BAB 2
LANDASAN TEORI

2.1 Machine Learning

Machine Learning merupakan cabang dari *Artificial Intelligence (AI)* yang menggambarkan kemampuan sistem komputer untuk mempelajari pola dari data pelatihan atau kumpulan data (*dataset*) sehingga proses pembuatan model analitis dapat dilakukan secara otomatis. Pendekatan ini memungkinkan sistem untuk mengidentifikasi hubungan yang bermakna dari data dan menggunakan pola tersebut untuk membantu proses prediksi maupun pengambilan keputusan [20] [21]. Kelebihan ini juga membuat *Machine Learning* cocok digunakan untuk menangani permasalahan yang melibatkan banyak variabel atau fitur [22]. Mengingat karakteristik ini, *Machine Learning* menjadi pilihan pendekatan yang relevan untuk memprediksi harga rumah. *Machine Learning* hadir dalam berbagai macam bentuk dan pada umumnya, dapat dibedakan menjadi tiga pendekatan utama, yaitu *Supervised Learning, Unsupervised Learning*, dan *Reinforcement Learning*, dan permasalahan prediksi harga rumah termasuk ke dalam kategori *Supervised Learning*.

2.2 Supervised Learning

Seperti namanya, *Supervised Learning* merupakan pendekatan pembelajaran yang menggunakan data berlabel, di mana setiap data pelatihan memiliki nilai target yang digunakan sebagai acuan selama proses pembelajaran model. Algoritma akan melatih model statistik dan akan dapat membuat prediksi terhadap suatu fitur yang diinginkan atau data baru yang belum memiliki label [23]. *Supervised Learning* juga dipecah menjadi dua kategori dasar, yaitu *Classification*, dan *Regression*. *Classification* berfokus pada data *categorical* atau nilai diskrit, sedangkan *Regression* lebih condong pada data *numerical* atau mengestimasi nilai yang bersifat kontinu [24]. Performa setiap algoritma dapat berbeda tergantung pada karakteristik data dan permasalahan yang dihadapi. Untuk *Regression*, masalah seperti prediksi harga perumahan merupakan contoh tugas *Regression* karena harga rumah tentunya tidak hanya ditentukan oleh 1 faktor, dan faktor-faktor yang memengaruhi harga rumah juga tidak selalu memiliki bobot pengaruh yang sama [25]. Oleh karena itu, pengaplikasian *Supervised Learning* sesuai digunakan

untuk memprediksi harga rumah karena mampu mempelajari hubungan antara berbagai fitur yang tersedia dengan nilai target yang ingin diprediksi.

2.3 Algoritma yang Digunakan dalam Penelitian

Penelitian ini menggunakan tiga model *Machine Learning*, yaitu *Random Forest*, *XGBoost*, dan model *Hybrid Ensemble* berbasis *weighted average*. Ketiga model tersebut dipilih karena mewakili pendekatan *ensemble learning* yang berbeda, yaitu *bagging*, *boosting*, dan kombinasi prediksi antar model. Masing-masing pendekatan memiliki karakteristik pembelajaran yang berbeda dalam menangani pola data yang kompleks dan hubungan non-linear. Oleh karena itu, model-model tersebut digunakan untuk menganalisis serta membandingkan performa prediksi pada kasus prediksi harga rumah di Jakarta.

2.3.1 Random Forest

Random Forest merupakan algoritma *supervised machine learning* yang menggunakan pendekatan *ensemble* untuk melakukan prediksi. Algoritma ini membangun beberapa *Decision Tree* dari *dataset* yang tersedia yang digunakan dan kemudian menggabungkan hasil dari keluaran *Decision Tree* tersebut untuk melakukan prediksi [26]. Salah satu kelebihan dari *Random Forest* yang membuat model ini sangat populer di bidang *machine learning* adalah kemampuannya untuk memproses variabel yang bersifat *categorical* dan *continuous*, sehingga model ini dapat diimplementasikan untuk masalah bersifat klasifikasi dan regresi. Selain itu, *Random Forest* tidak bergantung pada satu model matematis tunggal seperti *Linear Regression*, melainkan menggabungkan hasil dari banyak *Decision Tree* untuk meningkatkan akurasi prediksi [27].

