BAB 2
LANDASAN TEORI

2.1 Emisi Karbon

Emisi karbon adalah pelepasan gas atau senyawa CO_2 ke udara sebagai hasil dari suatu aktivitas, diukur dalam ton karbon atau ton CO_2 dan aktivitas ini akan menciptakan jejak karbon dalam suatu lingkungan [26]. Jejak karbon adalah indikator dari total emisi CO_2 yang dihasilkan oleh aktivitas secara langsung ataupun tidak langsung. Jejak karbon juga mencakup akumulasi dari semua tindakan individu, populasi, pemerintah, perusahaan, organisasi, proses, dan sektor industri [27].

Sumber emisi karbon dapat dibagi menjadi 4 kategori yaitu [28] :

- Bergerak, yaitu meliputi kendaraan bermotor seperti mobil, sepeda motor, pesawat udara, dan kendaraan bermotor lainnya.
- Tidak bergerak, yaitu meliputi tempat seperti perumahan, desa, pasar, pabrik.
- Industri, yaitu meliputi pembakaran bahan bakar fosil untuk tenaga, reaksi kimia, metalurgi, penambangan, manufaktur.
- Pembuangan sampah, yaitu meliputi pembuangan sampah ataupun limbah.

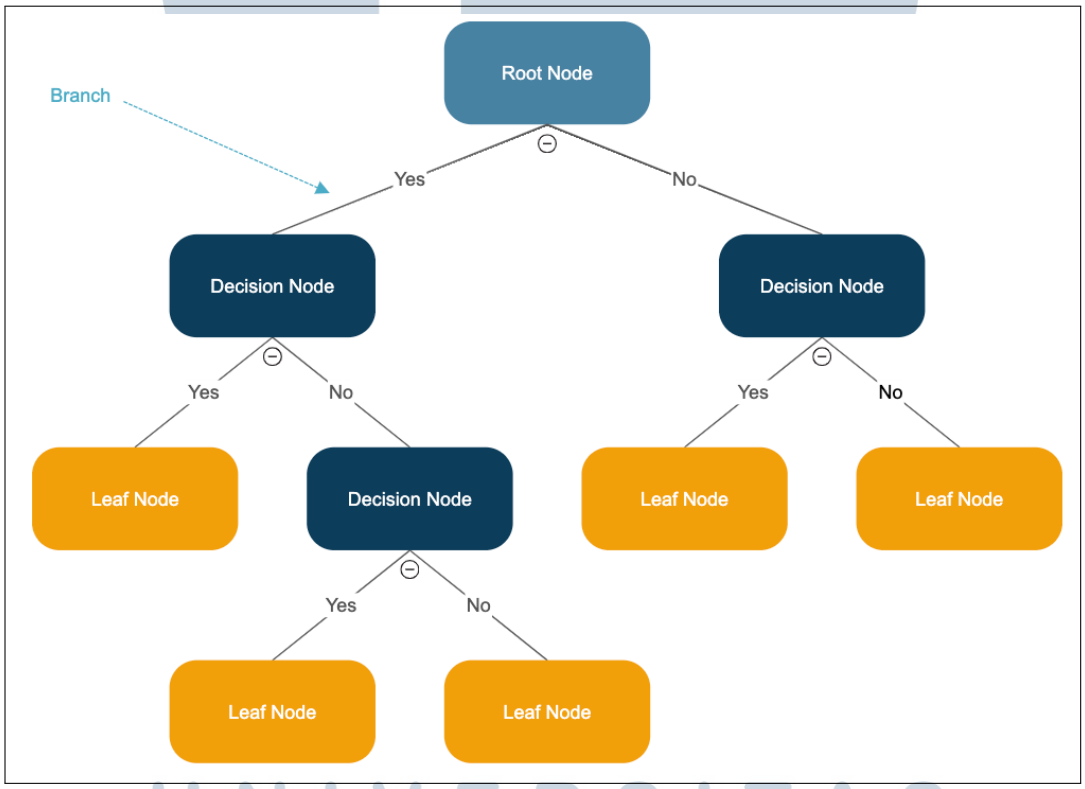
Dari keempat kategori tersebut, penelitian ini akan fokus pada kategori sumber tidak bergerak yaitu rumah tangga. Untuk mengetahui berapa banyak emisi Karbon yang dihasilkan oleh penggunaan listrik sebuah rumah tangga, mengutip dari hasil penelitian Hannah Ritchie (2023), Indonesia memiliki konversi rasio gram CO_2e sebesar 623.28 gram per KWh [29].

