**SEMINAR PROPOSAL**

**IMPLEMENTASI ENSEMBLE LEARNING METODE EXTREME GRADIENT BOOSTING DAN RANDOM FOREST UNTUK PREDIKSI PENGGANTIAN BATERAI AKI PADA LAYANAN PURNA JUAL KALLA TOYOTA**

**Diajukan untuk Memenuhi Salah Satu Syarat Memperoleh Gelar**

**Sarjana Teknik**

****

**MUHAMAD AMHAR RAYADIN**

**E1E120037**

**JURUSAN TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK**

**UNIVERSITAS HALU OLEO**

**KENDARI**

**2024**

# LEMBAR PENGESAHAN

**Seminar Proposal**

**IMPLEMENTASI ENSEMBLE LEARNING METODE EXTREME GRADIENT BOOSTING DAN RANDOM FOREST UNTUK PREDIKSI PENGGANTIAN BATERAI AKI PADA LAYANAN PURNA JUAL KALLA TOYOTA**

Adalah benar dibuat oleh saya sendiri dan belum pernah dibuat dan diserahkan sebelumnya baik sebagian ataupun seluruhnya, baik oleh saya ataupun orang lain, baik di Universitas Halu Oleo ataupun institusi pendidikan lainnya.

Kendari, Maret 2024

**Muhamad Amhar Rayadin**

**NIM. E1E120037**

|  |  |
| --- | --- |
| **Pembimbing I**  **Ir. Mustarum Musaruddin, ST., MIT., Ph.D**  **NIP.** **19730122 200112 1 002** | **Pembimbing II**  **Rizal Adi Saputra, ST., M.Kom**  **NIP.19910406 201903 1 021** |

**Mengetahui,**

**Ketua Jurusan Teknik Informatika**

**Fakultas Teknik Universitas Halu Oleo**

**Isnawaty, S.Si., MT.**

**NIP.19761117 200812 2 001­**

# DAFTAR ISI

[LEMBAR PENGESAHAN i](#_Toc159707478)

[DAFTAR ISI ii](#_Toc159707479)

[INTISARI iv](#_Toc159707480)

[*ABSTRACT* v](#_Toc159707481)

[DAFTAR GAMBAR vi](#_Toc159707482)

[DAFTAR TABEL vii](#_Toc159707483)

[BAB I PENDAHULUAN 1](#_Toc159707484)

[1.1 Latar Belakang 1](#_Toc159707485)

[1.2 Rumusan Masalah 4](#_Toc159707486)

[1.3 Batasan Masalah 4](#_Toc159707487)

[1.4 Tujuan Penelitian 4](#_Toc159707488)

[1.5 Manfaat Penelitian 4](#_Toc159707489)

[1.6 Sistematika Penulisan 5](#_Toc159707490)

[1.7 Tinjauan Pustaka 5](#_Toc159707491)

[BAB II LANDASAN TEORI 8](#_Toc159707492)

[2.1 Prediksi 8](#_Toc159707493)

[2.2 Baterai Aki 9](#_Toc159707494)

[2.3 Data Tabular 10](#_Toc159707495)

[2.4 Exploratory Data Analysis 11](#_Toc159707496)

[2.5 Machine Learning 11](#_Toc159707497)

[2.6 Ensemble Learning 13](#_Toc159707498)

[2.7 Regression 19](#_Toc159707499)

[2.8 Extreme Gradient Boosting 20](#_Toc159707500)

[2.9 Random Forest 23](#_Toc159707501)

[2.10 Python 26](#_Toc159707502)

[2.11 Jupyter Notebook dan Google Colaboratory 27](#_Toc159707503)

[2.12 FastAPI 28](#_Toc159707504)

[2.13 Website 29](#_Toc159707505)

[2.14 Node.js 29](#_Toc159707506)

[2.15 Express.js 30](#_Toc159707507)

[2.16 Flowchart 31](#_Toc159707508)

[2.17 Unified Modelling Language 33](#_Toc159707509)

[2.18 Rational Unified Process 39](#_Toc159707510)

[2.19 Pengujian Black Box 40](#_Toc159707511)

[2.20 Pengujian Metrik Regresi 41](#_Toc159707512)

[BAB III METODOLOGI PENELITIAN 44](#_Toc159707513)

[3.1 Metode Pengumpulan Data 44](#_Toc159707514)

[3.2 Metode Pengembangan Sistem 45](#_Toc159707515)

[3.3 Waktu dan Tempat Penelitian 46](#_Toc159707516)

[3.4 Analisis Kebutuhan Sistem 46](#_Toc159707517)

[3.5 Perancangan Sistem 48](#_Toc159707518)

[3.6 Perancangan UML 52](#_Toc159707519)

[3.7 Perancangan Interface 55](#_Toc159707520)

[3.8 Pengujian Fungsionalitas Sistem 57](#_Toc159707521)

[3.9 Pengujian Model Ensemble Learning 58](#_Toc159707522)

[DAFTAR PUSTAKA 61](#_Toc159707523)

# INTISARI

**Muhamad Amhar Rayadin**, E1E120037

**IMPLEMENTASI ENSEMBLE LEARNING METODE EXTREME GRADIENT BOOSTING DAN RANDOM FOREST UNTUK PREDIKSI PENGGANTIAN BATERAI AKI PADA LAYANAN PURNA JUAL KALLA TOYOTA**

Proposal, Fakultas Teknik, 2024.

**Kata Kunci**: *Baterai Aki, Ensemble Learning, XGBoost, Random Forest, website.*

Industri otomotif merupakan salah satu kontributor besar dalam perekonomian nasional. Dalam kendaraan bermotor termasuk mobil, baterai aki (*accu*) sebagai salah satu suku cadang memegang peranan penting dalam kendaraan bermotor, yakni menjadi tempat menyimpan energi listrik*.* Pada layanan purna jual Kalla Toyota, khususnya pada saat pelanggan ingin melakukan panggantian suku cadang, masih ada masalah yang dihadapi. Hal ini karena pada saat sebelum penggantian suku cadang khususnya baterai aki, diperlukan pemeriksaan yang prosesnya relatif memakan waktu yang lama. Hal ini dapat menciptakan ketidaknyamanan bagi pelanggan, mengingat mereka harus menunggu lebih lama sebelum kendaraan mereka dapat kembali beroperasi secara optimal.

Untuk menyelesaikan masalah ini, dapat dilakukan prediksi waktu penggantian baterai aki kendaraan dengan menggunakan pendekatan *machine learning*. *Ensemble Learning* merupakan teknik kombinasi algoritma *machine learning*, dimana beberapa model machine learning digabungkan untuk memperoleh hasil prediksi yang lebih baik dibanding menggunakan model tunggal. Model tunggal yang dapat digunakan dalam teknik *ensemble* untuk prediksi adalah *Extreme Gradient Boosting* dan *Random Forest*.

Pengujian dilakukan dengan menguji fungsionalitas sistem secara umum menggunakan *black box*. Performa model juga diuji menggunakan metric evaluasi R2 *score,* MSE, RMSE dan MAE. Dengan prediksi waktu penggantian baterai aki, dapat mempercepat proses kerja pada layanan *aftersales* dan perusahaan dapat menerapkan strategi bisnis berdasarkan hasil prediksi untuk setiap pelanggan.

# *ABSTRACT*

**Muhamad Amhar Rayadin**, E1E120037

***IMPLEMENTATION OF ENSEMBLE LEARNING EXTREME GRADIENT BOOSTING AND RANDOM FOREST METHODS FOR BATTERY REPLACEMENT PREDICTION AT KALLA TOYOTA AFTERSALES SERVICE***

*Proposal*, *Faculty of Engineering, 2024.*

***Keyword***: *Accu Battery, Ensemble Learning, XGBoost, Random Forest, website.*

*The automotive industry is one of the major contributors to the national economy. In motorized vehicles including cars, batteries as one of the spare parts play an important role in motorized vehicles, which is a place to store electrical energy. In Kalla Toyota's after-sales service, especially when customers want to replace spare parts, there are still problems faced. This is because before replacing spare parts, especially battery batteries, an inspection is required, which is a relatively time-consuming process. This can create inconvenience for customers, considering they have to wait longer before their vehicle can return to optimal operation.*

*To solve this problem, it is possible to predict the replacement time of a vehicle battery using a machine learning approach. Ensemble Learning is a combination technique of machine learning algorithms, where several machine learning models are combined to obtain better prediction results than using a single model. Single models that can be used in ensemble techniques for prediction are Extreme Gradient Boosting and Random Forest.*

*Testing is done by testing the general functionality of the system using black box. Model performance is also tested using R2 score, MSE, RMSE and MAE evaluation metrics. With the prediction of battery replacement time, it can speed up the work process in aftersales services and companies can implement business strategies based on the prediction results for each customer.*

# DAFTAR GAMBAR

[Gambar 2. 1 Baterai aki 9](#_Toc159707598)

[Gambar 2. 2 Baterai kering dan baterai basah 10](#_Toc159707599)

[Gambar 2. 3 Tahapan *Eksploratory Data Analysis* 11](#_Toc159707600)

[Gambar 2. 4 *Traditional programming vs machine learning* 12](#_Toc159707601)

[Gambar 2. 5 Jenis *Machine Learning* 13](#_Toc159707602)

[Gambar 2. 6 Metode *voting* pada *ensemble learning* (Peppes dkk, 2021) 15](#_Toc159707603)

[Gambar 2. 7 Metode *bagging* (Choi dan Hur, 2020) 16](#_Toc159707604)

[Gambar 2. 8 Metode *boosting* (Budu, 2023) 18](#_Toc159707605)

[Gambar 2. 9 Metode *stacking* (Shukla, 2022) 19](#_Toc159707606)

[Gambar 2. 10 Regresi linear 20](#_Toc159707607)

[Gambar 2. 11 Algoritma *XGBoost* 22](#_Toc159707608)

[Gambar 2. 12 Contoh *decision tree* 24](#_Toc159707609)

[Gambar 2. 13 Alur *random forest* untuk regresi 24](#_Toc159707610)

[Gambar 2. 14 Python 26](#_Toc159707611)

[Gambar 2. 15 Jupyter Notebook 27](#_Toc159707612)

[Gambar 2. 16 Google Colaboratory 28](#_Toc159707613)

[Gambar 2. 17 FastAPI 29](#_Toc159707614)

[Gambar 2. 18 Node.js 30](#_Toc159707615)

[Gambar 2. 19 Express.js 31](#_Toc159707616)

[Gambar 2. 20 Fase pengembangan sistem RUP 40](#_Toc159707617)

[Gambar 3. 1 Alur sistem secara umum 48](#_Toc159707618)

[Gambar 3. 2 Alur Proses *Modelling* 49](#_Toc159707619)

[Gambar 3. 3 Alur *Ensemble Learning* 51](#_Toc159707620)

[Gambar 3. 4 *Use case diagram* sistem 52](#_Toc159707621)

[Gambar 3. 5 *Activity diagram* prediksi waktu penggantian 53](#_Toc159707622)

[Gambar 3. 6 *Activity diagram* kelola saran aksi 53](#_Toc159707623)

[Gambar 3. 7 *Sequence diagram* prediksi waktu penggantian 54](#_Toc159707624)

[Gambar 3. 8 *Sequence diagram* kelola saran aksi 54](#_Toc159707625)

[Gambar 3. 9 *Class diagram* sistem 55](#_Toc159707626)

[Gambar 3. 10 Halaman form prediksi penggantian baterai aki 55](#_Toc159707627)

[Gambar 3. 11 Halaman hasil prediksi 56](#_Toc159707628)

[Gambar 3. 12 Halaman kelola saran aksi 56](#_Toc159707629)

[Gambar 3. 13 Halaman menambahkan saran aksi 57](#_Toc159707630)

# DAFTAR TABEL

[Tabel 2. 1 Simbol – simbol *flowchart* 32](#_Toc159707650)

[Tabel 2. 2 Simbol – simbol *use case diagram* 34](#_Toc159707651)

[Tabel 2. 3 Simbol – simbol *activity diagram* 36](#_Toc159707652)

[Tabel 2. 4 Simbol – simbol *sequence diagram* 37](#_Toc159707653)

[Tabel 2. 5 Simbol – simbol *class diagram* 38](#_Toc159707654)

[Tabel 3. 1 Waktu Penelitian 46](#_Toc159707675)

[Tabel 3. 2 Spesifikasi Perangkat Keras 47](#_Toc159707676)

[Tabel 3. 3 Spesifikasi Perangkat Lunak 48](#_Toc159707677)

[Tabel 3. 4 Skema pengujian *black box* 57](#_Toc159707678)

[Tabel 3. 5 Data *dummy* penggantian baterai aki 59](#_Toc159707679)

[Tabel 3. 6 Contoh skenario hasil pengujian model *ensemble learning* 60](#_Toc159707680)

# BAB I PENDAHULUAN

1. Latar Belakang

Industri otomotif merupakan salah satu kontributor besar dalam perekonomian nasional. Saat ini, terdapat 22 perusahaan industri kendaraan bermotor roda empat atau lebih yang ada di Indonesia. Ini menyumbangkan nilai investasi sebesar Rp99,16 triliun dengan total kapasitas produksi mencapai 2,35 juta unit per tahun dan menyerap tenaga kerja langsung sebanyak 38,39 ribu orang (Kementerian Perindustrian Republik Indonesia, 2021). Keberadaan industri ini tidak sekadar menciptakan lapangan kerja dan kontribusi ekonomi, melainkan juga membentuk fondasi transportasi modern di masyarakat. Kendaraan bermotor, sebagai elemen utama dalam mobilitas sehari-hari, tidak hanya memberikan kemudahan, tetapi juga menggambarkan dinamika perkembangan zaman.

Dalam kendaraan bermotor termasuk mobil, baterai aki (*accu*) sebagai salah satu suku cadang memegang peranan penting dalam kendaraan bermotor, yakni menjadi tempat menyimpan energi listrik. Baterai aki berfungsi sebagai sumber kelistrikan utama yang digunakan sebelum mesin kendaraan menyala. Saat mesin kendaran menyala, alternator menggantikan baterai aki menjadi sumber energi listrik. Akan tetapi, baterai aki tetap memiliki fungsi sebagai stabilitator tegangan listrik (Firdaus, Rustendi dan Herdiana, 2021).

Pada umumnya pengguna kendaraan bermotor tidak mengetahui kondisi baterai aki di kendaraannya (Fahmi dan Arsana, 2021). Padahal, penggunaan baterai aki yang sudah dalam keadaan kurang baik atau rusak dapat menggangu performa kendaraan yang digunakan. Kualitas baterai aki tidak hanya memengaruhi kemampuan kendaraan untuk menyala, tetapi juga berkontribusi pada efisiensi konsumsi bahan bakar dan stabilitas sistem kelistrikan. Menjaga kualitas baterai aki menjadi kunci dalam memastikan kendaraan tetap handal dan efisien.

Layanan purna jual sangat penting dalam memelihara dan meningkatkan performa kendaraan. Layanan purna jual mencakup adanya garansi, layanan yang berupa pra-instalasi, hingga pergantian akan suku cadang yang rusak (Eleanora dan Fitriana, 2023). Layanan purna jual menyediakan suku cadang yang berkualitas sehingga memudahkan konsumen untuk mengganti suku cadang saat terjadi kerusakan (Wibisono, 2020).

Kalla Toyota, salah satu unit bisnis Kalla Group (PT. Hadji Kalla) sejak berdiri pada 1952 hingga saat ini menjadi market leader roda empat di wilayah Sulawesi. Selain bergerak di bidang *sales*, Kalla Toyota juga memiliki layanan purna jual (aftersales) yakni *General Repair* dan *Body Paint* (Kalla Group, 2024). Layanan purna jual Kalla Toyota mencakup pemeriksaan kendaraan, jasa servis kendaraan, hingga penggantian dan pembelian sparepart. Layanan ini bermanfaat untuk memelihara dan meningkatkan performa kendaraan bermotor, termasuk dalam hal menjaga kualitas suku cadang kendaraan agar kendaraan dapat berfungsi optimal.

Pada layanan purna jual Kalla Toyota, khususnya pada saat pelanggan ingin melakukan panggantian suku cadang, masih ada masalah yang dihadapi. Hal ini karena pada saat sebelum penggantian suku cadang khususnya baterai aki, diperlukan pemeriksaan yang prosesnya relatif memakan waktu yang lama. Hal ini dapat menciptakan ketidaknyamanan bagi pelanggan, mengingat mereka harus menunggu lebih lama sebelum kendaraan mereka dapat kembali beroperasi secara optimal. Proses pemeriksaan yang memakan waktu dapat disebabkan oleh berbagai faktor, seperti tingginya volume layanan, keterbatasan tenaga kerja, atau kompleksitas pemeriksaan yang dilakukan untuk memastikan bahwa penggantian suku cadang dilakukan dengan akurat dan sesuai kebutuhan kendaraan.

Waktu yang dibutuhkan untuk proses pemeriksaan dan penggantian baterai aki dapat memiliki dampak langsung pada kepuasan pelanggan. Pelanggan, terutama yang memiliki mobilitas tinggi dalam kegiatan sehari-hari, mungkin menganggap penundaan ini sebagai ketidaknyamanan yang dapat merugikan. Oleh karena itu, efisiensi dan kecepatan dalam menangani penggantian baterai aki menjadi krusial untuk memastikan kepuasan pelanggan yang optimal.

Melihat tantangan mengenai efisiensi waktu pelayanan purna jual dan peran kunci baterai aki sebagai sumber kelistrikan pada kendaraan, perlu dilakukan prediksi waktu penggantian suku cadang, dalam hal ini baterai aki. Prediksi waktu penggantian baterai aki dapat menjadi acuan bagi pelanggan maupun pekerja di layanan purna jual untuk menentukan kelayakan baterai aki untuk diperiksa secara cepat.