Pada algoritma *Random Forest*, setiap *Decision Tree* tidak dibangun menggunakan seluruh dataset asli secara langsung. Sebaliknya, proses pelatihan dilakukan dengan memanfaatkan sejumlah dataset baru yang dihasilkan melalui teknik *Bootstrap Aggregation (Bagging)*. Teknik ini membentuk beberapa sampel data pelatihan dengan cara mengambil data secara acak dari dataset utama disertai pengembalian (*sampling with replacement*). Selain variasi data pelatihan, *Random Forest* juga menerapkan pemilihan fitur secara acak pada setiap proses pemisahan node dalam *Decision Tree*. Pendekatan tersebut menghasilkan keragaman yang lebih tinggi antar pohon keputusan sehingga dapat meningkatkan

kemampuan generalisasi model. Selanjutnya, masing-masing *Decision Tree* dilatih menggunakan dataset hasil proses *bootstrap* yang telah dibentuk sebelumnya. Pendekatan ini bertujuan untuk mengurangi variansi dan agar setiap *Decision Tree* menjadi beragam dan tidak saling bergantung satu sama lain. Setelah seluruh *Decision Tree* menghasilkan prediksi, hasil tersebut akan digabung, dan hasil akhir dari prediksinya akan ditentukan melalui proses agregasi, seperti rata-rata pada kasus regresi atau *voting* mayoritas pada kasus klasifikasi [28].

2.3.2 XGBoost (Extreme Gradient Boosting)

Seperti *Random Forest*, *XGBoost* (*eXtreme Gradient Boosting*) merupakan algoritma *supervised machine learning* yang menggunakan pendekatan *ensemble*. Perbedaan utama antara *XGBoost* dan *Random Forest* terletak pada cara pembangunan *Decision Tree*. Pada *Random Forest*, *Decision Tree* dibangun secara paralel menggunakan teknik *bagging* dengan pengambilan data dan fitur secara acak. Berbeda dengan *Random Forest*, *XGBoost* mengembangkan *Decision Tree* secara bertahap, di mana setiap pohon baru dibentuk berdasarkan kesalahan yang dihasilkan oleh model sebelumnya. Dengan cara ini, model secara terus-menerus melakukan penyesuaian untuk meningkatkan akurasi prediksi. Mekanisme tersebut merupakan karakteristik utama dari metode *gradient boosting*, yang bekerja dengan mengurangi nilai fungsi kesalahan (*loss function*) secara iteratif hingga diperoleh performa yang lebih optimal. Selain itu, *XGBoost* memiliki keunggulan dalam mengurangi *overfitting* berkat adanya regularisasi, mampu menangani dataset yang besar, dan juga fleksibilitas dalam penyetelan (*tuning*) berkat *hyperparameter*nya [29].

Dalam proses pembangunan *Decision Tree* pada model XGBoost, tahap awal dimulai dengan membangun *Decision Tree* pertama sebagai dasar prediksi awal. Setelah *Decision Tree* utama menghasilkan prediksi ($\hat{y}_i$), tingkat kesalahan antara hasil prediksi dan nilai aktual (y_i) akan diukur melalui nilai residual atau fungsi kesalahan (*loss function*) ($\mathscr{L}$), dan *Decision Tree* berikutnya dibangun untuk meminimalkan kesalahan tersebut. Proses ini dilakukan secara berulang hingga mencapai jumlah iterasi yang telah ditentukan atau hingga tidak terdapat peningkatan performa yang signifikan. XGBoost juga memiliki beberapa parameter yang dapat diatur untuk mengoptimalkan performa model. Beberapa parameter tersebut diantaranya adalah komponen regularisasi (Ω) yang memberikan penalti terhadap model yang kompleks untuk mencegah *overfitting*, *learning rate* (η) yang

mengatur kontribusi setiap pohon baru terhadap keseluruhan ensemble, jumlah *Decision Tree* (K) yang menentukan banyaknya iterasi dalam proses pembelajaran, dan *Tree Depth* (d) yang mengatur kedalaman maksimum dari setiap *Decision Tree* sehingga mempengaruhi kompleksitas model [30].

