

BAB I

PENDAHULUAAN

1.1 Latar Belakang

Berkembangnya teknologi informasi dan transformasi digital di sektor perbankan, institusi keuangan kini mampu mengelola data nasabah dalam jumlah besar secara digital, pemanfaatan data tersebut mulai dikembangkan melalui pendekatan *machine learning* untuk mendukung pengambilan keputusan, termasuk dalam memprediksi minat nasabah terhadap produk deposito berjangka (Rasikh et al., 2023). Di tengah meningkatnya persaingan dan tuntutan efisiensi, bank dituntut untuk beralih dari pendekatan tradisional menuju pengelolaan data yang lebih cerdas dan berbasis teknologi atau *data-driven decision making* (Pamuk & Schumann, 2024).

Pada ekosistem bisnis modern yang didukung oleh data, Perusahaan seringkali menghadapi kendala *lead overload* yang merupakan kondisi dimana jumlah calon pelanggan yang masuk tidak sebanding dengan kemampuan tim sales untuk menindaklanjuti secara manual. Permasalahan ini juga terjadi dalam praktik pemasaran pada sektor perbankan, di mana tim sales masih sering melakukan kampanye atau *outbound call* tanpa didukung analisis data yang mendalam (Zaki et al., 2024). Nasabah dihubungi secara acak atau hanya berdasarkan segmentasi demografis sederhana, sehingga sebagian besar waktu terbuang untuk prospek yang kurang potensial. Kondisi ini berdampak pada rendahnya tingkat konversi, meningkatnya biaya operasional, serta menurunnya efektivitas kerja tim. Untuk mengatasi tantangan tersebut, diperlukan sebuah sistem *predictive lead scoring* yang merupakan pendekatan berbasis *machine learning* yang secara otomatis mengevaluasi kualitas prospek dengan menganalisis pola perilaku dan data historis secara komprehensif (Wu et al., 2024). Berbeda dengan metode konvensional tradisional yang bersifat statis dan subjektif, *predictive lead scoring* mampu menghasilkan probabilitas konversi yang lebih akurat melalui algoritma prediktif. Sehingga, memungkinkan perusahaan untuk memprioritaskan prospek dengan potensi tinggi.

Seiring dengan kebutuhan tersebut, penerapan teknologi machine learning menjadi semakin relevan dalam meningkatkan kualitas prediksi dan pengambilan keputusan. Algoritma Random Forest dipilih karena mampu menghasilkan performa klasifikasi yang baik, tahan terhadap *overfitting* melalui mekanisme bagging, serta mampu mengolah data nasabah yang terdiri dari berbagai atribut numerik dan kategorikal (Sindhu, 2025).

Untuk mencapai performa model yang optimal, penelitian ini menerapkan teknik optimasi pada tahap *preprocessing* data. Penggunaan *robust scaler* dilakukan untuk menstabilkan skala data yang memiliki variasi nilai ekstrem sehingga tidak memengaruhi proses pembelajaran model. Tahapan *preprocessing* yang tepat diketahui memiliki peran penting dalam meningkatkan kinerja model dalam menghasilkan prediksi Nugraha et al. (2024). Optimasi model *random forest* mampu menghasilkan nilai error yang lebih kecil secara konsisten dan mempercepat waktu pelatihan model dibandingkan hanya menggunakan metode tunggal (Susilo et al., 2025).

Berdasarkan permasalahan tersebut, penelitian ini mengembangkan *System Predictive Lead Scoring* (PROSPECTA) yang memanfaatkan model *machine learning* menggunakan algoritma Random Forest yang telah dioptimasi melalui tahapan *preprocessing* data. Hasil prediksi dari model tersebut kemudian disajikan dalam bentuk *dashboard* berbasis web menggunakan Next.js sebagai alat bantu visualisasi untuk mendukung proses prioritas prospek nasabah deposito berjangka. Dengan adanya sistem ini, diharapkan tim sales perbankan dapat lebih efektif dalam melakukan prioritas prospek, meningkatkan efisiensi pemasaran, serta mendukung pengambilan keputusan berbasis data di sektor perbankan.

1.2 Perumusan Masalah

Berdasarkan latar belakang diatas, maka dapat dirumuskan masalah “Bagaimana merancang dan membangun *System Predictive Lead Scoring* (PROSPECTA) berbasis Next.js untuk membantu menentukan prioritas prospek nasabah deposito berjangka?”

1.3 Batasan Masalah

Berdasarkan rumusan masalah, batasan masalah diperlukan sebagai aturan untuk memfokuskan pembahasan pada tujuan awal dan mencegah menyimpang. Beberapa batasan tersebut meliputi:

A. Lingkup Sistem

1. Sistem yang dikembangkan berupa *dashboard* yang digunakan untuk membantu proses prioritisasi prospek oleh tim sales perbankan
2. Sistem hanya berfokus pada produk deposito berjangka (*term deposit*) sebagai objek prediksi
3. Sistem tidak mencakup proses transaksi perbankan secara langsung, seperti pembukaan rekening atau eksekusi penjualan produk.
4. *Dashboard* ini hanya digunakan untuk menampilkan hasil prediksi dan analisis dari model *machine learning*, bukan sebagai sistem inti operasional perbankan (*core banking system*)

B. Data dan Model *Machine Learning*

1. Data yang digunakan dalam penelitian ini adalah Bank Marketing Dataset yang diperoleh dari UCI *Machine Learning Repository*.
2. Model *Machine Learning* yang digunakan adalah pendekatan algoritma *random forest* untuk menghasilkan prediksi probabilitas prospek nasabah.
3. Model *Machine Learning* dioptimasi melalui tahapan *preprocessing* data menggunakan robust scaler sebelum dilakukan proses klasifikasi
4. Model *machine learning* tidak melakukan pelatihan ulang model secara dinamis, melainkan menggunakan model yang telah dilatih sebelumnya.
5. Sistem ini tidak menyediakan fitur penambahan data nasabah baru untuk proses pelatihan ulang model secara langsung.
6. Evaluasi model dilakukan menggunakan metrik *accuracy* mengukur tingkat ketepatan prediksi model, tanpa analisis dampak bisnis secara langsung.

C. Fungsi Utama Sistem

1. Sistem menampilkan hasil *Predictive Lead Scoring* dalam bentuk *dashboard*.

2. Sistem menyediakan fitur visualisasi data hasil prediksi, termasuk segmentasi prospek berdasarkan tingkat potensi
3. Sistem menyediakan fitur pendukung tindak lanjut non-transaksional, seperti pengarahannya panggilan telepon dan pengiriman email kepada nasabah sebagai bagian dari sistem alur kerja tim sales

D. Teknologi yang Digunakan

1. *Frontend* menggunakan Next.js, React.js, *TypeScript*, dan *Tailwind CSS*
2. *Backend* dikembangkan menggunakan Node.js, dan Express.js
3. Basis data yang digunakan adalah MongoDB
4. Model *machine learning* dikembangkan menggunakan algoritma *random forest* yang dioptimasi melalui tahapan *preprocessing* data, kemudian diintegrasikan ke dalam sistem untuk menampilkan hasil prediksi.

