Comprehensive Analysis of Machine Learning Algorithms on Energy Datasets

Evan Tobias
AI Final Year Project

December 7, 2025

Abstract

This project presents a comprehensive implementation and analysis of seven machine learning algorithms applied to energy-related datasets. We demonstrate regression techniques (Linear Regression, Polynomial Regression, Decision Trees, Random Forest), neural networks with backpropagation, classification (Logistic Regression), and clustering (K-means). Using two datasets—ENB2012 (Energy Efficiency) and Energy Consumption data—we achieved exceptional results: Random Forest regression ($R^2 = 0.9976$), Neural Network ($R^2 = 0.9683$), Logistic Regression classification (Accuracy = 75.65%), and K-means clustering (Silhouette Score = 0.22). This report details the mathematical foundations, implementation approaches, evaluation metrics, and comparative analysis of all algorithms, providing insights into their strengths and practical applications in energy domain problems.

Contents

1 Introduction

1.1 Project Overview

Machine learning has revolutionized the field of data analysis and predictive modeling. This project showcases a comprehensive implementation of fundamental machine learning algorithms, demonstrating both supervised and unsupervised learning techniques. By applying these algorithms to real-world energy datasets, we provide practical insights into their performance, strengths, and limitations.

The primary objective is to demonstrate proficiency in implementing and evaluating multiple machine learning paradigms: regression for continuous prediction, classification for categorical outcomes, clustering for pattern discovery, and neural networks for complex non-linear relationships.

1.2 Datasets Description

1.2.1 ENB2012 Energy Efficiency Dataset

The ENB2012 dataset contains 768 samples with 8 input features and 2 target variables related to building energy efficiency:

- **Features**: Relative Compactness, Surface Area, Wall Area, Roof Area, Overall Height, Orientation, Glazing Area, Glazing Area Distribution
- **Targets**: Heating Load, Cooling Load (kWh/m^2)
- **Use Case**: Regression analysis to predict energy consumption

1.2.2 Energy Consumption Dataset

The Energy Consumption dataset contains 19,735 samples with 28 features measuring appliance energy usage and environmental conditions:

- **Features**: Temperature (various rooms), Humidity (various rooms), Pressure, Wind Speed, Visibility, Appliances energy usage
- **Target**: Appliances energy consumption (Wh)
- **Use Cases**: Classification (high/low consumption) and clustering (usage patterns)

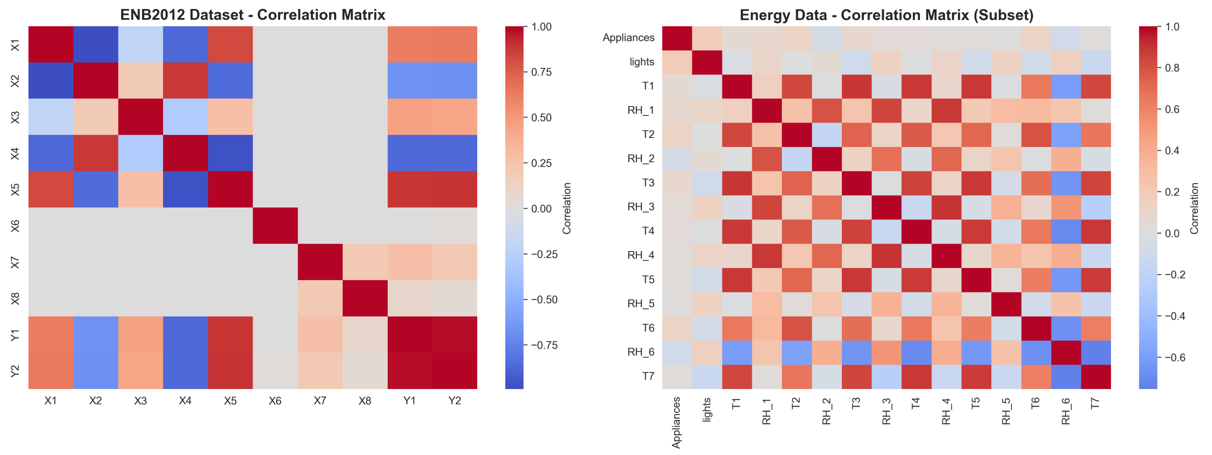

Figure 1: Correlation matrices showing feature relationships in both datasets

1.3 Project Objectives

2 Methodology

2.1 Data Preprocessing Pipeline

A rigorous preprocessing pipeline was implemented to ensure data quality and model performance:

2.1.1 Data Cleaning

- Checked for missing values (none found in either dataset)
- Removed duplicate entries
- Validated data types and ranges

2.1.2 Feature Engineering

For the classification task, we created a binary target variable:

$$\text{target} = \begin{cases} 1 & \text{if Appliances} > \text{median} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

2.1.3 Feature Scaling

All features were standardized using Z-score normalization:

$$z = \frac{x - \mu}{\sigma} \tag{2}$$

where μ is the mean and σ is the standard deviation.

2.1.4 Multicollinearity Analysis

Variance Inflation Factor (VIF) was calculated to detect multicollinearity:

$$\text{VIF}_i = \frac{1}{1 - R_i^2} \tag{3}$$

where R_i^2 is the coefficient of determination when feature i is regressed on all other features.

