BAB II

LANDASAN TEORI

2.1 Penelitian Terdahulu

Penelitian terdahulu menjadi dasar penting dalam penyusunan penelitian ini. Meskipun penelitian terkait prediksi *stunting* dengan menggunakan metode *time series* masih terbatas, beberapa penelitian di bidang yang serupa dapat dijadikan referensi untuk memperkuat landasan teori. Beberapa penelitian terdahulu yang dijadikan acuan dan pembanding untuk penelitian ini dapat dilihat pada Tabel 2.1.

Tabel 2.1 Penelitian Terdahulu

No	Identitas Penelitian	Metode	Hasil
1	**Jurnal:** G-Tech: Jurnal Teknologi Terapan, Vol. 8, No. 3, Juli 2024, hlm. 1890–1900 [13]. **Penulis:** Benny Putra, Alva Hendi Muhammad **Judul:** Prediksi Prevalensi *Stunting* di Indonesia dengan *Ordinary Least Square* (OLS)	*Neural Network* (NN), *Radial Basis Function* (RBF) *Network, Support Vector Regression* (SVR) Kernel RBF, dan *Ordinary Least Square* (OLS).	Hasil menunjukkan bahwa OLS memberikan performa terbaik dalam memprediksi prevalensi *stunting* dengan nilai MAE = 0.020390, MSE = 0.000816, RMSE = 0.028561, R^2 = 0.923281, dan MAPE = 0.044035. Model lain memiliki performa lebih rendah, yaitu NN (MAE 0.089, MSE 0.012), RBF *Network* (MAE 0.035, MSE 0.0024), dan SVR (MAE 0.061, MSE 0.0048).
2	**Jurnal:** Jurnal Teknologi Informatika dan Komputer MH. Thamrin, Vol. 10, No. 2, September 2024, p-ISSN 2656-9957, e-ISSN 2622-8475 [18]. **Penulis:** Irnanda Septian Ika Putri, Risqy Siwi Pradini, Mochammad Anshori **Judul:** *Decision Tree Regression* untuk Prediksi Prevalensi	*Decision Tree Regression*	Model *Decision Tree Regression* berhasil memprediksi prevalensi *stunting* di NTT dengan nilai RMSE = 0,093, menunjukkan tingkat kesalahan yang rendah dan performa prediksi yang baik. Model juga menghasilkan visualisasi pohon keputusan yang mengidentifikasi faktor dominan penyebab *stunting*, yaitu kemiskinan, IPM, ASI, dan imunisasi.

No	Identitas Penelitian	Metode	Hasil
	Stunting di Provinsi Nusa Tenggara Timur		
3	**Jurnal:** Informatics, Vol. 11, No. 6, 2024 [19]. **Penulis:** Novia Hasdyna, Rozzi Kesuma Dinata, Rahmi and T. Irfan Fajri **Judul:** *Hybrid Machine Learning for Stunting Prevalence: A Novel Comprehensive Approach to Its Classification, Prediction, and Clustering Optimization in* Aceh, Indonesia	SVM untuk klasifikasi, *Linear Regression* untuk prediksi, K-Medoids (dengan *Weighted Product*) untuk *clustering*.	Model SVM dengan kernel RBF mencapai akurasi 91,3%, sedangkan model regresi linear menghasilkan MSE = 0.137 untuk prediksi prevalensi *stunting*. Hasil *clustering* menggunakan K-Medoids dan WP menunjukkan peningkatan nilai *Silhouette Index*, menandakan klaster yang lebih optimal.
4	**Jurnal:** *Information Sciences*, Vol. 586, 2022 [20]. **Penulis:** M.A. Castan-Lascorz, P. Jimenez-Herrera, A. Troncoso, G. **Judul:** *A new hybrid method for predicting univariate and multivariate time series based on pattern forecasting*	Metode hybrid gabungan *clustering* (k-means), klasifikasi (*distance to centroid*, naive Bayes, SVM), dan berbagai algoritma *forecasting* (ARIMA, Holt-Winters, LSTM, KNN, CART, Random Forest).	LSTM (MAPE 3.0%) dan *Holt-Winters* (MAPE 3.3%) terbukti sebagai model univariate terbaik, mengalahkan ARIMA (MAPE 4.3%).
5	**Jurnal:** Alexandria Engineering Journal, Vol. 61, 2022 [21]. **Penulis:** K.E. ArunKumar, Dinesh V. Kalaga, Ch. Mohan Sai Kumar, Masahiro Kawaji, Timothy M. Brenza **Judul:** *Comparative analysis of Gated Recurrent*	Statistik Klasik yaitu ARIMA & SARIMA dan *Deep Learning* Modern yaitu LSTM & GRU	Hasilnya sangat bervariasi tergantung karakteristik data. Untuk data AS, model LSTM (RMSE 12.100) jauh lebih akurat daripada ARIMA (RMSE 407.000). Namun, untuk data Chili, model statistik SARIMA (RMSE 442) justru jauh mengungguli LSTM (RMSE 7.510).

No	Identitas Penelitian	Metode	Hasil
	Units (GRU), long Short-Term memory (LSTM) cells, autoregressive Integrated moving average (ARIMA), seasonal autoregressive Integrated moving average (SARIMA) for forecasting COVID-19 trends		
6	**Jurnal:** *Renewable Energy*, Vol. 221, 2024 [22]. **Penulis:** Shahram Hanifi, Andrea Cammarono, Hossein Zare-Behtash **Judul:** *Advanced hyperparameter optimization of deep learning models for wind power prediction*	Membandingkan 3 *framework* optimasi canggih yaitu Optuna, Scikit-opt, Hyperopt	Optuna terbukti memiliki efisiensi (kecepatan tuning) terbaik untuk model CNN dan LSTM. Untuk akurasi, LSTM yang dioptimasi (oleh Hyperopt & Optuna) mengalahkan model CNN. Tuning terbukti sangat penting.
7	**Jurnal:** Eur. Phys. J. Plus, Vol. 137, 2022 [23]. **Penulis:** Zhen Shan, Jianhua Yang, Miguel A. F. Sanjuán, Chengjin Wu, Houguang Liu **Judul:** *A novel adaptive moving average method for signal denoising in strong noise background*	Mengusulkan metode baru bernama *Adaptive Moving Average* (AMA) dan membandingkannya dengan *Moving Average, Variational Mode Decomposition* (VMD), *Wavelet Threshold Denoising* (WTD)	Metode AMA yang baru terbukti lebih unggul dari MA standar karena bisa beradaptasi dengan sinyal.
8	**Jurnal:** *Journal of Open Innovation: Technology, Market, and Complexity,* Vol. 11, 2025 [24]. **Penulis:**	Menggunakan LSTM sebagai model utama, menggunakan Bayesian Optimization (seperti Optuna) untuk *hyperparameter*	Membuktikan bahwa *Walk-Forward Validation* adalah metode yang tepat untuk data *time series* karena tidak "membocorkan masa depan". Membuktikan bahwa Hyperparameter Optimization (HPO) sangat penting untuk menemukan performa model terbaik (mengurangi RMSE sebesar 39.55%). Model LSTM terbaik (dengan HPO dan sentiment) mengalahkan model lain.

