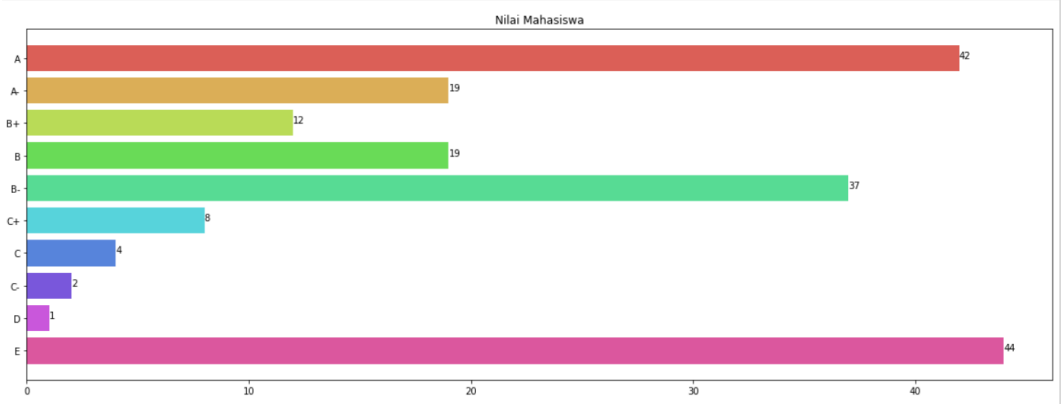
Tabel 4.1 Jumlah Mahasiswa

Program Studi	*2019*	*2020*
Ilmu Komputer	25	43
Matematika	38	0
Pendidikan Matematika	38	44

Dapat dilihat pada tabel 4.1 di atas, jumlah mahasiswa Program Studi Ilmu Komputer pada angkatan 2019 berjumlah 25 orang dan pada angkatan 2020 berjumlah 43 orang, lalu jumlah mahasiswa pada Program Studi Matematika angkatan 2019 berjumlah 38 orang, sementara pada Program Studi Pendidikan Matematika angkatan 2019 berjumlah 38 orang dan angkatan 2020 berjumlah 44 orang.

2.3. Sebaran nilai mahasiswa

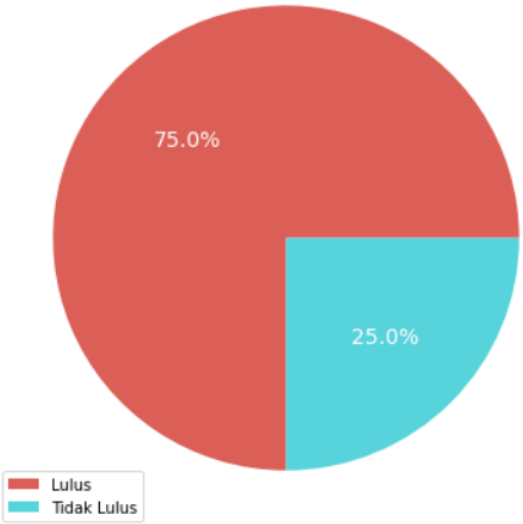
Analisis lain yang diperoleh adalah melihat sebaran data nilai mahasiswa dalam huruf. Data nilai mahasiswa dengan tiga program studi dan dua angkatan berbeda dapat dilihat sebaran nilai akhir mahasiswa dalam bentuk huruf seperti pada gambar 4.6 berikut.



Gambar 4.14 Sebaran nilai akhir mahasiswa

Dapat dilihat pada gambar 4.14 di atas, mahasiswa yang lulus atau mahasiswa yang mendapatkan nilai dengan rentang A – C sebanyak 141 mahasiswa. Sementara untuk mahasiswa yang tidak lulus atau mahasiswa yang mendapatkan nilai dengan rentang C- – D sebanyak 47 mahasiswa. Dapat disimpulkan kembali 75% mahasiswa pada *datasets* lulus mata kuliah,

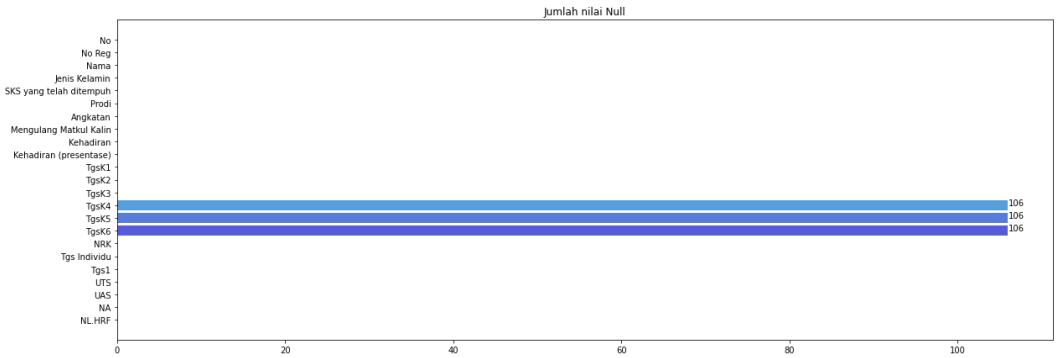
sementara sisanya 25% tidak lulus, grafik persentase dapat dilihat seperti pada gambar 4.15.



Gambar 4.15 Persentase kelulusan mahasiswa

2.4. Penulusuran nilai *null* dan nilai *outliers*

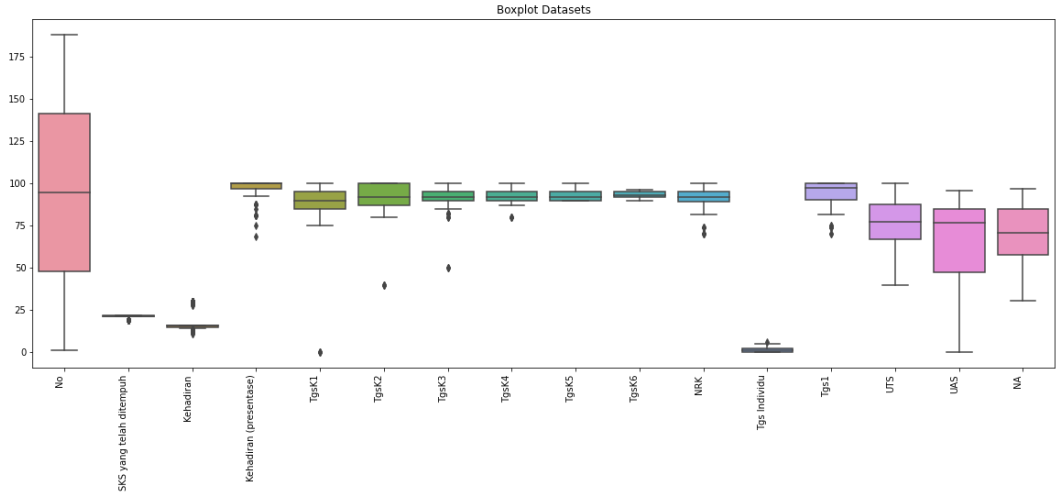
Dalam pengeksplorasian data, dilakukan juga penelusuran data untuk mengetahui apakah terdapat atau tidaknya nilai *null* atau nilai kosong dan nilai *outliers* pada datasets. Hal pertama yang akan ditelusuri adalah nilai *null*. Setelah dilakukan penelusuran, diperoleh hasil bahwa nilai *null* hanya terdapat pada tiga kolom, yaitu kolom TgsK4, kolom TgsK5, dan kolom TgsK6 yang masing-masing kolom memiliki jumlah nilai *null* yang sama, yaitu sebanyak 106 data yang bernilai *null*. Sementara itu, pada kolom lain tidak ditemukannya nilai *null*.



Gambar 4.16 Jumlah nilai *null* tiap kolom

Setelah dilakukan penelusuran nilai *null*, langkah selanjutnya yang dilakukan adalah penelusuran *outlier*s. Penelusuran *outliers* yang dilakukan

menggunakan diagram *boxplot* sebagaimana pada gambar 4.17 berikut. Pada gambar tersebut, dapat dilihat titik-titik yang terdapat pada luar *boxplot* adalah nilai *outliers*.



Gambar 4.17 *Boxplot* yang menunjukkan *outliers* pada *datasets*

3. Pemilihan kolom

Tahapan selanjutnya setelah melakukan pengeksplorasian data adalah menghilangkan kolom yang tidak akan diperlukan dalam pembuatan model *random forest*. Jumlah kolom awal sebelum dihilangkan sebanyak 23 kolom. Kemudian akan dihilangkan sebanyak tujuh kolom yaitu kolom "No", "No Reg", "Nama", "Kehadiran", "NRK", "NA", dan "Mengulang Matkul Kalin", sehingga tersisa 16 kolom yang akan digunakan dalam pembuatan model *random forest*.

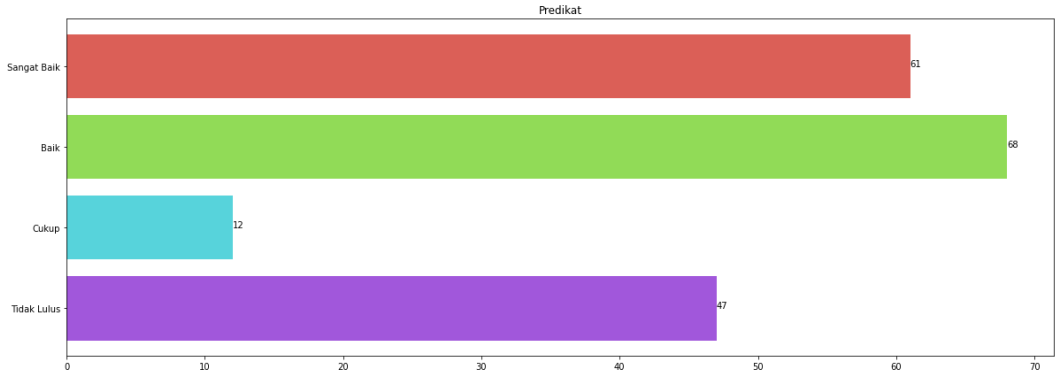
```
Data columns (total 23 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   No                       188 non-null    int64
 1   No Reg                   188 non-null    object
 2   Nama                     188 non-null    object
 3   Jenis Kelamin            188 non-null    object
 4   SKS yang telah ditempuh  188 non-null    int64
 5   Prodi                    188 non-null    object
 6   Angkatan                 188 non-null    object
 7   Mengulang Matkul Kalin   188 non-null    object
 8   Kehadiran                188 non-null    int64
 9   Kehadiran (presentase)   188 non-null    float64
 10  TgsK1                    188 non-null    int64
 11  TgsK2                    188 non-null    int64
 12  TgsK3                    188 non-null    int64
 13  TgsK4                    82 non-null     float64
 14  TgsK5                    82 non-null     float64
 15  TgsK6                    82 non-null     float64
 16  NRK                      188 non-null    float64
 17  Tgs Individu             188 non-null    int64
 18  Tgs1                     188 non-null    float64
 19  UTS                      188 non-null    float64
 20  UAS                      188 non-null    float64
 21  NA                       188 non-null    float64
 22  NL.HRF                   188 non-null    object
```

```
Data columns (total 16 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   JK         188 non-null    object
 1   SKS        188 non-null    int64
 2   Prodi      188 non-null    object
 3   Angkatan   188 non-null    object
 4   Kehadiran  188 non-null    float64
 5   TgsK1      188 non-null    int64
 6   TgsK2      188 non-null    int64
 7   TgsK3      188 non-null    int64
 8   TgsK4      82 non-null     float64
 9   TgsK5      82 non-null     float64
 10  TgsK6      82 non-null     float64
 11  TgsIdv     188 non-null    int64
 12  Tgs1       188 non-null    float64
 13  UTS        188 non-null    float64
 14  UAS        188 non-null    float64
 15  NL.HRF     188 non-null    object
```

Gambar 4.18 Perbedaan sebelum dan sesudah kolom dihilangkan

4. Pelabelan *datasests*

Setelah kolom-kolom yang tidak digunakan telah dihilangkan, langkah selanjutnya adalah pelabelan *datasests*. Kolom ‘NL.HRF’ dipilih sebagai acuan untuk membuat kolom baru yang nantinya akan menjadi *label datasest*. Kolom tersebut akan diberi nama *predicate* dengan *value* kolom akan diisi berdasarkan predikat kriteria nilai yang menjadi standar kelulusan mata kuliah pada Universitas Negeri Jakarta (UNJ) sebagaimana yang diatur pada Peraturan Rektor Nomor 7 tahun 2018 tentang peraturan akademik UNJ. Dalam penentuan predikat sesuai dengan nilai huruf yang telah diperoleh mahasiswa dapat dilihat pada tabel 2.8. Dari hasil akan diperoleh, label data baru terdiri dari nilai “Sangat Baik”, “Baik”, “Cukup” dan “Tidak Lulus”.



Gambar 4.19 Predikat mahasiswa

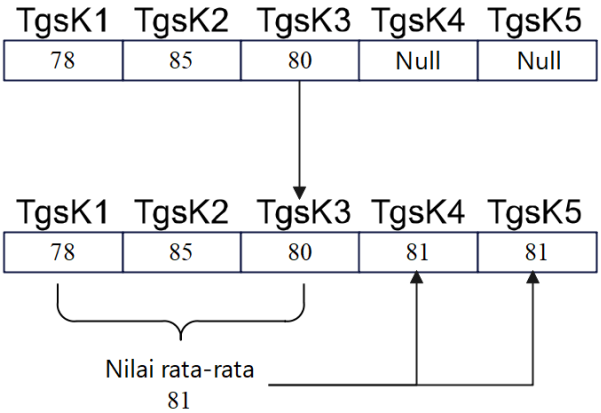
Dari gambar 4.19 di atas, dapat diketahui jumlah mahasiswa yang mendapatkan predikat ‘Sangat Baik’ sebanyak 61 orang, mahasiswa yang mendapatkan predikat ‘Baik’ sebanyak 68 orang, mahasiswa yang mendapatkan predikat ‘Cukup’ sebanyak 12 orang dan mahasiswa yang mendapatkan predikat ‘Tidak Lulus’ sebanyak 47 orang. Setelah kolom *predicate* dibuat, kolom “NL.HRF” akan di-*drop* atau dibuang karena sudah digantikan dengan kolom *predicate*.

5. Menangani nilai *null*

Setelah melakukan pengelompokan pada kolom yang nantinya menjadi label, tahap selanjutnya yang dilakukan adalah menangani nilai *null* pada *datasets*. Pada hasil penelusuran nilai *null* berdasarkan gambar 4.3, dapat disimpulkan bahwa kolom-kolom yang memiliki nilai *null* terdapat pada kolom TgsK4, kolom TgsK5 dan kolom TgsK6 dengan jumlah yang sama,

yaitu sebanyak 106. Setelah ditelusuri lebih jauh, kolom TgsK4, TgsK5 dan TgsK6 yang memiliki nilai *null* terdapat pada Program Studi Ilmu Komputer angkatan 2019 dan 2020 serta Program Studi Matematika angkatan 2020. Hal ini dikarenakan pada kelas-kelas tersebut tidak memiliki penilaian pada TgsK4, TgsK5 dan TgsK6.

Untuk menangani nilai *null* pada ketiga kolom tersebut, maka akan dilakukan perubahan/pengisian nilai yang sebelumnya nilai *null* menjadi sebuah nilai baru. Nilai baru yang akan mengisi nilai *null* ini adalah nilai rata-rata dari nilai tugas kelompok sebelumnya pada setiap mahasiswa.



Gambar 4.20 Ilustrasi pengisian nilai *null*.

Pada gambar 4.20 di atas, dapat dilihat ilustrasi untuk pengisian nilai *null*. Diilustrasikan nilai tugas kelompok pada TgsK1 sampai TgsK3 adalah 78, 85 dan 80. Dari nilai tugas kelompok tersebut, diperoleh nilai rata-rata sebesar 81. Nilai tersebut akan digunakan untuk mengisi nilai yang sebelumnya nilai *null*. Cara ini dilakukan dengan asumsi bahwa pengisian dengan nilai rata-rata pada nilai *null* tidak akan merubah nilai rata-rata dari tugas kelompok pada mahasiswa tersebut.

