BAB II

LANDASAN TEORI

2.1 Landasan Teori

2.1.1 Lama Rawat Inap Pasien

Length of Stay (LoS) adalah periode pasien menjalani perawatan di rumah sakit dari *admission* hingga *discharge*. LoS biasanya dihitung dalam satuan hari. Dalam laporan rumah sakit, LoS sering dirangkum dalam bentuk *Average Length of Stay* (ALoS), yaitu rata-rata durasi perawatan pasien dalam suatu periode tertentu. Secara umum, ALoS dihitung dari total hari perawatan (*bed-days*) dibagi jumlah pasien yang pulang (*discharge*) dalam periode tersebut. Ukuran ini banyak digunakan untuk melihat gambaran pemanfaatan tempat tidur dan efisiensi pelayanan rawat inap (World Health Organization European Region, 2024).

Dalam konteks klinis rawat inap umum, lama rawat inap sangat dipengaruhi oleh kondisi medis pasien. Pasien dengan diagnosis yang lebih berat atau kompleks biasanya membutuhkan waktu perawatan yang lebih lama dibandingkan pasien dengan kondisi ringan (Lee & Park, 2025). Selain itu, adanya komorbiditas (penyakit penyerta) juga dapat memperpanjang masa rawat karena penanganan menjadi lebih kompleks dan memerlukan pemantauan tambahan (Mustafa dkk., 2024). Literatur menunjukkan bahwa diagnosis utama, jumlah komorbiditas, serta tingkat keparahan kondisi saat masuk rumah sakit merupakan faktor klinis yang sering berhubungan dengan LoS yang lebih panjang.

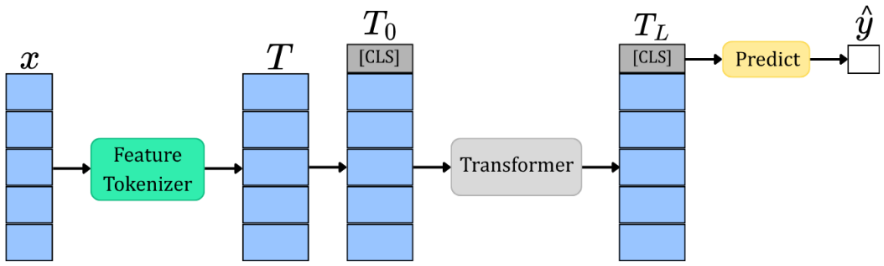
Selain kondisi saat masuk, perjalanan klinis selama dirawat juga mempengaruhi LoS. Jika selama perawatan terjadi komplikasi, seperti infeksi yang didapat di rumah sakit atau memburuknya kondisi penyakit, maka pasien biasanya memerlukan waktu rawat tambahan untuk stabilisasi dan terapi lanjutan. *The Agency for Healthcare Research and Quality (*AHRQ) menjelaskan bahwa pasien dengan kondisi klinis kompleks atau yang mengalami komplikasi cenderung memiliki LoS yang lebih panjang, sehingga pemantauan risiko klinis menjadi penting dalam manajemen rawat inap (Siddique dkk., 2021).

Prediksi LoS penting karena berkaitan langsung dengan perencanaan kapasitas tempat tidur, beban kerja klinis, biaya layanan, dan efisiensi operasional. Studi menunjukkan bahwa kemampuan memprediksi LoS sejak awal perawatan membantu rumah sakit mengoptimalkan pemanfaatan sumber daya dan meningkatkan manajemen pelayanan (Jain dkk., 2024). Prediksi LoS merupakan masalah yang cukup rumit karena dipengaruhi oleh sejumlah faktor yang saling berkaitan, seperti data demografis pasien, keadaan kesehatan, riwayat penyakit, riwayat rawat inap sebelumnya (Hirani dkk., 2025). Faktor-faktor tersebut biasanya tidak linier dan sulit untuk dimodelkan dengan tepat menggunakan metode statistik biasa. Maka dari itu, digunakan teknik *machine learning* untuk menggambarkan pola kompleks yang mempengaruhi durasi perawatan pasien.

2.1.2 FT-*Transformer*

FT-*Transformer* diperkenalkan dalam paper NeurIPS 2021 "*Revisiting Deep Learning Models for Tabular Data*" sebagai adaptasi *transformer* untuk data tabular yang memodelkan interaksi fitur dengan *self-attention* dan terbukti kompetitif pada

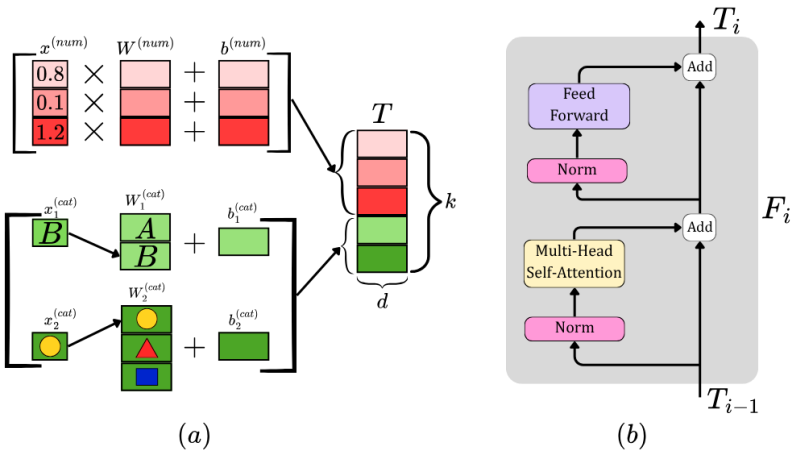
banyak tugas tabular (Gorishniy dkk., 2021). Model ini mengubah semua fitur numerik menjadi *embedding* dan menerapkan pada tumpukan lapisan *transformer* ke *embedding* tersebut. *Embedding* kemudian diproses oleh modul *Transformer* dan representasi akhir dari token [CLS] digunakan untuk prediksi seperti pada Gambar 2. 1



Gambar 2. 1 Arsitektur *Transformer*

Sumber: (Gorishniy dkk., 2021)

Proses tokenisasi fitur terdapat pada Gambar 2. 2, dimana pada gambar a terdapat tiga fitur numerik dan dua fitur kategorikal, sedangkan pada gambar b terdapat satu lapisan *transformer*.