No	Identitas Penelitian	Metode	Hasil
	Eko Putra Wahyuddin, Rezzy Eko Caraka, Robert Kurniawan, Wahyu Caesarendra, Prana Ugiana Gio, Bens Pardamean **Judul:** *Improved LSTM hyperparameters alongside sentiment walk-forward validation for time series prediction*	*tuning*, menggunakan dan merekomendasikan *Walk-Forward Validation* sebagai metode evaluasi yang benar (menggantikan cross-validation acak).	
9	**Jurnal:** Ultima Infosys: Jurnal Ilmu Sistem Informasi, Vol. 13, No. 1, 2022 [25]. **Penulis:** Raymond Sunardi Oetama, Ford Lumban Gaol, Benfano Soewito, Harco Leslie Hendric Spits Warnars **Judul:** *Finding Features of Multiple Linear Regression On Currency Exchange Pairs*	Menggunakan CRISP-DM sebagai kerangka kerja penelitian, menggunakan *Multiple Linear Regression* (MLR) untuk prediksi, menggunakan *Feature Selection* berbasis korelasi.	CRISP-DM berhasil diterapkan sebagai kerangka kerja untuk prediksi Forex dan model MLR terbaik (menggunakan 5 harga original harian sebelumnya) mencapai performa sangat tinggi dengan R-Squared = 0.99477 dan RMSE = 0.00408.
10	**Jurnal:** Data and Metadata, Vol. 3, 2024 [26]. **Penulis:** Deviana Ridhani, Krismadinata, Dony Novaliendry, Ambiyar, Hansi Effendi **Judul:** *Development of an Intelligent Learning Evaluation System Based on Big Data*	Menggunakan metodologi Agile, *backend & frontend* (*dashboard*) dibangun menggunakan *framework CodeIgniter* (CI), *database* menggunakan PostgreSQL, visualisasi data menggunakan Chart.js.	Berhasil membangun *Intelligent Learning Evaluation System* (ILES) menggunakan *CodeIgniter,* menghasilkan "*user-friendly dashboards"* (halaman *dashboard*) untuk visualisasi data *real-time*, membuktikan bahwa *framework CodeIgniter* efektif dan cepat untuk pengembangan sistem informasi/*dashboard.*

Penelitian terdahulu yang relevan dengan penelitian ini dapat dikelompokkan menjadi tiga kategori utama. Kategori pertama, yaitu penelitian yang berfokus pada topik *stunting*, menunjukkan bahwa sebagian besar penelitian di Indonesia masih berfokus pada analisis faktor penyebab yang bersifat *multivariate*. Penelitian oleh Benny Putra dan Alva Hendi Muhammad (2024) berfokus pada upaya memprediksi prevalensi *stunting* berdasarkan data sosial-ekonomi. Penelitian ini membandingkan empat model yaitu *Neural Network* (NN), *Radial Basis Function* (RBF) *Network*, *Support Vector Regression* (SVR) dengan kernel RBF, dan *Ordinary Least Square* (OLS). Hasil penelitian menunjukkan bahwa OLS memberikan performa terbaik dengan nilai MAE sebesar 0.020390, MSE sebesar 0.000816, RMSE sebesar 0.028561, R^2 sebesar 0.923281, dan MAPE sebesar 0.044035 [13]. Selanjutnya penelitian oleh Irnanda Septian Ika Putri, Risqy Siwi Pradini, dan Mochammad Anshori (2024) menggunakan model *Decision Tree Regression* untuk memprediksi prevalensi *stunting* di Provinsi Nusa Tenggara Timur. Hasil penelitian menunjukkan bahwa model ini memiliki performa yang baik dengan nilai RMSE sebesar 0,093, menandakan tingkat kesalahan prediksi yang rendah. Selain itu, model berhasil memvisualisasikan pohon keputusan yang mengidentifikasi faktor-faktor dominan penyebab *stunting*, yaitu kemiskinan, Indeks Pembangunan Manusia (IPM), pemberian ASI, dan imunisasi [18]. Berbeda dari kedua penelitian tersebut, penelitian oleh Novia Hasdyna, Rozzi Kesuma Dinata, Rahmi, dan T. Irfan Fajri (2024) mengusulkan pendekatan *hybrid machine learning* yang lebih komprehensif dengan menggabungkan beberapa model, yaitu *Support Vector Machine* (SVM) untuk klasifikasi, *Linear Regression* untuk prediksi, serta K-Medoids dengan *Weighted Product* untuk *clustering*. Hasilnya menunjukkan bahwa model SVM mencapai akurasi 91,3%, sedangkan *Linear Regression* menghasilkan MSE sebesar 0.137 untuk prediksi prevalensi *stunting*. Selain itu, kombinasi K-Medoids dan *Weighted Product* juga meningkatkan *Silhouette Index*, yang menandakan pembentukan klaster data yang lebih optimal [19]. Ketiga penelitian tersebut sama-sama berfokus pada isu *stunting* di Indonesia, namun sebagian besar masih menggunakan pendekatan *multivariate* dengan melibatkan banyak variabel sosial, ekonomi, dan kesehatan. Sementara penelitian ini berbeda karena berfokus pada analisis tren *univariate time series* yang hanya

menggunakan data angka prevalensi *stunting* untuk melihat pola dan memprediksi tren ke depan.

