BAB 2
LANDASAN TEORI

Pada penelitian ini digunakan teori-teori dan metode yang mendasari penelitian, yaitu penjualan produk toko retail, machine learning, supervised learning, interquartile range, transformasi logaritmik, deployment dengan streamlit, algoritma random forest, dan metrik evaluasi yang digunakan dalam penelitian.

2.1 Penjualan Produk Toko Retail

Penjualan produk di toko retail merujuk pada proses jual beli barang yang dilakukan oleh pengecer (*retailer*) kepada konsumen. Toko retail bisa berupa supermarket, minimarket, atau toko-toko kecil lainnya yang menyediakan kebutuhan sehari-hari bagi konsumen. Proses penjualan di toko retail seringkali dipengaruhi oleh berbagai faktor, seperti musim, tren pasar, promosi, dan pola konsumsi konsumen yang bersifat dinamis dan fluktuatif.

Dalam manajemen retail, pengelolaan penjualan yang baik sangat bergantung pada kemampuan untuk memprediksi permintaan produk secara akurat. Ketepatan dalam memprediksi penjualan dapat membantu toko dalam mengelola inventaris, mengurangi risiko overstock atau understock, dan meningkatkan profitabilitas. Oleh karena itu, sistem prediksi yang akurat sangat diperlukan untuk mendukung keputusan dalam pengelolaan produk dan distribusi stok [22].

2.2 Machine Learning

Machine learning (pembelajaran mesin) adalah cabang dari kecerdasan buatan (AI) yang memungkinkan sistem untuk belajar dari data dan membuat keputusan atau prediksi tanpa perlu diprogram secara eksplisit. Pembelajaran mesin bekerja dengan memanfaatkan data historis untuk menemukan pola atau hubungan yang dapat digunakan untuk memprediksi hasil di masa depan.

Pada machine learning, ada dua jenis utama pendekatan, yaitu supervised learning dan unsupervised learning. Supervised learning adalah teknik di mana model dilatih menggunakan data yang sudah memiliki label (output yang diketahui), sedangkan unsupervised learning digunakan untuk menemukan pola atau struktur dalam data yang tidak memiliki label [23].

Dalam konteks prediksi penjualan, machine learning dapat digunakan untuk mempelajari hubungan antara berbagai faktor yang mempengaruhi penjualan produk, seperti harga, waktu, cuaca, dan faktor eksternal lainnya. Dengan menggunakan model yang tepat, prediksi penjualan dapat dilakukan untuk memaksimalkan efisiensi operasional dan mengurangi biaya.

2.3 Supervised Learning

Supervised learning adalah teknik pembelajaran mesin di mana model dilatih menggunakan dataset yang sudah memiliki label atau output yang diketahui. Data input digunakan untuk memprediksi output yang sesuai berdasarkan pola yang ditemukan selama proses pelatihan. Setelah model dilatih, ia dapat digunakan untuk memprediksi output pada data yang belum pernah dilihat sebelumnya.

Supervised learning terbagi menjadi dua jenis utama: regresi dan klasifikasi. Regresi digunakan ketika output yang diprediksi bersifat kontinu (seperti prediksi harga atau penjualan), sementara klasifikasi digunakan untuk memprediksi kategori atau kelas (misalnya, apakah suatu email spam atau tidak). Dalam konteks penelitian ini, algoritma yang digunakan untuk prediksi penjualan adalah regresi, di mana model berusaha untuk memprediksi nilai numerik yang berkelanjutan, yaitu jumlah penjualan produk [24].

2.4 Interquartile Range (IQR)

Interquartile Range (IQR) adalah metode statistik yang digunakan untuk mengidentifikasi nilai ekstrem atau *outlier* dalam suatu distribusi data. IQR berfungsi sebagai ukuran penyebaran yang tidak dipengaruhi oleh nilai-nilai ekstrem (non-parametrik), sehingga sangat sesuai untuk digunakan pada dataset yang tidak berdistribusi normal atau mengandung *skewness* tinggi.

Secara matematis, IQR dihitung sebagai selisih antara kuartil ketiga (Q_3) dan kuartil pertama (Q_1), Seperti pada Rumus 2.1 di bawah:

$$IQR = Q_3 - Q_1 \tag{2.1}$$

1. Q_1 (kuartil pertama): nilai di bawah 25% dari distribusi data.
2. Q_3 (kuartil ketiga): nilai di bawah 75% dari distribusi data.

