BAB II

LANDASAN TEORI

2.1 Penelitian Terdahulu

Berikut ini merupakan tabel yang merangkum beberapa penelitian terkait. Tabel ini mencakup berbagai algoritma yang digunakan, hasil yang diperoleh, serta kesimpulan yang dapat diambil dari masing-masing studi tersebut. Penelitian ini memberikan gambaran mengenai efektivitas dan tantangan yang dihadapi dalam penerapan model-model seperti LSTM, GRU, dan XGBoost dalam konteks prediksi harga Bitcoin dan Ethereum.

No.	Penulis / Tahun	Fitur	Periode data	Frekuensi	Variabel Makro	Target prediksi	Horizon prediksi	Skema pembagian data	Model yang digunakan	Model terbaik & Hasil evaluasi	Keterbatasan
1.	Mallepudi Shamitha / 2025 [17]	*Historical market data* (OHLCV), *market cap*, dan *technical indicators* (MA, EMA, RSI, MACD)	Mar 2020 – Okt 2024	Harian	-	Harga penutupan Bitcoin	10 hari ke depan	80:20 *train – test split*	LSTM, ANN, GRU	LSTM, RMSE: 29130.18	Hanya satu aset, tidak ada ETH, tidak ada XGBoost, tidak ada makro-finansial.

2.	Wen, Ling, *et al* / 2023 [13]	*Historical market data* (OHLCV)	1 Jan 2018 – 31 Des 2022	Harian	-	Harga penutupan Bitcoin	1 hari ke depan (1, 3, 7 *windows length*)	80:20 *train – test split*	LSTM, CNNs	LSTM, RMSE: 1203.97 MAE: 875.06	Hanya satu aset, tidak ada ETH, tidak ada XGBoost, tidak ada makro-finansial.
3.	Kaur, Uppal, *et al* / 2025 [10]	*Historical market data* (*Close price*)	17 Agt 2017 – 10 Jul 2024	Harian	-	Harga penutupan Bitcoin dan Ethereum	1 hari ke depan	80:20 *train – test split*	LSTM, GRU	GRU, RMSE (BTC): 0.03258 RMSE (ETH): 0.06749	Tidak ada XGBoost, tidak ada makro-finansial.
4.	Li / 2023 [11]	*Historical market data* (*Close price*)	1 Jan 2020 – 1 Agt 2022	Harian	-	Harga penutupan Bitcoin dan Ethereum	1 hari ke depan	90:10 *train – test split*	LSTM	LSTM, MAPE (BTC): 0.0299 MAPE (ETH): 0.0432	Tidak ada GRU, XGBoost, tidak ada makro-finansial.
5.	Nair, Marie, *et al* / 2023 [14]	*Historical market data* (OHLCV)	17 Sep 2014 – 01 Feb 2022	Harian	-	Harga penutupan Bitcoin	1 hari ke depan	80:20 *train – test split*	RNN, LSTM, GRU, Bi-LSTM, Conv1D	LSTM, RMSE: 1978.68 MAE: 1537.24	Hanya satu aset, tidak ada ETH, tidak ada XGBoost, tidak ada makro-finansial.

6.	Shahbazi, Byun/ 2022 [18]	*Historical market data* (*daily exchange rate*, *high*, *low*, *open*, *close*, *quote volume, average weighted*), *macro indicators* (*crude oil*, DOW30, SSE, VIX, CNY/USD, EUR/USD), dan *external factors* (*on-chain metrics, Google Trends*)	2018 – 2021	Harian	*Macroeconomic, global currency ratio, public recognition, blockchain information*	*Exchange Rate* ETH	7, 30, 90 hari ke depan	80:20 *train – test split*	XGBoost	XGBoost, RMSE: 0.765 MAE: 0.608, MAPE: 0.005	Hanya satu aset, tidak ada ETH, tidak ada LSTM dan GRU, tidak ada makro-finansial.
7.	Hafid, Ebrahim, *et al* / 2024 [19]	*Historical market data* (OHLCV) dan *technical indicators* (EMA, MACD, RSI, MOM, PROC,	1 Feb 2021 – 1 Feb 2022	15 menit	-	Harga penutupan Bitcoin	15 menit ke depan	80:20 *train – test split*	XGBoost	XGBoost, RMSE: 59.9504	Hanya satu aset, tidak ada ETH, tidak ada LSTM dan GRU, tidak ada makro-finansial.

		stochastic oscillator)									
8.	Gbadebo / 2025 [15]	*Historical market data* (OHLCV)	1 Jan 2020 – 31 Des 2024	Harian	-	Harga penutupan Bitcoin dan Ethereum	1 hari ke depan	80:20 *train – test split*	LSTM, GRU, LSTM+GRU, SGD regression	GRU, RMSE (BTC): 0,019993 RMSE (ETH): 0,026268	Tidak ada XGBoost dan tidak ada fitur makro-finansial.
9.	Hadi, Ramli, *et al* / 2023 [12]	*Historical market data* (*Close price*)	2018 – 2022	Harian	-	Harga penutupan Bitcoin dan Ethereum	1 hari ke depan	80:20 *train – test split*	LSTM, GRU	GRU, RMSE (BTC): 654.0614 RMSE (ETH): 47.3271	Tidak ada XGBoost dan tidak ada fitur makro-finansial.
10.	Rodrigues, Machado/ 2025 [16]	*Historical market data* (OHLCV)	12 Mei 2024 – 11 Jun 2024	1 menit	-	Harga penutupan Bitcoin dan Ethereum	1 jam ke depan	80:10:10 *train – test – validation split*	LSTM, GRU, ARIMA	GRU, RMSE (BTC): 0.19 RMSE (ETH): 0.59	Tidak ada XGBoost dan tidak ada fitur makro-finansial.

Tabel 2.1 Penelitian Terdahulu.

Berdasarkan Tabel 2.1, penelitian terdahulu menunjukkan bahwa prediksi harga *cryptocurrency* masih didominasi oleh penggunaan data historis harian, terutama *close price* maupun *historical market data* (OHLCV). Pada studi yang sama-sama memakai data harian berbasis harga historis, kecenderungan yang muncul adalah GRU sering memberikan hasil lebih baik daripada LSTM, sebagaimana terlihat pada penelitian Hadi dkk [12], Kaur dkk [10], dan Gbadebo [15]. Namun demikian, hasil terbaik tidak selalu sama pada semua studi, karena penelitian Wen dkk [13] justru menunjukkan LSTM sebagai model yang paling sesuai untuk Bitcoin ketika dibandingkan dengan CNN pada skenario *window* 1, 3, dan 7 hari.

Pola serupa juga terlihat pada penelitian Nair dkk [14] yang membandingkan RNN, LSTM, GRU, Bi-LSTM, dan Conv1D pada prediksi harga Bitcoin berbasis *close price*, lalu menghasilkan LSTM sebagai model terbaik, serta pada penelitian Mallepudi Shamitha [17] yang menggunakan OHLCV, *market cap*, serta indikator teknikal seperti MA, EMA, RSI, dan MACD, dan juga menyimpulkan bahwa LSTM efektif untuk menangkap pola temporal harga Bitcoin, serta pada penelitian Li [11] yang menggunakan LSTM pada Bitcoin dan Ethereum, lalu menunjukkan bahwa Bitcoin merupakan aset yang paling sesuai untuk LSTM daripada Ethereum. Sementara itu, penelitian Rodrigues dan Machado [16] memperlihatkan bahwa pada pengaturan *high frequency* dengan data 1 menit dan *horizon* prediksi 1 jam ke depan, GRU menjadi model paling efektif. Di sisi lain, ketika ruang fitur diperkaya melalui indikator teknikal atau variabel tambahan, pendekatan non-sekuensial juga dapat menunjukkan performa yang kuat, seperti pada penelitian Hafid dkk [19] yang menggunakan XGBoost dengan indikator teknikal, serta Shahbazi dan Byun [18] yang menggabungkan XGBoost dengan faktor makroekonomi, *global currency ratio*, *public recognition*, dan *blockchain information*.