1.4 Tujuan dan Manfaat Penelitian

A. Tujuan Penelitian

Membangun *System Predictive Lead Scoring* (PROSPECTA) berbasis Next.js sebagai alat bantu untuk menampilkan hasil prediksi dalam bentuk *dashboard* serta membantu tim sales dalam menentukan prioritas prospek nasabah deposito berjangka secara lebih efektif dan berbasis data.

B. Manfaat Penelitian

1. Bagi Mahasiswa (Penulis)
 - a. Menambah pengalaman dan ketrampilan dalam merancang serta mengembangkan *dashboard*.
 - b. Meningkatkan pemahaman dalam implementasi teknologi *machine learning*, khususnya pada penerapan sistem *predictive lead scoring*.
 - c. Menjadi sarana penerapan ilmu pengetahuan yang diperoleh selama perkuliahan ke dalam bentuk karya nyata.
2. Bagi Tim Sales Perbankan
 - a. Membantu tim sales perbankan meningkatkan efisiensi proses pemasaran.
 - b. Sistem ini memungkinkan akses *dashboard* yang cepat, stabil, dan responsif sehingga mendukung proses pengambilan keputusan berbasis data.

- c. Membantu tim sales dalam memprioritaskan prospek nasabah dengan potensi tinggi berdasarkan hasil prediksi yang dihasilkan sistem.
 - d. Mengurangi waktu dan tenaga yang terbuang untuk menghubungi prospek dengan peluang rendah, sehingga meningkatkan efektivitas kerja tim
3. Bagi Universitas
- a. Menjadi kontribusi nyata dalam bentuk penelitian mahasiswa yang dapat diaplikasikan di Masyarakat
 - b. Memperkuat peran universitas dalam mendukung pengembangan teknologi yang relevan dengan kebutuhan industri dan Masyarakat
 - c. Menambah koleksi penelitian yang dapat dijadikan referensi bagi mahasiswa lain pada topik serupa.

1.5 Metodologi Penelitian

1. Observasi

Observasi dalam penelitian ini dilakukan dengan cara mengumpulkan dan menganalisis data yang digunakan sebagai dasar dalam pengembangan model *machine learning*. Data yang digunakan adalah Bank Marketing Dataset yang diperoleh dari *UCI Machine Learning Repository*.

Pada tahap ini, peneliti melakukan eksplorasi data (*Exploratory Data Analysis*) untuk memahami karakteristik dataset, seperti distribusi data, serta hubungan antar variabel. Hasil dari proses observasi ini digunakan sebagai dasar dalam proses pengolahan data dan pembangunan model *predictive lead scoring*.

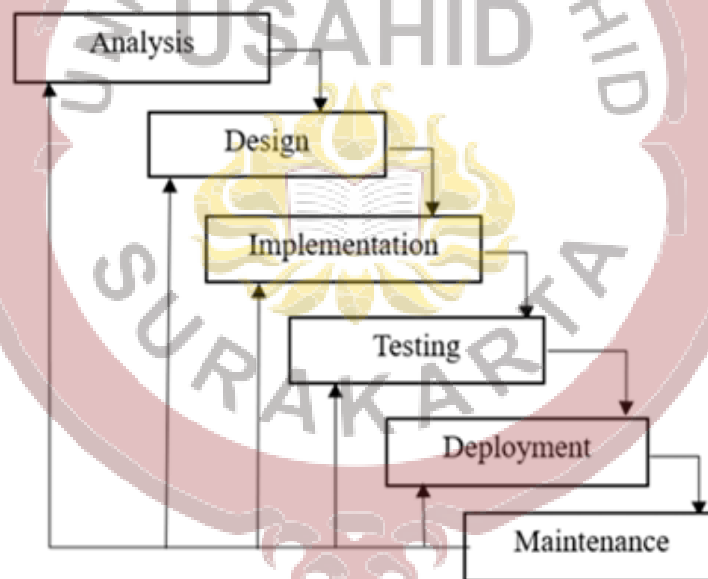
2. Studi Literatur

Studi literatur dilakukan dengan cara mengumpulkan dan mempelajari berbagai sumber referensi seperti jurnal ilmiah, buku, dan penelitian terdahulu yang relevan dengan topik penelitian. Pada tahap ini, peneliti mempelajari konsep-konsep yang berkaitan dengan *machine learning*, *random forest*, *preprocessing data*, *predictive lead scoring*, serta teknologi yang digunakan dalam pengembangan sistem seperti *Next.js* dan *dashboard*. Studi literatur ini bertujuan untuk memperkuat landasan teori serta menjadi acuan dalam perancangan dan pengembangan sistem.

3. Metode Pengembangan Sistem

Metode yang digunakan dalam pengembangan sistem pada penelitian ini adalah *Software Development Life Cycle (SDLC)* dengan pendekatan *Waterfall*. Metode *Waterfall* merupakan salah satu pendekatan pengembangan perangkat lunak yang dilakukan secara bertahap dan sistematis, dimulai dari tahap analisis kebutuhan hingga tahap pemeliharaan sistem. Tahapan dalam metode *Waterfall* meliputi analisis kebutuhan, perancangan sistem, implementasi, pengujian, deployment, dan pemeliharaan (Nurseptaji, 2021)

Metode ini dipilih karena memiliki alur kerja yang terstruktur sehingga memudahkan proses pengembangan sistem secara terencana dan sesuai dengan kebutuhan yang telah ditentukan. Ilustrasi metode *Waterfall* ditunjukkan pada Gambar 1.1.



Gambar 1. 1 Metode Pengembangan Waterfall

(Nurseptaji, 2021)

1. Analisis Kebutuhan Sistem

Tahap analisis kebutuhan dilakukan berdasarkan permasalahan pada proses pemasaran produk deposito berjangka di sektor perbankan. Kebutuhan sistem dibagi menjadi kebutuhan fungsional dan non-fungsional.

Kebutuhan fungsional meliputi pengelolaan akun pengguna dengan dua peran utama, yaitu admin dan sales, penampilan *dashboard* analitik berupa visualisasi

data prospek, penyajian daftar nasabah berdasarkan tingkat probabilitas hasil prediksi, fitur pencarian dan filter data, serta pengelolaan status prospek.

Kebutuhan non-fungsional mencakup aspek keamanan melalui autentikasi pengguna, performa sistem yang stabil, tampilan antarmuka yang responsif, serta kemudahan akses melalui berbagai perangkat tanpa instalasi tambahan.

2. Desain Sistem

Tahap desain dilakukan untuk merancang arsitektur sistem serta antarmuka pengguna sesuai dengan kebutuhan yang telah dianalisis. Perancangan antarmuka dilakukan menggunakan Figma dengan fokus pada kemudahan penggunaan (*user-friendly*), konsistensi tampilan, serta penyajian visualisasi data yang informatif.