Kategori kedua berkaitan dengan penelitian yang memvalidasi perbandingan model yang digunakan dalam penelitian ini. Meskipun topiknya berbeda dengan *stunting*, berbagai penelitian sebelumnya telah membuktikan bahwa perbandingan antara model ARIMA, LSTM, dan GRU efektif dalam analisis data deret waktu di berbagai bidang. Penelitian oleh M.A. Castan-Lascorz, P. Jimenez-Herrera, A. Troncoso, dan G. Asencio-Cortes (2022) mengembangkan metode *hybrid forecasting* yang menggabungkan *clustering* (K-Means), *classification (Distance to Centroid, Naive Bayes*, SVM), dan berbagai algoritma *forecasting* seperti ARIMA, *Holt-Winters*, LSTM, KNN, CART, dan Random Forest. Hasil penelitian menunjukkan bahwa LSTM mendapatkan MAPE sebesar 3,0% dan Holt-Winters mendapatkan MAPE sebesar 3,3% menjadi model *univariate* terbaik yang mengungguli ARIMA dengan MAPE sebesar 4,3% dalam prediksi konsumsi Listrik [20]. Selanjutnya, penelitian oleh K.E. ArunKumar, Dinesh V. Kalaga, Ch. Mohan Sai Kumar, Masahiro Kawaji, dan Timothy M. Brenza (2022) melakukan perbandingan antara model statistik klasik (ARIMA dan SARIMA) serta model *deep learning modern* (LSTM dan GRU) untuk memprediksi tren COVID-19. Hasil penelitian menunjukkan bahwa performa setiap model sangat bergantung pada karakteristik data. Untuk data di Amerika Serikat, model LSTM mendapatkan RMSE sebesar 12.100 jauh lebih akurat dibandingkan ARIMA yang mendapatkan RMSE sebesar 407.000. Namun, untuk data di Chili, model SARIMA mendapatkan RMSE sebesar 442 justru mengungguli LSTM yang mendapatkan RMSE sebesar 7.510 [21]. Hal ini menunjukkan bahwa model *deep learning* cenderung lebih unggul pada data dengan pola kompleks dan non-linear, sedangkan model statistik klasik lebih baik pada data yang stabil. Kedua penelitian tersebut memperkuat relevansi pendekatan dalam penelitian ini, yaitu melakukan perbandingan antara ARIMA, LSTM, dan GRU untuk menemukan model terbaik dalam memprediksi tren *stunting* berdasarkan pola historisnya.

Kategori ketiga mencakup penelitian yang mendukung teknik-teknik yang digunakan dalam metodologi penelitian ini. Karena data *time series* sering

mengandung *noise*, diperlukan proses *data smoothing* agar pola tren dapat terbaca dengan lebih baik. Penelitian oleh Zhen Shan, Jianhua Yang, Miguel A. F. Sanjuán, Chengjin Wu, Houguang Liu (2022) memvalidasi penggunaan *Moving Average* (MA) sebagai metode dasar yang efektif untuk *signal denoising*, dan bahkan mengusulkan metode baru bernama *Adaptive Moving Average* (AMA) yang terbukti lebih adaptif terhadap perubahan sinyal dibandingkan MA standar [23]. Selanjutnya, untuk tahap evaluasi model, penelitian oleh Eko Putra Wahyuddin, Rezzy Eko Caraka, Robert Kurniawan, Wahyu Caesarendra, Prana Ugiana Gio, Bens Pardamean (2025) menunjukkan bahwa metode *Walk-Forward Validation* merupakan pendekatan yang paling tepat untuk data *time series,* karena mampu mencegah terjadinya kebocoran informasi masa depan. Penelitian tersebut juga menekankan pentingnya *hyperparameter optimization* (HPO) menggunakan *framework* seperti Optuna untuk meningkatkan performa model [24]. Temuan ini sejalan dengan penelitian Shahram Hanifi, Andrea Cammarono, Hossein Zare-Behtash (2024) yang membandingkan tiga *framework* optimasi yaitu Optuna, Scikit-opt, dan Hyperopt. Penelitian tersebut menemukan bahwa Optuna memiliki efisiensi terbaik serta mampu menghasilkan akurasi model LSTM yang lebih tinggi [22]. Dalam aspek alur kerja dan implementasi, penelitian oleh Raymond Sunardi Oetama, Ford Lumban Gaol, Benfano Soewito, Harco Leslie Hendric Spits Warnars (2022) memvalidasi penggunaan kerangka kerja CRISP-DM sebagai panduan yang sistematis dalam proyek data mining, yang terbukti efektif dalam menghasilkan model prediksi [25]. Sementara itu, penelitian oleh Deviana Ridhani, Krismadinata, Dony Novaliendry, Ambiyar, Hansi Effendi (2024) menunjukkan bahwa *framework CodeIgniter* sangat efisien untuk pengembangan *dashboard* visualisasi data, karena mendukung integrasi data dan menghasilkan tampilan yang *user-friendly* [26].

Penelitian ini memiliki perbedaan dan keunggulan dibandingkan penelitian-penelitian sebelumnya karena berfokus pada analisis *time series univariate* untuk memprediksi tren prevalensi *stunting* di Kota Tangerang dengan hanya menggunakan data angka *stunting* tanpa variabel lain. Pendekatan ini memberikan gambaran yang lebih murni terhadap pola perubahan angka *stunting* dari waktu ke

waktu. Selain itu, penelitian ini menerapkan kerangka kerja CRISP-DM, mulai dari tahap pemahaman data hingga implementasi hasil. Dari sisi teknis, penelitian ini menggunakan enam variasi model, yaitu ARIMA (manual dan auto), LSTM (manual dan dengan optimasi menggunakan Optuna), serta GRU (manual dan dengan Optuna) untuk mendapatkan hasil prediksi yang paling optimal. Pemilihan model-model tersebut didasarkan pada hasil penelusuran penelitian terdahulu di bidang *time series*, yang menunjukkan bahwa ARIMA, LSTM, dan GRU merupakan algoritma yang paling banyak digunakan. Proses data *smoothing* dilakukan karena data *stunting* yang digunakan bersifat sangat acak, sehingga perlu distabilkan agar pola tren dapat terbaca dengan jelas. Tahap evaluasi menggunakan metode *Validasi Walk-Forward* yang mensimulasikan kondisi prediksi nyata. Misalnya, model akan dilatih menggunakan data (Januari 2024 - April 2025) untuk memprediksi data uji (Mei 2025), kemudian dilatih lagi pada data (Januari 2024 - Mei 2025) untuk memprediksi (Juni 2025), dan begitu seterusnya, yang sesuai untuk karakteristik data *time series* untuk mencegah kebocoran informasi masa depan. Setelah itu, akan dinilai performa dari masing-masing model menggunakan empat metrik utama yaitu MAE, RMSE, MAPE, dan *R-Sqaured.* Model dengan performa terbaik, akan diimplementasikan dalam bentuk *dashboard* berbasis *website* menggunakan *framework CodeIgniter3. Dashboard* juga akan divalidasi oleh pihak terkait untuk menilai kelayakan dan fungsionalitasnya. Dengan demikian, penelitian ini tidak hanya kuat dari sisi metode karena menggunakan berbagai model dan proses optimasi, tetapi juga memberikan manfaat nyata melalui pembuatan *dashboard*. *Dashboard* ini diharapkan dapat mempermudah pemerintah daerah dalam memahami tren *stunting* dan diharapkan dapat mendukung upaya dalam mengurangi dan mencegah kasus *stunting* di Kota Tangerang.