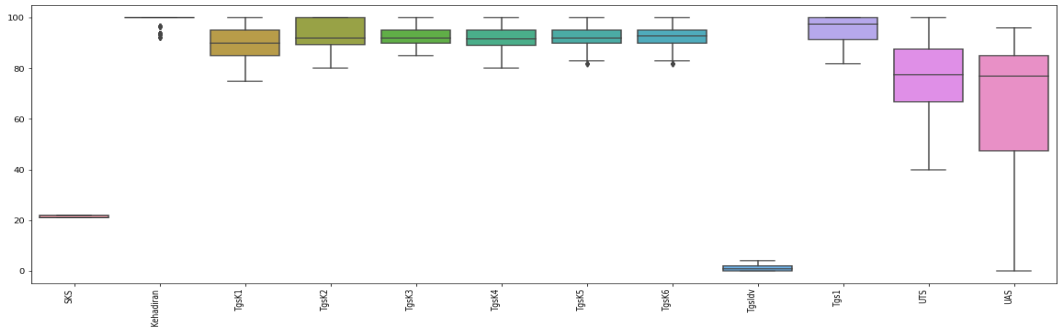
6. Menanangani nilai *outliers*

Selanjutnya untuk menangani nilai *outliers* dilakukan dengan mengganti nilai *outliers* menjadi nilai rata-rata dari kolom dimana nilai *outliers* tersebut berada. Dengan menggunakan rumus interkuartil, maka diperoleh nilai *outliers* seperti pada gambar 4.13 berikut ini.

```
({'TgsIdv': array([5, 5, 5, 5, 6, 5, 5, 5])},
 {'SKS': array([19, 19, 19, 19, 19, 19, 19, 19, 19, 19, 19, 19, 19, 19, 19, 19, 19,
        19, 19, 19, 19, 19, 19, 19, 19, 19, 19, 19, 19, 19, 19, 19, 19, 19,
        19, 19, 19, 19]),
  'Kehadiran': array([84.61538462, 81.25      , 87.5       , 87.5       , 87.5       ,
        87.5       , 81.25      , 87.5       , 81.25      , 81.25      ,
        81.25      , 68.75      , 81.25      , 75.        ]),
  'TgsK1': array([0, 0, 0, 0]),
  'TgsK2': array([40, 40, 40, 40]),
  'TgsK3': array([82, 80, 80, 50, 82, 50, 50, 82, 80, 82, 82, 82]),
  'TgsK4': array([46.66666667, 46.66666667, 46.66666667, 46.66666667, 74.        ,
        74.        , 74.        ]),
  'TgsK5': array([46.66666667, 46.66666667, 46.66666667, 46.66666667, 74.        ,
        74.        , 74.        ]),
  'TgsK6': array([46.66666667, 46.66666667, 46.66666667, 46.66666667, 74.        ,
        74.        , 74.        ]),
  'Tgs1': array([75., 70., 75., 70., 74., 74., 74.])})
```

Gambar 4.21 Nilai *outliers*

Nilai-nilai *outliers* di atas, diganti menjadi nilai rata-rata dari kolom dimana nilai *outliers* tersebut berada. Setelah dilakukan penanganan ini, jumlah *outliers* menjadi lebih sedikit dari sebelumnya. *Outliers* sesudah penanganan dapat dilihat pada gambar 4.22 berikut.



Gambar 4.22 *Outliers* sesudah ditangani

7. *Split datasets*

Langkah selanjutnya setelah menangani nilai *null* dan nilai *outliers* adalah membagi *datasets. Datasets* dibagi menjadi data X sebagai *feature datasets* dan data Y sebagai *label datasets*. Data X dan Y kemudian akan dibagi menjadi data latih dan data tes dengan data latih sebesar 70% dan data tes sebesar 30%.

```
print(X_train.shape, X_test.shape)
```

```
(131, 15) (57, 15)
```

```
print(y_train.shape, y_test.shape)
```

```
(131,) (57,)
```

Gambar 4.23 Jumlah fitur dan sampel pada data latih dan data tes

Pada gambar 4.23, dapat dilihat bahwa jumlah fitur pada data latih dan data tes memilki jumlah yang sama, yaitu sebanyak 15 dan jumlah sampel pada latih sebanyak 131 serta pada data tes sebanyak 57.

8. Encoding

Setelah *datasets* di-*split*, selanjutnya akan dilakukan *encoding* terhadap fitur pada data latih dan data tes yang memiliki tipe data kategori. Metode yang digunakan untuk melakukan *encoding* disesuaikan dengan karakteristik masing-masing fitur. Fitur-fitur yang termasuk ke dalam tipe data kategori adalah fitur "JK", fitur "Prodi" dan fitur "Angkatan". Ketiga fitur tersebut termasuk ke dalam data kategori dengan tipe nominal, maka akan digunakan dengan metode *One Hot Encoding*. Hasil dari *encoding* dapat dilihat pada lampiran 3 dan lampiran 4.

Setelah dilakukan *encoding* pada data X atau data fitur, selanjutnya akan dilakukan *encoding* pada data label. *Encoding* data label dilakukan dengan cara mengurutkan label secara alfabet, lalu diberi penomoran dari alfabet paling awal sampai paling akhir, sehingga diperoleh label baru dalam bentuk angka yang dapat dilihat pada tabel 4.2 sebagai berikut.

Tabel 4.2 Tabel hasil *encoding* label

Label Awal	Label Baru
Baik	0
Cukup	1
Sangat Baik	2
Tidak Lulus	3

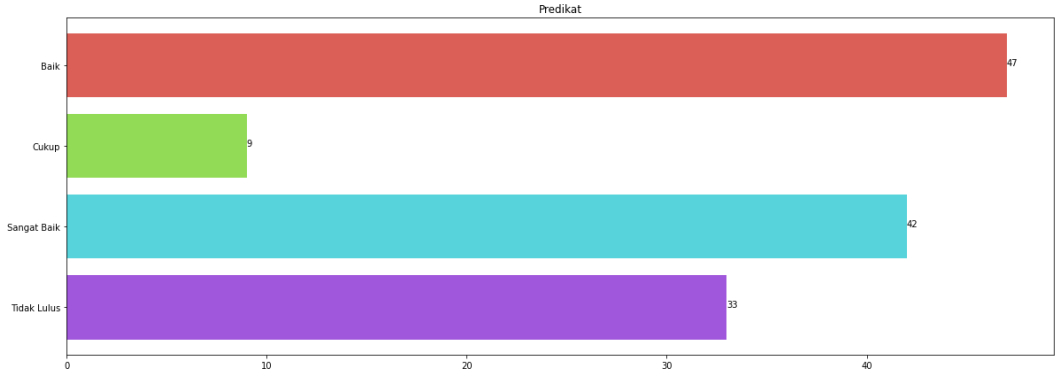
9. Feature scaling

Setelah dilakukan *encoding* pada fitur X yang bertipe data kategori, kemudian akan dilakukan normalisasi terhadap fitur X yang bertipe data

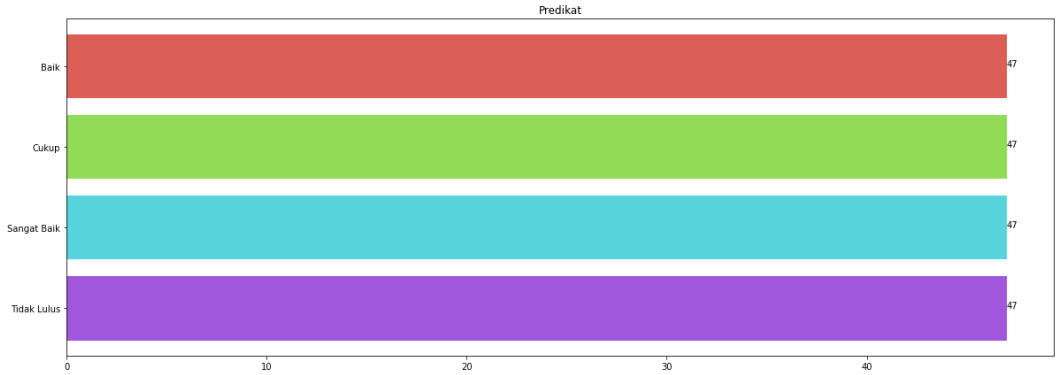
numerik. Fitur X yang bertipe data numerik akan dilakukan normalisasi dengan metode *standard scaler*. Hasil dari normalisasi dapat dilihat pada lampiran 5 dan lampiran 6.

10. *Oversampling*

Setelah dilakukan *encoding* dan normalisasi, selanjutnya akan dilakukan *balancing* data dengan cara *oversampling* menggunakan metode SMOTE pada data latih, agar data tersebut menjadi lebih *balance*. Pada gambar 4.24 berikut ini, dapat dilihat data latih tidak *balance*. Hal ini ditunjukkan karena adanya perbedaan jumlah label yang cukup besar.



Gambar 4.24 Data latih sebelum *balancing* data



Gambar 4.25 Data latih sesudah *balancing* data

Untuk menangani hal ini, maka dilakukan *oversampling* menggunakan metode SMOTE, sehingga label yang sebelumnya memiliki jumlah yang berbeda sekarang memiliki jumlah yang sama. Hal ini dapat dilihat pada gambar 4.24, dimana pada gambar tersebut memiliki jumlah label yang berbeda, sedangkan setelah dilakukan *oversampling* jumlah label yang sebelumnya berbeda kini memiliki jumlah label yang sama. Dari hasil

oversampling dengan SMOTE ini, jumlah sampel pada data latih yang sebelumnya berjumlah 131 bertambah menjadi 188.

Dari semua proses yang telah dilakukan sebelumnya, maka menghasilkan data yang bersih yang nantinya akan digunakan untuk memodelkan *random forest*. Data bersih tersebut memiliki jumlah fitur sebanyak 19, baik data latih maupun data tes. Sementara itu, jumlah sampel pada data latih sebanyak 188 sampel dan data tes sebanyak 57 sampel.

D. Hasil Tahapan Pemodelan *Random Forest*

Dalam memodelkan *random forest* akan menggunakan data latih dan data tes yang sebelumnya telah diperoleh. Untuk melakukan pemodelan *random forest* ini dilakukan dengan membatasi maksimal kedalaman pohon yang akan dibangun dalam model *random forest* sebesar 3. Selain itu, dibatasi juga *random state* sebesar 50, agar model menghasilkan keluaran yang sama ketika akan dijalankan ulang.

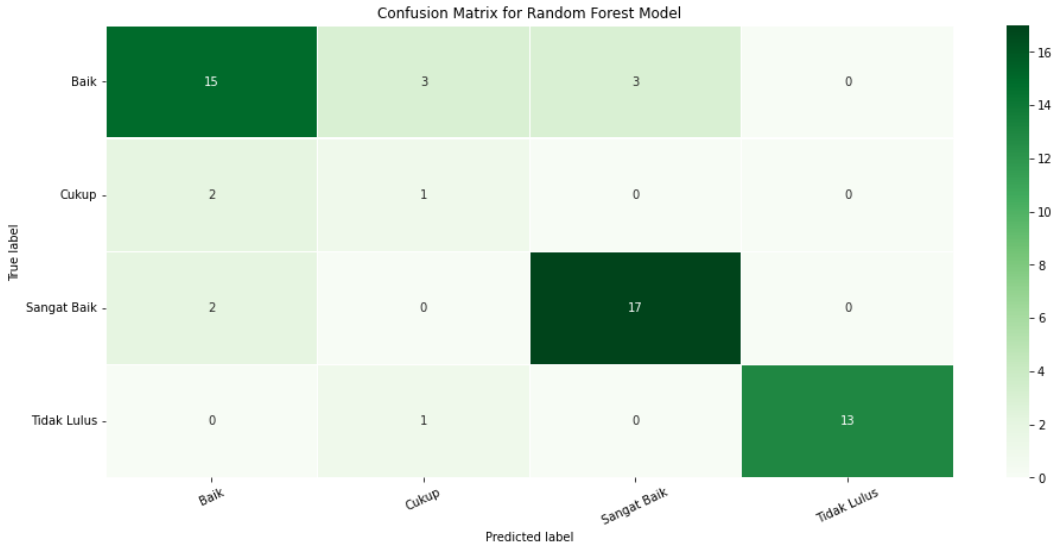
1. Pemodelan *random forest*

   Pada pemodelan *random forest* menggunakan parameter yang telah ditentukan di atas. Model dibuat sesuai dengan ilustrasi pada gambar 3.8, diawali dengan *bootstrapping*, kemudian selanjutnya dilakukan tahap *random feature selection* dengan jumlah maksimal fitur sebanyak $\left|\sqrt{p}\right|$, dengan *p* adalah jumlah keseluruhan fitur. Berdasarkan data latih yang telah diperoleh, maka jumlah fitur maksimal sebanyak $\left|\sqrt{19}\right|$ atau sama dengan 4 (pembulatan nilai ke bawah). Hasil data dari tahap *bootstrapping* dan *random feature selection*, selanjutnya akan digunakan untuk membangun pohon keputusan. Tahapan ini akan dilakukan secara terus-menerus sebanyak 50 kali. Dengan alasan akan dibangun pohon dalam model *random forest* sebanyak 50.

2. Penghitungan *confusion matrix*

   Setelah model *random forest* berhasil dibuat, kemudian model akan diujikan untuk memprediksi data latih dan data tes. Dari hasil prediksi model terhadap data latih dan data tes, kemudian disamakan dengan hasil aktualnya, maka diperoleh *confusion matrix* sebagai berikut.

Gambar 4.26 *Hasil confusion matrix* terhadap data latih



Gambar 4.27 *Hasil confusion matrix* terhadap data tes

Dari hasil *confusion matrix* pada gambar 4.26 dan 4.27, diperoleh hasil akurasi model menggunakan data latih sebesar 93% dan menggunakan data tes memperoleh hasil sebesar 81%. Dari kedua hasil akurasi tersebut, model dinilai tidak mengalami *overfitting* atau *underfitting*.

Berdasarkan pada gambar 4.27 di atas, dapat dilihat performa model dalam memprediksi dengan benar label “Baik” sebanyak 15 dari 21, label “Cukup” sebanyak 1 dari 3, label “Sangat Baik” sebanyak 17 dari 19, dan label “Tidak Lulus” sebanyak 13 dari 14. Berdasarkan hasil *confusion matriks*

tersebut, maka dapat diperoleh untuk nilai *precision*, *recall* dan *F-Measure* sebagai berikut:

Tabel 4.3 Nilai *precision*, *recall*, dan *F-Measure*

Predikat	*Precision*	*Recall*	*F-Measure*	*Support*
Sangat Baik	85%	89%	87%	19
Baik	79%	71%	75%	21
Cukup	20%	33%	25%	3
Tidak Lulus	100%	93%	96%	14

Pada tabel 4.3 di atas, diperoleh nilai *precision, recall,* dan *F-measure* pada label "Baik" sebesar 79%, 71% dan 75%, kemudian untuk label "Cukup" sebesar 20%, 33% dan 25%, lalu untuk label "Sangat Baik" sebesar 85%, 89% dan 87% dan untuk label "Tidak Lulus" sebesar 100%, 93% dan 96%.

3. Uji validitas model

Model yang telah dibuat sebelumnya akan dilakukan uji validitas model *random forest* menggunakan k-*fold cross validation* dengan *k* sebanyak 10. Data latih akan dibagi ke dalam 10 segmen dan pada setiap iterasinya 9 segmen akan berperan menjadi data latih dan 1 segmen menjadi data validasi. Jumlah *k* sebanyak 10 dipilih karena nilai akurasi pada *k* lebih dari 10 memiliki nilai yang tidak jauh berbeda. Hasil dari 10-*fold cross validation* diperoleh hasil akurasi pada tabel 4.4 sebagai berikut.

Tabel 4.4 Hasil akurasi dengan 10-*fold cross validation*

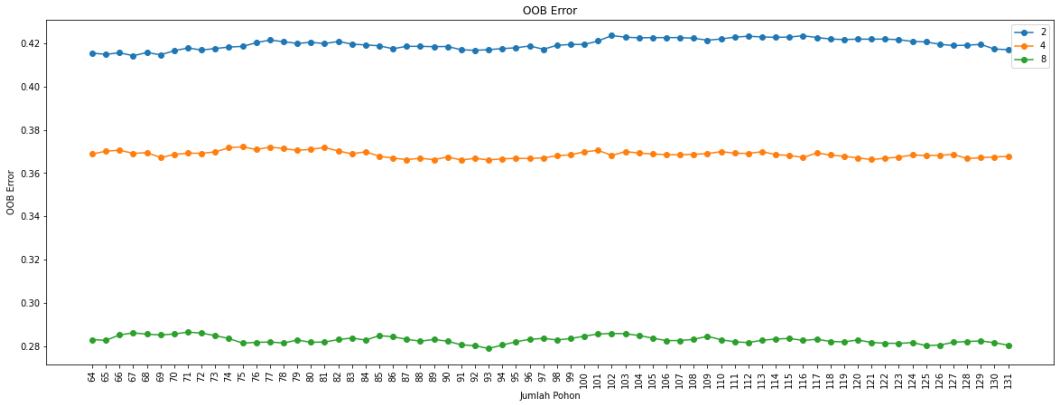
Iterasi	Akurasi
1	84%
2	95%
3	100%
4	89%
5	58%
6	89%
7	95%
8	84%
9	72%
10	83%

Dari hasil 10-*fold cross validation* pada tabel 4.4 di atas, dapat disimpulkan nilai rata-rata akurasi dari model sebesar 0.85 atau 85% dengan standar deviasi sebesar 0.12. Pada tabel tersebut, juga dapat dilihat bahwa akurasi pada iterasi ke-5 memiliki nilai akurasi yang kurang baik, yaitu sebesar 0.58.