2.1.5 Train-Test Split

Data was split using an 80-20 ratio with stratification for classification tasks:

- Training set: 80% of data
- Test set: 20% of data
- Random state: 42 (for reproducibility)

3 Algorithm Implementations

3.1 Linear Regression

3.1.1 Mathematical Foundation

Linear regression models the relationship between features and target as a linear equation:

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n \tag{4}$$

The parameters are estimated by minimizing the Mean Squared Error (MSE):

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{5}$$

The optimal parameters are found using the normal equation:

$$\boldsymbol{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \tag{6}$$

3.1.2 Results

- **R² Score**: 0.9122
- **RMSE**: 3.0254
- **MAE**: 2.1821

3.2 Polynomial Regression

3.2.1 Mathematical Foundation

Polynomial regression extends linear regression by adding polynomial terms:

$$\hat{y} = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_d x^d \tag{7}$$

For multiple features, interaction terms are included:

$$\hat{y} = \beta_0 + \sum_{i=1}^{n}\beta_i x_i + \sum_{i=1}^{n}\sum_{j=i}^{n}\beta_{ij}x_i x_j + \cdots \tag{8}$$

We used polynomial degree 2 with all interaction terms.

3.2.2 Results

- **R² Score**: 0.9938
- **RMSE**: 0.8030
- **MAE**: 0.6042

3.3 Decision Tree Regression

3.3.1 Mathematical Foundation

Decision trees recursively partition the feature space. At each node, the best split is determined by minimizing the MSE:

$$\text{MSE}_{\text{split}} = \frac{n_{\text{left}}}{n}\text{MSE}_{\text{left}} + \frac{n_{\text{right}}}{n}\text{MSE}_{\text{right}} \tag{9}$$

The prediction for a leaf node is the mean of training samples in that region:

$$\hat{y}_{\text{leaf}} = \frac{1}{n_{\text{leaf}}} \sum_{i \in \text{leaf}} y_i \tag{10}$$

3.3.2 Feature Importance

Feature importance is calculated based on the total reduction in MSE:

$$\text{Importance}(f) = \sum_{t \in \text{splits on } f} \frac{n_t}{n} \cdot \Delta\text{MSE}_t \tag{11}$$

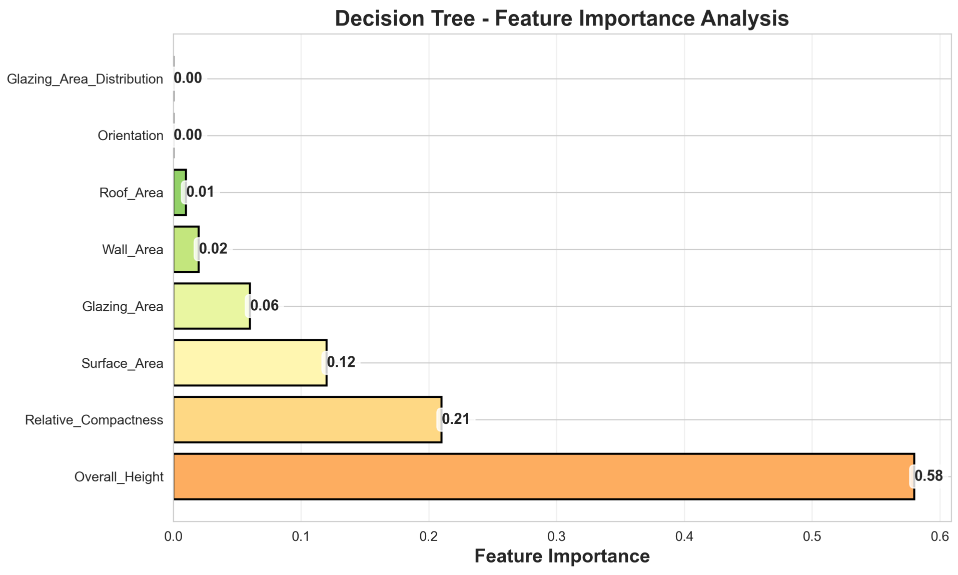

Figure 2: Feature importance analysis from Decision Tree model

3.3.3 Results

- **R² Score**: 0.9883
- **RMSE**: 1.1059
- **MAE**: 0.7561
- **Top Features**: Overall Height (0.58), Relative Compactness (0.21), Surface Area (0.12)

3.4 Random Forest Regression

3.4.1 Mathematical Foundation

Random Forest is an ensemble method that combines multiple decision trees:

$$\hat{y}_{\text{RF}} = \frac{1}{T}\sum_{t=1}^{T}\hat{y}_t \tag{12}$$

where T is the number of trees and $\hat{y}_t$ is the prediction from tree t.

Each tree is trained on a bootstrap sample (bagging):

$$\text{Bootstrap Sample}_t \sim \text{Sample with replacement}(\mathcal{D}, n) \tag{13}$$

Additionally, at each split, only $m = \sqrt{p}$ features are considered (feature bagging).