$$\mathscr{L}^{(t)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)\right) + \Omega(f_t) \tag{2.1}$$

2.3.3 Hybrid Ensemble Berbasis Weighted Average

Ensemble Learning merupakan pendekatan dalam *Machine Learning* yang menggabungkan beberapa model untuk menghasilkan prediksi akhir. Pendekatan ini didasarkan pada asumsi bahwa model yang berbeda dapat mempelajari pola data yang berbeda pula, sehingga kombinasi beberapa model berpotensi menghasilkan performa yang lebih stabil dibandingkan penggunaan satu model secara individual. Dengan menggabungkan beberapa model, kelemahan dari satu model dapat dikompensasi oleh kelebihan model lainnya. Beberapa teknik yang umum digunakan dalam *ensemble learning* antara lain *bagging*, *boosting*, dan *stacking* [8].

Dalam penelitian ini, pendekatan *hybrid ensemble* dibangun dengan mengombinasikan prediksi dari model *Random Forest* dan *XGBoost* menggunakan metode *weighted average*. Berbeda dengan *stacking* yang memerlukan model tambahan sebagai *meta-learner*, metode *weighted average* menghasilkan prediksi akhir melalui kombinasi linier dari prediksi masing-masing model dengan bobot tertentu. Pendekatan ini relatif sederhana, mudah diinterpretasikan, dan tetap mampu memanfaatkan karakteristik pembelajaran yang berbeda dari kedua model [31]. Selain itu, metode ini tidak memerlukan proses pelatihan tambahan sebagaimana yang umum dijumpai pada pendekatan *stacking*.

Prediksi akhir dari model *hybrid ensemble* secara umum dapat dinyatakan sebagai kombinasi dari prediksi *Random Forest* dan *XGBoost* menggunakan bobot yang telah ditentukan. Bobot yang lebih besar menunjukkan kontribusi yang lebih besar terhadap hasil prediksi akhir. Dengan demikian, perubahan nilai bobot dapat memengaruhi karakteristik prediksi yang dihasilkan oleh model *hybrid*. Pendekatan serupa telah digunakan dalam penelitian sebelumnya yang mengombinasikan *Random Forest* dan *XGBoost* untuk meningkatkan performa prediksi pada berbagai permasalahan regresi [19].

2.4 Evaluation Metrics

Kualitas prediksi yang dihasilkan oleh model *Machine Learning* dapat diukur menggunakan metrik evaluasi. Pemilihan metrik evaluasi bergantung pada jenis permasalahan yang dihadapi [32]. Karena penelitian ini berfokus pada permasalahan regresi, metrik evaluasi yang digunakan juga merupakan metrik regresi. Metrik yang digunakan dalam penelitian ini adalah *Mean Absolute Error* (MAE), *Mean Squared Error* (MSE), *Root Mean Squared Error* (RMSE), dan *R-Squared* (R^2).

2.4.1 Mean Absolute Error

Salah satu metrik evaluasi yang paling umum digunakan dalam tugas regresi adalah *Mean Absolute Error* (MAE). *Mean Absolute Error* (MAE) merupakan salah satu metrik evaluasi yang paling umum digunakan dalam tugas regresi. Perhitungan MAE dilakukan dengan mengambil nilai absolut dari selisih antara hasil prediksi dan nilai sebenarnya, sehingga seluruh nilai kesalahan menjadi positif [33].