Nilai-nilai yang berada jauh di bawah Q_1 atau di atas Q_3 dianggap sebagai *outlier*. Batas bawah dan batas atas ditentukan dengan rumus:

Rumus 2.4 menunjukkan cara perhitungan *Lower Bound*.

$$LowerBound = Q_1 - 1.5 \times IQR \tag{2.2}$$

Rumus 2.5 menunjukkan cara perhitungan *Upper Bound*.

$$UpperBound = Q_1 + 1.5 \times IQR \tag{2.3}$$

Nilai-nilai yang berada di luar rentang [Lower Bound, Upper Bound] akan diklasifikasikan sebagai *outlier*.

Metode IQR digunakan dalam penelitian ini karena:

1. Tidak bergantung pada distribusi data dan tahan terhadap *skewness* serta *noise*.
2. Lebih tahan terhadap pengaruh nilai ekstrem dibandingkan metode berbasis rata-rata dan standar deviasi.
3. Sederhana dan cepat diterapkan tanpa asumsi distribusi tertentu.
4. Cocok untuk proses *data cleaning* dalam machine learning, agar model tidak belajar dari pola yang menyimpang.

Outlier dapat menyebabkan model machine learning belajar pola yang tidak representatif dari mayoritas data. Hal ini dapat mengurangi akurasi dan kemampuan generalisasi model, khususnya pada algoritma yang sensitif terhadap skala nilai seperti regresi linear atau metode berbasis jarak seperti *K-Nearest Neighbors (KNN)*. Oleh karena itu, deteksi dan penanganan *outlier* merupakan langkah penting dalam tahapan *preprocessing*.

Tahapan umum dalam mendeteksi dan menangani outlier menggunakan IQR adalah sebagai berikut:

1. Hitung nilai kuartil pertama (Q_1) dan kuartil ketiga (Q_3).
2. Hitung nilai IQR dengan rumus $Q_3 - Q_1$.

3. Tentukan batas bawah dan batas atas:

   Rumus 2.4 menunjukkan cara perhitungan *Lower Bound*.

$$LowerBound = Q_1 - 1.5 \times IQR \tag{2.4}$$

   Rumus 2.5 menunjukkan cara perhitungan *Upper Bound*.

$$UpperBound = Q_1 + 1.5 \times IQR \tag{2.5}$$

4. Identifikasi data yang berada di luar rentang tersebut sebagai *outlier*.
5. Hapus atau tangani data *outlier* sesuai strategi pembersihan data.

Dengan pendekatan ini, data yang bersih dan bebas dari *outlier* diharapkan dapat meningkatkan kualitas pelatihan model dan akurasi prediksi. [25]

2.5 Transformasi Logaritmik

Transformasi logaritmik merupakan salah satu teknik prapemrosesan data yang digunakan untuk mengubah skala distribusi data yang tidak normal atau memiliki distribusi miring (skewed) menjadi lebih simetris. Teknik ini umum diaplikasikan pada tugas regresi dan pembelajaran mesin untuk mengurangi pengaruh nilai pencilan (outlier) serta menstabilkan varians antar fitur. Transformasi ini juga dapat membantu model mengenali pola dalam data yang semula sulit dipelajari karena skala nilai yang tidak proporsional.

Transformasi logaritmik dipilih dalam penelitian ini untuk mengatasi masalah skewness positif yang umum ditemukan pada data numerik seperti jumlah penjualan (units sold), harga, atau permintaan (demand). Data dengan skewness tinggi cenderung mengarahkan model ke hasil prediksi yang bias dan kurang akurat. Dengan menerapkan transformasi log, nilai-nilai ekstrem dikompresi agar berada dalam rentang yang lebih seimbang, sehingga distribusi menjadi lebih mendekati normal. Hal ini membantu model bekerja lebih efektif dalam menemukan hubungan antara fitur dan target.

Transformasi log juga sangat berguna ketika hubungan antara fitur dan target bersifat eksponensial. Dengan mengubahnya ke skala logaritmik, hubungan

tersebut menjadi lebih linear dan mudah dipelajari oleh algoritma seperti *Random Forest Regressor*.

Transformasi log dilakukan dengan rumus sebagai berikut:

Rumus 2.6 menunjukkan cara perhitungan *Transformasi Logaritmik*.

$$y' = \log(y+1) \tag{2.6}$$

Penambahan konstanta 1 dilakukan untuk menghindari kesalahan komputasi pada nilai nol atau negatif, karena $\log(0)$ tidak terdefinisi. Fungsi `np.log1p()` dari pustaka *NumPy* digunakan untuk menghitung nilai $\log(1+y)$ dengan stabil secara numerik.

