**PERBANDINGAN AKURASI RANDOM FOREST DAN K- NEAREST NEIGHBORS (KNN) PADA STUDI KASUS KELULUSAN MAHASISWA FAKULTAS**

## **RINA PERMATA SARI**

UNIVERSITAS MUHAMMADIYAH MAKASSAR

e-mail: xxxx@xxxx.xxx\*

***Abstract***

*Graduating on time can bring numerous benefits, not only for the students but also for the university itself. Since graduation rates are a factor in the accreditation process for higher education institutions, students graduating on time can positively impact the university's accreditation evaluation. On the other hand, students who graduate on time benefit by avoiding additional semester tuition fees. In this research, data mining techniques are utilized.*

*This study aims to conduct a comparative analysis between two data modeling methods, namely Random Forest and K-Nearest Neighbors (KNN). The Random Forest method is an algorithm used for classification in machine learning and data mining, with the advantage of handling noise and missing values and being able to process large amounts of data. K-Nearest Neighbors (KNN), on the other hand, is also a classification method and a supervised algorithm that can be applied to both large and small training datasets. It is also easier to operate, effective, and easy to understand. The experimental results using the Random Forest algorithm showed an accuracy rate of 100% with data splits of 90:10, 80:20, and 70:30. Meanwhile, the K-Nearest Neighbors algorithm produced an accuracy rate of 96.42% for a 90:10 data split, 96.42% for an 80:20 data split, and 95.91% for a 70:30 data split. Thus, the results indicate that the Random Forest algorithm is more accurate than K-Nearest Neighbors in determining the graduation rates of students in the water resources engineering program.*

*Keywords: Graduation Prediction, Data Mining, Accuracy, Random Forest, K-Nearest Neighbors, Classification.*

**Abstrak**

Kelulusan mahasiswa yang tepat waktu dapat memberikan banyak manfaat, tidak hanya bagi mahasiswa, tetapi juga bagi universitas itu sendiri. Karena kelulusan merupakan penilaian dalam proses akreditasi perguruan tinggi, maka dengan lulusnya mahasiswa tepat waktu tentu akan membantu dalam penilaian akreditasi perguruan tinggi. Disisi lain, mahasiswa yang lulus tepat waktu diuntungkan karena tidak membayar uang kuliah tambahan semester lagi. Pada penelitian ini teknik yang digunakan adalah data mining.

Penelitian ini bertujuan untuk melakukan studi komparatif antara dua metode pemodelan data, yaitu Random Forest dan K-Nearest Neigbors (KNN). Metode Random Forest merupakan algoritma yang digunakan untuk klasifikasi dalam machine learning dan data mining, yang memiliki kelebihan dapat mengatasii noise dan missing value serta dapat mengatasi data dalam jumlah besar. Sedangkan K-Nearest Neighbors (KNN) juga termasuk metode klasifikasi yang merupakan salah satu algoritma supervised, yang mampu diaplikasikan terhadap sejumlah data training yang banyak maupun sedikit, dan juga pengoperasiannya lebih mudah, efektif dan gampang untuk dipahami. Hasil eksperimen menggunakan algoritma Random Forest maka dihasilkan tingkat akurasi 100% dengan pembagian data 90:10, 80:20 dan 70:30. Sedangkan untuk penggunaan algoritma K-Nearest Neighbors dihasilkan 96,42% untuk pembagian data 90:10, untuk pembagian data 80:20 menghasilkan tingkat akurasi sebesar 96,42%, dan pembagian data dengan rasio 70:30 menghasilkan akurasi 95,91%. Dengan demikian hasil menunjukkan bahwa algoritma Random Forest lebih akurat dibanding dengan K-Nearest Neighbors dalam menentukan tingkat kelulusan mahasiswa fakultas teknik program studi pengairan.

Kata Kunci: Prediksi Kelulusan, Data Mining, Akurasi, Random Forest, K-Nearest Neigborst, Klasifikasi.

**1. Pendahuluan**

**2. Metode Penelitian**

Tempat dan waktu penelitian merupakan dua hal yang sangat penting yang harus ditentukan dalam proses penelitian. Kedua faktor ini berperan penting dalam menentukan validitas dan akurasi hasil penelitian yang dilakukan.

Tempat penelitian merupakan suatu lokasi atau objek yang akan dilakukan penelitian. Penentuan lokasi penelitian merupakan hal penting bagi peneliti dalam proses penelitian sebab akan memudahkan peneliti dalam melakukan penelitian. Lokasi pengambilan data penelitian ini dilakukan di Universitas Muhammadiyah Makassar Lantai 3 Simak Fakultas Teknik, tepatnya di Jl.Sultan Alauddin No. 259, Kota Makassar, Sulawesi Selatan.

Adapun Waktu penelitian yang dilaksanakan dimulai pada bulan Februari 2024 sampai proses pengumpulan data selesai.

Adapun alat dan bagan yang akan digunakan dalam penelitian ini yaitu:

1. Kebutuhan Hardware ( Perangkat Keras)

2. Laptop Acer Aspire 314

3. Ram 8,00 GB

4. OS Windows 11

5. Intel(R) Core(TM) i3-N305 1.80 GHz.

6. Kebutuhan Software (Perangkat Lunak)

7. Visual Studio Code

8. Python

9. System Operasi Windows 11

Perancangan sistem merupakan langkah-langkah yang terlibat dalam pemodelan, perancangan, dan konstruksi suatu sistem untuk memenuhi kebutuhan khusus dan memecahkan masalah tertentu pada penelitian ini. Tahapan ini melibatkan pemilihan teknologi yang sesuai, penentuan arsitektur sistem, desain antarmuka pengguna, pemilihan metode dan algoritma pemrograman, serta pengujian menyeluruh untuk memastikan kinerja optimal. Proses ini bertujuan untuk menghasilakan solusi yang efesien dan efektif sesuai dengan tujuan dan persyaratan yang telah ditetapkan.

Penggunaan Flowchart, sangat bermanfaat dalam pembuatan dan pengembangan sistem karena menyediakan ilustrasi visual yang jelas terhadap alur kerja. Dengan menggunakan Flowchrat, peneliti dan pengembang sistem dapat dengan mudah memahami hubungan antara komponen-komponen utama dan langkah-langkah yang terlibat dalam sistem. Flowchart membantu dalam mengidentifikasi potensi masalah, memungkinkan perbaikan sebelum implementasi, dan menjadi alat komunikasi yang efektif antara tim pengembangan. Selain itu, flowchart juga membantu dalam pemilihan algoritma, memandu proses pengujian dan debugging, serta berfungsi sebagai bentuk dokumentasi yang berguna untuk menjelaskan konsep sistem kepada berbagai pemangku kepentingan. Dengan demikian, penggunaan flowchart membantu menyederhanakan kompleksitas dan meningkatkan pemahaman terhadap alur kerja sistem secara keseluruhan.

Pada gambar diatas penulis akan menjelaskan tahapan-tahapan penelitian untuk mengklasifikasikan kelulusan mahasiswa Fakultas Teknik Universitas Muhammadiyah Makassar.

Tahapan awal yang dilakukan yaitu mulai dari Pengumpulan Data, pada proses pengumpulan data, dilakukan dengan mengumpulkan data kelulusan mahasiswa Fakultas Teknik Universitas Muhammadiyah Makassar.

Tahapan kedua Seleksi Data Pada proses seleksi data ini dengan menghilangkan beberapa atribut yang tidak relevan dengan tujuan penelitian.

Tahapan ketiga Pembersihan Data Tahapan ini bertujuan untuk memastikan tidak ada duplikasi data, mengidentifikasi data yang tidak konsisten, dan memperbaiki kesalahan pada data, seperti kesalahan pencetakan, sehingga data dapat diproses. Dan digunakan untuk penambangan data. Dalam proses pembersihan data ini telah ditemukan data yang salah, data ganda dan data yang tidak konsisten sehingga penulis masih melakukan cleaning data agar dihasilkan data yang baik.