2.2 Tinjauan Teori

Tinjauan teori berfungsi sebagai fondasi ilmiah pada penelitian ini. Bagian ini penting agar penelitian tidak hanya fokus pada hasil, tetapi juga memiliki landasan yang kuat. Dengan memahami teori yang relevan, penelitian dapat lebih terarah.

2.2.1 Stunting

Stunting adalah kondisi di mana anak-anak tidak tumbuh dengan baik, biasanya ditandai dengan tinggi badan yang lebih rendah dari standar anak-anak seusianya [27]. Hal ini disebabkan oleh kekurangan gizi. Pada usia dini, terutama dalam seribu hari pertama kehidupan (dari hamil hingga usia dua tahun), anak membutuhkan nutrisi yang sangat baik untuk mendukung pertumbuhannya. Jika anak kekurangan gizi pada masa-masa ini, dampaknya bisa sangat besar, seperti gangguan fisik dan kemampuan berpikir yang kurang baik, sehingga bisa memengaruhi kesehatan mereka di masa depan [28]. Oleh karena itu, penting untuk semua pihak, baik itu pemerintah, masyarakat, untuk bekerja sama dalam mengurangi *stunting* dan memastikan anak-anak bisa tumbuh sehat dan berkembang dengan baik.

2.2.2 Time Series

Time series adalah metode analisis data berdasarkan urutan waktu [29]. Analisis *time series* tidak hanya melihat jumlah kasus pada satu waktu tertentu, tetapi juga bagaimana perubahannya dari waktu ke waktu. Dalam penelitian ini, *time series* digunakan untuk membangun model tren *stunting* di Kota Tangerang, sehingga bisa diketahui pola kenaikan atau penurunan kasus di setiap kecamatan. Setelah model terbentuk melalui proses pelatihan dan pengujian, hasilnya dapat digunakan untuk melakukan prediksi tren *stunting*. Dengan pendekatan ini, pemerintah atau tenaga kesehatan bisa merencanakan langkah pencegahan yang lebih tepat.

2.2.3 Deep Learning

Deep learning menggunakan jaringan saraf tiruan dengan banyak lapisan untuk memproses data dalam jumlah besar [30]. Keunggulan dari *deep learning* adalah kemampuannya dalam mengenali pola yang kompleks [31]. Dalam konteks analisis *time series*, *deep learning* bisa digunakan untuk memprediksi tren berdasarkan data historis. Hal ini karena model *deep learning* mampu mengingat pola jangka panjang, sehingga penelitian ini menggunakan model seperti LSTM dan GRU.

2.2.4 Machine Learning

Machine learning atau pembelajaran mesin adalah cabang dari kecerdasan buatan yang memungkinkan komputer untuk belajar dari data dan membuat keputusan. Melalui pembelajaran mesin, komputer dapat mengenali pola, membuat prediksi seiring bertambahnya data yang dipelajari. Jenis-jenis *machine learning* dibedakan berdasarkan cara model belajar dan jenis data yang digunakan. Yang pertama ada *supervised learning* atau pembelajaran terarah merupakan metode di mana model belajar menggunakan data yang sudah memiliki label. Ketika model telah dilatih, ia dapat memprediksi hasil untuk data baru yang belum pernah dilihat. Yang kedua ada *unsupervised learning* atau pembelajaran tidak terarah yang digunakan untuk data yang tidak memiliki label. Tujuan utamanya bukan untuk memprediksi, melainkan untuk menemukan pola atau mengelompokkan data yang serupa. Yang ketiga adalah *semi-supervised learning* merupakan gabungan antara *supervised* dan *unsupervised learning*. Dalam metode ini, sebagian data memiliki label dan sebagian tidak. Yang terakhir adalah *reinforcement learning* yang merupakan metode pembelajaran berbasis pengalaman. Dalam metode ini, agen belajar mengambil keputusan terbaik melalui proses coba-coba. Jika keputusan yang diambil menghasilkan hasil baik, agen mendapatkan *reward*, jika buruk, ia mendapat *penalty* [32]. Dalam penelitian ini, metode yang digunakan termasuk ke dalam *supervised learning,* karena model ARIMA, LSTM, dan GRU dilatih menggunakan data historis prevalensi *stunting* untuk memprediksi nilai di periode berikutnya.

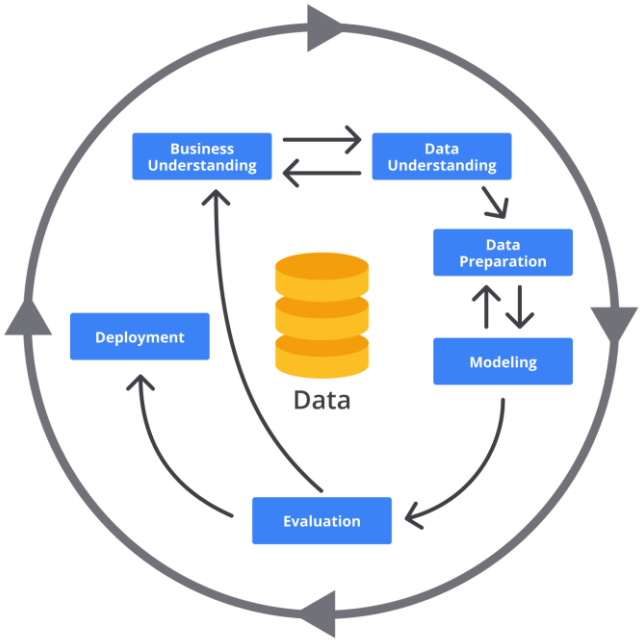
2.3 *Framework* dan Algoritma

Framework adalah kerangka kerja atau alat bantu yang digunakan untuk mempermudah proses pengembangan dan penerapan model. Algoritma adalah metode atau langkah-langkah yang digunakan untuk memproses data dan membuat prediksi. Dalam penelitian *stunting*, *framework* dan algoritma digunakan untuk menganalisis data secara efisien untuk menghasilkan hasil yang akurat.

2.3.1 *Framework* CRISP-DM

CRISP-DM adalah metodologi standar yang sering digunakan untuk proyek data mining atau pembelajaran mesin [33]. CRISP-DM memiliki

enam tahapan utama, yang saling berhubungan. Tahapan dari CRISP-DM dapat dilihat pada Gambar 2.1



Gambar 2.1 Framework CRISP-DM [16]

1) *Business Understanding*

Tahap ini bertujuan untuk memahami masalah yang ingin diselesaikan. Dalam konteks *stunting*, perlu dipahami bahwa *stunting* adalah masalah serius yang memengaruhi tumbuh kembang anak, sehingga penelitian ini berusaha mencari solusi dengan memanfaatkan data untuk membuat model tren *stunting*, agar pemerintah dapat mengambil langkah pencegahan sejak dini.

