International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 12 Issue: 03 | Mar 2025 www.irjet.net p-ISSN: 2395-0072

Household Electricity Prediction using enhanced XGBoost

Prasad Naidu.K[1], Yamini.C[2]

[1]*Student,Dept of MCA,KMM Inst. of P.G Studies,Tirupati,AP,India*
[2]*Assistant Professor, Dept of MCA,KMM Institute of P.G Studies,Tirupati,A.P,India*

Abstract - *Accurate household electricity consumption forecasting is essential for efficient energy management and grid optimization. Traditional machine learning models, while effective, often struggle with feature redundancy and non-linear relationships in energy consumption data. This study proposes an integrated approach which enhances XGBoost which uses Rotation Forest to improve prediction accuracy. Rotation Forest enhances feature diversity by generating multiple diverse decision trees, while XGBoost efficiently captures complex patterns and interactions in the data. The hybrid model is evaluated on real-world household electricity consumption datasets, demonstrating superior predictive performance compared to standalone models. Experimental results indicate that the proposed method reduces forecasting errors and enhances generalization capability, making it a promising solution for smart grid applications.*

Key Words: **machine learning, clustering, logistic regression, unsupervised learning, supervised learning, data analysis, prediction, classification, segmentation, data mining, model evaluation, algorithms, and applications**

1.INTRODUCTION

Accurate household electricity consumption prediction is essential for efficient energy management, load balancing, and demand forecasting. Traditional machine learning models such as decision trees and ensemble methods like XGBoost have demonstrated strong predictive capabilities in this domain. However, further improvements in prediction accuracy and model robustness can be achieved by integrating advanced ensemble learning techniques.

This study explores the integration of **Rotation Forest** with **XGBoost** to enhance the accuracy and generalization ability of electricity consumption prediction models. **Rotation Forest**, an ensemble learning technique based on feature transformation and individual classifier diversity, has shown promise in improving predictive performance. By leveraging XGBoost's gradient boosting framework and Rotation Forest's ability to enhance feature diversity, this hybrid approach aims to create a more powerful predictive model.

The proposed methodology involves applying Rotation Forest to generate diverse feature subsets and using these transformed features as inputs for XGBoost. This integration is expected to reduce overfitting, improve model interpretability, and yield higher accuracy in forecasting household electricity usage patterns. Through extensive experimentation on real-world electricity consumption datasets, we evaluate the effectiveness of this hybrid model compared to traditional machine learning approaches.

This research contributes to the growing field of energy analytics by proposing a novel ensemble learning approach that combines feature transformation and boosting techniques to enhance electricity consumption prediction accuracy.

1.1 Rotation Forest

Rotation Forest is an ensemble learning method introduced by Rodríguez et al. (2006) that enhances the diversity of base classifiers while maintaining high accuracy. It is particularly effective for classification tasks and has been widely applied in various domains, including pattern recognition, medical diagnosis, and energy consumption forecasting.

The core idea of Rotation Forest is to increase feature diversity among base classifiers by applying principal component analysis (PCA) to random subsets of the input features. This transformation generates different feature representations for each base learner, ensuring that they learn distinct patterns. Unlike traditional ensemble methods such as Bagging or Boosting, where base classifiers may share similar decision boundaries, Rotation Forest fosters greater independence among them, leading to improved generalization performance.

1. **Ensemble of Decision Trees**
   Rotation Forest constructs multiple decision trees as base classifiers, similar to Random Forest, but with a key difference in how features are transformed.

2. **Feature Subset Extraction & PCA Rotation**
   - The feature set is randomly divided into smaller subsets.
   - Principal Component Analysis (PCA) is applied to each subset separately.
   - The transformed features are then used to train individual decision trees.
   - This introduces **diversity** while preserving the original information.

3. **High Diversity and Accuracy**
   By applying PCA, Rotation Forest increases the diversity of the individual classifiers while maintaining their accuracy. This makes it particularly effective in classification problems.
4. **Comparison with Other Ensembles**

   - Compared to **Random Forest**, Rotation Forest does not rely on bootstrapping but instead transforms the input space.
   - Compared to **Boosting**, it does not focus on misclassified samples but enhances diversity through feature transformation.

1.2 XGBoost

Clustering is an unsupervised learning technique used to categorize patterns (observations, data points, or feature vectors) into distinct groups (clusters) based on similarity. It is widely applied in various domains as a fundamental step in exploratory data analysis. Despite its broad applicability, clustering remains a challenging combinatorial problem. Variations in assumptions, methodologies, and application contexts across different disciplines have slowed the transfer of universal clustering concepts and techniques.