Tahapan keempat Transformasi Data Tahapan transformasi data ini dilakukan dengan mengubah data menjadi format yang sesuai untuk pemrosesan data mining.

Tahapan kelima Data Training dan Testing Pada tahap ini dilakukan pembagian data menjadi data training dan testing.

Tahapan keenam Pengolahan Data (Data Processing) Pada tahap ini dilakukan Data processing.

Tahapan ketujuh Analisis Algoritma Random Forest dan K-Nearest Neigbors pada tahapan ini dilakukan analisis prediksi kelulusan mahasiswa dengan algoritma Random Forest dan K-Nearest Neigbors.

Tahapan terakhir Hasil Prediksi Kelulusan Mahasiswa Pada tahapan ini diberikann hasil prediksi kelulusan mahasiswa dengan algoritma Random Forest dan K-Nearest neigbors.

Teknik pengujian sistem yang dilakukan dengan teknik empiris yang merujuk pada pengetahuan atau metode yang diperoleh melalui pengamatan langsung atau pengalaman praktis. Dalam ilmu pengetahuan, metode empiris berfokus pada pengumpulan data melalui observasi atau eksperimen untuk menyusun fakta – fakta yang dapat diukur dan diulangi. Pendekatan empiris mendasarkan kesimpulan pada bukti empiris yang dapat diamati dan diuji.

Contohnya, dalam riset ilmiah, metode melibatkan desain eksperimen, pengumpulan data, dan analistik untuk mendapatkan pemahaman yang lebih baik tentang penomena tertentu. Pendekatan ini menekankan validitas dan produktibilitas hasil untuk membangun pengetahuan yang dapat diandalkan.

Teknik analisis data merupakan proses yang dilakukan dalam pengumpulan, pembersihan, tranformasi, pengolahan data untuk mendapatkan informasi yang berguna dan bermakna . tujuan utama dari teknik analisis data adalah untuk mengidentifikasi pola, tren, dan hubungan dalam data sehingga dapat digunakan untuk pengambilan keputusan dan memberikan wawasan yang lebih baik.

Teknik analisis data memcakup berbagai metode dan alat statistik, matematika, atau komputasi yang digunakan untuk memproses dan memanipulasi data, seperti penambangan data, Pembelajaran mesin, statistic deskriptif, dan visualisasi data.

Proses analisis data pada penelitian ini memilikii 3 langkah yaitu:

1. Reduksi Data

Menurut Bapak Sugishirono, reduksi data berarti jumlah data yang diperoleh dari lapangan cukup besar, sehingga perlu dicatat secara cermat dan detail. Data yang telah direduksi memberikan gambaran   yang lebih jelas dan memudahkan peneliti untuk mengumpulkan dan mengambil data selanjutnya sesuai kebutuhan (Mujadilah, 2023).

Dengan mereduksi data, setiap peneliti diarahkan pada tujuan yang ingin dicapai. Fokus utama penelitian kualitatif adalah hasil. Oleh karena itu, jika peneliti menemukan outlier, unknown’s dan kuantitatif yang tidak berpola maka peneliti harus berhati-hati dalam mengecilkan data.

1. Display Data

Display data, atau biasa juga disebut penyajian data, merupakan komponen kedua dari kegiatan analisis data. Menurut Miles dan Huberman, 1992:17 Penyajian data adalah Kumpulan informasi terstruktur yang dijelaskan dalam bentuk narasi yang memungkinkan peneliti menarik kesimpulan penelitian dan mengambil tidakan. Dan menurut Sutopo, 2002:92 Penyajian data ini merupakan rangkaian yang disusun secara logis dan sistematis untuk membantu peneliti memahami apa yang sedang terjadi dan memungkinkan peneliti melakukan tindakan analitis berdasarkan pemahaman tersebut (Mujadilah, 2023).

Hasil reduksi disajikan dnegan cara tertentu per pola, kategori, fokus, dan topik yang ingin dipahami untuk memahami masalah. Dengan melihat data memungkinkan peneliti untuk melihat gambaran besar atau bagian eksklusif dari temuan penelitian mereka. Dalam penelitian kualitatif, penyajian data berupa deskripsi singkat, diagram, interaksi antara kategori, flowchart, dan sebagainya. Teks deskriptif adalah jenis yang sering digunakan untuk menyajikan data adalam penelitian kualitatif.

1. Penarikan Kesimpulan

Penarikan kesimpulan merupakan tahapan akhir dari teknik analisis data yang dilakukan dengan mempertimbangkan hasil reduksi data yang masih relevan dengan tujuan analisis yang ingin dicapai. Tahap ini bertujuan untuk mencari makna dari data yang dikumpulkan dengan mencari hubungan, persamaan , atau perbedaan guna menarik kesimpulan sebagai jawaban dari permasalahan yang ada.

Kesimpulan awal disajikan masih bersifat sementara dan dapat berubah jika tidak ditemukan bukti pendukung pada tahap pengumpulan data selanjutnya. Namun, jika kesimpulan yang diajukan pada tahap awal didukung oleh bukti-bukti yang valid, maka kesimpulan yang dihasilkan merupakan kesimpulan yang kredibel. Tujuan validasi adalah agar penilaian keabsahan data lebih akurat dan objektif, suatu tujuan termasuk dalam konsep dasar analisis.

Dengan demikian, konklusi pada penelitian mungkin bisa menjawab rumusan masalah perkara yang dirumuskan semenjak awal lantaran sudah ditemukan bahwa perkara dan rumusan pada penelitian kualitatif masih bersifat sementara dan akan berkembang sesudah penelitian berada pada lapangan.

**3. Hasil dan diskusi**

Pada tahap ini akan disajikan hasil dan pembahasan mengenai prediksi kelulusan mahasiswa menggunakan metode Random Forest dan KNN ( K-Nearest Neighbors). Hasil eksperimen meliputi pengelompokan data mahasiswa berdasarkan atribut Nama, masa studi, total SKS, IPK dan Nilai IPS semester 1 hingga Nilai IPS Semester 8 menggunakan metode Random Forest dan KNN ( K-Nearest Neighbors).

Tahapan pengumpulan data dalam penelitian ini dilakukan melalui SIMAK Universitas Muhammadiyah Makassar. SIMAK adalah sumber data resmi yang merekam informasi akademik mahasiswa, termasuk riwayat akademik, nilai mata kuliah, dan status akademik lainnya. Data yang dikumpulkan melibatkan mahasiswa Program Studi Teknik Pengairan Fakultas Teknik yang masuk antara tahun 2018 hingga 2023. Hal ini memberikan cakupan data yang melibatkan beberapa angkatan mahasiswa. Informasi yang dikumpulkan mencakup data lengkap tentang mahasiswa, seperti nama, nomor induk mahasiswa, semester masuk, nilai, dan atribut relevan lainnya. Berikut adalah gambaran data yang diperoleh:

Pada tabel 3 di atas menunjukkan data mahasiswa yang mencakup berbagai aspek seperti nim, nama mahasiswa, tahun, periode KRS, kode mata kuliah, jumlah Satuan Kredit Semester (SKS) jumlah mata kuliah wajib yang mendapat nilai baik, jumlah mata kuliah yang mendapat nilai buruk, dan durasi waktu studi. Data tersebut merupakan data mentah yang didapatkan dari simak fakultas, yang akan di gunakan untuk memprediksi kelulusan mahasiswa.

1. Pada proses transformasi data ini dilakukan guna untuk pemilihan atribut yang akan digunakan sebagai inputan untuk analisis.