2.2 Decision Tree

Machine Learning adalah salah satu cabang ilmu yang mempelajari kapasitas atau kemampuan suatu mesin dalam mempelajari data untuk menyelesaikan suatu permasalahan atau tugas dengan membangun dan menggunakan sebuah model matematis [30]. Pada umumnya *Machine Learning* sering digunakan untuk melakukan klasifikasi untuk suatu permasalahan, hal ini disebabkan oleh kemampuan *Machine learning* dalam menganalisis,

mengidentifikasi, dan memprediksi suatu nilai dengan mempelajari data historis dan mengimplementasi suatu algoritma untuk membentuk *decision boundary* dalam sebuah *hyperplane* [12]. Salah satu algoritma tersebut adalah *Decision Tree*

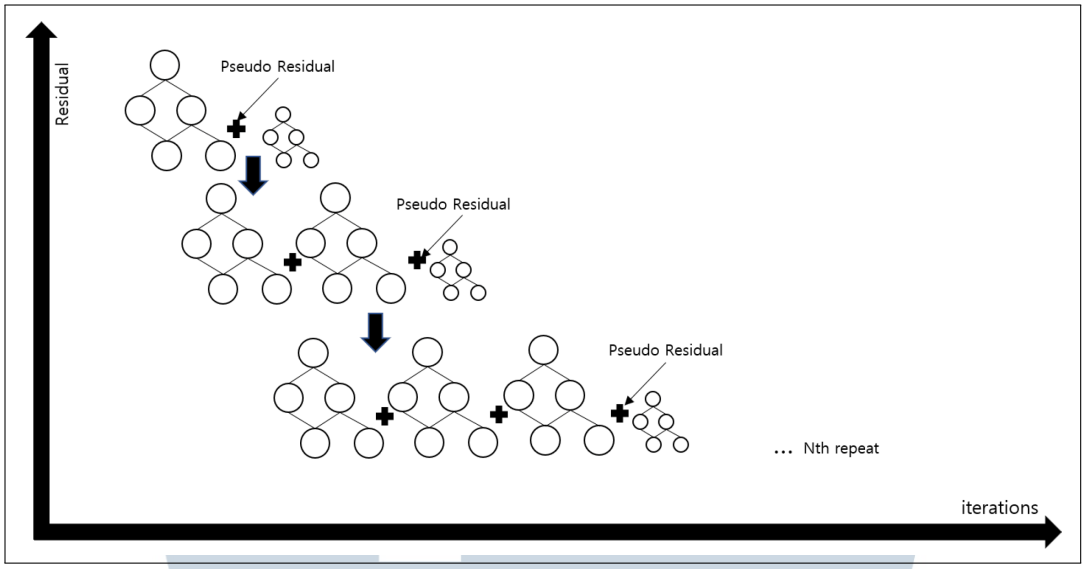
Decision tree adalah sebuah metode klasifikasi yang menggunakan prinsip sebuah *tree* yaitu sebuah kumpulan aturan dan kondisi untuk membentuk suatu keputusan [14]. Sebuah *decision tree* terdiri atas banyak node yang akan membentuk sebuah pohon, *node* yang tidak memiliki *incoming edges* tetapi memiliki *outgoing edges* disebut *root node*, lalu *node* yang memiliki *incoming dan outgoing edges* disebut *internal* atau *decision node*, lalu *node* yang hanya memiliki *incoming edges* disebut *leaf* atau *terminal node*. Berikut adalah ilustrasi struktur sebuah *decision tree*.



Gambar 2.1. struktur *decision tree* [1]

2.3 Gradient Boosting

Gradient Boosting adalah suatu algoritma yang menghasilkan dan menggabungkan banyak *decision tree* yang lemah atau disebut juga *weak learners*, untuk menjadi satu *decision tree* yang kuat atau *strong learners* [31]. Berbeda dengan algoritma *machine learning* pada umumnya yang hanya melatih satu model. Berikut adalah ilustrasi proses *boosting*.



