PENINGKATAN AKURASI MODEL BOOSTING PADA PREDIKSI KESEHATAN TIDUR MENGGUNAKAN OPTUNA

Muhammad Itqan Mazdadi[1], Triando Hamonangan Saragih[2], Irwan Budiman[3], Muhammad Naufal Anshory[4]

[1,2,3,4] Ilmu Komputer, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Lambung Mangkurat, Indonesia
[1]mazdadi@ulm.ac.id, [2]triando.saragih@ulm.ac.id, [3]irwan.budiman@ulm.ac.id, [4]muhammad.naufal210@gmail.com

Abstrak

Kualitas tidur memiliki peran penting dalam menjaga kesehatan fisik maupun mental, sementara gangguan tidur dapat meningkatkan risiko berbagai penyakit kronis. Perkembangan *machine learning* membuka peluang untuk melakukan prediksi kesehatan tidur secara lebih akurat melalui pemanfaatan data gaya hidup. Penelitian ini berfokus pada penerapan algoritma *boosting*, yaitu XGBoost, LightGBM, AdaBoost, dan *GradientBoosting*, dengan dukungan teknik *hyperparameter tuning* berbasis Optuna untuk meningkatkan akurasi prediksi. *Dataset* yang digunakan adalah *Sleep Health and Lifestyle Dataset* yang memuat variabel demografis, kebiasaan hidup, serta kondisi tidur. Tahapan penelitian meliputi praproses data, pembagian data latih dan uji, pelatihan model, optimasi *hyperparameter* menggunakan Optuna dengan metode *Tree-structured Parzen Estimator* (TPE), serta evaluasi model menggunakan metrik akurasi. Hasil eksperimen menunjukkan bahwa tuning dengan Optuna memberikan peningkatan akurasi pada beberapa model, khususnya LightGBM dan AdaBoost, dengan nilai akurasi mencapai 93,3% dan 90,7%. Sementara itu, XGBoost dan *GradientBoosting* menunjukkan performa stabil dengan akurasi tetap tinggi baik sebelum maupun sesudah tuning. Temuan ini menegaskan bahwa efektivitas tuning bergantung pada karakteristik algoritma yang digunakan. Secara keseluruhan, penelitian ini membuktikan bahwa Optuna dapat menjadi solusi efektif dalam meningkatkan kinerja model boosting untuk prediksi kesehatan tidur. Sebagai arah penelitian lanjutan, disarankan penggunaan metrik evaluasi yang lebih beragam, penerapan teknik penyeimbangan data, serta eksplorasi integrasi dengan metode *deep learning* untuk memperkaya hasil analisis.

Kata kunci: *boosting*, optuna, *hyperparameter tuning*, kesehatan tidur, *machine learning*

1. Pendahuluan

Kualitas tidur memiliki peran yang krusial dalam menjaga kesehatan fisik maupun mental. Berbagai studi telah menunjukkan bahwa gangguan tidur dapat meningkatkan risiko penyakit kronis seperti obesitas dan penyakit jantung, serta menurunkan produktivitas seseorang (Medic et al., 2017). Seiring berkembangnya pendekatan data science dan *machine learning*, muncul peluang untuk memprediksi status kesehatan tidur secara lebih dini melalui analisis data gaya hidup. Salah satu *dataset* yang sering digunakan untuk kajian tersebut adalah *Sleep Health and Lifestyle Dataset*, yang memuat variabel terkait faktor demografis dan kebiasaan hidup, seperti usia, jenis kelamin, pekerjaan, konsumsi kafein, kebiasaan merokok, hingga durasi tidur (Tharmalingam, 2021).

Metode klasifikasi berbasis *ensemble learning*, khususnya *boosting*, telah banyak digunakan dalam penelitian prediksi kesehatan. Algoritma seperti AdaBoost, *GradientBoosting*, XGBoost, dan LightGBM terbukti mampu menghasilkan prediksi dengan tingkat akurasi tinggi karena kemampuannya menggabungkan model lemah menjadi prediktor yang lebih kuat (Chen & Guestrin, 2016; Ke et al., 2017; Schapire et al., 1999). Namun demikian, kinerja optimal algoritma-algoritma tersebut sangat bergantung pada pemilihan *hyperparameter* yang tepat, yang sering kali menantang untuk ditentukan secara manual.

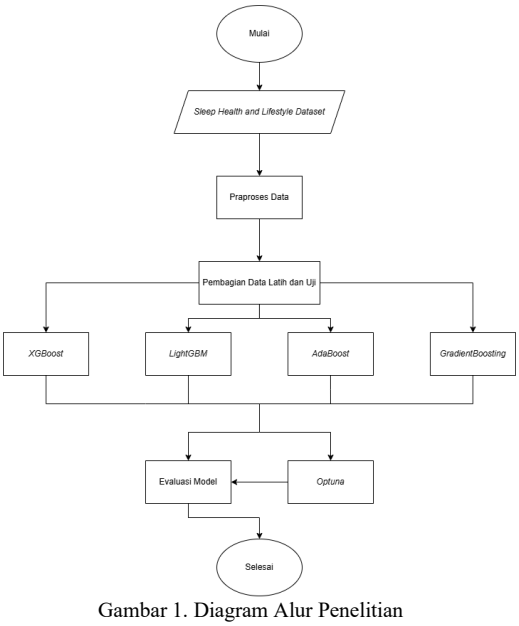
Dalam beberapa tahun terakhir, *hyperparameter tuning* berbasis optimasi otomatis mulai banyak digunakan, salah satunya melalui *framework* Optuna. Optuna memanfaatkan pendekatan *Bayesian optimization* dan *Tree-structured Parzen Estimator (TPE)* untuk mencari kombinasi *hyperparameter* terbaik secara efisien (Akiba et al., 2019). Dengan adanya metode ini, diharapkan akurasi model *boosting* dalam memprediksi kesehatan tidur dapat ditingkatkan secara signifikan.