*XGBoost Regression (Extreme Gradient Boosting Regression)* merupakan salah satu algoritma *machine learning* yang biasa digunakan untuk prediksi (Prasetyo, Christianto dan Hartomo, 2019). Dalam penelitian yang dilakukan oleh J. Avanija dkk (2021), algoritma regresi *XGBoost* digunakan untuk memprediksi harga rumah. Dataset dibagi menjadi tiga subset untuk pelatihan, validasi, dan pengujian. Setelah melatih setiap model, model tersebut disesuaikan dengan parameter regularisasi terbaik untuk menghindari overfitting. Hasil percobaan menunjukkan bahwa pembagian data pelatihan, pengujian, dan validasi sebesar 80%, 10%, 10% menunjukkan tingkat kesalahan rendah.

Selain *XGBoost*, algoritma machine learning lainnya yang digunakan untuk prediksi adalah *Random Forest Regressor*. *Random Forest Regressor* adalah *tool* yang populer digunakan untuk analisis data dengan dimensi yang besar (Borup dkk, 2023). Dalam penelitian yang dilakuka oleh Fachid dan Triayudi (2022), random forest digunakan untuk prediksi kasus positif Covid-19. Dari hasil penelitian yang dilakukan, *Random Forest* menghasilkan nilai RMSE sebesar 1886,555 dan MAPE 14,85. Ini lebih baik dari algoritma lainnya yakni regresi linear yang menghasilkan nilai RMSE sebesar 3031,127 dan MAPE 47,66.

*Ensemble Learning* merupakan teknik kombinasi algoritma *machine learning*, dimana beberapa model machine learning digabungkan untuk memperoleh hasil prediksi yang lebih baik dibanding menggunakan model tunggal (Dachi dan Sitompul, 2023; Sunarko dkk, 2023). Berdasarkan penelitian yang dilakukan oleh Indahyanti, Azizah dan Setiawan (2022), penggunaan teknik *majority vote* dari ensemble learning yang menerapkan beberapa algoritma ke dalam sebuah pemodelan dapat meningkatkan akurasi prediksi kinerja akademik mahasiswa dengan nilai RMSE sebesar 0,440.

Berdasarkan latar belakang dan penelitian terdahulu yang telah diuraikan maka Penulis mengambil topik penelitian dengan judul “**Implementasi *Ensemble Learning* Metode *Extreme Gradient Boosting* dan *Random Forest* Untuk Prediksi Penggantian Baterai Aki Pada Layanan Purna Jual Kalla Toyota**”. Dengan menggunakan teknik *ensemble learning* dari 2 model kuat yakni *XGBoost* dan *Random Forest*, diharapkan dapat menghasilkan prediksi waktu penggantian baterai aki yang akurat.

1. Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan di atas, rumusan masalah dalam penelitian ini adalah bagaimana mengimplementasikan e*ansemble learning* metode *Extreme Gradient Boosting* dan *Random Forest* untuk memprediksi waktu penggantian baterai aki pada layanan purna jual Kalla Toyota?

1. Batasan Masalah

Berdasarkan rumusan masalah di atas, batasan masalah dalam penelitian ini adalah sebagai berikut :

1. Penelitian ini berfokus pada prediksi waktu penggantian baterai aki mobil berdasarkan data historis penggantian.
2. Penelitian ini menggunakan data dari layanan purna jual Kalla Toyota.
3. Penelitian ini menggunakan pendekatan *ensemble learning* metode *Extreme Gradient Boosting* dan *Random Forest*.
4. Tujuan Penelitian

Adapun tujuan dari penelitian ini adalah untuk memprediksi waktu penggantian baterai aki pada layanan purna jual Kalla Toyota menggunakan *ensemble learning* metode *Extreme Gradient Boosting* dan *Random Forest*.

1. Manfaat Penelitian

Manfaat penelitian ini adalah dapat mengimplementasikan *ensemble learning* metode Extreme *Gradient Boosting* dan *Random Forest* untuk memprediksi waktu penggantian baterai aki pada layanan purna jual Kalla Toyota.

1. Sistematika Penulisan

Sistematika penulisan dalam penyusunan penelitian ini adalah sebagai berikut:

**BAB I PENDAHULUAN**

Bab ini berisi tentang deskripsi umum isi laporan yang meliputi latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, sistematika penulisan dan tinjauan pustaka.

**BAB II LANDASAN TEORI**

Bab ini berisi teori-teori penunjang yang berhubungan dengan penelitian ini berupa teori yang bersumber dari media cetak maupun media elektronik berupa pengertian prediksi, baterai aki, *machine learning, ensemble learning, regression*, *Extreme Gradient Boosting, Random Forest*, Python, Jupiter Notebook, *website*, SQL, UML, dan Node.js dan pendukung lainnya.

**BAB III METODOLOGI PENELITIAN**

Bab ini berisi tahapan tentang proses dan prosedur pengumpulan data, prosedur analisis metode, prosedur pengembangan sistem serta waktu dan tempat penelitian

1. Tinjauan Pustaka

Penelitian berjudul *Ensemble Machine Learning of Random Forest, AdaBoost and XGBoost for Vertical Total Electron Content Forecasting* yang dilakukan oleh Natras, Soja dan Schmidt (2022). Studi ini menyelidiki algoritma pembelajaran yang berbeda untuk memperkirakan proses cuaca luar angkasa *nonlinier* dan memperkirakan *Vertical Total Electron Content* (VTEC) untuk 1 jam dan 24 jam di masa depan untuk titik-titik *grid* ionosfer lintang rendah, menengah dan tinggi di sepanjang garis bujur yang sama. Model VTEC dikembangkan dengan menggunakan algoritma pembelajaran Decision Tree dan ensemble learning dari *Random Forest, Adaptive Boosting* (*AdaBoost*), dan *Extreme Gradient Boosting (XGBoost)*. Selanjutnya, model-model *ensemble* digabungkan menjadi satu *meta-model Voting Regressor*. Hasilnya menunjukkan bahwa model yang dikembangkan berkinerja baik dalam kondisi tenang dan badai, di mana *multi-tree ensemble learning* mengungguli *Decision Tree*. Secara khusus, *meta-estimator Voting Regressor* memberikan sebagian besar RMSE terendah untuk lintang rendah, menengah dan tinggi berturut – turut yaitu 1,09, 0,86 dan 0,69 untuk 1 jam dan 2,15, 1,88, dan 1,08 untuk 24 jam.

Penelitian yang dilakukan oleh Riza (2022) berjudul *Analisis dan Prediksi Data Penjualan Menggunakan Machine Learning dengan Pendekatan Ilmu Data*. Penelitian ini membahas tentang analisis dan prediksi data penjualan menggunakan pendekatan ilmu data dan algoritma *Machine Learning*. Hasil dari penelitian ini menunjukkan bahwa algoritma *XGBoost* memberikan hasil prediksi yang lebih baik dibandingkan dengan algoritma lainnya, dengan nilai evaluasi kinerja seperti RMSE sebesar 1153,23, MEA sebesar 893,23, dan R2 sebesar 0,56.

Berdasarkan penelitian berjudul *Optimasi Prediksi Kelulusan Tepat Waktu: Studi Perbandingan Algoritma Random Forest dan Algoritma K-NN Berbasis PSO* yang dilakukan oleh Irawan dkk (2023), algoritma *Random Forest* memperoleh nilai akurasi dan AUC yang lebih baik dari algoritma K-NN setelah dilakukan optimasi PSO dengan nilai 97,89% dan 0,993 vs 96,74% dan 0,986.

Menurut penelitian yang dilakukan oleh Lubis, Harahap dan Husein, (2019) yang berjudul *Analisis Prediktif untuk Keputusan Bisnis : Peramalan Penjualan*, semua hasil peramalan model yang digunakan dalam penelitian ini termasuk *XGBoost* dan *Random Forest* berguna untuk membantu dalam menentukan keputusan bisnis berdasarkan peramalan penjualan, dimana hasil pengujian RMSE untuk algortima *XGBoost* dan *Random Forest* yaitu 15701,00336 dan 17648,48278. Pengujian menggunakan MAE memberikan hasil evaluasi 13342,66667 untuk *XGBoost* dan 14621,75 untuk *Random Forest*. Adapun pengujian dengan R2 score memberikan nilai 99,13% dan 98,9% untuk model *XGBoost* dan *Random Forest*.

Penelitian lainnya dilakukan oleh Kiangala dan Wang (2021) berjudul *An Effective Adaptive Customization Framework for Small Manufacturing Plants using Extreme Gradient Boosting-XGBoost dnd Random Forest Ensemble Learning Algorithms in an Industry 4.0 Environment*. Objek penelitian ini adalah pabrik kecil permen dan cokelat yang ingin memperkenalkan konsep kustomisasi produk dengan algoritma *XGBoost* dan *Random Forest*. Hasilnya menunjukkan bahwa model regresi yang dibuat dengan *XGBoost* memperolah *error rate* terbaik yaitu 0,031%, sedangkan model klasifikasi dengan *random forest* memperoleh akurasi terbaik yakni 99,4%.

Berdasarkan dari tinjauan pustaka yang telah dikaji bahwa metode *Extreme Gradient Boosting* dan *Random Forest* merupakan algoritma prediksi dengan performa yang baik dan nilai *error* yang rendah, maka diusulkan **Implementasi Ensemble Learning Metode *Extreme Gradient Boosting* dan *Random Forest* untuk Prediksi Penggantian Baterai Aki pada Layanan Purna Jual Kalla Toyota** dengan harapan pada penelitian ini mencoba untuk mengimplementasikan teknik ensemble learning dari dua metode tersebut untuk memperoleh prediksi waktu penggantian baterai aki dengan performa yang lebih baik.

# BAB II LANDASAN TEORI

1. Prediksi

Prediksi merupakan suatu metode untuk meramalkan nilai suatu variabel pada masa mendatang dengan merujuk pada informasi yang ada pada periode sebelumnya. Dalam proses ini, biasanya data kuantitatif menjadi landasan utama untuk membuat estimasi terhadap peristiwa yang akan datang. Sebagai suatu bentuk analisis, prediksi tidak bersifat mutlak dan tidak selalu memberikan jawaban yang pasti terkait dengan kejadian yang akan terjadi. Sebaliknya, tujuan utamanya adalah mencari perkiraan seakurat mungkin berdasarkan informasi yang tersedia. Istilah prediksi sama dengan ramalan atau perkiraan (*forecast*) (Adiguno, Syahra dan Yetri, 2022).

Menurut kamus besar bahasa Indonesia, prediksi adalah hasil dari kegiatan memprediksi atau meramal atau memperkirakan nilai pada masa yang akan datang dengan menggunakan data masa lalu. Prediksi memberikan gambaran mengenai kemungkinan peristiwa di suatu kondisi tertentu dan berfungsi sebagai masukan untuk proses perencanaan dan pengambilan keputusan. Prediksi bisa berdasarkan metode ilmiah ataupun subjektif belaka (Kafil, 2019). Sebagai contoh, prediksi pergerakan pasar saham sering kali didasarkan pada analisis data historis dan faktor-faktor ekonomi terkini yang memengaruhi pasar. Contoh lainnya dapat ditemukan dalam prediksi hasil pemilihan umum, di mana analisis *polling* dan data suara sebelumnya digunakan untuk mencoba memproyeksikan hasil yang mungkin. Dengan kata lain, prediksi tidak hanya terbatas pada aspek ilmiah, tetapi juga dapat melibatkan elemen-elemen subjektif seperti pendapat ahli atau persepsi masyarakat. Dalam dunia bisnis, prediksi penjualan atau tren konsumen dapat menjadi landasan bagi perencanaan strategis perusahaan.

1. Baterai Aki

Baterai aki adalah sebuah penyimpan sumber energi listrik yang sangat dibutuhkan ada semua aspek kendaraan bermotor. Baterai aki atau dalam bahasa inggris disebut accu berkaitan erat dibidang industri dan otomotif (Asfan dan Arsana, 2021). Listrik yang tersimpan dalam baterai aki diperlukan untuk proses pembakaran pada kendaraan bermotor, termasuk mobil.



Gambar 2. 1 Baterai aki

Fungsi baterai aki (*accu*) antara lain :

1. Sebagai sumber tenaga listrik utama saat mesin belum menyala untuk menyalakan sistem kelistrikan kendaraan, misalnya lampu *sein* (lampu tanda belok), klakson (*horn*), indikator pada *dashboard*, dan sebagainya.
2. Pada saat melakukan *starter* menggunakan elektrik *starter*, baterai aki juga merupakan satu-satunya sumber listrik yang digunakan untuk menghidupkan atau menjalankan *motor starter*.
3. Pada saat mesin kendaraan hidup, baterai aki berfungsi sebagai stabilitator tegangan dari alternator. Mengingat kerja altenator kurang stabil dan bergantung dengan putaran altenator, walaupun pada alternator sudah dilengkapi dengan regulator namun masih memerlukan aki sebagai penstabil tegangan yang akan dialirkan ke sistem kelistrikan pada kendaraan (Firdaus, Rustendi dan Herdiana, 2021).

Ada dua jenis aki yang digunakan masyarakat untuk kendaraan yaitu aki basah dan aki kering (aki *maintenance free*) (Karimah, Zain dan Nofiansyah, 2023). Aki basah memiliki cairan elektrolit yang berada di dalamnya. Cairan tersebut terdiri dari campuran air dan asam sulfat (H2SO4) yang masyarakat secara umum mengenalnya sebagai air aki atau *accu* *zuur*. Fungsi utama dari accu zuur ini adalah merendam sel-sel pada aki basah. Pada dasarnya, semua jenis aki (kering dan basah) tetap menggunakan cairan elektrolit. Namun, pada aki kering, kadar berat jenis elektrolitnya lebih pekat atau kental. Aki kering di desain khusus untuk mencegah penguapan air dari dalam aki, sehingga penambahan air tidak lagi diperlukan. Karena tidak memerlukan penambahan air, maka aki jenis ini disebut aki *Maintenanfe Free* (MF) atau bebas perawatan.



Gambar 2. 2 Baterai kering dan baterai basah

1. Data Tabular

Data tabular adalah jenis data yang disusun dalam bentuk tabel dengan baris dan kolom. Setiap baris dalam tabel mewakili satu contoh atau sampel, sedangkan setiap kolom mewakili atribut atau fitur dari contoh tersebut. Data tabular sering digunakan dalam berbagai bidang seperti keuangan, kesehatan, penjualan, dan lainnya yang menggunakan struktur data terstruktur (Dian Rosadi, SKM dkk, 2021).

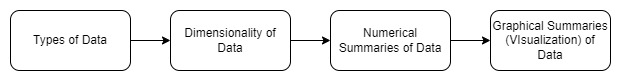
Data tabular dapat terdiri dari berbagai jenis atribut, termasuk numerik (misalnya, angka), ordinal (misalnya, tingkat pendidikan), dan kategorikal (misalnya, jenis kelamin). Data tabular juga dapat menghadapi tantangan seperti kekurangan data (*missing values*), ketidakteraturan, dan kompleksitas struktur data yang tidak selalu jelas seperti pada data teks atau gambar.

Dalam analisis data tabular, penting untuk memilih model yang sesuai untuk memproses data tersebut. Model *machine learning* tradisional seperti *XGBoost* sering digunakan dan dianggap unggul dalam pemodelan data tabular. Bahkan, belakangan ini telah ada upaya untuk mengembangkan model *deep learning* khusus untuk data tabular (Shwartz-Ziv dan Armon, 2022).

1. *Exploratory Data Analysis*

*Exploratory Data Analysis* (EDA) merupakan langkah awal yang krusial dalam merancang sebuah model, membentuk fondasi penting dalam dunia *data science*. Proses ini mencerminkan pendekatan statistik yang digunakan untuk menganalisis data konfirmatori, yang berkaitan dengan pembangunan model dan pengujian hipotesis (Tanuwidjaja dan Widjaja, 2022).

Proses EDA dilakukan dengan beberapa tahapan. Langkah pertama, yaitu *Types of Data* atau mengetahui tipe data dari masing-masing *field* (kolom) yang akan digunakan, apakah data tersebut tergolong kategorikal atau kuantitatif. Langkah selanjutnya adalah *Dimensionality of Data* atau mengetahui dimensi data, apakah data tersebut berdimensi satu (*univariate*), berdimensi dua (*bivariate*) atau berdimensi banyak (*multivariate*). Lalu dilakukan *Numerical Summaries of Data* atau melihat rangkuman numerik dari *datasets* seperti nilai *mean, median, standar deviation* dan lain-lain guna mengetahui kondisi statistik dari data. Tahap akhir yaitu *Graphical Summaries (Visualization) of Data* atau melakukan visualisasi guna mendapatkan pola-pola dan informasi menarik terkait dataset yang sedang dianalisis. Pemilihan grafik/chart disesuaikan dengan tujuan informasi yang ingin didapatkan, tipe dan dimensi data yang telah diketahui dari proses sebelumnya (Nissa dkk, 2020).



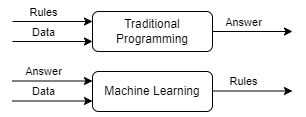
Gambar 2. 3 Tahapan *Eksploratory Data Analysis*

1. *Machine Learning*

Pembelajaran mesin (*machine learning*) adalah ilmu atau studi yang mempelajari tentang algortima dan model statistik yang digunakan oleh sistem komputer dalam mengerjakan task tertentu tanpa instruksi secara eksplisit (Aditya dkk, 2020). Dalam praktiknya, komputer "belajar" dari data, meningkatkan kinerjanya dalam suatu tugas melalui pengalaman yang diperoleh dari data yang diprosesnya. Proses belajar dari data ini memungkinkan algoritma machine learning untuk mengidentifikasi pola, membuat prediksi, dan mengoptimalkan hasil (Bi dkk, 2019).