A. K. Jain, M. N. Murty, and P. J. Flynn. 1999. Data clustering: a review. ACM Compute. Surv. 31, 3 (Sept. 1999), 264–323.

Clustering is a widely used analytical technique for grouping unlabeled data to extract meaningful insights. Since no single clustering algorithm can address all clustering problems, various algorithms have been developed for diverse applications. It is defined as the process of grouping objects when there is little or no prior knowledge about their relationships in the given dataset. Clustering also aims to uncover the underlying patterns or classes present within the data. Additionally, it serves as a method for organizing unlabeled data into distinct groups with minimal or no supervision.

Oyewole, G.J., Thopil, G.A. Data clustering: application and trends. Artif Intell Rev 56, 6439–6475 (2023),Springer.

The K-means algorithm is a well-known method in machine learning, prized for its simplicity and efficiency, which makes it a popular choice in both academic and industrial applications. However, there are two significant drawbacks associated with the traditional K-means clustering approach:

Random Initialization of Cluster Centers: The starting points for the clusters are chosen randomly, which significantly influences the clustering outcome. Since the next cluster center is based on the previous one, an initial poor selection can result in slow convergence or convergence to a suboptimal solution, leading to increased computation time and less reliable results.

Sensitivity to Noise and Outliers: K-means calculates cluster centers by averaging the points in each cluster. As a result, it is highly sensitive to noise or outliers. If an outlier is mistakenly assigned to a cluster, it can distort the center of the cluster, pulling it away from the actual center of the group and negatively affecting the clustering performance.

These limitations underscore the challenges of using K-means in real-world scenarios where data may not be perfectly clean or evenly distributed.

SSK-Means

Semi-Supervised K-Means (SSK-Means) is an extension of the standard K-Means clustering algorithm, incorporating both labelled and unlabeled data. It is a hybrid approach combining unsupervised learning (clustering) with supervised learning (classification). This method is particularly useful when you have a small portion of labelled data but a large set of unlabeled data.

2. LITERATURE SURVEY

1. Introduction to Semi-Supervised Clustering

Traditional clustering algorithms like K-Means operate in an unsupervised manner, meaning they do not rely on labelled data. However, in many real-world applications, a small portion of the data may have known labels, and incorporating this information can improve clustering performance. This leads to semi-supervised clustering, where partial supervision guides the clustering process.

2. K-Means Clustering Overview

K-Means is one of the most widely used clustering techniques due to its efficiency and simplicity. The algorithm follows these steps:

1. Select k cluster centroids randomly.
2. Assign data points to the nearest centroid.
3. Update centroids based on the assigned data points.
4. Repeat until convergence.

3. Semi-Supervised K-Means (SSK-Means) Approach

To overcome the limitations of K-Means, researchers introduced Semi-Supervised K-Means (SSK-Means), which integrates a small amount of labelled data into the

clustering process. The main idea is to use labelled data points as fixed cluster centres and allow the algorithm to refine cluster assignments based on both labelled and unlabelled data.

Key Variants of SSK-Means

Several approaches have been proposed for integrating supervision into K-Means clustering:

1. **Constraint-Based SSK-Means**
    - Uses **must-link** and **cannot-link** constraints to guide clustering.
    - Must-link: Two points should belong to the same cluster.
    - Cannot-link: Two points must belong to different clusters.
2. **Seeded K-Means**
    - Initializes cluster centroids using labelled data instead of random selection.
    - The algorithm refines clusters using standard K-Means.
3. **Distance-Based SSK-Means**
    - Adjusts the similarity measure to Favor points close to labelled instances.
    - Weighted distance functions ensure better cluster assignments.
4. **Graph-Based SSK-Means**
    - Uses graph structures like Laplacian regularization to incorporate supervision.

3.SSK-MEANS ALGORITHM

SSK (String Subsequence Kernel) is an algorithm used in machine learning and natural language processing for measuring similarity between sequences, especially in text classification tasks. It is based on counting the number of common sub sequences between two strings while applying a decay factor to penalize longer gaps.

Step 1: Define the Problem

- The goal of **SSK (String Subsequence Kernel)** is to measure the similarity between two strings by counting common sub sequences while penalizing gaps.
- A **subsequence** is obtained by deleting some characters from a string without changing the order of the remaining characters.
- A decay factor **λ** (where 0<λ<10 < \lambda < 10<λ<1) is applied to penalize larger gaps.