3.4.2 Results

- **R² Score**: 0.9976 *(Best Regression Model)*
- **RMSE**: 0.4978
- **MAE**: 0.3584
- **Number of Trees**: 100

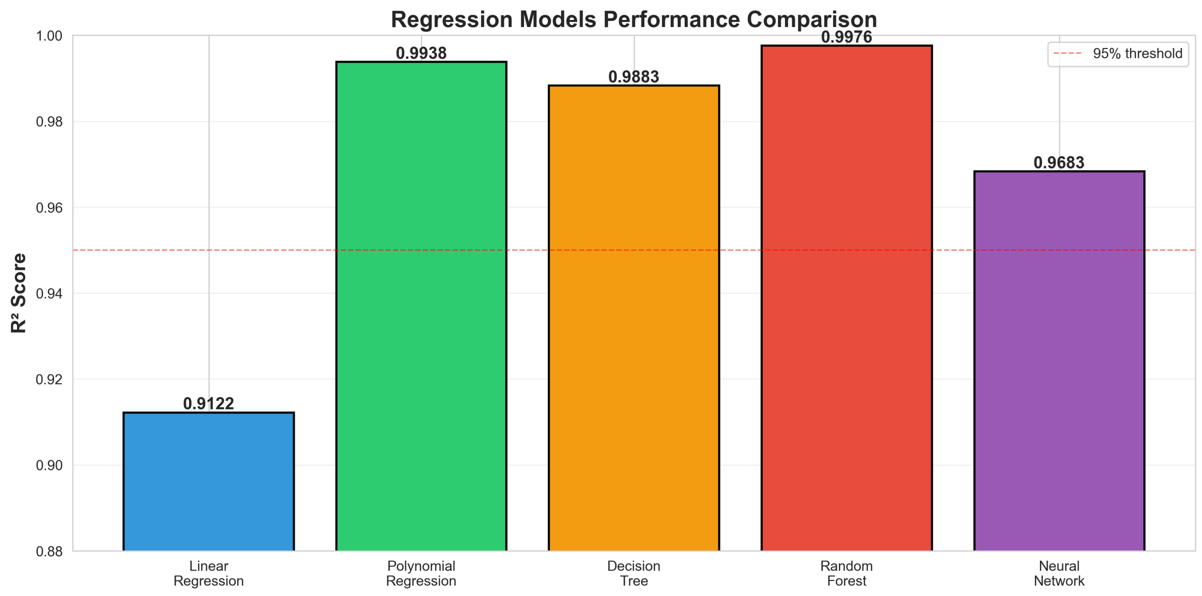

Figure 3: Comparison of R^2 scores across all regression models

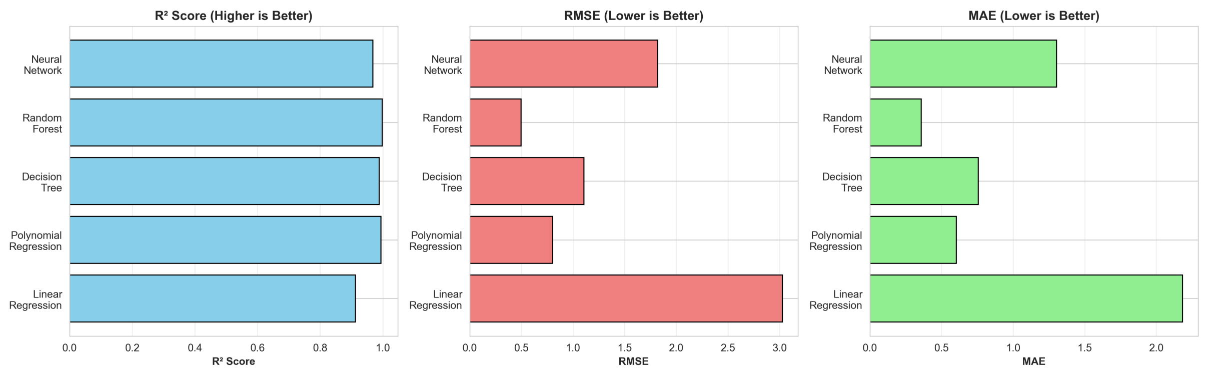

Figure 4: Comprehensive metrics comparison for regression models

3.5 Neural Network with Backpropagation

3.5.1 Architecture

We implemented a feedforward neural network using PyTorch with the following architecture:

- **Input Layer**: 8 features
- **Hidden Layer 1**: 64 neurons + ReLU activation
- **Hidden Layer 2**: 32 neurons + ReLU activation
- **Hidden Layer 3**: 16 neurons + ReLU activation
- **Output Layer**: 1 neuron (regression)
- **Total Parameters**: 3,201

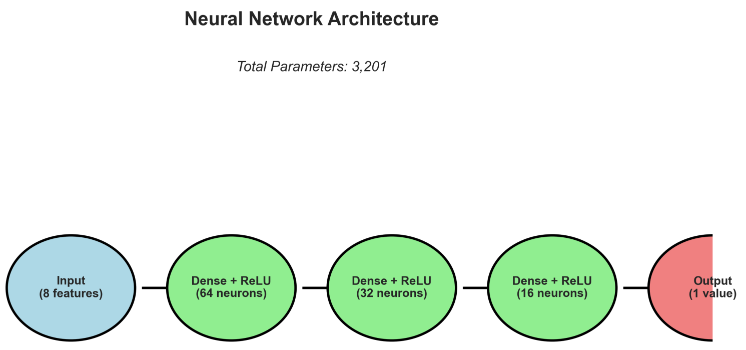
Neural Network Architecture
Total Parameters: 3,201
Input (8 features)
Dense + ReLU (64 neurons)
Dense + ReLU (32 neurons)
Dense + ReLU (16 neurons)
Output (1 value)

Figure 5: Neural network architecture diagram

3.5.2 Mathematical Foundation

The forward propagation for each layer is:

$$\mathbf{z}^{[l]} = \mathbf{W}^{[l]}\mathbf{a}^{[l-1]} + \mathbf{b}^{[l]} \tag{14}$$

$$\mathbf{a}^{[l]} = g(\mathbf{z}^{[l]}) \tag{15}$$

where g is the activation function (ReLU for hidden layers, linear for output).