2.4.2 Mean Squared Error

Selain MAE, *Mean Squared Error* (MSE) juga sering digunakan sebagai metrik untuk mengevaluasi performa model prediksi. Metrik ini mengukur rata-rata kuadrat selisih antara nilai yang diprediksi dan nilai aktual [34]. Jika MAE menggunakan nilai absolut dari kesalahan prediksi, MSE menghitung kesalahan dengan mengkuadratkan setiap selisih yang terjadi. Akibatnya, kesalahan dengan nilai yang lebih besar akan memperoleh bobot yang lebih tinggi dalam perhitungan akhir. Karakteristik tersebut membuat MSE sangat peka terhadap keberadaan *outlier*, sehingga metrik ini banyak diterapkan pada situasi yang menuntut pengurangan kesalahan besar secara lebih optimal.

2.4.3 Root Mean Squared Error

Karena MSE dinyatakan dalam satuan kuadrat dari variabel target, interpretasinya bisa sulit dipahami. Untuk mengatasi hal tersebut, digunakan *Root Mean Squared Error* (RMSE), yaitu akar kuadrat dari MSE yang mengembalikan nilai kesalahan ke satuan asli variabel target sehingga lebih mudah untuk diinterpretasikan. Seperti MSE, RMSE juga sensitif terhadap nilai *outlier* karena

memberikan penalti yang lebih besar terhadap kesalahan dengan nilai besar. Meskipun demikian, RMSE mampu memberikan keseimbangan antara sensitivitas terhadap kesalahan besar dan kemudahan interpretasi [35].

2.4.4 R-Squared

Selain metrik berbasis kesalahan, *R-squared* (R^2) atau koefisien determinasi juga banyak digunakan untuk menilai kinerja suatu model [35] [36]. Metrik ini menunjukkan sejauh mana variasi pada variabel dependen dapat diterangkan oleh variabel independen yang digunakan dalam model. Secara umum, nilai R^2 berada dalam rentang 0 sampai 1, dengan nilai yang semakin mendekati 1 menandakan kemampuan model yang semakin baik dalam merepresentasikan variasi data. Meskipun demikian, dalam beberapa kasus, nilai R^2 dapat menjadi negatif apabila performa model lebih buruk dibandingkan prediksi yang hanya menggunakan nilai rata-rata data. Karena dapat menggambarkan tingkat kecocokan model secara keseluruhan, metrik ini sering dimanfaatkan sebagai pelengkap ukuran kesalahan seperti MAE, MSE, dan RMSE. Persamaan untuk menghitung nilai *R-squared* ditunjukkan sebagai berikut.

2.5 Teknik Encoding Data Kategorikal

Fitur kategorikal berisi informasi dalam bentuk label atau kategori, seperti nama kota dan distrik pada dataset harga rumah. Karena sebagian besar algoritma *machine learning* memerlukan masukan berupa nilai numerik, data kategorikal perlu dikonversi terlebih dahulu ke dalam bentuk angka. Proses konversi tersebut dikenal sebagai *encoding* dan bertujuan agar informasi yang terkandung dalam kategori tetap dapat dimanfaatkan selama proses pelatihan model [37].

Beberapa teknik *encoding* yang umum digunakan antara lain *label encoding* dan *one-hot encoding*. *Label encoding* mengubah setiap kategori menjadi nilai bilangan bulat, namun metode ini dapat menimbulkan asumsi hubungan ordinal yang tidak sesuai dengan sifat data kategorikal. Sementara itu, *one-hot encoding* merepresentasikan setiap kategori sebagai vektor biner, sehingga tidak menimbulkan bias ordinal. Meskipun demikian, metode ini dapat meningkatkan dimensi data secara signifikan apabila jumlah kategori cukup banyak, yang pada akhirnya dapat mempengaruhi efisiensi model [37].