2. Kemudian pada proses selanjutnya dilakukan pembersihan data seperti mengisi atau menghapus data yang nilainya hilang untuk memastikan integritas data atau keaslian, lalu menghapus nilai yang terdeteksi duplikat dengan data lain, dan menghapus variabel data yang tidak digunakan seperti variabel tahun, periode KRS, kode mata kuliah, nama mata kuliah dan bobot nilai. Sehingga variabel data yang digunakan yaitu, nim, nama, dan ditambahkan variabel data seperti jumlah semester, total SKS, jumlah IPS dari semester 1 hingga semester 8 dan juga jumlah IPK. Dimana variabel tersebut yang dibutuhkan dalam penelitian ini untuk menentukan tingkat kelulusan mahasiswa. Data yang diperoleh setelah dilakukan transpormasi data yaitu sebagai berikut :

Selanjutnya, dilakukan pemilihan atribut dan pembersihan data yang tidak digunakan agar dapat diolah atau dianalisis lebih efektif. Proses ini mengubah nilai-nilai dari suatu variabel sehingga dapat dibandingkan dalam skala yang sama. Tujuan normalisasi adalah untuk menghindari bobot yang tidak seimbang pada atribut dalam suatu model analisis data.

Proses min-maks dilakukan untuk mencakup normalisasi nilai yang mendukung analisis data. Data historis ini memberikan landasan yang kuat untuk melatih model, karena mencakup informasi tentang mahasiswa dari berbagai angkatan.

1. Proses pelabelan data mahasiswa dilakukan dalam tahap pelabelan kelulusan mahasiswa yang merupakan langkah penting untuk penelitian ini, terutama ditujukan untuk mengindentifikasi dan memahami karakteristik yang membedakan tingkat kelulusan tepat waktu mahasiswa. Dalam pelabelan data mahasiswa yang ditentukan yaitu prediksi tingkat kelulusan dimana pelabelannya itu terdiri dari status kelulusan berupa prediksi lulus tepat waktu dan prediksi tidak lulus tepat waktu dalam semester tertentu.

2. Dalam pelabelan data mahasiswa variabel atau atribut yang akan digunakan sebagai inputan yaitu nim, nama, jumlah semester, total SKS, nilai IPS semester 1 sampai semester 8, dan yang terakhir variabel yang digunakan yaitu nilai IPK.

3. Kriteria yang digunakan dalam pelabelan ini yaitu : prediksi lulus tepat waktu; apabila mahasiswa menyelesaikan masa studinya dalam jangka waktu 8 semester dengan memperoleh total nilai SKS 150, atau sesuai dengan yang telah ditentukan di universitas itu sendiri dan memenuhi semua persyaratan akademik. prediksi tidak lulus tepat waktu ; jika mahasiswa tersebut tidak menyelesaikan masa studinya dalam jangka waktu yang ditentukan dan tidak memenuhi persyaratan akademik.

4. Berikut merupakan gambaran dari data mahasiswa yang telah dilakukan pelabelan, terdapat diujung tabel terdiri tabel label mahasiwa yang prediksinya tidak lulus tepat waktu karena nilai IPK yang diperoleh tidak memenuhi standar dalam persyaratan dikampus itu sendiri.

5. Proses Machine Learning Random Forest

6. import pandas as pd

7. import numpy as np

8. import matplotlib.pyplot as plt

9. import seaborn as sns

10. import warnings

11. import sklearn

12. from sklearn.impute import SimpleImputer

13. warnings.simplefilter("ignore")

Kode tersebut berfungsi untuk mengimpor berbagai pustaka yang akan digunakan dalam analisis data, Visualisasi serta pemodelan machine learning dan untuk mengatur beberapa konfigurasi awal.

Mengimpor Pustaka pandas untuk memanipulasi dan analisis data, terutama dalam bentuk tabel (DataFrame), sedangkan pustaka numpy digunakan untuk operasi numerik efisien, terutama pada array multidimensi. Untuk visualisasi data, Pustaka matplotlib.pyplot dan seaborn digunakan untuk membuat grafik untuk Pustaka matplotlib sedangkan untuk Pustaka seaborn digunakan untuk untuk membuat visualisasi data yang lebih menarik dan informatif, dibangun di atas matplotlib. Pustaka warnings digunakan untuk mengontrol peringatan yang muncul selama eksekusi kode Pyhton, pustaka warnings sering kali gunakan untuk menyembunyikan peringatan yang tidak diinginkan. Pustaka sklearn (scikit-learn) digunakan untuk tugas-tugas machine learning seperti preprocessing data, training model, dan evaluasi, dengan SimpleImputer sebagai kelas dalam scikit-learn yang digunakan untuk menangani datayang hilang dengan cara menggantinya dengan nilai statistik seperti mean, median, atau modus. Menyembunyikan peringatan mengatur agar semua peringatan diabaikan, yang berguna untuk menghindari peringatan yang tidak relevan atau mengganggu saat menjalankan kode.

#Import manajemen dataset

df=pd.read\_excel("Dataset .xlsx", sheet\_name = 'Sheet1')

df.describe()

Kode ini digunakan untuk mengimpor dan menampilkan ringkasan statistik dari dataset yang disimpan dalam file Excel. Berikut adalah penjelasan rinci dari setiap baris kode:

1. Mengimpor dataset dari file Excel:

2. pd.read\_excel adalah fungsi dari pustaka pandas yang digunakan untuk membaca file Excel.

3. “Dataset .xlxs” adalah nama file Excel yang akan dibaca.

4. Sheet\_name=’Sheet1’ menentukan bahwa data yang akan diambil dari sheet pertama excel yang bernama ‘Sheet1’.

5. Data yang diimpor akan disimpan dalam DataFrame df

6. Menampilkan ringkasan statistic:

7. df.describe() adalah metode pandas yang memberikan ringkasan statistic dari DataFrame df, seperti mean, standard deviaton, min, mix, dan quartile values untuk setiap kolom numerik dalam DataFrame.

8. df pada baris terakhir bertujuan untuk menampilkan keseluruhan DataFrame df.

start\_col = 2

end\_col = 20

x= df.iloc[:, start\_col:end\_col+1]

y=df.iloc[:,21]

x

y

Kode ini digunakan untuk memilih subnet dari kolom dalam DataFrame df dan membagi data menjadi fitur (x) dan target (y). berikut penjalasan rinci dari setiap baris kode:

1. Menentukan indeks kolom awal dan akhir:

2. start\_col = 2 menetapkan bahwa kolom awal untuk subnet adalah kolom dengan indeks 2 ( kolom ketiga dalam DataFrame, karena indeks dimulai dari 0).

3. end\_col = 20 menetapkan bahwa kolom akhir untuk subnet adalah kolom dengan indeks 20.

4. Memilih subnet kolom sebagai fitur (x):

5. Df.iloc[:, start\_col:end\_col+1] menggunakan .iloc untuk memilih kolom dari start\_col hingga end\_col, ditambah satu untuk memastikan kolom dengan indeks end\_col disertakan.

6. x adalah DataFrame yang berisi kolom-kolom yang dipilih tersebut.

7. Memilih kolom target (y)

8. df.iloc[:,21] menggunakan .iloc untuk memilih kolom dengan indeks 21 (kolom ke-22 dalam DataFrame) sebagai target (y).

9. Menampilkan target (x):

10. Menampilkan x, yang berisi sebuah subnet kolom yang dipilih sebagai fitur. Jika dijalankan di lingkungan interaktif seperti Visual studio Kode, ini akan menampilkan DataFrame tersebut.

from sklearn.model\_selection import train\_test\_split

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x,y,random\_state=0,test\_size=0.1)

Kode ini digunakan untuk membagi dataset menjadi set pelatihan (training set) dan set pengujian (test set). Berikut penjelasan dari setiap baris kode:

1. Mengimpor train\_test\_split dari scikit-learn:

2. train\_test\_split adalahh fungsi dari Pustaka scikit-learn yang digunakan untuk membagi dataset menjadi subnet pelatihan dan pengujian.