2) *Data Understanding*

Pada tahap ini, fokusnya adalah memahami data, seperti melakukan analisis awal dengan melihat distribusi data, dll. Tujuannya agar data tersebut cukup baik untuk dipakai dalam pemodelan.

3) *Data Preparation*

Tahap ini adalah proses menyiapkan data agar bisa diproses oleh model. Misalnya, melakukan pengecekan data kosong, menerapkan data

smoothing (Moving Average) untuk mengurangi *noise*, normalisasi data, pengecekan duplikat, dll. Kualitas data sangat penting karena akan sangat memengaruhi performa model.

4) *Modelling*

Setelah data siap, masuk ke tahap pemodelan. Di tahap ini, dipilih model yang cocok untuk modelling tren *stunting*. Model yang digunakan adalah enam variasi model yaitu ARIMA Manual, Auto ARIMA, LSTM Manual, LSTM dengan optimasi Optuna, GRU Manual, dan GRU dengan optimasi Optuna. Model akan dilatih menggunakan data historis untuk mengetahui pola dan membuat prediksi berdasarkan pola yang sudah dipelajari. Setelah itu, divalidasi menggunakan metode Validasi *Walk-Forward.* Metode ini meniru kondisi nyata di mana model dilatih menggunakan data masa lalu, kemudian diuji pada data berikutnya secara bertahap. Proses ini dilakukan berulang dengan terus memperbarui data pelatihan. Pada data deret waktu *(time series),* metode evaluasi standar seperti *train_test_split* tidak tepat karena dapat membuat model melihat data masa depan.

5) *Evaluation*

Setelah model dibuat, langkah berikutnya adalah mengevaluasi seberapa baik performanya. Tahap evaluasi sangat penting untuk memilih model yang paling akurat. Performa model diukur menggunakan metrik seperti MAE *(Mean Absolute Error)* dan RMSE *(Root Mean Squared Error),* MAPE *(Mean Absolute Percent Error)* dan *R-Squared.* Metrik ini membantu memastikan model sudah cukup akurat dalam melakukan modelling tren *stunting*.

6) *Deployment*

Tahap terakhir adalah mengimplementasikan model ke dunia nyata. Misalnya, membuat *dashboard* sederhana yang menampilkan angka *stunting* per kecamatan di Kota Tangerang dan proyeksi kedepannya. Hal ini akan membantu pemerintah atau pihak terkait untuk membuat kebijakan yang tepat.

2.3.2 Persiapan Data

Data yang akan digunakan dalam penelitian perlu dipersiapkan terlebih dahulu agar hasil analisis menjadi lebih akurat dan mudah dipahami oleh model. Proses persiapan data dilakukan dengan tujuan untuk mengurangi gangguan dari data mentah yang masih memiliki *noise* atau perbedaan skala antar variabel. Oleh karena itu, dilakukan beberapa langkah seperti data *smoothing* dan normalisasi data agar data menjadi lebih bersih, stabil, dan siap digunakan dalam proses pemodelan prediksi.

1) *Data Smoothing (Moving Average)*

   Data *smoothing* atau penghalusan data adalah proses yang digunakan untuk membuat data terlihat lebih stabil dan tidak terlalu berfluktuasi secara ekstrem. Dalam penelitian ini, teknik *Moving Average* digunakan dengan ukuran jendela (*window*) 3, yang berarti setiap nilai baru dihitung dari rata-rata tiga data di sekitarnya. Pemilihan *window* = 3 ini mengacu pada penelitian sebelumnya yang menggunakan ukuran jendela serupa untuk menghaluskan data [14]. Tujuannya adalah untuk mengurangi efek perubahan mendadak atau lonjakan acak pada data agar pola tren yang sebenarnya dapat terlihat lebih jelas. Dengan cara ini, data menjadi lebih stabil dan mudah dianalisis oleh model prediksi.

2) Normalisasi Data (*MinMaxScaler*)

   Normalisasi data dilakukan dengan *MinMaxScaler* supaya semua nilai berada dalam rentang 0 sampai 1. Langkah ini penting karena model seperti LSTM dan GRU bekerja lebih baik jika data yang masuk memiliki skala yang sama [34]. Dengan cara ini, proses pelatihan model jadi lebih cepat dan stabil. Selain itu, metode ini juga mudah digunakan karena tidak mengubah pola data aslinya, dan memudahkan proses pengembalian data ke skala awal setelah prediksi selesai.

2.3.3 Algoritma

Algoritma merupakan metode yang digunakan untuk memproses data agar dapat menghasilkan informasi. Dalam penelitian ini, algoritma dipilih untuk

melakukan modelling tren data *time series*, sehingga pola perubahan *stunting* dari waktu ke waktu dapat dianalisis dengan lebih akurat. Pemilihan algoritma yang tepat akan mempengaruhi kemampuan model dalam mengenali pola historis.

1) ARIMA

ARIMA merupakan salah satu metode yang banyak digunakan untuk memprediksi data *time series* atau deret waktu. ARIMA pertama kali dikenalkan oleh Box dan Jenkins pada tahun 1976 [35]. ARIMA sendiri terdiri dari tiga komponen utama yaitu AR *(AutoRegressive),* I *(Integrated),* dan MA *(Moving Average).*

Komponen yang pertama adalah AR, digunakan untuk melihat hubungan antara nilai data saat ini dengan beberapa nilai sebelumnya. Artinya, model mencoba untuk memprediksi nilai saat ini berdasarkan pola dari data masa lalu. Nilai p menunjukkan seberapa banyak data masa lalu *(lag)* yang digunakan. Rumus AR dapat dilihat pada Rumus 2.1.

$$y_t = C + \emptyset_1 y_{t-1} + \emptyset_2 y_{t-2} + \cdots + \emptyset_p + y_{t-p} + \varepsilon_t \quad [35]$$

Rumus 2.1 Rumus *AutoRegressive* (AR)

Komponen kedua adalah I, digunakan untuk membuat data menjadi stasioner, yaitu data yang tidak lagi memiliki tren naik-turun yang tajam. Caranya adalah dengan melakukan *differencing*, yaitu menghitung selisih antara data sekarang dan data sebelumnya. Untuk memastikan apakah data sudah benar-benar stasioner, dilakukan uji *Augmented Dickey-Fuller* (ADF *Test*) [36]. Uji ini membantu dalam menentukan berapa kali proses *differencing* perlu dilakukan. Jika hasil uji menunjukkan p-value < 0,05, maka data dianggap stasioner. Namun, jika nilainya lebih besar, maka proses differencing harus diulang hingga data menjadi stabil. Nilai d menunjukkan seberapa banyak proses *differencing* dilakukan agar data menjadi stabil.