Sejumlah penelitian sebelumnya telah mengkaji penerapan algoritma *boosting* di bidang kesehatan. Misalnya, (Das et al., 2025) mengimplementasikan XGBoost untuk prediksi risiko diabetes dengan hasil akurasi yang baik, sedangkan (Jain & Ganesan, 2024) menerapkan LightGBM untuk klasifikasi berbagai jenis gangguan tidur dengan memanfaatkan *dataset* CAP dan berhasil mencapai akurasi hingga 94,4%. Hasil-hasil tersebut menunjukkan potensi kuat metode *boosting* dalam meningkatkan performa model prediktif di bidang medis.

Namun demikian, sebagian besar penelitian terdahulu masih berfokus pada penerapan satu jenis algoritma *boosting* tanpa melakukan perbandingan sistematis antar model. Selain itu, penerapan teknik optimasi otomatis seperti Optuna dalam proses penyetelan *hyperparameter* pada konteks klasifikasi gangguan tidur masih jarang dilakukan. Celah penelitian ini menunjukkan perlunya kajian yang tidak hanya membandingkan beberapa algoritma *boosting* sekaligus, tetapi juga mengintegrasikan metode optimasi *hyperparameter* untuk memperoleh kinerja prediksi yang lebih optimal dan komprehensif.

Berdasarkan hal tersebut, penelitian ini diarahkan untuk mengevaluasi performa empat algoritma *boosting*, yaitu XGBoost, LightGBM, AdaBoost, dan *GradientBoosting*, dalam memprediksi status kesehatan tidur menggunakan *Sleep Health and Lifestyle Dataset*. Selanjutnya, dilakukan optimasi *hyperparameter* dengan Optuna untuk menguji sejauh mana peningkatan akurasi yang dapat dicapai dari masing-masing model. Melalui pendekatan kuantitatif berbasis eksperimen dengan evaluasi menggunakan metrik klasifikasi berupa akurasi, presisi, *recall*, dan *F1-score*, penelitian ini berupaya memberikan kontribusi berupa perbandingan menyeluruh mengenai efektivitas Optuna dalam meningkatkan hasil prediksi algoritma *boosting* pada analisis kesehatan tidur.

2. Metode



Gambar 1. Diagram Alur Penelitian

Gambar 1 merupakan alur metode penelitian yang diterapkan dalam studi ini, dimulai dari tahap pengumpulan data hingga proses evaluasi model. Setiap tahap dirancang secara sistematis untuk menghasilkan model prediksi kesehatan tidur yang optimal. Alur tersebut meliputi beberapa langkah utama, yaitu: pengumpulan data, pemrosesan data, pembagian data, pelatihan model menggunakan algoritma XGBoost, LightGBM, AdaBoost dan *GradientBoosting*, optimasi *hyperparameter* dengan Optuna, serta tahap evaluasi model dengan metrik kinerja yang relevan.

2.1 Pengumpulan Data

Dataset yang digunakan dalam penelitian ini adalah *Sleep Health and Lifestyle Dataset*, yang diperoleh dari Kaggle melalui tautan: https://www.kaggle.com/datasets/uom190346a/sleep-health-and-lifestyle-dataset. *Dataset* ini terdiri dari 374 baris dan 13 kolom yang mencakup variabel terkait kebiasaan tidur dan gaya hidup, seperti usia, jenis kelamin, durasi tidur, kualitas tidur, tingkat aktivitas fisik, tingkat stres, kategori BMI, tekanan darah, detak jantung, jumlah langkah harian, serta keberadaan gangguan tidur (none, insomnia, sleep apnea). Penjelasan lengkap mengenai setiap fitur yang terdapat dalam *dataset* dapat ditemukan pada Tabel 1.

Tabel 1. Atribut-Atribut yang terdapat pada *Dataset Sleep Health and Lifestyle*

Fitur	Deskripsi
Personal ID	Kode unik yang diberikan untuk mengidentifikasi setiap individu.
Gender	Jenis kelamin biologis dari individu (Laki-laki atau Perempuan).
Age	Usia individu yang dinyatakan dalam satuan tahun.
Occupation	Jenis pekerjaan atau peran profesional yang dijalankan oleh individu.
Sleep Duration	Jumlah total waktu tidur individu setiap hari, diukur dalam satuan jam.
Quality of Sleep	Penilaian subjektif individu terhadap kualitas tidurnya, menggunakan skala dari 1 hingga 10.
Physical Activity Level	Durasi harian dalam menit yang digunakan individu untuk melakukan aktivitas fisik.
Stress Level	Tingkat stres yang dirasakan individu berdasarkan penilaian pribadi, dengan skala 1 sampai 10.
BMI Category	Kategori Indeks Massa Tubuh (IMT) individu seperti Kurus, Normal, atau Kelebihan Berat Badan.
Blood Pressure	Tekanan darah individu yang ditampilkan sebagai tekanan sistolik dibagi tekanan diastolik.
Heart Rate	Jumlah detak jantung per menit saat individu dalam kondisi istirahat.
Daily Steps	Jumlah langkah yang ditempuh individu dalam satu hari.
Sleep Disorder	Menunjukkan apakah individu mengalami gangguan tidur, dan jika iya, jenisnya (Tidak Ada, Insomnia, Apnea Tidur).

2.2 Pemrosesan Data

Sebelum model dikembangkan, dilakukan beberapa tahap pemrosesan data sebagai berikut:

1) Data Cleaning

Data cleaning, yang juga dikenal sebagai *data* cleansing atau *data scrubbing*, merupakan proses

evaluasi kualitas data melalui modifikasi, perubahan, atau penghapusan terhadap data yang dianggap tidak relevan, tidak lengkap, tidak akurat, atau memiliki format yang tidak sesuai dalam basis data, dengan tujuan untuk menghasilkan data yang berkualitas tinggi (Darwis et al., 2021) Pada penelitian ini, proses *data cleaning* dilakukan dengan menghapus atribut yang tidak relevan untuk proses klasifikasi, seperti *Person ID* dan *Gender*.

2) Encoding Data Kategorikal

Proses mengubah data kategorikal menjadi format numerik dikenal sebagai *Label Encoding*. Hal ini memungkinkan algoritme pembelajaran mesin untuk memproses data (Low et al., 2022). Dalam penelitian ini, *Label Encoding* digunakan untuk mengonversi atribut kategorikal seperti *BMI Category*, *Blood Pressure*, *Sleep Disorder*, dan *Occupation* ke dalam bentuk numerik.