Selain itu, desain sistem dimodelkan menggunakan *Unified Modeling Language (UML)* yang meliputi *use case diagram*, *activity diagram*, *sequence diagram*, dan *class diagram*, *deployment diagram* dan *component diagram* untuk menggambarkan alur sistem serta interaksi antar komponen.

3. Implementasi

Tahap implementasi dilakukan dengan mengembangkan sistem sesuai dengan desain yang telah dibuat. Pada sisi *frontend*, aplikasi dibangun menggunakan Next.js dan React.js dengan TypeScript serta Tailwind CSS untuk menghasilkan antarmuka yang responsif dan interaktif.

Pada sisi *backend*, sistem dikembangkan menggunakan Express.js dengan MongoDB sebagai basis data untuk menyimpan data pengguna, data nasabah, serta hasil prediksi.

Model *Machine Learning* pada penelitian ini dikembangkan menggunakan algoritma *random forest* yang telah melalui tahap *preprocessing* menggunakan *robust scaler*. Model tersebut digunakan untuk menghasilkan probabilitas calon nasabah dalam berlangganan deposito berjangka. Hasil prediksi dari model kemudian disajikan melalui *dashboard* tanpa dilakukan proses pelatihan ulang (*retraining*) secara langsung di dalam sistem.

4. Pengujian

Tahap pengujian dilakukan untuk memastikan sistem berjalan sesuai dengan kebutuhan yang telah ditentukan. Pengujian sistem menggunakan metode *black box*

testing yang berfokus pada fungsi tanpa melihat kode program. Pengujian meliputi autentikasi pengguna, peran admin dan sales, pengelolaan data prospek, serta tampilan *dashboard*. Selain itu, dilakukan pengujian API menggunakan Postman untuk memastikan komunikasi *frontend* dan *backend* berjalan dengan baik.

Selain pengujian sistem, dilakukan evaluasi terhadap model *machine learning* yang digunakan dalam sistem *predictive lead scoring*. Pengujian model dilakukan menggunakan metrik *accuracy* untuk mengukur performa model dalam memprediksi kemungkinan nasabah berlangganan deposito berjangka.

Hasil pengujian tersebut digunakan untuk memastikan bahwa sistem dan model yang dikembangkan memiliki performa yang baik dan dapat digunakan sebagai alat bantu dalam menentukan prioritas prospek nasabah.

5. *Deployment*

Tahap *deployment* dilakukan untuk mempublikasikan sistem agar dapat diakses secara online. Proses *deployment* sistem dilakukan dengan menggunakan platform Railway untuk *backend* dan Vercel untuk *frontend*. Pemilihan platform ini dilakukan karena mendukung pengembangan aplikasi web modern dan memudahkan proses integrasi antara *frontend* dan *backend*.

6. *Maintenance* (Pemeliharaan)

Tahap pemeliharaan dilakukan setelah sistem berhasil melalui proses *deployment*. Kegiatan pada tahap ini meliputi perbaikan *bug*, peningkatan performa sistem, serta penyempurnaan antarmuka berdasarkan hasil evaluasi penggunaan. Tahap ini juga mencakup *monitoring* sistem untuk memastikan aplikasi tetap berjalan dengan stabil dan dapat digunakan sebagai prototipe dalam mendukung proses prioritas prospek oleh tim sales perbankan.

1.6 Sistematika Penulisan

Untuk memudahkan pemahaman dan penyusunan laporan ini, sistematika penulisan disusun sebagai berikut:

BAB I PENDAHULUAN

Bab ini berisi latar belakang masalah yang mendasari penelitian, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metode penelitian yang digunakan, serta sistematika penulisan tugas akhir.

BAB II LANDASAN TEORI

Bab ini menguraikan tinjauan pustaka dan landasan teori yang mendukung penelitian, meliputi kerangka pemikiran, konsep *predictive lead scoring*, *preprocessing data* menggunakan *robust scaler*, algoritma *random forest*, *dashboard* serta teknologi yang digunakan dalam pengembangan sistem seperti *Unified Modelling Language* (UML), Next.js, React.js, Express.js, MongoDB, dan teknologi pendukung lainnya.

BAB III ANALISIS DAN PERANCANGAN SISTEM

Bab ini membahas analisis kebutuhan sistem yang meliputi kebutuhan fungsional dan non-fungsional, analisis peran pengguna (admin dan sales), serta perancangan sistem menggunakan *Unified Modeling Language* (UML). Selain itu, bab ini juga menjelaskan perancangan arsitektur sistem, desain basis data, dan perancangan antarmuka pengguna (*user interface*) *dashboard* PROSPECTA.

BAB IV IMPLEMENTASI SISTEM DAN ANALISIS HASIL

Bab ini menjelaskan proses implementasi *System Predictive Lead Scoring* (PROSPECTA) berdasarkan hasil perancangan yang telah dilakukan. Implementasi sistem meliputi pengembangan *frontend*, *backend*, integrasi model *machine learning* menggunakan algoritma *random forest*, serta penyajian *dashboard*. Selain itu, pada bab ini juga dilakukan analisis terhadap hasil pengujian model menggunakan metrik *accuracy*, pengujian fungsional sistem menggunakan *Black Box Testing*, serta pengujian API menggunakan Postman untuk memastikan sistem dapat berjalan dengan baik.

BAB V SIMPULAN DAN SARAN

Bab ini memuat kesimpulan yang diperoleh dari hasil penelitian dan pengembangan sistem *dashboard* PROSPECTA, serta saran untuk pengembangan sistem di masa mendatang agar dapat diimplementasikan secara lebih luas dan optimal.

BAB II

LANDASAN TEORI

2.1 Tinjauan Pustaka

Penelitian terkait penerapan *machine learning* dalam sektor perbankan terus berkembang dalam beberapa tahun terakhir, khususnya dalam mendukung pengambilan keputusan berbasis data. Salah satu penelitian menunjukkan bahwa algoritma *random forest* memiliki keunggulan dalam klasifikasi perilaku nasabah untuk memprediksi ketertarikan terhadap produk perbankan karena kemampuannya menangani data dengan kompleksitas besar secara stabil (Apriadi & Bisri, 2025). Dalam penelitian tersebut, model *machine learning* yang digunakan adalah Random Forest, tahap *preprocessing* data meliputi *handling missing values*, *one-hot encoding* untuk variabel kategorikal, serta *Min-Max normalization* untuk fitur numerik. Sementara itu, proses pengujiannya mengukur performa model secara komprehensif menggunakan metrik *accuracy*, *precision*, *recall*, *F1-score*, *ROC-AUC*.