Selain itu, Tabel 2.1 juga memperlihatkan bahwa penelitian terdahulu belum sepenuhnya memberikan kerangka komparasi yang setara antara Bitcoin dan Ethereum dengan kombinasi model dan skenario fitur yang seragam. Sebagian penelitian terdahulu hanya berfokus pada prediksi satu aset kripto tertentu, sedangkan beberapa penelitian lain melakukan perbandingan performa model pada

lebih dari satu aset kripto. Namun, sebagian besar penelitian tersebut masih didominasi oleh penggunaan data historis kripto tanpa integrasi indikator makro-finansial sebagai variabel tambahan. Penelitian terdahulu juga menunjukkan variasi yang cukup besar pada aset yang diteliti, *horizon* prediksi, frekuensi data, serta model yang digunakan. Oleh karena itu, penelitian ini memanfaatkan ruang ini untuk melakukan penelitian dalam membandingkan LSTM, GRU, dan XGBoost pada Bitcoin dan Ethereum dalam satu kerangka eksperimen yang seragam, dengan dua skenario fitur yang dibedakan secara jelas, yaitu data historis kripto beserta fitur turunannya dan data historis kripto yang diperkaya indikator makro-finansial. Kerangka ini membuat perbedaan performa model dapat dianalisis secara lebih terarah, baik dari sisi karakteristik aset maupun dari sisi kontribusi fitur.

2.2 Teori yang berkaitan

Bagian ini memuat teori yang menjadi landasan konseptual dalam penelitian. Teori yang dibahas meliputi konsep dasar aset kripto, pendekatan prediksi deret waktu, *machine learning* dan *deep learning*, indikator makro-finansial, serta metrik evaluasi model. Teori tersebut digunakan untuk mendukung perancangan model, proses analisis, dan interpretasi hasil penelitian.

2.2.1 Cryptocurrency

Cryptocurrency adalah aset digital yang dirancang untuk bekerja sebagai media pertukaran menggunakan kriptografi untuk mengamankan dan memverifikasi transaksi serta untuk mengontrol pembuatan unit mata uang baru. *Cryptocurrency* bersifat desentralisasi dengan memanfaatkan teknologi *blockchain*. *Blockchain* adalah kumpulan data yang saling terhubung sehingga membentuk rantai, dimana teknologi ini dapat mencatat transaksi yang saling terhubung menggunakan kode unik di dalamnya dan tidak dapat diubah [20]. Setiap blok dalam rantai mencakup informasi transaksi, *timestamp*, dan *hash* kriptografis dari blok sebelumnya [21]. *Blockchain* memungkinkan transaksi *peer-to-peer* dengan cara yang aman dan dapat diverifikasi tanpa pihak yang terpusat [22]. *Cryptocurrency* juga memiliki karakteristik seperti transparansi pencatatan transaksi, dimana data transaksi dapat diaudit di jaringan, lalu memiliki sifat

pseudonim, dimana identitas pengguna direpresentasikan oleh alamat kriptografis, serta aturan pasokan atau suplai yang ditentukan oleh protokol.

2.2.2 Bitcoin

Bitcoin adalah mata uang kripto yang terdesentralisasi yang didirikan berdasarkan rencana jaringan *peer-to-peer* [23]. Bitcoin menggunakan mekanisme konsensus cryptographic *proof of work* (PoW) yang memungkinkan para pengguna (*peer*) dalam jaringan *peer-to-peer* (P2P) untuk membuat, memverifikasi, dan mencatat transaksi secara kolektif tanpa memerlukan otoritas pusat [24]. PoW mendefinisikan teka-teki komputasi sebagai fungsi dari setiap rantai blok [25]. Blok baru akan diterima jaringan jika hasil perhitungan tersebut memenuhi tingkat kesulitan yang ditetapkan oleh aturan blockchain saat itu. Transaksi yang telah diverifikasi akan dikelompokkan ke dalam blok dan disusun menjadi *blockchain*, yaitu rangkaian blok yang saling terhubung melalui *hash* sehingga perubahan pada satu blok akan mempengaruhi blok berikutnya dan membuat manipulasi data menjadi sulit. Identitas pengguna pada jaringan Bitcoin bersifat pseudonim karena direpresentasikan oleh alamat kriptografis, sehingga transaksi dapat ditelusuri pada jaringan tanpa secara langsung mengungkap identitas pemilik alamat. Pasokan Bitcoin diatur oleh protokol dan bersifat terbatas hingga mencapai yaitu 21 juta BTC [26], dengan mekanisme penerbitan melalui *block reward* serta penyesuaian *difficulty* secara berkala agar laju pembentukan blok relatif stabil.

2.2.3 Ethereum

Ethereum adalah *platform* aplikasi terdesentralisasi yang dibangun di atas teknologi *blockchain* [27]. Ethereum tidak hanya berfungsi sebagai mata uang digital, tetapi juga sebagai *platform* yang memungkinkan para pengembang untuk membangun dan mengembangkan aplikasi terdesentralisasi melalui *smart contracts* [28]. *Smart contract* adalah kode pada *blockchain* yang dieksekusi secara otomatis ketika kondisi tertentu terpenuhi [29]. *Smart contract* memastikan bahwa semua pihak mematuhi aturan tanpa perlu otoritas tengah. Aturan ini disimpan dengan cara yang terdesentralisasi, yang memungkinkan data untuk ditulis atau diambil di dalam *blockchain*. *Smart contract* ini akan dieksekusi secara otomatis berdasarkan kondisi sesuai perjanjian yang berlaku [30]. Fitur ini bukan hanya

meningkatkan transparansi dan keamanan tetapi juga menambah efisiensi dan mengurangi biaya transaksi karena tidak memerlukan perantara. Ethereum memiliki mata uang yaitu Ether (ETH) yang digunakan untuk membayar operasi operasi dan transaksi yang dilakukan di jaringan Ethereum [31]. Ether adalah *cryptocurrency* yang digunakan sebagai bahan bakar atau pembayaran untuk menjalankan transaksi dan aplikasi di jaringan Ethereum. Peran ethereum sebagai infrastruktur aplikasi dan perkembangan ekosistem membuat permintaan terhadap ethereum bukan hanya faktor spekulasi harga semata.

2.2.4 Forecasting

Forecasting adalah sebuah teknik yang berfokus pada identifikasi pola atau tren berdasarkan data masa lalu untuk memproyeksikan kejadian di masa yang akan datang. Dasar teori dari *forecasting* berangkat dari premis bahwa pengetahuan masa kini dan masa lalu dapat digunakan untuk meramalkan kondisi di masa depan [32]. Metode ini mengasumsikan bahwa pola masa lalu akan terus berlanjut di masa depan, dengan beberapa penyesuaian untuk faktor – faktor eksternal yang dapat mempengaruhi hasil. *Forecasting* berfungsi untuk membantu mengambil keputusan yang lebih baik di masa kini dengan mempertimbangkan kemungkinan yang akan datang [33]. *Forecasting* terdiri dari dua metode yaitu kualitatif dan kuantitatif.

1. Metode Kualitatif

   Metode kualitatif dalam *forecasting* mengandalkan penilaian subjektif, wawasan, dan pengalaman para ahli atau individu yang terlibat dalam proses peramalan.

