Prosiding Seminar Nasional Sains dan Teknologi Seri III
Fakultas Sains dan Teknologi, Universitas Terbuka
Vol. 2 No. 1 (2025)
e-ISSN: 3047-6569

KLASIFIKASI SEKUENSIAL UNTUK PENGELOMPOKAN DATA ABSENSI KARYAWAN MENGGUNAKAN ALGORITMA DECISION TREE DAN FEATURE ENGINEERING

Ade Rifal Fauzi E. R. R.*, Ika Nur Laily Fitriana
Program Studi Statistika, Universitas Terbuka, Tangerang Selatan

**Penulis korespondensi: aderifalfauzi@gmail.com*

ABSTRAK

Pendataan absensi karyawan di PT Sum Hing Indonesia masih mengandalkan metode manual meskipun menggunakan perangkat lunak *Microsoft Excel*. Proses ini memerlukan pengurutan, pencarian pasangan data, dan pengecekan data absensi karyawan dari mesin absensi secara manual oleh staf *payroll*, dilakukan hari demi hari. Terkadang metode manual tersebut menyebabkan keterlambatan dan ketidakakuratan dalam pelaporan absensi. Penelitian ini bertujuan untuk mengotomatisasi proses pendataan absensi karyawan dengan menggunakan model *decision tree* yang didukung dengan *feature engineering*. Model ini dirancang untuk menyelesaikan masalah utama, yaitu *sequential classification*. Dengan mengimplementasikan *decision tree* dan *feature engineering*, model dapat mengklasifikasikan data tap masuk dan tap keluar karyawan secara otomatis sebagai pasangan data, menghasilkan rekaman absensi harian yang telah dikelompokan berdasarkan pasangan data masuk dan data keluarnya dengan tingkat ketelitian yang lebih tinggi dibandingkan metode manual dan model tanpa *feature engineering*. Hasil penelitian menunjukkan bahwa model dengan *feature engineering* lebih unggul dengan akurasi 99,98%, serta precision, recall, dan F1-score yang lebih tinggi, dengan struktur pohon yang lebih sederhana dan efisien daripada model tanpa *feature engineering*. Dengan demikian, model ini diharapkan dapat meningkatkan efisiensi operasional di PT Sum Hing Indonesia dan dapat diaplikasikan di perusahaan lain dengan kebutuhan serupa.

Kata Kunci: *Decision Tree, Feature Engineering, Sequential Classification*

1 PENDAHULUAN

Pendataan absensi karyawan adalah aspek penting dalam manajemen sumber daya manusia yang mempengaruhi keakuratan penggajian dan kepatuhan terhadap peraturan perusahaan. Di PT Sum Hing Indonesia, proses ini masih dilakukan secara manual dengan perangkat lunak *Microsoft Excel*, yang walaupun membantu pengorganisasian data tapi tetap kurang optimal. Setiap hari, staf payroll harus mengecek kecocokkan data masuk dan keluar karyawan. Proses ini membutuhkan waktu dan terkadang meningkatkan risiko kesalahan serta keterlambatan dalam pencatatan data absensi. Penggunaan perangkat lunak bawaan dari mesin absensi juga kurang optimal dan akurat, terutama karena beberapa karyawan memiliki jadwal kerja yang fleksibel, sehingga sulit untuk dijadwalkan secara rutin. Kondisi ini dapat menyebabkan ketidaksesuaian antara laporan yang dihasilkan perangkat lunak bawaan dengan realitas di lapangan, yang pada akhirnya dapat berdampak pada keterlambatan hasil laporan hingga keterlambatan penggajian akibat data absensi yang belum siap digunakan.

Dengan berkembangnya teknologi kecerdasan buatan (*Artificial Inteligence*) saat ini, solusi berbasis pemebelajaran mesin (*Machine Learning*) semakin relevan untuk meningkatkan efisiensi dan akurasi dalam pengelolaan data seperti absensi karyawan. Salah satu algoritma *machine learning* yang sesuai untuk tugas ini adalah *Decision Tree*. Algoritma ini sangat baik dalam menangani data klasifikasi dan juga dapat berjalan di komputer yang tidak menggunakan GPU tambahan (hanya CPU). Kelebihan lain dari *decision tree* adalah model yang transparan, model yang transparan mampu menjelaskan hasil prediksinya secara langsung tanpa memerlukan metode tambahan, sehingga proses pengambilan keputusan model dapat dipetakan secara jelas (Setiawan et al., 2023; Rudin, 2019). Ini berarti bahwa proses logika if-else yang digunakan dalam model *decision tree* dapat dilihat langsung dan diimplementasikan ke dalam program lain, misalnya dalam bentuk fungsi VBA di perangkat lunak *Microsoft Excel*. Implementasi ini memungkinkan prediksi langsung dilakukan di *Microsoft Excel*, sehingga mempercepat dan memudahkan proses pengelolaan absensi.

Penelitian ini bertujuan untuk mengembangkan dan menerapkan model pendataan absensi karyawan menggunakan teknik *decision tree* dengan tambahan *feature engineering*. Model ini dirancang untuk menyelesaikan dua masalah utama dalam pendataan absensi manual, yaitu klasifikasi sekuensial (*Sequential Classification*) dari runtutan data absensi serta masalah pencocokan pasangan/pengelompokan data keluar masuk (*Pairing Problem*). Dengan menggunakan *decision tree* dan *feature engineering*, data tap masuk dan tap keluar karyawan dapat dikelompokan secara otomatis, sehingga memungkinkan pembuatan rekaman absensi harian dengan cepat dan akurat.

Dengan penerapan *feature engineering*, model *decision tree* diharapkan dapat menjadi lebih kompeten dalam memproses data, menghasilkan keputusan yang lebih akurat. Ini tidak hanya meningkatkan efisiensi penggunaan sumber daya komputasi dalam menangani data absensi, tetapi juga menurunkan risiko kesalahan prediksi yang mungkin timbul dari model yang kurang terlatih dalam memahami data. Jadi dengan teknologi ini, perusahaan diharapkan dapat meningkatkan efisiensi dan akurasi dalam mengelola sumber daya manusia.