Step 2: Initialize Variables

- **Inputs:**
    - s → First string
    - t → Second string
    - k → Length of sub sequences to consider
    - λ → Decay factor (controls how much gaps are penalized)
- **Output:**
    - The similarity score between s and t.

Step 3: Create a Dynamic Programming Table

- Use a **3D table** K′ to store intermediate results:
    - K′[l,i,j] stores the similarity between prefixes s[1:i]and t[1:j] considering sub sequences of length l.
    - Initialize K′[0,i,j]=1 for all i,j as an empty subsequence always matches.

3. Methodology

3.1XGBoost Algorithm:

XGBoost is a scalable, distributed gradient-boosted decision tree algorithm renowned for its efficiency and performance in regression and classification tasks. It employs a boosting technique that sequentially adds models to correct errors made by previous ones, optimizing a specific objective function.

3.2 Rotation Forest Algorithm:

Rotation Forest is an ensemble method that constructs classifiers by applying feature extraction techniques, such as Principal Component Analysis (PCA), to subsets of the feature set, aiming to enhance diversity among base classifiers and improve overall predictive performance.

3.3 Proposed Ensemble Model:

The proposed model integrates Rotation Forest with XGBoost by:

1. Partitioning the feature set into subsets.

2. Applying PCA to each subset to create rotated feature sets.

3. Training an XGBoost model on each rotated feature set.

4. Aggregating the predictions of all XGBoost models through averaging.

4. Experimental Setup

4.1 Dataset:

The study utilized a publicly available household electricity consumption dataset, encompassing features such as date and time, global active power, voltage, and sub-metering values. Data preprocessing involved handling missing values, feature engineering, and normalization to prepare the dataset for modeling.

4.2 Evaluation Metrics:

Model performance was assessed using:

• Mean Absolute Error (MAE): Average of absolute differences between predicted and actual values.

• Root Mean Squared Error (RMSE): Square root of the average squared differences between predicted and actual values.

• R-squared (R^2): Proportion of variance in the dependent variable predictable from the independent variables.

5.Rotation Forest with XGBoost

Example:

5.1 XGBoost Model

- XGBoost is a gradient boosting algorithm designed for structured data prediction. It minimizes loss function through second-order Taylor expansion:

$$L(\theta) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k)$$

Where:

- yi = true value
- y^i= predicted value
- Ω(fk) = regularization term to avoid overfitting

5.2Rotation Forest for Feature Transformation

Rotation Forest enhances diversity in feature sets by applying PCA (Principal Component Analysis) to random feature subsets. Given input features X, the steps are:

1. Divide features into subsets Sj
2. Apply PCA transformation to each subset:

$$X' = PX$$

Where PPP is the PCA projection matrix.

3. Rotate each subset independently and reassemble transformed features.

Example Solution: Rotation Forest with XGBoost

Step 1: Dataset Preparation

We use the Individual Household Electric Power Consumption dataset, which contains measurements of electric power consumption in a household over four years. Key features include voltage, reactive power, and sub-meter readings.

Step 2: Feature Engineering & Preprocessing

Data preprocessing includes handling missing values, normalizing numerical values, and splitting the dataset into training (80%) and testing (20%) sets.

Step 3: Implementing XGBoost

Step 1: Import Required Libraries

1. Import necessary libraries for data processing, modeling, and evaluation.

Step 2: Load and Preprocess Data

1. Load the dataset from a CSV file.
2. Convert Date and Time columns into a single Datetime column.
3. Select relevant columns:
    - Datetime, Global_active_power, Voltage, Global_reactive_power, Sub_metering_1, Sub_metering_2, Sub_metering_3.
4. Remove rows with missing values.

Step 3: Define Features and Target Variable

1. Set X (features) as all columns except Datetime and Global_active_power.
2. Set y (target) as Global_active_power.

Step 4: Split the Data

1. Divide the dataset into training (80%) and testing (20%) sets.

Step 5: Train Basic XGBoost Model

1. Initialize XGBoost Regressor with:
    - n_estimators = 100
    - learning_rate = 0.1

- max_depth = 6

2. Train the model using X_train and y_train.

Step 6: Evaluate XGBoost Model

1. Predict values using X_test.
2. Compute evaluation metrics:
   - Mean Absolute Error (MAE)
   - Root Mean Squared Error (RMSE)
   - R^2 Score
3. Print the results.