Pada paradigma pemrograman tradisional, aturan dan data adalah masukan atau input bagi sistem. Secara eksplisit, aturan diekspresikan dalam bahasa pemrograman. Tambahan masukan berupa data kemudian akan menghasilkan solusi sebagai keluaran. Paradigma pemrograman seperti pada diagram di atas sering disebut sebagai pemrograman tradisional. Pemrograman tradisional memiliki keterbatasan. Sifatnya rigid dengan sekumpulan aturan “if” dan “else” untuk memproses data atau menyesuaikan dengan masukan.

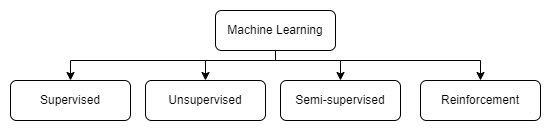
Paradigma baru pemrograman dengan machine learning adalah dengan memiliki banyak sekali data dan label dari data tersebut. Keterkaitan antara data dengan label diketuahui sebagai suatu solusi. Algoritma machine learning mencari pola tertentu dari setiap kumpulan data yang menentukan kekhasan masing-masing untuk kemudian menyimpulkan sebuah aturan. Selanjutnya, aturan ini dapat digunakan untuk melakukan identifikasi dan prediksi bagi data baru yang relevan dengan model yang kita miliki.



Gambar 2. 4 *Traditional programming vs machine learning*

*Machine learning* secara umum dibagi menjadi 4, yaitu *supervised learning, unsupervised learning, semi unsupervised learning*, dan *reinforcement learning* (Astuti, 2021)*.*

1. *Supervised learning*, dataset yang digunakan memiliki label. Label adalah *tag* pengenal dari data. Algoritma *supervised learning* mudah dipahami dan performa akurasinya pun mudah diukur. *Supervised learning* dapat dilihat sebagai sebuah mesin/robot yang belajar menjawab pertanyaan sesuai dengan jawaban yang telah disediakan manusia. Klasifikasi email spam adalah contoh *supervised learning*.
2. *Unsupervised learning*, dataset yang digunakan tidak memiliki label. Model ini melakukan belajar sendiri untuk melabeli atau mengelompokkan data. *Unsupervised learning* dapat dilihat sebagai robot/mesin yang berusaha belajar menjawab pertanyaan secara mandiri tanpa ada jawaban yang disediakan manusia.
3. *Semi Supervised* merupakan gabungan dari *supervised learning* dan *unsupervised learning*. Pada model *semi supervised*, dataset untuk pelatihan sebagian memiliki label dan sebagian tidak. Google Photos adalah contoh implementasi *semi supervised*. Pada Google Photos pengguna dapat memberi *tag* atau label untuk setiap orang yang ada dalam sebuah foto. Alhasil, ketika kita mengunggah foto baru dengan wajah orang yang sebelumnya sudah kita beri label, Google Photos akan secara otomatis mengenali orang tersebut.
4. *Reinforcement Learning* adalah model belajar menggunakan sistem *reward and penalties*. Model belajar mendapatkan *reward* dan menghindari *penalties*.



Gambar 2. 5 Jenis *Machine Learning*

1. *Ensemble Learning*

Dalam beberapa tahun terakhir, *ensemble learning* telah banyak diterapkan untuk meningkatkan akurasi dalam masalah klasifikasi maupun regresi melebihi tingkat yang dapat dicapai oleh data masukan yang terpisah. *Ensemble learning* adalah teknik dalam pembelajaran mesin (*machine learning*) dimana menggunakan beberapa model pembelajaran untuk pencapaian solusi prediksi yang lebih baik daripada algoritma yang bisa diperoleh dari salah satu model konstituen saja. Istilah *ensemble* merujuk pada konsep menggabungkan banyak elemen menjadi satu kesatuan. Konsep penggabungan berbagai pendekatan pembelajaran mesin diharapkan dapat mengatasi kelemahan individu dari masing-masing model dan menciptakan prediksi yang lebih akurat dan stabil. Penggunaan beberapa model pembelajaran dan bukannya satu model tunggal untuk mengungkap masalah sulit yang diberikan telah selalu menjadi solusi yang diterima. *Ensemble learning method* secara teoritis dan empiris telah diperoleh lebih baik kinerja yang lebih baik daripada pembelajar tunggal yang lemah, terutama saat menangani komputasi dengan dimensi tinggi, kompleks masalah regresi dan klasifikasi (Sahin, 2020; Mardiani dkk, 2023).

Secara garis besar, teknik ensemble terbagi menjadi 2 kategori besar, yakni *sequential ensemble techniques* dan *parallel ensemble teqhniques. Sequential ensemble techniques* menghasilkan pembelajaran dasar secara berurutan, dimana masing – masing model saling berkaitan. Kinerja model kemudian ditingkatkan dengan memberikan bobot yang lebih tinggi kepada pembelajaran yang keliru. Pada *parallel ensemble teqhniques*, pembelajaran dasar dibuat dengan format paralel, seperti *random forest*.

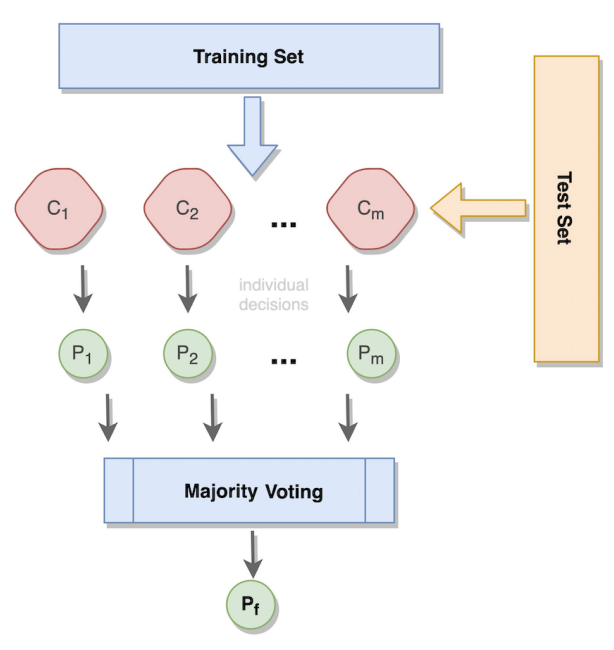
Adapun jenis utama dari *ensemble learning* adalah *voting, bagging, boosting* dan *stacking*.

1. Metode *Voting*

Metode voting adalah salah satu teknik dalam ensemble learning yang digunakan untuk menggabungkan hasil prediksi dari beberapa model pembelajaran mesin (machine learning) yang berbeda. Dalam metode ini, setiap model memberikan suara atau voting untuk kelas yang diprediksi, dan kelas dengan jumlah suara terbanyak dipilih sebagai hasil akhir dari ensemble model.

Ada dua jenis metode voting yang umum digunakan dalam ensemble learning, yaitu hard voting dan soft voting. Hard voting digunakan ketika model-model yang digunakan dalam ensemble menghasilkan prediksi kelas yang diskrit atau terbatas pada beberapa kelas saja. Dalam hard voting, kelas dengan jumlah suara terbanyak dipilih sebagai hasil akhir. Sedangkan, soft voting digunakan ketika model-model yang digunakan dalam ensemble menghasilkan probabilitas untuk setiap kelas. Dalam soft voting, probabilitas dari setiap model dijumlahkan dan kelas dengan probabilitas terbesar dipilih sebagai hasil akhir (Fazakis dkk, 2021).

Metode voting dalam ensemble learning dapat meningkatkan akurasi dan stabilitas model, karena menggabungkan hasil prediksi dari beberapa model yang berbeda. Hal ini dapat membantu mengatasi masalah *overfitting* dan *underfitting* yang sering terjadi pada model pembelajaran mesin tunggal.



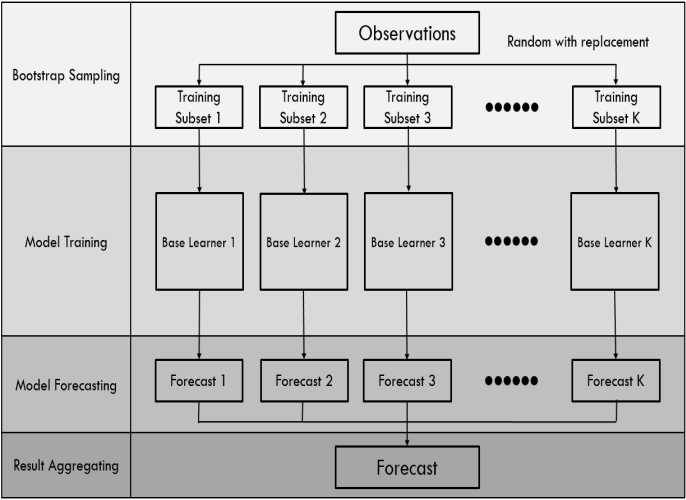
Gambar 2. 6 Metode *voting* pada *ensemble learning* (Peppes dkk, 2021)

1. Metode *Bagging*

Metode *bagging (bootstrap aggregating)* adalah salah satu pendekatan dalam *ensemble learning* di mana beberapa model pembelajaran digunakan secara paralel untuk meningkatkan stabilitas dan akurasi model pembelajaran mesin. Hasil prediksi dikombinasikan untuk menghasilkan prediksi akhir. Setiap model pembelajaran dilatih pada subset acak dari data pelatihan yang dihasilkan melalui proses *bootstrap*, di mana pengambilan sampel dilakukan dengan pengembalian. Dengan pendekatan ini, variasi di antara model-model yang dihasilkan dapat mengurangi overfitting dan meningkatkan generalisasi model.

Berikut adalah langkah-langkah utama dari algoritma *bagging* (Choi dan Hur, 2020):

1. Membuat B *bootstrap samples* , dimana adalah data pelatihan.
2. Untuk setiap *bootstrap samples*, , dibangun model *forecasting* .
3. Menggabungkan B model *forecasting* ke model *final .*
4. Dalam model regresi, model akhir merupakan nilai rata – rata dari total semua model *forecasting*, .



Gambar 2. 7 Metode *bagging* (Choi dan Hur, 2020)

Nilai *loss* yang diharapkan dari model *averange forecasting* lebih kecil dibandingkan nilai *loss* dari model tunggal sehingga *bagging* disebut dapat menigkatkan prediktabilitas. Model yang dibangun dari *training set* yang diberikan, sangat bergantung pada . Untuk menyoroti hal ini, dituliskan sebagai dan untuk model *forecasting* yang diberikan, model *forecasting* rata – rata didefenisikan sebagai . Di sini, nilai yang diharapkan menggunakan distribusi dari populasi data *train* yang diperoleh, dan teorema di bawah ini menunjukkan bahwa *loss* yang diharapkan dari model *forecasting* rata – rata lebih kecil dari *loss* model *forecasting* tunggal.

Dikatakan bahwa adalah hasil pengamatan di masa depan yang tidak terikat dengan , untuk *square loss function* , *loss* yang diharapkan dari dan , dan didefenisikan sebagai berikut :

(2.1)

Karena fungsi kuadrat adalah fungsi cembung, persamaan (2.2) ditetapkan oleh *Jensen inequality*:

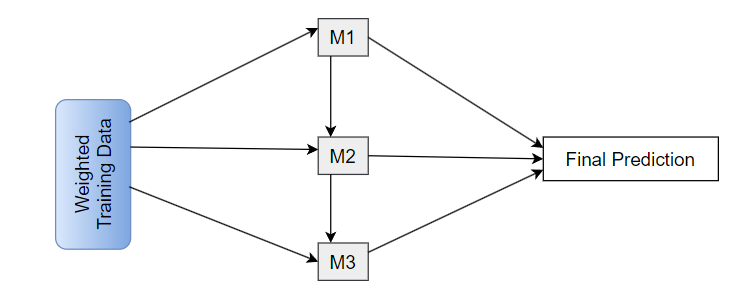
(2.2)

Sehingga, akan selalu lebih baik dari (Choi dan Hur, 2020).

1. Metode *Boosting*

*Boosting* adalah model *ensemble* yang dibentuk dari beberapa model dasar, kemudian secara berurutan model dasar tersebut dilatih dan digabungkan dalam prediksi. Algoritma *boosting* membangun model secara bertahap dengan mengoptimalkan suatu fungsi kerugian. Setiap model dilatih pada sebuah set pelatihan yang diberi bobot. Bobot ini ditentukan berdasarkan kesalahan model sebelumnya dalam urutan tersebut. Ini merupakan metode yang dapat mengonversi pembelajar lemah menjadi pembelajar kuat (Choi dan Hur, 2020; Pal dkk, 2020).

Ide utama di balik pelatihan berurutan adalah agar setiap model dapat memperbaiki kesalahan pendahulunya. Proses ini berlangsung hingga jumlah model yang telah ditentukan terlatih atau hingga kriteria lain terpenuhi.

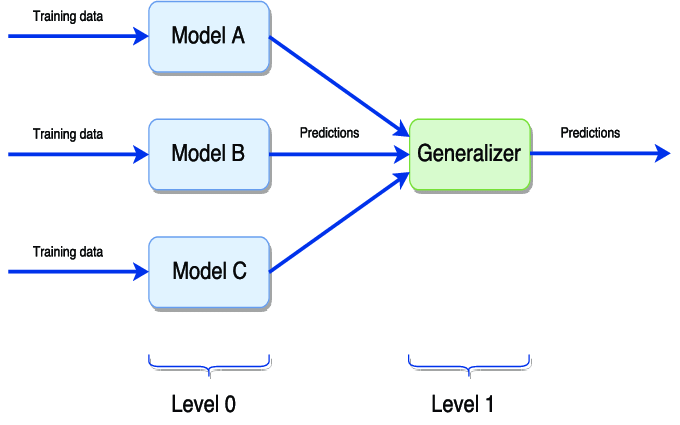


Gambar 2. 8 Metode *boosting* (Budu, 2023)

Selama pelatihan, *instances* yang diklasifikasikan secara tidak benar diberi bobot yang lebih tinggi untuk memberikan prioritas saat dilatih dengan model berikutnya. Selain itu, model yang lebih lemah diberi bobot lebih rendah dibandingkan dengan model yang lebih kuat saat menggabungkan prediksi menjadi keluaran akhir. Jadi, langkah-langkahnya dilakukan secara berulang sebagai berikut (Budu 2023):

1. Model dilatih pada semua *instances*.
2. Kesalahan pada keluaran model dihitung untuk semua instansi.
3. Bobot ditetapkan pada model (tinggi untuk kinerja baik dan sebaliknya).
4. Bobot data diperbarui: bobot yang lebih tinggi diberikan pada sampel dengan kesalahan tinggi.
5. Langkah-langkah sebelumnya diulangi jika kinerja tidak memuaskan atau jika kondisi berhenti lainnya terpenuhi.
6. Akhirnya, model digabungkan menjadi satu yang digunakan untuk prediksi.
7. Metode *Stacking*

*Stacking* adalah teknik *ensemble learning* yang menggabungkan beberapa model untuk meningkatkan akurasi prediksi. Ide dasar dari *stacking* adalah menggunakan *base learner* untuk menghasilkan *metadata* untuk input dan kemudian menggunakan *meta-learner*, yang umumnya disebut level 1 *learner*, untuk memproses metadata tersebut. *Base learner* biasanya disebut level 0 *learner*, dan *meta learner* yang ditumpuk pada *base learner* disebut *stacking* (Liang dkk, 2021).

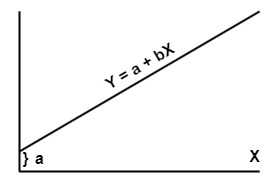


Gambar 2. 9 Metode *stacking* (Shukla, 2022)

Proses *stacking* melibatkan pembagian data pelatihan menjadi dua *subset* atau lebih, diikuti dengan pelatihan beberapa model dasar pada satu *subset* data. Model yang terlatih kemudian digunakan untuk membuat prediksi pada subset data lainnya. Selanjutnya, prediksi dari model dasar digabungkan ke dalam satu dataset baru. Proses ini dilanjutkan dengan melatih *meta-model* pada data baru ini, menggunakan nilai target aktual sebagai label. Terakhir, *meta-model* digunakan untuk membuat prediksi pada data uji (Sunarko dkk*.*, 2023). Ide dasar di balik stacking adalah bahwa setiap model dasar memiliki kekuatan dan kelemahan yang mungkin berbeda, dan dengan menggabungkan prediksi mereka, *meta-model* dapat mempelajari cara yang tepat untuk menimbang kekuatan dan kelemahan tersebut (Zhang, Liu dan Shen, 2022).

1. *Regression*

Metode regresi merupakan sebuah metode statistik yang melakukan prediksi menggunakan pengembangan hubungan matematis antara variabel, yaitu variabel dependen (Y) dengan variabel independen (X). Variabel dependen merupakan variabel akibat atau variabel yang dipengaruhi, sedangkan variabel independen merupakan variabel sebab atau variabel yang mempengaruhi. Prediksi terhadap nilai variabel dependen dapat dilakukan jika variabel independennya diketahui (Ayuni dan Fitrianah, 2019).



Gambar 2. 10 Regresi linear

Penggunaan regresi linear telah menjadi metode umum dalam industri untuk meramalkan atau memprediksi sifat kualitatif maupun kuantitatif. Ini disebabkan oleh kemampuannya untuk memproyeksikan kombinasi produk yang beragam, memungkinkan perusahaan untuk mengoptimalkan keuntungan dan menentukan jumlah produksi yang optimal.