2 METODE

Metodologi penelitian ini disusun untuk menguraikan langkah-langkah sistematis yang diambil dalam proses pembangunan model *decision tree* guna mencapai tujuan penelitian. Bagian ini mencakup lima tahapan utama, dimulai dari pengumpulan data, diikuti oleh pra-pemrosesan data untuk memastikan kualitas dan konsistensi *dataset* (kumpulan data). Selanjutnya, dilakukan *feature engineering* untuk menciptakan fitur-fitur yang mudah dikenali oleh model pada proses pelatihan. Tahapan keempat adalah inisiasi model *decision tree*. Terakhir, model dilatih menggunakan data yang telah disiapkan pada proses sebelumnya hingga model dievaluasi untuk mengukur kinerja dan akurasinya dalam prediksi.

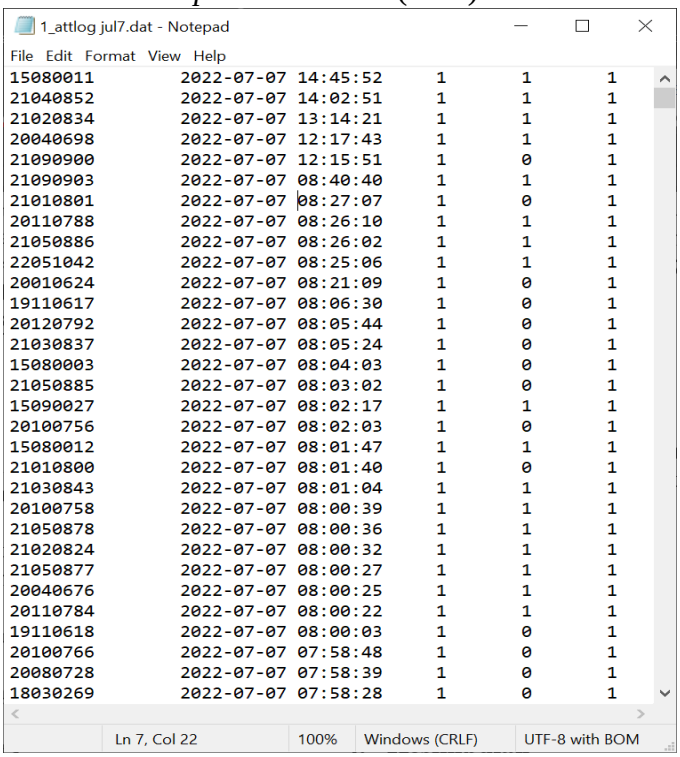
2.1 Pengumpulan Data

Data yang digunakan dalam penelitian ini berasal dari catatan absensi karyawan PT Sum Hing Indonesia yang dihasilkan oleh mesin absensi merk Solution tipe X100-C. *Dataset* ini terdiri dari 27.430 baris dengan 5 kolom, yaitu:

Tabel 1. Deskripsi kolom-kolom pada file log absensi

Kolom	Deskripsi	Nilai	Variabel
1	ID Karyawan	Angka	ID
2	*Timestamp* / tanggal dan jam	Teks format waktu (yyyy-mm-dd hh:mm:ss)	T
3	ID mesin absen	Angka	-
4	Kode status 0: In 1: Out 2: *Break In* 3: *Break Out* 4: Overtime In 5: *Overtime Out*	(0, 1, 2, 3, 4, 5)	IO
5	Kode metode absensi 0: *Password* 1: *Finger Print*	(0, 1)	-

Data yang diperoleh dari mesin absen ini berupa file dengan ekstensi .dat. Struktur data pada file tersebut menggunakan format *Tab Separated Values* (TSV).



Gambar 1. File attlog.dat, bentuk data yang dihasilkan mesin absensi

2.2 Pra- Pemrosesan Data

Pra-pemrosesan data merupakan tahap krusial dalam menyiapkan data mentah menjadi bentuk yang lebih dimengerti. Pada langkah awal pra-pemrosesan, kolom-kolom yang tidak relevan, seperti kolom 3 (ID Mesin) dan kolom 5 (Metode Absen), akan dihapus untuk memastikan bahwa hanya informasi yang relevan yang dipertahankan.

Selanjutnya, data diurutkan secara menaik berdasarkan kolom *timestamp* untuk memastikan urutan kronologis yang tepat. Setelah pengurutan berdasarkan *timestamp*, data kemudian diurutkan kembali berdasarkan kolom ID Karyawan. Pengurutan berdasarkan ID Karyawan dilakukan untuk memungkinkan model menemukan pasangan data absen untuk setiap individu pada waktu sebelum dan sesudahnya, ini merupakan salah satu teknik relevan yang bertujuan agar data runtutan absen memiliki ID yang berdekatan untuk setiap individu.

2.2.1. Feature Engineering

Feature engineering (rekayasa fitur) adalah proses mengubah data mentah menjadi fitur yang lebih bermakna untuk meningkatkan performa model *machine learning* seperti *decision tree*. Dengan langkah ini, model dapat lebih mudah mengenali pola dalam data, sehingga menghasilkan prediksi atau klasifikasi yang lebih akurat (Dong & Liu, 2018; Sitorus, Rizal, & Jajuli, 2020). Proses ini membantu menyajikan informasi yang relevan namun tidak langsung terlihat dari dataset asli, sehingga memperkaya kemampuan analisis model (Han, Pei, & Tong, 2022). Sebaliknya, pendekatan tanpa *feature engineering* hanya memanfaatkan fitur asli dari *dataset*, mengandalkan kemampuan alami *decision tree* dalam menemukan pola melalui pembagian yang dinilai paling optimal oleh model.