Gambar 2. 2 (a) *Feature Tokenizer* (b) *One Transformer Layer*

Sumber: (Gorishniy dkk., 2021)

FT-*Transformer* membutuhkan lebih banyak sumber daya (baik perangkat keras maupun waktu) untuk pelatihan daripada model karena kompleksitas pada kuadratik *Multi-Head Self-Attention* (MHSA) standar dengan jumlah fitur (Gorishniy dkk., 2021).

FT-*Transformer* merupakan arsitektur *deep learning* yang dibuat khusus untuk memproses data tabular dengan mengadaptasi mekanisme *Transformer* yang awalnya dikembangkan untuk pemrosesan bahasa alami. Berbeda dengan model *Transformer* konvensional yang memproses token kata, FT-*Transformer* mengubah setiap fitur tabular (baik numerik maupun kategorikal) menjadi representasi vektor berdimensi tetap melalui lapisan *Feature Tokenizer*, kemudian memproses seluruh representasi fitur tersebut secara bersamaan menggunakan mekanisme *multi-head self-attention* (Gorishniy dkk., 2021).

Gambar 2.1 menunjukkan arsitektur FT-*Transformer*. Secara umum, proses dimulai dari data tabular x yang terdiri atas fitur numerik dan fitur kategorikal. Kedua jenis fitur tersebut terlebih dahulu diubah menjadi representasi token melalui *Feature Tokenizer* sehingga menghasilkan matriks token T. Selanjutnya, sebuah token khusus [CLS] ditambahkan pada baris pertama matriks token, membentuk matriks T_0 yang kemudian diteruskan sebagai masukan *Transformer*. Seluruh token diproses melalui L lapisan *Transformer Encoder* yang masing-masing terdiri atas *Multi-Head Self-Attention* dan *Feed Forward Network* (FFN), sehingga menghasilkan matriks keluaran T_L. Representasi token [CLS] pada T_L selanjutnya diteruskan menuju *prediction head* untuk menghasilkan nilai prediksi $\hat{y}$ sesuai dengan tugas pembelajaran, baik klasifikasi maupun regresi. Arsitektur FT-

Transformer secara umum terdiri atas tiga komponen utama, yaitu *Feature Tokenizer*, *Transformer Encoder* (yang terdiri atas *Multi-Head Self-Attention* dan *Feed Forward Network*), dan *Output Layer*.

2.1.2.1 *Feature Tokenizer Layer*

Feature Tokenizer Layer merupakan lapisan pertama yang bertugas mengubah setiap fitur input menjadi token *embedding* berdimensi d. Untuk fitur numerik, tokenisasi dilakukan dengan perkalian skalar antara nilai fitur dengan vektor bobot yang dapat dipelajari. Untuk fitur kategorikal, tokenisasi dilakukan melalui *embedding lookup* (Gorishniy dkk., 2021). Secara matematis, token *embedding* untuk fitur numerik ke-j dapat dirumuskan pada Persamaan 2. 1

$$T_j = x_j \,.\, W_j + b_j \tag{2. 1}$$

di mana:

x_j	: nilai fitur numerik ke-j
$W_j \in R^d$	: vektor bobot yang dapat dipelajari untuk fitur ke-j
$b_j \in R^d$	: vektor bias
$T_j \in R^d$	: token *embedding* hasil tokenisasi

Untuk fitur kategorikal, FT-*Transformer* menggunakan mekanisme *embedding lookup*. Setiap kategori terlebih dahulu direpresentasikan sebagai indeks kategori, kemudian dipetakan ke sebuah vektor *embedding* yang juga dipelajari selama proses pelatihan. Dengan demikian, kategori yang memiliki karakteristik serupa dapat memiliki representasi *embedding* yang saling berdekatan di ruang fitur

sehingga hubungan antar kategori dapat dipelajari secara lebih efektif (Gorishniy dkk., 2021).

Setelah seluruh token fitur terbentuk, FT-*Transformer* menambahkan sebuah token khusus yang disebut [CLS] (*Classification Token*) pada awal urutan token. Token ini tidak merepresentasikan fitur tertentu, melainkan berfungsi sebagai representasi global seluruh data masukan. Selama proses *self-attention*, token [CLS] berinteraksi dengan seluruh token fitur sehingga secara bertahap mengumpulkan informasi penting dari setiap fitur. Setelah seluruh lapisan *Transformer* selesai diproses, representasi akhir token [CLS] dipakai sebagai masukan bagi *prediction head* untuk menghasilkan keluaran model (Gorishniy dkk., 2021).

Seluruh fitur dari satu sampel data direpresentasikan sebagai matriks token $T \in R^{(n\ x\ d)}$, di mana n menyatkan jumlah fitur dan d adalah dimensi *embedding* (Gorishniy dkk., 2021).

2.1.2.2 *Transformer Encoder*

Matriks T_0 kemudian diproses oleh L lapisan *Transformer Encoder* yang ditumpuk secara berurutan, menghasilkan matriks keluaran T_L pada lapisan terakhir. Setiap lapisan terdiri atas dua sublapisan, yaitu *Multi-Head Self-Attention* dan *Feed Forward Network*, yang masing-masing disertai dengan *Residual Connection* dan *Layer Normalization* untuk meningkatkan stabilitas pelatihan, mempercepat konvergensi, serta mendukung aliran gradien pada jaringan yang lebih dalam (Vaswani dkk., 2023).