Hal ini dikarenakan hasil akurasi pada iterasi ke-5 berbeda cukup jauh dengan hasil pada iterasi lainnya. Sehingga akan dilakukan pencarian kombinasi parameter model *random forest* untuk mendapatkan performa model yang lebih baik.

4. Mencari parameter optimum

Dari hasil uji validitas yang kurang baik pada tahap sebelumnya, maka akan dilakukan pencarian parameter terbaik untuk membangun model *random forest*. Kombinasi parameter yang digunakan untuk mencari parameter terbaik adalah parameter jumlah pohon dan parameter jumlah maksimal fitur. Untuk jumlah pohon yang akan diujikan beragam yaitu dari jumlah pohon sebanyak 64 sampai sebanyak 132, sementara untuk jumlah fitur maksimal adalah 2, 4 dan 8. Jumlah pohon dan jumlah maksimal fitur terbaik akan diperoleh dengan membuat model *random forest* dengan menggunakan kombinasi yang telah ditentukan dan kemudian melihat nilai galat OOB terkecil pada model yang telah dibuat.



Gambar 4.28 Galat OOB

Hasil dari galat OOB dengan kombinasi jumlah pohon dan jumlah fitur maksimal dapat dilihat pada lampiran 4. Dari lampiran tersebut, dapat dibuat grafik hasil galat OOB pada gambar 4.28. Berdasarkan gambar tersebut diperoleh nilai galat OOB terkecil sebesar 0.27909 pada kombinasi jumlah pohon sebanyak 93 dan jumlah fitur maksimal sebanyak 8.

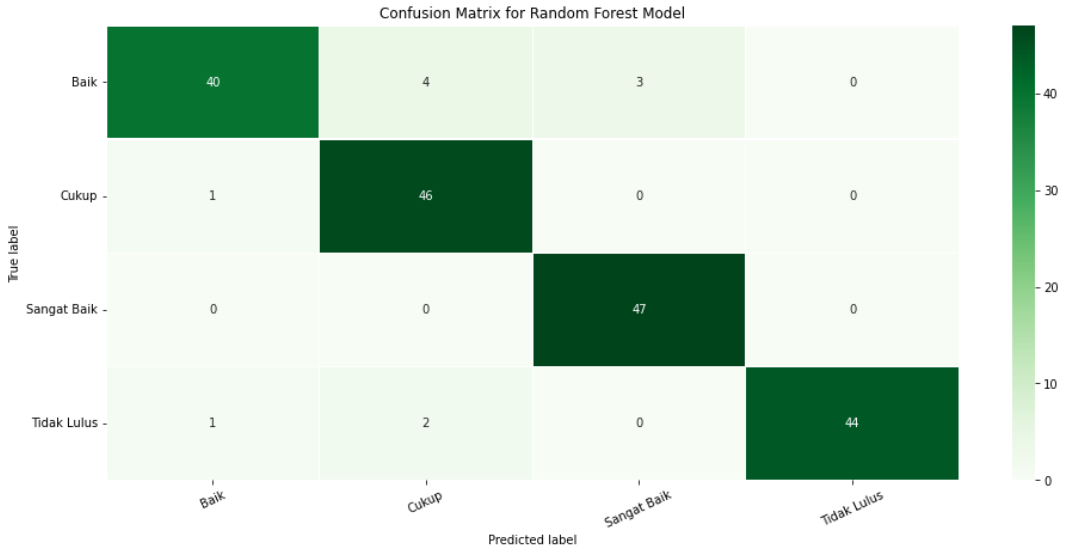
5. Pemodelan *random forest* baru

Setelah diperoleh kombinasi parameter terbaik, kemudian akan dilakukan kembali pembuatan ulang model *random forest* dengan mengganti

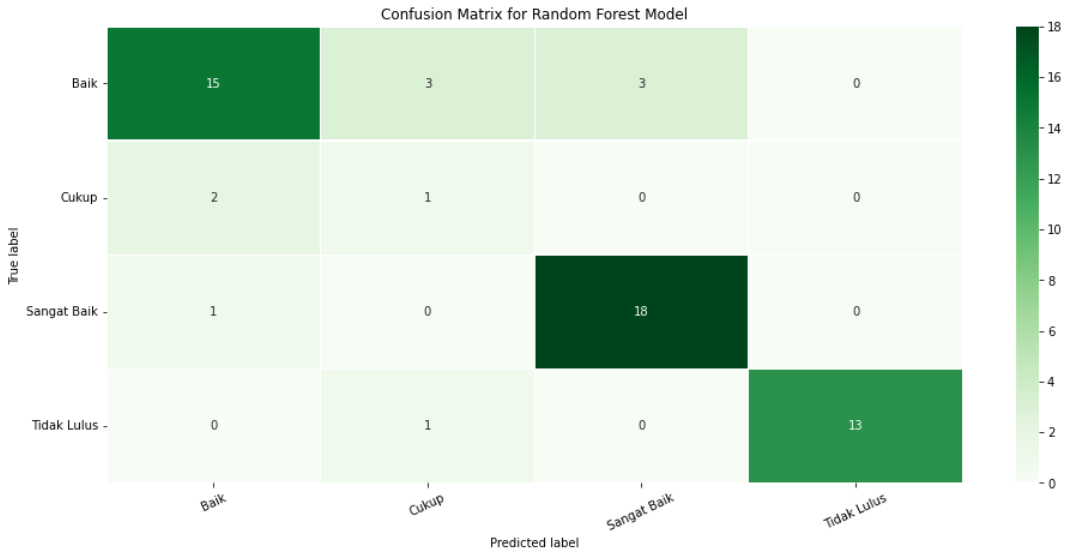
parameter untuk jumlah pohon yang digunakan sebanyak 93 dan jumlah fitur maksimal sebesar 8.

6. Penghitungan *confusion matrix* model baru

Dari pemodelan ulang yang dilakukan, kemudian model baru diujikan kembali untuk memprediksi data latih dan data tes. Dari hasil prediksi model baru terhadap data latih dan data tes, kemudian disamakan dengan hasil aktualnya, maka diperoleh *confusion matrix* sebagai berikut.



Gambar 4.29 *Hasil confusion matrix* model baru terhadap data latih



Gambar 4.30 Hasil *confusion matrix* model baru terhadap data tes

Dari hasil *confusion matrix* pada gambar 4.29 dan 4.30, diperoleh hasil akurasi model baru menggunakan data latih sebesar 94% dan menggunakan

data tes memperoleh hasil sebesar 82%. Dari kedua hasil akurasi tersebut, model baru dinilai tidak mengalami *overfitting* atau *underfitting*.

Berdasarkan pada gambar 4.30 di atas, dapat dilihat performa model dalam memprediksi dengan benar label "Baik" sebanyak 15 dari 21, label "Cukup" sebanyak 1 dari 3, label "Sangat Baik" sebanyak 18 dari 19, dan label "Tidak Lulus" sebanyak 13 dari 14. Berdasarkan *confusion matriks* tersebut, maka dapat diperoleh untuk nilai *precision*, *recall* dan *F-Measure* sebagai berikut:

Tabel 4.5 Nilai *precision*, *recall*, dan *F-Measure* model baru

Predikat	*Precision*	*Recall*	*F-Measure*	*Support*
Sangat Baik	86%	95%	90%	19
Baik	83%	71%	77%	21
Cukup	20%	33%	25%	3
Tidak Lulus	100%	93%	96%	14

Pada tabel 4.5 di atas, diperoleh nilai *precision*, *recall*, dan *F-measure* pada label "Baik" sebesar 83%, 71% dan 77%, kemudian untuk label "Cukup" sebesar 20%, 33% dan 25%, lalu untuk label "Sangat Baik" sebesar 86%, 95% dan 90% dan untuk label "Tidak Lulus" sebesar 100%, 93% dan 96%.

Berdasarkan hasil perhitungan model *random forest* yang pertama dengan model *random forest* yang menggunakan kombinasi parameter terbaik, maka dapat dibandingkan nilai dari *precision*, *recall*, dan *F-Measure* sebagai berikut:

Tabel 4.6 Perbandingan model lama dan model baru

Predikat	*Precision*		*Recall*		*F-Measure*		*Accuracy*	
	Model Pertama	Model Baru	Model Pertama	Model Baru	Model Pertama	Model Baru	Model Pertama	Model Baru
Sangat Baik	85%	86%	89%	95%	87%	90%	81%	82%
Baik	79%	83%	71%	71%	75%	77%		
Cukup	20%	20%	33%	33%	25%	25%		
Tidak Lulus	100%	100%	93%	93%	96%	96%		

Dari tabel 4.6 di atas, dapat disimpulkan bahwa hasil dari *random forest* yang baru untuk label "Baik" memiliki peningkatan nilai pada *precision* dan *F-Measure* sedangkan pada label "Sangat Baik" memiliki peningkatan nilai pada *precision*, *recall*, dan *F-Measure*. Sementara itu, untuk label "Cukup" dan "Tidak Lulus" tidak memiliki perubahan. Dari hasil tersebut model yang baru memiliki performa yang lebih baik daripada model yang pertama, sehingga model yang baru dapat disebut sebagai model optimum.

7. Uji validitas model optimum

Setelah menghitung *confusion matrix* pada model optimum, selanjutnya dapat dilakukan uji validitas dengan cara yang sama seperti sebelumnya, yaitu dengan *k-fold cross validation* dengan *k* sebanyak 10. Dengan *10-fold cross validation* mendapatkan hasil akurasi sebagai berikut.

Tabel 4.7 Hasil akurasi *k-fold cross validation* model optimum

Iterasi	Akurasi
1	95%
2	100%
3	100%
4	95%
5	79%
6	84%
7	89%
8	84%
9	83%
10	89%

Dari tabel 4.7 di atas dapat diketahui rata-rata akurasi dari model optimum sebesar 0.9 atau 90% dengan standar deviasi sebesar 0.07. Dari hasil rata-rata akurasi dan standar deviasi yang dihasilkan, maka model *random forest* optimum yang telah dibuat dengan kombinasi parameter terbaik dinilai memiliki kinerja yang lebih optimal atau lebih baik dibandingkan dengan model *random forest* yang pertama.

E. Hasil Analisis Model *Random Forest*

Setelah mendapatkan model yang optimum, maka dapat dilakukan penghitungan MDI (*Mean Decrease Impurity*). Penghitungan MDI dilakukan untuk

mengetahui *feature importance* atau fitur-fitur yang penting dalam menentukan hasil klasifikasi predikat pada *random forest*.

Tabel 4.8 Hasil *feature importance*

Fitur	*Importance*
UAS	26%
Prodi	22%
TgsK6	10%
TgsK1	7%
Angkatan	7%
UTS	7%
TgsK5	6%
TgsK3	4%
TgsIdv	2%
TgsK4	2%
Jenis Kelamin	2%
Tgs1	1%
Kehadiran	1%
TgsK2	1%
SKS	1%

Pada tabel 4.8 di atas dapat dilihat bahwa fitur ‘UAS’ menjadi fitur yang paling berpengaruh karena memiliki nilai *importance* tertinggi dalam menentukan hasil klasifikasi predikat. Selain pada fitur-fitur yang tergolong nilai mahasiswa, fitur ‘Prodi’ menempatkan pada urutan ke-2. Hal ini dapat disimpulkan bahwa fitur ‘Prodi’ juga mempunyai pengaruh yang cukup tinggi untuk menentukan klasifikasi predikat, dikarenakan masing-masing program studi dapat memiliki bobot atau SKS yang berbeda-beda.

BAB V
KESIMPULAN DAN SARAN

A. Kesimpulan

Berdasarkan tujuan penelitian yaitu membuat model *random forest* untuk mengklasifikasikan predikat kelulusan mata kuliah dan menguji validitas dari model yang dibuat, maka dapat disimpulkan hasil penelitian sebagai berikut:

1. Dalam membuat model *random forest* melakukan *bootsrapping* dan *random feature selection*, kemudian dilanjutkan dengan membentuk pohon keputusan. Proses ini akan dilakukan sebanyak *k* kali, dimana *k* adalah banyaknya pohon yang akan dibentuk.
2. Sebelum dilakukan pemodelan *random forest*, terlebih dahulu melakukan *pre-processing datasest* dimulai dari penanganan *missing values*, penanganan *outliers*, *encoding*, normalisasi sampai dengan *balancing data*.
3. Hasil dari model optimum diperoleh tingkat akurasi terhadap data latih sebesar 94% dan pada data tes sebesar 82%.
4. Uji validitas pada model *random forest* optimum dengan menggunakan 10-*fold cross validation* memiliki rata-rata sebesar 90% dengan standar deviasi sebesar 0.07
5. Fitur "UAS" memiliki peran terbesar dalam menentukan hasil akhir klasifikasi, sementara fitur "Prodi" sebagai salah satu fitur yang bukan termasuk nilai mahasiswa juga memiliki peran yang besar ke-2 dalam menentukan hasil akhir klasifikasi.

B. Saran

Pada penelitian ini penulis memiliki saran untuk dilakukan pada penelitian selanjutnya sebagai berikut.

1. Dapat dikembangkan lagi *datasets* yang telah ada dengan menambah jumlah data dan latar belakang mahasiswa seperti apakah mahasiswa tersebut kuliah dan bekerja bersamaan atau dengan penghasilan orang tua.

Hal ini untuk melihat apakah ada faktor eksternal selain nilai yang dapat mempengaruhi nilai akhir mahasiswa.

2. Dapat ditambahkan metode pengklasteran terlebih dahulu sebelum melakukan klasifikasi untuk melihat bagaimana karakteristik tiap mahasiswa sehingga dapat dilihat bagaimana penanganan yang cocok untuk tiap klaster.
3. Dapat diintegrasikan dengan *web* LMS (*Learning Management Systme*) atau EWS (*Early Warning System*) agar mahasiswa dapat mengetahui status predikat mereka sebelum akhir studi mata kuliah sehingga mahasiswa dapat mengatur strategi mereka untuk masa yang akan datang.

DAFTAR PUSTAKA

Abdul Bujang, S. D., Selamat, A., & Krejcar, O. (2021). A Predictive Analytics Model for Students Grade Prediction by Supervised Machine Learning. *IOP Conference Series: Materials Science and Engineering*, *1051*(1), 012005. https://doi.org/10.1088/1757-899X/1051/1/012005

Adinugroho, S., & Sari, Y. A. (2018). *Implementasi data mining menggunakan weka*. Universitas Brawijaya Press.

Ahmad, A. (2017). *Mengenal Artificial Intelligence, Machine Learning, Neural Network, dan Deep Learning*.

Alverina, D., Chrismanto, A. R., & Santosa, R. G. (2018). Perbandingan Algoritma C4.5 dan CART dalam Memprediksi Kategori Indeks Prestasi Mahasiswa. *Jurnal Teknologi dan Sistem Komputer*, *6*(2), 76–83. https://doi.org/10.14710/jtsiskom.6.2.2018.76-83

Arafiyah, R., Hasibuan, Z. A., & Budi Santoso, H. (2020). Learning progress modeling for monitoring student. *2020 5th International Conference on Informatics and Computing, ICIC 2020*, 1–6. https://doi.org/10.1109/ICIC50835.2020.9288613

Ardiningtyas, Y. E., & Rosa, P. H. P. (2021). Analisis Balancing Data Untuk Meningkatkan Akurasi Dalam Klasifikasi. *Prosiding Seminar Nasional Aplikasi Sains & Teknologi (SNAST)*, 24–28.

BAN-PT. (2019). Akreditasi Perguruan Tinggi Kriteria dan Prosedur 3.0. *Badan Akreditasi Nasional Perguruan Tinggi*, 18.

Biau, G., Devroye, L., & Lugosi, G. (2008). Consistency of random forests and other averaging classifiers. *Journal of Machine Learning Research*, *9*(2008), 2015–2033.