Data yang dimiliki diproses untuk menghasilkan 15 Jenis fitur dari tiga kolom data asli. Fitur-fitur ini dirancang untuk menangkap pola sekuensial yang lebih intuitif untuk dimengerti model. Fitur dirancang dengan menggunakan serangkaian logika seperti layaknya kegiatan manusia dalam membuat laporan absensi, memungkinkan model untuk lebih efektif dalam mengklasifikasikan pola-pola absensi yang ada. Fitur-fitur yang dihasilkan meliputi:

Fitur 1, Fitur 2 Perbandingan ID Karyawan. fitur diperoleh dengan membandingkan ID Karwayan pada i (runtun) selanjutnya (ID_{i+1}) apakah sama dengan ID Karwayan pada i saat ini (ID_i) untuk f_1:

$$f_1 = \begin{cases} 1 & ID_i \in ID_{i+1} \\ 0 & Lainnya. \end{cases} \quad (1)$$

Fitur yang sejenis juga dibuat untuk f_2 dengan membandingkan ID Karwayan pada i sebelumnya (ID_{i-1}) apakah sama dengan ID Karwayan pada i saat ini (ID_i).

$$f_2 = \begin{cases} 1 & ID_i \in ID_{i-1} \\ 0 & Lainnya. \end{cases} \quad (2)$$

Fitur 3, Fitur 4, Fitur 5. Pada bagian ini *fitur 3,4,5* dihasilkan dengan perbadingan logika apakah selisih waktu antar i tertentu kurang dari satu hari?. untuk f_3 memiliki logika ”apakah selisih antara waktu pada i selanjutnya dan i saat ini kurang dari satu hari?” $(T_{i+1} - T_i < 1)$:

$$f_3 = \begin{cases} 1 & T_{i+1} - T_i \leq 1 \\ 0 & Lainnya. \end{cases} \tag{3}$$

Untuk f_4, membandingkan apakah selisih antara waktu pada i saat ini dikurangi i sebelumnya kurang dari satu hari? $(T_i - T_{i-1} < 1)$:

$$f_4 = \begin{cases} 1 & T_i - T_{i-1} \leq 1 \\ 0 & Lainnya. \end{cases} \tag{4}$$

Untuk f_5, membandingkan apakah selisih antara waktu pada i selanjutnya dikurangi dengan i sebelumnya adalah kurang dari satu hari? $(T_{i+1} - T_{i-1} < 1)$:

$$f_5 = \begin{cases} 1 & T_{i+1} - T_{i-1} \leq 1 \\ 0 & Lainnya. \end{cases} \tag{5}$$

Fitur 6, Fitur 7, Fitur 8. Fitur fitur ini berisikan fungsi untuk melakukan ekstraksi jam, menit dan detik untuk setiap stampel waktu (*timestamp*) pada T_{i-1}, T_i dan T_{i+1}. Sebagai catatan, bentuk data *timestamp* (T) yang dimiliki sudah diubah ke dalam bentuk sistem *Microsoft Office Excel 1900 Date System*, sehingga hanya perlu mengurangi nilai T_i dengan $\lfloor T_i \rfloor$ dan menghasilkan nilai desimal dengan rentang 0-1 yang merupakan representasi dari jumlah detik dalam satu hari (Held, Moriarty, & Richardson, 2019).

$$f_6 = T_{i-1} - \lfloor T_{i-1} \rfloor$$

$$f_7 = T_i - \lfloor T_i \rfloor$$

$$f_8 = T_{i+1} - \lfloor T_{i+1} \rfloor \tag{6}$$

Fitur 9, Fitur 10, Fitur 11. Fitur-fitur ini berisikan kode keluar/masuk pada kolom 4 (IO), saat karyawan menggunakan mesin absen. Dari keenam kode yang ada, akan dikelompokan menjadi 2 jenis yang diinisiasikan kedalam 2 set $I = \{0,2,4\}$ dan $O = \{1,3,5\}$ sebagai kode masuk dan keluar.

$$f_9 = \begin{cases} 0 & IO_{i-1} \in I \\ 1 & IO_{i-1} \in O \end{cases}$$

$$f_{10} = \begin{cases} 0 & IO_i \in I \\ 1 & IO_i \in O \end{cases}$$

$$f_{11} = \begin{cases} 0 & IO_{i+1} \in I \\ 1 & IO_{i+1} \in O \end{cases}$$

(7)

Fitur 12, Fitur 13, Fitur 14, Fitur 15. Keempat fitur ini berisikan logika untuk membandingkan hari dan menghitung rentang waktu antara i saat ini dengan i sebelum atau sesudahnya. f_{12} berisi logika untuk mengecek apakah hari ini $\lfloor T_i \rfloor$ merupakan hari yang sama dengan $\lfloor T_{i+1} \rfloor$.

$$f_{12} = \begin{cases} 1 & \lfloor T_i \rfloor \in \lfloor T_{i+1} \rfloor \\ 0 & Lainnya. \end{cases}$$

(8)

f_{13} berisi rentang waktu antar i saat ini dengan i selanjutnya.

$$f_{13} = T_{i+1} - T_i$$

(9)

f_{14} mirip dengan f_{12}, namun disini berfungsi untuk membandingkan apakah $\lfloor T_i \rfloor$ merupakan hari yang sama dengan i sebelumnya $\lfloor T_{i-1} \rfloor$.

$$f_{14} = \begin{cases} 1 & \lfloor T_i \rfloor \in \lfloor T_{i-1} \rfloor \\ 0 & Lainnya. \end{cases}$$

(10)

Dan f_{15} adalah rentang waktu antara i saat ini T_i dengan i sebelumnya T_{i-1}.

$$f_{15} = T_i - T_{i-1}$$

(11)

Setelah alur proses *feature engineering* dikembangkan, peneliti melakukan anotasi manual pada data untuk memberikan label 0 dan 1, yang masing-masing merepresentasikan status "lanjut" dan "berhenti" guna menentukan suatu kelompok/pasangan data absensi. Proses ini menghasilkan total 27.430 baris data dengan rentang waktu mulai dari 8 April 2022 hingga 20 Juli 2022. Data yang telah dianotasi tersebut kemudian digunakan sebagai data *training* dan data *testing* dalam pengembangan model.

