BAB II

LANDASAN TEORI

2.1 Landasan Teori

2.1.1 *Machine learning*

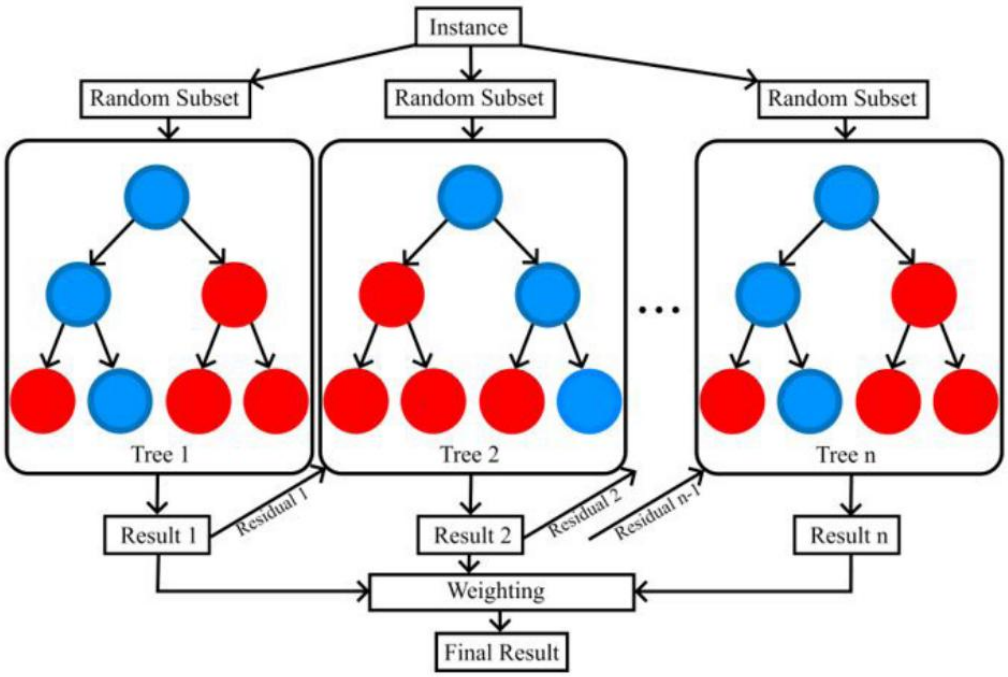
Machine learning (ML) hadir sebagai solusi efektif dalam menangani dan menganalisis kumpulan data berukuran besar secara efisien (Wahyono, 2019) Metode ini memungkinkan komputer untuk mengotomatisasi proses pengambilan keputusan, meningkatkan akurasi prediksi, serta mengoptimalkan berbagai tugas kompleks di berbagai bidang seperti kesehatan, keuangan, dan teknologi. *Machine learning* merupakan cabang khusus dari kecerdasan buatan yang berfokus pada mengajarkan mesin bagaimana belajar, sementara kecerdasan buatan secara umum bertujuan meniru kemampuan manusia.

Berbeda dengan pendekatan berbasis model dengan persamaan yang telah ditentukan sebelumnya, algoritma *Machine learning* memperoleh pemahaman langsung dari data melalui teknik komputasi (Cantt, 2023). Pendekatan ini memungkinkan mesin untuk mengidentifikasi pola, membuat prediksi, dan meningkatkan kinerjanya seiring bertambahnya data, sehingga dapat diterapkan dalam berbagai bidang seperti kesehatan, keuangan, dan teknologi. Prediksi harga tiket pesawat merupakan salah satu aplikasi *Machine learning* yang digunakan untuk memproses data historis dan mengidentifikasi pola harga. Berbagai algoritma telah digunakan untuk tugas ini, termasuk *Random Forest*, *XGBoost*, dan metode *Stacking Ensemble learning*. *Machine learning* dapat membantu dalam

mengoptimalkan strategi harga bagi maskapai penerbangan serta memberikan wawasan bagi pelanggan dalam menentukan waktu terbaik untuk membeli tiket.