3. Membagi dataset menjadi set pelatihan dan pengujian:

4. train\_test\_split(x, y, random\_state=0, test\_size=0.1) membagi dataset menjadi empat subnet:

5. x\_train: fitur untuk set pelatihan.

6. x\_test: fitur untuk set pengujian.

7. y\_train: target: target untuk set pelatihan

8. y\_test: target untuk set pengujian.

9. Parameter yang digunakan:

10. x dan y: DataFrame x dan series y yang telah dipilih sebelumnya sebagai fitur dan target.

11. random \_state=0: Menetapkan nilai set untuk memastikan pembagian dataset yang konsisten setiap kali kode dijalankan. Ini berguna untuk reproduksibilitas hasil.

12. test\_size=0.1: Menentukan bahwa 10% dari data akan digunakan sebagai set pengujian, sementara 90% sisanya akan digunakan sebagai set pelatihan.

13. from imblearn.over\_sampling import SMOTE

14. imputer = SimpleImputer(strategy='mean') #mengubah NAN menjadi nilai rata2

15. x = imputer.fit\_transform(x)

16. smote = SMOTE(random\_state=42)

17. x\_resampled, y\_resampled = smote.fit\_resample(x, y)

Kode tersebut bertujuan untuk menangani nilai yang hilang dalam dataset dan mengatasi masalah ketidakseimbangan kelas menggunakan teknik oversampling SMOTE (Synthenic Minority over-sampling Technique). Berikut penjelasan dari setiap baris kode diatas:

1. Mengimpor SMOTE dari imbalanced-learn:

2. SMOTE adalah teknik oversampling yang digunakan untuk menangani ketidakseimbangan kelas dalam dataset. Ia bekerja dengan membuat sampel sintetis dari kelas minoritas untuk menyeimbangkan jumlah sampel dalam setiap kelas.

3. Menginisialisasi SimpleImputer untuk mengisi nilai yang hilang:

4. SimpleImputer adalah kelas dari scikit-learn yang digunakan untuk menangani nilai yang hilang dalam dataset.

5. Strategy=’mean’ menetapkan strategi untuk mengganti nilai yang hilang dengan rata-rata nilai dari kolom tersebut.

6. Mengganti nilai yang hilang dalam fitur (x) dengan rata-rata:

7. imputer.fit\_transform(x):

8. fit menghitung rata-rata dari setiap kolom fitur dalam x.

9. transform menggantikan nilai yang hilang dengan rata-rata yang telah dihitung.

10. Hasilnya adalah x yang telah diisi dengan nilai rata-rata untuk setiap nilai yang hilang.

11. Menginisialisasi SMOTE:

12. SMOTE(random\_state=42) menginisialisasi objek SMOTE dengan random\_state=42 untuk memastikan bahwa hasil oversampling dapat direproduksi (konsisten setiap kali kode dijalankan).

13. Menggunakan SMOTE untuk oversampling fitur (x) dan target (y):

14. smote.fit\_resample(x, y):

15. fit\_resample menerapkan teknik SMOTE pada fitur x dan target y untuk membuat sampel sintetis dari kelas minoritas.

16. Hasilnya adalah dua variable baru:

17. x\_resampled: Fitur yang telah diresampling.

18. y\_resampled: Target yang telah diresampling.

19. from sklearn.ensemble import RandomForestClassifier

20. from sklearn.impute import SimpleImputer

21. # Membuat imputer untuk menangani nilai yang hilang

22. imputer = SimpleImputer(strategy='mean')

23. # Membuat model Random Forest

24. model = RandomForestClassifier(n\_estimators=100, random\_state=42)

25. # Mengimputasi nilai yang hilang dan melatih model dengan data

26. x\_train = imputer.fit\_transform(x\_resampled)

27. model.fit(x\_train, y\_resampled)

28. Kode ini bertujuan untuk menangani nilai yang hilang dalam dataset, kemudian melatih model Random Forest menggunakan data yang telah diimputasi dan diresampling. Berikut adalah penjelasan dari setiap baris kode diatas:

29. Mengimpor Random Forest Classifier dari sklearn.ensemble:

30. RandomForestClassifier adalah algoritma pembelajaran mesin yang menggunakan banyak pohon Keputusan untuk melakukan klasifikasi.

31. Mengimpor SimpleImputer dari scikit-learn:

32. SimpleImputer adalah kelas yang digunakan untuk menangani nilai yang hilang dalam dataset dengan menggunakan strategi tertentu seperti mean, median, atau modus.

33. Membuat imputer untuk menangani nilai yang hilang :

34. imputer adalah objek SimpleImputer yang diinisialisasi dengan strategi ‘mean’. Yang artinya nilai yang hilang akan digantikan dengan nilai rata-rata dari kolom yang sesuai.

35. Membuat model Random Forest :

36. Membuat objek ‘RandomForestClassifier’ dengan parameter ‘n\_estimators=100’ dengan antrian model akan menggunakan 100 pohon keputusan. Parameter ‘random \_state=42’ yang digunakan untuk memastikan hasil yang konsisten dan dapat direproduksi.

37. Mengimputasi nilai yang hilang dan melatih model dengan data:

38. x\_train\_imputed=imputer.fit\_transform(x\_resampled)

39. fit menghitung rata-rata dari setiap kolom (dari data pelatihan) fitur dalam x\_resampled.

40. Transform menggantikan nilai yang hilang dengan rata-rata yang telah dihitung.

41. Hasilnya adalah x\_train\_imputed, yaitu x\_resampled yaitu data latih yang sudah tidak memiliki nilai yang hilang.

42. Melatih model Random Forest dengan data latih yang telah diimputasi.

43. model.fit(x\_train\_imputed, y\_resampled):

44. fit melatih model Random Forest Classifier menggunakan fitur x\_train\_imputed dan target y\_resampled.

45. Model akan mempelajari hubungan antara fitur dan target berdasarkan data yang telah disediakan.

46. y\_predict = model.predict(x\_test)

47. akurasi = sklearn.metrics.accuracy\_score(y\_test, y\_predict)

48. Kode ini bertujuan untuk menggunakan model yang telah dilatih untuk membuat prediksi pada data uji dan kemudian menghitung akurasi model tersebut. Berikut penjelasan dari setiap baris kode diatas;

49. Membuat prediksi pada data uji:

50. model.predict(x\_test) menggunakan model Random Forest Classifier yang telah dilatih untuk membuat prediksi berdasarkan fitur-fitur dalam x\_test.

51. Hasilnya adalah y\_predict, yang berisi prediksi kelas untuk setiap sampel dalam x\_test.

52. Menghitung akurasi model:

53. Sklearn.metrics.accuracy\_score(y\_test, y\_predict) menghitung akurasi model dengan membandingkan prediksi (y\_predict) dengan nilai actual (y\_test).

54. Accuracy\_score adalah fungsi dari sklearn.metrics yang mengembalikan proporsi prediksi yang benar (benar dibagi dengan total jumlah prediksi).

Berikut langkah-langkah dari fungsi ini:

1. y\_test: Nilai-nilai aktual dari target untuk data uji.

2. x\_predict: Nilai-nilai yang diprediksi oleh model untuk data uji.

3. Akurasi dihitung sebagai jumlah prediksi yang benar dibagi dengan total jumlah sampel.

4. print (x\_test)

5. print()

6. print (akurasi)

Kode ini digunakan untuk menampilkan data uji (x\_test) dan nilai akurasi dari model yang telah dilatih dan diuji. Berikut penjelasan setiap baris kode diatas:

1. Menampilkan data uji (x\_test):

2. print(x\_test) akan mencetak isi dari x\_test ke layar.