Breiman, L. (2001). *Random Forests. Machine Learning*. 5–32. https://doi.org/10.1023/A:1010933404324

Breiman, L., & Cutler, A. (2003). *Manual on Setting Up, Using, and Understanding Random Forest V4.0*. http://oz.berkeley.edu/users/breiman/Using_random_forests_v4.0.pdf

Cahyanti, D., Rahmayani, A., & Husniar, S. A. (2020). Analisis performa metode

Knn pada Dataset pasien pengidap Kanker Payudara. *Indonesian Journal of Data and Science*, *1*(2), 39–43. https://doi.org/10.33096/ijodas.v1i2.13

Dewi, N. K., Mulyadi, S. Y., & Syafitri, U. D. (2012). Penerapan Metode Random Forest Dalam Driver Analysis. *Forum Statistika Dan Komputasi*, *16*(1), 35–43. http://journal.ipb.ac.id/index.php/statistika/article/view/5443

Izzah, A., & Widyastuti, R. (2016). Prediksi Kelulusan Mata Kuliah Menggunakan Hybrid Fuzzy Inference System. *Register: Jurnal Ilmiah Teknologi Sistem Informasi*, *2*(2), 60. https://doi.org/10.26594/r.v2i2.548

Kemendikbud. (2020). Statistik Pendidikan Tinggi (Higer Education Statistic) 2020. *PDDikti Kemendikbud*, 81–85. https://pddikti.kemdikbud.go.id/publikasi

Mambang, & Byna, A. (2017). Analisis Perbandingan Algoritma C.45, Random Forest Dengan Chaid Decision Tree Untuk Klasifikasi Tingkat Kecemasan Ibu Hamil. *Semnasteknomedia Online*, *5*(1), 103–108. https://ojs.amikom.ac.id/index.php/semnasteknomedia/article/view/1771

Nasution, D. A., Khotimah, H. H., & Chamidah, N. (2019). Perbandingan Normalisasi Data untuk Klasifikasi Wine Menggunakan Algoritma K-NN. *Computer Engineering, Science and System Journal*, *4*(1), 78. https://doi.org/10.24114/cess.v4i1.11458

Normawati, D., & Prayogi, S. A. (2021). Implementasi Naïve Bayes Classifier Dan Confusion Matrix Pada Analisis Sentimen Berbasis Teks Pada Twitter. *Jurnal Sains Komputer & Informatika (J-Sakti)*, *5*(2), 697–711.

Oshiro, T. M., Perez, P. S., & Baranauskas, J. A. (2012). How many trees in a random forest? *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *7376 LNAI*, 154–168. https://doi.org/10.1007/978-3-642-31537-4_13

Permana, A. Y., & Effendi, M. M. (2018). Optimasi Stemming Porter KBBI dan Cross Validation Naïve Bayes untuk Klasifikasi Topik Soal UN Bahasa Indonesia. *Jurnal Ilmiah Komputasi*, *17*(4). https://doi.org/10.32409/jikstik.17.4.2492

Pratama, A., Wihandika, R. C., & Ratnawati, D. E. (2018). Implementasi algoritme support vector machine (SVM) untuk prediksi ketepatan waktu kelulusan

mahasiswa. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, *2*(April), 1704–1708.

Ramdani, F., & Utami, I. Q. (2022). *Pengantar Data Science* (Pertama). Bumi Aksara.

Refaeilzadeh, P., Tang, L., & Liu, H. (2016). Cross-Validation. In *Encyclopedia of Database Systems* (hal. 1–7). Springer New York. https://doi.org/10.1007/978-1-4899-7993-3_565-2

Ristekdikti, P. (2015). *Permenristekdikti Nomor 44 Tahun 2015 tentang Standar Nasional Pendidikan Tinggi*. *1952*, 59–77. http://jdih.ristekdikti.go.id/?q=perundangan/konten/11343

Roihan, A., Sunarya, P. A., & Rafika, A. S. (2020). Pemanfaatan Machine Learning dalam Berbagai Bidang: Review paper. *IJCIT (Indonesian Journal on Computer and Information Technology)*, *5*(1), 75–82. https://doi.org/10.31294/ijcit.v5i1.7951

Sambodo, K. A., Rahayu, M. I., Indriasari, N., & Natsir, M. (2014). Klasifikasi Hutan-Non Hutan Data Alos Palsar Menggunakan Metode Random Forest. *Prosiding Seminar Nasional Penginderaan Jauh 2014*, 120–127. http://sinasinderaja.lapan.go.id/files/sinasja2014/prosiding/bukuprosiding_120-127.pdf

Santoso, H. B., & Subagyo, S. (2017). Peningkatan Aktifitas Dan Hasil Belajar Dengan Metode Problem Basic Learning (Pbl) Pada Mata Pelajaran Tune Up Motor Bensin Siswa Kelas Xi Di Smk Insan Cendekia Turi Sleman Tahun Ajaran 2015/2016. *Taman Vokasi*, *5*(1), 40. https://doi.org/10.30738/jtvok.v5i1.1428

Siringoringo, R. (2018). KLASIFIKASI DATA TIDAK SEIMBANG MENGGUNAKAN ALGORITMA SMOTE DAN k-NEAREST NEIGHBOR. *Jurnal ISD*, *3*(1), 44–49.

Sungkar, M. S., & Qurohman, M. T. (2021). Penerapan Algoritma C5.0 Untuk Prediksi Kelulusan Pembelajaran Mahasiswa Pada Matakuliah Arsitektur Sistem Komputer. *Jurnal Media Informatika Budidarma*, *5*(3), 1166. https://doi.org/10.30865/mib.v5i3.3116

Takdirillah, R. (2020, Agustus 19). *Apa itu Machine Learning? Beserta Pengertian*

dan Cara Kerjanya - Dicoding Blog. https://www.dicoding.com/blog/machine-learning-adalah/

Tanjung, R. H., & Kartiko. (2017). Penerapan Metode CART (Classification and Regression Trees) Untuk Menentukan Faktor-faktor yang Mempengaruhi. *Jurnal Statistika Industri dan Komputasi*, *2*(2), 78–83.

Vegari, A., & Budi, S. (2020). *Implementasi Exploratory Data Analysis Pada Dataset Video Trending Harian YouTube*. *2*(November), 397–411.

Wahyuni, E. D., Arifiyanti, A. A., & Kustyani, M. (2019). Exploratory Data Analysis dalam Konteks Klasifikasi Data Mining. *Prosiding Nasional Rekayasa Teknologi Industri dan Informasi XIV Tahun 2019 (ReTII)*, *2019*(November), 263–269. http://journal.itny.ac.id/index.php/ReTII

Widiastuti, J. (2018). *KLASIFIKASI PEMBIAYAAN WARUNG MIKRO MENGGUNAKAN METODE RANDOM FOREST DENGAN TEKNIK SAMPLING KELAS IMBALANCED (Studi Kasus: Data Nasabah Pembiayaan Warung Mikro Bank Syariah Mandiri KC Jambi)*.

LAMPIRAN

Lampiran 1 Surat Permintaan Data

KEMENTERIAN PENDIDIKAN, KEBUDAYAAN,
RISET DAN TEKNOLOGI
UNIVERSITAS NEGERI JAKARTA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
PROGRAM STUDI «PRODI2»
Kampus A, Gedung Hasjim Asj'arie Rawamangun, Jakarta Timur 13220
Telp/Fax : (021) 4894909, E-mail : dekanfmipa@unj.ac.id, www.fmipa.unj.ac.id

29 Oktober 2021

No. :704/Ilkom/DU/X/2021
Hal : Permohonan Surat Pengantar Permintaan Data
Lamp. : -

Kepada Yth.
Dra. Ellis Salsabila, M.Si
Dosen Program Studi Pendidikan Matematika
Di Tempat
Dengan Hormat,

Dengan ini kami sampaikan permohonan untuk mendapatkan Data untuk tugas Penelitian Mata Kuliah Skripsi untuk mahasiswa Program Studi Ilmu Komputer yang tersebut di bawah ini :

No	Nama	No. Reg
1	Farid Wujdi Mubarok	1313617006

Judul Skripsi : Prediksi Nilai Akhir Mata Kuliah dengan Metode SVM
Program Studi : Ilmu Komputer

Adapun data-data yang diperlukan adalah :
1. Nilai tugas semester 112, 113, 114
2. Nilai ujian tengah semester 112, 113, 114
3. Nilai ujian akhir semester 112, 113, 114

Demikian permohonan ini kami sampaikan. Atas bantuan dan kerjasamanya kami ucapkan terima kasih.

Koordinator Program Studi Ilmu Komputer

Ir. Fariani Hermin Indiyah, MT.
NIP. 19600211 198703 2 001

Lampiran 2 *Datasets*

No	Jenis Kelamin	SKS yang telah ditempuh	Prodi	Angkatan	Mengulang Matkul Kalin	Kehadiran	Kehadiran (presentase)	Tgs K1	Tgs K2	Tgs K3	Tgs K4	Tgs K5	Tgs K6	NRK	Tgs Individu	Tgs 1	UTS	UAS	NA	NL.HRF
1	L	21	Ilmu Komputer	2019	Tidak	13	100.00	75.00	98.00	95.00				89.33	2	99.33	95.00	76.79	71.34	B
2	L	21	Ilmu Komputer	2019	Tidak	12	92.31	100.00	85.00	95.00				93.33	3	100	87.50	81.96	70.84	B-
3	L	21	Ilmu Komputer	2019	Tidak	13	100.00	85.00	85.00	100.00				90.00	5	100	45.00	38.39	45.02	E
4	P	21	Ilmu Komputer	2019	Tidak	13	100.00	100.00	100.00	100.00				100.00	2	100	95.00	90.06	75.52	B
5	L	21	Ilmu Komputer	2019	Tidak	13	100.00	80.00	100.00	100.00				93.33	1	98.33	90.00	78.63	70.26	B-
6	L	21	Ilmu Komputer	2019	Tidak	13	100.00	100.00	100.00	95.00				98.33	3	100	85.00	84.58	70.88	B-
7	L	21	Ilmu Komputer	2019	Tidak	13	100.00	100.00	100.00	100.00				100.00	1	100	85.00	89.05	72.21	B
8	P	21	Ilmu Komputer	2019	Tidak	13	100.00	100.00	100.00	100.00				100.00	4	100	85.00	95.95	74.29	B
9	L	21	Ilmu Komputer	2019	Tidak	13	100.00	85.00	100.00	95.00				93.33	4	100	87.50	87.02	72.36	B
10	L	21	Ilmu Komputer	2019	Tidak	13	100.00	75.00	98.00	90.00				87.67	4	100	87.50	80.83	70.50	B-
11	L	21	Ilmu Komputer	2019	Tidak	13	100.00	100.00	85.00	95.00				93.33	0	93.33	85.00	83.33	69.17	B-
12	P	21	Ilmu Komputer	2019	Tidak	13	100.00	80.00	85.00	95.00				86.67	3	100	100.00	92.74	77.82	B+
13	L	21	Ilmu Komputer	2019	Tidak	13	100.00	75.00	98.00	90.00				87.67	3	100	87.50	79.88	70.21	B-
14	L	21	Ilmu Komputer	2019	Tidak	13	100.00	100.00	85.00	100.00				95.00	2	100	90.00	79.88	70.96	B-
15	L	21	Ilmu Komputer	2019	Tidak	13	100.00	75.00	98.00	90.00				87.67	1	92.67	75.00	61.96	59.62	C
16	L	21	Ilmu Komputer	2019	Tidak	12	92.31	100.00	98.00	95.00				97.67	1	100	95.00	83.45	73.54	B
17	L	21	Ilmu Komputer	2019	Tidak	13	100.00	100.00	98.00	95.00				97.67	2	100	95.00	85.77	74.23	B
18	L	21	Ilmu Komputer	2019	Tidak	11	84.62	75.00	100.00	100.00				91.67	0	91.67	85.00	76.37	66.74	B-
19	L	21	Ilmu Komputer	2019	Tidak	13	100.00	100.00	100.00	100.00				100.00	0	100.00	85.00	77.32	68.70	B-
20	L	21	Ilmu Komputer	2019	Tidak	13	100.00	85.00	100.00	95.00				93.33	1	98.33	72.50	75.12	63.95	C+
21	L	21	Ilmu Komputer	2019	Tidak	13	100.00	85.00	85.00	100.00				90.00	2	100.00	70.00	82.62	65.79	C+
22	L	21	Ilmu Komputer	2019	Tidak	12	92.31	0.00	40.00	100.00				70.00	1	75.00	65.00	52.38	50.21	D
23	L	21	Ilmu Komputer	2019	Tidak	13	100.00	0.00	40.00	100.00				70.00	0	70.00	85.00	78.87	63.16	C+
24	L	21	Ilmu Komputer	2019	Tidak	12	92.31	0.00	40.00	100.00				70.00	1	75.00	75.00	88.69	64.11	C+
25	L	21	Ilmu Komputer	2019	Tidak	13	100.00	0.00	40.00	100.00				70.00	0	70.00	85.00	82.44	64.23	C+
26	L	19	Pend Mat	2019	Tidak	16	100.00	95.00	100.00	95.00	100.00	98.00	96.00	97.33	5	100	92.50	80.83	92.00	A
27	L	19	Pend Mat	2019	Tidak	16	100.00	85.00	100.00	100.00	95.00	100.00	96.00	96.00	3	100	85.00	85.36	91.11	A
28	P	19	Pend Mat	2019	Tidak	16	100.00	85.00	80.00	85.00	100.00	90.00	95.00	89.17	0	89.17	95.00	80.71	88.39	A
29	P	19	Pend Mat	2019	Tidak	16	100.00	100.00	98.00	100.00	100.00	90.00	90.00	96.33	2	100	80.00	78.75	87.63	A
30	P	19	Pend Mat	2019	Tidak	16	100.00	85.00	80.00	85.00	100.00	90.00	95.00	89.17	3	100	100.00	87.68	96.30	A
31	L	19	Pend Mat	2019	Tidak	15	93.75	85.00	100.00	100.00	95.00	100.00	96.00	96.00	0	96.00	92.50	90.77	93.38	A
32	P	19	Pend Mat	2019	Tidak	16	100.00	95.00	100.00	95.00	80.00	95.00	95.00	93.33	5	100	95.00	94.52	96.86	A
33	P	19	Pend Mat	2019	Tidak	16	100.00	85.00	80.00	85.00	100.00	90.00	95.00	89.17	1	94.17	100.00	83.81	92.82	A
34	L	19	Pend Mat	2019	Tidak	16	100.00	100.00	100.00	95.00	100.00	95.00	92.00	97.00	0	97.00	87.50	86.19	90.91	A
35	P	19	Pend Mat	2019	Tidak	16	100.00	95.00	100.00	95.00	80.00	95.00	95.00	93.33	1	98.33	92.50	89.94	94.05	A
36	P	19	Pend Mat	2019	Tidak	16	100.00	85.00	95.00	90.00	90.00	90.00	90.00	90.00	3	100	92.50	85.60	93.43	A
37	P	19	Pend Mat	2019	Tidak	16	100.00	75.00	100.00	95.00	100.00	95.00	95.00	93.33	1	98.33	100.00	86.25	95.20	A
38	P	19	Pend Mat	2019	Tidak	16	100.00	85.00	100.00	100.00	95.00	100.00	96.00	96.00	4	100	92.50	87.92	94.13	A
39	L	19	Pend Mat	2019	Tidak	16	100.00	100.00	100.00	95.00	100.00	95.00	92.00	97.00	1	100	85.00	84.88	90.96	A
40	P	19	Pend Mat	2019	Tidak	16	100.00	100.00	98.00	100.00	100.00	90.00	90.00	96.33	2	100	87.50	79.70	90.16	A
41	L	19	Pend Mat	2019	Tidak	16	100.00	95.00	100.00	95.00	100.00	98.00	96.00	97.33	1	100	77.50	82.08	87.87	A
42	P	19	Pend Mat	2019	Tidak	16	100.00	75.00	100.00	95.00	100.00	95.00	95.00	93.33	3	100	85.00	79.94	89.48	A
43	P	19	Pend Mat	2019	Tidak	16	100.00	95.00	100.00	95.00	80.00	95.00	95.00	93.33	0	93.33	67.50	81.55	82.04	A-
44	L	19	Pend Mat	2019	Tidak	16	100.00	75.00	100.00	90.00	90.00	95.00	90.00	90.00	4	100	87.50	81.37	90.66	A
45	P	19	Pend Mat	2019	Tidak	16	100.00	95.00	100.00	95.00	80.00	95.00	95.00	93.33	5	100	92.50	86.55	93.72	A
46	L	19	Pend Mat	2019	Tidak	16	100.00	75.00	100.00	90.00	90.00	95.00	90.00	90.00	6	100	95.00	88.69	95.11	A
47	L	19	Pend Mat	2019	Tidak	15	93.75	95.00	100.00	95.00	100.00	98.00	96.00	97.33	0	97.33	72.50	89.17	87.42	A
48	L	19	Pend Mat	2019	Tidak	16	100.00	100.00	100.00	95.00	100.00	95.00	92.00	97.00	3	100	87.50	87.44	92.48	A
49	P	19	Pend Mat	2019	Tidak	16	100.00	75.00	100.00	95.00	100.00	95.00	95.00	93.33	0	93.33	80.00	75.30	83.91	A-
50	L	19	Pend Mat	2019	Tidak	16	100.00	75.00	100.00	90.00	90.00	95.00	90.00	90.00	3	100	67.50	81.37	84.66	A-
51	P	19	Pend Mat	2019	Tidak	16	100.00	75.00	100.00	95.00	100.00	95.00	95.00	93.33	0	93.33	100.00	82.26	92.00	A