2.1.1.1 Extreme Gradient Boosting

Extreme Gradient Boosting (*XGBoost*) membangun model secara bertahap dengan menambahkan pohon keputusan baru pada setiap tahap pelatihan, di mana pohon tersebut dirancang untuk mengoreksi kesalahan prediksi dari model sebelumnya (Oztornaci dkk., 2024). Proses ini terus berlangsung hingga jumlah pohon yang ditentukan tercapai. Setelah itu, seluruh *output* dari pohon-pohon yang telah dibentuk digabungkan, dan masing-masing pohon diberi bobot sesuai tingkat kontribusinya dalam meningkatkan akurasi prediksi akhir. *XGBoost* membangun model secara bertahap dengan menambahkan pohon keputusan baru pada setiap tahap pelatihan, di mana pohon tersebut dirancang untuk mengoreksi kesalahan prediksi dari model sebelumnya. Proses ini terus berlangsung hingga jumlah pohon yang ditentukan tercapai. Setelah itu, seluruh *output* dari pohon-pohon yang telah dibentuk digabungkan, dan masing-masing pohon diberi bobot sesuai tingkat kontribusinya dalam meningkatkan akurasi prediksi akhir. Gambar 2.1 merupakan Model *XGBoost* yang diperoleh dari penelitian (Oztornaci dkk., 2024).



Gambar 2. 1 Model *XGBoost* (Oztornaci dkk., 2024)

XGBoost adalah metode *Machine learning* yang membangun model secara bertahap dengan menambahkan pohon keputusan atau *decision tree* yang berfungsi memperbaiki kesalahan model sebelumnya (Li dkk., 2023). Struktur dasarnya adalah pohon *CART*, yang memisahkan data menjadi dua bagian secara rekursif untuk meminimalkan kesalahan.

Secara prosedural, alur pembentukan model *XGBoost* dengan parameter *default* dapat dijabarkan melalui notasi algoritma berikut. Algoritma ini menggambarkan bagaimana model membangun *ensemble* pohon keputusan secara bertahap untuk meminimalkan fungsi kerugian (*loss function*) berdasarkan parameter bawaan. Algoritma dari *XGBoost* dalam konteks regresi dapat dijelaskan melalui tahapan pada Tabel 2.1 yang diadaptasi oleh peneliti (Mienye & Sun, 2022) berikut.

Tabel 2. 1 Algoritma *XGBoost* Default

Algoritma *XGBoost Default*

Input:

D = {(x_i, y_i)} : Data latih (Fitur dan Target Harga Tiket)

L(y, F(x)) : Fungsi Kerugian (Squared Error untuk Regresi)

Parameter Default (Inisialisasi):

n_estimators (K) : 100 (Jumlah pohon iterasi)

learning_rate (η) : 0.3 (Langkah pembelajaran)

max_depth : 6 (Kedalaman maksimum pohon)

reg_lambda (λ) : 1 (Regularisasi L2)

gamma (γ) : 0 (Minimum loss reduction)

Output:

F(x) : Model Prediksi Akhir

Proses:

1. Inisialisasi model awal dengan nilai konstan:
   $F_0(x) = \text{argmin}_c \Sigma L(y_i, c)$

2. Untuk k = 1 sampai K (iterasi pohon ke-1 hingga 100):
   a. Hitung statistik gradien orde pertama (g_i) dan kedua (h_i)
      untuk setiap data ke-i berdasarkan prediksi iterasi sebelumnya:
      $g_i = \partial L(y_i, F_{k-1}(x_i)) / \partial F_{k-1}(x_i)$
      $h_i = \partial^2 L(y_i, F_{k-1}(x_i)) / \partial F_{k-1}(x_i)^2$

   b. Bangun pohon keputusan f_k(x) dengan batasan 'max_depth = 6':
      - Tentukan struktur pohon dengan mencari titik pemisahan (split) terbaik
        yang memaksimalkan Gain:
        $\text{Gain} = 1/2 * [(\Sigma g_L)^2/(\Sigma h_L + \lambda) + (\Sigma g_R)^2/(\Sigma h_R + \lambda) - (\Sigma g)^2/(\Sigma h + \lambda)] - \gamma$
      - Tetapkan bobot optimal (w) pada setiap daun (j):
        $w_j = - (\Sigma_{i \in Ij} g_i) / (\Sigma_{i \in Ij} h_i + \lambda)$

   c. Perbarui model ensemble dengan menambahkan pohon baru yang telah diskalakan
      oleh learning rate (η = 0.3):
      $F_k(x) = F_{k-1}(x) + \eta * f_k(x)$