Gambar 2.2. Ilustrasi proses *boosting* [2]

Berikut adalah rumus *Gradient Boosting*.

$$F_0(x) = arg\ \min_{\alpha} \sum_{i=1}^{N} L(y_i, \alpha) \tag{2.1}$$

$$r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=f_{n-1}(x)} \tag{2.2}$$

$$\gamma_n = arg\ \min_{\gamma} \sum_{i=1}^{N} L(y_i, F_{n-1}(x_i) + \gamma g_n(x_i)) \tag{2.3}$$

$$F_n(x) = F_{n-1}(x) + \upsilon \gamma_n g_n(x) \tag{2.4}$$

Keterangan:

- $F_0(x)$ adalah model dasar atau pertama
- r_{im} adalah residu dari *weak learners*
- g_n adalah model ke n dari proses *learning*
- γ_n adalah koefisien optimal untuk *update* model
- υ adalah *learning rate*
- F_n adalah model terakhir setelah *learning* sebanyak n

2.4 Extreme Gradient Boosting

Extreme Gradient Boosting atau *XGBoost* adalah algoritma yang memiliki fundamental sama dengan *Gradient Boosting* tetapi merupakan bentuk *Gradient Boosting* yang lebih ter-regularisasi [20]. Regularisasi adalah salah satu metode yang dapat diterapkan untuk mengurangi *overfitting* dari model regresi yang dibentuk berdasarkan data *training*. Hal ini dikarenakan model yang *overfitting* memiliki estimasi varians yang terlalu besar sehingga tidak konsisten dengan kasus lain yang berlaku pada praktik nyatanya. *Ridge Regularization* dapat dimanfaatkan untuk menghindari *overfitting* dengan menerapkan penalti kompleksitas pada fungsi akhir [21]. Berikut adalah metode regularisasi yang digunakan dalam *Extreme Gradient Boosting*.

- L1 Regularization

$$Loss = Error(Y - \widehat{Y}) + \lambda \sum_{1}^{n} |w_i| \tag{2.5}$$

  Keterangan:

  - λ adalah parameter atau nilai kekuatan regularisasi
  - $|w_i|$ adalah nilai mutlak koefisien model

- L2 Regularization

$$Loss = Error(Y - \widehat{Y}) + \lambda \sum_{1}^{n} w_i^2 \tag{2.6}$$

  Keterangan:

  - λ adalah parameter atau nilai kekuatan regularisasi
  - w_i^2 adalah nilai kuadrat koefisien model

Untuk melakukan suatu prediksi regresi, pada dasarnya proses perhitungan dalam *Extreme Gradient Boosting* dapat dibagi menjadi 6 bagian utama [3].

1. Kalkulasi *Residuals*. *Residuals* adalah suatu *metric* perbandingan antara nilai prediksi dengan nilai aktual dengan mengestimasi nilai *Mean Squared Error (MSE)* terendah dari suatu model regresi [32]. Berikut adalah rumus residual:

$$Residual = Nilai\ Aktual - Nilai\ prediksi \tag{2.7}$$

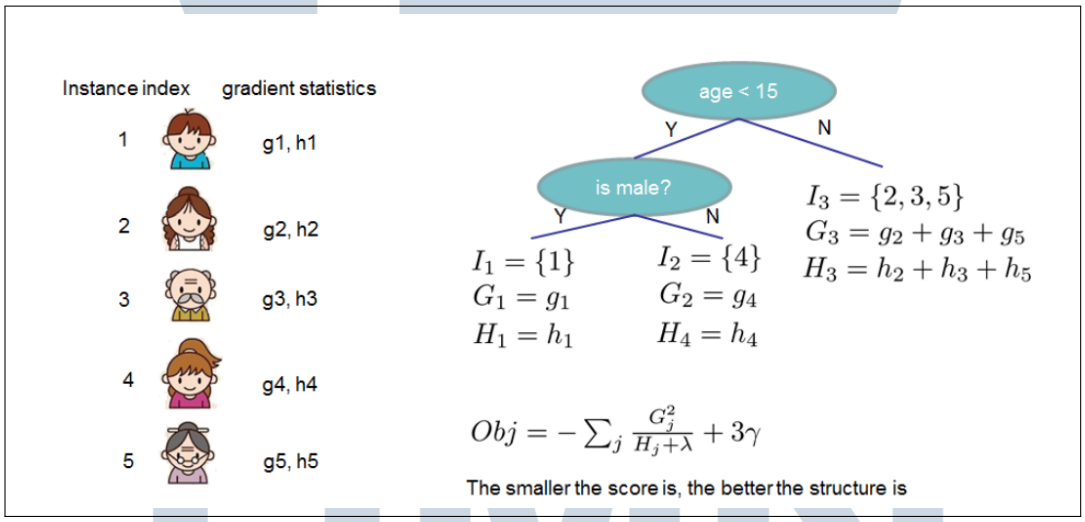
2. Penyusunan *XGBoost Tree*. Pada tahap ini, setelah mendapatkan nilai residual, dilakukannya perhitungan *similarity score* untuk daun pohon yang terbentuk. Berikut adalah rumus *similarity score* untuk daun pohon:

$$Similarity\ Score = \frac{(Jumlah\ Residual)^2}{(Banyak\ Residual + \lambda)} \tag{2.8}$$