No	Jenis Kelamin	SKS yang telah ditempuh	Prodi	Angkatan	Mengulang Matkul Kalin	Kehadiran	Kehadiran (presentase)	Tgs K1	Tgs K2	Tgs K3	Tgs K4	Tgs K5	Tgs K6	NRK	Tgs Individu	Tgs 1	UTS	UAS	NA	NL.HRF
52	P	19	Pend Mat	2019	Tidak	16	100.00	75.00	98.00	100.00	90.00	95.00	95.00	92.17	1	97.17	70.00	86.01	85.68	A-
53	P	19	Pend Mat	2019	Tidak	15	93.75	100.00	98.00	100.00	100.00	90.00	90.00	96.33	0	96.33	80.00	93.27	90.50	A
54	P	19	Pend Mat	2019	Tidak	15	93.75	75.00	98.00	100.00	90.00	95.00	95.00	92.17	1	97.17	92.50	76.67	89.63	A
55	P	19	Pend Mat	2019	Tidak	16	100.00	100.00	98.00	100.00	100.00	90.00	90.00	96.33	1	100	92.50	85.89	93.52	A
56	P	19	Pend Mat	2019	Tidak	16	100.00	75.00	98.00	100.00	90.00	95.00	95.00	92.17	4	100	92.50	91.79	95.29	A
57	P	19	Pend Mat	2019	Tidak	16	100.00	85.00	100.00	100.00	95.00	100.00	96.00	96.00	0	96.00	92.50	82.20	90.81	A
58	L	19	Pend Mat	2019	Tidak	16	100.00	95.00	100.00	95.00	100.00	98.00	96.00	97.33	1	100	62.50	78.27	82.23	A-
59	P	19	Pend Mat	2019	Tidak	16	100.00	85.00	95.00	90.00	90.00	90.00	90.00	90.00	1	95.00	92.50	83.81	90.89	A
60	P	19	Pend Mat	2019	Tidak	16	100.00	75.00	98.00	100.00	90.00	95.00	95.00	92.17	4	100	77.50	92.50	91.00	A
61	P	19	Pend Mat	2019	Tidak	16	100.00	85.00	95.00	90.00	90.00	90.00	90.00	90.00	1	95.00	92.50	85.77	91.48	A
62	P	19	Pend Mat	2019	Tidak	16	100.00	85.00	80.00	85.00	100.00	90.00	95.00	89.17	1	94.17	100.00	84.94	93.16	A
63	P	19	Pend Mat	2019	Tidak	15	93.75	85.00	95.00	90.00	90.00	90.00	90.00	90.00	2	100.00	85.00	84.35	90.81	A
64	P	22	Matematika	2019	Tidak	30	100.00	100.00	85.00	95.00				93.33	2	100	85.00	74.23	67.77	B-
65	L	22	Matematika	2019	Tidak	30	100.00	100.00	100.00	95.00				98.33	2	100	95.00	83.04	73.41	B
66	P	22	Matematika	2019	Tidak	30	100.00	100.00	100.00	95.00				98.33	3	100	75.00	90.00	69.50	B-
67	P	22	Matematika	2019	Tidak	30	100.00	100.00	85.00	95.00				93.33	2	100	85.00	92.92	73.38	B
68	L	22	Matematika	2019	Tidak	29	96.67	100.00	85.00	95.00				93.33	1	98.33	65.00	54.76	55.59	C-
69	P	22	Matematika	2019	Tidak	30	100.00	95.00	100.00	95.00				96.67	3	100	95.00	85.89	74.27	B
70	P	22	Matematika	2019	Tidak	30	100.00	100.00	85.00	95.00				93.33	3	100	77.50	83.69	68.36	B-
71	P	22	Matematika	2019	Tidak	29	96.67	100.00	100.00	85.00				95.00	2	100	90.00	86.73	73.02	B
72	L	22	Matematika	2019	Tidak	30	100.00	95.00	100.00	95.00				96.67	2	100	55.00	76.61	59.48	C
73	P	22	Matematika	2019	Tidak	29	96.67	95.00	100.00	95.00				96.67	2	100	85.00	74.76	67.93	B-
74	L	22	Matematika	2019	Tidak	29	96.67	98.00	100.00	90.00				96.00	2	100	80.00	76.19	66.86	B-
75	P	22	Matematika	2019	Tidak	29	96.67	95.00	100.00	95.00				96.67	2	100	45.00	90.24	60.57	C
76	P	22	Matematika	2019	Tidak	30	100.00	95.00	85.00	95.00				91.67	2	100	85.00	86.61	71.48	B
77	P	22	Matematika	2019	Tidak	29	96.67	100.00	85.00	100.00				95.00	1	100.00	75.00	91.19	69.86	B-
78	L	22	Matematika	2019	Tidak	30	100.00	100.00	85.00	100.00				95.00	1	100.00	72.50	89.52	68.61	B-
79	P	22	Matematika	2019	Tidak	30	100.00	100.00	100.00	95.00				98.33	2	100.00	70.00	90.36	68.11	B-
80	P	22	Matematika	2019	Tidak	30	100.00	100.00	85.00	95.00				93.33	2	100	90.00	76.79	70.04	B-
81	L	22	Matematika	2019	Tidak	29	96.67	95.00	100.00	95.00				96.67	1	100	85.00	56.49	62.45	C+
82	P	22	Matematika	2019	Tidak	30	100.00	95.00	100.00	95.00				96.67	2	100	75.00	90.30	69.59	B-
83	P	22	Matematika	2019	Tidak	30	100.00	100.00	100.00	85.00				95.00	2	100	70.00	92.74	68.82	B-
84	L	22	Matematika	2019	Tidak	30	100.00	100.00	100.00	85.00				95.00	2	100	70.00	88.39	67.52	B-
85	P	22	Matematika	2019	Tidak	30	100.00	100.00	100.00	95.00				98.33	2	100	72.50	89.52	68.61	B-
86	P	22	Matematika	2019	Tidak	30	100.00	100.00	85.00	95.00				93.33	2	100	55.00	64.40	55.82	C-
87	L	22	Matematika	2019	Tidak	30	100.00	100.00	85.00	100.00				95.00	2	100	85.00	75.12	68.04	B-
88	P	22	Matematika	2019	Tidak	30	100.00	98.00	100.00	90.00				96.00	3	100	80.00	83.45	69.04	B-
89	P	22	Matematika	2019	Tidak	30	100.00	95.00	100.00	95.00				96.67	1	100	67.50	87.92	66.63	B-
90	P	22	Matematika	2019	Tidak	30	100.00	98.00	100.00	90.00				96.00	2	100	70.00	90.77	68.23	B-
91	P	22	Matematika	2019	Tidak	30	100.00	100.00	100.00	95.00				98.33	1	100	62.50	82.56	63.52	C+
92	P	22	Matematika	2019	Tidak	28	93.33	100.00	100.00	85.00				95.00	2	100	72.50	86.31	67.64	B-
93	P	22	Matematika	2019	Tidak	29	96.67	95.00	100.00	95.00				96.67	1	100	62.50	64.88	58.21	C
94	P	22	Matematika	2019	Tidak	30	100.00	100.00	85.00	100.00				95.00	2	100	80.00	74.94	66.48	B-
95	P	22	Matematika	2019	Tidak	29	96.67	100.00	85.00	95.00				93.33	1	98.33	87.50	87.20	72.08	B
96	L	22	Matematika	2019	Tidak	30	100.00	100.00	85.00	95.00				93.33	3	100	95.00	88.81	75.14	B
97	P	22	Matematika	2019	Tidak	30	100.00	95.00	85.00	95.00				91.67	2	100	92.50	77.56	71.02	B
98	L	22	Matematika	2019	Tidak	30	100.00	100.00	100.00	95.00				98.33	1	100	90.00	92.92	74.88	B
99	P	22	Matematika	2019	Tidak	28	93.33	100.00	100.00	95.00				98.33	1	100	72.50	83.45	66.79	B-
100	P	22	Matematika	2019	Tidak	30	100.00	98.00	100.00	90.00				96.00	2	100	80.00	77.02	67.11	B-
101	L	22	Matematika	2019	Tidak	30	100.00	95.00	85.00	95.00				91.67	1	96.67	80.00	79.82	67.28	B-
102	L	21	Ilmu Komputer	2020	Tidak	15	93.75	85.00	82.00	82.00				83.00	0	83.00	55.00	0.00	33.10	E

No	Jenis Kelamin	SKS yang telah ditempuh	Prodi	Angkatan	Mengulang Matkul Kafin	Kehadiran	Kehadiran (presentase)	Tgs K1	Tgs K2	Tgs K3	Tgs K4	Tgs K5	Tgs K6	NRK	Tgs Individu	Tgs 1	UTS	UAS	NA	NL.HRF
103	L	21	Ilmu Komputer	2020	Tidak	16	100.00	80.00	85.00	80.00				81.67	0	81.67	62.50	0.00	35.09	E
104	P	21	Ilmu Komputer	2020	Tidak	16	100.00	85.00	92.00	90.00				89.00	0	89.00	67.50	0.00	38.05	E
105	P	21	Ilmu Komputer	2020	Tidak	16	100.00	90.00	90.00	87.00				89.00	0	89.00	57.50	0.00	35.05	E
106	L	21	Ilmu Komputer	2020	Tidak	15	93.75	85.00	92.00	90.00				89.00	0	89.00	67.50	0.00	38.05	E
107	L	21	Ilmu Komputer	2020	Tidak	15	93.75	90.00	90.00	85.00				88.33	0	88.33	67.50	0.00	37.91	E
108	L	21	Ilmu Komputer	2020	Tidak	15	93.75	80.00	85.00	80.00				81.67	0	81.67	60.00	0.00	34.34	E
109	P	21	Ilmu Komputer	2020	Tidak	16	100.00	85.00	87.00	90.00				87.33	0	87.33	70.00	0.00	38.46	E
110	L	21	Ilmu Komputer	2020	Tidak	13	81.25	90.00	92.00	92.00				91.33	0	91.33	55.00	0.00	34.76	E
111	L	21	Ilmu Komputer	2020	Tidak	15	93.75	85.00	87.00	50.00				74.00	0	74.00	60.00	0.00	32.80	E
112	L	21	Ilmu Komputer	2020	Tidak	15	93.75	90.00	90.00	87.00				89.00	0	89.00	85.00	0.00	43.30	E
113	L	21	Ilmu Komputer	2020	Tidak	14	87.50	90.00	92.00	92.00				91.33	1	96.33	72.50	0.00	41.01	E
114	L	21	Ilmu Komputer	2020	Tidak	15	93.75	85.00	90.00	87.00				87.33	0	87.33	65.00	0.00	36.96	E
115	L	21	Ilmu Komputer	2020	Tidak	15	93.75	85.00	85.00	85.00				85.00	1	90.00	77.50	0.00	41.25	E
116	L	21	Ilmu Komputer	2020	Tidak	16	100.00	90.00	92.00	92.00				91.33	0	91.33	60.00	0.00	36.26	E
117	L	21	Ilmu Komputer	2020	Tidak	16	100.00	85.00	87.00	90.00				87.33	0	87.33	62.50	0.00	36.21	E
118	L	21	Ilmu Komputer	2020	Tidak	16	100.00	90.00	90.00	85.00				88.33	0	88.33	70.00	0.00	38.66	E
119	L	21	Ilmu Komputer	2020	Tidak	14	87.50	85.00	90.00	82.00				85.67	0	85.67	45.00	0.00	30.64	E
120	L	21	Ilmu Komputer	2020	Tidak	16	100.00	85.00	90.00	87.00				87.33	0	87.33	52.50	0.00	33.21	E
121	L	21	Ilmu Komputer	2020	Tidak	15	93.75	85.00	87.00	50.00				74.00	0	74.00	67.50	0.00	35.05	E
122	P	21	Ilmu Komputer	2020	Tidak	14	87.50	85.00	85.00	87.00				85.67	1	90.67	67.50	0.00	38.39	E
123	L	21	Ilmu Komputer	2020	Tidak	16	100.00	85.00	92.00	90.00				89.00	5	100	62.50	0.00	38.75	E
124	L	21	Ilmu Komputer	2020	Tidak	14	87.50	90.00	90.00	87.00				89.00	0	89.00	67.50	0.00	38.05	E
125	P	21	Ilmu Komputer	2020	Tidak	15	93.75	85.00	92.00	90.00				89.00	0	89.00	70.00	0.00	38.80	E
126	L	21	Ilmu Komputer	2020	Tidak	16	100.00	85.00	92.00	90.00				89.00	5	100	60.00	0.00	38.00	E
127	L	21	Ilmu Komputer	2020	Tidak	16	100.00	85.00	90.00	87.00				87.33	2	97.33	40.00	0.00	31.46	E
128	L	21	Ilmu Komputer	2020	Tidak	16	100.00	85.00	85.00	87.00				85.67	0	85.67	60.00	0.00	35.14	E
129	P	21	Ilmu Komputer	2020	Tidak	15	93.75	90.00	90.00	85.00				88.33	0	88.33	50.00	0.00	32.66	E
130	P	21	Ilmu Komputer	2020	Tidak	13	81.25	85.00	87.00	50.00				74.00	0	74.00	70.00	0.00	35.80	E
131	L	21	Ilmu Komputer	2020	Tidak	16	100.00	85.00	85.00	85.00				85.00	0	85.00	75.00	0.00	39.50	E
132	L	21	Ilmu Komputer	2020	Tidak	15	93.75	85.00	90.00	82.00				85.67	0	85.67	72.50	0.00	38.89	E
133	L	21	Ilmu Komputer	2020	Tidak	14	87.50	87.00	87.00	85.00				86.33	0	86.33	45.00	0.00	30.76	E
134	L	21	Ilmu Komputer	2020	Tidak	15	93.75	85.00	87.00	90.00				87.33	5	100	80.00	0.00	44.00	E
135	L	21	Ilmu Komputer	2020	Tidak	15	93.75	85.00	85.00	85.00				85.00	0	85.00	65.00	0.00	36.50	E
136	L	21	Ilmu Komputer	2020	Tidak	15	93.75	80.00	85.00	80.00				81.67	0	81.67	75.00	0.00	38.84	E
137	L	21	Ilmu Komputer	2020	Tidak	15	93.75	85.00	82.00	82.00				83.00	0	83.00	72.50	0.00	38.35	E
138	L	21	Ilmu Komputer	2020	Tidak	13	81.25	85.00	90.00	82.00				85.67	0	85.67	62.50	0.00	35.89	E
139	L	21	Ilmu Komputer	2020	Tidak	13	81.25	85.00	85.00	87.00				85.67	0	85.67	72.50	0.00	38.89	E
140	P	21	Ilmu Komputer	2020	Tidak	15	93.75	85.00	82.00	82.00				83.00	1	88.00	60.00	0.00	35.60	E
141	L	21	Ilmu Komputer	2020	Tidak	13	81.25	87.00	87.00	85.00				86.33	1	91.33	70.00	0.00	39.26	E
142	P	21	Ilmu Komputer	2020	Tidak	11	68.75	87.00	87.00	85.00				86.33	0	86.33	45.00	0.00	30.76	E
143	L	21	Ilmu Komputer	2020	Tidak	13	81.25	87.00	87.00	85.00				86.33	0	86.33	57.50	0.00	34.51	E
144	L	21	Ilmu Komputer	2020	Tidak	16	100.00	85.00	92.00	90.00				89.00	0	89.00	65.00	0.00	37.30	E
145	P	22	Pend Mat	2020	Tidak	16	100.00	90.00	92.00	92.00	90.00	92.00	92.00	91.33	2	100.00	82.50	80.00	88.75	A
146	P	22	Pend Mat	2020	Tidak	16	100.00	90.00	95.00	93.00	92.00	95.00	95.00	93.33	0	93.33	55.00	40.00	65.82	C+
147	P	22	Pend Mat	2020	Tidak	16	100.00	90.00	95.00	93.00	92.00	95.00	95.00	93.33	0	93.33	77.50	80.00	84.57	A-
148	P	22	Pend Mat	2020	Tidak	16	100.00	90.00	93.00	92.00	90.00	95.00	93.00	92.17	1	97.17	62.50	62.50	76.38	B+
149	L	22	Pend Mat	2020	Tidak	16	100.00	93.00	95.00	93.00	95.00	95.00	93.00	94.00	3	100.00	90.00	90.00	94.00	A
150	P	22	Pend Mat	2020	Tidak	16	100.00	90.00	92.00	90.00	92.00	90.00	92.00	91.00	0	91.00	90.00	60.00	81.40	A-
151	L	22	Pend Mat	2020	Tidak	16	100.00	90.00	92.00	93.00	90.00	93.00	93.00	91.83	2	100.00	82.50	80.00	88.75	A
152	P	22	Pend Mat	2020	Tidak	16	100.00	90.00	92.00	90.00	92.00	90.00	92.00	91.00	2	100.00	60.00	67.50	78.25	B+
153	L	22	Pend Mat	2020	Tidak	16	100.00	90.00	92.00	90.00	92.00	92.00	93.00	91.50	1	96.50	60.00	75.00	79.10	B+