3. Kembalikan model akhir F_K(x)

Pada Tabel 2.1 terdapat algoritma *XGBoost default* diawali dengan inisialisasi model prediksi dasar (F_0) yang memberikan nilai konstan untuk meminimalkan fungsi objektif awal. Proses pembelajaran kemudian berlangsung secara iteratif sebanyak 100 putaran (*n_estimators*), di mana pada setiap iterasinya, sistem menghitung statistik gradien orde pertama (g_i) dan orde kedua (h_i) dari fungsi kerugian (*loss function*) terhadap prediksi sebelumnya untuk mengidentifikasi arah kesalahan model. Berdasarkan nilai gradien tersebut, sebuah pohon keputusan baru dibangun untuk memprediksi *residual error*, dengan kompleksitas struktur yang dibatasi oleh kedalaman maksimal (*max_depth*) sebesar 6 level guna mencegah terjadinya *overfitting*. Struktur pohon ditentukan melalui pencarian titik pemisahan terbaik yang memaksimalkan nilai *Gain*, dan setiap daun pada pohon tersebut diberikan bobot optimal (w) yang merepresentasikan nilai koreksi harga. Akhirnya, model diperbarui secara aditif pada setiap langkah dengan menambahkan kontribusi pohon baru yang telah diskalakan oleh *learning rate* (η) sebesar 0.3, sehingga menghasilkan model *ensemble* akhir (F_K)yang merupakan akumulasi cerdas dari seluruh pohon keputusan yang terbentuk.

Berikut merupakan formulasi matematika dari *XGBoost* yang mana prediksi akhir $\widehat{y_i}$ untuk data ke-i diperoleh dari penjumlahan seluruh fungsi prediksi dari κ pohon keputusan. Masing-masing fungsi $f_{k(x_i)}$ adalah model pohon ke-k yang termasuk dalam himpunan $\mathcal{F}$, yaitu kumpulan semua pohon keputusan yang dilatih. Penjumlahan seluruh fungsi prediksi dari κ pohon keputusan dihitung dengan Persamaan 2.1.

$$\widehat{y_i} = \sum_{k=1}^{K} f_{k(x_i),\ f_k \in \mathcal{F}} \tag{2.1}$$

Fungsi kerugian $l(y_{i,}\, \hat{y}_i)$ yang menghitung seberapa jauh prediksi terhadap nilai sebenarnya. Fungsi regularisasi $\Omega(f_t)$ yang mengontrol kompleksitas model agar tidak *overfitting*. Fungsi objektif yang terdiri dari dua bagian: *loss function* dan *regularization term* dihitung dengan Persamaan 2.2.

$$L^{(t)} = \sum_{i=1}^{n} l(y_{i,}\, \hat{y}_i{}^{(t-1)} + f_{t(x_i))} + \Omega(f_t) \tag{2.2}$$

Fungsi regularisasi ini berfungsi untuk mengendalikan kompleksitas dari pohon keputusan f_k. Dimana γ dan λ adalah parameter regulasi untuk menghindari pohon menjadi terlalu rumit, T adalah jumlah daun pohon, dan ω_j adalah bobot pada daun ke-j. Fungsi regularisasinya sendiri dihitung dengan Persamaan 2.3.

$$\Omega(f_k) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} \omega_j^2 \tag{2.3}$$

Untuk memudahkan optimasi, fungsi objektif ini dikembangkan dengan pendekatan Taylor, yang melibatkan turunan pertama dan kedua dari fungsi *loss*. Fungsi objektif dengan ekspansi Taylor orde dua dihitung dengan Persamaan 2.4.