Hal ini diperkuat oleh penelitian Jadli et al. (2023) yang menyatakan bahwa penggunaan random forest sangat efektif dalam memprediksi ketertarikan nasabah terhadap produk perbankan berkat ketahanannya terhadap pola data yang tidak beraturan. Riset ini menerapkan metode perancangan sistem berbasis eksperimental laboratorium menggunakan *tech stack* Python (Pandas, NumPy, dan Matplotlib), dengan fokus model *machine learning* pada Random Forest *Ensemble*. Tahap *preprocessing* data yang dilakukan meliputi eliminasi fitur dengan data hilang di atas 70%, pembersihan data menggunakan statistika deskriptif, dan transformasi variabel kategorikal menggunakan *One-Hot Encoding*. Selanjutnya, pengujian model dievaluasi secara mendalam menggunakan K-Fold Cross-Validation dengan metrik penilaian berupa tingkat akurasi (mencapai 93.04%), recall, presisi, F1-Score.

Penelitian terbaru juga sangat menekankan pentingnya tahap *preprocessing* data dalam meningkatkan performa akurasi model. Nugraha et al. (2024) dalam penelitiannya menunjukkan bahwa penskalaan fitur menggunakan metode Robust

Scaler sangat efektif untuk menormalisasi data sekaligus menekan dampak negatif dari data pencilan (*outlier* ekstrem). Riset tersebut berbasis *tech stack* Python dengan menguji dampak penskalaan terhadap algoritma klasifikasi. Melalui pendekatan statistik yang memanfaatkan parameter median dan jangkauan antarkuartil atau *Interquartile Range* (IQR), penggunaan *Robust Scaler* terbukti mampu mencegah bias akibat perbedaan skala antar-fitur sehingga menghasilkan performa model yang lebih optimal dan stabil.

Pada penelitian Nuraeni (2021) menunjukkan bahwa penggunaan teknik klasifikasi yang tepat dapat memberikan hasil yang lebih optimal dalam meminimalkan kesalahan penargetan potensi nasabah. Penelitian ini menggunakan metode perancangan eksperimen komparatif dengan *tech stack* RapidMiner. Model *machine learning* yang diterapkan dan dibandingkan meliputi Naive Bayes, Decision Tree, dan K-Nearest Neighbors (KNN). Tahap *preprocessing* data yang dilakukan mencakup konversi tipe data (nominal ke numerik) serta seleksi fitur berbasis *Information Gain* untuk mengoptimalkan atribut prediktor. Selanjutnya, proses pengujian kualitas model divalidasi menggunakan teknik X-Means dan *K-Fold Cross Validation* berdasarkan metrik akurasi serta kurva ROC.

Selain itu, pada penelitian Saputri et al. (2024) membuktikan bahwa penerapan *machine learning* pada shenda *florist* mampu menghasilkan estimasi kebutuhan yang akurat untuk periode mendatang, di mana hasil pengolahan data tersebut menjadi dasar yang krusial dalam proses pengambilan keputusan strategis guna meminimalkan risiko kerugian. Penelitian ini menggunakan metode perancangan sistem *prototyping* dengan *tech stack* utama berbasis PHP dan MySQL. Model yang diimplementasikan adalah metode peramalan *Holt-Winters Exponential Smoothing*, di mana tahap *preprocessing* datanya melibatkan agregasi data transaksi penjualan bulanan, serta diuji akurasinya menggunakan perhitungan *Mean Absolute Percentage Error* (MAPE).

Penelitian Deshmukh & Ansari (2026) pada dataset Bank UCI Marketing yang sama menegaskan bahwa pemanfaatan teknik *machine learning* mampu meningkatkan efektivitas dalam memprediksi respons nasabah terhadap produk deposito berjangka secara lebih efisien dibandingkan pendekatan konvensional.

Penelitian tersebut memanfaatkan *tech stack* Python dan Google Colab dengan menerapkan serta membandingkan performa model Logistic Regression, Random Forest, dan XGBoost. Proses *preprocessing* data dilakukan melalui tahapan penskalaan fitur menggunakan Standard Scaler serta penanganan ketidakseimbangan data (*imbalanced data*) secara khusus menggunakan metode SMOTE Augmentation. Sementara itu, tahap pengujian performa komparatifnya dievaluasi secara mendalam menggunakan metrik Akurasi, Presisi, Recall, F1-Score, dan nilai ROC-AUC.

Meskipun demikian, sebagian besar penelitian terdahulu masih berfokus pada pengujian performa algoritma *machine learning* secara terpisah dan belum banyak yang mengimplementasikan hasil prediksi ke dalam *dashboard* yang dapat digunakan langsung oleh pengguna. Penelitian ini mengembangkan sistem PROSPECTA yang memanfaatkan algoritma yang dioptimasi melalui tahapan *preprocessing* menggunakan *robust scaler*, serta menyajikan hasil prediksi dalam bentuk *dashboard* berbasis Next.js untuk mendukung proses pengambilan keputusan secara lebih praktis.