2. Metode Kuantitatif

   Metode kuantitatif dalam *forecasting* menggunakan data historis yang tersedia untuk membangun model matematis dan statistik guna meramalkan nilai di masa depan. Metode ini mengandalkan data numerik yang objektif dan dapat diuji untuk menghasilkan prediksi yang lebih terukur

Dalam penelitian ini, pendekatan forecasting yang digunakan adalah *forecasting* kuantitatif berbasis deret waktu karena prediksi harga Bitcoin dan

Ethereum dibangun dari data historis numerik dan variabel makro-finansial yang diamati secara terstruktur.

2.2.5 Machine Learning

Machine Learning adalah cabang dari kecerdasan buatan (AI) yang menggunakan data untuk melatih komputer melakukan tugas-tugas tertentu. *Machine Learning* digunakan untuk mengajari mesin cara menangani data dengan lebih efisien. Berbeda dengan pemrograman tradisional yang memerlukan aturan yang diprogram secara eksplisit, *machine learning* menggunakan algoritma untuk secara otomatis membangun kumpulan aturan tersebut. *Machine learning* mengajarkan mesin untuk belajar dari pengalaman masa lalunya, menggunakan teknik komputasi dan secara adaptif meningkatkan kinerjanya dengan lebih banyak contoh [34]. Hal ini dapat meningkatkan efisiensi dan kontribusi pada pengambilan keputusan. Kemampuannya untuk menganalisis kumpulan data besar serta mengidentifikasi pola dan tren yang tidak terlihat melalui analisis manual membuat *machine learning* relevan untuk digunakan. Hal ini karena pergerakan harga *cryptocurrency* dipengaruhi oleh hubungan antarvariabel yang kompleks dan tidak selalu linear.

2.2.6 Deep Learning

Deep learning adalah cabang dari *machine learning* yang bertujuan untuk mencapai kecerdasan buatan [35]. *Deep learning* memanfaatkan arsitektur – arsitektur model yang terinspirasi dari struktur dan fungsi otak manusia, yaitu jaringan saraf tiruan, untuk mempelajari pola dan fitur yang lebih kompleks dari data. *Deep learning* mengandalkan penggunaan jaringan saraf tiruan yang terdiri dari berbagai lapisan, yang dikenal sebagai jaringan saraf dalam (*deep neural networks*). Setiap lapisan dalam struktur ini berfungsi untuk mengekstraksi informasi atau pola dari data secara progresif, dimulai dengan fitur yang sederhana dan berkembang menuju fitur yang lebih kompleks seiring dengan bertambahnya kedalaman jaringan. Proses pelatihan dilakukan dengan algoritma *backpropagation*, dimana dapat belajar dengan cepat dengan mengkomputasi pembaruan sinapsis menggunakan koneksi umpan balik untuk mengirimkan sinyal kesalahan [36]. Hal ini memungkinkan model untuk memperbaiki kesalahan dan

mengoptimalkan parameter dengan mengembalikan kesalahan yang terjadi pada *output* ke seluruh jaringan, sehingga bobot dan parameter lainnya dapat diperbarui untuk meningkatkan akurasi prediksi. Keunggulan utama *deep learning* terletak pada kemampuannya untuk mengelola data besar dan tidak terstruktur tanpa memerlukan definisi fitur manual, sehingga dalam penelitian ini, *deep learning* digunakan karena pergerakan harga kripto bersifat *non-linear* dan dinamis, sehingga memerlukan model yang mampu mempelajari pola kompleks dari data historis dan variabel eksternal.

2.2.7 US Dollar Index

US Dollar Index (DXY) adalah indeks yang mengukur nilai tukar dolar Amerika Serikat relatif terhadap enam mata uang asing dari enam negara yang berbeda [37]. *US Dollar Index* dapat memberikan gambaran komprehensif tentang kekuatan dolar AS di pasar global. Keunggulan dolar AS sebagai mata uang cadangan global dan alat utama dalam perdagangan internasional membuat pemahaman terhadap nilai tukarnya sangat penting bagi pelaku ekonomi dan kebijakan. Dalam konteks *cryptocurrency*, kekuatan atau kelemahan dolar AS sering kali berkorelasi dengan minat investor terhadap aset berisiko seperti *cryptocurrency*. *US Dollar Index* (DXY) cenderung memiliki korelasi negatif dengan *cryptocurrency* (Bitcoin dan Ethereum) [38]. Korelasi negatif ini terjadi karena penguatan DXY sering mencerminkan menguatnya dolar AS dan pengetatan likuiditas global, sehingga aset berisiko seperti *cryptocurrency* menjadi kurang menarik bagi investor. Sebaliknya, ketika DXY melemah, minat terhadap aset alternatif seperti Bitcoin dan Ethereum cenderung meningkat.

2.2.8 Gold (XAU)

Gold merupakan komoditas logam mulia yang secara luas dipandang sebagai aset *store of value*, lindung nilai terhadap inflasi (*inflation hedge*), dan *safe haven* [39], terutama ketika terjadi ketidakpastian ekonomi, inflasi, atau gejolak pasar keuangan. Secara teori, emas cenderung diminati saat investor mengurangi eksposur pada aset berisiko dan mencari instrumen yang dianggap lebih defensif, sehingga pergerakannya sering digunakan sebagai indikator perubahan preferensi risiko (*risk sentiment*) [40]. Selain itu, harga emas juga dapat dipengaruhi oleh

indikator makro-finansial, risiko geopolitik, dan kebijakan moneter [41]. Emas digunakan sebagai indikator makro-finansial untuk merepresentasikan kondisi pasar defensif dan tingkat ketidakpastian dinamika pergerakan pasar.

2.2.9 S&P 500

S&P 500 adalah indeks pasar saham yang sering digunakan sebagai barometer kondisi pasar ekuitas Amerika Serikat dan proksi sentimen investor terhadap aset berisiko (*risk appetite*). Secara umum, penguatan indeks mencerminkan optimisme pasar (*risk-on*) dan meningkatnya minat pada aset berisiko, sedangkan pelemahan indeks sering diasosiasikan dengan sikap defensif (*risk-off*) akibat meningkatnya ketidakpastian atau pengetatan kondisi keuangan. S&P 500 relevan digunakan sebagai indikator makro untuk menangkap perubahan kondisi pasar yang lebih luas. Dalam penelitian ini, S&P 500 dimanfaatkan sebagai fitur tambahan untuk membantu model memahami hubungan antara dinamika pasar ekuitas dan pergerakan harga Bitcoin maupun Ethereum.

2.2.10 Moving Average

Moving Average (MA) merupakan metode peramalan sederhana yang menghitung rata-rata nilai dari sejumlah periode sebelumnya untuk menghasilkan prediksi pada periode berikutnya. Metode ini banyak digunakan dalam analisis deret waktu dan pasar keuangan untuk mengidentifikasi tren serta mengurangi fluktuasi jangka pendek pada data. Dalam penelitian ini, *Moving Average* 7 hari juga digunakan sebagai *baseline forecasting* untuk membandingkan performa model *machine learning* dan *deep learning*. Pemilihan periode 7 hari dilakukan karena data yang digunakan bersifat harian, sehingga rata-rata satu minggu dianggap cukup untuk merepresentasikan tren jangka pendek tanpa terlalu terpengaruh oleh volatilitas harian. Penggunaan *baseline* bertujuan untuk mengevaluasi apakah model yang dikembangkan mampu memberikan peningkatan akurasi dibandingkan metode peramalan sederhana.