$$L^{(t)} \approx \sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \Omega(f_t) \tag{2.4}$$

Untuk menentukan seberapa besar kontribusi *output* dari daun ke-j terhadap prediksi akhir. Bobot ini akan digunakan saat pohon tersebut menyumbangkan nilainya dalam proses penjumlahan prediksi akhir $(\widehat{y_i})$. Agar nilai *output* dari

setiap daun dalam pohon bisa meminimalkan kesalahan prediksi sambil tetap memperhatikan regularisasi dapat dihitung dengan menggunakan Persamaan 2.5.

$$\omega_j^* = -\frac{\sum_{i\epsilon Ij} g_i}{\sum_{i\epsilon Ij} h_i + \lambda} \tag{2.5}$$

Persamaan 2.6 digunakan untuk menghitung nilai minimum dari fungsi objektif setelah optimasi bobot daun. Semakin kecil nilai ini, semakin baik struktur pohon dalam meminimalkan kesalahan prediksi. Nilai minimal dari fungsi objektif setelah pemangkasan pohon dapat dihitung dengan Persamaan 2.6.

$$\tilde{L}^{(t)} = -\frac{1}{2}\sum_{j=1}^{T}\frac{\left(\sum_{i\epsilon Ij} g_i\right)^2}{\sum_{i\epsilon Ij} h_i + \lambda} + \gamma T \tag{2.6}$$

Nilai *Gain* mengukur seberapa besar peningkatan kualitas model jika data dibagi, semakin besar *Gain*, semakin baik pemisahan yang dipilih. Selanjutnya, untuk memilih pemisahan terbaik pada setiap *node*, *XGBoost* menghitung nilai *Gain* dengan menggunakan Persamaan 2.7.

$$Gain = \frac{1}{2}\left[\frac{\left(\sum_{i\epsilon I_L} g_i\right)^2}{\sum_{i\epsilon I_L} h_i + \lambda} + \frac{\left(\sum_{i\epsilon I_R} g_i\right)^2}{\sum_{i\epsilon I_R} h_i + \lambda} - \frac{\left(\sum_{i\epsilon I} g_i\right)^2}{\sum_{i\epsilon I} h_i + \lambda}\right] - \gamma \tag{2.7}$$

2.1.2 *Hyperparameter Tuning*

Hyperparameter Tuning merupakan aspek penting dalam *machine learning*, karena berperan dalam mengatur perilaku serta tingkat kompleksitas model, yang secara langsung memengaruhi kinerja dan akurasi prediksi (Ilemobayo dkk., 2024). *Hyperparameter Tuning* adalah proses untuk menemukan kombinasi parameter terbaik yang dapat meningkatkan performa model *machine learning*.

Hyperparameter Tuning sangat penting untuk meningkatkan kinerja dan kemampuan model *machine learning* dalam mengenali pola secara akurat.

Beberapa faktor utama yang memengaruhi performa model, seperti kualitas data, jenis algoritma yang digunakan, dan tingkat kompleksitas model, juga berperan besar dalam hasil akhir. Selain itu, pengaturan *Hyperparameter* seperti laju pembelajaran dan ukuran *batch* dapat berdampak langsung pada proses pelatihan (Ilemobayo dkk., 2024). *Hyperparameter* berbeda dari parameter model karena tidak diperoleh melalui proses pelatihan, melainkan harus ditentukan sebelum pelatihan dilakukan. Teknik *Tuning* yang umum digunakan meliputi *Grid Search, Random Search, Bayesian Optimization*, dan *Tree-structured Parzen Estimator* (TPE) digunakan untuk menemukan kombinasi terbaik guna meningkatkan akurasi dan efisiensi model. Penelitian tersebut juga menyajikan contoh hasil *Tuning* yang menunjukkan peningkatan akurasi model serta memberikan rekomendasi untuk penyetelan model yang lebih efisien dan efektif (Koch dkk., 2017). Dengan begitu penelitian ini akan menggunakan *Bayesian Optimization* sebagai metode *Hyperparameter Tuning* yang dipakai.