Komponen ketiga adalah MA, digunakan untuk memperbaiki hasil prediksi dengan melihat kesalahan (*error)* dari periode sebelumnya. Jadi, model tidak hanya melihat nilai datanya, tetapi juga memperhitungkan

seberapa besar kesalahan prediksi sebelumnya. Nilai q menunjukkan jumlah error masa lalu yang digunakan dalam model. Rumus MA dapat dilihat pada Rumus 2.2.

$$y_t = C + \varepsilon_t + \theta_1\varepsilon_{t-1} + \theta_2\varepsilon_{t-2} + \cdots + \theta_q\epsilon_{t-q} \qquad [35]$$

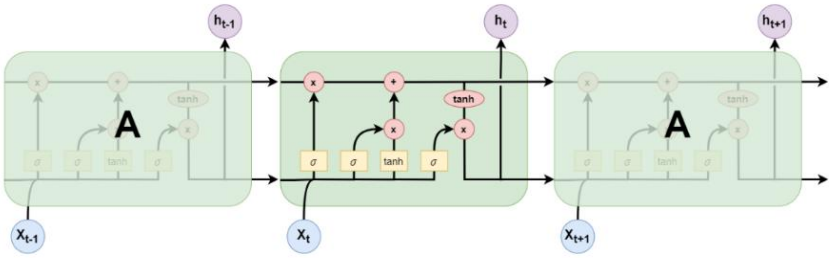
Rumus 2.2 Rumus *Moving Average* (MA)

Secara keseluruhan model ARIMA menggabungan ketiga komponen tersebut menjadi satu. Dengan kata lain, ARIMA berusaha untuk memprediksi nilai masa depan berdasarkan gabungan pola dari data masa lalu (AR), kestasioneran data (I), dan koreksi kesalahan sebelumnya (MA). Dalam penelitian ini, model ARIMA manual digunakan untuk menentukan orde p dan q dengan cara melihat pola dari plot ACF *(Autocorrelation Function)* dan PACF *(Partial Autocorrelation Function)* [37]. Pendekatan ini umum digunakan untuk mengenali hubungan antar data dalam deret waktu serta membantu memilih model yang paling sesuai untuk data yang sedang dianalisis.

Penelitian ini juga menggunakan Auto ARIMA yang merupakan versi otomatis dari model ARIMA menggunakan *library* pmdarima. Auto ARIMA ini membantu dalam mencari kombinasi parameter terbaik, yaitu (p, d, q), tanpa harus mencobanya satu per satu secara manual. Pemilihan model terbaik dilakukan dengan menggunakan ukuran *Akaike Information Criterion* (AIC). Semakin kecil nilai AIC, semakin baik model tersebut [38].

2) LSTM

LSTM merupakan sebuah sistem penyimpanan informasi dengan kemampuan untuk memilih mana yang penting dan mana yang harus dibuang. Di dalamnya terdapat jalur utama yang disebut *cell state*. Arsitektur LSTM dapat dilihat pada Gambar 2.2.



Gambar 2.2 Arsitektur LSTM [39]

LSTM mempunyai *forget gate*, yang bekerja seperti penyaring yang memutuskan informasi lama mana yang sudah tidak berguna. *Input gate* berfungsi untuk menentukan informasi baru yang perlu disimpan, dan *output gate* memilih informasi mana yang relevan untuk dikeluarkan sebagai hasil [39]. LSTM adalah algoritma yang dirancang untuk memahami data berdasarkan waktu [40]. Dalam konteks *stunting*, LSTM bisa digunakan untuk mempelajari pola jumlah kasus *stunting* dari waktu ke waktu, lalu memprediksi berapa kasus yang mungkin terjadi di masa depan.

Langkah pertama yang dilakukan adalah *forget gate*, yang bertugas untuk menentukan informasi yang masih relevan untuk disimpan di dalam memori, dan yang sudah tidak dibutuhkan dihapus. Rumus *forget gate* dapat dilihat pada Rumus 2.3.

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \qquad [39]$$

Rumus 2.3 Rumus *Forget Gate*

Langkah kedua adalah *input gate*, yang bertugas untuk menentukan informasi baru yang akan ditambahkan ke memori. Hasil dari keduanya digabungkan untuk memperbarui *cell state*, sehingga memori memiliki informasi yang lebih lengkap. Rumus *input gate* dapat dilihat pada Rumus 2.4.

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i \qquad [39]$$

$$\widetilde{C_t} = tanh(W_c[h_{t-1}, x_t] + b_c \qquad [39]$$

Rumus 2.4 Rumus *Input Gate*

Langkah berikutnya adalah memperbarui memori utama. Proses ini berisi kombinasi dari informasi lama yang masih relevan dan informasi baru yang penting untuk dipelajari. Rumus *update cell state* dapat dilihat pada Rumus 2.5.

$$C_t = f_t * C_{t-1} + i_t * \widetilde{C}_t \qquad [39]$$

Rumus 2.5 Rumus *Update Cell State*

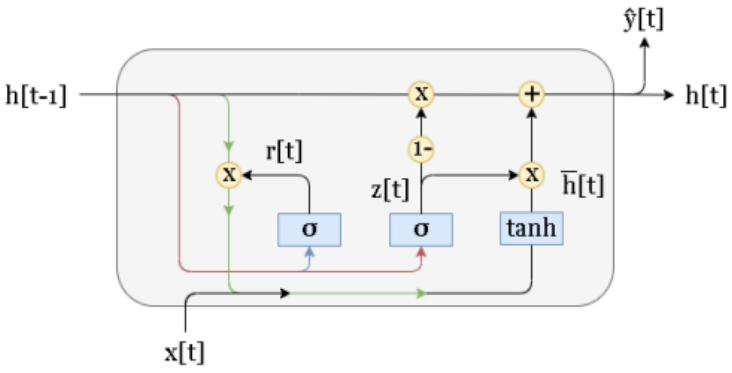
Langkah terakhir adalah menentukan hasil keluaran. Pada tahap ini, hasilnya digabung, sehingga keluaran yang dihasilkan benar-benar sesuai dengan informasi yang dianggap penting oleh model. Rumus *output gate* dapat dilihat pada Rumus 2.6.

$$h_t = o_t * \tanh(C_t) \qquad [39]$$

Rumus 2.6 Rumus *Output Gate*

3) GRU

GRU merupakan salah satu jenis jaringan saraf yang digunakan untuk memproses data deret waktu. Model ini merupakan pengembangan dari *Recurrent Neural Network* (RNN). GRU memiliki cara kerja yang mirip dengan LSTM, tetapi strukturnya lebih sederhana dan lebih cepat dilatih. Arsitektur GRU dapat dilihat pada Gambar 2.3



Gambar 2.3 Arsitektur GRU

GRU mempunyai dua gerbang yaitu *update gate* dan *reset gate. Update gate* berfungsi untuk menentukan informasi masa lalu mana yang masih perlu disimpan, sedangkan *reset gate* membantu untuk

menggabungkan informasi baru dengan memori lama agar model bisa menyesuaikan diri dengan perubahan data [41]. Karena lebih ringan, GRU sering digunakan untuk memprediksi data yang memiliki pola dari waktu ke waktu.