No	Jenis Kelamin	SKS yang telah ditempuh	Prodi	Angkatan	Mengulang Matkul Kelin	Kehadiran	Kehadiran (presentase)	Tgs K1	Tgs K2	Tgs K3	Tgs K4	Tgs K5	Tgs K6	NRK	Tgs Individu	Tgs 1	UTS	UAS	NA	NILHRF
154	P	22	Pend Mat	2020	Tidak	16	100.00	87.00	90.00	92.00	87.00	90.00	92.00	89.67	0	89.67	90.00	70.00	83.88	A-
155	P	22	Pend Mat	2020	Tidak	16	100.00	90.00	92.00	93.00	90.00	93.00	93.00	91.83	3	100.00	65.00	70.00	80.50	B+
156	P	22	Pend Mat	2020	Tidak	16	100.00	90.00	92.00	92.00	90.00	90.00	92.00	91.00	2	100.00	90.00	75.00	89.50	A
157	P	22	Pend Mat	2020	Tidak	16	100.00	90.00	93.00	92.00	90.00	95.00	93.00	92.17	0	92.17	52.50	60.00	70.63	B-
158	P	22	Pend Mat	2020	Tidak	16	100.00	90.00	92.00	90.00	92.00	92.00	93.00	91.50	1	96.50	75.00	75.00	83.60	A-
159	P	22	Pend Mat	2020	Tidak	16	100.00	90.00	93.00	92.00	90.00	95.00	93.00	92.17	1	97.17	90.00	70.00	86.88	A
160	P	22	Pend Mat	2020	Tidak	16	100.00	87.00	92.00	90.00	90.00	90.00	92.00	90.17	0	90.17	82.50	80.00	84.83	A-
161	L	22	Pend Mat	2020	Tidak	16	100.00	90.00	92.00	92.00	90.00	90.00	92.00	91.00	0	91.00	40.00	65.00	67.90	B-
162	P	22	Pend Mat	2020	Tidak	16	100.00	90.00	92.00	92.00	92.00	90.00	93.00	91.50	1	96.50	57.50	80.00	79.85	B+
163	L	22	Pend Mat	2020	Tidak	16	100.00	87.00	92.00	90.00	90.00	90.00	92.00	90.17	1	95.17	57.50	75.00	77.83	B+
164	P	22	Pend Mat	2020	Tidak	16	100.00	90.00	92.00	90.00	92.00	90.00	92.00	91.00	0	91.00	65.00	75.00	78.40	B+
165	L	22	Pend Mat	2020	Tidak	16	100.00	90.00	92.00	92.00	92.00	90.00	93.00	91.50	4	100.00	57.50	80.00	81.25	A-
166	P	22	Pend Mat	2020	Tidak	16	100.00	90.00	95.00	93.00	92.00	95.00	95.00	93.33	1	98.33	80.00	80.00	87.32	A
167	P	22	Pend Mat	2020	Tidak	16	100.00	87.00	92.00	92.00	87.00	90.00	92.00	90.00	0	90.00	67.50	40.00	68.25	B-
168	L	22	Pend Mat	2020	Tidak	16	100.00	90.00	92.00	92.00	90.00	92.00	92.00	91.33	0	91.33	90.00	70.00	84.52	A-
169	L	22	Pend Mat	2020	Tidak	16	100.00	87.00	90.00	92.00	87.00	90.00	92.00	89.67	0	89.67	85.00	67.50	81.63	A-
170	P	22	Pend Mat	2020	Tidak	15	93.75	90.00	95.00	93.00	92.00	92.00	93.00	92.50	1	97.50	75.00	65.00	81.00	A-
171	P	22	Pend Mat	2020	Tidak	15	93.75	90.00	95.00	93.00	92.00	92.00	93.00	92.50	1	97.50	70.00	67.50	80.25	B+
172	L	22	Pend Mat	2020	Tidak	16	100.00	93.00	95.00	93.00	95.00	95.00	93.00	94.00	1	99.00	85.00	85.00	90.60	A
173	P	22	Pend Mat	2020	Tidak	16	100.00	90.00	92.00	92.00	90.00	90.00	92.00	91.00	1	96.00	82.50	77.50	86.40	A
174	P	22	Pend Mat	2020	Tidak	16	100.00	90.00	92.00	92.00	90.00	92.00	92.00	91.33	0	91.33	52.50	70.00	73.27	B
175	L	22	Pend Mat	2020	Tidak	12	75.00	90.00	95.00	93.00	92.00	92.00	93.00	92.50	0	92.50	57.50	55.00	70.75	B-
176	P	22	Pend Mat	2020	Tidak	16	100.00	93.00	95.00	93.00	95.00	95.00	93.00	94.00	0	94.00	85.00	75.00	85.60	A-
177	P	22	Pend Mat	2020	Tidak	16	100.00	87.00	92.00	92.00	87.00	90.00	92.00	90.00	0	90.00	82.50	72.50	82.50	A-
178	L	22	Pend Mat	2020	Tidak	16	100.00	87.00	92.00	90.00	90.00	90.00	92.00	90.17	0	90.17	85.00	75.00	84.08	A-
179	P	22	Pend Mat	2020	Tidak	16	100.00	90.00	92.00	90.00	92.00	92.00	93.00	91.50	0	91.50	87.50	80.00	86.85	A
180	L	22	Pend Mat	2020	Tidak	16	100.00	90.00	92.00	93.00	90.00	93.00	93.00	91.83	2	100.00	72.50	77.50	85.00	A-
181	P	22	Pend Mat	2020	Tidak	16	100.00	87.00	92.00	90.00	90.00	90.00	92.00	90.17	1	95.17	90.00	65.00	84.58	A-
182	P	22	Pend Mat	2020	Tidak	16	100.00	87.00	92.00	90.00	90.00	90.00	92.00	90.17	0	90.17	77.50	65.00	78.83	B+
183	P	22	Pend Mat	2020	Tidak	16	100.00	87.00	90.00	92.00	87.00	90.00	92.00	89.67	0	89.67	82.50	47.50	74.88	B
184	L	22	Pend Mat	2020	Tidak	16	100.00	87.00	90.00	92.00	87.00	90.00	92.00	89.67	0	89.67	75.00	55.00	74.88	B
185	P	22	Pend Mat	2020	Tidak	16	100.00	87.00	92.00	90.00	90.00	90.00	92.00	90.17	2	100.00	70.00	60.00	79.00	B+
186	P	22	Pend Mat	2020	Tidak	15	93.75	90.00	95.00	93.00	92.00	92.00	93.00	92.50	0	92.50	42.50	70.00	70.75	B-
187	P	22	Pend Mat	2020	Tidak	16	100.00	90.00	92.00	92.00	92.00	90.00	93.00	91.50	1	96.50	50.00	47.50	67.85	B-
188	P	22	Pend Mat	2020	Tidak	16	100.00	87.00	92.00	92.00	87.00	90.00	92.00	90.00	0	90.00	87.50	60.00	80.25	B+

Lampiran 3 Hasil *encoding* data latih

L	P	Ilmu Komputer	Matematika	Pend Mat	2019	2020
0	1	0	1	0	1	0
0	1	0	0	1	1	0
0	1	1	0	0	0	1
1	0	1	0	0	0	1
0	1	0	0	1	1	0
0	1	0	0	1	1	0
0	1	0	0	1	1	0
0	1	0	0	1	0	1
1	0	1	0	0	0	1
0	1	0	0	1	0	1
0	1	0	0	1	0	1
1	0	1	0	0	1	0
1	0	1	0	0	0	1
0	1	0	1	0	1	0
0	1	0	0	1	0	1
0	1	0	0	1	0	1
0	1	1	0	0	0	1
1	0	1	0	0	0	1
1	0	1	0	0	1	0
1	0	1	0	0	0	1
1	0	0	0	1	0	1
1	0	0	0	1	1	0
0	1	0	0	1	0	1
0	1	0	0	1	1	0
0	1	0	0	1	0	1
0	1	0	0	1	0	1
0	1	0	0	1	0	1
0	1	0	1	0	1	0
0	1	0	1	0	1	0
0	1	0	0	1	0	1
1	0	0	1	0	1	0
0	1	0	0	1	0	1
0	1	0	1	0	1	0
1	0	1	0	0	1	0
1	0	1	0	0	0	1
1	0	1	0	0	1	0
0	1	0	1	0	1	0
0	1	0	1	0	1	0
0	1	0	0	1	0	1
1	0	1	0	0	1	0
1	0	0	0	1	0	1
1	0	1	0	0	1	0

L	P	Ilmu Komputer	Matematika	Pend Mat	2019	2020
1	0	0	0	1	0	1
1	0	0	1	0	1	0
0	1	0	1	0	1	0
0	1	0	0	1	1	0
1	0	1	0	0	0	1
0	1	0	1	0	1	0
1	0	1	0	0	0	1
1	0	0	0	1	1	0
0	1	0	0	1	1	0
0	1	0	1	0	1	0
0	1	0	1	0	1	0
1	0	0	0	1	0	1
0	1	0	0	1	0	1
0	1	0	1	0	1	0
1	0	0	0	1	0	1
1	0	1	0	0	0	1
1	0	1	0	0	0	1
1	0	1	0	0	1	0
0	1	0	0	1	1	0
0	1	0	0	1	1	0
0	1	0	0	1	0	1
1	0	0	0	1	1	0
1	0	0	1	0	1	0
0	1	0	0	1	0	1
0	1	0	0	1	0	1
0	1	0	0	1	1	0
1	0	0	1	0	1	0
1	0	1	0	0	0	1
1	0	1	0	0	1	0
1	0	1	0	0	0	1
1	0	1	0	0	1	0
1	0	0	0	1	0	1
0	1	1	0	0	0	1
0	1	0	0	1	0	1
1	0	1	0	0	0	1
1	0	0	0	1	0	1
1	0	0	1	0	1	0
1	0	1	0	0	1	0
0	1	0	0	1	1	0
0	1	0	0	1	0	1
1	0	1	0	0	0	1
1	0	0	1	0	1	0
0	1	0	0	1	1	0
1	0	0	0	1	1	0

L	P	Ilmu Komputer	Matematika	Pend Mat	2019	2020
0	1	0	1	0	1	0
1	0	0	0	1	1	0
1	0	1	0	0	1	0
0	1	1	0	0	1	0
0	1	0	0	1	0	1
1	0	0	0	1	1	0
1	0	1	0	0	0	1
1	0	0	0	1	0	1
1	0	0	0	1	1	0
0	1	1	0	0	0	1
1	0	1	0	0	0	1
0	1	0	0	1	0	1
1	0	1	0	0	1	0
0	1	0	0	1	1	0
0	1	1	0	0	0	1
1	0	1	0	0	1	0
1	0	0	0	1	1	0
1	0	1	0	0	1	0
0	1	0	0	1	1	0
0	1	0	1	0	1	0
0	1	1	0	0	1	0
0	1	0	0	1	0	1
0	1	1	0	0	0	1
1	0	0	1	0	1	0
0	1	0	1	0	1	0
0	1	0	1	0	1	0
0	1	0	0	1	1	0
1	0	1	0	0	0	1
0	1	0	0	1	0	1
0	1	0	0	1	0	1
0	1	1	0	0	0	1
0	1	0	0	1	0	1
1	0	1	0	0	0	1
1	0	1	0	0	1	0
1	0	0	0	1	1	0
1	0	1	0	0	0	1
1	0	0	1	0	1	0
0	1	1	0	0	0	1
0	1	0	0	1	0	1
0	1	0	1	0	1	0
1	0	1	0	0	0	1
1	0	1	0	0	0	1
0	1	0	1	0	1	0
1	0	1	0	0	0	1

L	P	Ilmu Komputer	Matematika	Pend Mat	2019	2020
0	1	0	0	1	1	0

Lampiran 4 Hasil *encoding* data tes

L	P	Ilmu Komputer	Matematika	Pend Mat	2019	2020
0	1	0	0	1	1	0
0	1	0	1	0	1	0
1	0	1	0	0	0	1
1	0	1	0	0	0	1
0	1	0	1	0	1	0
1	0	0	0	1	0	1
0	1	0	0	1	1	0
1	0	1	0	0	0	1
0	1	0	1	0	1	0
0	1	0	1	0	1	0
0	1	1	0	0	0	1
1	0	1	0	0	1	0
1	0	0	1	0	1	0
0	1	0	0	1	0	1
1	0	1	0	0	0	1
1	0	0	1	0	1	0
1	0	1	0	0	0	1
1	0	0	0	1	1	0
1	0	1	0	0	1	0
1	0	0	0	1	1	0
0	1	0	0	1	1	0
1	0	0	1	0	1	0
0	1	0	0	1	0	1
1	0	0	0	1	0	1
0	1	0	0	1	1	0
0	1	0	0	1	1	0
0	1	1	0	0	1	0
1	0	1	0	0	0	1
0	1	0	0	1	0	1
0	1	0	0	1	1	0
1	0	1	0	0	1	0
1	0	1	0	0	0	1
0	1	0	0	1	0	1
1	0	1	0	0	1	0
0	1	0	1	0	1	0
0	1	0	1	0	1	0
1	0	1	0	0	0	1
0	1	0	1	0	1	0

L	P	Ilmu Komputer	Matematika	Pend Mat	2019	2020
0	1	0	0	1	1	0
1	0	0	0	1	0	1
1	0	0	0	1	1	0
0	1	0	1	0	1	0
1	0	1	0	0	0	1
0	1	0	0	1	1	0
0	1	0	1	0	1	0
0	1	0	0	1	1	0
1	0	1	0	0	1	0
1	0	1	0	0	1	0
1	0	1	0	0	0	1
0	1	0	0	1	1	0
0	1	0	0	1	0	1
0	1	0	0	1	0	1
1	0	1	0	0	0	1
1	0	0	0	1	0	1
1	0	1	0	0	0	1
1	0	1	0	0	1	0
1	0	0	0	1	0	1

Lampiran 5 Hasil normalisasi data latih

SKS	Kehadiran	TgsK1	TgsK2	TgsK3	TgsK4	TgsK5	TgsK6	TgsIdv	Tgs1	UTS	UAS
1.10469	0.491386723	1.434617	-1.33056	0.518934	0.283364	0.309812	0.313052	1.638617	0.845956	0.113121	0.684709
-0.90523	0.491386723	1.434617	0.890953	1.675064	1.692119	-0.53748	-0.60645	-0.10658	0.845956	1.135113	0.74878
-0.90523	0.491386723	-0.73053	-0.98879	-0.63719	-0.98452	-1.21532	-1.34205	-0.97917	-1.62394	-0.39787	-1.75316
-0.90523	0.491386723	-0.00881	-0.47613	-1.79332	-0.7732	-0.96113	-1.0662	-0.97917	-1.42895	-0.39787	-1.75316
-0.90523	0.491386723	1.434617	0.890953	1.675064	1.692119	-0.53748	-0.60645	0.76602	0.845956	0.794449	0.568468
-0.90523	0.491386723	1.434617	0.890953	1.675064	1.692119	-0.53748	-0.60645	0.76602	0.845956	0.283453	0.540795
-0.90523	0.491386723	0.712901	1.232724	0.518934	-2.53415	0.733458	0.772804	-0.10658	0.845956	1.135113	0.768006
1.10469	0.491386723	-0.00881	-0.13436	-0.17474	-0.42101	-0.02911	-0.05475	-0.97917	-0.84397	-1.5902	0.285911
-0.90523	-2.064084336	-0.73053	-0.98879	-0.17474	-0.10404	-0.02911	0.083176	-0.97917	0.342228	-1.0792	-1.75316
1.10469	0.491386723	-0.00881	-0.13436	-0.17474	-0.42101	-0.53748	-0.05475	-0.10658	0.06599	0.453785	0.504383
1.10469	0.491386723	0.424215	0.378296	0.056483	0.635553	0.733458	0.221102	-0.97917	-0.32399	0.624117	0.431559
-0.90523	0.491386723	-2.17396	0.890953	-0.63719	-0.91408	-1.13059	-1.2501	1.638617	0.845956	0.794449	0.573739
-0.90523	-2.064084336	-0.73053	-0.13436	-0.63719	-0.63233	-0.79167	-0.8823	-0.97917	-1.29895	-0.56821	-1.75316
1.10469	0.491386723	1.434617	1.232724	-1.79332	0.635553	0.733458	0.772804	0.76602	0.845956	-0.39787	0.948262
1.10469	0.491386723	-0.00881	0.378296	0.056483	0.001613	0.733458	0.772804	-0.97917	-0.45399	-1.41987	-0.58798
1.10469	0.491386723	-0.44184	-0.47613	-0.17474	-1.05495	-0.53748	-0.05475	-0.97917	-1.16896	0.453785	-0.3695
-0.90523	0.491386723	-0.73053	-0.13436	-0.63719	-0.63233	-0.79167	-0.8823	-0.97917	-1.29895	-0.56821	-1.75316
-0.90523	0.491386723	-0.73053	-0.98879	-0.63719	-0.98452	-1.21532	-1.34205	-0.97917	-1.62394	-0.90887	-1.75316
-0.90523	0.491386723	1.434617	-1.33056	1.675064	0.635553	0.733458	0.772804	0.76602	0.845956	0.964781	0.573739
-0.90523	0.491386723	-0.73053	-0.13436	-0.63719	-0.63233	-0.79167	-0.8823	-0.10658	0.845956	-0.90887	-1.75316
1.10469	0.491386723	-0.44184	-0.47613	-0.17474	-1.05495	-0.53748	-0.05475	-0.97917	-1.16896	0.624117	0.213087
-0.90523	0.491386723	-2.17396	1.232724	-0.63719	-0.42101	0.733458	-0.60645	-0.10658	0.845956	1.305445	0.830343
1.10469	0.491386723	-0.00881	-0.13436	-0.63719	0.001613	-0.53748	-0.05475	-0.97917	-0.90897	0.964781	-0.00538
-0.90523	0.491386723	-2.17396	1.232724	0.518934	1.692119	0.733458	0.772804	-0.10658	0.52097	1.646109	0.759267