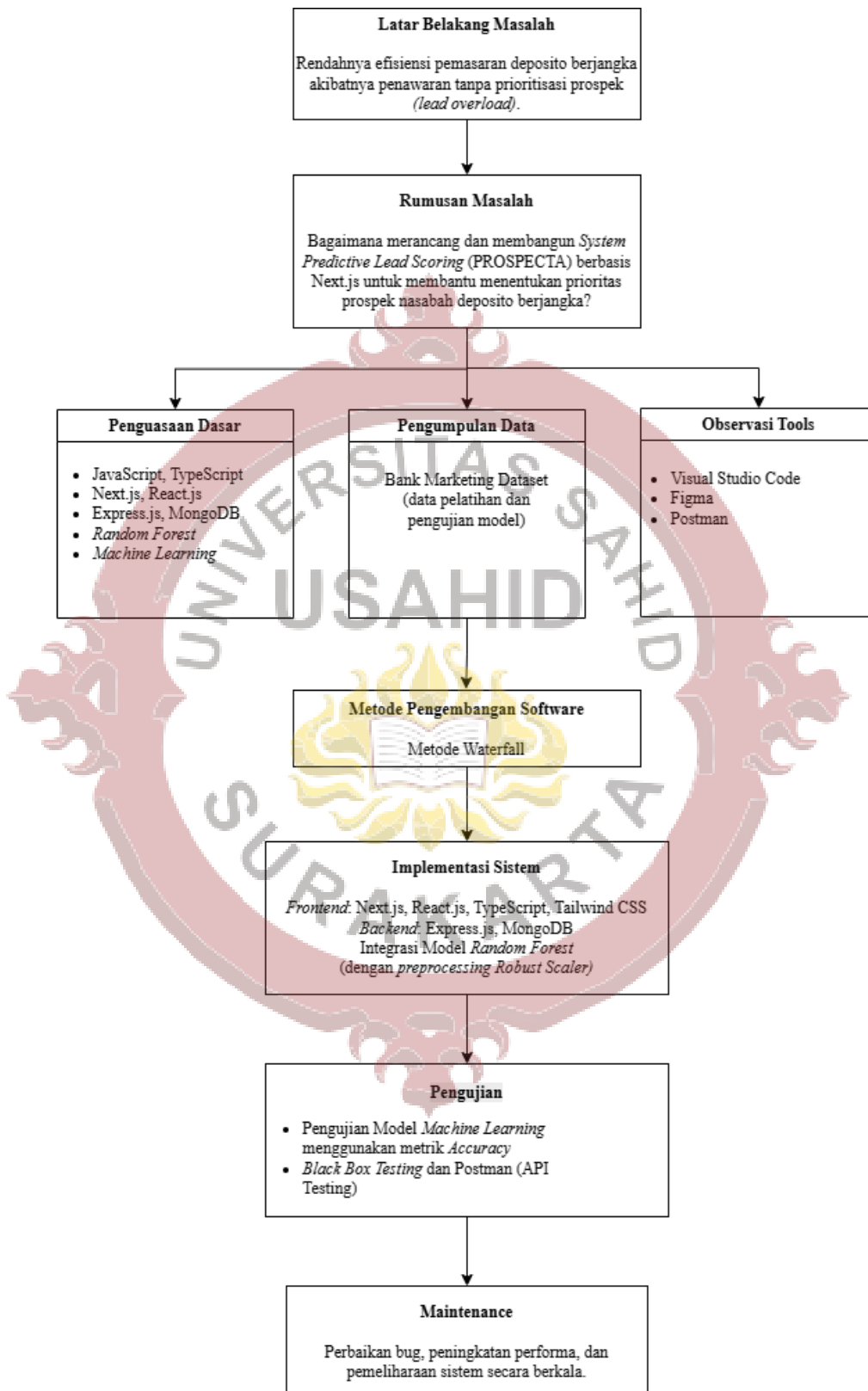
2.2 Kerangka Pemikiran

Kerangka pemikiran ini menggambarkan alur penelitian dalam pengembangan *dashboard* PROSPECTA sebagai *predictive lead scoring* berbasis *machine learning* pada portal perbankan. Penelitian diawali dari identifikasi permasalahan pada proses pemasaran produk deposito berjangka, dimana tim sales melakukan *outbound call* hanya berdasarkan dengan nomor urut pelanggan tanpa ada indikator prioritas sehingga dapat menyebabkan rendahnya efisiensi dan tingkat konversi

Berdasarkan permasalahan tersebut, dirumuskan kebutuhan akan sebuah sistem yang mampu memprediksi dan memprioritaskan prospek nasabah menggunakan pendekatan berbasis data. Proses penelitian dilanjutkan dengan penguasaan teknologi pendukung seperti JavaScript, TypeScript, Next.js, React.js, Express.js, MongoDB, serta pemahaman algoritma *machine learning random forest*. Data yang digunakan diperoleh dari Bank Marketing Dataset yang bersifat statis dan digunakan sebagai dasar pelatihan model prediktif.

Pengembangan sistem dilakukan menggunakan metode waterfall, yang meliputi tahap analisis kebutuhan, perancangan sistem, implementasi, pengujian, *deployment*, dan pemeliharaan. Sistem yang dibangun untuk mengintegrasikan model *machine learning random forest* ke dalam *backend* untuk menghasilkan probabilitas berlangganan deposito berjangka, kemudian disajikan melalui *dashboard* interaktif berbasis web untuk peran admin dan sales. Pengujian sistem dilakukan menggunakan metode *black box testing* dan *postman collection* untuk memastikan fungsionalitas sistem berjalan sesuai kebutuhan. Kerangka pemikiran ini diharapkan dapat memastikan setiap tahapan penelitian berjalan secara sistematis dan menghasilkan sistem yang sesuai dengan tujuan penelitian. Kerangka pemikiran penelitian ini dapat dilihat pada Gambar 2.1





Gambar 2. 1 Kerangka Pemikiran PROSPECTA

Kerangka pemikiran ini menggambarkan langkah-langkah sistematis dalam mengembangkan *dashboard*, khususnya *dashboard* PROSPECTA. Berikut adalah penjelasan singkat setiap langkah:

1. Latar Belakang Masalah

Tahap ini menjelaskan permasalahan utama berupa rendahnya efektivitas pemasaran deposito berjangka akibat proses penawaran yang masih dilakukan tanpa prioritas prospek. Kondisi *lead overload* menyebabkan tim sales kesulitan menentukan calon nasabah yang memiliki potensi tinggi untuk ditindaklanjuti.

2. Rumusan Masalah

Rumusan masalah pada penelitian ini adalah bagaimana merancang dan membangun *System Predictive Lead Scoring* (PROSPECTA) berbasis Next.js untuk membantu menentukan prioritas prospek nasabah deposito berjangka.

3. Penguasaan Dasar

Tahap ini meliputi pemahaman teknologi pengembangan sistem seperti JavaScript, TypeScript, Next.js, React.js, Express.js, MongoDB, *machine learning*, *preprocessing data* dan algoritma Random Forest.

4. Pengumpulan Data

Data diperoleh dari Bank Marketing Dataset yang digunakan sebagai data pelatihan dan pengujian model *machine learning*.

5. Observasi Tools

Penggunaan dan observasi tools pendukung pengembangan sistem seperti Visual Studio Code untuk pengkodean, Figma untuk desain antarmuka, dan Postman untuk pengujian API.

6. *Preprocessing Data*

Tahap *preprocessing* dilakukan menggunakan metode *robust scaler* untuk menstabilkan skala data dan mengurangi pengaruh ekstrem (*outlier*) sehingga data menjadi lebih optimal sebelum digunakan dalam proses pelatihan model *Machine Learning*.

7. Metode Pengembangan Software

Metode Waterfall digunakan sebagai pendekatan pengembangan sistem yang meliputi tahapan analisis, desain, implementasi, pengujian, *deployment*, dan pemeliharaan.

8. Implementasi Sistem

Sistem diimplementasikan menggunakan Next.js, React.js, TypeScript, dan Tailwind CSS pada sisi *frontend*, serta Express.js dan MongoDB pada sisi *backend*. Model *machine learning* menggunakan *random forest* yang telah melalui tahap *preprocessing* menggunakan *robust scaler* kemudian diintegrasikan untuk menampilkan probabilitas prospek nasabah.

9. Pengujian Model *Machine Learning*

Pengujian model dilakukan menggunakan metrik *accuracy* untuk mengukur tingkat ketepatan model *random forest* dalam memprediksi kemungkinan nasabah berlangganan deposito berjangka.

10. Pengujian Sistem

Pengujian dilakukan menggunakan *black box testing* dan *Postman Collection* untuk memastikan seluruh fitur sistem berjalan sesuai dengan kebutuhan yang telah ditetapkan.

11. Maintenance

Tahap pemeliharaan dilakukan untuk memperbaiki bug, meningkatkan performa sistem, serta memastikan sistem tetap berjalan stabil.