a. Multi-Head-Self-Attention

Multi-Head Self-Attention merupakan inti dari arsitektur *Transformer* yang memungkinkan model mempelajari hubungan dan interaksi antar fitur secara paralel. Setiap token fitur berperan sebagai *query, key,* dan *value* secara bersamaan (Vaswani dkk., 2023; Gorishniy dkk., 2021) *Mekanisme self-attention* dihitung dengan menggunakan persamaan berikut yang disajikan pada Persamaan 2. 2

$$Attention\ (Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right).V \tag{2. 2}$$

di mana $Q = TW^Q$, $K = TW^K$, dan $V = TW^V$ adalah matriks *query*, *key*, dan *value* yang merupakan hasil proyeksi linear dari token *embedding* T. d_k merupakan dimensi *key* yang berperan sebagai faktor penskalaan untuk menjaga kestabilan gradien (Vaswani dkk., 2023). Pada mekanisme *multi-head attention*, proses *attention* dilakukan secara paralel sebanyak h kali (jumlah *head*) dengan ruang proyeksi yang berbeda-beda (Gorishniy dkk., 2021), kemudian hasilnya digabungkan dan diproyeksikan kembali pada Persamaan 2. 3

$$MultiHead\ (Q,K,V) = Concat(head_1, \dots, head_h)\ .W^o \tag{2. 3}$$

b. Feed Forward Layer

Feed-Forward Layer merupakan lapisan jaringan saraf sederhana yang diterapkan secara independen pada setiap token setelah mekanisme *self-attention*. Lapisan ini tersusun dari dua transformasi linear dengan fungsi aktivasi ReLU di

antaranya, yang berfungsi untuk meningkatkan kapasitas representasi model (Vaswani dkk., 2023) yang ditunjukkan pada Persamaan 2. 4

$$FFN\ (x) = \max(0, xW_1 + b_1)\ .W_2 + b_2 \quad (2.\ 4)$$

Gorishniy dkk. (2021) merekomendasikan varian aktivasi ReGLU (*Rectified Gated Linear Unit*) sebagai pengganti ReLU pada Persamaan 2.4. ReGLU menggunakan mekanisme *gating* yang memungkinkan model secara selektif melewatkan atau memblokir informasi, sebagaimana didefinisikan pada Persamaan 2. 5 dan 2. 6

$$ReGLU\ (x) = a(x) \odot ReLU(b(x)), dimana\ [a(x); b(X)] = xW_1 + b_1 \quad (2.5)$$

$$FFN_{ReGLU}(x) = Dropout\big(a(x) \odot ReLU(b(x))\big)\ .W_2 + b_2 \quad (2.\ 6)$$

Penggunaan ReGLU memungkinkan model meningkatkan kapasitas representasi tanpa mengubah struktur dasar *Transformer,* sehingga kemampuan model dalam mempelajari hubungan antarfitur menjadi lebih optimal.

2.1.2.3 *Output Layer*

Output Layer merupakan lapisan terakhir yang menghasilkan prediksi berdasarkan representasi fitur yang telah dipelajari oleh FT-*Transformer*. Setelah melewati seluruh L lapisan *Transformer Encoder*, representasi token [CLS] pada matriks keluaran T_L, dinotasikan h_CLS, digunakan sebagai representasi global seluruh fitur masukan. Representasi ini kemudian diteruskan ke *prediction head*

untuk menghasilkan nilai prediksi ŷ sebagaimana dirumuskan pada Persamaan 2. 7 (Gorishniy dkk., 2021).

$$\hat{y} = W_{head} \,.\, (LN(h_{CLS})) + b_{head} \quad (2.\ 7)$$

Pada penelitian ini, FT-*Transformer* digunakan untuk tugas regresi sehingga keluaran model berupa satu nilai kontinu yang merepresentasikan prediksi LoS pasien.

2.1.3 SHAP

SHapley Additive exPlanations (SHAP) adalah kerangka kerja terpadu yang digunakan untuk menginterpretasikan prediksi model *machine learning* melalui pendekatan atribusi fitur (Lamane dkk., 2025). Dalam literatur *eXplainable Artificial Intelligence* (XAI), SHAP hadir sebagai solusi atas tantangan kotak hitam (*black box*) pada model-model kompleks seperti *ensemble learning* atau *deep learning* (Salih dkk., 2025). Metode ini mengadaptasi konsep *Shapley Values* dari teori permainan kooperatif yang diperkenalkan oleh Lloyd Shapley (1953), di mana setiap fitur dalam dataset dianggap sebagai pemain dan nilai Shapley untuk mengukur besarnya kontribusi setiap fitur terhadap prediksi yang dihasilkan oleh model (Qin dkk., 2025).

Keunggulan utama SHAP adalah kemampuannya memberikan penjelasan pada dua tingkatan, yaitu penjelasan lokal dan global. Penjelasan lokal menunjukkan faktor-faktor yang mempengaruhi prediksi untuk satu observasi tertentu, sedangkan penjelasan global diperoleh dengan mengagregasi seluruh

penjelasan lokal sehingga dapat diketahui faktor-faktor yang paling berpengaruh secara keseluruhan (S. M. Lundberg dkk., 2020).

SHAP mengukur kontribusi setiap fitur dengan membandingkan hasil prediksi saat fitur tersebut dimasukkan dan saat fitur tersebut tidak dimasukkan dalam perhitungan. Proses ini dilakukan pada berbagai kemungkinan kombinasi fitur yang disebut koalisi. Nilai SHAP yang diperoleh menunjukkan besarnya kontribusi suatu fitur terhadap selisih antara prediksi spesifik dengan nilai dasar (*base value*), yaitu rata-rata prediksi dari seluruh data (Ponce-Bobadilla dkk., 2024).

Mengingat adanya interaksi antar variabel yang dapat mempengaruhi kinerja model, SHAP menghitung rata-rata tertimbang berdasarkan kontribusi fitur tersebut di seluruh kemungkinan kombinasi fitur yang tersedia. Prinsip ini menjamin distribusi kontribusi yang adil, sehingga nilai prediksi akhir dapat didekomposisi secara aditif menjadi jumlah dari pengaruh masing-masing fitur. Hal ini memastikan bahwa model penjelasan tetap konsisten dan akurat secara lokal terhadap model aslinya (Qin dkk., 2025).