Setelah proses pelatihan dan prediksi dilakukan dalam skala logaritmik, hasil prediksi harus dikembalikan ke skala aslinya menggunakan fungsi eksponensial:

Rumus 2.7 menunjukkan cara perhitungan *Transformasi Eksponensial*.

$$\hat{y} = \exp(y') - 1 \tag{2.7}$$

Transformasi balik ini dilakukan agar metrik evaluasi seperti MAE, RMSE, dan MAPE dapat dihitung pada skala data yang sesungguhnya.

Langkah-langkah penggunaan transformasi logaritmik dalam pengolahan data adalah sebagai berikut:

1. Identifikasi variabel target atau fitur yang memiliki distribusi tidak normal atau skewed.
2. Terapkan transformasi logaritmik menggunakan fungsi `log1p()` terhadap variabel tersebut.
3. Lakukan pelatihan model dengan data yang telah ditransformasi.
4. Lakukan prediksi pada data uji dalam skala logaritmik.
5. Kembalikan hasil prediksi ke skala asli menggunakan fungsi `expm1()` untuk interpretasi dan evaluasi model.

Transformasi log merupakan teknik yang sering digunakan dalam penelitian regresi yang melibatkan data berskala besar, seperti prediksi penjualan, konsumsi,

atau permintaan. Penggunaan transformasi ini membantu meningkatkan generalisasi model dan mengurangi ketergantungan model terhadap outlier. [26]

2.6 Deployment dengan Streamlit

Streamlit adalah sebuah framework open-source berbasis Python yang dirancang khusus untuk membangun aplikasi web interaktif dengan cepat dan sederhana, terutama untuk keperluan data science dan machine learning. Framework ini memungkinkan para pengembang untuk mengubah skrip Python biasa menjadi antarmuka web hanya dengan beberapa baris kode, tanpa memerlukan pengetahuan mendalam tentang pengembangan front-end seperti HTML, CSS, atau JavaScript [27].

Keunggulan utama Streamlit terletak pada kemampuannya untuk menampilkan visualisasi data, model machine learning, serta komponen input-output interaktif secara langsung dari kode Python. Hal ini menjadikannya sangat populer di kalangan praktisi data, peneliti, dan mahasiswa dalam membangun prototipe cepat maupun aplikasi akhir untuk demonstrasi model.

Beberapa fitur unggulan yang disediakan oleh Streamlit antara lain:

1. Komponen UI interaktif seperti `st.slider`, `st.selectbox`, `st.number_input`, dan `st.button` yang memudahkan pengguna dalam memberikan input ke aplikasi.
2. Dukungan penuh untuk visualisasi dari pustaka populer seperti Matplotlib, Plotly, Altair, dan Seaborn.
3. Integrasi langsung dengan model machine learning menggunakan Scikit-learn, TensorFlow, PyTorch, dan joblib.
4. Kemampuan untuk menampilkan hasil prediksi, grafik evaluasi model, serta interpretasi fitur secara real-time.

Selain Streamlit lokal, terdapat pula layanan bernama **Streamlit Cloud**, yaitu platform hosting berbasis web yang disediakan secara gratis oleh pengembang Streamlit. Layanan ini memungkinkan aplikasi Streamlit yang disimpan di repository GitHub untuk dijalankan langsung di cloud tanpa konfigurasi server tambahan. Dengan menghubungkan akun GitHub, pengguna dapat dengan mudah melakukan *deployment* aplikasi dan membagikannya melalui tautan URL publik [28].

Dalam penelitian ini, penggunaan Streamlit dipilih sebagai platform pengembangan aplikasi prediksi penjualan karena beberapa alasan berikut:

1. Streamlit memiliki kurva belajar yang rendah dan mudah digunakan, sehingga cocok untuk pengembangan aplikasi oleh peneliti atau mahasiswa dengan latar belakang data science.
2. Integrasi dengan Python dan pustaka machine learning sangat baik, sehingga memungkinkan penggunaan model langsung tanpa perlu konversi ke format lain.
3. Aplikasi dapat dijalankan secara lokal maupun dihosting secara online menggunakan Streamlit Cloud, memungkinkan akses dari berbagai perangkat dan lokasi.
4. Alternatif framework lain seperti Flask atau Django membutuhkan konfigurasi dan pengaturan front-end yang lebih kompleks, sedangkan Streamlit menyediakan antarmuka visual yang lebih simpel dari segi development.