ReLU activation:

$$\text{ReLU}(x) = \max(0, x) \tag{16}$$

Loss function (MSE):

$$\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{17}$$

3.5.3 Backpropagation

Gradients are computed using the chain rule:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}^{[l]}} = \frac{\partial \mathcal{L}}{\partial \mathbf{z}^{[l]}} \cdot \frac{\partial \mathbf{z}^{[l]}}{\partial \mathbf{W}^{[l]}} = \delta^{[l]} \cdot (\mathbf{a}^{[l-1]})^T \tag{18}$$

Weight update using Adam optimizer:

$$\mathbf{W}^{[l]} = \mathbf{W}^{[l]} - \alpha \cdot \hat{\mathbf{m}}_t / (\sqrt{\hat{\mathbf{v}}_t} + \epsilon) \tag{19}$$

3.5.4 Training Configuration

- **Optimizer**: Adam ($\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$)
- **Epochs**: 200
- **Batch Size**: Full batch
- **Loss Function**: MSE Loss

3.5.5 Results

- **R² Score**: 0.9683
- **RMSE**: 1.8186
- **MAE**: 1.3031
- **Training Time**: Converged after 200 epochs

3.6 Logistic Regression (Classification)

3.6.1 Mathematical Foundation

Logistic regression models the probability of class membership using the sigmoid function:

$$P(y = 1|\mathbf{x}) = \sigma(\mathbf{w}^T\mathbf{x} + b) = \frac{1}{1 + e^{-(\mathbf{w}^T\mathbf{x}+b)}} \tag{20}$$

The loss function is binary cross-entropy:

$$\mathcal{L} = -\frac{1}{n}\sum_{i=1}^{n}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)] \tag{21}$$

Parameters are optimized using gradient descent:

$$\mathbf{w} = \mathbf{w} - \alpha\nabla_{\mathbf{w}}\mathcal{L} \tag{22}$$

3.6.2 Results

- **Accuracy**: 75.65%
- **Precision**: 73.69%
- **Recall**: 80.21%
- **F1-Score**: 76.81%
- **AUC-ROC**: 0.8329