Nilai SHAP dihitung menggunakan formula nilai Shapley yang merepresentasikan kontribusi fitur i terhadap hasil prediksi. Nilai tersebut dihitung berdasarkan rata-rata kontribusi marjinal fitur pada seluruh kemungkinan kombinasi fitur lainnya sehingga menghasilkan atribusi yang adil dan konsisten untuk setiap fitur (S. Lundberg & Lee, 2017). Persamaan matematis untuk menentukan nilai atribusi ϕ_i didefinisikan pada Persamaan 2. 8

$$SHAP(\phi_i) = \sum_{S \subseteq N\{i\}} \frac{|S|!\,(N - |S| - 1)!}{N!} [\nu(S \cup \{i\} - \nu(S)] \quad (2.\ 8)$$

Dalam persamaan tersebut, N merupakan himpunan semua fitur input, sedangkan S adalah subset dari fitur tanpa fitur i dengan $|S|$ sebagai ukuran subset. Nilai $\nu(S \cup \{i\})$ menunjukkan prediksi ketika fitur i disertakan, dan $\nu(S)$ ketika tidak disertakan. Kontribusi fitur dihitung dengan membandingkan perubahan prediksi tersebut pada seluruh kemungkinan subset S, sehingga mencerminkan pengaruh rata-rata fitur i terhadap model. Kontribusi fitur i (ϕ_i) merepresentasikan pengaruh fitur tersebut terhadap hasil prediksi model (Gebreyesus dkk., 2024).

Nilai SHAP kemudian memenuhi sifat efisiensi atau *local accuracy*, di mana total kontribusi dari seluruh fitur (ϕ_i) ditambah dengan nilai dasar (ϕ_0) akan menghasilkan nilai prediksi model yang tepat, dengan M sebagai jumlah fiturnya (S. Lundberg & Lee, 2017). Prediksi model ditunjukkan oleh Persamaan 2. 9

$$f(x) = \phi_0 + \sum_{i=1}^{M} \phi_i \quad (2.\ 9)$$

Sifat matematis ini memastikan bahwa penjelasan yang dihasilkan tetap konsisten secara numerik dengan output model, sehingga interpretasi yang diberikan bersifat akurat dan dapat diandalkan (S. Lundberg & Lee, 2017).

2.1.4 *Data Preprocessing*

Data preprocessing merupakan tahapan kritis dalam membangun model *machine learning* yang andal, terutama pada data *healthcare* yang kompleks

seperti untuk prediksi LoS. Kualitas serta konsistensi data secara langsung menentukan performa dan generalisasi model, algoritma pembelajaran mesin sangat ditentukan oleh kualitas dan persiapan data input, di mana data klinis dunia nyata sering mengandung *missing values*, berbagai tipe data, dan potensi kesalahan, sehingga *preprocessing* data menjadi esensial untuk meningkatkan *robustness* dan performa model (Hidayaturrohman & Hanada, 2024). Tahapan *preprocessing* yang umum dilakukan meliputi penanganan *missing value*, seleksi fitur, standardisasi, dan *encoding* variabel kategorikal.

2.1.4.1 Pengecekan *Missing Value*

Data rekam medis sering kali mengandung nilai yang hilang (*missing values*) akibat kesalahan pencatatan, sistem yang terputus, atau prosedur klinis yang tidak dilakukan (Digitale dkk., 2025). Mengabaikan *missing value* dapat menyebabkan bias dalam analisis (Gurupur dkk., 2025).

2.1.4.2 Penghapusan Kolom yang Tidak Relevan

Seleksi fitur bertujuan untuk menurunkan dimensi data dengan mempertahankan variabel yang paling informatif dan menghilangkan yang redundan atau tidak relevan. Hal ini dapat meningkatkan kecepatan pelatihan model, mengurangi kompleksitas, dan mencegah *overfitting* (Rajeashwari & Arunesh, 2025). Kolom yang akan dihapus seperti id pasien, *visit date*, *discharged date*, dan *length of stay*.

2.1.4.3 *Feature Engineering*

Feature Engineering atau rekayasa fitur adalah proses mengubah data mentah menjadi representasi yang lebih optimal untuk meningkatkan kinerja model *machine learning*. Proses ini mencakup pembuatan fitur baru, penanganan nilai *missing*, serta transformasi variabel numerik dan kategorikal. Dalam konteks data tabular, rekayasa fitur menjadi semakin krusial karena data tabular seringkali memiliki tipe fitur yang heterogen (Butt dkk., 2026).

Salah satu teknik yang umum dilakukan adalah pembuatan fitur agregasi, di mana beberapa fitur biner digabungkan menjadi sebuah skor kumulatif. Pendekatan ini memungkinkan model untuk mengindetifikasi efek sinergis antar fitur secara lebih ringkas. Dalam prediksi lama rawat inap, agregasi indikator kondisi kesehatan pasien terbukti meningkatkan akurasi prediksi (Jain dkk., 2024).

2.1.4.4 *Encoding*

Encoding data kategorikal merupakan proses transformasi data yang nilainya terpilih dari sekelompok kategori atau label tertentu, seperti jenis kelamin atau ID fasilitas kesehatan, menjadi bentuk numerik yang dapat diolah oleh algoritma pembelajaran mesin (Arat, 2022). Proses ini menjadi tahap krusial dalam *preprocessing* karena mayoritas algoritma pembelajaran mesin, mulai dari regresi linier hingga jaringan saraf dalam, dirancang untuk memproses data dalam format numerik dan tidak dapat secara langsung memahami data dalam bentuk teks atau label. Pemilihan teknik *encoding* yang tepat berperan penting dalam menentukan kinerja model yang dihasilkan, karena *encoding* yang tidak sesuai dapat menyebabkan distorsi pada distribusi data, menciptakan hubungan ordinal

palsu antar kategori yang seharusnya independen, atau bahkan meningkatkan dimensi fitur secara drastis yang berujung pada *overfitting* (Pargent dkk., 2022).