Dengan menggunakan Streamlit, model prediksi yang telah dibangun dapat diubah menjadi aplikasi web yang interaktif, mudah digunakan, dan dapat diakses oleh pengguna non-teknis secara praktis. Hal ini sejalan dengan tujuan dari penelitian, yaitu mengimplementasikan model machine learning ke dalam bentuk sistem pendukung keputusan yang aplikatif dan bermanfaat secara nyata.

2.7 Random Forest

Random Forest adalah salah satu algoritma pembelajaran mesin berbasis ensemble learning yang populer dan banyak digunakan untuk tugas klasifikasi maupun regresi. Algoritma ini dikembangkan oleh Leo Breiman pada tahun 2001 sebagai pengembangan dari metode Bagging (Bootstrap Aggregating) dan Decision Tree. Keunggulan utama Random Forest terletak pada kemampuannya dalam mengurangi overfitting, menangani dataset besar, dan menghasilkan prediksi yang akurat meskipun terdapat data yang tidak lengkap, outlier, maupun noise.

Secara umum, Random Forest membangun banyak pohon keputusan (decision tree) selama proses pelatihan. Setiap pohon dilatih pada subset acak dari data pelatihan (dengan teknik bootstrap sampling), dan pada setiap node pohon, hanya subset acak dari fitur yang dipertimbangkan untuk pemisahan. Proses ini

menghasilkan variasi yang tinggi antar pohon, yang ketika digabungkan (dengan cara rata-rata untuk regresi, atau voting mayoritas untuk klasifikasi), menghasilkan model yang lebih stabil dan akurat.

Keunggulan metode ini berasal dari dua aspek utama:

1. Bootstrap Sampling (Bagging): teknik pengambilan sampel dengan pengembalian untuk membangun berbagai decision tree dari subset yang berbeda.

2. Feature Randomness: pada setiap split node, hanya sebagian dari fitur yang dipertimbangkan, sehingga memperkaya keanekaragaman antar pohon.

Alur kerja algoritma *Random Forest* untuk tugas regresi secara umum dapat dijelaskan sebagai berikut:

1. Input Data: Dataset pelatihan D terdiri dari pasangan data (X_i, y_i), dengan $i = 1, 2, \ldots, n$.

2. Bootstrap Sampling: Dari dataset pelatihan, diambil beberapa subset secara acak *(dengan pengembalian)* untuk membangun beberapa pohon keputusan (*decision tree*).

3. Pelatihan Pohon:

   (a) Setiap pohon dilatih menggunakan subset data hasil *bootstrap*.

   (b) Pada setiap node pohon, hanya subset acak dari fitur yang dipertimbangkan untuk proses split.

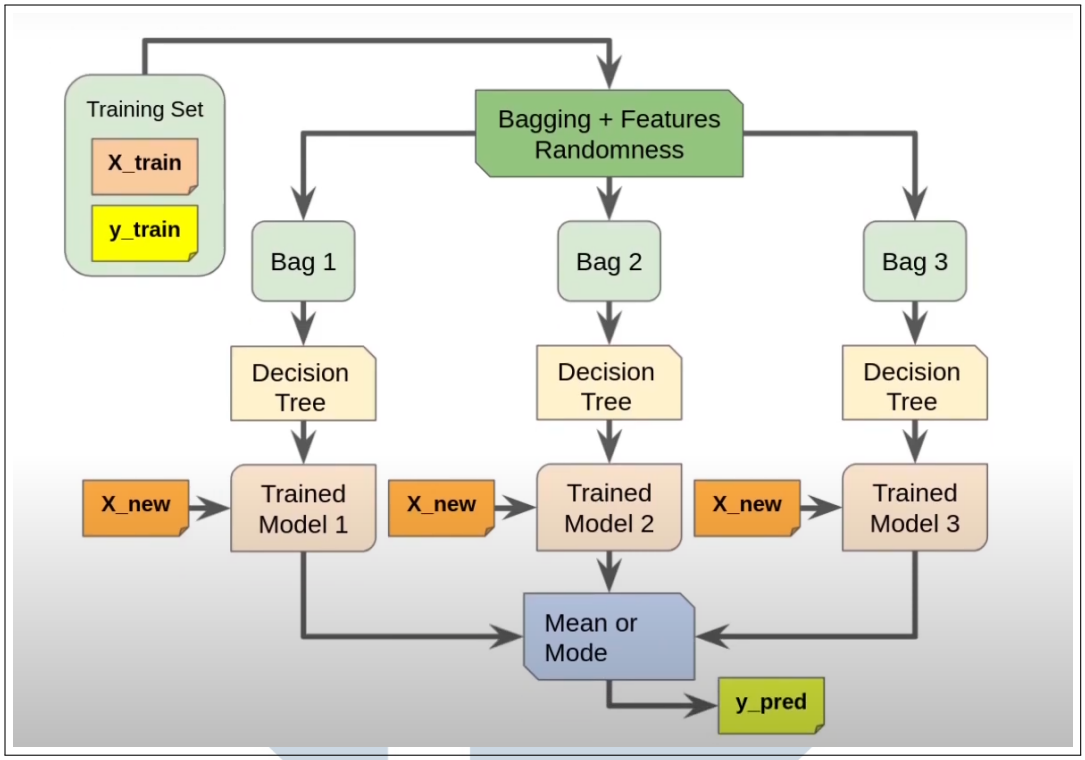
   (c) Split terbaik dipilih berdasarkan kriteria minimisasi *mean squared error* (MSE).