2.1.2.1 *Bayesian Optimization*

Bayesian Optimization adalah metode yang menggunakan model probabilistik untuk mencari kombinasi *Hyperparameter* terbaik secara efisien. Pengoptimalan *Bayesian* telah terbukti efektif dalam menyesuaikan pengaturan *Hyperparameter* secara otomatis, sehingga membantu berbagai model bekerja lebih optimal dan efisien (Yang dkk., 2024) Pendekatan ini menggunakan fungsi perkiraan untuk memprediksi kinerja model berdasarkan sampel yang telah diuji,

kemudian menentukan kombinasi *Hyperparameter* baru yang diharapkan dapat meningkatkan performa secara keseluruhan. Salah satu implementasi populer dari *Bayesian Optimization* adalah *BayesSearchCV*, yang digunakan dalam penelitian ini.

Penelitian ini membahas bagaimana teknik Optimasi *Bayesian* digunakan untuk menyelesaikan masalah optimasi dalam fisika akselerator. Algoritma ini dianggap unggul karena dapat memprediksi hasil dengan lebih efisien menggunakan model statistik sebagai perkiraan, tanpa harus menguji setiap kemungkinan secara langsung. Hal ini sangat bermanfaat dalam menangani tantangan yang kompleks, terutama ketika ada gangguan atau kebisingan selama operasi akselerator, serta dalam simulasi fisika yang membutuhkan banyak sumber daya (Roussel dkk., 2024). Dengan keberhasilan Optimasi *Bayesian* dalam menangani berbagai masalah kompleks, serta pengakuan luas terhadap keunggulannya dalam fisika akselerator, penelitian yang akan dilakukan memiliki potensi besar untuk mencapai hasil yang optimal dan berkontribusi secara signifikan dalam meningkatkan efisiensi serta akurasi model yang dikembangkan.

2.1.3 *Evaluation Metrics*

Metrik seperti *Mean Absolute Error* (MAE), *Mean Squared Error* (MSE), *Root Mean Squared Error* (RMSE), dan *R-squared* (R^2) digunakan untuk mengevaluasi model (Kumar, 2023).

2.1.3.1 Mean Absolute Error (MAE)

Mean Absolute Error (MAE) merupakan rata-rata dari selisih absolut antara nilai aktual dan nilai prediksi. MAE memberikan bobot yang sama pada setiap

perbedaan dan tidak terlalu sensitif terhadap *outlier*. Nilai MAE dapat dihitung dengan Persamaan 2.8.

$$MAE = \frac{1}{n}\sum_{i=1}^{n} | y_i - \hat{y}_i | \quad (2.8)(\text{Kumar, 2023})$$

2.1.3.2 Mean Squared Error (MSE)

Mean Squared Error (MSE) adalah rata-rata dari kuadrat selisih antara nilai estimasi dan nilai aktual. MSE memberikan penalti yang lebih besar terhadap kesalahan yang lebih besar, sehingga model dengan nilai MSE yang lebih kecil dianggap lebih baik karena memiliki tingkat kesalahan yang lebih rendah. Nilai MSE dapat dihitung dengan Persamaan 2.9.

$$MSE = \frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \quad (2.9)(\text{Kumar, 2023})$$

2.1.3.3 Root Mean Squared Error (RMSE)

Root Mean Squared Error (RMSE) dihitung dengan mengambil akar kuadrat dari MSE. RMSE memberikan penalti lebih besar untuk kesalahan yang besar, sehingga metrik ini berguna ketika kesalahan besar tidak diinginkan. Nilai RMSE dapat dihitung dengan Persamaan 2.10.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2} \quad (2.10)(\text{Kumar, 2023})$$

2.1.3.4 Koefisien Determinasi (R^2)

Koefisien Determinasi (R^2) adalah ukuran statistik yang menunjukkan seberapa baik variabel independen dapat menjelaskan perubahan dalam variabel dependen. Semakin mendekati 1 nilai R^2, semakin baik model dalam menjelaskan variasi dalam data. Nilai R^2 dapat dihitung dengan Persamaan 2.11.

$$R^2 = 1 - \frac{RSS}{TSS} \qquad (2.11)(\text{Kumar, 2023})$$

Keterangan:

y_i = nilai aktual,

$\hat{y}_i$ = nilai prediksi,

n = jumlah total data,

RSS = *Residual Sum of Squares*,

TSS = *Total Sum of Squares*.