Rumus untuk Regresi Linear dengan metode kuadrat terkecil atau sederhana adalah :

(2.3)

(2.4)

(2.5)

1. *Extreme Gradient Boosting*

*Extreme Gradient Boosting (XGBoost)* adalah sebuah sistem pohon *boosting* yang efisien dan *scalable* yang dapat digunakan untuk memprediksi nilai numerik atau kategori dari suatu data (Chen & Guestrin, 2016). Algoritma ini menggunakan teknik *gradient boosting* untuk meningkatkan akurasi prediksi dan mengatasi masalah *overfitting*. *XGBoost* juga dilengkapi dengan fitur-fitur seperti regularisasi, *parallel processing*, dan *handling missing values* yang membuatnya menjadi salah satu algoritma *machine learning* yang paling populer dan efektif saat ini.

Model pohon *boosting* tradisional hanya menggunakan informasi turunan pertama. Ketika melatih pohon ke-n, sulit untuk mengimplementasikan pelatihan terdistribusi karena sisa dari pohon sebelumnya ke-n-1 digunakan. *XGBoost* melakukan ekspansi Taylor orde kedua pada fungsi kerugian dan secara otomatis dapat menggunakan *multithreading* dari CPU untuk komputasi paralel. Selain itu, *XGBoost* menggunakan berbagai metode untuk menghindari *overfitting* (Li dkk, 2019).

Algortima *XGBoost* secara sederhana dituliskan sebagai berikut. Integrasi model pohon dengan metode penjumlahan, diasumsikan jumlah dari K pohon, dan F menggambarkan model pohon dasar, maka :

(2.6)

*Objective function* dituliskan :

(2.7)

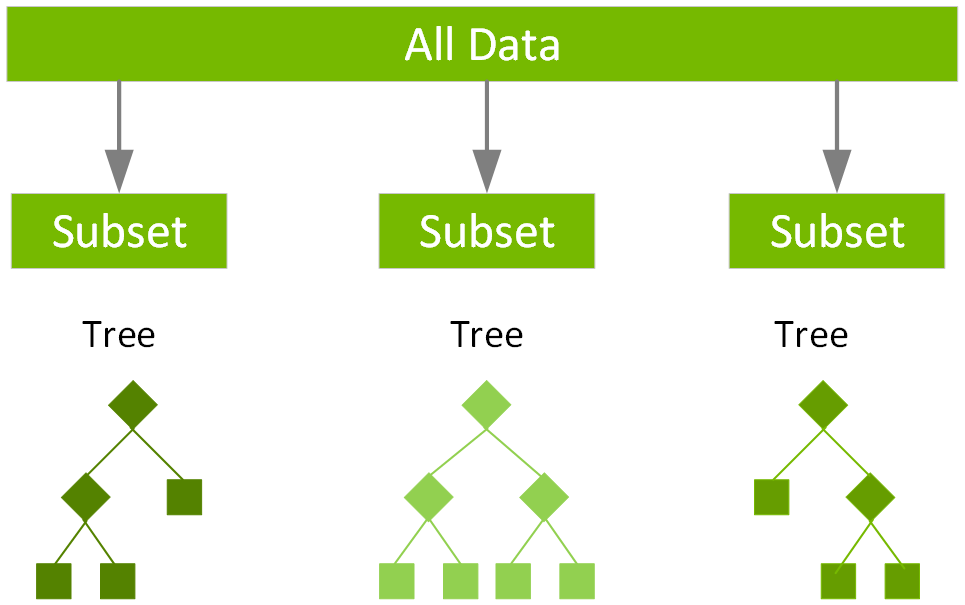
Dimana adalah *loss function*, yang mana menggambarkan *error* antara nilai prediksi dengan nilai sebenarnya. Kemudian adalah fungsi yang digunakan untuk *regularization* untuk mencegah *overfitting*, sebagai berikut :

(2.8)

Dimana mewakili jumlah daun per pohon, dan mewakili bobot daun dari setiap pohon. Setelah ekspansi Taylor orde dua dari *objective funtion* dan perhitungan rinci lainnya, didapatkan *information gain* dari *objective function* setelah setiap pemisahan adalah :

(2.9)

Seperti yang dapat dilihat pada (2.9), untuk menekan pertumbuhan dari pohon dan mencegah model dari overfitting, pemisahan *threshold* γ ditambahkan. Simpul daun diperbolehkan untuk dipecah jika dan hanya jika information gain lebih besar dari γ. Hal ini ini setara dengan *pre-pricing* pohon sambil mengoptimalkan *objective function*.



Gambar 2. 11 Algoritma *XGBoost*

Saat menggunakan model *XGBoost*, terdapat beberapa parameter yang perlu disesuaikan untuk membuat model dalam performa terbaiknya. Parameter tersebut adalah sebagai berikut (Li dkk, 2019):

1. *n\_estimators*

*n\_estimator* adalah jumlah iterasi dalam pelatihan. Nilai yang terlalu *n\_estimator* yang terlalu kecil dapat menyebabkan *underfitting*, yang membuat membuat model tidak dapat melakukan kemampuan pembelajarannya secara maksimal. Namun, nilai yang terlalu estimator yang terlalu besar biasanya juga tidak baik, karena akan menyebabkan *overfitting*.

1. *min\_child\_weight*

*min\_child\_weight* mendefinisikan jumlah dari berat sampel dari simpul daun terkecil untuk mencegah *overfitting.*

1. *max\_depth*

Ini adalah kedalaman maksimum pohon. Semakin besar kedalaman pohon, semakin kompleks model pohonnya, dan semakin kuat kemampuan pemasangannya, tetapi pada saat yang sama, modelnya jauh lebih mudah untuk *overfit*.

1. *subsample*

Parameter ini berarti laju pengambilan sampel dari semua sampel pelatihan.

1. *colsample\_bytree*

*colsample\_bytree* adalah tingkat pengambilan sampel fitur saat membangun setiap pohon.

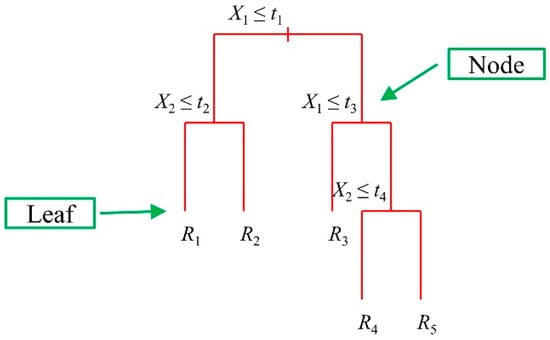
1. *learning\_rate*

Di sebagian besar algoritma, *learning\_rate* adalah parameter yang sangat penting yang perlu disesuaikan, begitu juga dalam *XGBoost*. Ini sangat mempengaruhi kinerja model. Kita dapat mengurangi bobot dari setiap langkah untuk membuat model lebih kuat.

1. *Random Forest*

*Random forest* adalah salah satu algoritma *machine learning* yang paling sering digunakan. Algortima ini pertama kali dipublikasikan oleh Breiman (2001). Metode ini menggabungkan beberapa pohon keputusan (*decision tree*) yang dibangun secara acak dan independen satu sama lain. Setiap pohon keputusan dalam *random forest* dibangun dengan menggunakan subset acak dari fitur-fitur yang tersedia dalam data.

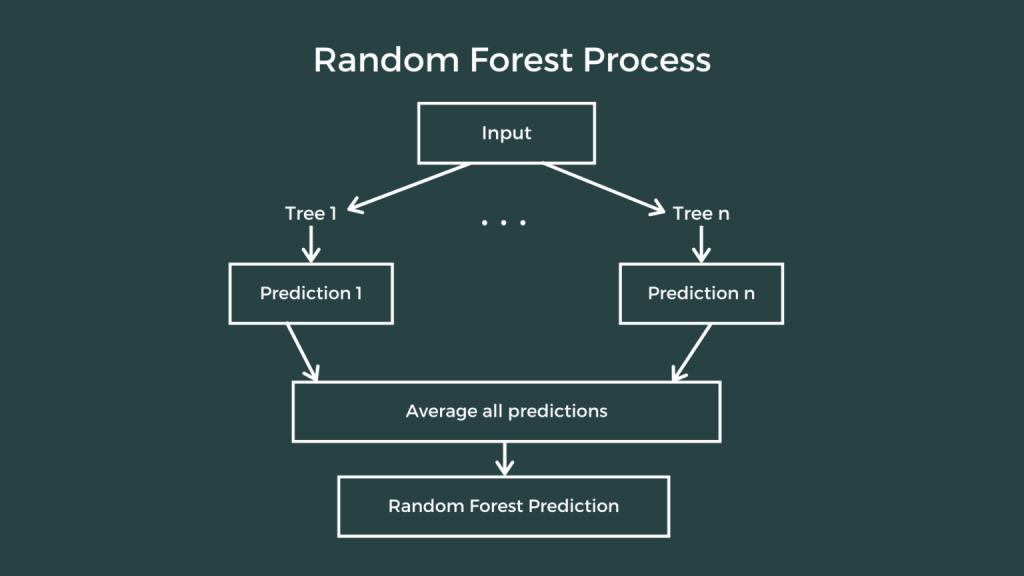
*Random forest* merupakan *bagging* dari *Classification and Regression Trees* CART dengan beberapa tingkat pengacakan tambahan. CART adalah metode untuk mempartisi ruang variabel berdasarkan sekumpulan aturan yang tertanam dalam *decision tree*, di mana setiap simpul terpecah menurut aturan keputusan. Dengan cara ini, ruang variabel dipartisi ke dalam sekumpulan persegi panjang, dan sebuah model dipasang pada setiap set, yang dalam kasus yang paling sederhana dapat berupa sebuah konstanta. Pada pohon regresi, aturan keputusan untuk pemisahan simpul disetel/dipelajari dengan mengoptimalkan jumlah kuadrat deviasi, sedangkan pada klasifikasi dengan mengoptimalkan indeks Gini (Tyralis dan Papacharalampous, 2019).



Gambar 2. 12 Contoh *decision tree*

*Bagging* CART diperlukan untuk mengurangi ketidakstabilannya. Selanjutnya, pengacakan digunakan untuk mengurangi korelasi antar pohon dan, akibatnya, mengurangi varians prediksi (yaitu, rata-rata pohon). Pengacakan dilakukan dengan memilih secara acak variabel prediktor *mtry* sebagai kandidat untuk pemisahan.

Prediksi dalam regresi dilakukan dengan merata-ratakan prediksi dari setiap pohon. Adapun dalam klasifikasi, prediksi dilakukan dengan mendapatkan suara kelas mayoritas dari suara kelas pohon individu (Tyralis dan Papacharalampous, 2019).



Gambar 2. 13 Alur *random forest* untuk regresi

Saat melakukan pemodelan dengan *random forest*, terdapat beberapa parameter yang perlu disesuaikan agar model berada pada performa terbaiknya. Parameter – parameter tersebut adalah sebagai berikut :

1. *n\_estimators*

*n\_estimators* merupakan jumlah pohon yang dibangun oleh algoritma sebelum membuat rata-rata prediksi.

1. *max\_features*

Jumlah maksimum fitur yang dipertimbangkan oleh *random forest* untuk memisahkan sebuah *node*.

1. *mini\_sample\_leaf*

*mini\_sample\_leaf* berguna untuk menentukan jumlah daun minimum yang diperlukan untuk membagi *node* internal.

1. *Criterion*

Parameter ini berfungsi untuk mengatur bagaimana cara membagi *node* di setiap pohon. Cara tersebut antara lain *entropy, gini impurity,* atau *log loss*.

1. *max\_leaf\_nodes*

*max\_leaf\_nodes* mengatur jumlah daun maksimal untuk setiap *node*.

1. *n\_jobs*

Parameter ini memberi tahu mesin berapa banyak prosesor yang diizinkan untuk digunakan. Jika nilainya 1, maka hanya dapat menggunakan satu prosesor, tetapi jika nilainya -1, maka tidak ada batasan.

1. *random\_state*

*random\_state* mengontrol keacakan sampel. Model akan selalu menghasilkan hasil yang sama jika memiliki nilai yang pasti dari keadaan acak dan telah diberikan *hyperparameter* dan data pelatihan yang sama.

1. *oob\_score*

OOB berarti *out of the bag*. Ini adalah metode validasi silang *random forest*. Dalam hal ini, sepertiga dari sampel tidak digunakan untuk melatih data, melainkan digunakan untuk mengevaluasi kinerjanya. Sampel ini disebut sampel *out-of-bag*.

1. Python

Python merupakan salah satu bahasa pemrograman yang paling populer saat ini. Python pertama kali ditemukan oleh Guido van Rossum pada 1991. Python adalah bahasa pemrograman yang menggunakan interpreter untuk menjalankan kode programnya. Interpreter tersebut dapat menerjemahkan kode secara langsung, dan Python dapat dijalankan di berbagai platform seperti Windows, Linux, dan lain-lain. Python mengadopsi paradigma pemrograman dari beberapa bahasa lain, termasuk paradigma pemrograman prosedural seperti bahasa C, pemrograman berorientasi objek seperti Java, dan bahasa fungsional seperti Lisp. Kombinasi paradigma ini memudahkan para programmer dalam mengembangkan berbagai proyek menggunakan Python (Rahman dkk, 2023).



Gambar 2. 14 Python

Dalam perkembangan *machine learning* dan *data science*, bahasa pemrograman Python telah menjadi pilihan utama bagi para praktisi karena kombinasi uniknya antara kemudahan penggunaan dan performa yang baik. Python dikenal dengan sintaksisnya yang mudah dipahami, sehingga memungkinkan pengguna untuk dengan cepat mempelajari dan mengimplementasikan berbagai konsep dalam *machine learning* dan *data science*. Selain itu, Python memiliki ekosistem yang sangat luas dengan berbagai library dan *framework* yang mendukung pengembangan model *machine learning*, seperti Scikit-learn, TensorFlow, dan Keras, yang mempercepat proses pengembangan dan eksperimen (Raschka, Patterson dan Nolet, 2020).

Selain kemudahan penggunaan, Python juga mampu untuk mengoptimalkan performa komputasi melalui penggunaan library yang mengimplementasikan kode-kode yang lebih rendah, seperti Fortran, C/C++, dan CUDA. Hal ini memungkinkan Python untuk tetap efisien dalam komputasi yang membutuhkan kecepatan dan efisiensi tinggi, seperti dalam pengolahan data besar dan pelatihan model machine learning yang kompleks. Dengan kombinasi antara kemudahan penggunaan, ekosistem yang kaya, dan performa yang baik, Python terus menjadi bahasa yang dominan dalam dunia *machine learning* dan *data science*, memungkinkan para praktisi untuk mengembangkan solusi-solusi inovatif dengan lebih efisien (Raschka, Patterson dan Nolet, 2020).

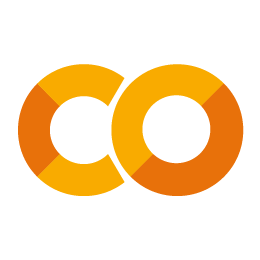
1. Jupyter Notebook dan Google Colaboratory

Jupyter Notebook adalah format publikasi yang memungkinkan pembuatan alur kerja komputasi yang dapat dieksekusi dan dimodifikasi oleh orang lain (Sengkey dkk, 2020). Jupyter Notebook memungkinkan pengguna untuk membuat dan berbagi dokumen interaktif yang menggabungkan teks/narasi, kode, persamaan, tampilan hasil, gambar statis, dan visualisasi grafis dalam satu file *notebook* yang interaktif. Jupyter Notebook mendukung bahasa pemrograman Julia, Python dan R.



Gambar 2. 15 Jupyter Notebook

Google Colaboratory (Google Colab) adalah *platform online* Google yang menyediakan *environment* Jupyter Notebook berbasis *cloud* (Guntara, 2023). Pengguna dapat membuat, berbagi, dan berkolaborasi pada dokumen Jupyter Notebook. Konektivitas Google Colab dengan Google Drive adalah salah satu keunggulan utamanya, memungkinkan pengguna untuk menyimpan dan memuat *notebook* dan kumpulan data dengan mudah langsung dari akun Google Drive. Google Colab berjalan di *browser web* dan memungkinkan siapa pun yang memiliki akses internet untuk bereksperimen dengan pembelajaran mesin dan pengkodean untuk kecerdasan buatan. Google Colab hanya mendukung bahasa pemrograma Python.



Gambar 2. 16 Google Colaboratory

1. FastAPI

FastAPI merupakan *framework* berbasis Python yang dikhususkan untuk membangun RESTful API (Azhari, 2022). Dengan fokus pada kinerja tinggi, FastAPI menonjol dengan kecepatan yang sebanding dengan NodeJS dan Go. FastAPI memiliki desain yang mudah digunakan, mengurangi kebutuhan untuk membaca dokumentasi, dan minimalisasi duplikasi kode melalui fitur-fitur seperti deklarasi parameter (Wijaya, Wibawa dan Darmawan, 2022). FastAPI juga menawarkan keandalan dengan menghasilkan kode siap produksi dan dokumentasi interaktif otomatis. FastAPI berbasis pada standar terbuka untuk API, seperti OpenAPI dan JSON Schema, memastikan kompatibilitas dengan alat dan layanan lain yang mengikuti standar yang sama. Dengan kombinasi fitur-fitur ini, FastAPI menjadi pilihan yang kuat untuk pengembangan API yang cepat, andal, dan efisien dalam lingkungan Python (FastAPI, 2024).



Gambar 2. 17 FastAPI

1. *Website*

*Website* merupakan sejumlah halaman yang membentuk suatu arsitektur yang saling terkait, dapat berupa statis atau dinamis, yang menampilkan informasi dalam berbagai bentuk multimedia, seperti teks, gambar dan lainnya (Agustin dkk, 2021; Indriani, 2022). *Website* menggunakan internet untuk diakses. *Website* dapat digunakan untuk berbagai keperluan, seperti bisnis, pendidikan, hiburan, dan lainnya.