2.1.4.5 Standardisasi Fitur Numerik

Fitur numerik sering kali memiliki skala pengukuran yang berbeda-beda, yang dapat mempengaruhi algoritma pembelajaran mesin yang sensitif terhadap perbedaan *magnitude* data. Oleh karena itu, diperlukan proses *feature scaling* untuk menyetarakan rentang nilai antar fitur (Wongoutong, 2024). *StandardScaler* (Z-*score normalization*) merupakan metode paling umum digunakan untuk mentransformasikan data agar memiliki nilai rata-rata 0 dan deviasi standar 1. Transformasi ini dihitung dengan rumus $X_{norm} = \frac{(x - \mu)}{\sigma}$ (Pinheiro dkk., 2025). *StandardScaler* efektif untuk algoritma yang mengasumsikan distribusi normal dan mempercepat konvergensi pada optimasi berbasis gradien.

2.1.5 Optuna

Optuna adalah *framework* optimasi *hyperparameter* otomatis yang dikembangkan oleh *Preferred Networks* dan pertama kali diperkenalkan oleh Akiba dkk., 2019 dalam publikasi di *Proceedings of the* 25th ACM SIGKDD *International Conference on Knowledge Discovery & Data Mining*. *Framework open-source* berbasis Python ini dikembangkan sebagai solusi atas berbagai keterbatasan yang terdapat pada metode konvensional, seperti *grid search* dan *random search*, terutama dalam hal efisiensi waktu dan kualitas hasil optimasi (Kee & Ho, 2025).

Menurut Akiba dkk., 2019 arsitektur ini dibangun di atas dua entitas utama, yaitu *study* yang merepresentasikan keseluruhan proses optimasi dan *trial* yang

merupakan eksekusi tunggal dari fungsi tujuan dengan serangkaian *hyperparameter* tertentu, sehingga memungkinkan pengelolaan proses optimasi secara efisien termasuk kemampuan paralelisasi dan kelanjutan optimasi yang terputus. Optuna menggunakan konsep *define-by-run* yang memungkinkan ruang pencarian (*search space*) ditentukan secara dinamis selama kode dijalankan. Pendekatan ini berbeda secara fundamental dengan metode tradisional yang bersifat statis, sehingga pendekatan tersebut memungkinkan konfigurasi *hyperparameter* disusun secara dinamis dengan fleksibilitas yang lebih tinggi serta struktur yang lebih modular. Pengguna dapat membangun ruang *hyperparameter* secara adaptif berdasarkan kondisi yang muncul pada saat *runtime*.

Keunggulan utama Optuna terletak pada kombinasi algoritma *Tree-structured Parzen Estimator* (TPE) sebagai sebagai strategi pengambilan sampel (*sampling*) dan mekanisme *pruning* yang mengimplementasikan *Median Stopping Rule*. TPE bekerja dengan memodelkan distribusi *hyperparameter* berdasarkan hasil *trial* sebelumnya, kemudian memisahkan distribusi tersebut menjadi dua kelompok, yaitu *trial* dengan performa baik dan buruk. Selanjutnya, proses sampling dilakukan dari distribusi yang memiliki kemungkinan lebih tinggi untuk menghasilkan nilai fungsi tujuan yang lebih optimal. Sementara itu, mekanisme *pruning* diterapkan untuk menghentikan pelatihan secara dini apabila metrik yang diamati menunjukkan performa model berada di bawah nilai median *trial* sebelumnya, sehingga meningkatkan efisiensi komputasi dengan mengurangi eksekusi yang tidak optimal (Akiba dkk., 2019).

Keunggulan Optuna dalam meningkatkan efisiensi dan akurasi model telah dibuktikan melalui berbagai studi empiris di berbagai penelitian. Sebagai contoh, hasil penelitian oleh Kee & Ho, 2025 diperoleh bahwa Optuna secara substansial mengungguli *Grid Search* dan *Random Search* dengan kecepatan 6,77 hingga 108,92 kali lebih cepat, dengan tetap mencapai nilai eror yang lebih rendah.

Di bidang kesehatan, Munshi dkk., 2025 berhasil meningkatkan akurasi model XGBoost untuk prediksi diabetes hingga mencapai 83% setelah dioptimasi dengan Optuna. Sementara itu, Gaib dkk., 2025 juga mendemonstrasikan peningkatan akurasi yang signifikan pada berbagai algoritma klasifikasi gangguan tidur, dari kisaran 93-96% menjadi 97-98% setelah penerapan Optuna. Dengan berbagai keunggulan dan bukti empiris tersebut, Optuna menjadi pilihan *framework* yang sangat mumpuni untuk melakukan optimasi *hyperparameter* dalam proses pengembangan model *machine learning* yang andal dan efisien.

2.1.6 Matriks Evaluasi

2.1.6.1 *Mean Absolute Error* (MAE)

Mean Absolute Error (MAE) dikenal juga sebagai L1 *loss* yang merupakan selisih absolut antara nilai yang diprediksi dengan nilai sebenarnya seperti yang terlihat pada Persamaan 2. 10

$$L1 = \left| y_{actual} - y_{predicted} \right| \tag{2. 10}$$

Gabungan dari semua nilai *loss* disebut *function cost*, di mana *function cost* untuk *absolute error* dikenal sebagai *mean absolute error*. *Mean Absolute Error*

adalah *loss function* yang sederhana namun efektif sehingga banyak digunakan dalam model regresi, terutama karena keberadaan *outlier* yang dapat menyebabkan distribusi variabel tidak sepenuhnya mengikuti distribusi Gaussian (Jadon dkk., 2022). Rumus MAE terdapat pada Persamaan 2. 11

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{2. 11}$$

di mana:

N : jumlah sampel data

y_i : nilai sebenernya

$\hat{y}_i$: nilai prediksi

2.1.6.2 *Root Mean Squared Error* (RMSE)