3) Feature Scaling

Feature scaling adalah langkah prapemrosesan penting yang menstandarkan rentang data melalui teknik seperti normalisasi dan standardisasi (Ahmed Ouameur et al., 2020). Dalam penelitian ini, metode yang digunakan untuk penskalaan fitur adalah *MinMaxScaler*. *MinMaxScaler* adalah teknik penskalaan yang mengubah nilai fitur ke dalam rentang tertentu, biasanya antara 0 dan 1(de Amorim et al., 2023). Rumus *MinmaxScaler* bisa dilihat pada Persamaan (1) (Deepa & Ramesh, 2022).

$$X_{std} = \frac{(X - X.min)}{(X.max - X.)} \quad (1)$$

Pada Persamaan (1), menyatakan nilai asli suatu fitur, dan masing-masing merupakan nilai minimum dan maksimum dari fitur tersebut pada *dataset*. Hasil normalisasi merepresentasikan nilai fitur yang telah dipetakan ke dalam rentang [0,1], sehingga seluruh fitur memiliki skala yang sebanding dan tidak mendominasi proses pembelajaran model.

2.3 Pembagian Data

Membagi data menjadi set pelatihan, pengujian, dan validasi merupakan prosedur standar dalam pembelajaran mesin yang berfungsi untuk mendukung proses pembangunan model serta menilai kinerjanya (Birba, 2020). Dalam penelitian ini digunakan metode *train-test split*, yaitu teknik yang membagi data secara acak ke dalam subset pelatihan dan pengujian untuk memastikan evaluasi model yang objektif serta representatif (Scikit-learn Developers, n.d.). Pembagian dilakukan menggunakan fungsi *train_test_split* dari pustaka scikit-learn dengan parameter *test_size=0.2*, yang berarti 20% data digunakan sebagai data uji dan 80% sisanya sebagai data latih. Rasio 80:20 dipilih karena sederhana, banyak digunakan dalam eksperimen *machine learning*, dan sesuai untuk kondisi jumlah data yang relatif terbatas. Dengan proporsi ini, diperoleh keseimbangan antara ketersediaan data yang cukup untuk proses pelatihan dan data yang memadai untuk evaluasi secara andal. Selain itu, rasio ini telah umum diterapkan dalam berbagai penelitian karena menawarkan kompromi yang baik antara efisiensi komputasi dan keakuratan evaluasi.

2.4 XGBoost

Extreme GradientBoosting (XGBoost) merupakan salah satu implementasi dari algoritma *GradientBoosting* yang dirancang untuk mencapai kecepatan komputasi tinggi serta akurasi prediksi yang optimal, dan telah banyak digunakan baik untuk tugas regresi maupun klasifikasi (Safavi et al., 2024). XGBoost membangun serangkaian pohon keputusan secara iteratif, di mana setiap pohon baru dilatih untuk memperbaiki kesalahan prediksi (residual error) dari pohon sebelumnya. Hasil prediksi akhir diperoleh dari penjumlahan kontribusi seluruh pohon yang dibangun dalam proses pelatihan.

Fungsi objektif yang diminimalkan pada XGBoost dirumuskan sebagai berikut:

$$L(y, \hat{y}) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \Omega(\hat{f}) \quad (2)$$

Dengan y adalah label sebenarnya, $\hat{y}$ merupakan label hasil prediksi, l adalah fungsi kerugian (loss function), dan $\Omega(\hat{f})$ adalah istilah regularisasi yang digunakan untuk mengontrol kompleksitas model. Regularisasi tersebut didefinisikan sebagai:

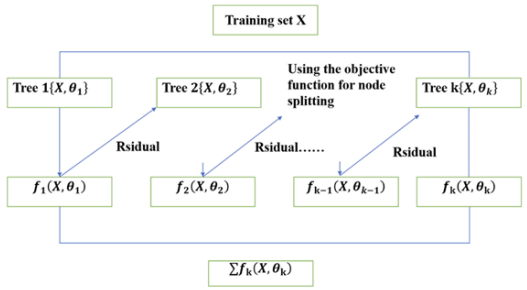
$$\Omega(\hat{f}) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{n} \mathrm{w}_j^2 \quad (3)$$

dengan T adalah jumlah daun (leaves) dalam pohon, m adalah jumlah fitur, γ merupakan parameter regularisasi terhadap jumlah daun, λ adalah parameter regularisasi L2, dan w_j adalah bobot dari daun ke-j (Safavi et al., 2024).

Pada Persamaan (2), indeks menunjukkan data ke-i dalam *dataset*, adalah nilai aktual kelas gangguan tidur, dan adalah nilai prediksi model pada iterasi tertentu. Fungsi *loss* mengukur kesalahan prediksi, sedangkan istilah regularisasi pada Persamaan (3) mengontrol kompleksitas model melalui jumlah daun pohon dan bobot daun. Parameter dan berfungsi untuk mencegah *overfitting* dengan memberikan penalti pada struktur pohon yang terlalu kompleks.

Struktur kerja XGBoost secara umum dapat dilihat pada Gambar 2, yang menunjukkan proses pembentukan model melalui serangkaian pohon keputusan. Skema proses pelatihan XGBoost dimulai dari *training set*, kemudian membangun pohon pertama berdasarkan data awal. Selanjutnya, residual dari pohon pertama digunakan untuk melatih pohon berikutnya, dan proses ini berulang hingga terbentuk

sejumlah pohon. Setiap pohon baru dihasilkan dengan memanfaatkan fungsi objektif untuk pemisahan node yang optimal. Hasil akhir merupakan penjumlahan dari seluruh kontribusi pohon, sehingga model mampu menghasilkan prediksi dengan performa yang lebih baik (Su et al., 2024).



Gambar 2. Skema Prinsip Kerja XGBoost (Su et al., 2024)

2.5 LightGBM

Light Gradient-Boosting Machine (LightGBM) merupakan salah satu algoritma *boosting* berbasis pohon keputusan yang dirancang untuk memberikan kinerja tinggi pada dataset berukuran besar. Algoritma ini membangun model pohon secara *leaf-wise* dengan memilih pemisahan berdasarkan nilai *gain* terbesar terlebih dahulu, berbeda dengan metode *level-wise* yang umum digunakan. Strategi ini membuat LightGBM mampu mencapai akurasi yang lebih tinggi dengan tetap mempertahankan efisiensi komputasi (Daniel, 2024).