2.2.11 Feature Engineering

Feature engineering adalah serangkaian proses untuk mentransformasi fitur, bertujuan untuk menemukan fitur-fitur baru yang berharga yang dapat

mencerminkan aspek wawasan dari data [42]. Proses ini intinya membentuk fitur baru dari data mentah yang bertujuan untuk meningkatkan kemampuan model dalam menangkap pola. Dalam penelitian ini, *feature engineering* digunakan untuk membentuk representasi data yang lebih informatif melalui *return*, volatilitas, indikator teknikal seperti *Moving Average* (MA) untuk mengidentifikasi tren harga, *Relative Strength Index* (RSI) mengukur kondisi *overbought* dan *oversold*, dan *Moving Average Convergence Divergence* (MACD) menangkap arah serta momentum pergerakan harga. Selain itu, terdapat fitur *lag* yang dibentuk dari nilai pada periode sebelumnya agar model dapat memanfaatkan informasi historis untuk memahami ketergantungan waktu dan pola pergerakan harga jangka pendek secara lebih efektif, sehingga model dapat menangkap pola harga dan konteks historis jangka pendek secara lebih efektif.

2.2.12 Lag Feature

Lag feature merupakan fitur yang dibentuk dari nilai suatu variabel pada periode sebelumnya dan digunakan sebagai masukan (*input*) dalam proses prediksi data deret waktu (*time series*). Konsep ini didasarkan pada asumsi bahwa nilai suatu variabel pada masa lalu memiliki pengaruh terhadap nilai pada periode berikutnya, sehingga informasi historis dapat dimanfaatkan untuk membantu model mengenali pola, tren, maupun ketergantungan temporal dalam data. Secara umum, *lag* ke-k merepresentasikan nilai suatu variabel pada k periode sebelumnya, sehingga semakin banyak *lag* yang digunakan maka semakin banyak informasi historis yang tersedia bagi model untuk dipelajari. Dalam konteks prediksi harga aset keuangan seperti Bitcoin dan Ethereum, penggunaan *lag feature* memungkinkan model menangkap pola pergerakan harga jangka pendek, momentum pasar, serta hubungan antarperiode yang tidak selalu terlihat dari nilai saat ini saja.

2.2.13 Multivariate Time Series Forecasting

Multivariate time series forecasting merupakan metode peramalan deret waktu yang menggunakan lebih dari satu variabel yang diamati secara bersamaan dari waktu ke waktu untuk memprediksi nilai pada periode mendatang [43]. Berbeda dengan *univariate time series* yang hanya memanfaatkan satu seri data, pendekatan multivariat memungkinkan model menangkap hubungan, keterkaitan,

dan pengaruh antar variabel yang dapat berubah secara dinamis sepanjang waktu. Dalam konteks pasar keuangan, pendekatan ini relevan karena pergerakan suatu aset tidak selalu ditentukan oleh riwayat harganya sendiri, tetapi juga dapat dipengaruhi oleh variabel lain yang berkaitan. Pendekatan ini sesuai dengan desain penelitian ini karena prediksi harga Bitcoin dan Ethereum tidak hanya dibangun dari riwayat harga aset itu sendiri, tetapi juga dari variabel makro-finansial yang diasumsikan ikut mempengaruhi pergerakan harga.

2.2.14 Forward-fill

Forward-fill, yang juga dikenal sebagai *last observation carried forward* (LOCF), merupakan metode imputasi pada data deret waktu yang digunakan untuk mengisi nilai yang hilang dengan meneruskan nilai observasi terakhir yang tersedia ke periode setelahnya [44]. Pendekatan ini digunakan ketika data tersusun secara kronologis dan nilai terakhir yang tersedia masih dianggap merepresentasikan kondisi hingga muncul observasi baru berikutnya. Dalam penelitian ini, *forward-fill* digunakan untuk menyelaraskan data *Gold*, *US Dollar Index* (DXY), dan S&P 500 dengan kalender harian aset kripto agar seluruh variabel berada dalam kerangka waktu yang konsisten.

2.2.15 Log Transformation

Log transformation merupakan teknik transformasi data yang mengubah nilai ke dalam skala logaritmik agar pola data menjadi lebih stabil dan lebih mudah dimodelkan dalam analisis deret waktu. Transformasi ini digunakan untuk mengurangi variabilitas data serta membantu mengatasi distribusi yang cenderung miring (*skewed*) ke bentuk yang lebih normal [45]. Dalam penelitian ini, *log transformation* diterapkan pada target pemodelan agar proses pembelajaran model menjadi lebih stabil karena perubahan harga tidak lagi dilihat hanya sebagai selisih absolut, tetapi lebih mencerminkan perubahan relatif atau proporsional.

$$\boldsymbol{y_t = ln(P_{t+1})}$$

Rumus 1. *Log Transformation*

Keterangan:

$\boldsymbol{y_t}$ = target pada waktu ke-t

$\boldsymbol{P_{t+1}}$ = harga penutupan pada hari berikutnya

Setelah prediksi dihasilkan pada domain logaritmik, nilai tersebut dikonversi kembali ke domain harga asli melalui *inverse transformation* menggunakan fungsi eksponensial agar hasil prediksi dapat diinterpretasikan secara langsung dalam satuan harga.

$$\widehat{\boldsymbol{P}}_{\boldsymbol{t+1}} = \boldsymbol{e}^{\widehat{\boldsymbol{y}}_{\boldsymbol{t}}}$$

Rumus 2. *Inverse Log Transformation*

Keterangan:

$\widehat{\boldsymbol{P}}_{\boldsymbol{t+1}}$ = hasil prediksi harga pada domain asli

$\widehat{\boldsymbol{y}}_{\boldsymbol{t}}$ = hasil prediksi pada domain logaritmik

2.2.16 Hyperparameter Tuning

Hyperparameter tuning adalah proses penentuan konfigurasi *hyperparameter* yang paling sesuai untuk meningkatkan performa prediksi model. Proses ini sangat penting untuk mengoptimalkan kinerja dan kemampuan generalisasi model [46]. Dalam penelitian ini, proses tuning mencakup penentuan *sequence window* pada model sekuensial serta *hyperparameter* pada LSTM, GRU, dan XGBoost karena parameter-parameter tersebut mempengaruhi kemampuan model dalam mempelajari pola data, kestabilan pelatihan, dan kemampuan generalisasi terhadap data baru. Oleh karena itu, *hyperparameter tuning* menjadi tahap penting untuk memperoleh konfigurasi model yang optimal.

2.2.17 Analisis Fundamental dan Teknikal dalam Trading

Dalam dunia *trading*, terdapat dua pendekatan utama yang umum digunakan untuk menganalisis pergerakan harga aset, yaitu analisis fundamental dan analisis teknikal. Analisis fundamental berfokus pada penilaian nilai suatu aset berdasarkan faktor-faktor ekonomi, keuangan, dan kondisi eksternal yang mempengaruhi pasar, seperti kebijakan moneter, kondisi makroekonomi, sentimen pasar, dan perkembangan industri terkait. Sementara itu, analisis teknikal berfokus pada pergerakan harga historis, volume perdagangan, pola grafik, serta indikator teknikal untuk mengidentifikasi tren dan memperkirakan arah harga di masa depan.

2.2.18 Shapley Additive Explanations

SHAP (SHapley Additive Explanations) merupakan metode interpretabilitas model yang digunakan untuk menjelaskan kontribusi setiap fitur terhadap hasil prediksi model *machine learning* maupun *deep learning*. Metode ini mengadopsi konsep *Shapley Value* dari teori permainan (*cooperative game theory*), di mana setiap fitur dianggap sebagai pemain yang berkontribusi terhadap hasil prediksi model. SHAP menghitung kontribusi rata-rata suatu fitur terhadap seluruh kemungkinan kombinasi fitur yang dapat terbentuk sehingga menghasilkan ukuran kontribusi yang konsisten dan dapat diinterpretasikan [47]. Nilai SHAP yang lebih besar menunjukkan bahwa suatu fitur memiliki pengaruh yang lebih besar terhadap prediksi model, sedangkan nilai yang mendekati nol menunjukkan kontribusi yang relatif kecil.