SKS	Kehadiran	TgsK1	TgsK2	TgsK3	TgsK4	TgsK5	TgsK6	TgsIdv	Tgs1	UTS	UAS
1.10469	0.491386723	-0.00881	-0.13436	-0.63719	0.001613	-0.53748	-0.05475	0.76602	0.845956	-1.0792	0.213087
1.10469	0.491386723	-0.44184	-0.13436	-0.17474	-1.05495	-0.53748	-0.05475	-0.97917	-1.10396	0.453785	0.358735
1.10469	0.491386723	-0.44184	-0.13436	-0.63719	-0.42101	-0.53748	-0.05475	-0.97917	-1.07146	0.113121	0.140263
1.10469	-2.234449075	1.434617	1.232724	-1.79332	0.635553	0.733458	0.772804	0.76602	0.845956	-0.22754	0.761001
1.10469	0.491386723	1.145931	1.232724	-0.63719	0.846866	0.987646	1.048654	0.76602	0.845956	-0.39787	0.891044
1.10469	-2.064084336	-0.00881	0.378296	0.056483	0.001613	-0.02911	0.221102	-0.97917	-0.61648	-2.27153	0.285911
1.10469	0.491386723	1.434617	1.232724	0.518934	1.33993	1.580751	1.692306	-0.10658	0.845956	0.964781	0.953464
1.10469	0.491386723	-0.00881	-0.13436	-0.63719	0.001613	-0.53748	-0.05475	-0.97917	-0.90897	-0.73854	0.431559
1.10469	0.491386723	0.712901	-1.33056	0.518934	-0.06882	-0.11383	-0.1467	0.76602	0.845956	0.624117	0.76967
-0.90523	-2.653808426	-0.00881	-0.13436	1.675064	-0.10404	-0.02911	0.083176	-0.10658	0.342228	-0.73854	-0.22732
-0.90523	0.491386723	-0.73053	-0.13436	-0.63719	-0.63233	-0.79167	-0.8823	-0.97917	-1.29895	-0.73854	-1.75316
-0.90523	0.491386723	-0.73053	-1.33056	1.675064	-0.42101	-0.53748	-0.60645	-0.10658	0.845956	-2.10119	-0.63479
1.10469	-0.871531174	1.434617	1.232724	-1.79332	0.635553	0.733458	0.772804	0.76602	0.845956	0.964781	0.773138
1.10469	0.491386723	1.434617	-1.33056	0.518934	0.283364	0.309812	0.313052	0.76602	0.845956	-1.41987	0.122924
1.10469	0.491386723	-0.00881	-0.13436	0.056483	-0.42101	0.225082	0.221102	1.638617	0.845956	-0.73854	0.285911
-0.90523	0.491386723	1.434617	1.232724	1.675064	1.692119	2.004397	2.152058	-0.10658	0.845956	0.624117	0.84076
1.10469	0.491386723	0.424215	0.378296	0.056483	0.635553	0.733458	0.221102	-0.10658	0.650964	0.624117	0.722855
-0.90523	0.491386723	-2.17396	0.890953	0.518934	-0.56189	-0.70694	-0.79035	0.76602	0.715961	1.305445	0.483576
1.10469	0.491386723	-0.00881	0.378296	0.056483	0.001613	-0.02911	0.221102	-0.97917	-0.61648	-1.24953	-0.15103
1.10469	0.491386723	1.434617	-1.33056	1.675064	0.635553	0.733458	0.772804	-0.10658	0.845956	-0.22754	0.854632
1.10469	0.491386723	1.434617	1.232724	0.518934	1.33993	1.580751	1.692306	1.638617	0.845956	-0.05721	0.868503
-0.90523	0.491386723	-2.17396	0.890953	1.675064	-0.42101	0.733458	0.772804	2.511214	0.845956	1.135113	0.920645
-0.90523	0.491386723	-0.73053	-1.33056	-1.33087	-1.3367	-1.63896	-1.8018	-0.97917	-1.94892	-0.22754	-1.75316
1.10469	0.491386723	0.712901	1.232724	0.518934	0.987742	1.157104	1.232555	-0.10658	0.845956	-0.56821	0.807816
-0.90523	0.491386723	-0.73053	-0.47613	-0.17474	-1.3367	-1.63896	-1.8018	-0.97917	-1.94892	-2.10119	-1.75316

SKS	Kehadiran	TgsK1	TgsK2	TgsK3	TgsK4	TgsK5	TgsK6	TgsIdv	Tgs1	UTS	UAS
-0.90523	-2.064084336	0.712901	1.232724	0.518934	1.692119	1.496021	1.048654	-0.97917	0.325978	-0.22754	0.844325
-0.90523	-2.064084336	-0.73053	0.378296	-0.63719	-0.42101	-0.53748	-0.60645	0.76602	0.845956	0.624117	0.703921
1.10469	-0.871531174	0.712901	1.232724	0.518934	0.987742	1.157104	1.232555	0.76602	0.845956	-2.10119	0.875438
1.10469	-0.871531174	0.712901	1.232724	0.518934	0.987742	1.157104	1.232555	-0.10658	0.845956	-0.90887	0.136795
1.10469	0.491386723	-0.00881	-0.13436	0.056483	-0.42101	0.225082	0.221102	0.76602	0.845956	0.453785	0.577207
1.10469	0.491386723	-0.00881	-0.13436	-0.63719	0.001613	-0.02911	0.221102	-0.97917	-0.81147	0.794449	0.577207
1.10469	0.491386723	1.145931	1.232724	-0.63719	0.846866	0.987646	1.048654	1.638617	0.845956	0.283453	0.677773
1.10469	0.491386723	-0.44184	-0.47613	-0.17474	-1.05495	-0.53748	-0.05475	-0.97917	-1.16896	-0.05721	-0.15103
-0.90523	-2.064084336	-0.73053	-1.84322	-0.17474	-1.90021	-2.3168	-2.53741	-0.97917	-2.4689	-0.22754	-1.75316
-0.90523	0.491386723	-0.73053	-1.33056	-1.33087	-1.3367	-1.63896	-1.8018	-0.97917	-1.94892	-1.0792	-1.75316
-0.90523	0.491386723	1.434617	-1.33056	0.518934	0.283364	0.309812	0.313052	-0.97917	-0.45399	0.624117	0.674306
-0.90523	0.491386723	0.712901	1.232724	0.518934	-2.53415	0.733458	0.772804	-0.10658	0.52097	1.135113	0.866755
-0.90523	0.491386723	-0.73053	-2.18499	-1.79332	1.692119	-0.53748	0.772804	-0.10658	-0.2915	1.646109	0.688191
1.10469	0.491386723	-0.00881	-0.13436	-0.17474	-0.42101	-0.53748	-0.05475	0.76602	0.845956	0.964781	0.431559
-0.90523	0.491386723	1.434617	1.232724	0.518934	1.692119	0.733458	-0.05475	-0.10658	0.845956	0.624117	0.719359
1.10469	0.491386723	1.434617	1.232724	-1.79332	0.635553	0.733458	0.772804	0.76602	0.845956	-0.39787	0.821687
1.10469	0.491386723	-0.00881	0.036525	-0.17474	-0.42101	0.733458	0.221102	-0.10658	0.29348	-0.90887	0.067439
1.10469	0.491386723	-0.44184	-0.13436	-0.17474	-1.05495	-0.53748	-0.05475	-0.97917	-1.10396	0.794449	-0.00538
-0.90523	0.491386723	-0.73053	-2.18499	-1.79332	1.692119	-0.53748	0.772804	-0.97917	-1.26645	1.305445	0.597889
1.10469	0.491386723	0.712901	-1.33056	0.518934	-0.06882	-0.11383	-0.1467	-0.10658	0.195984	0.283453	0.572005
-0.90523	0.491386723	-0.73053	-0.13436	-0.63719	-0.63233	-0.79167	-0.8823	-0.10658	0.845956	-1.0792	-1.75316
-0.90523	-2.653808426	-0.00881	-0.13436	1.675064	-0.10404	-0.02911	0.083176	-0.10658	0.342228	-0.05721	0.830357
-0.90523	0.491386723	-0.73053	-0.47613	-1.33087	-0.98452	-1.21532	-1.34205	0.76602	0.325978	-2.44186	-1.75316
-0.90523	0.491386723	-2.17396	0.890953	-0.63719	-0.91408	-1.13059	-1.2501	2.511214	0.845956	0.794449	0.601482
1.10469	0.491386723	-0.00881	-0.13436	-0.17474	-0.42101	-0.53748	-0.05475	-0.97917	-0.90897	-2.44186	0.140263

SKS	Kehadiran	TgsK1	TgsK2	TgsK3	TgsK4	TgsK5	TgsK6	TgsIdv	Tgs1	UTS	UAS
-0.90523	0.491386723	-0.00881	-0.47613	-1.33087	-0.63233	-0.79167	-0.8823	-0.97917	-1.29895	-1.24953	-1.75316
1.10469	-2.064084336	-0.00881	0.378296	0.056483	0.001613	-0.02911	0.221102	-0.10658	0.358477	-0.39787	0.213087
-0.90523	-2.064084336	-0.00881	-0.47613	-1.33087	-0.63233	-0.79167	-0.8823	-0.97917	-1.29895	0.624117	-1.75316
1.10469	0.491386723	-0.44184	-0.13436	-0.63719	-0.42101	-0.53748	-0.05475	-0.97917	-1.07146	0.624117	0.431559
1.10469	0.491386723	1.434617	1.232724	0.518934	1.33993	1.580751	1.692306	0.76602	0.845956	1.305445	0.665636
-0.90523	-2.653808426	1.434617	0.890953	0.518934	1.199055	1.411292	1.508406	-0.10658	0.845956	1.305445	0.677773
-0.90523	0.491386723	-2.17396	1.232724	0.518934	1.692119	0.733458	0.772804	1.638617	0.845956	0.624117	0.575459
1.10469	0.491386723	-0.44184	-0.47613	-0.17474	-1.05495	-0.53748	-0.05475	-0.97917	-1.16896	0.964781	0.285911
-0.90523	0.491386723	-0.00881	-0.13436	-0.17474	-0.13926	-0.19856	-0.23865	-0.97917	-0.84397	-1.41987	-1.75316
1.10469	-0.871531174	0.712901	1.232724	0.518934	0.987742	1.157104	1.232555	-0.10658	0.845956	0.624117	-0.10769
-0.90523	-2.064084336	1.434617	0.890953	1.675064	1.692119	-0.53748	-0.60645	-0.97917	0.130987	0.283453	0.963757
-0.90523	0.491386723	-0.73053	1.232724	1.675064	0.635553	2.004397	1.048654	1.638617	0.845956	0.624117	0.733341
1.10469	-0.871531174	1.434617	-1.33056	1.675064	0.635553	0.733458	0.772804	-0.10658	0.845956	-0.05721	0.903181
-0.90523	0.491386723	-2.17396	1.232724	-0.63719	-0.42101	0.733458	-0.60645	2.511214	0.845956	0.794449	0.617114
-0.90523	0.491386723	-0.00881	-0.13436	1.675064	-0.10404	-0.02911	0.083176	-0.97917	0.342228	0.624117	0.648297
-0.90523	0.491386723	1.434617	1.232724	1.675064	1.692119	2.004397	2.152058	0.76602	0.845956	1.305445	0.870237
1.10469	0.491386723	-0.44184	-0.13436	-0.63719	-0.42101	-0.53748	-0.05475	-0.97917	-1.07146	0.453785	0.577207
-0.90523	0.491386723	-2.17396	1.232724	-0.63719	-0.42101	0.733458	-0.60645	1.638617	0.845956	-0.56821	0.617114
-0.90523	-2.064084336	-0.73053	-0.47613	-0.17474	-1.3367	-1.63896	-1.8018	-0.97917	-1.94892	-0.22754	-1.75316
1.10469	0.491386723	-0.00881	-0.13436	0.056483	-0.42101	0.225082	0.221102	0.76602	0.845956	-0.22754	0.504383
-0.90523	-2.064084336	-0.73053	1.232724	1.675064	0.635553	2.004397	1.048654	-0.97917	0.06599	1.135113	0.890933
-0.90523	0.491386723	-0.73053	-1.33056	-1.33087	-1.3367	-1.63896	-1.8018	-0.10658	-0.97397	-0.56821	-1.75316
-0.90523	0.491386723	-0.00881	-0.13436	-0.17474	-0.13926	-0.19856	-0.23865	-0.97917	-0.84397	-1.0792	-1.75316
1.10469	0.491386723	-0.00881	0.378296	0.056483	0.001613	0.733458	0.772804	-0.10658	0.52097	0.283453	0.577207
-0.90523	0.491386723	-0.73053	1.232724	0.518934	0.283364	0.309812	0.313052	2.511214	0.845956	0.794449	0.781808

SKS	Kehadiran	TgsK1	TgsK2	TgsK3	TgsK4	TgsK5	TgsK6	TgsIdv	Tgs1	UTS	UAS
-0.90523	0.491386723	-0.73053	0.378296	-0.63719	-0.42101	-0.53748	-0.60645	-0.10658	-0.129	1.135113	0.745285
-0.90523	-2.064084336	-0.00881	-0.47613	-1.79332	-0.7732	-0.96113	-1.0662	-0.97917	-1.42895	-1.76053	-1.75316
-0.90523	-2.653808426	1.434617	-1.33056	0.518934	0.283364	0.309812	0.313052	1.638617	0.845956	0.794449	0.634426
-0.90523	0.491386723	0.712901	1.232724	0.518934	1.692119	1.496021	1.048654	-0.10658	0.845956	-0.90887	0.526813
-0.90523	0.491386723	-0.00881	-0.13436	1.675064	-0.10404	-0.02911	0.083176	-0.97917	0.342228	0.624117	0.544263
-0.90523	0.491386723	-0.73053	-2.18499	-1.79332	1.692119	-0.53748	0.772804	-0.10658	-0.2915	1.646109	0.721107
1.10469	0.491386723	1.434617	-1.33056	0.518934	0.283364	0.309812	0.313052	0.76602	0.845956	0.624117	0.409018
-0.90523	0.491386723	1.434617	1.232724	1.675064	1.692119	2.004397	2.152058	2.511214	0.845956	0.624117	1.041893
1.10469	0.491386723	-0.44184	-0.13436	-0.63719	-0.42101	-0.53748	-0.05475	0.76602	0.845956	-0.39787	-0.00538
-0.90523	0.491386723	-0.44184	-0.98879	-1.79332	-1.19583	-1.4695	-1.6179	-0.97917	-1.81893	-2.10119	-1.75316
1.10469	0.491386723	1.434617	-1.33056	1.675064	0.635553	0.733458	0.772804	0.76602	0.845956	0.624117	0.435027
1.10469	0.491386723	0.712901	-1.33056	0.518934	-0.06882	-0.11383	-0.1467	0.76602	0.845956	1.135113	0.506117
1.10469	0.491386723	0.712901	1.232724	0.518934	0.987742	1.157104	1.232555	1.638617	0.845956	1.305445	0.748863
-0.90523	0.491386723	-2.17396	1.232724	0.518934	1.692119	0.733458	0.772804	-0.97917	-0.45399	1.646109	0.64304
-0.90523	0.491386723	-0.44184	-0.98879	-1.79332	-1.19583	-1.4695	-1.6179	-0.97917	-1.81893	-1.24953	-1.75316
1.10469	0.491386723	-0.00881	-0.13436	-0.63719	0.001613	-0.02911	0.221102	-0.10658	0.163485	-0.05721	0.431559
1.10469	0.491386723	-0.00881	0.036525	-0.17474	-0.42101	0.733458	0.221102	-0.97917	-0.68148	-1.5902	-0.00538
-0.90523	-2.064084336	-0.73053	-1.84322	-0.17474	-1.90021	-2.3168	-2.53741	-0.10658	-1.49394	-1.0792	-1.75316
1.10469	0.491386723	-0.00881	0.378296	0.056483	0.001613	0.733458	0.772804	-0.97917	-0.45399	0.113121	0.577207
-0.90523	-2.064084336	-0.73053	-1.84322	-0.17474	-1.90021	-2.3168	-2.53741	-0.97917	-2.4689	-1.41987	-1.75316
-0.90523	0.491386723	-0.73053	-1.33056	1.675064	-0.42101	-0.53748	-0.60645	0.76602	0.845956	-0.39787	0.653499
-0.90523	0.491386723	0.712901	1.232724	0.518934	1.692119	1.496021	1.048654	-0.10658	0.845956	1.135113	0.601384
-0.90523	-2.064084336	-0.73053	-1.33056	-1.79332	-1.47758	-1.80842	-1.9857	-0.97917	-2.07892	-0.73854	-1.75316
1.10469	0.491386723	0.712901	1.232724	0.518934	0.987742	1.157104	1.232555	0.76602	0.845956	-1.41987	0.478374
-0.90523	0.491386723	-0.73053	-0.98879	-0.17474	-0.10404	-0.02911	0.083176	-0.97917	0.342228	-0.39787	-1.75316