*Website* dapat digunakan sebagai tempat untuk menampilkan dan menggunakan model *machine learning* bagi pengguna. Beberapa *platform* seperti Firebase, AWS, dan Google Cloud Platform menyediakan solusi untuk deploy model machine learning dengan mudah dan efektif. Selain itu, pengguna juga dapat menggunakan praktik MLOps untuk mengintegrasikan proses ML dengan DevOps dan mengelola model dengan lebih efektif.

1. Node.js

Node.js adalah lingkungan *runtime* JavaScript yang memungkinkan pengembang untuk mengeksekusi kode JavaScript di sisi server. Dibangun di atas mesin JavaScript V8 dari Chrome, Node.js menawarkan model *non-blocking*, *event-driven* yang membuatnya ideal untuk aplikasi *real-time* yang membutuhkan kinerja tinggi dan responsif. Dengan pendekatan single-threaded dan event looping, Node.js dapat menangani banyak koneksi secara bersamaan tanpa menghambat kinerja server (Kadi, 2021).

Salah satu keunggulan utama Node.js adalah NPM (*Node Package Manager*), yang merupakan manajer paket yang memungkinkan pengembang untuk dengan mudah mengelola dependensi proyek mereka. Dengan NPM, pengembang dapat dengan cepat menginstal paket-paket modul yang diperlukan untuk proyek mereka dan memperluas fungsionalitas aplikasi dengan mudah (Koishybayev dan Kapravelos, 2020).

Node.js sangat populer untuk pengembangan *server-side*, seperti pembuatan API, aplikasi *web*, dan layanan *backend*. Platform ini juga dikenal karena kemampuannya untuk berjalan di berbagai platform seperti Windows, macOS, dan Linux, membuatnya sangat fleksibel dan mudah diadopsi oleh berbagai jenis proyek pengembangan.

Dengan ekosistem yang besar dan aktif, Node.js menawarkan ribuan paket modul yang tersedia melalui NPM, memungkinkan pengembang untuk memanfaatkan kembali kode dan mempercepat proses pengembangan. Node.js juga sangat cocok untuk aplikasi real-time seperti aplikasi obrolan, game online, dan streaming, karena kemampuannya dalam menangani banyak koneksi secara efisien dan responsif. Dengan kinerja yang cepat, skalabilitas, dan fleksibilitasnya, Node.js telah menjadi salah satu platform pengembangan *server-side* yang paling populer di dunia pengembangan perangkat lunak saat ini (Kadi 2021).



Gambar 2. 18 Node.js

1. Express.js

Express.js adalah sebuah *framework web* yang berjalan pada *runtime* JavaScript Node.js. *Framework* ini digunakan untuk membangun aplikasi *web* berbasis server dengan mudah dan cepat. Express.js menyediakan fitur-fitur yang memudahkan pengembangan aplikasi web seperti *routing*, *middleware*, dan *template engine* (Jonsson dan Qvarnström, 2022).

Menggunakan Express.js sebagai *web framework* memiliki banyak keuntungan. Express.js ringan dan memungkinkan *routing* API yang efektif. Modul *middleware* mengaktifkan Express.js setelah menerima permintaan dari server dan sebelum server mengembalikan tanggapan kepada klien. Dengan modul *middleware*, pengembang dapat meningkatkan fitur keamanan aplikasi, mengoptimalkan output sebelum pengiriman sisi klien, atau berinteraksi dengan objek permintaan sebanyak yang mereka butuhkan (Nguyen, 2022). Selain itu, Express.js juga memiliki dokumentasi yang lengkap dan komunitas yang besar, sehingga memudahkan pengembang untuk mencari solusi jika mengalami masalah dalam pengembangan aplikasi *web*.



Gambar 2. 19 Express.js

1. *Flowchart*

*Flowchart* adalah representasi grafis dari algoritma atau prosedur yang digambarkan dalam bentuk bagan dengan simbol-simbol tertentu. Tujuannya adalah untuk menggambarkan tahapan penyelesaian suatu masalah dengan cara yang sederhana, rapi, dan jelas. Dalam *flowchart*, langkah-langkah dari suatu prosedur atau algoritma dijelaskan menggunakan simbol-simbol standar yang mudah dimengerti. Sebagai alternatif terhadap penulisan teks dalam struktur English atau struktur Indonesia, *flowchart* memberikan cara visual yang lebih universal dan dapat dipahami oleh berbagai pihak tanpa dipengaruhi oleh tata bahasa pembuatnya. Dengan menggunakan notasi grafis, *flowchart* membantu menyederhanakan dan mengklarifikasi urutan langkah dari suatu file atau prosedur dalam media, menjadikannya alat yang efektif dalam komunikasi dan dokumentasi (Budiman, 2015; Furqani dan Muliono, 2021).

*Flowchat* dapat dikategorikan dalam 2 jenis, yaitu *program flowchart* dan *system flowchart*. *Program flowchart* berperan seperti cerminan program komputer yang digambarkan dalam simbol – simbol *flowchart.* Hal ini berisi langkah – langkah penyelesaian unit masalah untuk hasil tertentu. *System flowchart* berisi solusi dari banyak unit masalah secara bersamaan yang terkait satu sama lain dan berinteraksi satu sama lain untuk mencapai suatu tujuan (Chaudhuri, 2020).

Simbol – simbol yang umumnya digunakan dalam *flowchart* diberikan pada Tabel 2. 1 (Chaudhuri 2020).

Tabel 2. 1 Simbol – simbol *flowchart*

|  |  |  |
| --- | --- | --- |
| Nama Simbol | Deskripsi | Gambar |
| *Terminal* | Digunakan untuk menunjukkan permulaai dan akhir dari proses yang digambarkan. |  |
| *Input/Output* | Digunakan untuk menunjukkan suatu operasi *input/output.* |  |
| *Computer Processing* | Digunakan untuk menunjukkan suatu proses yang dijalankan oleh sistem komputer . |  |
| *Predefined Processing* | Digunakan untuk menunjukkan proses yang tidak didefinisikan secara khusus dalam *flowchart.* |  |
| *Comment* | Digunakan untuk menulis pernyataan penjelasan yang diperlukan untuk mengklarifikasi sesuatu. |  |
| *Flow Line* | Digunakan untuk menghubungkan simbol – simbol. |  |
| *Document Input/Output* | Digunakan ketika input datang dari dokumen dan output masuk ke dokumen. |  |

**Tabel 2. 1 Simbol – simbol *flowchart* (Lanjutan)**

|  |  |  |
| --- | --- | --- |
| Nama Simbol | Deskripsi | Gambar |
| *Decision* | Digunakan untuk menunjukkan suatu titik dalam proses di mana keputusan harus dibuat untuk menentukan tindakan lebih lanjut. |  |
| *On-Page Connector* | Digunakan untuk menghubungkan bagian-bagian dari *flowchart* dilanjutkan pada halaman yang sama. |  |
| *Off-Page Connector* | Digunakan untuk menghubungkan bagian-bagian dari *flowchart* dilanjutkan pada halaman yang terpisah. |  |

1. *Unified Modelling Language*

*Unified Modeling Language* (UML) adalah sebuah bahasa standar yang digunakan dalam pengembangan perangkat lunak untuk memodelkan, mendokumentasikan, dan merancang sistem secara visual. UML menyediakan notasi grafis yang kaya dan beragam, memungkinkan para pengembang untuk menggambarkan struktur dan perilaku sistem dengan jelas. Selain sebagai alat visualisasi sistem, UML juga berperan penting dalam komunikasi antar pemangku kepentingan proyek, seperti pengembang, manajer proyek, dan pemilik produk. Dengan menggunakan UML, informasi tentang sistem dapat disampaikan dengan lebih efektif dan dipahami oleh berbagai pihak terkait. Selain itu, UML digunakan dalam tahap perancangan sistem untuk merancang struktur sistem secara rinci, termasuk kelas, objek, relasi antar objek, dan perilaku sistem (Ahmad dkk, 2019).

Dokumentasi yang dihasilkan dari model UML juga menjadi acuan yang berguna selama siklus hidup pengembangan perangkat lunak, membantu dalam pemahaman, analisis, dan pengujian sistem secara lebih terstruktur dan efisien. Dengan berbagai kegunaan yang dimilikinya, UML menjadi alat yang sangat berguna dalam pengembangan perangkat lunak modern, meningkatkan kualitas, produktivitas, dan kolaborasi dalam pengembangan sistem perangkat lunak.

1. *Use Case Diagram*

*Use case diagram* merupakan salah satu jenis UML yang digunakan untuk memodelkan interaksi antara aktor dan sistem. Dalam diagram ini, aktor mewakili entitas yang berinteraksi dengan sistem, seperti pengguna atau sistem eksternal lainnya, sedangkan use case mewakili fungsionalitas sistem yang berbeda (Anggraini dkk, 2020). *Use case diagram* membantu dalam merencanakan dan memvisualisasikan skenario penggunaan sistem, serta menunjukkan bagaimana aktor akan berinteraksi dengan sistem untuk mencapai tujuan mereka. Dengan demikian, diagram ini menjadi alat yang berguna dalam merancang antarmuka pengguna yang sesuai dengan kebutuhan pengguna dan tujuan bisnis yang diinginkan.

Simbol – simbol pada *use case diagram* diberikan pada tabel 2. 2 (Ariansyah dan Wijaya, 2021).

Tabel 2. 2 Simbol – simbol *use case diagram*

|  |  |  |
| --- | --- | --- |
| Nama Simbol | Deskripsi | Gambar |
| *Use Case* | *Use case* adalah representasi dari fungsionalitas sistem yang disajikan sebagai unit-unit yang berinteraksi dengan aktor melalui pertukaran pesan, yang dijelaskan dengan menggunakan kata kerja. |  |
| *Actor* | *Actor* adalah entitas yang menggunakan sistem. *Actor* tidak hanya terbatas pada manusia, tetapi juga mencakup aplikasi lain yang berkomunikasi dengan sistem, apabila membutuhkan input dari sistem atau memberikan output ke sistem. | Intro to Use Case Types | Systems Flow, Inc |

**Tabel 2. 2 Simbol – simbol *use case diagram* (Lanjutan)**

|  |  |  |
| --- | --- | --- |
| Nama Simbol | Deskripsi | Gambar |
| Asosiasi | Asosiasi digunakan untuk menyatakan hubungan antara *actor* dengan *use case.* |  |
| *Extend* | Menunjukkan bahwa suatu *use case* merupakan tambahan fungsional dari *use case* lainnya jika suatu kondisi terpenuhi. | <<*Extend*>> |
| *Include* | Hubungan tambahan dari satu *use case* ke *use case* lain menunjukkan bahwa *use case* yang ditambahkan memerlukan *use case* yang terhubung tersebut untuk menjalankan fungsinya. | <<*Include*>> |

1. *Activity Diagram*

*Activity diagram* adalah representasi visual dari aliran aktivitas atau urutan kerja dalam suatu sistem yang akan digunakan. Selain itu, *activity diagram* juga berfungsi untuk menggambarkan atau mengatur alur dari antarmuka sistem yang akan digunakan. Diagram ini terdiri dari komponen-komponen dengan bentuk khusus yang terhubung dengan panah, yang menunjukkan urutan aktivitas yang terjadi di dalamnya. Umunya, *activity diagram* dibuat berdasarkan satu atau beberapa *use case* yang terdapat pada *use case diagram* (Kurniawan dan Sewaka, 2023).

*Activity diagram* merupakan *tool* yang berguna dalam menggambarkan rangkaian lengkap aktivitas untuk satu proses tunggal, seperti halnya pendekatan alur proses lainnya. Salah satu keunggulan utama dari teknik pemodelan alur proses ini adalah kemampuannya untuk secara efektif menangkap dan mengkomunikasikan proses-proses kompleks dalam rantai proses *back-office* yang panjang. Activity diagram membantu dalam menyajikan informasi secara terperinci namun mudah dipahami (Kratochvíl dan McGibbon, 2002).

Simbol – simbol pada *use case diagram* diberikan pada tabel 2. 3.

Tabel 2. 3 Simbol – simbol *activity diagram*

|  |  |  |
| --- | --- | --- |
| Nama Simbol | Deskripsi | Gambar |
| *Initial State* | Awal *activity diagram.* |  |
| *Activities* | Aktivitas yang dilakukan atau sedang terjadi dalam sistem. Biasanya diawali dengan kata kerja. |  |
| *Decision* | Menggambarkan suatu keputusan/ tindakan yang perlu dipilih pada kondisi tertentu. |  |
| *Join* | Beberapa aliran pada tahap tertentu bergabung menjadi satu. |  |
| *End State* | Akhir *activity diagram.* |  |
| *Swimlane* | Memisahkan organisasi bisnis yang bertanggung jawab terhadap ativitas yang terjadi |  |

1. *Sequence Diagram*

*Sequence diagram* adalah salah satu jenis UML yang digunakan untuk menggambarkan interaksi antara objek dalam suatu sistem secara urutan waktu. Diagram ini menunjukkan bagaimana objek-objek saling berinteraksi dalam suatu skenario tertentu, menunjukkan pesan-pesan yang dikirim antar objek, serta urutan kejadian yang terjadi (Trilaksono, 2022).

Dalam *sequence diagram*, objek-objek direpresentasikan sebagai kotak vertikal yang terhubung dengan garis-garis panah yang menunjukkan aliran pesan antar objek. Pesan-pesan ini menggambarkan komunikasi antar objek dalam urutan waktu tertentu, sehingga membantu dalam memvisualisasikan bagaimana objek-objek saling berinteraksi dalam suatu proses atau skenario.

Sequence diagram sangat berguna dalam memodelkan interaksi antar objek dalam suatu sistem, membantu dalam memahami alur kerja sistem, serta memperjelas bagaimana objek-objek saling berkomunikasi dalam suatu proses. Diagram ini juga dapat digunakan untuk mendukung analisis dan perancangan sistem secara lebih terperinci.

Simbol – simbol pada *sequence diagram* diberikan pada tabel 2. 4.

Tabel 2. 4 Simbol – simbol *sequence diagram*

|  |  |  |
| --- | --- | --- |
| Nama Simbol | Deskripsi | Gambar |
| *Actor* | Menggambarkan seorang pengguna yang berada di luar sistem dan sedang berinteraksi dengan sistem | Intro to Use Case Types | Systems Flow, Inc |
| *Object* | Menggambarkan *object* atau *class.* |  |
| *Lifeline* | Menyatakan kehidupan suatu *object* |  |
| *Activation* | Mewakili proses durasi aktivasi sebuah operasi |  |
| *Send Message* | Menyatakan bahwa suatu *object* mengirimkan data/ masukan/ informasi ke objek lainnya |  |
| *Return Messege* | Pesan yang dikembalikan dari *object* tertentu. |  |
| *Recursice/ Self Messege* | Pesan untuk *object* itu sendiri. |  |

1. *Class Diagram*

*Class diagram* adalah salah satu jenis diagram UML yang digunakan untuk menggambarkan struktur statis dari suatu sistem berbasis objek (Trilaksono 2022). Diagram ini menunjukkan kelas-kelas (*classes*) dalam sistem beserta atribut-atribut dan metode-metode yang dimiliki oleh setiap kelas, serta hubungan antar kelas seperti pewarisan (*inheritance*), asosiasi, agregasi, dan lain sebagainya.

Dalam *class diagram*, kelas direpresentasikan sebagai sebuah kotak yang terdiri dari tiga bagian yaitu bagian atas yang berisi nama kelas, bagian tengah yang berisi atribut-atribut kelas, dan bagian bawah yang berisi metode-metode kelas. Hubungan antar kelas ditunjukkan dengan panah yang menghubungkan kelas-kelas tersebut, dengan label yang menjelaskan jenis hubungan yang terjadi.

Simbol – simbol pada *class diagram* diberikan pada tabel 2. 5.

Tabel 2. 5 Simbol – simbol *class diagram*

|  |  |  |
| --- | --- | --- |
| Nama Simbol | Deskripsi | Gambar |
| Kelas | Kelas pada struktur sistem. | |  | | --- | | Class | | * attributes | | + operations() | |
| Asosiasi | Menggambarkan hubungan antara dua kelas yang bersifat statis. |  |
| Agregasi | Menggambarkan hubungan antara dua kelas, di mana satu kelas merupakan bagian dari kelas lain, namun keduanya dapat berdiri secara independen. |  |
| Pewarisan/ *Inheritence* | Menggambarkan hubungan antar kelas dimana satu kelas mewarisi kelas lainnya. |  |

1. *Rational Unified Process*

*Rational Unified Process* (RUP) adalah pendekatan pengembangan perangkat lunak yang terstruktur dan disiplin yang menekankan pendekatan iteratif dan inkremental, fokus pada kualitas tinggi, integrasi UML, manajemen proyek yang efektif, dan orientasi pada kebutuhan pengguna. RUP memberikan panduan komprehensif untuk mengelola siklus hidup pengembangan perangkat lunak, mulai dari analisis kebutuhan hingga pengiriman produk akhir kepada pengguna, dengan tujuan menghasilkan solusi perangkat lunak yang berkualitas, sesuai dengan kebutuhan pengguna, dan meminimalkan risiko proyek (Iacob dan Mihălcescu, 2021).

Dalam RUP, terdapat beberapa fase pengembangan sistem yang terbagi menjadi 4 fase utama, masing-masing terkait dengan tujuan dan hasil tertentu. Fase pada RUP adalah sebagai berikut :

1. Fase *Inception*

Fase awal ini bertujuan untuk melakukan studi awal yang mendetail untuk mengidentifikasi kebutuhan sistem yang akan datang. Ini melibatkan analisis tujuan bisnis, analisis peluang, identifikasi cakupan proyek, solusi yang akan diimplementasikan, kriteria efisiensi, penilaian risiko, dan estimasi sumber daya. Fase ini juga mencakup pembuatan jadwal awal untuk eksekusi sistem yang mengkoordinasikan keempat fase kerja.