Root Mean Squared Error (RMSE) diperoleh dengan menghitung akar kuadrat dari *Mean Squared Error* (MSE). *Root Mean Square Deviation* adalah nama lain untuk RMSE. RMSE merupakan suatu metrik yang tidak hanya sekadar menghitung selisih antara nilai prediksi dan nilai aktual, tetapi juga secara implisit mempertimbangkan adanya variasi yang terkandung dalam nilai sebenarnya. Dalam proses perhitungannya, RMSE mengukur rata-rata besarnya kesalahan prediksi dengan menghitung kuadrat dari setiap perbedaan antara nilai aktual dan nilai prediksi, kemudian mengambil akar kuadrat dari rata-rata hasil kuadrat tersebut, sehingga menghasilkan suatu indikator yang merepresentasikan tingkat penyimpangan prediksi terhadap nilai sebenarnya dengan tetap memperhatikan variasi data aktual yang ada. RMSE memberikan gambaran mengenai besarnya

rata-rata kesalahan prediksi yang dihasilkan oleh model. RMSE dapat diterapkan pada berbagai fitur dalam model regresi karena membantu dalam mengevaluasi apakah suatu fitur berkontribusi terhadap peningkatan kinerja prediksi model atau tidak (Jadon dkk., 2022). Rumus RMSE terdapat pada Persamaan 2. 12

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{2. 12}$$

RMSE merepresentasikan besarnya rata-rata galat prediksi model dalam satuan yang sama dengan variabel yang diamati, sehingga dapat digunakan untuk menilai tingkat kesesuaian prediksi model terhadap data aktual. Nilai RMSE yang lebih rendah menunjukkan bahwa hasil prediksi semakin mendekati nilai sebenarnya, yang mencerminkan kinerja dan akurasi model yang lebih baik. Sebaliknya, nilai RMSE yang tinggi mengindikasikan adanya penyimpangan yang lebih besar antara hasil prediksi dengan nilai aktual (*ground truth*), sehingga residual yang dihasilkan juga cenderung lebih besar (Jadon dkk., 2022).

2.1.6.3 *Relative Absolute Error* (RAE)

Perhitungan kesalahan absolut relatif atau *Relative Absolute Error* (RAE) dilakukan dengan membagi total kesalahan absolut terhadap selisih absolut antara nilai aktual dan nilai rata-rata sebagai nilai acuan. Persamaan 2. 13 menunjukkan rumus RAE dengan Persamaan 2. 14 yang menunjukkan rumus perhitungan rata-rata.

$$RAE = \frac{\sum_{i=1}^{N} |y_i - \hat{y}_i|}{\sum_{i=1}^{N} |y_i - \bar{y}|} \quad (2.\ 13)$$

dimana

$$\bar{y} = \frac{1}{N} \sum_{i=1}^{N} (y_i) \quad (2.\ 14)$$

RAE merupakan metrik evaluasi berbasis rasio yang digunakan untuk mengukur tingkat kesalahan suatu model prediktif relatif terhadap model acuan. Nilai RAE berada pada rentang 0 hingga 1, dengan nilai yang semakin mendekati 0 menunjukkan bahwa model memiliki performa prediksi yang semakin baik. Nilai 0 merepresentasikan kondisi ideal, yaitu ketika kesalahan prediksi mencapai tingkat minimum. Metrik ini mengukur perbandingan antara rata-rata kesalahan absolut model dengan rata-rata penyimpangan nilai aktual terhadap nilai rata-ratanya (Jadon dkk., 2022). RAE menunjukkan sejauh mana rata-rata residual model lebih baik dibandingkan dengan pendekatan sederhana yang menggunakan nilai rata-rata sebagai prediktor.

2.1.6.4 *Relative Squared Error* (RSE)

Relative Squared Error (RSE) merupakan metrik evaluasi yang digunakan untuk menilai tingkat kesalahan prediksi suatu model dengan membandingkannya terhadap prediktor acuan yang lebih sederhana. Prediktor sederhana yang dimaksud umumnya berupa nilai rata-rata dari data aktual. Dalam perhitungannya, RSE diperoleh dengan membagi total kesalahan kuadrat model terhadap total kesalahan kuadrat yang dihasilkan oleh prediktor sederhana tersebut. Proses normalisasi ini

memungkinkan hasil evaluasi dapat dibandingkan antar model, meskipun kesalahan dihitung dalam satuan atau skala yang berbeda (Jadon dkk., 2022). RSE memberikan gambaran mengenai seberapa besar peningkatan kinerja model dibandingkan dengan pendekatan dasar yang hanya menggunakan nilai rata-rata sebagai prediksi.

Persamaan 2. 15 untuk menentukan kesalahan kuadrat relatif:

$$RSE = \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \hat{y})^2} \quad (2.\ 15)$$

dimana Persamaan 2. 16 merupakan rata-rata aktual prediktor sederhana.

$$\frac{1}{N}\sum_{i=1}^{N} y_i \quad (2.\ 16)$$

2.2 Penelitian Terdahulu

2.2.1 *State of the Art*

State of the Art akan menampilkan beberapa penelitian sebelumnya yang relevan terhadap penelitian ini seperti yang dijelaskan dalam Tabel 2. 1

Tabel 2. 1 *State of The Art*

No	Peneliti	Judul	Algoritma	Hasil Penelitian
1	(Peng & Gao, 2025)	*Prediction of hospital length of stay: leveraging ensemble tree models and intelligent feature selection*	CNN, XGBoost, GBM, dan *Random Forest*	Penelitian menggunakan dataset LoS dari Microsoft dan membandingkan beberapa model di mana model XGBoost dengan 40 *features* mengungguli model lain dengan MAE 0,2974, RMSE 0,3881, RAE 0,1553, dan RSE 0,0270.
2	(Mekhaldi dkk., 2021)	*A Comparative Study of Machine Learning Models for Predicting Length of Stay in Hospitals*	MLR, SVM, *Random Forest*, *Gradient Boosting*	Penelitian ini merupakan studi komparatif antar beberapa model *machine learning*. Dataset yang digunakan yaitu dataset LoS dari Microsoft. Model *Gradient Boosting* mengungguli model lain dengan akurasi tanpa SMOTER MAE 0,44 R^2 0,94 Adj 0,94 dan *time* 250,37 sedangkan hasil dengan SMOTER MAE 0,45 R^2 0,93 Adj 0,93 dan *time* 171,37.
3	(Balik, 2024)	*Comparing Federated Stochastic Gradient Descent and Federated Averaging for Predicting Hospital Length of Stay*	FedSGD, FedAVGv1, FedAVGv2	Penelitian ini membandingkan tiga metode *federated learning* dengan menggunakan dataset LoS dari Microsoft. Model FedSGD memberikan hasil terbaik dibandingkan dengan kedua versi FedAVG. Pada data uji, FedSGD mencapai rata-rata MSE sebesar 1.354, sedangkan FedAVGv1 dan FedAVGv2 masing-masing memperoleh MSE sebesar 1.798 dan 1.897.