2.2.19 Wilcoxon Signed-Rank Test

Wilcoxon Signed-Rank Test merupakan metode uji statistik non-parametrik yang digunakan untuk membandingkan dua kelompok data berpasangan guna menentukan apakah terdapat perbedaan yang signifikan di antara keduanya [48]. Uji ini dikembangkan sebagai alternatif dari *Paired Sample t-Test* ketika data tidak memenuhi asumsi distribusi normal. Dalam bidang *machine learning* dan *time series forecasting*, Wilcoxon Signed-Rank Test sering digunakan untuk membandingkan performa dua model prediksi berdasarkan nilai *error* yang dihasilkan pada data pengujian yang sama. Pengujian dilakukan dengan menghitung perbedaan antara setiap pasangan observasi, memberikan peringkat terhadap nilai absolut perbedaan tersebut, kemudian menghitung statistik uji untuk memperoleh nilai p-value. Keputusan pengujian didasarkan pada tingkat signifikansi (α) yang umumnya ditetapkan sebesar 0,05. Jika nilai p-value lebih kecil dari 0,05, maka hipotesis nol ditolak dan dapat disimpulkan bahwa terdapat perbedaan performa yang signifikan secara statistik antara kedua model yang dibandingkan

2.2.20 Evaluasi Model

Evaluasi model adalah proses untuk mengukur kinerja model dalam memprediksi atau melakukan klasifikasi data. Tujuan dari evaluasi model adalah

mengetahui performa model dalam melakukan prediksi dan klasifikasi data. Evaluasi model melibatkan penggunaan berbagai metrik atau ukuran yang dapat mencerminkan keakuratan, kestabilan, Terdapat beberapa alat ukur atau metrik evaluasi yang dapat digunakan, di antaranya adalah *Mean Absolute Error* (MAE), *Root Mean Squared Error* (RMSE), dan *Mean Absolute Percentage Error* (MAPE). Perhitungan nilai MAE, RMSE, dan MAPE dapat dilakukan menggunakan rumus berikut:

1. Mean Absolute Error (MAE)

   Mean Absolute Error adalah metrik evaluasi yang mengukur rata-rata kesalahan absolut antara nilai prediksi dan nilai aktual. Secara umum, *semakin mean absolute error* (MAE) mendekati 0, maka semakin kecil kesalahan dan semakin besar akurasinya [49]. MAE memberikan informasi yang jelas terkait besaran kesalahan rata-rata model dalam unit yang sama dengan data.

$$\boldsymbol{MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|}$$

Rumus 3. *Mean Absolute Error* [50]

   Keterangan:

   n = jumlah data.

   i = urutan data.

   y_i = prediksi nilai ke-i.

   $\hat{y}_i$ = nilai aktual ke-i.

2. Root Mean Squared Error (RMSE)

   Root Mean Squared Error adalah akar kuadrat dari *Mean Squared Error* (MSE). RMSE mengukur rata-rata kesalahan prediksi dengan cara mengkuadratkan selisih antara nilai prediksi dan nilai aktual, kemudian mengambil rata-ratanya. Hasil rata-rata kuadrat *error* tersebut kemudian diakarkan, memberikan penalti lebih besar untuk kesalahan yang lebih besar. Secara umum, semakin RMSE mendekati 0, maka semakin kecil kesalahan dan semakin besar akurasinya [49].

$$\boldsymbol{RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}}$$

Rumus 4. *Root Mean Squared Error* [50]

Keterangan:

n = jumlah data.

i = urutan data.

y_i = prediksi nilai ke-i.

$\hat{y}_i$ = nilai aktual ke-i.

3. Mean Absolute Percentage Error (MAPE)

Mean Absolute Percentage Error (MAPE) adalah metrik yang mengukur kesalahan prediksi dalam bentuk persentase. MAPE dihitung dengan mengambil selisih absolut antara nilai prediksi dan nilai aktual, membaginya dengan nilai aktual, dan kemudian menghitung rata-rata dari hasil tersebut.

$$\boldsymbol{MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100}$$

Rumus 5. *Mean Absolute Percentage Error* [51]

Keterangan:

n = jumlah data.

i = urutan data.

y_i = prediksi nilai ke-i.

$\hat{y}_i$ = nilai aktual ke-i.

2.3 Framework/Algoritma yang digunakan

Bagian ini menjelaskan *framework* dan algoritma utama yang digunakan dalam penelitian. Pembahasan mencakup metodologi CRISP-DM sebagai kerangka kerja penelitian, serta algoritma LSTM, GRU, dan XGBoost sebagai model prediksi yang dibandingkan.

2.3.1 CRISP-DM

CRISP-DM merupakan metodologi pengembangan proyek *data mining* yang menyediakan kerangka kerja terstruktur untuk memfasilitasi implementasi proyek

dengan mendefinisikan tugas dan tingkat abstraksi secara hirarkis [52]. Berikut ini merupakan merupakan enam tahapan utama dari CRISP-DM.

1. Business Understanding

   Tahap ini memfokuskan pada pemahaman masalah dan tujuan yang ingin dicapai. Tujuan dari tahap ini adalah mendefinisikan tujuan *data mining*, yang melibatkan penentuan bagaimana *data mining* dapat mengatasi masalah bisnis yang dihadapi [53]. Tahap ini mencakup identifikasi konteks masalah, pemahaman terhadap kebutuhan proyek, Langkah-langkah dalam menyelesaikan masalah, dan penentuan kriteria keberhasilan.

2. Data Understanding

   Tahap ini melibatkan pengumpulan data awal, identifikasi atau eksplorasi data, mendeskripsikan data, serta memeriksa kualitas data. Tujuan dari tahap ini adalah memahami dan mengenali data, mengidentifikasi masalah kualitas data, dan menemukan *insight* awal.

3. Data Preparation

   Tahap ini merupakan tahapan untuk mempersiapkan data agar siap melanjutkan ke proses pembangunan model. Pada tahap ini, data yang telah dikumpulkan akan diproses untuk mengatasi masalah seperti data yang hilang, duplikat, atau tidak konsisten. Proses tersebut dapat meliputi, *encoding*, pembersihan data, transformasi, normalisasi data, pemilihan fitur dan target.

4. Modeling

   Tahap ini berfokus pada pembangunan model prediktif yang digunakan untuk memecahkan masalah yang telah diidentifikasi pada tahap sebelumnya. Pada tahap ini, berbagai teknik dan algoritma digunakan untuk membangun model yang akan mengatasi permasalahan yang ada dan mencapai tujuan yang telah ditetapkan. Selain itu, dalam pembangunan model, parameter-parameter spesifik harus ditetapkan, dimana dapat menggunakan teknik *grid search* untuk mencari kombinasi *hyperparameter* terbaik dengan menguji berbagai kemungkinan nilai dari parameter yang relevan secara sistematis.

5. Evaluation

   Setelah tahap pemodelan, evaluasi dilakukan untuk menilai model secara kritis melalui berbagai metrik evaluasi dan meninjau langkah-langkah yang telah diambil untuk memastikan model dengan tepat memenuhi tujuan bisnis yang ditetapkan pada tahap pertama. Tahap evaluasi adalah tahap penilaian sebelum keputusan untuk meluncurkan model ke produksi dilakukan. Tahap evaluasi juga mencakup peninjauan keseluruhan proses untuk memastikan bahwa model yang dibangun tidak hanya akurat secara teknis, tetapi juga relevan dan efektif dalam konteks bisnis.