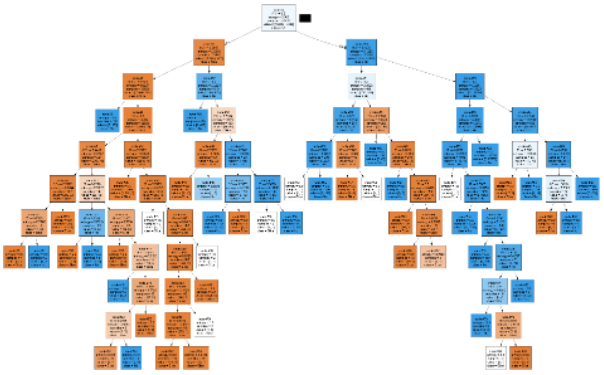
id	timestamp	tap	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	f12	f13	f14	f15	y
15080002	44748,7105	1	1	1	1	1	1	0,3269	0,7105	0,3260	1	0	1	0	0,6155	1	0,3836	1
15080002	44749,3260	0	1	1	1	1	1	0,7105	0,3260	0,7200	0	1	0	1	0,3940	0	0,6155	0
15080002	44749,7200	1	1	1	1	1	1	0,3260	0,7200	0,3265	1	0	1	0	0,6064	1	0,3940	1
15080002	44750,3265	0	1	1	1	1	1	0,7200	0,3265	0,7693	0	1	0	1	0,4429	0	0,6064	0
15080002	44750,7693	1	1	1	0	1	0	0,3265	0,7693	0,3261	1	0	1	0	2,5567	1	0,4429	1
15080002	44753,3261	0	1	1	1	0	0	0,7693	0,3261	0,9973	0	1	0	1	0,6713	0	2,5567	0
15080002	44753,9973	1	1	1	1	1	1	0,3261	0,9973	0,3228	1	0	1	0	0,3254	1	0,6713	1
15080002	44754,3228	0	1	1	1	1	1	0,9973	0,3228	0,7190	0	1	0	1	0,3962	0	0,3254	0
15080002	44754,7190	1	1	1	1	1	1	0,3228	0,7190	0,3331	1	0	1	0	0,6142	1	0,3962	1
15080002	44755,3331	0	1	1	1	1	1	0,7190	0,3331	0,7280	0	1	0	1	0,3949	0	0,6142	0
15080002	44755,7280	1	1	1	1	1	1	0,3331	0,7280	0,3292	1	0	1	0	0,6012	1	0,3949	1
15080002	44756,3292	0	1	1	1	1	1	0,7280	0,3292	0,6693	0	1	0	1	0,3401	0	0,6012	0
15080002	44756,6693	1	1	1	0	1	0	0,3292	0,6693	0,3253	1	0	1	0	3,6560	1	0,3401	1
15080002	44760,3253	0	1	1	1	0	0	0,6693	0,3253	0,7183	0	1	0	1	0,3931	0	3,6560	0
15080002	44760,7183	1	1	1	1	1	1	0,3253	0,7183	0,3205	1	0	1	0	0,6021	1	0,3931	1
15080002	44761,3205	0	1	1	1	1	1	0,7183	0,3205	0,7247	0	1	0	1	0,4042	0	0,6021	0
15080002	44761,7247	1	1	1	1	1	1	0,3205	0,7247	0,3341	1	0	1	0	0,6094	1	0,4042	1
15080002	44762,3341	0	0	1	1	1	1	0,7247	0,3341	0,3370	0	1	1	0	-99,9971	0	0,6094	1
15080003	44662,3370	0	1	0	1	1	1	0,3341	0,3370	0,7094	1	1	0	1	0,3724	0	-99,9971	0
15080003	44662,7094	1	1	1	1	1	1	0,3370	0,7094	0,3337	1	0	1	0	0,6243	1	0,3724	1
15080003	44663,3337	0	1	1	1	1	1	0,7094	0,3337	0,6267	0	1	0	1	0,2930	0	0,6243	0
15080003	44663,6267	1	1	1	1	1	1	0,3337	0,6267	0,3260	1	0	1	0	0,6993	1	0,2930	1
15080003	44664,3260	0	1	1	1	1	1	0,6267	0,3260	0,7098	0	1	0	1	0,3838	0	0,6993	0
15080003	44664,7098	1	1	1	1	1	1	0,3260	0,7098	0,3318	1	0	1	0	0,6219	1	0,3838	1
15080003	44665,3318	0	1	1	1	1	1	0,7098	0,3318	0,7099	0	1	0	1	0,3781	0	0,6219	0
15080003	44665,7099	1	1	1	0	1	0	0,3318	0,7099	0,3331	1	0	1	0	3,6231	1	0,3781	1
15080003	44669,3331	0	1	1	1	0	0	0,7099	0,3331	0,7108	0	1	0	1	0,3777	0	3,6231	0

Gambar 2. Data yang telah melalui pra-pemrosesan, *feature engineering* dan anotasi

2.3 Model Decision Tree

Decision Tree adalah salah satu model Pembelajaran Mesin (*Machine Learning*) dalam domain kecerdasan buatan yang digunakan untuk tugas klasifikasi dan regresi, dengan cara mempartisi *dataset* berdasarkan aturan keputusan. Model ini bekerja dengan memecah data secara rekursif ke dalam kelompok yang lebih homogen, yang memudahkan interpretasi hasil (Sinambela et al., 2023). Setiap simpul (*node*) internal pada pohon merepresentasikan batasan/*threshold* untuk fitur tertentu, sementara simpul daun (*leaf node*) memberikan prediksi label atau nilai target (Charbuty & Abdulazeez, 2021). Metode ini fleksibel karena dapat menangani data numerik maupun kategorikal serta mampu mempelajari hubungan yang kompleks. Breiman (2017) menyebutkan bahwa struktur hierarki *decision tree* membuatnya menjadi alat analisis yang intuitif dan powerful dalam berbagai domain aplikasi.