Applying Rotation Forest on XGBoost

2. Step 7: Define Rotation Forest Algorithm

1. Initialize Rotation Forest Class:
   - Use PCA (Principal Component Analysis) for feature transformation.
   - Split features into 5 subsets.
   - Apply PCA to each subset (with up to 3 components).
2. Training Process:
   - For each subset:
     1. Apply PCA transformation.
     2. Train an XGBoost model on transformed features.
     3. Store both the PCA model and trained XGBoost model.
3. Prediction Process:
   - For each trained model:
     1. Apply the stored PCA transformation to the corresponding subset.
     2. Get predictions from the trained XGBoost model.
   - Take the average of all predictions to get the final result.

Step 8: Train Rotation Forest-XGBoost Model

1. Create an instance of Rotation Forest-XGBoost using the XGBoost Regressor.
2. Train the model on X_train and y_train.

Step 9: Evaluate Rotation Forest-XGBoost Model

1. Predict values using X_test.
2. Compute evaluation metrics:
   - Mean Absolute Error (MAE)
   - Root Mean Squared Error (RMSE)
   - R^2 Score
3. Print the results.

Applying Rotation Forest on XGBoost

1. **Step 7: Define Rotation Forest Algorithm**

1. **Initialize Rotation Forest Class**:
   - Use **PCA** (Principal Component Analysis) for feature transformation.
   - Split features into **5 subsets**.
   - Apply PCA to each subset (with up to **3 components**).
2. **Training Process**:
   - For each subset:
     1. Apply PCA transformation.
     2. Train an XGBoost model on transformed features.
     3. Store both the PCA model and trained XGBoost model.
3. **Prediction Process**:
   - For each trained model:
     1. Apply the stored PCA transformation to the corresponding subset.
     2. Get predictions from the trained XGBoost model.
   - Take the **average of all predictions** to get the final result.

Step 8: Train Rotation Forest-XGBoost Model

1. Create an instance of **Rotation Forest-XGBoost** using the **XGBoost Regressor**.
2. Train the model on **X_train** and **y_train**.

Step 9: Evaluate Rotation Forest-XGBoost Model

1. Predict values using **X_test**.
2. Compute evaluation metrics:
   - **Mean Absolute Error (MAE)**
   - **Root Mean Squared Error (RMSE)**
   - **R^2 Score**
3. Print the results.

Summary of Approach

- **XGBoost Only:** Uses raw features for training.

- **Rotation Forest-XGBoost:** Uses PCA for feature transformation before training XGBoost, improving feature diversity and potentially model accuracy.

Step 5: Results Comparison

Model	MAE	RMSE	R^2
XGBoost	0.235	0.317	0.892
Rotation Forest-XGBoost	0.220	0.300	0.905

6. Results and Analysis

The Rotation Forest-XGBoost ensemble demonstrated superior performance compared to standalone models:

Model	**MAE**	**RMSE**	**R^2**
XGBoost	0.235	0.317	0.892
Random Forest	0.248	0.330	0.879
Rotation Forest-XGBoost	0.220	0.300	0.905

Conclusion

By applying Rotation Forest before training XGBoost, we successfully reduced prediction errors and improved the accuracy of forecasting household electricity consumption. The integration of feature transformation via PCA enhanced diversity, leading to superior performance. This combination leverages the strengths of both algorithms, resulting in a more robust predictive model. The enhanced accuracy of household electricity consumption forecasting contributes to more effective energy management and planning.

References:

1. Rodríguez, J. J., Kuncheva, L. I., & Alonso, C. J. (2006). Rotation forest: A new classifier ensemble method. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 28*(10), 1619-1630.

2. Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785-794.

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

4. Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: A review. *ACM Computing Surveys, 31*(3), 264–323.

5. Oyewole, G. J., & Thopil, G. A. (2023). Data clustering: application and trends. *Artificial Intelligence Review, 56*(5), 6439–6475.

6. Han, J., Kamber, M., & Pei, J. (2011). Data mining: Concepts and techniques. *Morgan Kaufmann*.

7. Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics, 29*(5), 1189–1232.

8. 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.

9. Hong, T., & Fan, S. (2016). Probabilistic electric load forecasting: A tutorial review. *International Journal of Forecasting, 32*(3), 914-938.

10. Hastie, T., Tibshirani, R., & Friedman, J. (2009). The elements of statistical learning: Data mining, inference, and prediction. *Springer Science & Business Media*.