6. Deployment

   Deployment merupakan tahap terakhir dari siklus CRISP-DM. Hasil dari tahap ini dapat berupa laporan akhir atau komponen perangkat lunak. Pada tahap ini, model yang telah dibangun, diuji, dan dievaluasi diterapkan ke dalam lingkungan operasional untuk memberikan manfaat yang maksimal. Jika hasilnya berupa laporan akhir, laporan tersebut akan merangkum temuan utama dari seluruh proses *data mining*, termasuk analisis hasil model, rekomendasi berdasarkan model yang dihasilkan, dan saran tindak lanjut yang dapat dilakukan. Laporan ini dapat digunakan oleh pihak yang berkepentingan untuk membuat keputusan berbasis data yang lebih *valid.*

2.3.2 Long Short-Term Memory

Long Short-Term Memory (LSTM) merupakan sebuah pengembangan dari *recurrent neural network* atau RNN yang dirancang untuk mengatasi kekurangan dari RNN [54]. LSTM terdiri dari sel memori, dengan masing-masing sel terdiri atas tiga jenis gerbang yaitu *input gate, output gate, dan forget gate* [55]. Gerbang-gerbang ini memutuskan informasi apa yang akan ditambahkan, dihapus, dan dikeluarkan dari sel memori. Inti dari LSTM terletak pada transisi sel dan struktur gerbang yang mengontrol aliran informasi [56]. Hal ini membuat LSTM unggul dalam mengingat informasi penting dalam jangka waktu panjang, sehingga dapat mengatasi keterbatasan yang dihadapi oleh RNN tradisional akibat masalah *vanishing gradient* [57]. LSTM mampu menangkap pola jangka panjang dari data *time-series*, dimana sangat penting dalam memprediksi data yang memiliki pola

data *non-linear*. Model LSTM menawarkan solusi yang menjanjikan untuk mengolah data berurutan dan mempertahankan informasi penting dari waktu ke waktu [58]. Selain itu, kemampuan LSTM untuk mempelajari ketergantungan jangka panjang dari data berurutan historis menjadikannya cocok untuk meramalkan pergerakan harga mata uang kripto [59]. Berikut ini adalah rumus-rumus yang menggambarkan fungsi dari masing-masing gerbang dalam LSTM, yang berperan penting dalam mengontrol aliran informasi pada setiap langkah waktu.

1. Forget gate

   Forget gate berfungsi untuk menentukan seberapa banyak informasi dari sel memori sebelumnya yang akan dilupakan, dimana dapat dinyatakan sebagai berikut:

$$\boldsymbol{f_t = \sigma(w_f[h_t - 1, x_t] + b_f)}$$

Rumus 6. *Forget gate* [56]

   Keterangan:

   f_t = *forget gate*.

   σ = fungsi aktivasi sigmoid.

   w_f = matriks bobot *forget gate*.

   $h_t - 1$ = *output* dari blok LSTM sebelumnya.

   x_t = *input* data pada *timestep* t.

   b_f = bias pada *forget gate*.

2. Input gate

   Input gate berfungsi untuk mengontrol seberapa banyak informasi baru yang akan ditambahkan ke dalam sel memori, dimana dapat dinyatakan sebagai berikut:

$$\boldsymbol{i_t = \sigma(w_i[h_t - 1, x_t] + b_i)}$$

Rumus 7. *Input gate* [56]

   Keterangan:

   i_t = *input gate*.

   σ = fungsi aktivasi sigmoid.

w_i = matriks bobot *input gate*.

$h_t - 1$ = *output* dari blok LSTM sebelumnya.

x_t = *input* data pada *timestep* t.

b_i = bias pada *input gate*.

3. Output gate

Output gate berfungsi mengontrol seberapa banyak informasi dari *cell state* yang diteruskan ke *hidden state* untuk digunakan dalam prediksi dan input pada langkah waktu berikutnya, dimana dapat dinyatakan sebagai berikut:

$$\boldsymbol{o_t = \sigma(w_o[h_t - 1, x_t] + b_o)}$$

Rumus 8. *Output gate* [56]

Keterangan:

o_t = *output gate*

σ = fungsi aktivasi sigmoid.

w_o = matriks bobot terkait dengan *output gate*.

$h_t - 1$ = *output* dari blok LSTM sebelumnya.

x_t = *input* data pada *timestep* t.

b_o = bias pada *output gate*.

4. Candidate Memory Cell

Candidate Memory Cell adalah nilai atau informasi baru yang dihasilkan dari proses input dan informasi dari status sebelumnya yang akan dipertimbangkan untuk disimpan ke dalam *cell state*, dimana dapat dinyatakan sebagai berikut:

$$\boldsymbol{L_t = tanh(w_c[h_t - 1, x_t] + b_c)}$$

Rumus 9. *Candidate Memory Cell* [56]

Keterangan:

L_t = c*andidate memory cell*

$tanh$ = fungsi aktivasi tanh.

w_c = matriks bobot pada *new candidate state*.

$h_t - 1$ = *hidden state.*

x_t = *input* data pada *timestep* t.

b_c = bias pada *new candidate state*.

5. Cell State

Cell State berfungsi untuk memperbarui *cell state* dengan menggabungkan informasi dari *cell state* sebelumnya yang dipertahankan oleh *forget gate* dan informasi baru yang ditambahkan berdasarkan *input gate* dan calon memori. *Cell State* dapat dinyatakan sebagai berikut:

$$\boldsymbol{C_t = f_t \times C_{t-1} + i_t \times L_t}$$

Rumus 10. *Cell State* [56]

Keterangan:

C_t = *cell state*.

f_t = *forget gate*.

$C_t - 1$ = *cell state* pada waktu sebelumnya.

i_t = *input gate*.

L_t = c*andidate memory cell.*

6. Hidden State

Hidden state adalah hasil *output* dari LSTM yang menyimpan informasi terpilih dan digunakan untuk mempengaruhi keputusan pada langkah waktu berikutnya, serta menjadi input untuk proses prediksi selanjutnya. *Hidden state* dapat dinyatakan sebagai berikut:

$$\boldsymbol{h_t = O_t \times \tanh\,(C_t)}$$

Rumus 11. *Hidden State* [56]

Keterangan:

h_t = *hidden state.*

O_t = *output gate.*

$tanh$ = fungsi aktivasi tanh.

C_t = *cell state*.

2.3.3 Gated Recurrent Unit

Gate Recurrent Unit (GRU) merupakan jenis dari *Recurrent Neural Network* (RNN) yang dirancang untuk mengatasi masalah *vanishing gradient* [60]. GRU memiliki arsitektur yang lebih sederhana daripada LSTM, dimana model GRU memiliki lebih sedikit parameter dan struktur yang tidak terlalu rumit dibandingkan dengan model LSTM [61]. Hal ini membuat model GRU lebih efisien secara komputasi. GRU memiliki dua gerbang untuk mengontrol aliran informasinya yaitu *update gate* dan *reset gate*. Konsep dari GRU adalah secara selektif memperbarui status tersembunyi di setiap langkah waktu dengan menggunakan mekanisme gating. Berikut ini merupakan beberapa rumus dari algoritma GRU:

1. Reset Gate

   Reset gate menentukan seberapa banyak informasi sebelumnya yang akan diabaikan atau dipertahankan [62], dimana dinyatakan sebagai berikut:

$$\boldsymbol{r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r)}$$

Rumus 12. *Reset Gate*

   Keterangan:

   r_t = *reset gate.*

   σ = fungsi aktivasi sigmoid.

   W_r = matriks bobot terkait dengan *reset gate*.

   x_t = *input* data pada *timestep* t.

   U_r = *input* pada *timestep* saat ini.