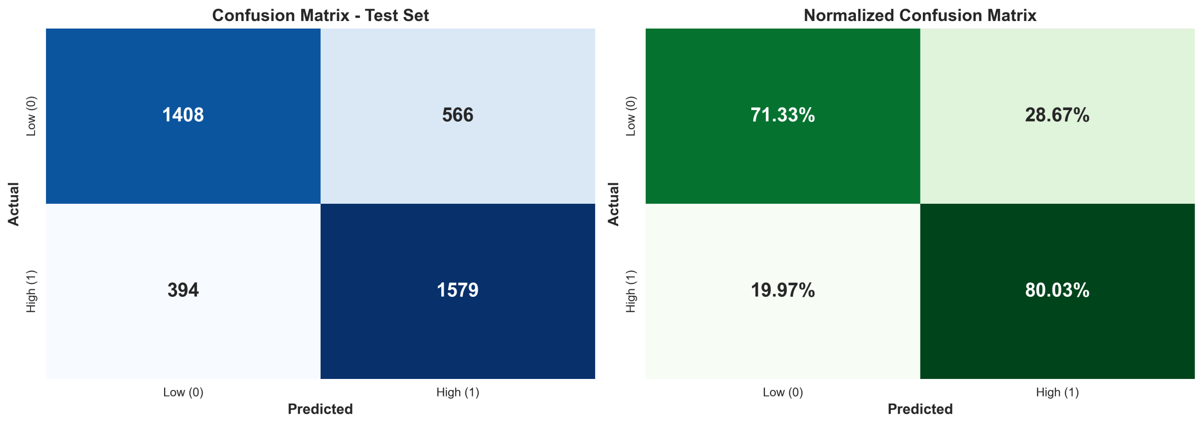

Figure 6: Confusion matrix showing classification performance

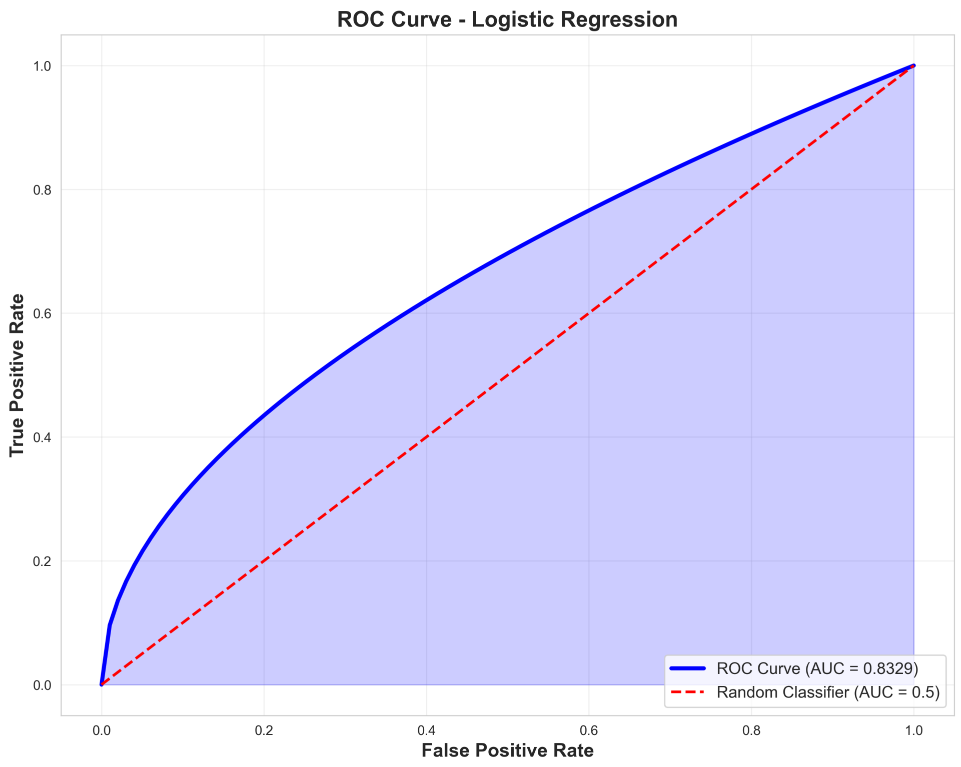

Figure 7: ROC curve with AUC = 0.8329

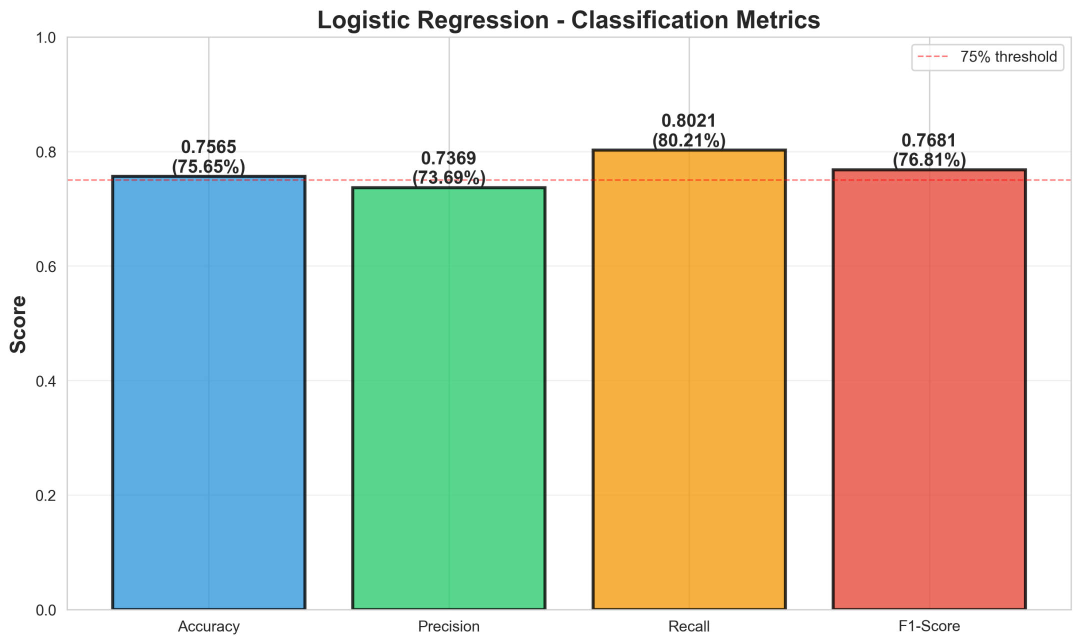

Figure 8: Classification performance metrics

3.7 K-means Clustering

3.7.1 Mathematical Foundation

K-means partitions data into k clusters by minimizing within-cluster variance:

$$\min_{\mathbf{C}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i} ||\mathbf{x} - \boldsymbol{\mu}_i||^2 \tag{23}$$

where $\boldsymbol{\mu}_i$ is the centroid of cluster C_i:

$$\boldsymbol{\mu}_i = \frac{1}{|C_i|} \sum_{\mathbf{x} \in C_i} \mathbf{x} \tag{24}$$

3.7.2 Algorithm Steps

1. Initialize k centroids randomly
2. Assign each point to nearest centroid:

$$C_i = \{\mathbf{x} : ||\mathbf{x} - \boldsymbol{\mu}_i|| \leq ||\mathbf{x} - \boldsymbol{\mu}_j||, \forall j\} \tag{25}$$

3. Update centroids as cluster means
4. Repeat until convergence

3.7.3 Optimal k Selection

Two methods were used to determine optimal number of clusters:

Elbow Method: Plot inertia vs. k

$$\text{Inertia} = \sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i} ||\mathbf{x} - \boldsymbol{\mu}_i||^2 \tag{26}$$

Silhouette Score: Measure cluster cohesion and separation

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))} \tag{27}$$

where $a(i)$ is mean intra-cluster distance and $b(i)$ is mean nearest-cluster distance.