2.2 Penelitian Terkait

2.2.1 State of The Art

Pada tabel 2.2 dapat dilihat beberapa penelitian terdahulu yang telah mengkaji penerapan *machine learning* dalam prediksi harga tiket pesawat maupun optimasi model menggunakan *Bayesian Optimization*, yang menjadi landasan bagi penelitian ini.

Tabel 2. 2 *State of The Art*

No	Peneliti	Judul	Objek Penelitian	Algoritma	Hasil Penelitian
1	(Dalal dkk., 2022)	*Hybrid XGBoost model with Hyperparameter Tuning for prediction of liver disease with better accuracy*	*Prediction of liver disease*	*Hybrid XGBoost*	Algoritma pohon keputusan seperti CART dan CHAID tidak memberikan tingkat akurasi yang tinggi dalam memprediksi penyakit hati pada kelompok pasien di India. Namun, model *Hybrid XGBoost* yang dikembangkan dalam penelitian ini

					menunjukkan tingkat akurasi yang lebih baik, mencapai 93,65%.
2	(Porter & Tech, n.d.)	*Studying the Acquisition Function of Bayesian Optimization With Machine learning With DNA Reads.*	*DNA Reads*	*Bayesian Optimization With Machine learning*	Penelitian ini menganalisis pengaruh fungsi akuisisi dalam optimasi menggunakan random forests dan multilayer perceptrons pada prediksi performa pemetaan DNA pendek yang telah mengalami perlakuan bisulfit, metode sekuensing untuk mempelajari metilasi epigenetik. Hasilnya menunjukkan bahwa *Bayesian Optimization* memiliki performa serupa dengan grid search, namun kurang efisien dalam

					paralelisasi saat diimplementasikan menggunakan skopt.
3	(Tiastama & Budi, 2024)	Perbandingan *Random Search* dan Algoritma Genetika dalam Penyetelan *Hyperparameter XGBoost* pada *Retail Sales Forecasting*	*Retail Sales Forecasting*	Perbandingan *Random Search* dan Algoritma Genetika dalam Penyetelan *Hyperparameter XGBoost*	Penelitian ini memperoleh hasil *Random Search* lebih unggul dari algoritma genetika dengan nilai RMSE pada proses latih dan proses uji sebesar 0.123 dan 0.122. Sementara itu, nilai RMSE yang dihasilkan oleh algoritma genetika pada proses latih dan proses uji yaitu sebesar 0.333 dan 0.332.
4	(H. Darmawan dkk., 2023)	*GRU and XGBoost Performance with*	*Weather Prediction System*	GRU and *XGBoost*	Hasil pengujian dengan data dari BMKG Juanda dan IoT menunjukkan

		Hyperparameter Tuning Using GridSearchCV and Bayesian Optimization on an IoT-Based Weather Prediction System		Performance with *Hyperparameter Tuning Using GridSearchCV and Bayesian Optimization*	bahwa dalam tahap pertama, *XGBoost* regression lebih unggul dengan RMSE 1.2728 dibanding GRU regression (RMSE 1.5517). Namun, pada tahap kedua, GRU regression lebih baik (RMSE 2.23) dibanding *XGBoost* (RMSE 2.28). Dalam klasifikasi, GRU memiliki skor F1 0.88, lebih tinggi dibanding *XGBoost* (0.86), meskipun keduanya memiliki akurasi yang sama (0.75) dengan data IoT. Model GRU classification dianggap lebih unggul karena mempertimbangkan konteks