   $h_t - 1$ = *output* dari blok LSTM sebelumnya.

   b_r = nilai bias pada *reset gate*.

2. Update Gate

   Update gate mengontrol seberapa banyak informasi sebelumnya untuk diteruskan ke kondisi saat ini [62], yang dinyatakan sebagai berikut:

$$\boldsymbol{Z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z)}$$

Rumus 13. *Update Gate*

Keterangan:

z_t = *update gate*

σ = fungsi aktivasi sigmoid.

W_z = matriks bobot terkait dengan *update gate*.

x_t = *input* data pada *timestep* t.

U_z = *input* pada *timestep* saat ini

$h_t - 1$ = *output* dari blok LSTM sebelumnya.

b_z = nilai bias pada *update gate*.

3. Candidate Hidden State

Candidate Hidden State adalah kandidat dari status tersembunyi yang akan diperbarui pada waktu tertentu. Nilai ini menghimpun informasi baru yang dihasilkan berdasarkan input saat ini dan *hidden state* sebelumnya untuk diperhitungkan dalam proses pembaruan, yang dinyatakan sebagai berikut [62]:

$$\boldsymbol{\hat{h}_t = tanh(W_h x_t + r_t \cdot U_h h_{t-1} + b_h)}$$

Rumus 14. *Candidate Hidden State*

Keterangan:

$\hat{h}_t$ = *candidate hidden gate.*

$tanh$ = fungsi aktivasi tanh.

W_h = matriks bobot terkait dengan *hidden gate*.

x_t = *input* data pada *timestep* t.

r_t = *reset gate*.

U_h = matriks bobot terkait dengan *hidden gate.*

$h_t - 1$ = *output* dari blok LSTM sebelumnya.

b_h = nilai bias pada *update gate*.

4. Hidden State

Hidden State adalah representasi akhir menyimpan informasi dari semua langkah sebelumnya dalam urutan, yang digunakan untuk memprediksi atau menghasilkan *output* pada langkah waktu berikutnya, yang dinyatakan sebagai berikut [62]:

$$\boldsymbol{h_t = z_t \cdot h_{t-1} + (1 - z_t) \cdot \hat{h}_t)}$$

Rumus 15. *hidden State*

Keterangan:

h_t = *hidden state*

z_t = *update gate*

$h_t - 1$ = *output* dari blok LSTM sebelumnya.

$\hat{h}_t$ = *candidate hidden state.*

2.3.4 Extreme Gradient Boosting

Extreme Gradient Boosting (XGBoost) adalah model terintegrasi berdasarkan pohon keputusan [63], dimana sering digunakan untuk menangani dataset yang besar, kompleks, dan memiliki banyak fitur. XGBoost menggunakan metode *ensemble* ML yang didasarkan pada pendekatan *gradient boosting* dalam meningkatkan prinsip pohon keputusan [64]. XGBoost bekerja secara berulang dalam memperbaiki model yang lemah dengan menambahkan lebih banyak model yang akan bekerja berdasarkan perbaikan kesalahan pada model sebelumnya, serta mencakup fitur – fitur regularisasi, *parallel processing*, dan *handling missing values*. XGBoost menggunakan fungsi objektif dengan dua komponen utama yaitu *loss function* dan regularisasi yang dinyatakan dalam rumus berikut ini.

1. Objective Function

Objective Function berfungsi sebagai fungsi matematis dalam masalah optimisasi untuk mengukur seberapa baik dan buruknya solusi yang dihasilkan oleh model. XGBoost memperbaiki model secara iteratif dengan mengurangi kesalahan prediksi melalui optimasi fungsi objektif yang dirancang untuk meminimalkan kesalahan prediksi bersamaan dengan mengontrol kompleksitas

model. Konsep dasar dari algoritma XGBoost adalah meminimalkan fungsi objektif [65], dimana diformulasikan sebagai berikut:

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

Rumus 16. *Objective Function* [65]

Keterangan:

$\mathcal{L}^{(t)}$ = fungsi objektif pada iterasi t.

n = jumlah data.

y_i = label target.

$\hat{y}_i^{(t-1)}$ = nilai prediksi pada iterasi sebelumnya.

$f_i(x_i)$ = fungsi prediksi.

$\Omega(f_t)$ = regularisasi model pada iterasi t.

2. Loss Function

Loss function menggambarkan *error* antara nilai prediksi dengan nilai sebenarnya dan dinyatakan sebagai berikut:

$$l(y_i, \hat{y}_i) = (y_i \hat{y}_i)^2$$

Rumus 17. *Loss Function*

Keterangan:

l = *loss function.*

y_i = nilai sebenarnya.

$\hat{y}_i$ = prediksi model.

3. Regularization

Regularization berfungsi untuk mencegah *overfitting* [65], dengan menambahkan penalti terhadap kompleksitas model yang dinyatakan sebagai berikut:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda ||w||^2$$

Rumus 18. *Regularization* [65]

Keterangan:

$\Omega(f)$ = regularisasi.

γ = penalti pada jumlah daun dalam pohon.

T = jumlah daun pada *decision tree.*

λ = parameter regularisasi untuk bobot daun.

$||w||^2$ = kuadrat dari bobot semua daun.

4. Second-order Approximation

Rumus ini digunakan dalam algoritma XGBoost untuk mempercepat proses optimasi fungsi objektif pada setiap iterasi. Formula ini menggabungkan informasi dari *gradient* pertama dan *hessian* untuk melakukan pendekatan orde kedua terhadap fungsi kerugian dan dinyatakan sebagai berikut:

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

Rumus 19. *Second-order Approximation*

Keterangan:

$L^{(t)}$ = fungsi objektif.

n = jumlah data.

g_i = *gradient*.

h_i = *hessian*.

$f_t(x_i)$ = prediksi pada iterasi ke- t.

$\Omega(\boldsymbol{f_t})$ = regularisasi pada fungsi prediksi $\boldsymbol{f_t}$.

5. Gradient dan Hessian

Gradient dan *Hessian* adalah dua elemen kunci yang diperoleh dari *Taylor expansion* untuk mendekati fungsi objektif pada iterasi. *Gradient* adalah turunan pertama dari fungsi kerugian terhadap prediksi dan berfungsi menunjukkan arah perbaikan dalam optimasi, sementara *Hessian* adalah turunan kedua dari fungsi kerugian terhadap prediksi dan berfungsi mengatur langkah optimasi yang lebih tepat, yang dinyatakan sebagai berikut:

$$\boldsymbol{g_i = \frac{\partial \mathcal{L}}{\partial \hat{y}_i}, h_i = \frac{\partial^2 \mathcal{L}}{\partial \hat{y}_i^2}}$$

Rumus 20. *Gradient* dan *Hessian*

Keterangan:

g_i = *gradient* pertama dari *loss function*.

h_i = *hessian* (*gradient*) kedua dari *loss function*.

$\hat{y}_i$ = prediksi model.

$\mathcal{L}$ = fungsi objektif.

6. Weight of Leaf

Dalam struktur pohon, setiap daun memiliki bobot optimal yang dihitung dengan menggunakan jumlah *gradient* dan jumlah *hessian* pada data yang berada di daun ke-j, dimana dinyatakan sebagai berikut:

$$\boldsymbol{w_j^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda}}$$

Rumus 21. *Weight of leaf*

Keterangan:

w_j^* = bobot optimal untuk daun ke-j.

I_j = indeks data yang jatuh ke daun ke-j.

$\sum_{i \in I_j} g_i$ = jumlah *gradient* untuk semua data pada daun ke-j.

$\sum_{i \in I_j} h_i$ = jumlah *hessian* untuk semua data pada daun ke-j.

λ = parameter regularisasi untuk bobot *leaf*.