Langkah pertama adalah menghitung *reset gate*, yang berfungsi untuk menentukan seberapa banyak informasi dari masa lalu yang masih relevan dan perlu digunakan. Gerbang ini membantu model untuk melupakan informasi lama yang tidak penting, sehingga prediksi menjadi lebih akurat. Rumus *reset gate* dapat dilihat pada Rumus 2.7

$$r_t = \sigma(W_Y h_t + U_Y x_{t\,1} + b_Y) \quad [41]$$

Rumus 2.7 Rumus *Reset Gate*

Langkah selanjutnya setelah *reset gate* dihitung, model membuat memori baru yang disebut *candidate hidden state.* Bagian ini menggabungkan informasi dari data baru dan Sebagian informasi lama yang sudah disaring oleh *reset gate.* Proses ini memungkinkan model untuk memahami konteks data sekarang tanpa bergantung pada informasi masa lalu. Rumus *candidate hidden state* dapat dilihat pada Rumus 2.8

$$\bar{h}[t] = tanh(W(r_t * h_{t\,1}) + Ux_t + b) \quad [41]$$

Rumus 2.8 Rumus *Candidate Hidden State*

Langkah selanjutnya GRU menghitung *update gate,* yang bertugas untuk menentukan seberapa besar pengaruh informasi baru dibandingkan dengan informasi lama. Dengan kata lain, gerbang ini memutuskan bagian mana dari memori sebelumnya yang tetap disimpan dan mana yang perlu diperbarui. Rumus *update gate* dapat dilihat pada Rumus 2.9.

$$z_t = \sigma(W_z h_{t\,1} + U_Z x_t + b_{z)} \quad [41]$$

Rumus 2.9 Rumus *Update Gate*

Tahapan terakhir adalah menghitung *hidden state* final, yaitu hasil gabungan antara memori lama dan informasi baru. Nilai inilah yang akan

digunakan oleh model untuk membuat prediksi pada langkah waktu berikutnya. Rumus *hidden state* final dapat dilihat pada Rumus 2.10.

$$h_t = (1 - z_t) * h_{t\,1} + z_t * \bar{h}\,[t] \qquad [41]$$

Rumus 2.10 Rumus *Hidden State* Final

2.3.4 Optimasi, Validasi, dan Evaluasi Model

Tahap ini berfokus pada optimasi dan evaluasi model untuk memastikan hasil prediksi memiliki performa yang baik dan stabil. Proses ini penting agar model tidak hanya mampu menyesuaikan diri dengan data pelatihan, tetapi juga dapat memberikan hasil yang konsisten pada data baru.

1) *Hyperparameter Tuning* (Optuna)

   Hyperparameter tuning adalah proses untuk mencari kombinasi nilai parameter terbaik agar model dapat bekerja secara optimal. *Hyperparameter* sendiri merupakan nilai yang tidak dipelajari langsung oleh model, tetapi ditentukan sebelum proses pelatihan dimulai misalnya jumlah *neuron*, jumlah *epoch* dan *look_back*. Penentuan nilai ini sangat berpengaruh terhadap hasil pelatihan model, karena nilai yang kurang tepat dapat membuat model sulit mencapai hasil yang baik atau justru *overfitting*. Proses mencari nilai terbaik untuk *hyperparameter* sering kali rumit dan memakan waktu, karena perlu dilakukan banyak percobaan dengan berbagai kombinasi nilai [42].

   Proses pencarian kombinasi parameter terbaik pada penelitian ini menggunakan Optuna agar dapat dilakukan secara otomatis. Optuna merupakan alat optimasi yang mampu nilai *hyperparameter* paling optimal. Metode seperti *grid search* dan *random search* sering kali membutuhkan waktu yang lama karena mencoba banyak kemungkinan tanpa arah yang jelas [15]. Optuna memanfaatkan algoritma *Tree-structured Parzen Estimator* (TPE) untuk menelusuri ruang pencarian secara lebih terarah, sehingga proses pelatihan menjadi lebih cepat dan hasil prediksi lebih akurat tanpa perlu banyak percobaan manual.

2) Validasi *Walk-Forward*

Validasi *Walk-Forward* merupakan metode validasi yang sering digunakan dalam pemodelan data *time series.* Prinsip utamanya adalah melatih model dengan data di masa lalu, lalu menguji model pada data yang lebih baru. Validasi ini membantu dalam menilai kinerja model dalam kondisi yang mendekati dunia nyata, di mana model selalu belajar dari data yang sudah lewat dan memprediksi data di masa depan secara bertahap [24].

Metode validasi standar seperti *train_test_split* tidak cocok diterapkan pada data t*ime series* karena dapat membocorkan masa depan ke model. Penelitian ini menerapkan Validasi *Walk-Forward*, yang meniru kondisi nyata ketika data baru terus berdatangan dari waktu ke waktu. Prosesnya dimulai dengan melatih model pada sebagian data awal, lalu memprediksi satu titik data berikutnya. Setelah hasil prediksi dihitung, data baru tersebut dimasukkan ke dalam data pelatihan, dan model dilatih ulang. Langkah ini diulang secara terus-menerus, sehingga model selalu memprediksi data yang benar-benar baru. Metode ini membuat model menjadi lebih akurat dan mencerminkan kemampuan model dalam menghadapi data nyata yang terus berubah.

3) Metrik Evaluasi

Penilaian kinerja model dilakukan dengan menggunakan metrik evaluasi. Metrik ini berfungsi sebagai alat ukur yang menunjukkan seberapa jauh hasil prediksi model berbeda dari data sebenarnya. Nilai metrik evaluasi membantu dalam menentukan apakah model sudah akurat atau masih perlu disempurnakan. Berikut adalah beberapa metrik evaluasi yang digunakan:

a) MAE

MAE menghitung rata-rata dari selisih antara nilai sebenarnya dengan nilai prediksi, lalu diubah menjadi nilai absolut. Dengan cara ini, semua kesalahan dihitung sebagai angka positif. Semakin kecil nilai MAE, artinya hasil prediksi model semakin mendekati data sebenarnya. Rumus MAE dapat dilihat pada Rumus 2.11.