					prediksi, sehingga lebih akurat dalam memprediksi kemungkinan hujan.
5	(N dkk., 2024)	*XGBOOST HYPERPARAMETER OPTIMIZATION USING RANDOMIZEDSEARCHCV FOR ACCURATE FOREST FIRE DROUGHT CONDITION PREDICTION*	*ACCURATE FOREST FIRE DROUGHT CONDITION PREDICTION*	*XGBOOST HYPERPARAMETER OPTIMIZATION USING RANDOMIZEDSEARCHCV*	Hasil penelitian menunjukkan bahwa model *XGBoost* yang telah dioptimasi menghasilkan penurunan *Mean Squared Error* (MSE) dan peningkatan nilai *R-squared* dibandingkan dengan model default. Model yang dioptimasi mencapai MSE sebesar 0.0210 dan R2 sebesar 0.9820 pada data uji, yang mengindikasikan peningkatan akurasi prediksi secara signifikan terhadap

					kondisi kekeringan bahan bakar hutan.
6	(Putatunda & Rama, 2019)	A Modified *Bayesian Optimization* based Hyper-Parameter *Tuning* Approach for *Extreme Gradient Boosting*	*Extreme Gradient Boosting*	*Bayesian Optimization* based Hyper-Parameter *Tuning*	Penelitian ini mengusulkan metode Randomized-Hyperopt untuk meningkatkan optimasi *Hyperparameter* pada *XGBoost*. Dibandingkan dengan Grid Search, Random Search, dan Hyperopt, uji coba pada 10 dataset menunjukkan bahwa Randomized-Hyperopt memberikan hasil terbaik dalam hal akurasi prediksi dan waktu eksekusi.

7	(Elina dkk., 2022)	Penerapan Metode *Extreme Gradient Boosting* (*XGBOOST*) pada Klasifikasi Nasabah Kartu Kredit	Klasifikasi Nasabah Kartu Kredit	*Extreme Gradient Boosting* (*XGBOOST*)	Hasil dari penelitian ini membuktikan bahwa penggunaan algoritma dengan *Hyperparameter Tuning* dapat meningkatkan kinerja algoritma *Extreme Gradient Boosting* dalam proses klasifikasi nasabah kartu kredit dengan akurasi sebesar 80,039%, presisi sebesar 81,338% dan nilai recall sebesar 96,854%.
8	(M, 2024)	*MODELING AND PREDICTING INDONESIA RICE PRICES USING HYPERPARAMETER OPTIMIZATION XGBOOST*	*PREDICTING INDONESIA RICE PRICES*	*HYPERPARAMETER OPTIMIZATION XGBOOST*	Studi ini menggunakan model *XGBoost* dengan *Hyperparameter Tuning* dan *cross-validation* untuk meramalkan harga beras di Indonesia hingga dua tahun setelah Juli 2024.

					Dengan data Januari 2010 – Juli 2024 dan 4-*fold cross-validation*, model dievaluasi melalui 26.000+ fits. Hasilnya menunjukkan *XGBoost* mampu menangkap pola kompleks dalam data dan menghasilkan prediksi akurat.
9	(Dinanthi dkk., 2024)	*Diabetes Detection Using Extreme Gradient Boosting (XGBoost) with Hyperparameter Tuning*	*Diabetes Detection*	*Extreme Gradient Boosting (XGBoost) with Hyperpara*	Studi ini meneliti kinerja *XGBoost* dalam klasifikasi diabetes dengan optimasi GridSearchCV dan RandomSearchCV serta penyeimbangan data menggunakan SMOTE. Hasilnya, GridSearchCV

				meter Tuning	mencapai akurasi 85%, sementara RandomSearchCV mencapai 83%.
10	(Rhouas dkk., 2024)	*Enhancing currency prediction in international e-commerce: Bayesian-optimized random forest approach using the Klarna dataset*	*Prediction in international e-commerce*	*Bayesian-optimized random forest*	Penelitian ini mengoptimalkan algoritma Random Forest menggunakan *Bayesian Optimization* pada dataset Klarna *E-commerce* untuk meningkatkan prediksi nilai tukar mata uang dalam perdagangan global. Dengan 14 iterasi, 1 job, dan random state 684, model mencapai MSE -0.9891 dan akurasi 74.63%. Studi ini menyoroti peran *Bayesian Optimization* dalam meningkatkan kinerja model, memberikan strategi

					yang lebih baik bagi bisnis dalam manajemen risiko keuangan global.