Sebagai model *ensemble*, LightGBM menggabungkan sejumlah pohon regresi lemah menjadi pohon regresi kuat, yang secara matematis dapat direpresentasikan sebagai:

$$F(x) = \sum_{k=1}^{K} f_k(x) \qquad (4)$$

$F(x)$ merupakan nilai prediksi akhir untuk suatu sampel x, sedangkan $f_k(x)$ merepresentasikan fungsi regresi dari pohon keputusan ke-k dalam *ensemble* LightGBM. Setiap pohon memodelkan sebagian kecil pola dalam data, dan keseluruhan prediksi diperoleh dari penjumlahan kontribusi seluruh pohon, sehingga LightGBM mampu menangkap hubungan *nonlinier* dalam data kesehatan tidur. (Mao et al., 2024).

Untuk meningkatkan efisiensi dan mengurangi kompleksitas, LightGBM menerapkan beberapa teknik optimasi, di antaranya *Gradient-based One Side Sampling* (GOSS) yang mengabaikan sebagian besar sampel dengan gradien kecil dan hanya menggunakan sampel dengan gradien besar dalam perhitungan *information gain*, serta *Exclusive Feature Bundling* (EFB) yang menggabungkan fitur-fitur saling eksklusif sehingga dimensi data dapat diperkecil tanpa kehilangan informasi penting (Mao et al., 2024). Selain itu, LightGBM mendukung penggunaan fitur kategorikal secara langsung tanpa memerlukan proses *one-hot encoding*, serta mampu melakukan pelatihan paralel dan terdistribusi, menjadikannya algoritma yang sangat efisien dan skalabel (Daniel, 2024).

2.6 Adaboost

AdaBoost (Adaptive Boosting), yang diperkenalkan oleh Freund dan Schapire, merupakan algoritma *ensemble* populer yang bekerja dengan melatih sejumlah pengklasifikasi lemah secara berulang pada data berbobot. Hasil dari setiap pengklasifikasi tersebut kemudian digabungkan menggunakan skema penjumlahan berbobot, sehingga membentuk model akhir yang lebih tangguh (Wang & Sun, 2021).

Algoritma AdaBoost memiliki keunggulan dalam mengombinasikan sejumlah pengklasifikasi lemah dengan memberikan bobot lebih besar pada pengklasifikasi yang menunjukkan performa lebih baik, sehingga mampu meningkatkan akurasi model secara keseluruhan. Walaupun pada mulanya tidak ditujukan untuk menangani data yang tidak seimbang, kemampuannya dalam memberikan perhatian lebih pada kelas yang kurang terwakili menjadikannya metode yang potensial untuk digunakan dalam kondisi tersebut (de Giorgio et al., 2023). *Pseudocode* dari Algoritma AdaBoost dapat dilihat pada Algoritma 1.

Given: $(x_1, y_1), \dots, (x_m, y_m)$ where $\mathrm{x_i} \in X, y_i \in \{-1, +1\}$

Initialize: $D_1(i) = \frac{1}{m}$ for $i = 1, \dots, \mathrm{m}$

For $t = 1, \dots, T$:

- Train weak learner using distribution D_t
- Get weak hypothesis $h_t: X \to \{-1, +1\}$
- Aim: Select h_t to minimalize the weighted erorr:
$$\epsilon_t = \boldsymbol{Pr}_{i \sim D_t}[h_t(x_i) \neq y_i]$$
- Choose:
$$\alpha_t = \frac{1}{2}\log\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$$
- Update, for $1 = 1, \dots, \mathrm{m}$:
$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} e^{\alpha_t} & \text{if } \mathrm{h_t(x_i = y_i)} \\ e^{\alpha_t} & \text{if } \mathrm{h_t(x_i \neq y_i)} \end{cases} = \frac{D_t(i)\exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

Where Z_t is a normalization factor (choosen so that D_{t+1} will be a distribution).

Output the final hypothesis:

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

Algoritma 1. Ilustrasi *Pseudocode* Algoritma Adaboost

2.7 GradientBoosting

GradientBoosting (GB) merupakan salah satu algoritma *boosting* yang digunakan dalam penelitian ini untuk meningkatkan akurasi prediksi pada domain kesehatan tidur. GB bekerja dengan membangun pohon keputusan secara bertahap, di mana setiap pohon baru dilatih untuk memperbaiki kesalahan

prediksi (residual) dari pohon sebelumnya. Proses iteratif ini memungkinkan model untuk terus disempurnakan sehingga kesalahan prediksi dapat diminimalkan, dan akurasi hasil klasifikasi meningkat (Ouadi et al., 2024).

Dalam penerapannya, GB menggunakan fungsi kerugian yang dapat diturunkan secara matematis untuk mengukur tingkat kesalahan prediksi, serta memanfaatkan teknik penurunan gradien (gradient descent) dalam memperbarui model secara berurutan. Algoritma ini juga dilengkapi dengan mekanisme regularisasi, seperti pengaturan laju pembelajaran (learning rate) dan kedalaman pohon, guna mengendalikan kompleksitas model serta mencegah terjadinya *overfitting*. Selain itu, proses pelatihan GB secara alami menyoroti fitur-fitur yang paling signifikan dalam memengaruhi prediksi, sehingga dapat memberikan wawasan tambahan terkait pola dalam data (Wu et al., 2024). *Pseudocode* dari Algoritma *GradientBoosting* dapat dilihat pada Algoritma 2.