4. Prediksi:

   (a) Untuk data baru x, setiap pohon menghasilkan prediksi $\hat{y}_i$.

   (b) Prediksi akhir diambil sebagai rata-rata dari seluruh hasil prediksi pohon

Dengan mekanisme tersebut, *Random Forest* dapat mengurangi varians model dan mencegah *overfitting* yang umum terjadi pada *decision tree* tunggal. Lebih jelasnya lagi, alur dari cara random forest bekerja digambarkan pada Gambar 2.1 di bawah.



Gambar 2.1. Alur Kerja Random Forest

Beberapa alasan mengapa algoritma Random Forest dipilih dalam penelitian ini adalah:

1. Robust terhadap Overfitting: meskipun masing-masing pohon mungkin overfit, hasil gabungan cenderung lebih stabil.
2. Dapat Menangani Data yang Tidak Seimbang: tetap memberikan performa yang baik meskipun data mengandung outlier atau nilai hilang.
3. Mendukung Data Numerik dan Kategorik: fleksibel dalam berbagai tipe data.
4. Tidak Perlu Scaling: tidak memerlukan normalisasi seperti pada algoritma berbasis jarak.
5. Dapat Menangani Dataset Berdimensi Tinggi dengan banyak fitur.

Kinerja Random Forest sangat dipengaruhi oleh pengaturan hyperparameter. Beberapa hyperparameter utama yang digunakan dalam proyek ini dan fungsinya dijelaskan sebagai berikut:

1. n_estimators: jumlah pohon dalam hutan. Semakin banyak pohon, semakin stabil prediksi yang dihasilkan. Namun, peningkatan jumlah pohon juga berdampak pada waktu komputasi.

2. max_depth: kedalaman maksimum dari setiap pohon. Semakin dalam pohon, semakin kompleks model, tetapi risiko overfitting juga meningkat. Pembatasan max_depth membantu menjaga generalisasi model.

3. min_samples_leaf: jumlah minimum sampel yang harus ada di daun pohon. Meningkatkan nilai ini membuat pohon lebih konservatif, mengurangi overfitting.

4. min_samples_split: jumlah minimum sampel yang diperlukan untuk membagi node. Jika nilainya tinggi, pembelahan akan lebih jarang terjadi sehingga pohon menjadi lebih dangkal.

Pengaturan hyperparameter yang optimal sangat penting. Oleh karena itu, dalam penelitian ini digunakan metode RandomizedSearchCV untuk mencari kombinasi hyperparameter terbaik dengan efisiensi tinggi.

Random Forest merupakan algoritma regresi yang andal dengan berbagai keunggulan baik dari sisi akurasi maupun kestabilan prediksi. Algoritma ini cocok digunakan untuk kasus prediksi permintaan penjualan karena dapat menangani berbagai tipe fitur, mengatasi noise, serta memberikan hasil prediksi yang dapat dijelaskan dan ditafsirkan melalui evaluasi performa dan pentingnya fitur. [29].