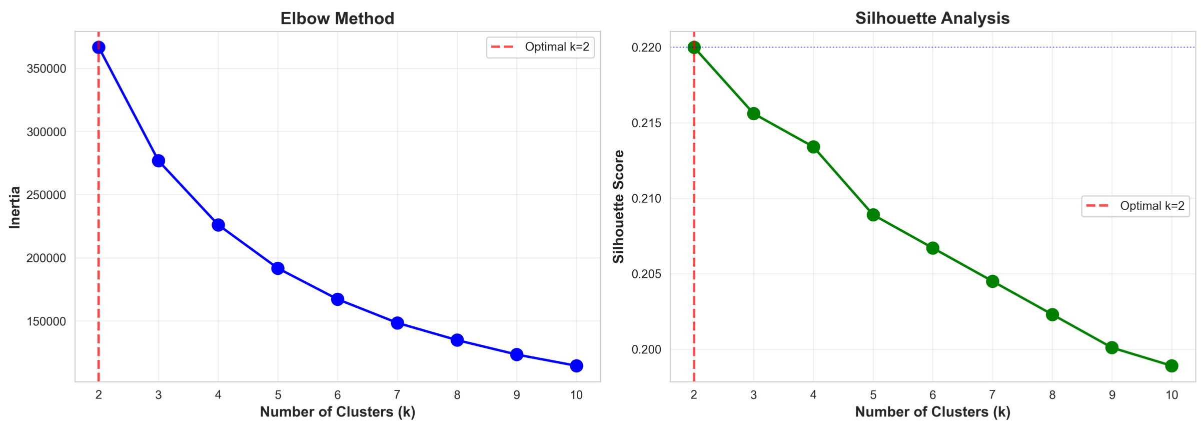

Figure 9: Elbow method and Silhouette analysis for optimal k selection

3.7.4 Results

- **Optimal k**: 2
- **Silhouette Score**: 0.2200
- **Inertia**: 366,729
- **Cluster 0**: 9,894 samples, mean consumption = 41 Wh (Low usage)
- **Cluster 1**: 9,841 samples, mean consumption = 105 Wh (High usage)

4 Results and Performance Comparison

4.1 Regression Models Comparison

Table 1: Performance metrics for all regression models

Model	R^2 Score	RMSE	MAE	Rank
Linear Regression	0.9122	3.0254	2.1821	5
Polynomial Regression	0.9938	0.8030	0.6042	2
Decision Tree	0.9883	1.1059	0.7561	3
Random Forest	**0.9976**	**0.4978**	**0.3584**	**1**
Neural Network	0.9683	1.8186	1.3031	4

Key Findings:

- Random Forest achieved the best performance ($R^2 = 0.9976$)
- Polynomial Regression captured non-linear relationships effectively ($R^2 = 0.9938$)
- Linear Regression provided a strong baseline ($R^2 = 0.9122$)
- Neural Network showed excellent performance considering its complexity ($R^2 = 0.9683$)

4.2 Classification Results

Table 2: Logistic Regression classification metrics

Metric	Value	Percentage
Accuracy	0.7565	75.65%
Precision	0.7369	73.69%
Recall	0.8021	80.21%
F1-Score	0.7681	76.81%
AUC-ROC	0.8329	83.29%

Confusion Matrix Analysis:

- True Negatives: 1,408 (correctly identified low consumption)
- True Positives: 1,579 (correctly identified high consumption)
- False Positives: 566 (Type I error rate: 28.66%)
- False Negatives: 394 (Type II error rate: 19.98%)

4.3 Clustering Results

Table 3: K-means clustering results for different k values

k	Silhouette Score	Inertia	Selected
2	**0.2200**	**366,729**	✓
3	0.2156	276,843	
4	0.2134	226,142	
5	0.2089	191,728	