Salah satu algoritma populer untuk membangun model *decision tree* adalah CART (*Classification and Regression Tree*), yang diperkenalkan oleh Breiman dan koleganya. CART menghasilkan pohon biner di mana setiap simpul memiliki dua cabang, menggunakan Gini Impurity sebagai kriteria utama untuk tugas klasifikasi. Implementasi CART telah diadopsi oleh banyak pustaka (*library*) *machine learning* modern, termasuk *Scikit-Learn*, karena efisiensi dan kemampuannya untuk menghasilkan model yang sederhana namun akurat (Breiman et al., 2017; Pedregosa et al., 2011). Algoritma ini dikenal luas karena performanya yang handal, terutama dalam mengelola data yang kompleks dan besar.



Gambar 3. Salah satu bentuk model *decision tree* yang telah divisualisasikan

Sebelumnya, proses *feature engineering* telah menghasilkan *dataset* x yang terdiri dari 15 fitur $x_i = [f_1, f_2, \dots, f_{15}] \in \mathbb{R}^n$ dan label kelas $y \in \mathbb{R}^l$ yang terdiri dari kelas 0 dan 1. Setiap fitur x_i merupakan atribut-atribut yang menggambarkan setiap sampel dalam *dataset*, sedangkan label y adalah kelas atau kategori yang akan diprediksi oleh model. Kategori 0 pada label y untuk memberikan klasifikasi bahwa kelompok sekuen absensi untuk individu tersebut di tanggal tersebut masih berlanjut ke data di i selanjutnya dan kategori 1 memberikan klasifikasi bahwa kelompok sekuen absensi selesai pada data tersebut dan menjadi sebuah penutup kelompok.

Lanjut ke pembahasan model, pada setiap node m dalam *decision tree*, data Q_m yang terdapat pada *node* tersebut akan dipisahkan menjadi dua subset berdasarkan suatu fitur x_j dan nilai ambang

batas (*threshold*) t_m. Pemisahan ini bertujuan untuk memaksimalkan homogenitas kelas dalam subset yang dihasilkan. Setiap kandidat pemisahan $\theta = (j, t_m)$ terdiri dari fitur *j* dan *threshold* t_m. Data di node m dipisahkan menjadi dua subset:

$$Q_m^{left}(\theta) = \{(x, y) | x_j \leq t_m\}$$

$$Q_m^{right}(\theta) = Q_m / Q_m^{left}(\theta) \tag{12}$$

Kemudian, untuk mengukur kualitas pemisahan, digunakan kriteria *Gini*, yang menghitung tingkat ketidakteraturan atau ketidakpastian dalam data pada setiap node. *Gini* pada node m dihitung dengan rumus:

$$H(Q_m) = \sum_k p_{mk} (1 - p_{mk}) \tag{13}$$

Di mana p_{mk} adalah proporsi dari kelas k pada node m, yang didefinisikan sebagai:

$$p_{mk} = \frac{1}{n_m} \sum_{y \in Q_m} I(y = k) \tag{14}$$

Dalam persamaan tersebut, n_m adalah jumlah sampel pada node m, dan $I(y = k)$ adalah fungsi indikator yang bernilai 1 jika label y sama dengan k, dan 0 jika tidak.

Setelah setiap pemisahan dilakukan, model mengevaluasi kualitas pemisahan tersebut degan menghitung nilai gain $G(Q_m, \theta)$, yang merupakan rata-rata *gini* tertimbang dari subset hasil pemisahan:

$$G(Q_m, \theta) = \frac{n_m^{left}}{n_m} H\left(Q_m^{left}(\theta)\right) + \frac{n_m^{right}}{n_m} H\left(Q_m^{right}(\theta)\right) \tag{15}$$

Di sini, n_m^{left} dan n_m^{right} masing-masing adalah jumlah sampel dalam subset kiri dan kanan setelah pemisahan. Model memilih pemisahan θ^* yang meminimalkan nilai *gini* tertimbang tersebut:

$$\theta^* = argmin_\theta G(Q_m, \theta) \tag{16}$$

Pemisahan terbaik dipilh karena menghasilkan subset data dengan homogenitas yang lebih tinggi, yang berarti ketidakpastian dalam prediksi merupakan yang terkecil (Wibawa et al., 2018). Proses pemisahan ini diulangi secara rekursif pada subset data $Q_m^{left}(\theta^*)$ dan $Q_m^{right}(\theta^*)$ hingga kondisi

penghentian tercapai. Kondisi penghentian ini dapat berupa kedalaman maksimum pohon (*maximum depth*) yang telah ditentukan, jumlah sampel minimum untuk membentuk sebuah daun (*minimun sample leaf*), atau ketika setiap node tersisa hanya mengandung satu kelas (Scikit-learn, n.d.).

2.4 Pelatihan & Evaluasi

Pada penelitian ini, model *decision tree* dilatih dengan dua pendekatan berbeda, dengan *feature engineering* (FE) dan tanpa *feature engineering*. *Dataset* yang digunakan dalam penelitian ini terdiri dari 27.430 baris data. Untuk pembagian data, *train-test split* digunakan dengan perbandingan 80:20, di mana 80% data digunakan untuk data pelatihan model, dan 20% sisanya digunakan sebagai data uji.

Dalam proses pelatihan, beberapa parameter diuji untuk menemukan konfigurasi terbaik dari model. Parameter *max_depth* (kedalaman maksimal) diuji dengan nilai dari 2 hingga 10. Kedalaman pohon yang lebih besar memungkinkan model menangkap pola yang lebih kompleks, namun juga berisiko menyebabkan *overfitting* terhadap data pelatihan, pohon yang lebih dalam dapat menghasilkan hasil dengan bias rendah tetapi varians tinggi, terutama pada *dataset* kecil (Breiman, 2017). Selain itu, *min_samples_leaf* dan *min_samples_split* diuji dengan nilai 2. Parameter ini mengatur jumlah minimum sampel yang diperlukan untuk membentuk daun/simpul akhir dan untuk membagi simpul pembagian pohon. Pengaturan ini mencegah model menjadi terlalu spesifik (*overfiting*) pada data pelatihan yang terbatas.