GradientBoosting	
Input:	Data Latih: $= (x_i, y_i)_{i=1}^{n}$ Dimana, x_i adalah titik data dan y_i adalah label untuk x_i Fungsi Loss: $= L\ (y_i, f(x))$
Output:	Fungsi pohon keputusan $f(x)$
Langkah 1:	Inisialisasi model $f(x)$: $=$ argmin$\sum_{i=1}^{n} L(y_i, z)$.
Langkah 2:	$\boldsymbol{for}\ k\ =\ 1,2,3\ \ldots, K\ \boldsymbol{do}$
Langkah 3:	$\boldsymbol{for}\ I\ =\ 1,2,3\ \ldots, K\ \boldsymbol{do}$
Langkah 4:	Hitung pseudo residual error:: $R_{ki} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(y_i)}\right]_{f(x)=f_{k-1}(x)}$
Langkah 5:	**Selesai**
Langkah 6:	**Selesai**
Langkah 7:	Bangun pohon keputusan baru $T_k(x; \theta_k)$, berdasarkan pada $R_{ki}, \theta_k = \{R_{kj} j = [1,2,3\ \ldots J]\}$
Langkah 8:	$\boldsymbol{for}\ k\ =\ 1,2,3\ \ldots, K\ \boldsymbol{do}$
Langkah 9:	$z_{kj} = argmin \sum_{x_i \in R_{ki}}^{n} L(y_i, f_{k-1}(x) + z)$
Langkah 10:	**Selesai**
Langkah 11:	Perbarui model $f_k(x) = f_{k-1}(x) + \sum_{j=1}^{j} z_{kj} I(x \hat{I} R_{kj})$
Langkah 12:	$f(x) = \sum_{k=1}^{K} \sum_{j=1}^{J} z_{kj} I(x \hat{I} R_{kj})$

Algoritma 2. Ilustrasi *Pseudocode* Algoritma *GradientBoosting*

Algoritma 2 merangkum langkah-langkah utama dari algoritma *GradientBoosting*, yaitu inisialisasi model awal dengan fungsi *loss*, perhitungan *pseudo-residual* pada setiap iterasi, pembangunan pohon keputusan berdasarkan residual tersebut, serta pembaruan model dengan menggabungkan hasil pohon baru menggunakan learning rate. Proses ini diulang hingga jumlah iterasi yang telah ditentukan tercapai, sehingga menghasilkan model akhir dengan performa yang lebih baik.

2.8 Optuna

Dalam penelitian ini, proses *hyperparameter tuning* dilakukan untuk mengevaluasi tingkat akurasi model dalam menghasilkan prediksi. Tahap ini merupakan bagian penting dari pelatihan menggunakan data *training* dan *testing* guna memperoleh performa yang optimal (Dinathi et al., 2024). Sayangnya, masih banyak yang melakukan pencarian kombinasi *hyperparameter* secara manual saat melatih dan menguji model, sehingga menjadikan prosesnya berulang-ulang dan dapat memakan waktu berjam-jam hingga berhari-hari. Pendekatan semacam ini kurang efisien untuk optimisasi *hyperparameter* karena hanya menguras waktu dan sumber daya. Pada titik inilah Optuna hadir sebagai solusi yang efektif (Srinivas & Katarya, 2022).

Optuna merupakan metode optimisasi yang memberikan tiga keunggulan utama dalam proses pemilihan model maupun penyesuaian *hyperparameter*. Pertama, adanya API *define-by-run* yang menawarkan fleksibilitas tinggi dalam melakukan konfigurasi. Kedua, Optuna dilengkapi dengan mekanisme *pruning* dan *sampling* yang efisien, sehingga mempercepat proses pencarian parameter terbaik. Ketiga, kemudahan dalam pengaturan membuat Optuna menjadi pilihan praktis bagi para pengguna. Konsep API *define-by-run* ini sendiri terinspirasi dari kerangka kerja pembelajaran mendalam, yang memungkinkan pengguna mendefinisikan ruang pencarian *hyperparameter* secara dinamis dan lebih adaptif (Lai et al., 2023).

Dalam penelitian ini, Optuna tidak hanya digunakan untuk mengoptimalkan *hyperparameter*, tetapi juga membantu meminimalkan risiko *overfitting* dengan secara sistematis mengeksplorasi kombinasi terbaik antara *n_estimators* dan *learning_rate* pada Algoritma. Penyetelan *hyperparameter* yang tepat menjadi krusial untuk mencegah terbentuknya model yang terlalu kompleks, yang meskipun mampu menunjukkan performa sangat baik pada data pelatihan, namun cenderung gagal ketika dihadapkan pada data baru yang belum pernah ditemui sebelumnya (Akiba et al., 2019).

2.9 Evaluasi Model

Evaluasi model pada klasifikasi bertujuan untuk menilai sejauh mana algoritma dapat membedakan kelas-kelas target dengan benar. Metrik evaluasi yang sering digunakan pada penelitian klasifikasi, seperti akurasi dapat memberikan gambaran yang jelas tentang kinerja model untuk menangani data klasifikasi (Farhadi et al., 2024; Krzywicka & Wosiak, 2023; Shirdel et al., 2024).

Akurasi adalah metrik yang sering digunakan untuk klasifikasi biner guna mengukur proporsi prediksi yang benar terhadap total jumlah data. Metrik ini sangat cocok untuk dataset yang seimbang (Shirdel et al., 2024). Namun, pada *dataset* yang tidak seimbang, akurasi dapat memberikan hasil yang kurang tepat karena model mungkin cenderung

memprediksi kelas mayoritas saja untuk mencapai skor akurasi yang tinggi (Krzywicka & Wosiak, 2023). Formula akurasi dapat dilihat pada persamaan 5 berikut.

$$akurasi = \frac{TP + TN}{TP + TN + FN + FN} \quad (5)$$

Nilai TP (True Positive) menunjukkan jumlah data gangguan tidur yang diprediksi dengan benar, TN (True Negative) menyatakan data tanpa gangguan tidur yang diklasifikasikan dengan benar, FP (False Positive) menunjukkan kesalahan prediksi positif, dan FN (False Negative) menunjukkan kesalahan prediksi negatif. Dengan demikian, akurasi merepresentasikan proporsi total prediksi yang sesuai dengan kondisi sebenarnya (Farhadi et al., 2024).

3. Hasil dan Pembahasan

Bagian ini menyajikan hasil eksperimen dari penelitian, berupa hasil tuning menggunakan *Hyperparameter* Optuna, serta menampilkan evaluasi kinerja model berdasarkan metrik akurasi pada empat algoritma: XGBoost, LightGBM, AdaBoost dan *GradientBoosting* dengan dua skenario yaitu sebelum dan sesudah tuning.