3. x\_tets berisi fitur-fitur dari data uji yang digunakan untuk membuat prediksi menggunkan model yang telah dilatih.

4. Dengan menampilkan x\_test, kita dapat melihat data yang digunakan untuk menguji model.

5. Menampilkan akurasi model (akurasi):

6. print(akurasi) akan mencetak nilai akurasi ke layar.

7. akurasi adalah nilai yang dihitung sebelumnya menggunakan sklearn.metrids.accuracy\_score(y\_tets, y\_predit).

8. Dengan menampilkan akurasi, kita dapat melihat seberapa baik model melakukan klasifikasi pada data uji.

9. # Membuat DataFrame results dengan menggunakan .iloc untuk mengakses kolom

10. results = pd.DataFrame({'Feature 1': x\_test.iloc[:, 0], 'Feature 2': x\_test.iloc[:, 1], 'Predicted': y\_predict, 'Actual': y\_test})

11. # Menampilkan DataFrame results

12. print(results)

13. # Simpan DataFrame ke dalam file Excel

14. results.to\_excel('hasil\_prediksi.xlsx', index=False)

15. Kode ini menggabungkan data fitur uji, hasil prediksi, dan nilai actual kedalam satu DataFrame. Setelah itu, DataFrame ditaampilkan konsol untuk verifikasi dan kemudian disimpan dalam file Excel. Berikut penjelasan singkat setiap baris kode diatas:

16. Membuat DataFrame results:

17. Kode ini menggunakan pd.DataFrame() dari pandas untuk membuat DataFrame results.

18. DataFrame ini memiliki empat kolom:

19. ‘Feature 1’ dan ‘Feature 2’: Fitur-fitur ini dari x\_test yang diasumsikan bahwa x\_test memiliki minimal dua fitur, sehingga kita bisa mengakses kolom pertama dan kedua menggunakan x\_tets[:,0] dan x\_test[:,1].

20. ‘Predicted’ : Prediksi yang dihasilkan oleh model (y\_predict).

21. ‘Actual’: Nilai aktual dari y\_tets.

22. Menampilkan DataFrame results:

23. Kode ini mencetak DataFrame results ke layer. Jika dijalankan dalam lingkungan seperti Visual Studio Code maka hasilnya akan ditampilkan dalam bentuk tabel.

24. Menyimpan DataFrame ke dalam file Excel:

25. Kode ini menyimpan DataFrame results ke dalam file Excel dengan nama ‘hasil\_prediksi.xlsx’.

26. Index=False mengatur agar indeks baris tidak disertakan dalam file Excel yang disimpan.

Pada tabel 9 diatas dapat dilihat bahwa hasil prediksi tingkat akurasi dari Algoritma Random Forest. Dimana mahasiswa yang yang diprediksi lulus tepat waktu merupakan mahasiswa yang memcapai total nilai SKS yang telah ditentukan di universitas, dan apabila mahasiswa tersebut lulus dengan 8 semester dengan jumlah SKS yang di peroleh 150 maka mahasiswa tersebut dinyatakan lulus tepat waktu.

1. Proses Machine Learning K-Nearest Neighbors

2. import pandas as pd

3. import numpy as np

4. import matplotlib.pyplot as plt

5. import seaborn as sns

6. import warnings

7. import sklearn

8. from sklearn.impute import SimpleImputer

9. warnings.simplefilter("ignore")

10. from sklearn.metrics import accuracy\_score  # Menambahkan impor accuracy\_score

Kode tersebut berfungsi untuk mengimpor berbagai pustaka yang akan digunakan dalam analisis data, Visualisasi serta pemodelan machine learning dan untuk mengatur beberapa konfigurasi awal.

Mengimpor Pustaka pandas untuk memanipulasi dan analisis data, terutama dalam bentuk tabel (DataFrame), sedangkan pustaka numpy digunakan untuk operasi numerik efisien, terutama pada array multidimensi. Untuk visualisasi data, Pustaka matplotlib.pyplot dan seaborn digunakan untuk membuat grafik untuk Pustaka matplotlib sedangkan untuk Pustaka seaborn digunakan untuk untuk membuat visualisasi data yang lebih menarik dan informatif, dibangun di atas matplotlib. Pustaka warnings digunakan untuk mengontrol peringatan yang muncul selama eksekusi kode Pyhton, pustaka warnings sering kali gunakan untuk menyembunyikan peringatan yang tidak diinginkan. Pustaka sklearn (scikit-learn) digunakan untuk tugas-tugas machine learning seperti preprocessing data, training model, dan evaluasi, dengan SimpleImputer sebagai kelas dalam scikit-learn yang digunakan untuk menangani data yang hilang dengan cara menggantinya dengan nilai statistik seperti mean, median, atau modus. Menyembunyikan peringatan mengatur agar semua peringatan diabaikan, yang berguna untuk menghindari peringatan yang tidak relevan atau mengganggu saat menjalankan kode.

#Import manajemen dataset

df=pd.read\_excel("Dataset .xlsx", sheet\_name = 'Sheet1')

df.describe()

Kode ini digunakan untuk mengimpor dan menampilkan ringkasan statistik dari dataset yang disimpan dalam file Excel. Berikut adalah penjelasan rinci dari setiap baris kode:

1. Mengimpor dataset dari file Excel:

2. pd.read\_excel adalah fungsi dari pustaka pandas yang digunakan untuk membaca file Excel.

3. “Dataset .xlxs” adalah nama file Excel yang akan dibaca.

4. Sheet\_name=’Sheet1’ menentukan bahwa data yang akan diambil dari sheet pertama excel yang bernama ‘Sheet1’.

5. Data yang diimpor akan disimpan dalam DataFrame df

6. Menampilkan ringkasan statistic:

7. df.describe() adalah metode pandas yang memberikan ringkasan statistic dari DataFrame df, seperti mean, standard deviaton, min, mix, dan quartile values untuk setiap kolom numerik dalam DataFrame.

8. df pada baris terakhir bertujuan untuk menampilkan keseluruhan DataFrame df.

start\_col = 2

end\_col = 20

x= df.iloc[:, start\_col:end\_col+1]

y=df.iloc[:,21]

x

y

Kode ini digunakan untuk memilih subnet dari kolom dalam DataFrame df dan membagi data menjadi fitur (x) dan target (y). berikut penjalasan rinci dari setiap baris kode:

1. Menentukan indeks kolom awal dan akhir:

2. start\_col = 2 menetapkan bahwa kolom awal untuk subnet adalah kolom dengan indeks 2 ( kolom ketiga dalam DataFrame, karena indeks dimulai dari 0).

3. end\_col = 20 menetapkan bahwa kolom akhir untuk subnet adalah kolom dengan indeks 20.

4. Memilih subnet kolom sebagai fitur (x):

5. Df.iloc[:, start\_col:end\_col+1] menggunakan .iloc untuk memilih kolom dari start\_col hingga end\_col, ditambah satu untuk memastikan kolom dengan indeks end\_col disertakan.

6. x adalah DataFrame yang berisi kolom-kolom yang dipilih tersebut.

7. Memilih kolom target (y)

8. df.iloc[:,21] menggunakan .iloc untuk memilih kolom dengan indeks 21 (kolom ke-22 dalam DataFrame) sebagai target (y).