2.3 Landasan Teori

2.3.1 *Predictive Lead Scoring*

Predictive Lead Scoring adalah teknik *data-driven* dalam pemasaran dan penjualan yang menggunakan algoritma *machine learning* untuk memprediksi kemungkinan suatu prospek akan menjadi pelanggan berdasarkan data historis, perilaku, dan interaksi mereka. Dibandingkan pendekatan tradisional yang bersifat *rule-based*, model prediktif mampu menilai jutaan kombinasi fitur dan memberikan

skor probabilitas yang lebih akurat untuk prioritasasi prospek (González-Flores et al., 2025).

Di Indonesia, penerapan *Predictive Lead Scoring* telah diimplementasikan dalam berbagai sektor industri untuk mengatasi permasalahan *lead overload* dan mengoptimalkan efisiensi *customer relationship management*. Pada sektor pemasaran digital dan komersial, penelitian Sihite & Bunyamin (2023) membuktikan bahwa implementasi *Predictive Lead Scoring* menggunakan algoritma Random Forest mampu memprediksi prospek pelanggan secara akurat berdasarkan analisis durasi interaksi di situs website perusahaan.

Sementara itu, pada sektor keuangan dan pemasaran perbankan, pendekatan *Predictive Lead Scoring* berbasis algoritma *ensemble* dan *tree-based models* terbukti secara efektif mendeteksi prospek nasabah berpotensi tinggi untuk produk finansial, sekaligus menekan biaya operasional akibat kesalahan penargetan (Nuraeni, 2021; Deshmukh & Ansari, 2026).

2.3.2 *Unified Modeling Language (UML)*

Unified Modeling Language (UML) merupakan bahasa pemodelan standar yang digunakan untuk memvisualisasikan, menspesifikasikan, membangun, dan mendokumentasikan sistem perangkat lunak, khususnya sistem yang dikembangkan dengan pendekatan berorientasi objek. UML membantu pengembang dalam memahami struktur sistem, alur proses, serta interaksi antara pengguna dan sistem sebelum tahap implementasi dilakukan. Dengan menggunakan UML, perancangan sistem menjadi lebih terstruktur, mudah dipahami, dan dapat mengurangi kesalahan dalam pengembangan perangkat lunak (Sandfreni et al., 2021).

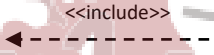
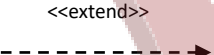
UML terdiri dari berbagai jenis diagram yang masing-masing memiliki fungsi dan tujuan berbeda dalam merepresentasikan sistem. Dalam penelitian ini, hanya akan digambarkan tiga diagram sebagai berikut:

a. *Use case Diagram*

Use Case Diagram adalah diagram UML yang menggambarkan interaksi fungsional antara aktor (pengguna atau sistem lain) dengan sistem secara keseluruhan. Diagram ini menunjukkan layanan atau fungsi yang tersedia dari sudut

pandang pengguna tanpa menjelaskan detail internal implementasinya. *Use Case Diagram* membantu mengidentifikasi kebutuhan fungsional sistem serta batasan interaksi antara peran-peran pengguna dengan fitur-fitur yang disediakan sistem (Ramdany et al., 2024).

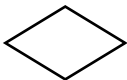
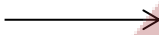
Tabel 2. 1 *Use Case Diagram*

Simbol	Nama	Keterangan
	Aktor	Aktor: Mewakili peran orang, sistem yang lain, atau alat ketika berkomunikasi
	Use case	Use Case: Abstraksi dan interaksi antara sistem dan aktor.
	Association	Association: Abstraksi dari penghubung antara aktor dengan use case.
	Include	Menunjukkan bahwa suatu use case seluruhnya merupakan fungsionalitas dari use case lainnya.
	Extend	Menunjukkan bahwa suatu use case seluruhnya merupakan fungsionalitas dari use case lainnya jika suatu kondisi terpenuhi.

b. Activity Diagram

Activity Diagram adalah diagram UML yang memodelkan alur kegiatan atau proses dalam sistem, dimulai dari suatu proses awal sampai akhir, termasuk keputusan dan pembagian alur kerja. Activity Diagram sangat bermanfaat untuk memetakan proses kerja sistem, misalnya alur pengambilan data nasabah, perhitungan probabilitas melalui model *Machine learning*, hingga penampilan hasil di dashboard.

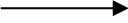
Tabel 2. 2 *Activity Diagram*

Simbol	Nama	Keterangan
	Status Awal	Sebuah diagram aktivitas memiliki sebuah status awal.
	Aktivitas	Aktivitas yang dilakukan sistem, aktivitasi biasanya diawali dengan kata kerja
	Percabangan/ Decision	Percabangan dimana ada pilihan aktivitas yang lebih dari satu.
	Penggabungan/ Join	Penggabungan dimana yang mana lebih dari satu aktivitas digabungkan jadi satu.
	Status Akhir	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
	Swimlane	Swimlane memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.

c. Sequence Diagram

Sequence diagram mempresentasikan interaksi sejumlah objek yang terjadi secara berurutan berdasarkan waktu. Diagram ini merupakan salah satu jenis pemodelan pada UML yang digunakan untuk memperlihatkan kolaborasi dinamis antar objek. Fungsi utamanya adalah menampilkan aliran pesan yang dikirimkan dari satu objek ke objek lain sekaligus menjelaskan interaksi yang terjadi di antara objek-objek tersebut.

Tabel 2. 3 *Sequence Diagram*

Simbol	Nama	Keterangan
	Entity Class	Gambaran sistem sebagai landasan dalam Menyusun basis data
	Message call	Menggambarkan alur message yang merupakan kejadian objek pengirim ke objek penerima.
	Message return	Menggambarkan alur pengembalian message ke objek pemanggil
	Life Line	Komponen yang digambarkan garis putus-putus dengan objek.
	Boundary Class	Menangani komunikasi antar lingkungan sistem

d. Class Diagram

Class Diagram digunakan untuk memodelkan struktur serta hubungan antar kelas dalam suatu sistem. Diagram ini menampilkan peran dan tanggung jawab tiap entitas yang berpengaruh terhadap perilaku sistem. Representasi class digambarkan dalam bentuk persegi panjang yang terdiri atas tiga bagian, yaitu nama kelas, atribut, dan metode. Kumpulan kelas beserta relasinya membentuk rancangan dasar yang jika diimplementasikan akan menghasilkan objek pada sistem. Class diagram menjadi bagian penting dalam perancangan berorientasi objek dan pemodelan alur data pada sistem yang sedang dikembangkan.

Tabel 2. 4 *Class Diagram*

Simbol	Nama	Keterangan
	Generalization	Hubungan dimana objek anak (descendent) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (ancestor).
	Nary Association	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
	Class	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
	Collaboration	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
	Realization	Operasi yang benar-benar dilakukan oleh suatu objek.
	Dependency	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (independent) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri.

2.3.3 Dashboard

Dashboard merupakan suatu media visualisasi informasi yang digunakan untuk menyajikan data secara ringkas, terstruktur, dan mudah dipahami dalam satu tampilan antarmuka. Dashboard dirancang untuk membantu pengguna dalam

memantau kondisi, menganalisis data, serta mendukung proses pengambilan keputusan secara cepat dan tepat (Mantik, 2021).