No	Peneliti	Judul	Algoritma	Hasil Penelitian
4	(Puspita dkk., 2023)	*Hospital Length of Stay Prediction with Ensemble Learning Method*	RF *Ensemble Learning,* SVM, NN	Penelitian ini menerapkan model *ensemble learning Random Forest* yang dikombinasikan dengan banyak *decision tree* untuk memprediksi LoS, dengan menggunakan data *real* dari rumah sakit. *Random Forest* memberikan kinerja terbaik dibandingkan SVM dan NN, dengan nilai *precision* 0,87, *recall* 0,87, F1-*score* 0,87, dan MCC 0,56.
5	(Adawiyah dkk., 2021)	*Hospital Length of Stay Prediction Based on Patient Examination Using Neural Network*	*Neural Network*	Penelitian ini menerapkan model *Neural Network* yang dikombinasikan dengan teknik optimasi *Grid Search*, dataset yang digunakan merupakan dataset LoS publik yang diperoleh dari *platform* Kaggle. Model *Neural Network* dengan parameter *default* menghasilkan akurasi 89,22%, lalu dilakukan optimasi parameter menggunakan *Grid Search*, dan akurasi meningkat secara signifikan menjadi 92,20%.
6	(Mengistu dkk., 2026)	*Machine learning predicts prolonged patient length of stay in a resource constrained Ethiopian hospital*	LR, DT, RF, GB, MLP, SVM, KNN, NB	Penelitian menggunakan dataset dari Rumah sakit Debre Markos di Ethiopia yang berisi 27.268 data pasien dari tahun 2017-2024 dan membandingkan delapan model *machine learning* dan dilakukan SMOTE. Hasil penelitian menunjukkan model terbaik diperoleh *Gradient Boosting* dengan SMOTE, dimana AUC 0,70, Akurasi 81%, *Precision* 72% dan *Recall* 63%.
7	(Alsinglawi dkk., 2024)	*Predicting Hospital Stay Length Using Explainable Machine Learning*	XGBoost, RF, GB, LR, dan MLP	Penelitian menggunakan *Electronic Health Record* (EHR) dari Al-Ain Hospital (UAE) dengan membandingkan lima model. Model XGBoost menunjukkan performa terbaik dibandingkan model lain. Nilai AUC pada *validation set* mencapai 77,9%, sedangkan pada *testing set* sebesar 74,4%. Pada tahap *explainability* menggunakan XAI (SHAP & *ExplainerDashboard*), performa XGBoost terhadap klasifikasi akhir menunjukkan *accuracy* 94,6%, *precision* 91,4% (*short* LoS) dan 96% (*long* LoS), serta ROC-AUC 98%.

No	Peneliti	Judul	Algoritma	Hasil Penelitian
8	(Fa dkk., 2025)	*Prediction of prolonged length of stay from first 2 days of hospitalization data: a SHAP value-based variable selection method for a simplified model*	LR, SVM, MLP, dan XGBoost	Penelitian ini menggunakan data dari *Clinical Data Warehouse* (PREDIMED) sebuah rumah sakit universitas di Prancis dengan membandingkan empat algoritma. Model XGBoost dengan *undersampling* (XGB-US) menunjukkan performa terbaik dibandingkan model lainnya, dengan nilai AUC-ROC sebesar 0,802 dan F2-*score* sebesar 0,533. Setelah dilakukan seleksi variabel berbasis SHAP untuk menyederhanakan model, performa tetap stabil bahkan sedikit meningkat dengan AUC-ROC sebesar 0,804, meskipun jumlah variabel dikurangi secara signifikan.
9	(Jain dkk., 2024)	*Predicting hospital length of stay using machine learning on a large open health dataset*	LR, RF, CatBoost	Pada penelitian ini menggunakan dataset publik dari SPARCS 2016 berisi 2,3 juta catatan pasien yang telah dianonimkan. Model terbaik untuk kelompok bayi baru lahir adalah LR dengan skor R^2 sebesar 0,82, sedangkan untuk bukan bayi baru lahir model terbaik adalah *CatBoost Regression* dengan skor R^2 sebesar 0,43. SHAP juga mengungkapkan bahwa fitur ras, gender, dan etnis tidak signifikan dalam memprediksi lama rawat inap, yang mengindikasikan tidak adanya bias sistemik berbasis demografis.
10	(Rahman dkk., 2022)	*Hospital patients' length of stay prediction: A federated learning approach*	*Linear Regression, Lasso Regression,* dan *Ridge Regression*	Penelitian ini menggunakan dataset publik SPARCS 2015 yang dipisahkan menjadi data dari 10 rumah sakit berbeda. Model dilatih secara lokal di masing-masing rumah sakit. Lalu dilakukan *federated learning* dan data digabungkan kembali. Model *Linear Regression* memberikan hasil terbaik dengan MAE sebesar 1.389, RMSE sebesar 2.0320, dan R^2 sebesar 0.873.