Tabel 2.2 merangkum berbagai penelitian yang membahas optimasi *XGBoost* dan *Bayesian Optimization* di berbagai bidang, seperti prediksi penyakit, klasifikasi, peramalan harga, dan sistem berbasis IoT. Secara umum, penelitian-penelitian ini menunjukkan bahwa penyetelan *Hyperparameter*, terutama dengan *Bayesian Optimization, GridSearchCV,* dan *RandomizedSearchCV*, dapat meningkatkan performa model secara signifikan. Misalnya, dalam prediksi penyakit hati, penggunaan *Hybrid XGBoost* berhasil mencapai tingkat akurasi 93,65% (Dalal dkk., 2022). Selain itu, *Bayesian Optimization* juga diterapkan untuk meningkatkan efisiensi pemetaan DNA *Reads*, meskipun hasilnya masih menunjukkan bahwa metode ini perlu dioptimalkan lebih lanjut dalam hal paralelisasi (Porter & Tech, n.d.).

Dalam *Retail Sales Forecasting*, penelitian menemukan bahwa *Random Search* lebih unggul dibandingkan Algoritma Genetika, dengan nilai RMSE yang lebih rendah (Tiastama & Budi, 2024). Sementara itu, dalam prediksi cuaca berbasis IoT, kombinasi GRU dan *XGBoost* memberikan hasil yang bervariasi. *XGBoost* lebih unggul dalam regresi tahap awal, sementara GRU lebih baik dalam klasifikasi, ditunjukkan dengan skor F1 yang lebih tinggi (H. Darmawan dkk., 2023). Dalam bidang mitigasi bencana, optimasi *Hyperparameter XGBoost* menggunakan *RandomizedSearchCV* untuk prediksi kekeringan akibat kebakaran hutan berhasil meningkatkan akurasi model dengan R^2 sebesar 0.982, jauh lebih baik dibandingkan model default (N dkk., 2024)

Lebih lanjut, penelitian oleh Putatunda & Rama menemukan bahwa metode *Randomized-Hyperopt* lebih efektif dibandingkan *Grid Search* dan *Random Search*

dalam menyetel *Hyperparameter XGBoost*. Di bidang keuangan, model *XGBoost* diterapkan untuk klasifikasi nasabah kartu kredit, dan hasilnya menunjukkan bahwa penyetelan *Hyperparameter* dapat meningkatkan akurasi hingga 80,039% (Elina dkk., 2022). Studi lain juga membuktikan bahwa *XGBoost* dengan *Hyperparameter Tuning* dan *cross-validation* mampu mengidentifikasi pola kompleks dalam data historis harga beras di Indonesia, menghasilkan prediksi yang lebih akurat (M, 2024). Selain itu, dalam deteksi diabetes, penggunaan *GridSearchCV* menghasilkan akurasi tertinggi sebesar 85%, lebih tinggi dibandingkan dengan *RandomSearchCV* (Dinanthi dkk., 2024).

Terakhir, *Optimasi Bayesian* juga diterapkan dalam model *Random Forest* untuk prediksi nilai tukar mata uang dalam *e-commerce*, dengan hasil yang menunjukkan peningkatan akurasi hingga 74,63% (Rhouas dkk., 2024). Berdasarkan berbagai penelitian sebelumnya, dapat disimpulkan bahwa optimasi *Hyperparameter*, khususnya dengan *Bayesian Optimization*, memiliki peran besar dalam meningkatkan akurasi model berbasis *XGBoost*. Oleh karena itu, penelitian ini bertujuan untuk memperkuat temuan tersebut dengan menerapkan metode serupa dalam prediksi harga tiket pesawat, serta mengevaluasi seberapa efektif *Bayesian Optimization* dibandingkan metode *Tuning* lainnya.