Dalam penelitian ini, dashboard digunakan sebagai media untuk menampilkan hasil prediksi dari sistem *Predictive Lead Scoring (PROSPECTA)* dalam bentuk visualisasi yang interaktif. Dashboard ini memungkinkan tim sales untuk melihat skor probabilitas prospek nasabah secara lebih mudah dan terstruktur, sehingga dapat membantu dalam menentukan prioritas prospek secara efektif dan berbasis data.

2.3.4 JavaScript dan TypeScript

JavaScript merupakan salah satu bahasa pemrograman web yang memiliki peran krusial dalam memperkaya tampilan serta meningkatkan fungsionalitas interaktif pada aplikasi berbasis web (Nusantara et al., 2023). Sebagai bahasa yang bersifat dinamis, JavaScript memungkinkan pengembang untuk membangun antarmuka yang responsif, namun fleksibilitasnya dalam pengelolaan tipe data seringkali menimbulkan tantangan pada pengembangan aplikasi skala besar.

TypeScript merupakan *superset* dari JavaScript yang menyediakan sistem pengetikan statis dan fitur modern untuk meningkatkan keandalan, keterbacaan, serta pemeliharaan kode dalam proyek besar seperti aplikasi PROSPECTA. Menurut Prayetno (2023), penggunaan *TypeScript* memungkinkan pengembangan aplikasi menjadi lebih terstruktur dan terorganisasi, baik pada sisi antarmuka pengguna (*frontend*) maupun integrasi dengan *backend*, sehingga mampu mendeteksi kesalahan sejak tahap kompilasi dan meminimalkan *bug* yang muncul saat *runtime*. Pendekatan ini memastikan pengembangan sistem tetap stabil dan mendukung kolaborasi pengembang yang lebih efektif dalam skala besar.

2.3.5 Next.js

Next.js merupakan *framework* React.js yang mendukung rendering sisi server (*Server-Side Rendering / SSR*), generasi situs statis (*Static Site Generation / SSG*), serta kemampuan pengoptimalan performa dan SEO yang lebih baik dibandingkan aplikasi *single-page-app* tradisional. Penggunaan Next.js membantu membangun antarmuka dashboard web yang cepat, responsif, dan *SEO-friendly*, yang penting dalam mengembangkan aplikasi portal seperti Dashboard

PROSPECTA. Dalam penelitian terkait Next.js yang tersedia pada jurnal teknis *Repeater Journal*, disebutkan bahwa penggunaan Next.js dalam pengembangan aplikasi web modern membantu meningkatkan performance, responsivitas, serta pengalaman pengguna (*user experience*) (Maharani, 2025).

2.3.6 Tech Stack Front-End dan Back-End

Pengembangan Dashboard PROSPECTA didukung oleh berbagai teknologi modern yang saling terintegrasi untuk membangun sistem yang responsif, *scalable*, dan mudah dikembangkan. Pemilihan teknologi ini bertujuan untuk mendukung proses pengolahan data prospek, visualisasi informasi, serta pengelolaan pengguna secara optimal.

a. Teknologi *Frontend*

Pada sisi *frontend*, sistem PROSPECTA dikembangkan menggunakan Next.js dan React.js sebagai dasar dalam membangun antarmuka pengguna yang interaktif dan modular. Next.js mendukung proses *server-side rendering* dan pengelolaan routing secara otomatis, sehingga mampu meningkatkan performa aplikasi serta pengalaman pengguna dalam mengakses dashboard secara real-time

Untuk meningkatkan kualitas dan keamanan kode program, digunakan TypeScript sebagai pengembangan dari JavaScript. TypeScript menyediakan fitur *static typing* yang membantu pengembang dalam mendeteksi kesalahan sejak tahap awal pengembangan, sehingga meminimalkan potensi bug dan meningkatkan *maintainability* sistem

Selain itu, tampilan antarmuka sistem dirancang menggunakan Tailwind CSS, yaitu framework CSS berbasis *utility-first* yang mempermudah pembuatan desain yang responsif, konsisten, dan modern. Penggunaan Tailwind CSS dapat meningkatkan efisiensi pengembangan antarmuka serta konsistensi desain aplikasi berbasis web

Dengan kombinasi teknologi tersebut, *frontend* PROSPECTA mampu menampilkan dashboard, grafik, dan informasi prospek secara optimal dengan tampilan yang *user-friendly*.

b. Teknologi *Backend*

Pada sisi *backend*, sistem PROSPECTA dikembangkan menggunakan Express.js yang berjalan di atas Node.js. Express.js berfungsi sebagai framework untuk membangun RESTful API yang menangani proses komunikasi antara *frontend* dan *database*.

Node.js digunakan sebagai *runtime environment* untuk menjalankan JavaScript di sisi server. Teknologi ini bersifat *event-driven* dan *non-blocking*, sehingga mampu menangani banyak permintaan pengguna secara bersamaan dengan efisien.

Untuk penyimpanan data, sistem menggunakan MongoDB sebagai basis data NoSQL berbasis dokumen. MongoDB menyimpan data dalam format JSON yang fleksibel, sehingga memudahkan pengelolaan data pengguna, data nasabah, data leads, serta laporan penjualan. Struktur database yang fleksibel ini juga mendukung pengembangan sistem dalam jangka panjang.

Selain itu, dataset hasil pengolahan model *Machine Learning* yang berisi skor probabilitas dan prioritas prospek diintegrasikan ke dalam database *backend*, sehingga dapat langsung digunakan oleh sistem untuk proses visualisasi dan analisis data pada dashboard.

c. *Version Control dan Deployment*

Dalam proses pengembangan, sistem PROSPECTA menggunakan GitHub sebagai media penyimpanan source code dan sistem kontrol versi. GitHub memfasilitasi pengelolaan perubahan kode, kolaborasi tim, serta pencatatan riwayat pengembangan proyek.

Untuk proses deployment, *frontend* PROSPECTA diunggah menggunakan Vercel, yang mendukung proses *build* dan *deployment* otomatis setiap kali terjadi pembaruan kode. Sementara itu, *backend* Express.js dideploy menggunakan Railway, yang menyediakan layanan *cloud* untuk menjalankan aplikasi Node.js secara stabil dan terintegrasi dengan GitHub.

Penggunaan platform *deployment* ini memungkinkan sistem PROSPECTA dapat diakses secara online, mudah dipelihara, serta mendukung pengembangan berkelanjutan.

d. Pengujian Model *Machine Learning*

Pengujian model dilakukan menggunakan metrik *accuracy* untuk mengukur tingkat ketepatan model Random Forest dalam memprediksi kemungkinan nasabah berlangganan deposito berjangka.

e. Pengujian Sistem

Pengujian *backend* dilakukan menggunakan tools Postman untuk memastikan setiap *endpoint* API berfungsi sesuai dengan kebutuhan sistem. Pengujian meliputi proses autentikasi, pengambilan data dashboard, pengelolaan *leads*, pembuatan laporan, serta pembaruan data profil pengguna.