3.1 Hasil *Hyperparameter Tuning*

Sebelum dilakukan proses penyetelan hiperparameter (hyperparameter tuning), setiap model dievaluasi menggunakan parameter default yang telah ditentukan. Evaluasi awal ini memberikan gambaran mengenai performa dasar dari masing-masing algoritma, yaitu XGBoost, LightGBM, AdaBoost, dan *GradientBoosting*. Hasil evaluasi menunjukkan variasi nilai akurasi antar model, yang menjadi dasar untuk dilakukan peningkatan performa melalui tuning.

Proses hyperparameter tuning dilakukan dengan memanfaatkan Optuna menggunakan metode *Tree-structured Parzen Estimator* (TPE) sebagai algoritma optimisasi. Pemilihan TPE bertujuan untuk mengeksplorasi ruang pencarian parameter secara lebih efisien dan sistematis, sehingga dapat menemukan kombinasi parameter terbaik. Adapun parameter yang disesuaikan pada setiap model adalah sebagai berikut:

- XGBoost: jumlah estimator (n_estimators), laju pembelajaran (learning_rate), dan kedalaman maksimum pohon (max_depth).
- LightGBM: jumlah estimator (n_estimators), laju pembelajaran (learning_rate), dan kedalaman maksimum pohon (max_depth).
- AdaBoost: jumlah estimator (n_estimators).
- *GradientBoosting*: jumlah estimator (n_estimators) dan laju pembelajaran (learning_rate).

Hasil tuning parameter terbaik untuk setiap model dapat dilihat pada Tabel 2.

Tabel 2. Hasil Hyperparameter Tuning pada Setiap Model

No	Model	Parameter Terbaik
1	XGBoost	*n_estimators* = 300 *learning_rate* = 0.0413 *max_depth* = 6
2	LightGBM	*n_estimators* = 200 *learning_rate* = 0.1169 *max_depth* = 5
3	AdaBoost	*n_estimators* = 100
4	*GradientBoosting*	*n_estimators* = 100 *learning_rate* = 0.0725

Berdasarkan Tabel 2, dapat dilihat bahwa setiap algoritma memiliki konfigurasi *hyperparameter* yang berbeda sesuai dengan karakteristik internal masing-masing model. Pada XGBoost, jumlah estimator optimal berada pada 300 dengan laju pembelajaran relatif kecil (0.0413), yang menandakan bahwa model membutuhkan iterasi lebih banyak dengan pembaruan bobot yang lebih hati-hati untuk mencapai performa optimal. LightGBM memperoleh hasil terbaik dengan jumlah estimator 200 dan laju pembelajaran 0.1169 serta kedalaman pohon maksimum 5, yang menunjukkan keseimbangan antara kompleksitas dan generalisasi. Untuk AdaBoost, jumlah estimator terbaik adalah 100, menegaskan bahwa penambahan estimator berlebih tidak selalu meningkatkan performa. Sementara itu, *GradientBoosting* mencapai hasil terbaik dengan jumlah estimator 100 dan laju pembelajaran 0.0725, menunjukkan bahwa kombinasi jumlah estimator moderat dengan laju pembelajaran menengah mampu menghasilkan model yang stabil.

3.2 Evaluasi Model

Model klasifikasi dievaluasi menggunakan metrik akurasi untuk membandingkan kinerja sebelum dan sesudah dilakukan *tuning hyperparameter* dengan Optuna. Hasil evaluasi ini disajikan dalam bentuk tabel, visualisasi diagram batang, serta *confusion matrix* agar dapat memberikan gambaran yang lebih jelas mengenai performa masing-masing model.

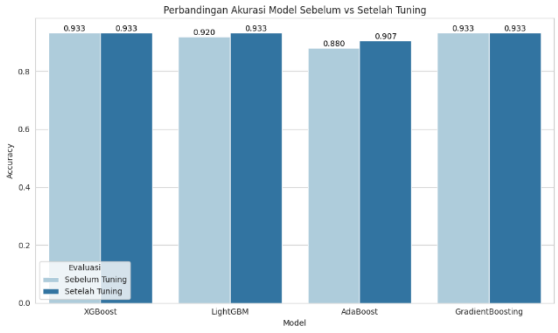
Tabel 3 berikut menyajikan nilai akurasi model XGBoost, LightGBM, AdaBoost, dan *GradientBoosting* pada dua skenario, yaitu sebelum dan sesudah tuning.

Tabel 3. Hasil Evaluasi Akurasi Model Sebelum dan Sesudah Tuning

Model	Sebelum Tuning	Sesudah Tuning
XGBoost	0.933	0.933
LightGBM	0.920	0.933
AdaBoost	0.880	0.907
GradientBoosting	0.933	0.933

Berdasarkan tabel tersebut, dapat dilihat bahwa terjadi peningkatan akurasi pada model LightGBM dan AdaBoost setelah dilakukan tuning hyperparameter. Model XGBoost dan *GradientBoosting* cenderung stabil dengan nilai akurasi yang sama sebelum maupun sesudah tuning.

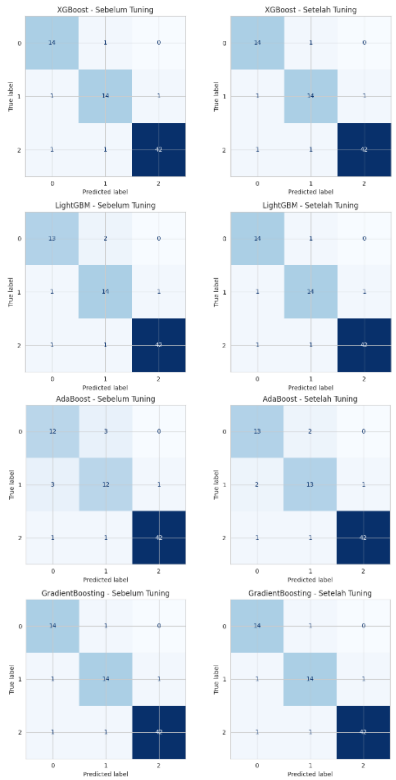
Selanjutnya, perbandingan akurasi model dalam bentuk visualisasi ditunjukkan pada Gambar 3 berikut.