SKS	Kehadiran	TgsK1	TgsK2	TgsK3	TgsK4	TgsK5	TgsK6	TgsIdv	Tgs1	UTS	UAS
1.10469	0.491386723	-0.00881	-0.13436	-0.17474	-0.42101	-0.02911	-0.05475	0.76602	0.845956	0.453785	0.577207
1.10469	0.491386723	1.434617	-1.33056	0.518934	0.283364	0.309812	0.313052	0.76602	0.845956	0.964781	0.483576
-0.90523	-2.064084336	-0.73053	-0.98879	-0.63719	-0.98452	-1.21532	-1.34205	-0.10658	0.845956	0.283453	-1.75316
-0.90523	0.491386723	-0.73053	-0.47613	-0.17474	-1.3367	-1.63896	-1.8018	-0.97917	-1.94892	-0.90887	-1.75316
1.10469	-0.871531174	0.712901	1.232724	0.518934	0.987742	1.157104	1.232555	0.76602	0.845956	0.624117	0.424623
-0.90523	0.491386723	-0.44184	-0.98879	-1.79332	-1.19583	-1.4695	-1.6179	-0.10658	-0.84397	-0.39787	-1.75316
-0.90523	0.491386723	-0.73053	1.232724	1.675064	0.635553	2.004397	1.048654	2.511214	0.845956	1.135113	0.807913

Lampiran 6 Hasil normaliasi data tes

SKS	Kehadiran	TgsK1	TgsK2	TgsK3	TgsK4	TgsK5	TgsK6	TgsIdv	Tgs1	UTS	UAS
-0.90523	0.4913867	0.712901	1.232724	0.518934	-2.53415	0.733458	0.772804	-0.10658	0.845956	1.305445	1.000169
1.10469	0.4913867	1.434617	-1.33056	0.518934	0.283364	0.309812	0.313052	0.76602	0.845956	0.624117	0.953464
-0.90523	0.4913867	-0.00881	-0.13436	-0.17474	-0.13926	-0.19856	-0.23865	-0.10658	0.130987	-0.22754	-1.75316
-0.90523	-2.064084	-0.73053	-0.98879	-0.17474	-0.10404	-0.02911	0.083176	-0.97917	0.342228	-0.56821	-1.75316
1.10469	0.4913867	0.712901	1.232724	0.518934	0.987742	1.157104	1.232555	0.76602	0.845956	-0.05721	0.877172
1.10469	0.4913867	-0.00881	-0.13436	-0.63719	0.001613	-0.02911	0.221102	-0.10658	0.163485	-1.0792	0.431559
-0.90523	0.4913867	-0.73053	0.378296	-0.63719	-0.42101	-0.53748	-0.60645	1.638617	0.845956	1.135113	0.740333
-0.90523	0.4913867	-1.45225	-1.33056	-0.17474	-2.18196	-2.65571	-2.90521	-0.97917	-2.72889	-0.90887	-1.75316
1.10469	0.4913867	1.145931	1.232724	-0.63719	0.846866	0.987646	1.048654	0.76602	0.845956	0.283453	0.490512
1.10469	-2.234449	1.434617	1.232724	0.518934	1.33993	1.580751	1.692306	-0.10658	0.845956	-0.22754	0.677773
-0.90523	-2.064084	-0.73053	-0.13436	-0.63719	-0.63233	-0.79167	-0.8823	-0.97917	-1.29895	-0.39787	-1.75316
-0.90523	0.4913867	1.434617	0.890953	0.518934	1.199055	1.411292	1.508406	0.76602	0.845956	1.305445	0.745396
1.10469	-0.871531	1.145931	1.232724	-0.63719	0.846866	0.987646	1.048654	0.76602	0.845956	0.283453	0.466237
1.10469	0.4913867	-0.00881	0.036525	-0.17474	-0.42101	0.733458	0.221102	-0.10658	0.29348	0.964781	0.285911

SKS	Kehadiran	TgsK1	TgsK2	TgsK3	TgsK4	TgsK5	TgsK6	TgsIdv	Tgs1	UTS	UAS
-0.90523	0.4913867	-0.44184	-0.98879	-1.79332	-1.19583	-1.4695	-1.6179	-0.97917	-1.81893	-2.10119	-1.75316
1.10469	-0.871531	1.434617	-1.33056	0.518934	0.283364	0.309812	0.313052	-0.10658	0.52097	-0.73854	-0.15797
-0.90523	0.4913867	-0.73053	-0.47613	-1.33087	-0.98452	-1.21532	-1.34205	-0.97917	-1.62394	-1.5902	-1.75316
-0.90523	0.4913867	0.712901	1.232724	0.518934	1.692119	1.496021	1.048654	-0.10658	0.845956	0.113121	0.637796
-0.90523	0.4913867	-1.45225	1.232724	1.675064	0.283364	0.309812	0.313052	-0.10658	0.52097	0.964781	0.537327
-0.90523	0.4913867	1.434617	1.232724	0.518934	1.692119	0.733458	-0.05475	1.638617	0.845956	0.794449	0.793931
-0.90523	0.4913867	-2.17396	0.890953	1.675064	-0.42101	0.733458	0.772804	-0.10658	0.29348	-0.39787	0.752276
1.10469	0.4913867	1.434617	-1.33056	0.518934	0.283364	0.309812	0.313052	1.638617	0.845956	1.305445	0.833825
1.10469	0.4913867	-0.44184	-0.13436	-0.63719	-0.42101	-0.53748	-0.05475	-0.10658	-0.0965	0.964781	0.140263
1.10469	0.4913867	0.424215	0.378296	0.056483	0.635553	0.733458	0.221102	1.638617	0.845956	0.964781	0.868503
-0.90523	0.4913867	-0.73053	-2.18499	-1.79332	1.692119	-0.53748	0.772804	1.638617	0.845956	1.646109	0.800922
-0.90523	-2.064084	-2.17396	0.890953	1.675064	-0.42101	0.733458	0.772804	-0.10658	0.29348	1.135113	0.480205
-0.90523	0.4913867	-1.45225	-1.33056	0.518934	-1.12539	-1.38477	-1.52595	1.638617	0.845956	1.646109	0.948262
-0.90523	-2.064084	-1.45225	-1.33056	-0.17474	-2.18196	-2.65571	-2.90521	-0.97917	-2.72889	-1.0792	-1.75316
1.10469	-2.064084	-0.00881	0.378296	0.056483	0.001613	-0.02911	0.221102	-0.10658	0.358477	-0.05721	0.140263
-0.90523	0.4913867	-0.73053	1.232724	1.675064	0.635553	2.004397	1.048654	-0.97917	0.06599	1.135113	0.641292
-0.90523	0.4913867	-2.17396	0.890953	-0.63719	-0.91408	-1.13059	-1.2501	-0.10658	-0.58398	-0.05721	0.051834
-0.90523	-2.064084	-0.00881	-0.47613	-1.79332	-0.7732	-0.96113	-1.0662	-0.97917	-1.42895	-0.56821	-1.75316
1.10469	0.4913867	-0.00881	-0.13436	-0.17474	0.001613	-0.53748	0.221102	-0.10658	0.163485	-1.76053	-0.3695
-0.90523	0.4913867	-0.73053	1.232724	0.518934	0.283364	0.309812	0.313052	-0.10658	0.52097	-0.22754	0.435027
1.10469	0.4913867	1.434617	1.232724	0.518934	1.33993	1.580751	1.692306	0.76602	0.845956	-0.39787	0.878906
1.10469	0.4913867	1.434617	1.232724	0.518934	1.33993	1.580751	1.692306	-0.10658	0.845956	-0.90887	0.651765
-0.90523	-2.064084	-0.73053	-0.47613	-1.33087	-0.98452	-1.21532	-1.34205	-0.97917	-1.62394	-0.73854	-1.75316
1.10469	0.4913867	1.434617	1.232724	0.518934	1.33993	1.580751	1.692306	0.76602	0.845956	-0.22754	0.854632
-0.90523	0.4913867	-2.17396	0.890953	1.675064	-0.42101	0.733458	0.772804	2.511214	0.845956	0.113121	0.941327

SKS	Kehadiran	TgsK1	TgsK2	TgsK3	TgsK4	TgsK5	TgsK6	TgsIdv	Tgs1	UTS	UAS
1.10469	0.4913867	-0.44184	-0.13436	-0.63719	-0.42101	-0.53748	-0.05475	-0.10658	-0.0965	-1.24953	0.431559
-0.90523	0.4913867	1.434617	1.232724	0.518934	1.692119	0.733458	-0.05475	-0.97917	0.260981	0.794449	0.757519
1.10469	-0.871531	1.434617	-1.33056	0.518934	0.283364	0.309812	0.313052	-0.10658	0.52097	0.794449	0.787009
-0.90523	0.4913867	-0.00881	-0.47613	-1.33087	-0.63233	-0.79167	-0.8823	-0.97917	-1.29895	-0.56821	-1.75316
-0.90523	0.4913867	-0.73053	0.378296	-0.63719	-0.42101	-0.53748	-0.60645	-0.10658	-0.129	1.135113	0.688191
1.10469	0.4913867	1.434617	-1.33056	1.675064	0.635553	0.733458	0.772804	0.76602	0.845956	0.283453	0.429825
-0.90523	0.4913867	0.712901	1.232724	0.518934	-2.53415	0.733458	0.772804	-0.97917	-0.45399	-0.56821	0.622358
-0.90523	0.4913867	-2.17396	1.232724	1.675064	-0.06882	-0.11383	-0.1467	-0.97917	-0.77897	0.624117	0.471439
-0.90523	0.4913867	1.434617	1.232724	0.518934	1.33993	1.580751	1.692306	1.638617	0.845956	0.624117	0.710717
-0.90523	-2.064084	-0.73053	-1.33056	-1.79332	-1.47758	-1.80842	-1.9857	-0.10658	-1.10396	0.113121	-1.75316
-0.90523	0.4913867	-2.17396	1.232724	0.518934	1.692119	0.733458	0.772804	-0.97917	-0.45399	0.283453	0.440298
1.10469	0.4913867	-0.00881	-0.13436	-0.17474	0.001613	-0.53748	0.221102	-0.10658	0.163485	-1.24953	0.577207
1.10469	0.4913867	-0.44184	-0.13436	-0.17474	-1.05495	-0.53748	-0.05475	-0.97917	-1.10396	-0.56821	-0.58798
-0.90523	-2.064084	-1.45225	-1.33056	-0.17474	-2.18196	-2.65571	-2.90521	-0.97917	-2.72889	-0.05721	-1.75316
1.10469	0.4913867	-0.00881	-0.13436	-0.17474	-0.42101	-0.02911	-0.05475	-0.97917	-0.84397	0.964781	0.285911
-0.90523	0.4913867	-0.73053	-1.33056	-1.79332	-1.47758	-1.80842	-1.9857	-0.97917	-2.07892	-0.05721	-1.75316
-0.90523	0.4913867	1.434617	1.232724	1.675064	1.692119	2.004397	2.152058	-0.97917	0.845956	0.624117	0.499181
1.10469	0.4913867	-0.00881	-0.13436	-0.17474	0.001613	-0.53748	0.221102	2.511214	0.845956	-1.24953	0.577207

Lampiran 7 Tabel hasil uji OOB *Error*

Jumlah Pohon	Jumlah fitur maksimal		
	2	4	8
64	0.41536	0.36878	0.28321
65	0.41479	0.37012	0.28288
66	0.41553	0.37057	0.28538
67	0.41416	0.36909	0.28631
68	0.41564	0.36946	0.28572
69	0.4146	0.36718	0.28536
70	0.41645	0.3686	0.28579
71	0.41774	0.36921	0.28666
72	0.41677	0.36912	0.28621
73	0.41748	0.3698	0.28503
74	0.41814	0.37175	0.28372
75	0.41847	0.37213	0.28156
76	0.42016	0.37087	0.28194
77	0.42139	0.37199	0.28209
78	0.42062	0.37138	0.28163
79	0.41983	0.37049	0.28301
80	0.42034	0.371	0.28197
81	0.41976	0.37183	0.28207
82	0.42074	0.37021	0.28323
83	0.41951	0.36889	0.28394
84	0.41913	0.36985	0.28298
85	0.41874	0.36778	0.28507
86	0.41736	0.36692	0.28451
87	0.4185	0.36625	0.28336
88	0.41847	0.36685	0.28248
89	0.41834	0.36622	0.28329
90	0.41841	0.36747	0.28245
91	0.41689	0.36605	0.28079
92	0.41665	0.36683	0.28042
93	0.41696	0.36607	**0.27909**
94	0.41745	0.36662	0.28073
95	0.4178	0.36681	0.28221
96	0.41866	0.36676	0.28335
97	0.41715	0.36707	0.28377

Jumlah Pohon	Jumlah fitur maksimal		
	2	4	8
98	0.41896	0.36798	0.28306
99	0.41936	0.3685	0.28371
100	0.41937	0.36981	0.28482
101	0.42094	0.37052	0.28579
102	0.42344	0.36817	0.28607
103	0.42271	0.36993	0.28592
104	0.42243	0.36927	0.2851
105	0.42249	0.36883	0.28388
106	0.42253	0.36854	0.28275
107	0.42252	0.36848	0.28278
108	0.42221	0.3687	0.28334
109	0.4213	0.36899	0.28474
110	0.42188	0.36991	0.28316
111	0.42273	0.36919	0.28218
112	0.42314	0.36909	0.28183
113	0.4228	0.36988	0.28293
114	0.42265	0.36852	0.28346
115	0.42277	0.36817	0.28377
116	0.42337	0.36719	0.28282
117	0.42256	0.3693	0.28337
118	0.42186	0.36835	0.28236
119	0.4216	0.36775	0.28214
120	0.42181	0.36703	0.28306
121	0.42178	0.36628	0.28188
122	0.42183	0.36685	0.28154
123	0.42155	0.36739	0.28146
124	0.42075	0.3684	0.28184
125	0.42052	0.36809	0.28046
126	0.41942	0.36827	0.2807
127	0.41884	0.36861	0.28197
128	0.41904	0.36673	0.28233
129	0.41938	0.3671	0.28257
130	0.41722	0.36749	0.28178
131	0.41696	0.3677	0.28063

Lampiran 8 *Source Code*

Seluruh repositori program yang dibuat dapat diakses melalui *link* berikut:

https://github.com/Faridwm/Skripsi

DAFTAR RIWAYAT HIDUP

FARID WUJDI MUBAROK. Penulis lahir di Jakarta, 07 April 1999. Penulis merupakan anak keempat dari empat bersaudara. Saat ini penulis berdomisili di Jakarta Barat. Email: farid.wm@gmail.com

Riwayat Pendidikan: Penulis mengawali pendidikan di TK Sa'adatuddarain pada tahun 2004-2005. Kemudian penulis menajutkan pendidikan ke SD Negeri Petojo Utara 03 Pagi pada tahun 2005-2011. Lalu penulis melanjutkan pendidikan ke SMP Negeri 72 Jakarta pada tahun 2011-2014. Selanjutnya penulis melanjutkan pendidikan ke SMA Negeri 25 Jakarta pada tahun 2014-2017. Lalu pada tahun 2017, penulis melanjutkan pendidikan ke Universitas Negeri Jakarta, Program Studi Ilmu Komputer melalui jalur SNMPTN dengan mendapatkan bantuan dana pendidikan Kartu Jakarta Mahasiswa Unggul (KJMU).

Riwayat Organisasi: Selama di bangku perkuliahan, penulis mengikuti beberapa organisasi diantara adalah sebagai Koordinator Program Studi Ilmu Komputer Forum Kartu Jakarta Mahasiswa Unggul Universitas Negeri Jakarta (FKJMU UNJ) pada tahun 2018-2019. Organisasi lain yang penulis ikuti selama di bangku perkuliahan ialah sebagai staf Departemen Kaderisasi Badan Eksekutif Mahasiswa (BEM) Program Studi Ilmu Komputer pada tahun 2018-2019.