Pengujian *frontend* dilakukan secara fungsional dengan memastikan seluruh halaman dan fitur dapat diakses dengan baik oleh pengguna sesuai dengan peran masing-masing, baik sebagai admin maupun sales.

2.3.7 Algoritma Random Forest

Random Forest merupakan salah satu algoritma *ensemble learning* yang populer dalam *classification* dan *prediction* karena kemampuannya menggabungkan banyak *decision tree* untuk mengurangi *overfitting* dan meningkatkan akurasi prediksi. Algoritma ini mampu menangani dataset berskala besar dengan *features* campuran (numerik dan kategorikal) serta distribusi yang tidak seimbang (*imbalanced data*), yang umum terjadi pada data nasabah perbankan (Jadli et al., 2023).

Dalam proses klasifikasi, *Random Forest* menggunakan metode *majority voting*, yaitu memilih kelas yang paling sering muncul dari seluruh hasil prediksi pohon keputusan. Secara sistematis, prediksi akhir dapat dirumuskan pada Persamaan (1) (Modami et al., 2025)

$$\hat{y} = \text{mode}(h_1(x), h_2(x), \dots, h_B(x)) \quad (1)$$

Di mana $h_i(x)$ merupakan hasil prediksi dari pohon ke- i terhadap data input x , B adalah jumlah pohon yang digunakan dalam model. Dan Fungsi *mode* digunakan untuk menentukan kelas yang paling sering muncul sebagai hasil akhir prediksi

2.3.8 Preprocessing Data

Preprocessing data merupakan tahapan awal dalam *Machine Learning* yang bertujuan untuk mempersiapkan data mentah agar lebih optimal sebelum dilakukan proses pemodelan. Tahapan ini meliputi pembersihan data (*data cleaning*), transformasi, serta normalisasi atau penskalaan fitur untuk meningkatkan kualitas data sehingga model dapat mempelajari pola dengan lebih baik. Tahapan *preprocessing* yang tepat dan terstruktur diketahui memiliki peran yang sangat krusial dalam mendongkrak performa serta akurasi model, terutama dalam menangani variasi data yang kompleks, tidak konsisten, maupun mengandung nilai ekstrem (Nugraha et al., 2024). Dengan kondisi data yang telah diproses secara baik, model *Machine Learning* dapat melakukan proses pelatihan secara optimal serta menghasilkan prediksi yang lebih akurat dan stabil.

2.3.9 Robust Scaler

Menurut Nugraha et al. (2024) Robust Scaler merupakan teknik *preprocessing* yang digunakan untuk menormalkan rentang nilai data numerik dengan mengurangi pengaruh *outlier*. Robust Scaler bekerja menggunakan estimasi statistik yang jauh lebih tangguh, yaitu nilai median dan kuartil data. Secara matematis, proses transformasi nilai asli (X) menjadi nilai terskala (X_{scaled}) dirumuskan pada persamaan (2) (Rachmawati et al., 2024)

$$X_{scaled} = \frac{X - \text{median}(X)}{Q3 - Q1} \quad (2)$$

Di mana :

1. Q1 merupakan kuartil pertama
2. Q3 merupakan kuartil ketiga
3. dan IQR (*Interquartile Range*) merupakan hasil dari Q3-Q1.

2.3.10 MongoDB

MongoDB merupakan salah satu sistem manajemen basis data NoSQL yang menyimpan data dalam bentuk dokumen berformat BSON (Binary JSON). Berbeda dengan basis data relasional yang menggunakan tabel dan relasi, MongoDB menyimpan data dalam bentuk collection dan document sehingga memiliki struktur penyimpanan yang lebih fleksibel (Alfa Susetyo, 2023). Fleksibilitas tersebut

memungkinkan pengembang menyesuaikan struktur data sesuai kebutuhan aplikasi tanpa harus melakukan perubahan skema secara kompleks.

Pada penelitian ini, MongoDB digunakan sebagai media penyimpanan data nasabah, data pengguna, aktivitas, laporan, serta hasil prediksi *lead scoring*. Pemilihan MongoDB didasarkan pada kemampuannya dalam menyimpan data semi-terstruktur, kemudahan integrasi dengan backend berbasis Express.js, serta kemampuannya dalam menangani perubahan struktur data apabila diperlukan selama proses pengembangan sistem.

2.3.11 Pengujian Model (Accuracy)

Pengujian pada model *Machine Learning* dilakukan untuk mengetahui tingkat performa model dalam menghasilkan prediksi yang akurat. Evaluasi model dalam penelitian ini menggunakan metrik *Accuracy* untuk mengukur tingkat ketepatan prediksi secara keseluruhan. Mengingat data penelitian telah melalui tahap optimasi menggunakan Robust Scaler, metrik *accuracy* menjadi indikator yang kuat untuk melihat kemampuan model dalam mengenali pola nasabah secara konsisten. Semakin tinggi nilai *accuracy*, maka semakin baik kemampuan model dalam menghasilkan prediksi yang sesuai dengan kondisi sebenarnya (Nugroho H et al., 2025).

Pada penelitian ini, evaluasi model menggunakan metrik *accuracy* yang mengukur tingkat ketepatan prediksi model terhadap keseluruhan data. Secara matematis, *accuracy* dirumuskan pada persamaan (3) (Prameswari Natakusumah, 2022)

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (3)$$

Dimana :

1. TP (*True Positive*) merupakan jumlah data *positif* yang di prediksi benar
2. TN (*True Negative*) adalah jumlah data *negatif* yang di prediksi benar
3. FP (*False Positive*) adalah jumlah data *negatif* yang salah di prediksi sebagai *positif*

4. FN (*False Negative*) merupakan jumlah data *positif* yang di prediksi sebagai *negative*

2.3.12 Pengujian Blackbox

Selain pengujian model, sistem juga diuji menggunakan metode *black box testing*, yaitu metode pengujian yang berfokus pada fungsi sistem berdasarkan input dan output tanpa melihat struktur kode internal. Metode ini digunakan untuk memastikan bahwa setiap fitur dalam sistem dapat berjalan sesuai dengan kebutuhan yang telah ditentukan (Anwar, 2025).

2.3.13 Pengujian API Menggunakan Postman

Pengujian sistem juga dilakukan pada bagian *backend* menggunakan Postman Collection. Pengujian ini bertujuan untuk memastikan bahwa setiap *endpoint* API dapat berfungsi dengan baik serta mampu menangani pertukaran data antar sistem secara sesuai dengan kebutuhan yang telah ditentukan. Melalui pengujian ini, validasi terhadap respon API, struktur data, serta proses komunikasi antara *frontend* dan *backend* dapat dilakukan untuk memastikan sistem berjalan secara stabil dan konsisten (Ramadhan & Mulyati, 2024)