Gambar 3. Diagram Batang Perbandingan Akurasi Model Sebelum dan Sesudah Tuning

Visualisasi pada Gambar 3 memperjelas perbedaan kinerja antar model. LightGBM dan AdaBoost terlihat mengalami kenaikan akurasi setelah tuning, sedangkan XGBoost dan *GradientBoosting* tetap konsisten pada tingkat akurasi yang tinggi.

Untuk analisis yang lebih mendalam, Gambar 4 berikut menampilkan *confusion matrix* masing-masing model sebelum dan sesudah tuning.



Gambar 4. *Confusion Matrix* Model Sebelum dan Sesudah Tuning

Berdasarkan Gambar 4, terlihat bahwa hasil prediksi model menunjukkan distribusi klasifikasi yang cukup baik, dengan sebagian besar data berhasil diklasifikasikan secara benar. Peningkatan kinerja terutama terlihat pada LightGBM dan AdaBoost setelah tuning, di mana jumlah prediksi benar meningkat dibandingkan sebelum tuning.

Secara keseluruhan, perbedaan respons model terhadap proses *hyperparameter tuning* dapat dijelaskan dari karakteristik internal masing-masing algoritma *boosting*. LightGBM dan AdaBoost menunjukkan peningkatan akurasi setelah *tuning* karena kedua algoritma ini memiliki sensitivitas tinggi terhadap pengaturan jumlah estimator dan laju pembelajaran. LightGBM menggunakan strategi pertumbuhan pohon secara *leaf-wise* yang berfokus pada *node* dengan *information gain* tertinggi. Oleh karena itu, pengaturan nilai *learning rate*, *max_depth*, dan jumlah *n_estimators* sangat memengaruhi kemampuan model dalam menyeimbangkan kompleksitas dan generalisasi. Optuna mampu menemukan kombinasi parameter yang lebih optimal sehingga *overfitting* dapat ditekan dan akurasi meningkat.

Sementara itu, AdaBoost bekerja dengan mekanisme pemberian bobot lebih besar pada data yang salah diklasifikasikan pada iterasi sebelumnya. Jumlah estimator yang tepat menentukan sejauh mana model dapat belajar dari kesalahan tanpa menjadi terlalu kompleks. Hasil *tuning* menunjukkan bahwa nilai *n_estimators* yang optimal mampu meningkatkan fokus AdaBoost terhadap pola data yang sulit, sehingga menghasilkan peningkatan akurasi.

Berbeda dengan LightGBM dan AdaBoost, XGBoost dan GradientBoosting menunjukkan stabilitas akurasi baik sebelum maupun sesudah *tuning*. Hal ini dapat dijelaskan oleh adanya mekanisme regularisasi internal pada XGBoost seperti penalti L1 dan L2 serta kontrol kompleksitas pohon yang membuat model relatif *robust* terhadap perubahan *hyperparameter*. Demikian pula, GradientBoosting pada *scikit-learn* memiliki *parameter default* yang cenderung konservatif, sehingga model sudah berada pada konfigurasi yang mendekati optimal. Oleh karena itu, ruang peningkatan melalui *tuning* menjadi lebih terbatas.

4. Kesimpulan

Penelitian ini berhasil menunjukkan bahwa penggunaan metode Optuna untuk *hyperparameter tuning* berkontribusi terhadap peningkatan kinerja sebagian model *boosting* dalam memprediksi kesehatan tidur. Berdasarkan hasil evaluasi, algoritma LightGBM dan AdaBoost mengalami peningkatan akurasi setelah proses tuning, masing-masing dari 0.920 menjadi 0.933 dan dari 0.880 menjadi 0.907. Sementara itu, *XGBoost* dan *GradientBoosting* menunjukkan performa yang stabil dengan akurasi tetap pada 0.933. Temuan ini mengindikasikan bahwa efektivitas Optuna dalam optimasi parameter bersifat model-dependent, di mana pengaruh tuning lebih terasa pada model dengan kompleksitas parameter yang lebih tinggi.

Dengan demikian, proses *hyperparameter tuning* dapat dipandang sebagai tahap penting untuk mencapai keseimbangan antara akurasi dan stabilitas model pada kasus prediksi kesehatan tidur.

Selain itu, penelitian ini membuka peluang untuk pengembangan lebih lanjut (*future works*) melalui penerapan metrik evaluasi tambahan seperti presisi, *recall*, *F1-score*, dan AUC-ROC guna memperoleh analisis kinerja yang lebih menyeluruh. Eksplorasi terhadap teknik *data balancing* maupun integrasi dengan pendekatan *deep learning* juga dapat dipertimbangkan untuk meningkatkan kemampuan generalisasi model. Dengan hasil yang diperoleh, penelitian ini memberikan kontribusi empiris terhadap penerapan *Optuna* sebagai metode optimasi parameter pada algoritma *boosting*, sekaligus menjadi dasar bagi pengembangan sistem prediksi kesehatan tidur yang lebih akurat dan adaptif di masa depan.

Daftar Pustaka:

Ahmed Ouameur, M., Caza-Szoka, M., & Massicotte, D. (2020). Machine learning enabled tools and methods for indoor localization using low power wireless network. Internet of Things (Netherlands), 12, 100300. https://doi.org/10.1016/j.iot.2020.100300

Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. (2019). Optuna: A Next-generation Hyperparameter Optimization Framework. KDD '19: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2623–2631. https://doi.org/10.1145/3292500.3330701

Birba, D. E. (2020). A Comparative study of data splitting algorithms for machine learning model selection. Degree Project in Computer Science and Engineering, 2020(1), 1–23. https://www.diva-portal.org/smash/get/diva2:1506870/FULLTEXT01.pdf

Chen, T., & Guestrin, C. (2016) XGBoost: A scalable tree boosting system. Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 13-17-Augu, 785–794. https://doi.org/10.1145/2939672.2939785

Daniel, C. (2024). A robust LightGBM model for concrete tensile strength forecast to aid in resilience-based structure strategies. Heliyon, 10(20), e39679. https://doi.org/10.1016/j.heliyon.2024.e39679

Darwis, D., Siskawati, N., & Abidin, Z. (2021). Penerapan Algoritma Naive Bayes Untuk Analisis Sentimen Review Data Twitter Bmkg Nasional. Jurnal Tekno Kompak, 15(1), 131. https://doi.org/10.33365/jtk.v15i1.744