Evaluasi model dilakukan dengan menggunakan beberapa metrik. Akurasi mengukur persentase prediksi benar dari seluruh prediksi yang dibuat. Namun, akurasi bisa menyesatkan jika *dataset* tidak seimbang, sehingga metrik lain juga digunakan untuk menggambarkan performa model secara lebih mendalam. *Confusion Matrix* memberikan gambaran lebih lengkap tentang jumlah prediksi benar dan salah untuk setiap kelas. *Precision* mengukur proporsi prediksi positif yang benar dan dihitung dengan rumus:

$$Precision = \frac{TP}{TP + FP} \tag{17}$$

Recall (Sensitifitas) mengukur seberapa baik model dapat mendeteksi sampel positif, dihitung dengan rumus:

$$Recall = \frac{TP}{TP + FN} \tag{18}$$

Sedangkan *F1-Score* merupakan rata-rata harmonis antara *precision* dan *recall*, yang memberikan keseimbangan antara keduanya, terutama pada *dataset* dengan distribusi kelas yang tidak seimbang.

$$F_1\ Score = 2 \cdot \frac{Precision \cdot Recall}{Precision \times Recall} \tag{19}$$

True Negative (TN) mengacu pada jumlah data negatif yang diidentifikasi dengan benar oleh model. *False Positive* (FP) terjadi ketika data negatif diklasifikasikan secara keliru sebagai positif. Sementara itu, *True Positive* (TP) adalah data positif yang diprediksi dengan benar, dan *False Negative* (FN) adalah kebalikan dari *True Positive,* yaitu data positif yang salah diklasifikasikan sebagai negatif. (Hesar & Foing, 2024; Sang, Sutoyo, & Darmawan, 2021)

Untuk membandingkan kedua model, baik dengan *feature engineering* (FE) maupun tanpa *feature engineering*, dilakukan perbandingan grafis yang ditunjukan pada Gambar 6, di antaranya akurasi vs. *max_depth*, yang menunjukkan bagaimana kedalaman pohon memengaruhi akurasi model. Grafik lainnya adalah Jumlah simpul (*node*) vs. *max_depth*, yang menggambarkan jumlah simpul yang dihasilkan pada kedalaman tertentu. Jumlah *node* yang lebih banyak menunjukkan kompleksitas pohon, dimana kompleksitas pohon harus diseimbangkan untuk menghindari *overfitting* (Leiva et al., 2019). Untuk itu parameter *max_depth* dibutuhkan untuk mengendalikan kompleksitas pohon.

3 HASIL PENELITIAN

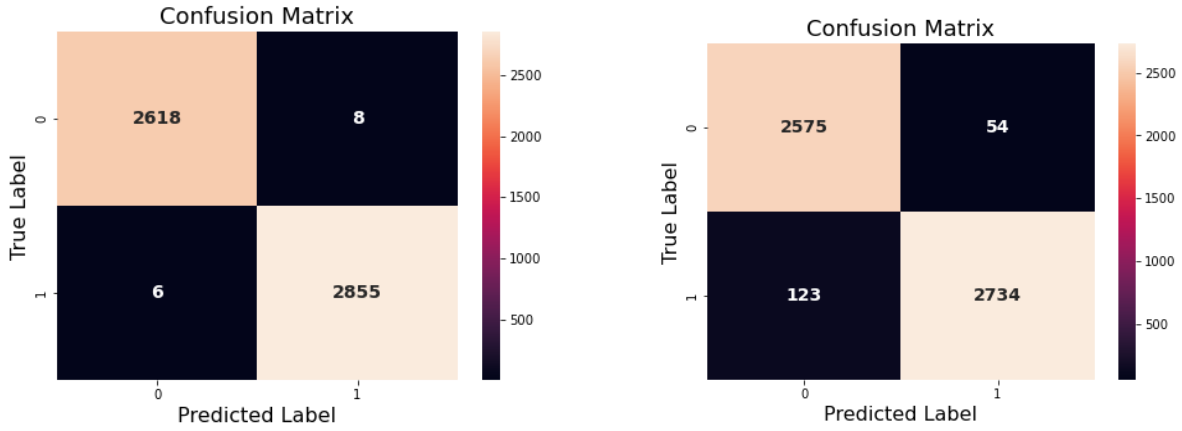
Penelitian ini melakukan eksperimen terhadap dua model *decision tree* yang digunakan untuk klasifikasi data absensi karyawan, yaitu model dengan *feature engineering* dan model tanpa *feature engineering*. Hasil menunjukkan bahwa model dengan *feature engineering* menghasilkan kinerja yang lebih baik dibandingkan model tanpa *feature engineering*. Dari pengukuran metrik *accuracy, precision, recall*, dan *F1-score*, model dengan *feature engineering* menunjukkan peningkatan dalam semua aspek.

Tabel 2. Hasil dari pengujian model dengan *feature engineering* dan tanpa *feature engineering*

Model	Max Depth	Precision	Recall	F1 Score	Acc.
Dengan FE	2	98.49%	98.49%	98.49%	98.49%
	3	99.22%	99.22%	99.22%	99.22%
	4	99.78%	99.78%	99.78%	99.78%
	5	99.89%	99.89%	99.89%	99.89%
	6	99.95%	99.95%	99.95%	99.95%
	7	99.95%	99.95%	99.95%	99.95%
	8	99.98%	99.98%	99.98%	99.98%
	9	99.98%	99.98%	99.98%	99.98%
	10	99.98%	99.98%	99.98%	99.98%
Tanpa FE	2	96.61%	96.57%	96.57%	96.57%
	3	96.61%	96.57%	96.57%	96.57%
	4	96.77%	96.74%	96.74%	96.74%
	5	96.81%	96.77%	96.77%	96.77%
	6	96.81%	96.77%	96.77%	96.77%
	7	96.79%	96.76%	96.76%	96.76%
	8	96.62%	96.59%	96.59%	96.59%
	9	96.48%	96.45%	96.45%	96.45%
	10	96.41%	96.37%	96.37%	96.37%

Model dengan *feature engineering* mencapai nilai recall sebesar 99.98%, yang menunjukkan kemampuan model ini dalam mendeteksi klasifikasi sekuensial secara lebih akurat. *Precision* dari model ini juga menunjukkan tingkat keakuratan prediksi yang lebih tinggi dengan nilai sebesar 99.98%. yang mencerminkan bahwa hasil prediksi dari model ini lebih sering benar daripada salah. *F1-score*, yang menggabungkan *precision* dan *recall*, juga lebih tinggi pada model dengan *feature engineering* daripada tanpa *feature engineering*, dengan nilai sebesar 99.98%.