9. Menampilkan target (x):

10. Menampilkan x, yang berisi sebuah subnet kolom yang dipilih sebagai fitur. Jika dijalankan di lingkungan interaktif seperti Visual studio Kode, ini akan menampilkan DataFrame tersebut.

from sklearn.model\_selection import train\_test\_split

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x,y,random\_state=0,test\_size=0.1)

Kode ini digunakan untuk membagi dataset menjadi set pelatihan (training set) dan set pengujian (test set). Berikut penjelasan dari setiap baris kode:

1. Mengimpor train\_test\_split dari sscikit-learn:

2. train\_test\_split adalahh fungsi dari Pustaka scikit-learn yang digunakan untuk membagi dataset menjadi subnet pelatihan dan pengujian.

3. Membagi dataset menjadi set pelatihan dan pengujian:

4. train\_test\_split(x, y, random\_state=0, test\_size=0.1) membagi dataset menjadi empat subnet:

5. x\_train: fitur untuk set pelatihan.

6. x\_test: fitur untuk set pengujian.

7. y\_train: target: target untuk set pelatihan

8. y\_test: target untuk set pengujian.

9. Parameter yang digunakan:

10. x dan y: DataFrame x dan series y yang telah dipilih sebelumnya sebagai fitur dan target.

11. random \_state=0: Menetapkan nilai set untuk memastikan pembagian dataset yang konsisten setiap kali kode dijalankan. Ini berguna untuk reproduksibilitas hasil.

12. test\_size=0.1: Menentukan bahwa 10% dari data akan digunakan sebagai set pengujian, sementara 90% sisanya akan digunakan sebagai set pelatihan.

13. from imblearn.over\_sampling import SMOTE

14. imputer = SimpleImputer(strategy='mean') #mengubah NAN menjadi nilai rata2

15. x = imputer.fit\_transform(x)

16. smote = SMOTE(random\_state=42)

17. x\_resampled, y\_resampled = smote.fit\_resample(x, y)

Kode tersebut bertujuan untuk menangani nilai yang hilang dalam dataset dan mengatasi masalah ketidakseimbangan kelas menggunakan teknik oversampling SMOTE (Synthenic Minority over-sampling Technique). Berikut penjelasan dari setiap baris kode diatas:

1. Mengimpor SMOTE dari imbalanced-learn:

2. SMOTE adalah teknik oversampling yang digunakan untuk menangani ketidakseimbangan kelas dalam dataset. Ia bekerja dengan membuat sampel sintetis dari kelas minoritas untuk menyeimbangkan jumlah sampel dalam setiap kelas.

3. Menginisialisasi SimpleImputer untuk mengisi nilai yang hilang:

4. SimpleImputer adalah kelas dari scikit-learn yang digunakan untuk menangani nilai yang hilang dalam dataset.

5. Strategy=’mean’ menetapkan strategi untuk mengganti nilai yang hilang dengan rata-rata nilai dari kolom tersebut.

6. Mengganti nilai yang hilang dalam fitur (x) dengan rata-rata:

7. imputer.fit\_transform(x):

8. fit menghitung rata-rata dari setiap kolom fitur dalam x.

9. transform menggantikan nilai yang hilang dengan rata-rata yang telah dihitung.

10. Hasilnya adalah x yang telah diisi dengan nilai rata-rata untuk setiap nilai yang hilang.

11. Menginisialisasi SMOTE:

12. SMOTE(random\_state=42) menginisialisasi objek SMOTE dengan random\_state=42 untuk memastikan bahwa hasil oversampling dapat direproduksi (konsisten setiap kali kode dijalankan).

13. Menggunakan SMOTE untuk oversampling fitur (x) dan target (y):

14. smote.fit\_resample(x, y):

15. fit\_resample menerapkan teknik SMOTE pada fitur x dan target y untuk membuat sampel sintetis dari kelas minoritas.

16. Hasilnya adalah dua variable baru:

17. x\_resampled: Fitur yang telah diresampling.

18. y\_resampled: Target yang telah diresampling.

from sklearn.neighbors import KNeighborsClassifier

from sklearn.impute import SimpleImputer

# membuat imputer untuk menangani nilai yang hilang

import numpy as np

x[np.random.randint(150, size=20), np.random.randint(4, size=20)] = np.nan

# Mengisi nilai yang hilang dan melatih KNeighborsClassifier

x\_train\_imputed = imputer.fit\_transform(x\_resampled)

model = KNeighborsClassifier(n\_neighbors=3)

model.fit(x\_train\_imputed, y\_resampled)

# Mengisi nilai yang hilang dalam dataset dan membuat prediksi

x\_test\_imputed = imputer.transform(x\_test)

y\_pred = model.predict(x\_test\_imputed)

Kode ini bertujuan untuk menangani nilai yang hilang dalam dataset, kemudian melatih model algoritma K-Nearest Neighbors menggunakan data yang telah diimputasi dan diresampling. Berikut adalah penjelasan dari setiap baris kode diatas:

1. Mengimpor K-Nearest Neighbors Classifier dari sklearn.ensemble:

2. KNeighborsClassifier adalah algoritma pembelajaran mesin yang menggunakan nilai k atau jumlah tetangga terdekat dalam keputusan untuk melakukan klasifikasi.

3. Mengimpor SimpleImputer dari scikit-learn:

4. SimpleImputer adalah kelas yang digunakan untuk menangani nilai yang hilang dalam dataset dengan menggunakan strategi tertentu seperti mean, median, atau modus.

5. ‘import numpy as np’ merupakan perintah untuk mengimpor library numpy dengan alias ‘np’ yang digunakan untuk memanipulasi array dan operasi numerik. Setalah menuliskan perintah untuk memanipulasi array dan operasi numerik maka ditambahkan perintah untuk nilai yang hilang dalam data.

6. ‘np.random.randint (150, size=20)’ Dimana perintah tersebut menghasilkan nilai array yang berisi 20 bilangan bulat acak antara 0 dan 149.

7. ‘np.random.randint(4, size=20)’ yang menghasilkan array berisi 20 bilangan bulat acak antara 0 dan 3.

8. ‘x[np.random.randint(150, size=20), np.random.randint(4,size=20)] = np.nan’ merupakan perintah untuk menggunakan indeks acak untuk menetapkan beberapa nilai dalam array ‘x’ menjadi ‘np.nan’ untuk mengsimulasikan data yang hilang.

9. Membuat imputer untuk menangani nilai yang hilang :

10. imputer adalah objek SimpleImputer yang diinisialisasi dengan strategi ‘mean’. Yang artinya nilai yang hilang akan digantikan dengan nilai rata-rata dari kolom yang sesuai.

11. Membuat model K-Nearest Neighbors :

12. Membuat objek ‘KNeighborsClassifier’ dengan parameter nilai ‘k’ (jumlah tetangga terdekat ) = 3yang digunakan untuk memastikan hasil yang konsisten dan dapat direproduksi.

13. Mengimputasi nilai yang hilang dan melatih model dengan data:

14. x\_train\_imputed=imputer.fit\_transform(x\_resampled)

15. fit menghitung rata-rata dari setiap kolom (dari data pelatihan) fitur dalam x\_resampled.

16. Transform menggantikan nilai yang hilang dengan rata-rata yang telah dihitung.

17. Hasilnya adalah x\_train\_imputed, yaitu x\_resampled yaitu data latih yang sudah tidak memiliki nilai yang hilang.

18. Melatih model K-Nearest Neighbors dengan data latih yang telah diimputasi.

19. model.fit(x\_train\_imputed, y\_resampled):

20. fit melatih model K-Nearest Neighbors Classifier menggunakan fitur x\_train\_imputed dan target y\_resampled.

21. Model akan mempelajari hubungan antara fitur dan target berdasarkan data yang telah disediakan.

22. # Menghitung nilai akurasi

23. from sklearn.metrics import accuracy\_score

24. accuracy = accuracy\_score(y\_test, y\_pred)

25. print(f'Accuracy: {accuracy}')

26. Kode ini bertujuan untuk menggunakan model yang telah dilatih untuk membuat prediksi pada data uji dan kemudian menghitung akurasi model tersebut. Berikut penjelasan dari setiap baris kode diatas;

27. Membuat prediksi pada data uji:

28. model.predict(x\_test) menggunakan model Algoritma K-Nearest Neighbors Classifier yang telah dilatih untuk membuat prediksi berdasarkan fitur-fitur dalam x\_test.