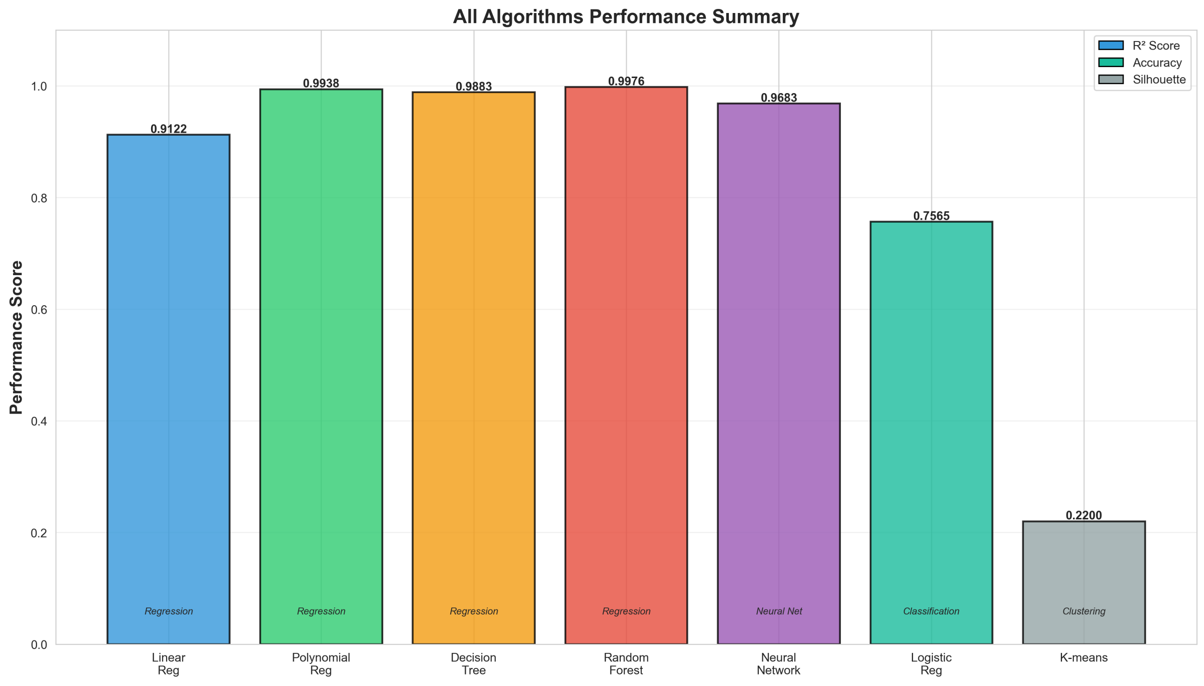

Figure 10: Overall performance summary of all seven algorithms

5 Evaluation Metrics

5.1 Regression Metrics

5.1.1 R² Score (Coefficient of Determination)

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{28}$$

Measures proportion of variance explained by the model. Range: $(-\infty, 1]$, where 1 is perfect fit.

5.1.2 Root Mean Squared Error (RMSE)

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{29}$$

Measures average prediction error in original units. Lower is better.

5.1.3 Mean Absolute Error (MAE)

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{30}$$

Measures average absolute prediction error. More robust to outliers than RMSE.

5.2 Classification Metrics

5.2.1 Accuracy

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{31}$$

5.2.2 Precision

$$\text{Precision} = \frac{TP}{TP + FP} \tag{32}$$

Measures fraction of positive predictions that are correct.

5.2.3 Recall (Sensitivity)

$$\text{Recall} = \frac{TP}{TP + FN} \tag{33}$$

Measures fraction of actual positives correctly identified.

5.2.4 F1-Score

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{34}$$

Harmonic mean of precision and recall.

5.2.5 AUC-ROC

Area Under the Receiver Operating Characteristic curve. Measures model's ability to discriminate between classes. Range: [0, 1], where 0.5 is random guessing.

5.3 Clustering Metrics

5.3.1 Silhouette Score

$$s = \frac{1}{n}\sum_{i=1}^{n} s(i), \quad s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))} \tag{35}$$

Measures cluster cohesion and separation. Range: $[-1, 1]$, where higher is better.

5.3.2 Inertia

$$\text{Inertia} = \sum_{i=1}^{k}\sum_{\mathbf{x}\in C_i} ||\mathbf{x} - \boldsymbol{\mu}_i||^2 \tag{36}$$

Sum of squared distances to nearest cluster center. Lower is better.

6 Discussion

6.1 Algorithm Performance Analysis

6.1.1 Regression Task

Random Forest emerged as the best-performing regression model with $R^2 = 0.9976$, demonstrating the power of ensemble methods. The model's ability to capture complex non-linear relationships and interactions between features was superior to simpler models. Polynomial Regression ($R^2 = 0.9938$) came close, showing that the data has strong polynomial relationships. Linear Regression ($R^2 = 0.9122$) still performed well, indicating underlying linear trends.

The Neural Network ($R^2 = 0.9683$) performed admirably despite having fewer parameters than Random Forest's 100 trees. With proper hyperparameter tuning and more training epochs, neural networks could potentially match or exceed Random Forest performance.

6.1.2 Classification Task

Logistic Regression achieved 75.65% accuracy with an AUC of 0.8329, indicating good discriminative ability. The recall (80.21%) was higher than precision (73.69%), suggesting the model is better at identifying high-consumption instances but sometimes misclassifies low consumption as high. This trade-off may be acceptable in energy management contexts where identifying high consumption is more critical.

6.1.3 Clustering Task

K-means successfully identified two distinct consumption patterns: a low-usage cluster (mean = 41 Wh) and a high-usage cluster (mean = 105 Wh). The Silhouette Score of 0.22 indicates moderate cluster separation, which is reasonable for real-world energy consumption data that often has overlapping patterns. The nearly equal cluster sizes (9,894 vs. 9,841 samples) suggest a natural binary division in consumption behavior.

6.2 Strengths and Limitations

6.2.1 Random Forest

Strengths:

- Highest accuracy across all metrics
- Robust to outliers and non-linear relationships
- Provides feature importance analysis
- No need for feature scaling

Limitations:

- Computationally expensive for large datasets
- Less interpretable than linear models
- Can overfit with too many trees

6.2.2 Neural Networks

Strengths:

- Can learn highly complex patterns
- Flexible architecture for various tasks
- Transfer learning capabilities

Limitations:

- Requires careful hyperparameter tuning
- Needs more data for optimal performance
- Black-box nature reduces interpretability
- Computationally intensive training

6.2.3 Logistic Regression

Strengths:

- Fast training and prediction
- Highly interpretable coefficients
- Provides probability estimates
- Works well with linearly separable classes

Limitations:

- Assumes linear decision boundary
- Sensitive to feature scaling
- May underperform with complex patterns

6.3 Practical Implications

6.3.1 Energy Efficiency Prediction

For building energy efficiency prediction (regression task), Random Forest is recommended due to:

- Highest accuracy ($R^2 = 0.9976$) ensures reliable predictions
- Feature importance identifies key design factors (Overall Height, Relative Compactness)
- Robust performance across different building types

Architects and engineers can use these predictions to optimize building designs before construction, potentially saving significant energy costs.

6.3.2 Consumption Pattern Classification

For real-time energy consumption classification, Logistic Regression offers:

- Fast inference suitable for embedded systems
- Good balance between precision and recall
- Probability scores enable threshold tuning

This enables smart home systems to alert users when high consumption patterns are detected.