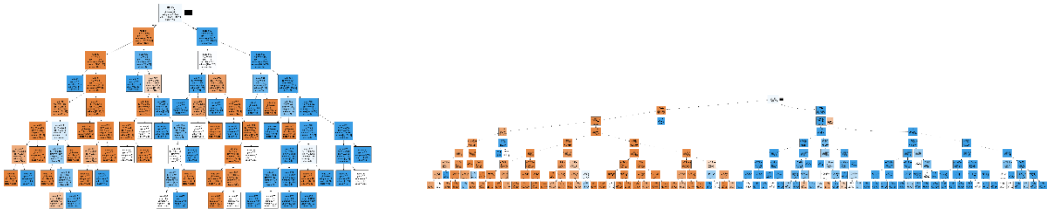
Matriks kebingungan (*confusion matrix*) untuk kedua model juga menampilkan distribusi yang berbeda dalam klasifikasi benar dan salah; model dengan *feature engineering* memiliki lebih sedikit kesalahan klasifikasi dibandingkan model lainnya.



(a) (b)

Gambar 4. Matriks Konfusi model (a) dengan *feature engineering* dan model (b) tanpa *feature engineering* pada *max-depth* 4

Selain itu, dari segi struktur model, gambar berikut menunjukkan visualisasi perbandingan struktur kedua model *decision tree*.



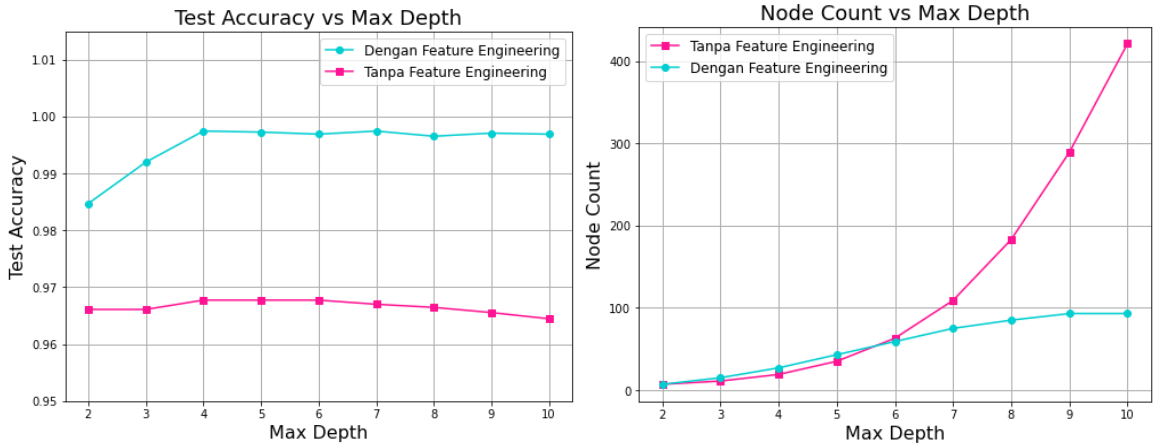
(a) (b)

Gambar 5. Struktur model pada *max-depth* 8. Terlihat perbedaan signifikan pada struktur model (a) dengan *feature engineering* dan model (b) tanpa *feature engineering*, masing masing model memiliki jumlah 85 dan 183

Pada model dengan *feature engineering*, terlihat bahwa pohon memiliki kedalaman yang lebih efektif dengan jumlah *node* yang lebih sedikit dengan akurasi yang lebih baik, karena fitur-fitur

tambahan membantu mengurangi kompleksitas pemisahan. Sebaliknya, model tanpa *feature engineering* terlihat lebih kompleks dengan struktur bercabang yang lebih luas, khususnya pada *max depth* yang lebih dalam (terlihat pada Gambar 6) saat model mencoba memahami data dengan lebih baik. Hal ini disebabkan oleh kurangnya fitur yang informatif, sehingga pohon harus memproses lebih banyak informasi dan mengembangkan *node* lebih banyak untuk mencapai pemisahan yang optimal.

Selain itu peneliti juga mengujicobakan pada data lain diluar data uji (*test data*) sebagai gambaran. Dapat dilihat pada Gambar 7, model berhasil membagi data absensi menjadi kelompok-kelompok yang masing-masing mencerminkan pasangan absensi pada waktu keluar-masuk. Selain pasangan data keluar-masuk model juga dapat mengelompokan data yang tidak lengkap seperti data masuk tanpa keluar dan data keluar tanpa masuk. Angka prediksi ”1” menandakan ujung/penutup dari kelompok dengan tepat seperti yang telah dihasilkan oleh hasil penelitian. Keberhasilan model dalam melakukan klasifikasi sekuensial ini menegaskan efektivitas metode ini yang menggunakan data yang sudah diproses melalui *feature engineering*, yang membantu model dalam mengenali pola absensi dan menghasilkan prediksi yang lebih akurat.