2.2.2 Matriks Penelitian

Matriks penelitian memuat ringkasan terkait penelitian terdahulu yang memuat algoritma, implementasi XAI SHAP, dan dataset yang digunakan sebagai pembanding pada penelitian ini dimuat pada Tabel 2. 2

Tabel 2. 2 Matriks Penelitian

No	Nama Penulis (Tahun)	Algoritma	SHAP	Dataset Publik
1	(Peng & Gao, 2025)	CNN, XGBoost, GBM, dan RF	-	✓
2	(Mekhaldi dkk., 2021)	MLR, SVM, RF, GB	-	✓
3	(Balik, 2024)	FedSGD, FedAVGv1, FedAVGv2	-	✓
4	(Puspita dkk., 2023)	RF *Ensemble Learning,* SVM, NN	-	-
5	(Adawiyah dkk., 2021)	*Neural Network*	-	✓
6	(Mengistu dkk., 2026)	LR, DT, RF, GB, MLP, SVM, KNN, NB	-	-
7	(Alsinglawi dkk., 2024)	XGBoost, RF, GB, LR, dan MLP	✓	-
8	(Fa dkk., 2025)	LR, SVM, MLP, dan XGBoost	✓	-
9	(Jain dkk., 2024)	LR, RF, CatBoost	✓	✓
10	(Rahman dkk., 2022)	LR*, Lasso Regression*, dan *Ridge Regression*	-	✓

No	Nama Penulis (Tahun)	Algoritma	SHAP	Dataset Publik
11	Penelitian ini	FT-*Transformer*	✓	✓

2.2.3 Penelitian Terdekat

Sejumlah penelitian telah membuktikan efektivitas berbagai pendekatan *machine learning* dalam memprediksi lama rawat inap pasien. Peng & Gao, (2025) menunjukkan bahwa model *ensemble* berbasis pohon keputusan, khususnya XGBoost dengan 40 fitur, mampu menghasilkan prediksi terbaik pada dataset Microsoft LoS dengan MAE 0,2974, RMSE 0,3881, RAE 0,1553, dan RSE 0,0270. Temuan serupa diperoleh Mekhaldi dkk., (2021) yang juga menggunakan dataset Microsoft LoS, di mana *Gradient Boosting* mengungguli model lain termasuk MLR, SVM, dan *Random Forest*, meski penerapan teknik SMOTER untuk menangani ketidakseimbangan data hanya memberikan peningkatan marginal pada akurasi. Dalam konteks privasi data, Balik, (2024) mengeksplorasi pendekatan *federated learning* menggunakan dataset yang sama dan menemukan bahwa FedSGD menghasilkan MSE lebih rendah dibanding dua varian FedAVG, meski secara umum performa *federated learning* masih berada di bawah model terpusat konvensional.

Selain penggunaan dataset Microsoft LoS, beberapa penelitian memanfaatkan data klinis dari berbagai konteks. Puspita dkk., (2023) menerapkan *ensemble Random Forest* pada data rumah sakit riil dan memperoleh F1-*score* 0,87, sedangkan Adawiyah dkk., (2021) menggunakan *Neural Network* yang dioptimasi dengan *Grid Search* pada dataset publik Kaggle dan berhasil meningkatkan akurasi dari 89,22% menjadi 92,20%. Mengistu dkk., (2026) menghadapi tantangan ketidakseimbangan kelas pada data dari rumah sakit Ethiopia dan menemukan

bahwa *Gradient Boosting* dengan SMOTE memberikan AUC terbaik sebesar 0,70 di antara delapan model yang dibandingkan.

Beberapa penelitian juga mulai mengintegrasikan interpretabilitas model ke dalam proses prediksi. Alsinglawi dkk., (2024) menggabungkan XGBoost dengan SHAP dan *ExplainerDashboard* pada data *Electronic Health Record* dari Uni Emirat Arab dan memperoleh ROC-AUC hingga 98% pada tahap klasifikasi. Fa dkk., (2025) menerapkan seleksi variabel berbasis SHAP pada data dua hari pertama rawat inap dari rumah sakit universitas di Prancis dan menemukan bahwa penyederhanaan fitur berbasis SHAP tidak menurunkan, bahkan sedikit meningkatkan, performa model XGBoost. Sementara itu, Jain dkk., (2024) menggunakan SHAP pada dataset SPARCS 2016 berisi lebih dari 2,3 juta catatan dan mengungkap bahwa fitur demografis seperti ras, gender, dan etnis tidak memiliki pengaruh signifikan terhadap prediksi LoS. Rahman dkk., (2022) melengkapi lanskap ini dengan pendekatan *federated learning* berbasis regresi linier pada dataset SPARCS 2015 yang dipartisi ke sepuluh rumah sakit dan memperoleh MAE 1,389 serta R^2 0,873.

Berdasarkan kajian terhadap penelitian-penelitian yang dirangkum dalam Tabel 2. 1 dan Tabel 2. 2, dapat disimpulkan bahwa eksplorasi metode prediksi LoS telah mencakup rentang yang luas, mulai dari *machine learning* konvensional seperti XGBoost, *Random Forest*, dan *Linear Regression*, hingga pendekatan *federated learning* dan *deep learning*. Namun, penerapan *deep learning* pada data tabular masih terbatas dan sebagian besar berfokus pada arsitektur *Convolutional Neural Network* dan *Multi-Layer Perceptron*. Di sisi interpretabilitas, penerapan

XAI dengan metode SHAP sudah mulai diadopsi dalam beberapa penelitian, tetapi penelitian yang secara spesifik menggunakan dataset Microsoft LoS belum mengimplementasikannya, pendekatan interpretasi yang digunakan masih berupa mutual information, seperti pada Peng & Gao (2025). Temuan mengenai adanya kesenjangan pada penelitian terdahulu menjadi dasar pengembangan penelitian ini. Sebagai alternatif penyelesaian, penelitian ini mengusulkan penggunaan FT-*Transformer*, sebuah arsitektur *Transformer* yang dirancang untuk memodelkan data tabular secara lebih efektif, yang diterapkan pada dataset Microsoft LoS. Berbeda dari pendekatan sebelumnya, penelitian ini mengintegrasikan SHAP secara langsung ke dalam *pipeline* sebagai mekanisme seleksi fitur berbasis kontribusi prediktif sekaligus sebagai alat interpretabilitas *post-hoc*, sehingga hasil prediksi tidak bersifat *black-box* dan diharapkan dapat dipahami oleh tenaga ahli kesehatan dalam konteks pengambilan keputusan klinis.