1. Fase *Elaboration*

Pada tahap ini, arsitektur sistem dikembangkan berdasarkan domain bisnis yang diidentifikasi. Rencana kerja dibuat, dan arsitektur yang dirancang dioptimalkan secara efisien. Keputusan tentang arsitektur sistem dibuat berdasarkan kebutuhan yang dijelaskan dengan benar oleh para pemangku kepentingan dan diimplementasikan melalui kasus penggunaan bisnis. Faktor risiko utama diatasi, dan tujuan ditetapkan untuk mengoptimalkan kualitas arsitektur sistem

1. Fase *Construction*

Fase ini melibatkan strukturisasi proyek kompleks menjadi beberapa subproyek, masing-masing mewakili iterasi dalam pengembangan RUP. Ini mengikuti pendekatan iteratif dan inkremental untuk mengurangi risiko pengembangan sistem yang tidak berfungsi dengan baik. Aktivitas meliputi penanganan kebutuhan yang tidak dijelaskan dalam tahap pengembangan, desain sistem yang detail, realisasi, dan pengujian fungsionalitas sistem.

1. Fase *Transition*

Tahap terakhir dari siklus proyek RUP berfokus pada pengiriman sistem kepada pengguna akhir. Aktivitas meliputi pengujian dan perbaikan produk, mengidentifikasi aspek non-fungsional, dan menangani kemungkinan kesalahan. Versi sistem yang sedang diuji secara bertahap digantikan oleh versi final setelah semua koreksi telah dilakukan berdasarkan hasil uji.



Gambar 2. 20 Fase pengembangan sistem RUP

1. Pengujian *Black Box*

Pengujian *black box* adalah salah satu teknik pengujian perangkat lunak di mana pengujian dilakukan tanpa memperhatikan struktur internal atau logika dari program yang diuji. Pengujian ini berfokus pada input dan output yang dihasilkan oleh perangkat lunak berdasarkan spesifikasi kebutuhan yang telah ditentukan. Dalam pengujian *black box*, pengujian dilakukan dari sudut pandang pengguna akhir tanpa memerlukan pengetahuan tentang implementasi internal dari perangkat lunak (Nidhra dan Dondeti, 2012).

Keuntungan pengujian *black box* meliputi kemudahan dalam penerapan karena fokus pada spesifikasi fungsional, independen dari implementasi internal, serta dapat dilakukan tanpa pengetahuan detail tentang kode program. Namun, kelemahan pengujian *black box* adalah keterbatasan dalam menemukan kesalahan yang terkait dengan logika internal program dan kurangnya cakupan pengujian terhadap semua kemungkinan jalur eksekusi program (Praniffa dkk, 2023).

1. Pengujian Metrik Regresi

Evaluasi metrik digunakan untuk mengukur kinerja model regresi, dalam hal ini model ensemble learning yang melibatkan *XGBoost* dan *Random Forest*. Evaluasi metrik yang dapat digunakan diantaranya *R-Squared Score,* *Mean Squared Error, Root Mean Squared Error,* dan *Mean Abosolute Error*.

1. *R-Squared Score*

*R-Squared Score* (R­­2 *Score*) atau biasa dikenal dengan koefisien determinasi adalah metrik evaluasi yang digunakan dalam analisis regresi untuk mengukur seberapa baik model regresi sesuai dengan data yang diamati. Nilai *R-squared* berkisar antara 0 hingga 1 dan memberikan informasi tentang seberapa besar variabilitas variabel dependen dapat dijelaskan oleh variabel independen dalam model regresi. Secara lebih spesifik, R-squared mengindikasikan persentase variabilitas dari variabel dependen yang dapat dijelaskan oleh variabel independen dalam model regresi. Semakin tinggi nilai R-squared, semakin baik model regresi dapat menjelaskan variasi dalam data. Sebaliknya, nilai R-squared yang rendah menunjukkan bahwa model regresi tidak mampu menjelaskan variasi yang signifikan dalam data (Chicco, Warrens dan Jurman, 2021).

Secara matematis, *R-squared* dituliskan dalam persamaan 2.10.

(2.10)

Dimana :

m = Jumlah data dalam dataset

= Data ke-i

= Prediksi nilai ke-

= Nilai aktual data ke-

= Nilai rata-rata aktual

1. *Mean Squared Error*

*Mean Squared Error* (MSE) adalah salah satu metrik evaluasi yang umum digunakan dalam analisis regresi untuk mengukur seberapa baik model regresi memprediksi nilai aktual dari variabel dependen. MSE menghitung rata-rata dari kuadrat selisih antara nilai prediksi yang dihasilkan oleh model regresi dan nilai aktual dari variabel dependen untuk setiap observasi dalam dataset (Chicco, Warrens dan Jurman, 2021).

Secara matematis, MSEdituliskan dalam persamaan 2.11.

(2.11)

MSE memberikan informasi tentang seberapa dekat prediksi model regresi dengan nilai aktual dari variabel dependen. Semakin kecil nilai MSE, semakin baik model regresi dalam memprediksi nilai aktual. Namun, MSE cenderung memberikan bobot yang lebih besar pada pencilan (outliers) karena kuadrat selisihnya.

1. *Root Mean Squared Error*

*Root Mean Square Error* (RMSE) adalah metrik evaluasi yang umum digunakan dalam analisis regresi untuk mengukur tingkat kesalahan prediksi dari model regresi dalam unit yang sama dengan variabel dependen. RMSE menghitung akar kuadrat dari rata-rata dari kuadrat selisih antara nilai prediksi yang dihasilkan oleh model regresi dan nilai aktual dari variabel dependen untuk setiap observasi dalam dataset (Chicco, Warrens dan Jurman, 2021).

Secara matematis, MSEdituliskan dalam persamaan 2.12.

(2.12)

RMSE memiliki interpretasi yang lebih intuitif daripada MSE karena hasilnya dalam unit yang sama dengan variabel dependen, sehingga lebih mudah dipahami oleh orang yang tidak terbiasa dengan statistik. Seperti halnya MSE, RMSE juga cenderung memberikan bobot yang lebih besar pada pencilan (outliers) karena kuadrat selisihnya.

1. *Mean Absolute Error*

*Mean Absolute Error* (MAE) adalah metrik evaluasi yang digunakan dalam analisis regresi untuk mengukur kesalahan prediksi dari model regresi dalam skala yang sama dengan variabel dependen. MAE menghitung rata-rata dari selisih absolut antara nilai prediksi yang dihasilkan oleh model regresi dan nilai aktual dari variabel dependen untuk setiap observasi dalam dataset (Chicco, Warrens dan Jurman, 2021).

Secara matematis, MSEdituliskan dalam persamaan 2.13.

(2.13)

MAE memberikan informasi tentang seberapa besar kesalahan rata-rata dari prediksi model regresi dalam skala yang sama dengan variabel dependen. MAE lebih toleran terhadap pencilan (*outliers*) daripada MSE dan RMSE karena menggunakan nilai absolut dari selisih, sehingga tidak memberikan bobot yang lebih besar pada pencilan.

# BAB III METODOLOGI PENELITIAN

1. Metode Pengumpulan Data

Beberapa metode pengumpulan data yang digunakan dalam penelitian ini adalah :

1. Studi Literatur

Peneliti melakukan eksplorasi yang mendalam mengenai metode yang digunakan yakni *ensemble learning* berdasarkan tinjauan literatur terdahulu yang diperoleh dari berbagai sumber seperti jurnal, buku, *website* dan video pembelajaran. Peneliti juga mempelajari dan menganalisis model *XGBoost* dan *Random Forest* berdasarkan pustaka yang ada.

1. Wawancara

Pengumpulan informasi melalui percakapan secara langsung bertujuan untuk mendapatkan gambaran mengenai data, pemodelan dan alur sistem yang akan dibuat. Wawancara juga dilakukan untuk memperdalam pemahaman mengenai metode dan model yang digunakan. Wawancara dilakukan dengan Bapak Syahrul Usman, Data Management Officer di Kalla Group, yang memberikan informasi tentang data dan kebutuhan sistem.

1. Dataset

Dalam penelitian ini, dikumpulkan dataset yang berasal dari *Database Management Service* (DMS) *Service* Kalla Toyota yang dilakukan *export* ke dalam format *Comma Separated Values* (CSV). Dataset terdiri dari 3 tabel yakni *workorder* sebagai dataset utama yang terdiri dari 63 *colums* dan 770.472 *entries*, *partorder* yang terdiri dari 34 *columns* dan 2.064.733 *entries*, dan *retailsales* yang terdiri dari 72 *columns* dan 222.600 *entries*. Tabel *workroder* berisi data service pada layanan Aftersales Kalla Toyota rentang 2017 sampai 2023. Tabel *partoder* berisi data penjualan/penggantian *sparepart*. Adapun *retailsales* berisi data penjualan kendaraan.

1. Metode Pengembangan Sistem

Dalam penelitian ini, digunakan metode pengembangan sistem yaitu Rational Unified Process (RUP), yang terdiri dari tahapan – tahapan sebagai berikut :

1. *Inseption*

Pada fase *inseption*, dilakukan analisis mendalam terkait proses bisnis yang berjalan dan kebutuhan sistem. Selain itu, dilakukan peninjauan kebutuhan data yang disertai dengan analisis teori mengenai metode yang digunakan berdasarkan referensi yang ada, baik model tunggal *XGBoost* & *Random Forest*, maupun penggunaan teknik *ensemble learning*.

1. *Elaboration*

Pada fase ini dilakukan perencanaan perencanaan pembangunan model *ensemble* dari *XGBoost* dan *Random Forest*. Selain itu, dilakukan pengembangan sistem dengan perancangan arsitektur menggunakan UML yang terdiri dari *use case diagram, actoivity diagram, sequence diagram,* dan *class diagram.* Perancangan dilanjutkan dengan melakukan desain *interface web app*.

1. *Construction*

Pada fase ketiga ini, dilakukan implementasi, pengujian dan penyempurnaan sistem yang dibuat. Implementasi dimulai dengan membangun model ensemble dengan bahasa pemrograman Python. Kemudian dilakukan pengkodean untuk membangun *web app* dengan Express.js dari *environment* Node.js. Implementasi model *ensemble* dilakukan dengan bantuan FastAPI. Setelah itu dilakukan pengujian kinerja sistem secara individual dan penyesuaian berdasarkan hasil pengujian.

1. *Transtition*

*Transition* merupakan fase terakhir dimana dilakukan pengembangan akhir dan pengujian sistem secara menyeluruh. Pada tahap ini juga, dilakukan *deployment* agar fungsionalitas sistem dapat digunakan.

1. Waktu dan Tempat Penelitian
2. Waktu

Waktu pelaksanaan penelitian berlangsung dari bulan Februari 2024 sampai Mei 2024 dengan rincian yang ditampilkan pada tabel 3. 1.

Tabel 3. 1 Waktu Penelitian

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| No. | Detail Kegiatan | Waktu Penelitian (2024) | | | | | | | | | | | | | | | |
| Januari | | Februari | | | | Maret | | | | April | | | | Mei | |
| 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 |
| 1 | *Inseption* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2 | *Elaboration* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3 | *Construction* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4 | *Transtition* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

1. Tempat

Penelitian ini bertempat di Laboratorium *Computer Science & Artificial Intelligence*, Fakultas Teknik, Universitas Halu Oleo

1. Analisis Kebutuhan Sistem

Analisis kebutuhan sistem merupakan proses penentuan spesifikasi kebutuhan sistem yang meliputi elemen atau komponen yang diperlukan untuk sistem yang akan dibangun sampai dengan sistem tersebut diimplementasikan. Objektif analisis kebutuhan sistem mencakup kebutuhan fungsional (*input*, proses, *output*) dan kebutuhan nonfungsional (*hardware*, *software*).

1. Kebutuhan Fungsional

Analisis kebutuhan fungsional merupakan proses identifikasi dan terhadap hal – hal yang dibutuhkan sistem agar dapat membangun sistem yang berjalan sesuai prosedur. Kebutuhan dalam mebangun sistem mencakup kebutuhan *input*, proses, dan *output*. Dalam hal prediksi waktu penggantian baterai aki menggunakan *ensemble learning* metode *XGBoost* dan *Random Forest*, analisis kebutuhan fungsionalnya dijelaskan sebagai berikut :

1. Kebutuhan *Input*

Sistem ini membutuhkan *input* berupa data spesifikasi kendaraan mobil yang akan dilakukan prediksi. Atribut input berdasarkan *features selection* yang dilakukan.

1. Kebutuhan Proses

Kebutuhan proses dalam sistem ini berupa kemampuan untuk melakukan prediksi waktu penggantian baterai aki berdasarkan model *ensembel* dari *XGBoost* dan *Random Forest*.

1. Kebutuhan Output

*Output* yang dibutuhakn dalam sistem ini berupa hasil prediksi waktu penggantian baterai aki, dalam hal ini tanggal prediksi waktu penggantian.

1. Kebutuhan Nonfungsional

Analisis kebutuhan nonfungsional mencakup prosesm analisis terhadap kebutuhan sumber daya dan *environment* yang dibutuhkan dalam membangun sistem. Secara garis besar, kebutuhan nonfungsional dibagi mejadi kebutuhan perangkat keras (*hardware*) dan kebutuhan perangkat lunak (*software*).

1. Kebutuhan Perangkat Keras

Untuk membangun sistem yang dirancang, dibutuhkan perangkat keras sebagai media implementasi sistem, dimana kebutuhan perangkat keras untuk sistem ini ditunjukkan pada tabel 3. 2.

Tabel 3. 2 Spesifikasi Perangkat Keras

|  |  |  |
| --- | --- | --- |
| No. | Nama Perangkat | Spesifikasi |
| 1 | Laptop | Victus *by* HP Gaming Laptop 16-s0xxx |
| 2 | *Processor* | AMD Ryzen 5 7640HS w/ Radeon 760M Graphics, 4301 Mhz, 6 Core(s), 12 Logical Processor(s) |
| 3 | *Monitor* | 16,1 Inch |
| 4 | *Memori* | 16 GB DDR5-5600 MHz RAM |
| 5 | Penyimpanan | 512 GB PCIe® Gen4 NVMe™ TLC M.2 SSD |

1. Kebutuhan Perangkat Lunak

Kebutuhan perangkat lunak ditunjukkan pada tabel 3. 3.

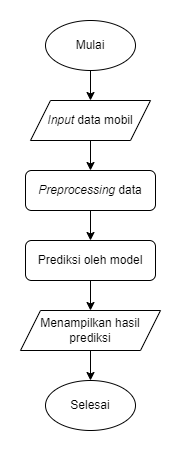
Tabel 3. 3 Spesifikasi Perangkat Lunak

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Nama | Fungsi | Spesifikasi |
| 1 | Windows | Sistem operasi | Windows 11 *Home* |
| 2 | Python | Bahasa pemrograman | Versi 3.11.15 |
| 3 | Node.js | Javascript *runtime* | Versi 20.11.1 |
| 4 | Express.js | Node.js *framework* | Versi 4.18.1 |
| 5 | Visual Studio Code | *Integrated Development Environment* | Versi 1.86 |

1. Perancangan Sistem

Perancangan sistem mencakup perencanaan alur kerja yang akan dilakukan selama penelitian guna mencapai hasil yang diinginkan. Dalam penelitian ini akan dibuat sebuah sistem prediksi waktu penggantian baterai aki yang mengimplementasikan *ensemble learning* metode *XGBoost* dan *Random Forest.*

1. Perancangan Alur Sistem Secara Umum



Gambar 3. 1 Alur sistem secara umum

Berikut penjelasan alur sistem berdasarkan gambar 3. 1 di atas :

1. *Input* Data Mobil

Pada langkah awal, pengguna memasukan data mobil sesuai dengan aitribut/fitur pada model yang dibuat. Adapun atribut *input* berupa *mileage, age, area, vehicleModel, batteryType, numOfReplacement*.

1. *Preprocessing* Data

Data dikirimkan ke FastAPI untuk selanjutnya dilakukan *preprocessing* data berupa transformasi data agar sesuai dengan input model yang akan dibuat.

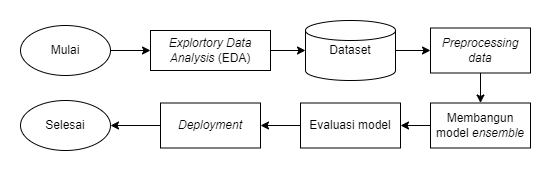
1. Prediksi oleh Model

Hasil dari *preprocessing* data menjadi input dari model *ensemble* yang akan dibuat. Selanjutnya hasil dari prediksi dikirimkan dari model berupa *respone* ke *website*.

1. Menampilkan Hasil Prediksi

*Website* menerima respon API berupa hasil prediksi yang bertipe data integer (prediksi waktu penggantian dalam hari). Selanjutnya dilakukan perhitungan prediksi tanggal penggantian berdasarkan waktu penggantian terakhir untuk ditampilkan pada laman *web*.

1. Perancangan Alur Proses *Modelling*



Gambar 3. 2 Alur Proses *Modelling*

Dalam proses *modelling* terdapat beberapa tahapan, dari *Exploratory Data Analysis* (EDA)sehingga dihasilkan dataset, *preprocessing, modelling*, evaluasi, hingga deployment. Penjelasan setiap tahapan adalah sebagai berikut :

1. *Exploratory Data Analysis*

Pada langkah awal, dilakukan EDA terhadap data mentah untuk mendapatkan dataset yang dibutuhkan untuk *modelling*. Dilakukan proses EDA mulai dari mengetahui tipe data, analisis dimensi data (*multivariate*), dan *numerical summary*. Setelah itu dilakukan *features selection* untuk menyeleksi fitur/atribut untuk dataset baru yang akan digunakan dalam *modelling*.