2.7.1 Hyperparameter

Random Forest memiliki beberapa parameter penting yang mempengaruhi kinerja model. Berikut penjelasan dari beberapa parameter utama:

1. **n_estimators**: Jumlah pohon dalam hutan. Semakin banyak pohon, hasil prediksi lebih stabil, namun waktu komputasi bertambah.

2. **max_depth**: Kedalaman maksimum tiap pohon. Nilai terlalu besar berisiko overfitting.

3. **min_samples_split**: Jumlah minimum sampel yang dibutuhkan untuk memisahkan node.

4. **min_samples_leaf**: Jumlah minimum sampel pada daun pohon.

5. **max_features**: Jumlah fitur maksimum yang digunakan untuk mencari split terbaik pada setiap node.

2.7.2 Keunggulan dan kekurangan

Kelebihan Random Forest:

1. Mampu mengatasi overfitting lebih baik dibandingkan *decision tree* tunggal.
2. Dapat menangani data dengan dimensi tinggi tanpa seleksi fitur eksplisit.
3. Mampu menangkap hubungan non-linear antar variabel.
4. Memberikan informasi tentang pentingnya fitur (*feature importance*).

Tuning parameter ini dilakukan untuk mencapai keseimbangan antara bias dan varians, serta mengurangi risiko overfitting.
Kekurangan Random Forest:

1. Interpretasi model sulit karena hasil prediksi berasal dari banyak pohon.
2. Ukuran model besar dan waktu prediksi relatif lambat.
3. Tidak cocok untuk kebutuhan real-time jika jumlah pohon sangat besar.

Pada penerapan dalam kasus Prediksi Penjualan, Random Forest sangat berguna untuk memodelkan hubungan non-linear antara berbagai faktor seperti harga, diskon, harga pesaing, hingga promosi dengan jumlah penjualan. Model ini juga mampu menangkap interaksi antar fitur tanpa memerlukan spesifikasi eksplisit.

2.8 Metriks Evaluasi

Untuk mengevaluasi kinerja model prediksi, beberapa metrik yang sering digunakan dalam analisis regresi adalah Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), dan Mean Absolute Percentage Error (MAPE). Metrik ini digunakan untuk mengukur seberapa akurat model dalam memprediksi nilai yang sebenarnya [30]. Berikut penjelasan mengenai ketiga metrik tersebut:

2.8.1 Root Mean Squared Error (RMSE)

RMSE adalah metrik yang mengukur selisih antara nilai yang diprediksi dengan nilai yang sebenarnya. RMSE memberikan penekanan lebih pada kesalahan besar, karena selisih yang lebih besar akan dikuadratkan. Semakin kecil nilai RMSE, semakin baik performa model dalam memprediksi data yang sebenarnya. Formula RMSE adalah sebagai berikut:

Rumus 2.8 menunjukkan cara perhitungan *RMSE*.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{2.8}$$

Dimana:

1. y_i adalah nilai aktual,
2. $\hat{y}_i$ adalah nilai prediksi,
3. n adalah jumlah data.

2.8.2 Mean Absolute Error (MAE)

MAE mengukur rata-rata kesalahan absolut antara nilai yang diprediksi dengan nilai yang sebenarnya. MAE tidak memberikan bobot lebih pada kesalahan besar, sehingga dapat memberikan gambaran yang lebih adil tentang akurasi model. Formula MAE adalah sebagai berikut:

Rumus 2.9 menunjukkan cara perhitungan *MAE*.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{2.9}$$

Dimana:

1. y_i adalah nilai aktual,
2. $\hat{y}_i$ adalah nilai prediksi

2.8.3 Mean Absolute Percentage Error (MAPE)

MAPE mengukur seberapa besar kesalahan dalam bentuk persentase, yang memudahkan untuk menginterpretasikan hasil dalam konteks relatif. MAPE memberikan gambaran yang lebih jelas mengenai seberapa besar perbedaan antara nilai prediksi dan nilai aktual dalam bentuk persentase. Formula MAPE adalah sebagai berikut:

Rumus 2.10 menunjukkan cara perhitungan *MAPE*.

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{|y_i - \hat{y}_i|}{y_i}\right) \times 100 \tag{2.10}$$

Dimana:

1. y_i adalah nilai aktual,
2. $\hat{y}_i$ adalah nilai prediksi

Ketiga metrik ini digunakan untuk mengevaluasi kinerja model prediksi dalam penelitian ini, yang bertujuan untuk memprediksi penjualan produk pada toko retail.