Das, D., Aayushman, Kumar, S., Hussain, M. A., & Reddy, B. R. (2025). Diabetes Prediction using Ensemble Learning Techniques. Procedia Computer Science, 258, 3155–3164. https://doi.org/10.1016/j.procs.2025.04.573

de Amorim, L. B. V., Cavalcanti, G. D. C., & Cruz, R. M. O. (2023). The choice of scaling technique matters for classification performance. Applied Soft Computing, 133, 1–37. https://doi.org/10.1016/j.asoc.2022.109924

de Giorgio, A., Cola, G., & Wang, L. (2023). Systematic review of class imbalance problems in manufacturing. Journal of Manufacturing Systems, 71(September), 620–644. https://doi.org/10.1016/j.jmsy.2023.10.014

Deepa, B., & Ramesh, K. (2022). Epileptic seizure detection using deep learning through min max scaler normalization. International Journal of Health Sciences, 6(May), 10981–10996. https://doi.org/10.53730/ijhs.v6ns1.7801

Dinathi, D. A., Ramadanti, E., & Chandranegara, D. R. (2024). Diabetes Detection Using Extreme Gradient Boosting (XGBoost) with Hyperparameter Tuning. Indonesian Journal of Electronics, Electromedical Engineering, and Medical Informatics, 6(2), 78–84.

Farhadi, S., Tatullo, S., Boveiri Konari, M., & Afzal, P. (2024). Evaluating StackingC and ensemble models for enhanced lithological classification in geological mapping. Journal of Geochemical Exploration, 260, 107441. https://doi.org/10.1016/j.gexplo.2024.107441

Jain, R., & Ganesan, R. A. (2024). Effective Diagnosis of Various Sleep Disorders by LEE Classifier: LightGBM-EOG-EEG. IEEE, 29(4), 2581–2588. https://doi.org/10.1109/JBHI.2024.3524079

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T. Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. Advances in Neural Information Processing Systems, 2017-Decem(Nips), 3147–3155.

Krzywicka, M., & Wosiak, A. (2023). Sensitivity of Standard Evaluation Metrics for Disease Classification and Progression Assessment Based on Whole-Body Imaging. Procedia Computer Science, 225, 4314–4323. https://doi.org/10.1016/j.procs.2023.10.428

Lai, J. P., Lin, Y. L., Lin, H. C., Shih, C. Y., Wang, Y. P., & Pai, P. F. (2023). Tree-Based Machine Learning Models with Optuna in Predicting Impedance Values for Circuit Analysis. Micromachines, 14(2). https://doi.org/10.3390/mi14020265

Low, M. X., Yap, T. T. V., Soo, W. K., Ng, H., Goh, V. T., Chin, J. J., & Kuek, T. Y. (2022). Comparison of Label Encoding and Evidence Counting for Malware Classification. Journal of System and Management Sciences, 12(6), 17–30. https://doi.org/10.33168/JSMS.2022.0602

Mao, X., Ren, N., Dai, P., Jin, J., Wang, B., Kang, R., & Li, D. (2024). A variable weight combination prediction model for climate in a greenhouse

based on BiGRU-Attention and LightGBM. Computers and Electronics in Agriculture, 219, 108818. https://doi.org/10.1016/j.compag.2024.108818

Medic, G., Wille, M., & Hemels, M. E. H. (2017). Short- and long-term health consequences of sleep disruption. Nature and Science of Sleep, 9, 151–161. https://doi.org/10.2147/NSS.S134864

Ouadi, B., Khatir, A., Magagnini, E., Mokadem, M., Abualigah, L., & Smerat, A. (2024). Optimizing silt density index prediction in water treatment systems using pressure-based gradient boosting hybridized with Salp Swarm Algorithm. Journal of Water Process Engineering, 68(September), 106479. https://doi.org/10.1016/j.jwpe.2024.106479

Safavi, V., Mohammadi Vaniar, A., Bazmohammadi, N., Vasquez, J. C., Keysan, O., & Guerrero, J. M. (2024). Early prediction of battery remaining useful life using CNN-XGBoost model and Coati optimization algorithm. Journal of Energy Storage, 98, 113176. https://doi.org/10.1016/j.est.2024.113176

Schapire, R. E., Labs, T., Avenue, P., Room, A., & Park, F. (1999). A Brief Introduction to Boosting. Ijcai 99, 1401–1406. http://u.math.biu.ac.il/~louzouy/courses/seminar/boost1.pdf

Scikit-learn Developers. (n.d.). train_test_split — dokumentasi scikit-learn 1.6.1. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

Shirdel, M., Di Mauro, M., & Liotta, A. (2024). Worthiness Benchmark: A novel concept for analyzing binary classification evaluation metrics. Information Sciences, 678, 120882. https://doi.org/10.1016/j.ins.2024.120882

Srinivas, P., & Katarya, R. (2022). hyOPTXg: OPTUNA hyper-parameter optimization framework for predicting cardiovascular disease using XGBoost. Biomedical Signal Processing and Control, 73(June 2021), 103456. https://doi.org/10.1016/j.bspc.2021.103456

Su, Q., Chen, L., & Qian, L. (2024). Optimization of big data analysis resources supported by XGBoost algorithm: Comprehensive analysis of industry 5.0 and ESG performance. Measurement: Sensors, 36, 101310. https://doi.org/10.1016/j.measen.2024.101310

Tharmalingam, L. (2021). Sleep Health and Lifestyle Dataset. Kaggle. https://www.kaggle.com/datasets

Wang, W., & Sun, D. (2021). The improved AdaBoost algorithms for imbalanced data classification. Information Sciences, 563, 358–374. https://doi.org/10.1016/j.ins.2021.03.042

Wu, Z., Liu, Y., Fang, S., Shen, W., Li, X., Mao, Z., & Wu, S. (2024). Integration of geographic features and bathymetric inversion in the Yangtze River's Nantong Channel using gradient boosting machine algorithm with ZY-1E satellite and multibeam data. Geomatica, 76, 100027. https://doi.org/10.1016/j.geomat.2024.100027

Halaman ini sengaja dikosongkan