1. *Dataset*

Hasil dari proses EDA menghasilkan *dataset* baru yang akan digunakan untuk membangun model. Dataset ini telah melalui proses *features selection* sehingga hanya berisi atribut yang dibutuhkan dalam *modelling*.

1. *Preprocessing* *Data*

Sebelum masuk ke tahap *modelling*, dilakukan *preprocessing* dalam hal ini *data cleaning* berupa penangan *missing value*, *removing outlier*, hingga normalisasi data.

1. Membangun Model *Ensemble*

Selanjutnya membangun model *ensemble* berdasarkan model *XGBoost* dan *Random Forest.* Dalam membangun model, digunakan teknik ensemble learning berupa *bagging, boosting, voting*, atau *stacking*.

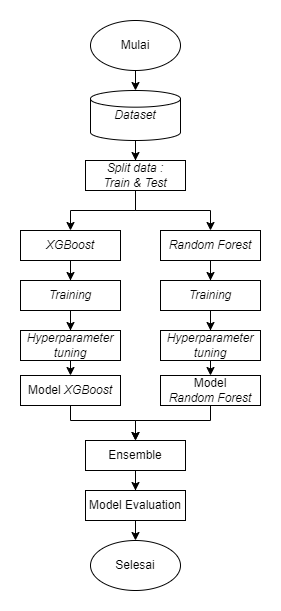
1. Evaluasi Model

Evaluasi model dilakukan dengan metric evaluasi berupa R2 *score*, MSE, RMSE dan MAE. Setelah diperoleh model yang optimal, dilakukan proses *deployment*.

1. *Deployment*

Proses *deployment* menggunakan FastAPI yang berbasis bahasa pemrograman Python. API ini dapat menerima *request* dari *website* dan mengembalikan *response* ke *website* berupa hasil prediksi berdasarkan model yang dibangun.

1. Perancangan Alur *Ensemble Learning*



Gambar 3. 3 Alur *Ensemble Learning*

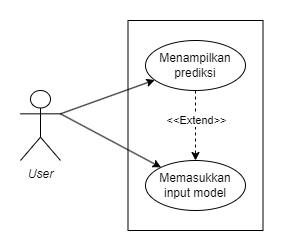
Alur *ensemble learning* dimulai dari pembagian *dataset* ke *data training* dan *data testing*. Selanjutnya data dilatih menggunakan 2 arsitektur yakni *XGBoost* dan *Random Forest*. Masing – masing proses selanjutnya dilakukan *hyperparameter* tuning untuk memperoleh model tunggal terbaik. Kemudian dihasilkan dua model yakni model *XGBoost* dan *Random* *Forest*. Selanjutnya dilakukan *ensemble* terhadap kedua model tadi. Teknik ensemble yang digunakan dapat berupa *voting, bagging, boosting* atau *stacking*. Terakhir dilakukan evaluasi untuk mengetahui tingkat keakuratan model menggunakan metrik evaluasi.

1. Perancangan UML

Pengembangan perangkat lunak melibatkan perancangan *Unified Modeling Language* (UML) untuk memodelkan, mendokumentasikan, dan merancang sistem secara visual. UML yang dirancang terdiri dari *use case diagram, activity diagram, sequence diagram,* dan *class diagram*.

1. *Use Case Diagram*

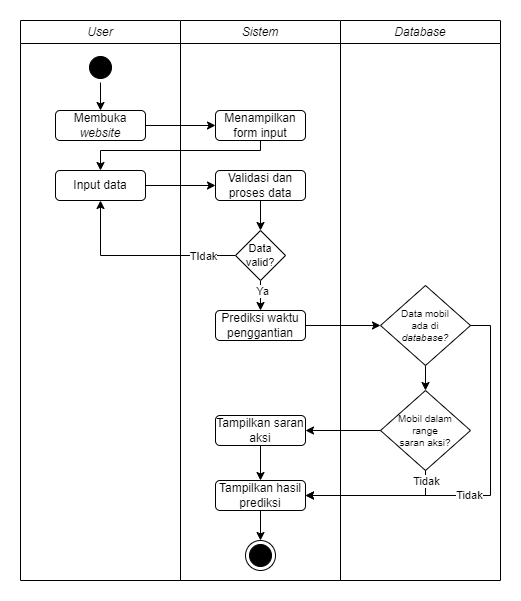
*Use case diagram* memodelkan interaksi antara aktor dan sistem. *Use case diagram* membantu dalam merencanakan dan memvisualisasikan skenario penggunaan sistem, serta menunjukkan bagaimana aktor akan berinteraksi dengan sistem untuk mencapai tujuan. Berikut *use case diagram* dari sistem yang akan dibangun.



Gambar 3. 4 *Use case diagram* sistem

1. *Activity Diagram*
2. *Activity Diagram* Prediksi Waktu Penggantian

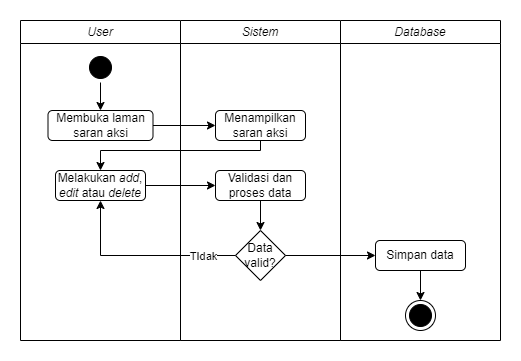
Adapun *acitivity diagram* prediksi waktu penggantian adalah sebagai berikut.



Gambar 3. 5 *Activity diagram* prediksi waktu penggantian

1. *Activity Diagram* Kelola Saran Aksi

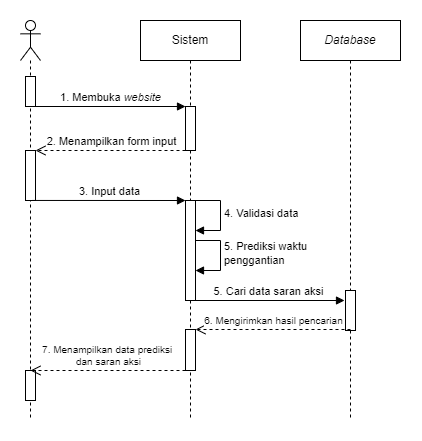
Adapun *acitivity diagram* kelola saran aksi dalam sistem yang akan dibangun adalah sebagai berikut.



Gambar 3. 6 *Activity diagram* kelola saran aksi

1. *Sequence Diagram*
2. *Sequence Diagram* Prediksi Waktu Penggantian

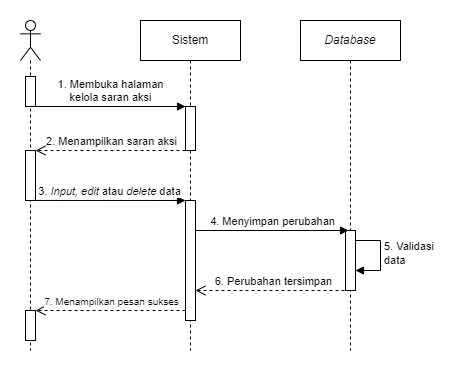
Berikut ditampilkan *sequence diagram* prediksi waktu penggantian.



Gambar 3. 7 *Sequence diagram* prediksi waktu penggantian

1. *Sequence Diagram* Kelola Saran Aksi

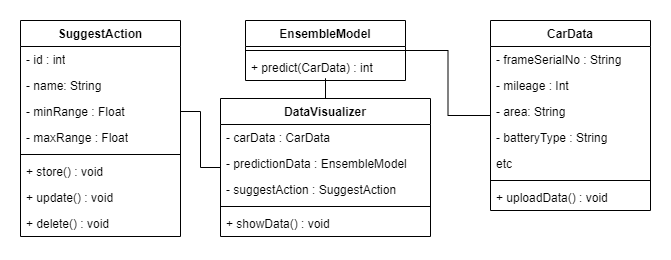
Berikut ditampilkan *sequence diagram* kelola saran aksi.



Gambar 3. 8 *Sequence diagram* kelola saran aksi

1. *Class Diagram*

Perancangan *class diagram* untuk sistem yang akan dibuat ditunjukkan pada gambar berikut.

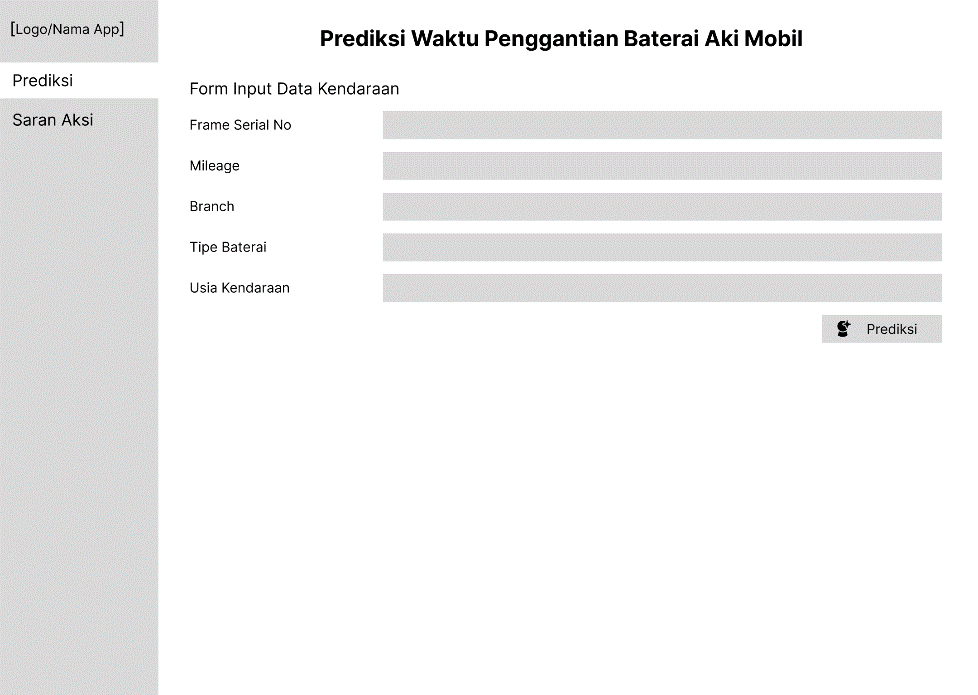


Gambar 3. 9 *Class diagram* sistem

1. Perancangan *Interface*

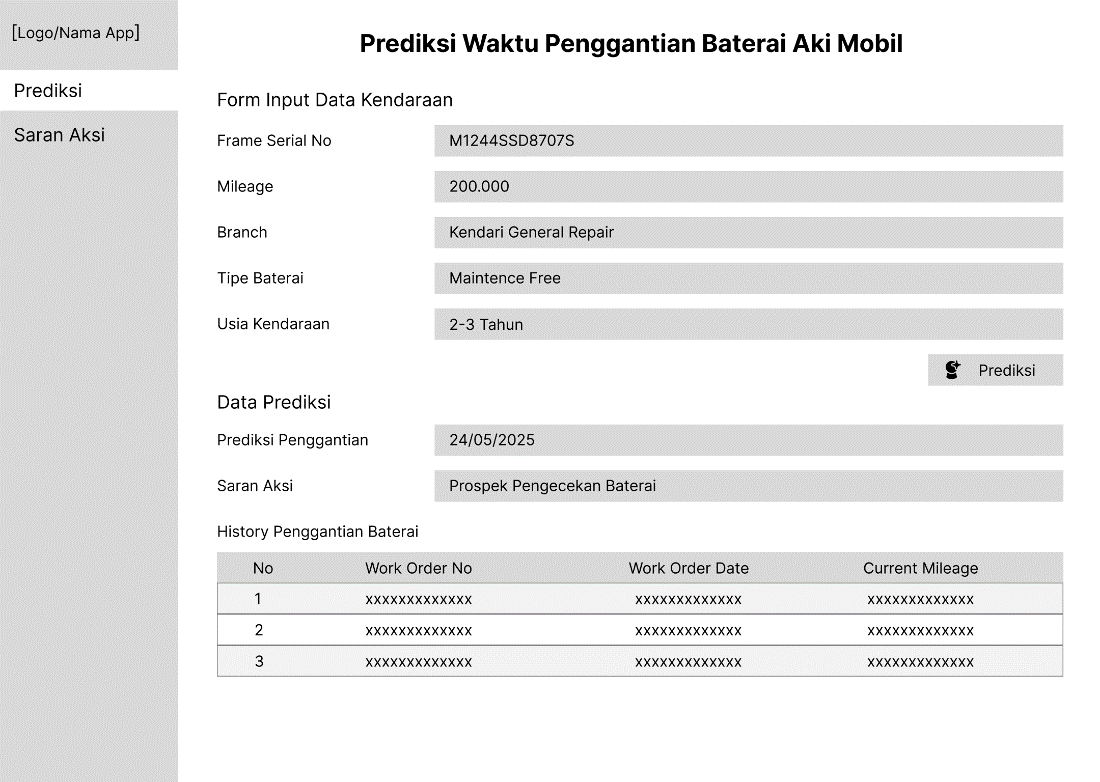
Perancangan *interface* dilakukan dengan membuat gambaran umum tampilan *website*. Dalam konteks sistem prediksi, *interface* akan menghubungkan sistem dengan user melalui media visual dan berperan dalam proses *input-output* sistem.

1. Halaman Form Prediksi Penggantian Baterai Aki



Gambar 3. 10 Halaman form prediksi penggantian baterai aki

1. Halaman Hasil Prediksi



Gambar 3. 11 Halaman hasil prediksi

1. Halaman Kelola Saran Aksi



Gambar 3. 12 Halaman kelola saran aksi

1. Halaman Menambahkan Saran Aksi



Gambar 3. 13 Halaman menambahkan saran aksi

1. Pengujian Fungsionalitas Sistem

Pengujian fungsionalitas sistem menggunakan metode *black box*. Pengujian *black box* dilakukan tanpa memperhatikan struktur internal atau logika dari program yang diuji. Pengujian ini berfokus pada *input* dan *output* yang dihasilkan oleh perangkat lunak berdasarkan spesifikasi kebutuhan yang telah ditentukan dari sudut pandang pengguna.

Skenario pengujian *black box* pada sistem yang akan dibuat ditunjukkan pada Tabel 3. 4.

Tabel 3. 4 Skema pengujian *black box*

|  |  |  |  |
| --- | --- | --- | --- |
| No | Fitur yang Diuji | Aksi | Reaksi yang Diharapkan |
| 1 | Halaman prediksi | Membuka sistem | Menampilkan form prediksi |
| 2 | Fitur prediksi | Mengisi form dan menekan tombol prediksi | Menampilkan data hasil prediksi |

**Tabel 3. 4 Skema pengujian *black box***

|  |  |  |  |
| --- | --- | --- | --- |
| No | Fitur yang Diuji | Aksi | Reaksi yang Diharapkan |
| 1 | Halaman prediksi | Membuka sistem | Menampilkan form prediksi |
| 2 | Fitur prediksi | Mengisi form dan menekan tombol prediksi | Menampilkan data hasil prediksi |
| 3 | Halaman saran aksi | Membuka halaman saran aksi | Menampilkan halaman saran aksi |
| 4 | Tambah saran aksi | Mengisi form tambah saran aksi dan menekan tombol simpan | Menampilkan pesan hasil menambahkan saran aksi |
| 5 | Edit saran aksi | Mengisi form edit dan menekan tombol simpan | Menampilkan pesan hasil edit saran aksi |
| 6 | Hapus saran aksi | Menekan tombol hapus saran aksi | Menampilkan pesan saran aksi terhapus |

1. Pengujian Model *Ensemble Learning*

Evaluasi model *ensemble learning* dilakukan dengan metric evaluasi, yakni koefisien determinasi (R2 *score*), *Mean Squared Error* (MSE), *Root Mean Squared Error* (RMSE), dan *Mean Absolute Error* (MAE). Tujuan pengujian ini adalah untuk membandingkan performa model *ensemble learning* dengan model individual *XGBoost* dan *Random Forest*, menentukan model *ensemble learning* terbaik, dan mengevaluasi stabilitas serta ketahanannya terhadap variasi data.

Berikut merupakan skema pengujian model *ensembel learning*. Skema pengujian dibuat dengan membuat data dummy dan menampilkan contoh hasil pengujian. Data *dummy* dalam skema pengujian ini ditunjukkan pada tabel 3. 5.

Tabel 3. 5 Data *dummy* penggantian baterai aki

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No. | Kilometer | Area | Usia Kendaraan | Jenis Baterai | *Time* (*Target*) | *Time* (Prediksi) |
| 1 | 10.000 | InCity | 0-1 Tahun | Kering | 365 | 358 |
| 2 | 20.000 | OutCity | 1-2 Tahun | Basah | 274 | 281 |
| 3 | 30.000 | InCity | 2-3 Tahun | Kering | 330 | 323 |
| 4 | 40.000 | OutCity | 3-4 Tahun | Basah | 248 | 255 |
| 5 | 50.000 | InCity | 4-5 Tahun | Kering | 305 | 312 |
| 6 | 60.000 | OutCity | 5-6 Tahun | Basah | 221 | 214 |
| 7 | 70.000 | InCity | 6-7 Tahun | Kering | 282 | 275 |
| 8 | 80.000 | OutCity | 7-8 Tahun | Basah | 205 | 212 |
| 9 | 90.000 | InCity | >8 Tahun | Kering | 259 | 266 |
| 10 | 100.000 | OutCity | >8 Tahun | Basah | 182 | 189 |
| 11 | 110.000 | InCity | 0-1 Tahun | Kering | 372 | 365 |
| 12 | 120.000 | OutCity | 1-2 Tahun | Basah | 289 | 296 |
| 13 | 130.000 | InCity | 2-3 Tahun | Kering | 345 | 338 |
| 14 | 140.000 | OutCity | 3-4 Tahun | Basah | 262 | 269 |
| 15 | 150.000 | InCity | 4-5 Tahun | Kering | 320 | 327 |
| 16 | 160.000 | OutCity | 5-6 Tahun | Basah | 235 | 228 |
| 17 | 170.000 | InCity | 6-7 Tahun | Kering | 297 | 304 |
| 18 | 180.000 | OutCity | 7-8 Tahun | Basah | 219 | 226 |
| 19 | 190.000 | InCity | >8 Tahun | Kering | 274 | 281 |
| 20 | 200.000 | OutCity | >8 Tahun | Basah | 197 | 204 |

Berikut merupakan contoh skenario hasil pengujian model ensemble learning yang ditujukkan pada tabel 3. 6.