Sebagai alternatif, digunakan teknik *target encoding*, yaitu metode yang

mengubah setiap kategori menjadi nilai rata-rata dari variabel target pada kategori tersebut. Pendekatan ini memungkinkan model untuk menangkap hubungan antara kategori dengan variabel target secara lebih informatif dibandingkan metode *encoding* sederhana. Secara matematis, *target encoding* sederhana dapat dinyatakan sebagai yang ditampilkan di 2.2.

$$\hat{x}_l = \frac{\sum_{i:x_i=l} y_i}{N_l} \tag{2.2}$$

di mana $\hat{x}_l$ adalah nilai *encoding* untuk kategori l, y_i adalah nilai target, dan N_l adalah jumlah data pada kategori tersebut [38]. Meskipun efektif, metode ini rentan terhadap *overfitting*, terutama pada kategori dengan jumlah data yang sedikit.

Untuk mengatasi permasalahan tersebut, digunakan pendekatan *smoothed target encoding*, yaitu teknik regularisasi yang mengombinasikan rata-rata target pada kategori dengan rata-rata global seluruh data. Pendekatan ini bertujuan untuk menyeimbangkan pengaruh antara informasi lokal dari kategori tertentu dan informasi global dari keseluruhan dataset. Secara matematis, *smoothed target encoding* dapat dirumuskan sebagai 2.3.

$$\tilde{x}_l = \frac{n_l \cdot \hat{x}_l + k \cdot \hat{x}}{n_l + k} \tag{2.3}$$

di mana $\tilde{x}_l$ adalah nilai *encoding* yang telah dihaluskan, n_l adalah jumlah data pada kategori l, $\hat{x}_l$ adalah rata-rata target pada kategori tersebut, $\hat{x}$ adalah rata-rata global, dan k adalah parameter *smoothing* yang mengontrol tingkat regularisasi [37]. Nilai k yang lebih besar akan membuat hasil *encoding* lebih mendekati rata-rata global, terutama untuk kategori dengan jumlah data kecil. Sebaliknya, jika jumlah data dalam suatu kategori besar, maka nilai *encoding* akan lebih dipengaruhi oleh rata-rata kategori tersebut.

Pemilihan *smoothed target encoding* dalam penelitian ini didasarkan pada karakteristik dataset yang memiliki fitur kategorikal berupa *city* dan *district* dengan jumlah kategori yang cukup banyak. Dibandingkan *one-hot encoding*, metode ini lebih efisien karena tidak meningkatkan dimensi data secara signifikan. Selain itu, penggunaan parameter *smoothing* membantu mengurangi risiko *overfitting* pada kategori yang memiliki jumlah data sedikit sehingga kemampuan generalisasi model diharapkan menjadi lebih baik dalam memprediksi harga rumah. Dengan demikian, metode ini diharapkan dapat meningkatkan kemampuan generalisasi

model dalam memprediksi harga rumah [38].

2.6 GridSearchCV

GridSearchCV merupakan salah satu metode *hyperparameter tuning* yang tersedia pada pustaka *Scikit-Learn*. Metode ini bekerja dengan menguji seluruh kombinasi parameter yang telah ditentukan dalam suatu *parameter grid* secara sistematis untuk menemukan kombinasi parameter yang memberikan performa terbaik. Setiap kombinasi parameter dievaluasi menggunakan *cross-validation*, kemudian kombinasi dengan hasil evaluasi terbaik dipilih sebagai parameter optimal untuk model yang akan digunakan. GridSearchCV menggunakan pendekatan *exhaustive search*, yaitu mengevaluasi seluruh kombinasi parameter yang tersedia dalam ruang pencarian [39]. Meskipun metode ini mampu menghasilkan konfigurasi parameter yang optimal, kebutuhan komputasinya akan meningkat seiring bertambahnya jumlah parameter dan nilai yang diuji. Oleh karena itu, pemilihan ruang pencarian parameter perlu dilakukan secara cermat agar diperoleh keseimbangan antara kualitas hasil dan waktu komputasi.