7. Split Evaluation

Split Evaluation digunakan untuk menghitung pengurangan kerugian atau *loss reduction* yang dihasilkan oleh suatu pembagian (*split*) pada simpul pohon keputusan dan dinyatakan sebagai berikut:

$$\boldsymbol{L_{split} = \frac{1}{2}\left(\frac{(\sum_{i \in I_L} g_i)^2}{\sum_{i \in I_L} h_i + \lambda} + \frac{(\sum_{i \in I_R} g_i)^2}{\sum_{i \in I_R} h_i + \lambda} - \frac{(\sum_{i \in I} g_i)^2}{\sum_{i \in I} h_i + \lambda}\right) - \gamma}$$

Rumus 22. *Split Evaluation*

Keterangan:

L_{split} = *split evaluation*

g_i = *gradient*

h_i = *hessian*

I = indeks data dari seluruh dataset sebelum *split*.

I_L = indeks data di subset kiri (*left*) setelah *split.*

L_R = indeks data di subset kanan (*right*) setelah *split.*

ג = *regularization parameter* (lambda).

γ = *regularization term* (gamma).

2.4 Tools/software yang digunakan

Bagian ini menjelaskan *tools* dan *software* yang digunakan untuk mendukung proses penelitian, mulai dari pengumpulan data, pengolahan data, pemodelan, evaluasi, hingga visualisasi hasil.

2.4.1 Python

Python merupakan bahasa pemrograman tingkat tinggi yang ditafsirkan untuk keperluan umum, interaktif, berorientasi objek, dan tingkat tinggi [66]. Python memungkinkan eksekusi kode secara langsung tanpa memerlukan kompilasi, yang meningkatkan kecepatan pengembangan dan *debugging*. Karakteristik berorientasi objeknya memfasilitasi pemodelan data dan aplikasi yang kompleks, dengan memungkinkan pengguna untuk mengorganisir kode mereka secara modular dan *reusable*.

2.4.2 TensorFlow

TensorFlow adalah sebuah kerangka kerja *machine learning open-source* yang kuat, memungkinkan pengembangan dan penerapan model AI yang skalabel untuk meningkatkan akurasi dan efisiensi dalam ekstraksi pengetahuan, klasifikasi, dan otomatisasi alur kerja [67]. TensorFlow menyediakan berbagai *tools* dan *library* yang digunakan untuk membangun serta melatih model LSTM dan GRU pada data deret waktu penelitian ini. Keunggulan utama *TensorFlow* adalah kemampuannya dalam memproses perhitungan yang berat dengan menggunakan

baik CPU maupun GPU, yang sangat meningkatkan efisiensi waktu pelatihan model dengan dataset besar.

2.4.3 Keras

Keras adalah API atau *library* Python tingkat tinggi untuk pengembangan model *deep learning* yang berjalan di atas TensorFlow [68]. Keras mendukung pengembangan model yang fokus pada konsep *deep learning* yaitu pembuatan lapisan untuk jaringan saraf. Keras dapat menggunakan beberapa *computational backend*, sehingga kode pemodelan relatif portabel dan fleksibel. Keras dapat terintegrasi dalam *TensorFlow* dan dapat digunakan untuk melakukan pembelajaran mendalam dengan cepat karena menyediakan modul bawaan untuk semua komputasi jaringan neural.

2.4.4 Numpy

Numpy merupakan pustaka dasar untuk komputasi numerik menyediakan dukungan untuk *array* dan matriks besar, serta banyak fungsi matematika tingkat tinggi untuk operasi pada *array* tersebut [69]. Numpy menjadi pilihan utama untuk manipulasi data numerik dalam python karena mendukung berbagai jenis komputasi ilmiah dan analisis data yang memerlukan pengolahan *array* yang efektif dan efisien.

2.4.5 Pandas

Pandas adalah sebuah *library data science* dan analitik di Python [70]. Pandas berguna untuk mengolah data, menawarkan kemampuan yang kuat untuk memanipulasi data dan mendapatkan kesimpulan yang berarti [71]. Pandas menyediakan objek utama seperti *DataFrame* yang memudahkan proses pembacaan, pembersihan, transformasi, penggabungan, dan analisis data. Kemampuannya dalam menangani data terindeks waktu juga menjadikan Pandas sangat relevan dalam penelitian yang menggunakan data deret waktu.

2.4.6 Matplotlib

Matplotlib adalah sebuah *library Python* yang menyediakan solusi yang ringkas, efisien, dan kode yang sederhana untuk visualisasi data [72]. Matplotlib menyediakan fungsi untuk menampilkan data secara grafis dan menghasilkan

berbagai macam *plot*. Dalam penelitian ini, visualisasi memiliki peran penting untuk membantu interpretasi data, pola, dan hasil evaluasi model secara lebih informatif.

2.4.7 Seaborn

Seaborn adalah perpustakaan visualisasi grafis dibangun di atas *Matplotlib* [73]. *Library* ini menyediakan antarmuka tingkat tinggi untuk menggambar grafik statistik yang menarik dan informatif dengan cara yang sederhana. Seaborn juga terintegrasi erat dengan struktur pandas [74], sehingga memudahkan analisis data yang tersimpan dalam *DataFrame*.

2.4.8 MinMaxScaler

MinMaxScaler merupakan metode normalisasi data yang mentransformasikan nilai setiap fitur ke dalam rentang antara 0 dan 1 [75]. MinMaxScaler mengubah nilai asli berdasarkan nilai minimum dan maksimum untuk menyamakan skala antarvariabel numerik tanpa mengubah pola distribusi relatifnya. Dalam konteks *machine learning* dan *deep learning*, MinMaxScaler digunakan untuk mencegah fitur dengan rentang nilai yang besar mendominasi proses pelatihan model, serta membantu model melakukan pembelajaran dengan lebih stabil dan efisien.

2.4.9 Google Colab

Google Colab merupakan lingkungan *notebook* berbasis *cloud* yang memungkinkan pengguna menulis, menjalankan, dan mendokumentasikan kode Python dalam satu dokumen terintegrasi. *Platform* ini kompatibel dengan format Jupyter Notebook, dapat diakses melalui peramban tanpa instalasi awal, serta mendukung akselerator komputasi seperti GPU dan TPU [76]. Penelitian ini menggunakan Google Colab dalam mendukung proses pengolahan data, pelatihan model, evaluasi, dan visualisasi hasil secara efisien.

2.4.10 Yahoo Finance

Yahoo Finance sebagai salah satu penyedia data pasar keuangan yang menyediakan informasi harga aset dan indeks secara publik. Data diakses secara terprogram menggunakan pustaka Python yfinance, yang berfungsi sebagai

antarmuka untuk melakukan pengambilan data historis dari Yahoo Finance. Pada implementasinya, setiap instrumen direpresentasikan dengan simbol (*ticker*) tertentu. Data yang diambil umumnya mencakup variabel harga harian seperti *Open, High, Low, Close,* dan *Volume*, yang kemudian digunakan sebagai dasar pembentukan dataset prediksi.

2.4.11 Github

GitHub merupakan *platform* berbasis *web* yang digunakan untuk menyimpan, mengelola, dan melacak perkembangan proyek perangkat lunak dengan sistem *version control Git*. GitHub sebagai media pengelolaan repositori kode dan artefak pendukung secara terstruktur, sehingga mendukung keterlacakan versi dan integrasi pada tahap implementasi maupun *deployment*.

2.4.12 Streamlit

Streamlit merupakan *framework open-source* berbasis Python yang digunakan untuk membangun aplikasi web interaktif secara cepat. Streamlit memungkinkan model prediksi, hasil evaluasi, dan visualisasi ditampilkan dalam antarmuka web yang lebih aplikatif dan mudah diakses.