Keterangan:

- λ adalah parameter atau nilai kekuatan regularisasi

Pembentukan daun pohon akan dilakukan dengan membagi nilai-nilai residual menjadi suatu daun dalam pohon berdasarkan nilai yang terdapat dalam *feature* tersebut. Berikut adalah ilustrasinya:



Gambar 2.3. Proses pembentukan daun pohon [3]

Berdasarkan ilustrasi diatas pembagian nilai-nilai residual dilakukan pada *feature* umur dan jenis kelamin. Setelah melakukan pembagian tersebut, akan dilakukan perhitungan *similarity score* untuk setiap daun baru yang terbentuk dan keputusan untuk menetapkan atau membuat daun baru yang terbentuk ini akan berdasar pada perhitungan nilai *gain*, jika nilai *gain* daun positif, maka pembagian/*split* akan disimpan dan jika negatif maka pembagian/*split* tidak akan disimpan. Berikut adalah rumus *gain*:

$$Gain = Score\ Daun\ Kiri + Score\ Daun\ Kanan - Similarity\ Score\ Root \tag{2.9}$$

3. *Tree Pruning*. Pada tahap ini akan dilakukan pemangkasan/*pruning* terhadap

pohon untuk melakukan kompresi terhadap pohon sehingga ukurannya lebih kecil dengan memangkas bagian dari pohon yang dinilai tidak penting atau bagian yang berulang. Penentuan bagian pohon yang akan dipangkas akan ditentukan dengan membandingkan nilai *gain* setiap daun (dari bawah keatas) dengan nilai *gamma* γ yang telah ditentukan. Jika nilai gain jatuh dibawah *gamma* γ maka daun tersebut akan dipangkas.

4. Kalkulasi hasil *output* dari daun. Pada tahap ini akan dilakukan perhitungan nilai output dari daun-daun pohon. Berikut adalah rumus *Output Value*:

$$Nilai\ Output = \frac{Jumlah\ Residual}{(Banyak\ Residual + \lambda)} \tag{2.10}$$

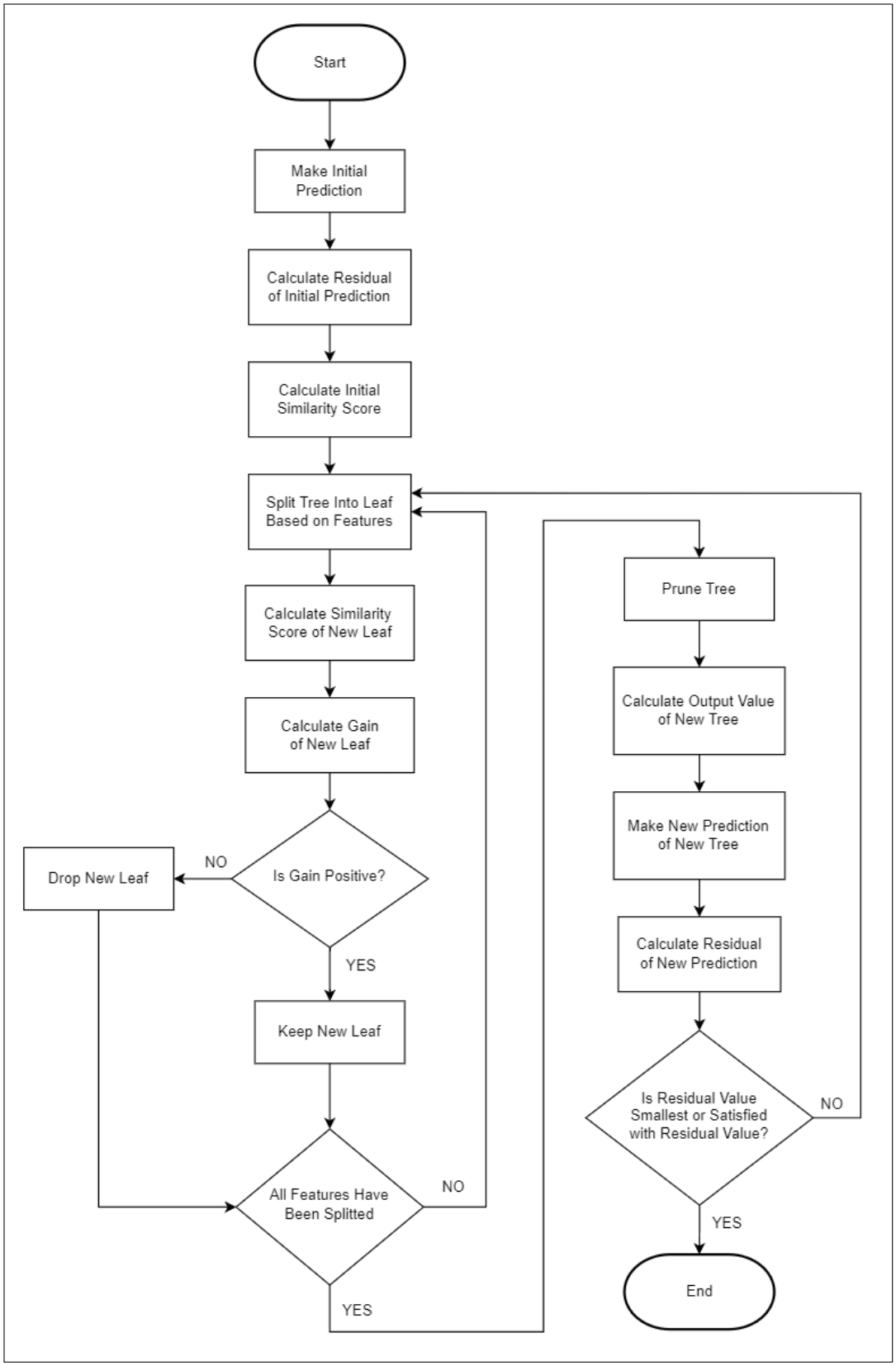
5. Membuat prediksi baru. Pada tahap ini akan dilakukan prediksi baru untuk menentukan peningkatan akurasi prediksi berdasarkan proses yang telah dilakukan sebelumnya. Berikut adalah rumus untuk melakukan prediksi baru:

$$Prediksi\ Baru = Prediksi\ Awal + (learning\ rate * Output\ Value) \tag{2.11}$$

6. Kalkulasi ulang *Residuals* untuk prediksi baru. Pada tahap ini dilakukan kalkulasi ulang *Residuals* terhadap nilai prediksi baru untuk menentukan peningkatan akurasi model, jika nilai *residual* baru lebih kecil dari pengulangan sebelumnya, maka model telah mengalami kemajuan.

7. Mengulang proses 2 hingga 6, hingga model telah mencapai jumlah pengulangan atau *epochs* yang telah ditentukan atau telah mencapai parameter *early stopping* yang telah ditentukan atau telah mencapai nilai residual yang cukup rendah.

Berikut adalah *flowchart* cara kerja algoritma XGBoost Regeressor



Gambar 2.4. Flowchart cara kerja algoritma XGBoost Regressor

2.5 Principal Component Analysis

Principal Component Analysis (PCA) adalah sebuah metode reduksi dimensi yang digunakan untuk menciptakan variabel baru bernama *components* dari kombinasi linear variabel asli. Jumlah maksimum variabel baru ini akan sama dengan jumlah variabel asli, dan variabel baru ini tidak memiliki korelasi antar satu variabel dengan yang lain [33]. PCA adalah sebuah metode yang akan mengurangi dimensi dari Data matriks input. Untuk melakukan pengurangan atau reduksi tersebut PCA memerlukan beberapa hal [34].

- *Covariance Matrix*, berfungsi untuk menangkap relasi antar variabel. Berikut adalah rumus *Covariance Matrix*.

$$\mathbf{C} = \frac{1}{n-1}\mathbf{Z}^T\mathbf{Z} \tag{2.12}$$

  Keterangan:

  - Z adalah matriks *input data* yang sudah distandardisasi
  - pc adalah *principal component* dari S
  - w adalah *weight* atau beban komponen utama

- *Eigenvalue Decomposition*, berfungsi untuk melakukan dekomposisi *eigenvalue* untuk mengidentifikasi *principal component* dari *Covariance Matrix*. Berikut adalah rumus *Eigenvalue Decomposition*.

$$\mathbf{C}\mathbf{v} = \lambda\mathbf{v} \tag{2.13}$$

  Keterangan:

  - C adalah *Covariance Matrix*
  - v adalah *eigenvector*
  - λ adalah nilai *eigenvalue*

- *Principal Components*, adalah proyeksi data asli terhadap eigenvector. Berikut adalah rumus skor *Principal Components*.

$$\mathbf{PC} = \mathbf{ZV} \tag{2.14}$$

  Keterangan:

– *PC* adalah matriks skor *Principal Components*

– *V* adalah matriks *eigenvector*

2.6 Tabel Perbandingan Related Works

Berikut adalah tabel perbandingan penelitian terdahulu yang membahas hal yang mirip dengan penelitian ini.