$$MAE = \frac{1}{N}\sum_{i=1}^{n} |f_i - y_i| \quad [39]$$

Rumus 2.11 Rumus MAE

b) RMSE

RMSE mengukur seberapa besar perbedaan antara nilai prediksi dengan nilai sebenarnya, tetapi kesalahannya dikuadratkan terlebih dahulu sebelum dihitung rata-ratanya, lalu diakarkan kembali. Jika nilai RMSE mendekati nol, berarti model mampu memberikan hasil prediksi yang lebih akurat. Rumus RMSE dapat dilihat pada Rumus 2.12.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \quad [39]$$

Rumus 2.12 Rumus RMSE

c) MAPE

MAPE adalah metode yang digunakan untuk mengukur seberapa besar kesalahan hasil prediksi dibandingkan dengan nilai sebenarnya, dalam bentuk persentase. Cara kerjanya adalah dengan menghitung rata-rata selisih antara nilai prediksi dan nilai sebenarnya, lalu membaginya dengan nilai sebenarnya dan dikalikan 100%. Semakin kecil nilai MAPE, berarti hasil prediksi semakin akurat karena tingkat kesalahannya rendah. Rumus MAPE dapat dilihat pada Rumus 2.13.

$$MAPE = \frac{\frac{\sum_{i=1}^{n} |y_i - \hat{y}_i|}{y_i}}{n} \; x\; 100\% \quad [39]$$

Rumus 2.13 Rumus MAPE

d) *R-Squared*

R-Squared menunjukkan seberapa baik model mampu menyesuaikan hasil prediksi dengan data aslinya. Nilai 1 berarti model dapat menggambarkan data dengan sangat baik, sedangkan nilai 0 menunjukkan bahwa model tidak lebih baik dari sekadar menebak rata-rata. Nilai negatif menandakan performa model jauh lebih buruk dibandingkan tebakan sederhana. Dalam penelitian ini,

nilai *R-Squared* digunakan untuk menilai keberhasilan model dalam memprediksi data *time series*, di mana nilai positif menunjukkan model bekerja dengan baik, sedangkan nilai negatif menandakan model gagal menggambarkan pola data secara akurat. Rumus *R-Squared* dapat dilihat pada Rumus 2.14.

$$R - Squared = 1 - \frac{\sum(\hat{Y}_i - \bar{Y})^2}{\sum(Y_i - \bar{Y})^2} \quad [43]$$

Rumus 2.14 Rumus R-Squared

2.3.5 Tools Penelitian

Tools atau alat bantu sangat penting untuk mendukung proses analisis data hingga pembuatan model. *Tools* penelitian berfungsi untuk mempermudah dalam mengelola data, penerapan algoritma, serta menampilkan hasil. Dengan adanya *tools* yang tepat, penelitian dapat dilakukan dengan lebih efisien dan mudah dipahami.

1) *Jupyter Notebook*

   Jupyter Notebook adalah platform interaktif berbasis *website* yang dirancang untuk memudahkan pengolahan data dan pemrograman. Dalam penelitian ini, *Jupyter Notebook* digunakan untuk mendokumentasikan proses penelitian secara terstruktur. Alat ini memungkinkan eksekusi kode secara interaktif dan langsung melihat hasil dari setiap langkah analisis [44]. Selain itu, fitur *markdown* di *Jupyter Notebook* mempermudah penulisan catatan, penjelasan, atau deskripsi terkait penelitian, sehingga semua langkah dapat terdokumentasi dengan baik di satu tempat. Dengan fleksibilitasnya, *Jupyter Notebook* dapat dijalankan secara lokal maupun melalui layanan berbasis *cloud* seperti *Google Colab*, yang memberikan kemudahan akses di berbagai perangkat.

2) *Visual Studio Code*

   Visual Studio Code merupakan *code* editor yang digunakan untuk menulis dan mengedit kode program [45]. Salah satu kelebihannya adalah kemampuannya untuk berjalan di berbagai sistem operasi seperti

Windows, MacOS, maupun Linux. Dengan menggunakan *Visual Studio Code*, proses pengembangan dan pengolahan data dalam penelitian dapat dilakukan lebih terstruktur dan efisien.

3) Laragon

Laragon adalah aplikasi yang berfungsi sebagai lingkungan pengembangan untuk membuat *website* maupun aplikasi dengan mudah. Cara kerjanya mirip dengan XAMPP [46]. Di dalamnya sudah terdapat beberapa kebutuhan seperti *web server*, *database*, dll. Dengan laragon, pengguna tidak perlu melakukan instalasi manual satu per satu, sehingga pekerjaan menjadi lebih efisien dan terorganisir.

4) Python

Python adalah bahasa pemrograman yang dikenal dengan sintaks yang sederhana dan mudah dipahami [47]. Dalam penelitian ini, Python digunakan karena kemampuannya dalam mengolah data, melakukan analisis, serta menerapkan algoritma. Pustaka seperti pandas dan numpy membantu dalam manipulasi data, mulai dari pembersihan hingga transformasi. Selain itu, pustaka visualisasi seperti matplotlib dan seaborn dapat digunakan untuk membuat grafik, sehingga analisis bisa ditampilkan dengan lebih informatif. *Statsmodels* dan *Pmdarima* digunakan untuk pemodelan statistik, seperti membangun model ARIMA manual dan Auto ARIMA otomatis. *Scikit-learn* membantu dalam persiapan data dan perhitungan metrik evaluasi. *TensorFlow* (Keras) dipakai untuk membangun dan melatih model *deep learning* seperti LSTM dan GRU [48]. Optuna berfungsi mencari *hyperparameter* terbaik secara otomatis.

5) *CodeIgniter3*

CodeIgniter3 merupakan salah satu *framework* PHP yang sangat populer dan banyak digunakan. *Framework* ini membantu dalam membuat aplikasi *website* dengan lebih cepat dan terstruktur menggunakan konsep MVC *(Model-View-Controller).* Struktur folder pada *CodeIgniter3* sudah diatur agar pengembangan aplikasi menjadi lebih rapi. Folder *controllers* untuk menyimpan file pengatur alur

program, folder *models* untuk mengelola data dan *database*, serta folder *views* untuk menyimpan tampilan yang akan dilihat pengguna. Selain itu, ada juga folder konfigurasi yang memungkinkan untuk mengatur pengaturan penting seperti koneksi *database* dan pengaturan *routing* URL [49].

6) *Flask*

Flask merupakan sebuah *framework website* yang populer dan banyak digunakan untuk membuat API *(Application Programming Interface). Flask* merupakan antarmuka yang menghubungkan satu aplikasi dengan aplikasi lain. Dengan menggunakan *flask,* preses membangun API dapat dilakukan dengan langkah sederhana, seperti menginstal *flask* melalui pip, membuat aplikasi dasar, mendefinisikan rute URL, dan menjalankan aplikasi [50]. Flask dikenal sebagai *framework* yang cepat, mudah, dan fleksibel dalam pengembangan API.