(a) (b)

Gambar 6. Grafik perbandingan (a) *max-depth* dengan *Accuracy* dan (b) *max-depth* dengan *node count* (jumlah simpul)

Id	timestamp	tap	predict
15090032	23/09/2022 07:29	0	0
15090032	23/09/2022 17:00	1	1
15090032	24/09/2022 07:27	0	0
15090032	24/09/2022 16:06	1	1
15090032	25/09/2022 07:20	0	1
15090034	12/09/2022 07:22	0	0
15090034	12/09/2022 07:23	0	0
15090034	12/09/2022 17:00	1	1
15090034	13/09/2022 17:05	1	1

Gambar 7. Model mampu mengelompokan runtutan absensi dengan membuat klasifikasi, dimana ujung kelompok diklasifikasikan sebagai angka prediksi 1

4 KESIMPULAN

Penelitian ini telah berhasil mengimplementasikan model klasifikasi sekuensial menggunakan algoritma *decision tree* dengan dukungan *feature engineering* pada data absensi karyawan PT Sum Hing Indonesia. Hasil penelitian menunjukkan bahwa penggunaan model *decision tree* dengan *feature engineering* memberikan dampak signifikan terhadap performa model. Dengan menambahkan fitur-fitur yang dirancang secara khusus untuk menangkap pola sekuensial dalam data absensi, model dapat mencapai akurasi hingga 99,98%, serta nilai *precision*, *recall,* dan *F1-score* yang sama tinggi yaitu sebesar 99,98%. Dibandingkan dengan model tanpa *feature engineering*, peningkatan performa mencakup akurasi yang lebih tinggi, efisiensi struktur pohon dengan jumlah simpul lebih sedikit, dan pemisahan data yang lebih optimal. Model yang dihasilkan mampu melakukan proses pengelompokan pasangan data masuk dan keluar, serta menangani kasus data yang tidak lengkap seperti tap masuk tanpa tap keluar atau sebaliknya. Hasil prediksi dari model ini mempermudah proses pencatatan absensi harian dengan lebih cepat dan akurat dibandingkan metode manual atau model tanpa *feature engineering*. Secara operasional, pendekatan ini menawarkan solusi yang lebih baik bagi perusahaan dalam meningkatkan efisiensi dan mengurangi kesalahan manusia dalam pengelolaan data absensi. Selain itu, metode ini dapat dengan mudah diadaptasi untuk digunakan di berbagai perusahaan lain yang memiliki kebutuhan serupa. Dengan demikian, penelitian ini tidak hanya berkontribusi pada digitalisasi proses bisnis di PT Sum Hing Indonesia tetapi juga berpotensi memberikan manfaat yang luas di berbagai sektor industri.

Untuk penelitian lebih lanjut, peneliti menyarankan untuk mengeksplorasi model yang lebih kompleks seperti *ensemble methods* (misalnya, *Random Forest* atau *XGBoost*) untuk membandingkan performa dan efisiensi model prediksi. alasan utama peneliti tidak menggunakan model seperti *XGBoost* atau *ensemble methods* lainnya dalam penelitian ini adalah kebutuhan teknis yang cepat saat penelitian dilakukan. Model *decision tree* dipilih karena kemampuannya yang unik dalam menyediakan logika keputusan yang eksplisit dan mudah diekstraksi. Hal ini memungkinkan implementasi langsung ke dalam fungsi *Microsoft Excel*, sehingga mempermudah pengguna akhir, terutama staf administrasi, untuk memahami dan menggunakan hasil model tanpa memerlukan pemahaman mendalam tentang algoritma *machine learning*. Selain pemilihan model lain, pengembangan *dataset* atau implementasi pada kasus lain juga akan memberikan pengetahuan lebih dalam mengenai kehandalan metode ini dalam praktiknya untuk menangani bentuk data yang berbeda.

DAFTAR PUSTAKA

Breiman, L., Friedman, J., Olshen, R. and Stone, C. (2017). Classification and regression trees. Routledge, Boca Raton.

Charbuty, B., & Abdulazeez, A. (2021). Classification based on decision tree algorithm for machine learning. *Journal of Applied Science and Technology Trends*, 2(1), 20-28.

Dong, G., & Liu, H. (Eds.). (2018). Feature engineering for machine learning and data analytics. CRC Press.

Han, J., Pei, J., & Tong, H. (2022). Data mining: Concepts and techniques. Morgan Kaufmann.

Held, B., Moriarty, B., & Richardson, T. (2019). Microsoft Excel functions and formulas with Excel 2019/Office 365. Mercury Learning and Information.

Hesar, F. F., & Foing, B. (2024). Evaluating Classification Algorithms: Exoplanet Detection using

Kepler Time Series Data. arXiv preprint arXiv:2402.15874.

Leiva, R. G., Anta, A. F., Mancuso, V., & Casari, P. (2019). A novel hyperparameter-free approach to decision tree construction that avoids overfitting by design. *Ieee Access*, 7, 99978-99987.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., … Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *The Journal of Machine Learning Research*, 12, 2825-2830.

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. arXiv [Stat.ML]. Diakses dari http://arxiv.org/abs/1811.10154

Sang, A. I., Sutoyo, E., & Darmawan, I. (2021). Analisis Data Mining Untuk Klasifikasi Data Kualitas Udara DKI Jakarta Menggunakan Algoritma Decision Tree Dan Support Vector Machine. *eProceedings of Engineering*, 8(5).

Scikit-learn. 1.10. Decision Trees. Diakses 10 September 2024, dari https://scikit-learn.org/dev/modules/tree.html#tree-mathematical-formulation

Setiawan, I., Cahyani, R. F. A., & Sadida, I. (2023). Exploring Complex Decision Trees: Unveiling Data Patterns And Optimal Predictive Power. *Journal of Innovation And Future Technology (IFTECH)*, 5(2), 112-123.

Sinambela, D. P., Naparin, H., Zulfadhilah, M., & Hidayah, N. (2023). Implementasi algoritma decision tree dan random forest dalam prediksi perdarahan pascasalin. *Jurnal Informasi dan Teknologi*, 58-64.

Sitorus, C. M., Rizal, A., & Jajuli, M. (2020). Prediksi risiko perjalanan transportasi online dari data telematik menggunakan algoritma Support Vector Machine. *Jurnal Teknik Informatika dan Sistem Informasi*, 6(2).

Wibawa, A. P., Guntur, M., Purnama, A., Akbar, M. F., & Dwiyanto, F. A. (2018). Metode-metode klasifikasi. *In Prosiding Seminar Ilmu Komputer dan Teknologi Informasi* (Vol. 3, No. 1).