Tabel 2.1. Tabel *Related works*

Research Titles	This Research	Data Driven Prediction Models of Energy by Luis M. (2017) [25]	Enhancement Methods for Energy Consumption by Musa A (2023) [35]	Time-series Clustering and Forecasting household electricity by Hyojeoung K (2023) [36]
Algorithm	XGBoost Regressor	SVM Radial, RF, MLR Gradient Boost Machine	Desicion Tree, Gradient Boost Machine	ARIMA, DSHW, TBATS, NARX , NN-AR
Sample Size	19735	19735	503910	2183
Feature	Temperature, Humidity, Lights (wh), Time	Temperature, Humidity, Wind Speed, Pressure, Lights (wh), Visibility, Tdewpoint, Time	Temperature, Preassure, windspeed	Temperature, humidity, Solar Radiation
Lable	Total Energy Usage of appliances (wh)	Total Energy Usage of appliances (wh)	Total Energy Usage in KW	Total Energy Usage in KW
Training Accuracy	XGBoost Regressor RMSPE: 2.86% RMSE: 1.42 R-squared : 0.9989 MAE: 1.06 MAPE: 1.82%	SVM Radial RMSE: 39.35 R-squared : 0.85 MAPE: 15.6% Random Forest RMSE: 29.61 R-squared : 0.92 MAPE: 13.43% Multiple Linear Regression RMSE: 93.21 R-squared: 0.18 MAPE: 61.32% Gradient Boost Machine RMSE: 17.56 R-squared : 0.97 MAPE: 16.27%	Decision Tree RMSPE: 0.96% MAE: 0.17 Gradient Boost Machine RMSPE: 0.96% MAE: 0.17	ARIMA RMSE: 14.79 MAPE: 26.19 DSHW RMSE: 6.82 MAPE: 9.05% TBATS RMSE: 6.69 MAPE: 8.88% NARX RMSE: 13.72 MAPE: 12.74% NN-AR RMSE: 21.73 MAPE: 19.528%

2.7 Evaluasi Model

Untuk mengukur akurasi dari model regresi, perlu dihitung *Root Mean Square Error* (RMSE), *R-Squared*, dan *Mean Absolute Error* (MAE) [37]. Berikut adalah penjelasan dan nilai terbaik ketiga metrik menurut Davide Chicco (2021) [38]:

- *Root Mean Square Error* adalah jumlah dari kuadrat perbedaan nilai prediksi dengan aktual, lalu dibagi dengan jumlah data prediksi. Nilai terbaik untuk

RMSE adalah 0, semakin mendekati 0 maka semakin baik model. Berikut adalah rumus RMSE.

$$\text{RMSE}(y, \hat{y}) = \sqrt{\frac{\sum_{i=0}^{N-1}(y_i - \hat{y}_i)^2}{N}} \tag{2.15}$$

Keterangan:

- $\hat{y}_i$ adalah nilai prediksi
- y_i adalah nilai aktual atau observasi
- N adalah jumlah data atau observasi

- *R-Squared* adalah metrik untuk menilai seberapa besar variabel independen memengaruhi variabel dependen. Nilai terbaik untuk *R-Squared* adalah 1, semakin mendekati angka 1 maka semakin baik model. Dengan nilai di atas 0.7 dikategorikan cukup baik.

$$R^2 = 1 - \frac{RSS}{TSS} \tag{2.16}$$

Keterangan:

- RSS adalah jumlah kuadrat akibat regresi
- TSS adalah jumlah total kuadrat

- *Mean Absolute Error* adalah rata-rata perbedaan mutlak antara nilai aktual dan nilai prediksi. Nilai terbaik untuk MAE adalah 0, semakin mendekati 0 maka semakin baik model.

$$\text{MAE}(y, \hat{y}) = \frac{\sum_{i=0}^{N-1}|y_i - \hat{y}_i|}{N} \tag{2.17}$$

Keterangan:

- $\hat{y}_i$ adalah nilai prediksi
- y_i adalah nilai aktual atau observasi
- N adalah jumlah data atau observasi