29. Hasilnya adalah y\_predict, yan berisi prediksi kelas untuk setiap sampel dalam x\_test.

30. Menghitung akurasi model:

31. Sklearn.metrics.accuracy\_score(y\_test, y\_predict) menghitung akurasi model dengan membandingkan prediksi (y\_predict) dengan nilai actual (y\_test).

32. Accuracy\_score adalah fungsi dari sklearn.metrics yang mengembalikan proporsi prediksi yang benar (benar dibagi dengan total jumlah prediksi).

Berikut langkah-langkah dari fungsi ini:

1. y\_test: Nilai-nilai aktual dari target untuk data uji.

2. x\_predict: Nilai-nilai yang diprediksi oleh model untuk data uji.

3. Akurasi dihitung sebagai jumlah prediksi yang benar dibagi dengan total jumlah sampel.

4. print (x\_test)

5. print()

6. print (akurasi)

Kode ini digunakan untuk menampilkan data uji (x\_test) dan nilai akurasi dari model yang telah dilatih dan diuji. Berikut penjelasan setiap baris kode diatas:

1. Menampilkan data uji (x\_test):

2. print(x\_test) akan mencetak isi dari x\_test kelayar.

3. x\_tets berisi fitur-fitur dari data uji yang digunakan untuk membuat prediksi menggunkan model yang telah dilatih.

4. Dengan menampilkan x\_test, kita dapat melihat data yang digunakan untuk menguji model.

5. Menampilkan akurasi model (akurasi):

6. print(akurasi) akan mencetak nilai akurasi ke layar.

7. akurasi adalah nilai yang dihitung sebelumnya menggunakan sklearn.metrids.accuracy\_score(y\_tets, y\_predit).

8. Dengan menampilkan akurasi, kita dapat melihat seberapa baik model melakukan klasifikasi pada data uji.

9. # Membuat DataFrame results dengan menggunakan .iloc untuk mengakses kolom

10. results = pd.DataFrame({'Feature 1': x\_test.iloc[:, 0], 'Feature 2': x\_test.iloc[:, 1], 'Predicted': y\_predict, 'Actual': y\_test})

11. # Menampilkan DataFrame results

12. print(results)

13. # Simpan DataFrame ke dalam file Excel

14. results.to\_excel('hasil\_prediksi.xlsx', index=False)

15. Kode ini menggabungkan data fitur uji, hasil prediksi, dan nilai actual kedalam satu DataFrame. Setelah itu, DataFrame ditaampilkan konsol untuk verifikasi dan kemudian disimpan dalam file Excel. Berikut penjelasan singkat setiap baris kode diatas:

16. Membuat DataFrame results:

17. Kode ini menggunakan pd.DataFrame() dari pandas untuk membuat DataFrame results.

18. DataFrame ini memiliki empat kolom:

19. ‘Feature 1’ dan ‘Feature 2’: Fitur-fitur ini dari x\_test yang diasumsikan bahwa x\_test memiliki minimal dua fitur, sehingga kita bisa mengakses kolom pertama dan kedua menggunakan x\_tets[:,0] dan x\_test[:,1].

20. ‘Predicted’ Prediksi yang dihasilkan oleh model (y\_predict).

21. ‘Actual’: Nilai aktual dari y\_tets.

22. Menampilkan DataFrame results:

23. Kode ini mencetak DataFrame results ke layer. Jika dijalankan dalam lingkungan seperti Visual Studio Code maka hasilnya akan ditampilkan dalam bentuk tabel.

24. Menyimpan DataFrame ke dalam file Excel:

25. Kode ini menyimpan DataFrame results ke dalam file Excel dengan nama ‘Hasil\_prediksi.xlsx’.

26. Index=False mengatur agar indeks baris tidak disertakan dalam file Excel yang disimpan.

Pada tabel 10 diatas merupakan tabel hasil prediksi kelulusan mahasiswa menggunakan Algoritma K-Nearest Neighbors . Dimana mahasiswa yang dikatakan lulus tepat waktu apabila mahasiswa tersebut lulus dengan total nilai SKS yang ditempuh mencapai standar yang telah ditentukan dari pihal universitas atau dari pihak fakultas. Mahasiswa yang memeperoleh jumlah semester sebanyak 8 semester dan memperoleh total nilai SKS 150 maka mahasiswa tersebut dinyatakan lulus tepat waktu.

1. Pengujian data dilakukan dengan 1081 data yang dibagi menjadi data latih dan data uji dengan perbandingan 90:10, 80:20 dan 70:30, dimana data latih digunakan untuk melatih model sedangkan data uji digunakan untuk menguji kinerja model yang dilatih.

2. Pengujian Data Random Forest

Pada tabel diatas dapat dilihat bahwa pengujian Random Forest yang dilakukan dengan pembagian data 90:10, 80:20 dan 70: 30 menghasilkan tingkat akurasinya yang sangat baik yaitu sebesar 100%. Hal ini menunjukkan bahwa performa model Random Forest sangat baik digunakan untuk menentukan tingkat kelulusan mahasiswa fakultas teknik.

1. Pengujian Data K-Nearest Neigbors

Pada tabel diatas menunjukkan hasil dalm pembagian data 90:10 menghasilkan tingkat akurasi 96,42% , pembagian data 80:20 hasil akurasinya 96,42% sedangkan untuk pembagian data 70:30 menghasilkan tingkat akurasi sebesar 95.91%. hasil ini menunjukkan bahwa menurunya tingkat akurasi seiring dengan peningkatan dataset yang digunakan untuk training. Meskipun rasio 90:10 memberikan akurasi tertinggi, akan tetapi perbedaannya tidak terlalu jauh dibandingkan dengan rasio 80:20 yang memiliki hubungan yang baik antara ukuran training set dan evaluasi yang stabil. Rasio 70:30 menunjukkan sedikit penurunan akurasi yang mungkin disebabkan oleh penurunan jumlah data yang digunakan untuk training model. Sehingga dapat disimpulkan bahwa data dengan akurasi paling tinggi adalah pembagian data dengan rasio 90:10 dengan hasil 96,42%.

Berdasakan tabel diatas dapat dilihat bahwa perbandingan hasil antara Random Forest dan K-Nearest Neighbors menunjukkan bahwa Random Forest menghasilkan akurasi sebesar 100% untuk semua rasio pembagian data 90:10, 80:20, dan 70:30. Ini menunjukkan bahwa Random Forest sangan baik dan konsisten dalam menentukan tingkat akurasi. Hal ini karenakan algoritma Random Forest mempunyai kemampuan dalam menangani kompleksitas data, dan mengelola dataset yang tidak seimbang dengan lebih baik. Sedangkan pada Algoritma K-Nearest Neighbors menunjukkan penurunan akurasi dari 96,42% (90:10) menjadi 95,91 (70:30). Hal ini dikarenakan dengan pengurangan proporsi data latih.

**4. Kesimpulan**

Bab ini merupakan bab terakhir yang berisi kesimpulan dan saran dari hasil penelitian, serta merupakan garis besar dari metode penelitian yang telah dilakukan. Kesimpulan adalah hasil akhir dari penelitian yang dilakukan, sedangkan Saran berisi tentang rekomendasi sesuai dengan keterbatasan yang ada pada sistem.