Tabel 3. 6 Contoh skenario hasil pengujian model *ensemble learning*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Skenario | Jumlah Data | Panjang Epoch | Learning Rate | Optimizer | R2 Score | MSE | RMSE | MAE |
| 1 | 12000 | 100 | 0,01 | Adam | 0,82 | 0,48 | 0,22 | 0,18 |
| 2 | 11000 | 200 | 0,005 | SGD | 0,87 | 0,31 | 0,17 | 0,14 |
| 3 | 10000 | 300 | 0,001 | RMSprop | 0,91 | 0,22 | 0,15 | 0,12 |
| 4 | 8000 | 400 | 0,1 | Adam | 0,93 | 0,18 | 0,14 | 0,11 |
| 5 | 6000 | 500 | 0,050 | SGD | 0,94 | 0,15 | 0,12 | 0,1 |

# DAFTAR PUSTAKA

Adiguno, S., Syahra, Y. & Yetri, M., 2022, Prediksi Peningkatan Omset Penjualan Menggunakan Metode Regresi Linier Berganda, *Jurnal Sistem Informasi Triguna Dharma (JURSI TGD)*, 1,4, 275.

Aditya, M.A., Mulyana, R.D., Eka, I.P. & Widianto, S.R., 2020, Penggabungan Teknologi untuk Analisa Data Berbasis Data Science, *Seminar Nasional Teknologi Komputer & Sains (SAINTEKS)*, 7,3, 51–56.

Agustin, W., Rio, U., Muzawi, R., Nasution, T. & Haryono, D., 2021, Penguatan Pengelolaan Website Desa Untuk Meningkatkan Layanan Administrasi Kependudukan di Desa Pasir Baru Rokan Hulu, *Abdiformatika: Jurnal Pengabdian Masyarakat Informatika*, 1,1, 8–17.

Ahmad, T., Iqbal, J., Ashraf, A., Truscan, D. & Porres, I., 2019, Model-Based Testing Using UML Activity Diagrams : A Systematic Mapping Study, *Computer Science Review*, 33, 98–113.

Anggraini, Y., Pasha, D., Damayanti, D. & Setiawan, A., 2020, Sistem Informasi Penjualan Sepeda Berbasis Web Menggunakan Framework Codeigniter, *Jurnal Teknologi dan Sistem Informasi*, 1,2, 64–70.

Ariansyah, P.M. & Wijaya, K., 2021, Rancang Bangun Sistem Informasi Akademik Berbasis Web: Studi Kasus: SD Negeri 18 Tanah Abang, *Jurnal Pengembangan Sistem Informasi dan Informatika*, 2,3, 138–156.

Asfan, M.J. & Arsana, I.M., 2021, Rancang Bangun Baterai Charger Otomotif, *Jurnal Rekayasa Mesin*, 06,01, 105–109.

Astuti, F.A., 2021, Pemanfaatan Teknologi Artificial Intelligence untuk Penguatan Kesehatan dan Pemulihan Ekonomi Nasional, *Jurnal Sistem Cerdas*, 4,1, 25–34.

Ayuni, G.N. & Fitrianah, D., 2019, Penerapan metode Regresi Linear untuk prediksi penjualan properti pada PT XYZ, *Jurnal Telematika*, 14,2, 79–86.

Azhari, R.Y., 2022, Web Service Framework : Flask Dan Fastapi, *Technology and Informatics Insight Journal*, 1,1, 58–65.

Bi, Q., Goodman, K.E., Kaminsky, J. & Lessler, J., 2019, What is Machine Learning? A Primer for The Epidemiologist, *American Journal of Epidemiology*, 188,12, 2222–2239.

Borup, D., Christensen, B.J., Mühlbach, N.S. & Nielsen, M.S., 2023, Targeting Predictors in Random Forest Regression, *International Journal of Forecasting*, 39,2, 841–868.

Breiman, L., 2001, Random Forests, *Machine Learning 45*, 5,32, 5–32.

Budiman, E., 2015, *Algoritma dan Pemrograman*, Fakultas Teknologi Informasi dan Komputer (FTIK) Universitas Mulawarman, Samarinda.

Budu, E., 2023, *Bagging, Boosting, and Stacking in Machine Learning*.

Chaudhuri, A.B., 2020, *Flowchart and Algorithm Basics The Art of Programming*, Mercury Learning and Information, New Delhi.

Chen, T. & Guestrin, C., 2016, *XGBoost: A Scalable Tree Boosting System*, *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, vols 13-17-Augu, 785–794.

Chicco, D., Warrens, M.J. & Jurman, G., 2021, The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation, *PeerJ Computer Science*, 7, 1–24.

Choi, S. & Hur, J., 2020, An Ensemble Learner-Based Bagging Model Using Past Output Data for Photovoltaic Forecasting, *Energies*, 13,6.

Dachi, J.M.A.S. & Sitompul, P., 2023, Analisis Perbandingan Algoritma XGBoost dan Algoritma Random Forest Ensemble Learning pada Klasifikasi Keputusan Kredit, *Jurnal Riset Rumpun Matematika dan Ilmu Pengetahuan Alam (JURRIMIPA)*, 2,2, 87–103.

Dian Rosadi, SKM, M., Dr. dr. Iwan Aflanie, M.Kes, Sp.F, S., Fauzie Rahman, S.KM, M., Rudi Fakhriadi, SKM, M.K. (Epid), Noor Ahda Fadillah, SKM, M.K. (Epid), Anggun Wulandari, S.KM, M.K., Nur Laily, S.KM, M.K., Nita Pujianti, S.Farm., Apt., M., Lia Anggraini, S.K. & Agus Muhammad Ridwan, S., 2021, *Manajemen data dalam perencanaan penyusunan program dan laporan bidang kesehatan*, Program Studi Kesehatan Masyarakat Fakultas Kedokteran Universitas Lambung Mangkurat, Banjarbaru.

Eleanora, F.N. & Fitriana, D., 2023, Layanan Purna Jual dan Perlindungan Hukum Bagi Konsumen, *Krtha Bhayangkara*, 17,2, 247–254.

Fachid, S. & Triayudi, A., 2022, Perbandingan Algoritma Regresi Linier dan Regresi Random Forest Dalam Memprediksi Kasus Positif Covid-19, *Jurnal Media Informatika Budidarma*, 6,1, 68.

Fahmi, A.F. & Arsana, I.M., 2021, Alat Pendeteksi Kerusakan Cell Accu, *Jurnal Rekayasa Mesin*, 06,03, 1–7.

FastAPI, 2024, *FastAPI Documentation*.

Fazakis, N., Kocsis, O., Dritsas, E., Alexiou, S., Fakotakis, N. & Moustakas, K., 2021, Machine Learning Tools for Long-Term Type 2 Diabetes Risk Prediction, *IEEE Access*, 9,July, 103737–103757.

Firdaus, H., Rustendi, E. & Herdiana, A., 2021, Analisis Konsumsi Arus Listrik pada Mobil Multi Purpose Vehicle, *Jurnal Ilmiah Teknologi Infomasi Terapan*, 8,1, 150–158.

Furqani, N. El & Muliono, R., 2021, Web-Based Library Information System Design at SDN 056004 Basilam, *Journal of Research Computer Science*, 1,1, 14–26.

Guntara, R.G., 2023, Deteksi Atap Bangunan Berbasis Citra Udara Menggunakan Google Colab dan Algoritma Deep Learning YOLOv7, *Jurnal Manajemen Sistem Informasi (JMASIF)*, 2,1, 9–18.

Iacob, I. & Mihălcescu, C.O., 2021, Designing An It System Using The Unified Relational Process, *Journal of Information Systems & Operations Management*, 15,2, 165–174.

Indahyanti, U., Azizah, N.L. & Setiawan, H., 2022, Pendekatan Ensemble Learning Untuk Meningkatkan Akurasi Prediksi Kinerja Akademik Mahasiswa, *Jurnal Sains dan Informatika*, 8,2, 160–169.

Indriani, D., 2022, Aplikasi Kecantikan dan Perawatan Wajah Berbasis Website, *Jurnal Teknik dan Science*, 1,2, 32–42.

Irawan, I., Qisthiano, M.R., Syahril, M. & Jakak, P.M., 2023, Optimasi Prediksi Kelulusan Tepat Waktu: Studi Perbandingan Algoritma Random Forest dan Algoritma K-NN Berbasis PSO, *Jurnal Pengembangan Sistem Informasi dan Informatika*, 4,4, 26–35.

J. Avanija, Sunitha, G., Madhavi, K.R., Kora, P. & SaiVittal, R.H., 2021, Prediction of House Price Using XGBoost Regression Algorithm, *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, 12,2, 2151–2155.

Jonsson, M. & Qvarnström, E., 2022, *A Performance Comparison On Rest APIs in Express.Js, Flask And ASP.NET Core* – PhD thesis, Mälardalen Universitet .

Kadi, S., 2021, *Measuring Maintainability and Latency of Node. js Frameworks* – PhD thesis, Blekinge Institute of Technology .

Kafil, M., 2019, Penerapan Metode K-Nearest Neighbors untuk Prediksi Penjualan Berbasis Web Padaboutiq Dealove Bondowoso, *Jurnal Mahasiswa Teknik Informatika (JATI)*, 3,2, 59–66.

Kalla Group, 2024, *Kalla Toyota*.

Karimah, C.N., Zain, A.T. & Nofiansyah, A.L., 2023, Analisa Baterai Sebagai Sumber Kelistrikan Kendaraan Roda Dua Ditinjau Dari Kapasitas Dan Efisiensi, *J-TETA : Jurnal Teknik Terapan*, 2,1, 1–11.

Kementerian Perindustrian Republik Indonesia, 2021, *Menperin: Industri Otomotif Jadi Sektor Andalan Ekonomi Nasional*.

Kiangala, S.K. & Wang, Z., 2021, An Effective Adaptive Customization Framework for Small Manufacturing Plants using Extreme Gradient Boosting-XGBoost and Random Forest Ensemble Learning Algorithms in an Industry 4.0 Environment, *Machine Learning with Applications*, 4,December 2020, 1–16.

Koishybayev, I. & Kapravelos, A., 2020, Mininode: Reducing the attack surface of node.js applications, *RAID 2020 Proceedings - 23rd International Symposium on Research in Attacks, Intrusions and Defenses*, 121–134.

Kratochvíl, M. & McGibbon, B., 2002, *UML Xtra-Light How to Specify Your Software Requirements*, Cambridge University Press.

Kurniawan, R. & Sewaka, 2023, Perancangan Sistem Informasi Aplikasi Klinik Berbasis Web Menggunakan Metode SDLC Protoypepada PT Pratama Abadi Industri, *LOGIC : Jurnal Ilmu Komputer dan Pendidikan*, 1,4, 854–865.

Li, W., Yin, Y., Quan, X. & Zhang, H., 2019, Gene Expression Value Prediction Based on XGBoost Algorithm, *Frontiers in Genetics*, 10,November, 1–7.

Liang, M., Chang, T., An, B., Duan, X., Du, L., Wang, X., Miao, J., Xu, L., Gao, X., Zhang, L., Li, J. & Gao, H., 2021, A Stacking Ensemble Learning Framework for Genomic Prediction, *Frontiers in Genetics*, 12,March, 1–9.

Lubis, F.R., Harahap, M.K. & Husein, A.M., 2019, Analisis Prediktif untuk Keputusan Bisnis : Peramalan Penjualan, *Jurnal Data Science Indonesia*, 1,1, 32–40.

Mardiani, E., Rahmansyah, N., Ningsih, S. & ..., 2023, Komparasi Metode Knn, Naive Bayes, Decision Tree, Ensemble, Linear Regression Terhadap Analisis Performa Pelajar Sma, *Innovative: Journal Of …*, 3,2, 13880–13892.

Natras, R., Soja, B. & Schmidt, M., 2022, Ensemble Machine Learning of Random Forest, AdaBoost and XGBoost for Vertical Total Electron Content Forecasting, *Remote Sensing*, 14,15, 1–34.

Nguyen, N., 2022, *Development & Deployment of a Web Server as An Executable with Node.js, Express.js and Vercel/pkg* – PhD thesis, Metropolia University.

Nidhra, S. & Dondeti, J., 2012, Black Box and White Box Testing Techniques - A Literature Review, *International Journal of Embedded Systems and Applications (IJESA)*, 2,2, 29–50.

Nissa, N.K., Nugraha, Y., Finola, C.F., Ernesto, A., Kanggrawan, J.I. & Suherman, A.L., 2020, Evaluasi Berbasis Data: Kebijakan Pembatasan Mobilitas Publik dalam Mitigasi Persebaran COVID-19 di Jakarta, *Jurnal Sistem Cerdas*, 3,2, 84–94.

Pal, S.C., Arabameri, A., Blaschke, T., Chowdhuri, I., Saha, A., Chakrabortty, R., Lee, S. & Band, S.S., 2020, Ensemble of machine-learning methods for predicting gully erosion susceptibility, *Remote Sensing*, 12,22, 1–25.

Peppes, N., Daskalakis, E., Alexakis, T., Adamopoulou, E. & Demestichas, K., 2021, Performance of Machine Learning-Based Multi-Model Voting Ensemble Methods for Metwork Threat Detection in Agriculture 4.0, *Sensors*, 21,22.

Praniffa, A.C., Syahri, A., Sandes, F., Fariha, U., Giansyah, Q.A. & Hamzah, M.L., 2023, Pengujian Black Box dan White Box Sistem Informasi Parkir Berbasis Web, *Jurnal Testing dan Implementasi Sistem Informasi*, 1,1, 1–16.

Prasetyo, S.Y.J., Christianto, Y.B. & Hartomo, K.D., 2019, Analisis Data Citra Landsat 8 OLI Sebagai Indeks Prediksi Kekeringan Menggunakan Machine Learning di Wilayah Kabupaten Boyolali dan Purworejo, *Indonesian Journal of Modeling and Computing*, 2,2, 25–36.

Rahman, S., Sembiring, A., Siregar, D., Khair, H., Gusti Prahmana, I., Puspadini, R. & Zen, M., 2023, *Python : Dasar dan Pemrograman Berorientasi Objek*, Penerbit Tahta Media.

Raschka, S., Patterson, J. & Nolet, C., 2020, Machine Learning in Python : Main Developments and Technology Trends in Data Science, Machine Learning, and Artificial Intelligence, *Information (Switzerland)*, 11,4.

Riza, F., 2022, Analisis dan Prediksi Data Penjualan Menggunakan Machine Learning dengan Pendekatan Ilmu Data, *Data Sciences Indonesia (DSI)*, 1,2, 62–68.

Sahin, E.K., 2020, Assessing the predictive capability of ensemble tree methods for landslide susceptibility mapping using XGBoost, gradient boosting machine, and random forest, *SN Applied Sciences*, 2,7, 1–17.

Sengkey, D.F., Kambey, F.D., Lengkong, S.P., Joshua, S.R. & Kainde, H.V.F., 2020, Pemanfaatan Platform Pemrograman Daring dalam Pembelajaran Probabilitas dan Statistika di Masa Pandemi CoVID-19, *Jurnal Informatika*, 15,4, 217–224.

Shukla, P., 2022, *Stacking Algorithms in Machine Learning*.

Shwartz-Ziv, R. & Armon, A., 2022, Tabular Data: Deep learning is Not All You Need, *Information Fusion*, 81,June 2021, 84–90.

Sunarko, B., Hasanah, U., Hidayat, S., Muhammad, N., Ardiansyah, M.I., Hakiki, M.K., Ananda, B.P. & Baroroh, L., 2023, Penerapan Stacking Ensemble Learning untuk Klasifikasi Efek Kesehatan Akibat, *Edu Komputika Journal*, 10,1, 55–63.

Tanuwidjaja, K. & Widjaja, A., 2022, Prediksi dan Analisis Time Series pada Data COVID-19, *Jurnal Strategi*, 4,1, 144–158.

Trilaksono, R.J., 2022, Rancang Bangun Aplikasi Kasir Berbasis Offline pada Primkop Kartika Viyata Virajati, *ULIL ALBAB : Jurnal Ilmiah Multidisiplin*, 1,4, 857–865.

Tyralis, H. & Papacharalampous, G., 2019, Scientists and Practitioners and Their Recent History, *Water*, 11,5, 1–37.

Wibisono, A., 2020, *Pengaruh Kualitas Pelayanan Purna Jual Terhadap Kepuasan dan Loyalitas Pelanggan pada Dealer Honda Cendana Megah Santosa Purwodadi* – PhD thesis, Universitas Muhammadiyah Surakarta .

Wijaya, A.C., Wibawa, I.G.A. & Darmawan, I.D.M.B.A., 2022, Pengembangan Restful Api uUntuk Model Machine Learning Indoor-Outdoor Dalam Aplikasi Peminjaman Ruangan, *Jurnal Pengabdian Informatika*, 1,1, 19–26.

Zhang, Y., Liu, J. & Shen, W., 2022, A Review of Ensemble Learning Algorithms Used in Remote Sensing Applications, *Applied Sciences (Switzerland)*, 12,17.