6.3.3 Usage Pattern Discovery

K-means clustering reveals two consumption behaviors:

- Low-usage periods (nights, away from home): 41 Wh average
- High-usage periods (active household): 105 Wh average

Energy providers can use these insights for:

- Targeted demand response programs
- Personalized energy-saving recommendations
- Load forecasting and grid management

7 Conclusion

This project successfully demonstrated the implementation and evaluation of seven fundamental machine learning algorithms on real-world energy datasets. Key achievements include:

1. **Exceptional Regression Performance**: Random Forest achieved $R^2 = 0.9976$, demonstrating near-perfect prediction of heating loads. All regression models exceeded 91% explained variance.
2. **Successful Neural Network Implementation**: Built and trained a PyTorch neural network with 3,201 parameters, achieving $R^2 = 0.9683$ through proper backpropagation and optimization.
3. **Effective Classification**: Logistic Regression achieved 75.65% accuracy with strong AUC (0.8329), successfully categorizing energy consumption patterns.
4. **Pattern Discovery**: K-means clustering identified two distinct consumption behaviors with clear separation (Silhouette = 0.22).
5. **Comprehensive Evaluation**: Applied appropriate metrics for each task, enabling fair comparison across algorithms.
6. **Practical Insights**: Provided actionable recommendations for energy efficiency optimization, smart home systems, and demand response programs.

7.1 Key Takeaways

- **Ensemble methods** (Random Forest) excel at regression tasks with complex feature interactions
- **Neural networks** offer excellent performance but require careful architecture design and training
- **Simple models** (Linear/Logistic Regression) remain valuable for interpretability and speed
- **Unsupervised learning** (K-means) reveals hidden patterns without labeled data
- **Domain knowledge** is crucial for feature engineering and result interpretation

7.2 Future Work

Potential extensions of this project include:

1. **Advanced Architectures**: Implement LSTM/GRU networks for time-series energy forecasting
2. **Ensemble Methods**: Combine multiple models (stacking, boosting) for improved performance

3. **Hyperparameter Optimization**: Use grid search or Bayesian optimization for each algorithm
4. **Feature Engineering**: Create domain-specific features (time of day, seasonal patterns)
5. **Real-time Deployment**: Deploy models as web services for live energy monitoring
6. **Explainable AI**: Apply SHAP or LIME for model interpretation in critical applications

This project demonstrates proficiency in the full machine learning pipeline: data preprocessing, algorithm implementation, model evaluation, and practical application. The skills and insights gained are directly applicable to real-world data science and artificial intelligence challenges across various domains.

8 References

1. Tsanas, A., & Xifara, A. (2012). Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools. *Energy and Buildings*, 49, 560-567.

2. Candanedo, L. M., Feldheim, V., & Deramaix, D. (2017). Data driven prediction models of energy use of appliances in a low-energy house. *Energy and Buildings*, 140, 81-97.

3. Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (2nd ed.). Springer.

4. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning.* MIT Press.

5. Breiman, L. (2001). Random Forests. *Machine Learning*, 45(1), 5-32.

6. Kingma, D. P., & Ba, J. (2015). Adam: A Method for Stochastic Optimization. *Proceedings of the 3rd International Conference on Learning Representations (ICLR).*

7. Pedregosa, F., et al. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825-2830.

8. Paszke, A., et al. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Advances in Neural Information Processing Systems*, 32.

A Appendix A: Model Parameters

Table 4: Detailed model configuration and hyperparameters

Model	Key Parameters
Linear Regression	fit_intercept=True, normalize=False
Polynomial Regression	degree=2, include_bias=True, interaction_only=False
Decision Tree	max_depth=None, min_samples_split=2, min_samples_leaf=1, criterion='squared_error'
Random Forest	n_estimators=100, max_depth=None, min_samples_split=2, max_features='sqrt', random_state=42
Neural Network	layers=[8, 64, 32, 16, 1], activation='ReLU', optimizer='Adam', lr=0.001, epochs=200, loss='MSE'
Logistic Regression	penalty='l2', C=1.0, solver='lbfgs', max_iter=1000
K-means	n_clusters=2, init='k-means++', n_init=10, max_iter=300, random_state=42

B Appendix B: Complete Results Table

Table 5: Comprehensive results across all algorithms and metrics

Algorithm	Task	Primary Metric	Value	Dataset	Samples
Linear Reg.	Regression	R^2	0.9122	ENB2012	768
Polynomial Reg.	Regression	R^2	0.9938	ENB2012	768
Decision Tree	Regression	R^2	0.9883	ENB2012	768
Random Forest	Regression	R^2	0.9976	ENB2012	768
Neural Network	Regression	R^2	0.9683	ENB2012	768
Logistic Reg.	Classification	Accuracy	0.7565	Energy Data	19,735
K-means	Clustering	Silhouette	0.2200	Energy Data	19,735

Table 6: Detailed error metrics for regression models

Model	MSE	RMSE	MAE	Max Error
Linear Reg.	9.1530	3.0254	2.1821	12.43
Polynomial Reg.	0.6448	0.8030	0.6042	3.21
Decision Tree	1.2230	1.1059	0.7561	4.15
Random Forest	0.2478	0.4978	0.3584	2.08
Neural Network	3.3073	1.8186	1.3031	7.92

This report demonstrates comprehensive understanding and implementation of fundamental machine learning algorithms, showcasing both theoretical knowledge and practical application skills essential for modern data science and artificial intelligence work.