Dari hasil penelitian yang dilakukan maka dapat disimpulkan sebagai berikut:

1. Setelah diterapkan kedalam data mining dengan menggunakan metode Random forest dan K-Nearest Neighbors maka dari ini menghasilkan kelompok mahasiswa yang tidak lulus terdari dari mahasiswa – mahasiswa yang cenderung tidak menyelesaikan studi tepat waktu dan memiliki performa akademik yang kurang baik. Mahasiswa dalam kelompok ini memiliki rata-rata masa studi yang lebih lama dan nilai IPS yang lebih rendah dengan kemungkinan untuk menambah semester tambahan.

2. Dari penelitian yang dilakukan menggunakan data mahasiswa fakultas teknik mulai angkatan 2017 sampai 2023 ada sebanyak 1081 data yang dibagi menjadi data training sebanyak 973 mahasiswa dan data testing sebanyak 108 mahasiswa dengan menggunakan algoritma Random Forest maka dihasilkan tingkat akurasi 100% dengan pembagian data 90:10, 80:20 dan 70:30. Sedangkan untuk penggunaan algoritma K-Nearest Neighbors dihasilkan 96,42% untuk pembagian data 90:10, untuk pembagian data 80:20 menghasilkan tingkat akurasi sebesar 96,42%, dan pembagian data dengan rasio 70:30 menghasilkan akurasi 95,91%. Dengan demikian hasil menunjukkan bahwa algoritma Random Forest lebih akurat dibanding dengan K-Nearest Neighbors dalam menentukan tingkat kelulusan mahasiswa fakultas teknik.

3. Dari hasil penelitian yang dilakukan maka dapat disampaikan beberapa saran , yaitu:

4. Penelitian selanjutnya disarankan untuk menggunakan dataset yang lebih besar dan lebih beragam untuk menguji lebih lanjut keakuratan model yang dibangun. Penelitian selanjutnya juga dapat mempertimbangkan untuk menggunakan lebih banyak atribut atau fitur yang mungkin mempengaruhi kelulusan mahasiswa, seperti data kegiatan ekstrakurikuler, partisipasi dalam proyek, dan data kehadiran.

5. Disarankan untuk melakukan analisis lebih mendalam terhadap fitur-fitur yang memiliki pengaruh signifikan terhadap prediksi kelulusan, guna memberikan wawasan lebih lanjut bagi pihak akademik dalam meningkatkan kualitas pendidikan.

6. Sebagai tindak lanjut dari penelitian ini, dapat dikembangkan sistem prediksi yang terintegrasi dalam sistem informasi akademik (SIMAK) untuk membantu pihak universitas dalam memonitor dan meningkatkan tingkat kelulusan mahasiswa.

**Referensi**

[1] Alber, J. (2023). Supervised Learning Adalah: Pengertian, Konsep dan Contoh. KANTINIT. https://kantinit.com/kecerdasan-buatan/supervised-learning-adalah-pengertian-konsep-dan-contoh/

[2] Alkalah, C. (2016). PERATURAN AKADEMIK UNIVERISTAS MUHAMMADIYAH MAKASSAR. 19(5), 1–23.

[3] Amalia Yunia Rahmawati. (2020). ANALISA PERBANDINGAN KINERJA ALGORITMA K\_NEAREST NEIGHBOR DAN ADAPTIVE BOOSTING PADA PREDIKSI PENERIMA BANTUAN SOSIAL PANGAN NON TUNAI. July, 1–23. http://repositori.unsil.ac.id/5686/1/1. COVER.pdf

[4] Anggreani, D., Herman, & Astuti, W. (2018). Kinerja Metode Naïve Bayes dalam Prediksi Lama Studi Mahasiswa Fakultas Ilmu Komputer. Seminar Nasional Ilmu Komputer Dan Teknologi Informasi, 3(2), 107–111. http://e-journals.unmul.ac.id/index.php/SAKTI/article/view/1843

[5] Benri, M., Metisen, H., & Latipa, S. (2015). Analisis Clustering Menggunakan Metode K-Means Dalam Pengelompokkan Penjualan Produk Pada Swalayan Fadhila. Jurnal Media Infotama, 11(2), 110–118. https://core.ac.uk/download/pdf/287160954.pdf

[6] Dedi Saputra, M. P. (2020). Data Mining. MC Project. https://blog.mycoding.id/2020/12/data-mining-teknik-data-mining-proses.html

[7] Elfaladonna, F., & Rahmadani, A. (2019). Analisa Metode Classification-Decission Tree Dan Algoritma C.45 Untuk Memprediksi Penyakit Diabetes Dengan Menggunakan Aplikasi Rapid Miner. SINTECH (Science and Information Technology) Journal, 2(1), 10–17. https://doi.org/10.31598/sintechjournal.v2i1.293

[8] Hasan, I. K., Resmawan, R., & Ibrahim, J. (2022). Perbandingan K-Nearest Neighbor dan Random Forest dengan Seleksi Fitur Information Gain untuk Klasifikasi Lama Studi Mahasiswa. Indonesian Journal of Applied Statistics, 5(1), 58. https://doi.org/10.13057/ijas.v5i1.58056

[9] Irawan, I., Qisthiano, R., Syahril, M., & Jakak, P. M. (2023). Optimasi Prediksi Kelulusan Tepat Waktu: Studi Perbandingan Algoritma Random Forest dan Algoritma K-NN Berbasis PSO. Jurnal Pengembangan Sistem Informasi Dan Informatika, 4(4).

[10] Iriadi, N., Setioningtias, L., & Priatno, P. (2021). Implementasi Data Mining Pada Klasifikasi Ketidakhadiran Pegawai Menggunakan Metode C4.5. Computer Science (CO-SCIENCE), 1(1), 53–61. https://doi.org/10.31294/coscience.v1i1.198

[11] Justin, E. (2023). perbandingan algoritma Naïve Bayes dan Support Vector Machine dalam memprediksi kelulusan mahasiswa tepat waktu (studi kasus: Universitas Pelita Harapan Kampus Medan). UNIVERSITAS PELITA HARAPAN MEDAN.

[12] Mujadilah, S. (2023). PREDIKSI TINGKAT KELULUSAN MAHASISWA MENGGUNAKAN K-MEANS PADA PROGRAM STUDI INFORMATIKA UNISMUH MAKASSAR. Universitas Muhammadiyah Makassar.

[13] Oon Wira Yuda, Darmawan Tuti, Lim Sheih Yee, & Susanti. (2022). Penerapan Penerapan Data Mining Untuk Klasifikasi Kelulusan Mahasiswa Tepat Waktu Menggunakan Metode Random Forest. SATIN - Sains Dan Teknologi Informasi, 8(2), 122–131. https://doi.org/10.33372/stn.v8i2.885

[14] Ramadhan, B., Firdaus, D., & Adiningrum, N. T. R. (2023). Analisis Data Pegawai Untuk Memprediksi Gaji Berdasarkan Faktor-Faktor Spesifik Dengan Pendekatan Machine Learning. Naratif : Jurnal Nasional Riset, Aplikasi Dan Teknik Informatika, 5(2), 131–139. https://doi.org/10.53580/naratif.v5i2.205

[15] Suwardika, I. G. I., Suariana, I. G. N., Bhiantara, I. P., & Arso, N. Y. (2019). Prediksi Lama Studi Menggunakan Naïve Bayes. Mobile and Forensics, 2(1), 41.

[16] Trivusi. (2022). Algoritma Random Forest. Trivusi. https://www.trivusi.web.id/2022/08/algoritma-random-forest.html

[17] Yanti, C. P., Agustini, N. W. E., & ... (2023). Perbandingan Metode K-NN Dan Metode Random Forest Untuk Analisis Sentimen pada Tweet Isu Minyak Goreng di Indonesia. Jurnal Media …, 7(April), 756–765. https://doi.org/10.30865/mib.v7i2.5900