Improving predictive performance of GBoost and XGBoost Algorithms on Time Series Data Forecasting by modifying learning rate

Aasneh Prasad 210101001 Pranav Jain 210101078 Sahil Danayak 210101092 Sanjana Kolisetty 210101093

1. Introduction

1.1. Motivation

Time series forecasting has been gaining a lot of popularity lately. Algorithms like GBoost & XGBoost have been proved to work quite impressively on time series forecasting problems. Our focus is to implement these algorithms from scratch & improve the performance by working on learning rate. By doing so, we will be addressing several problems with time series data like seasonality and trend detection.

1.2. Target Problem

Our main focus is to optimize the predictive performance of the GBoost and XGBoost algorithms on time series datasets, such as the Hourly Energy Consumption Dataset, by exploring and comparing various learning rate methods, with the objective of enhancing model accuracy, robustness and efficiency in forecasting future observations.

1.3. Major Challenges

The major challenge would be implementing the algorithms and integrating them to work on our proposed dataset. Other challenges include the preprocessing of the data and tuning the learning rate to achieve the best possible results.

1.4. Outline of Proposed Direction

In the data pre-processing phase, outliers will be removed from the chosen dataset to ensure the integrity of the data.

During feature engineering and exploratory data analysis (EDA), we will use trigonometric encoding which will hep us in preserving and capturing essential patterns and cyclic nature of time series data.

Moving on to model training algorithms, we will initially employ a fixed learning rate and subsequently enhance it using advanced techniques like simulated annealing and adaptive methods such as the Adam optimizer.

To further refine our model's performance, if time permits, we will delve into pruning techniques aimed at mitigating over-fitting issues. Additionally, we may explore hyperparameter tuning methods, including Bayesian optimization and grid search, to fine-tune model parameters and potentially enhance forecasting accuracy.

2. Methods

2.1. Exploratory Data Analysis

Exploratory Data Analysis (EDA) is a crucial step in understanding the characteristics and patterns present in time series data. Time series data represents observations collected sequentially over time, making it essential to explore the temporal dependencies, trends, and seasonal patterns before applying any modeling techniques. Features such as months, days, weekdays, hours, minutes, seconds are all cyclical in nature. Numerous methods exist to encode cyclic data in machine learning include:

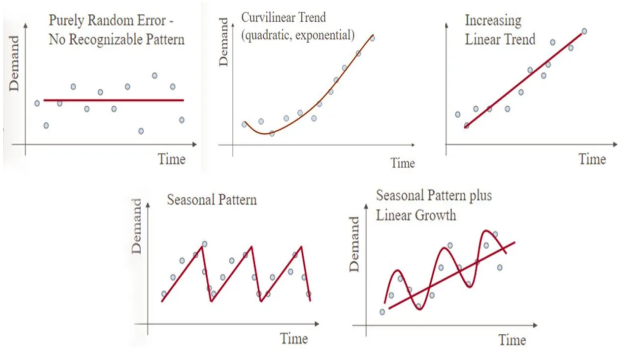

Figure 1. Different types of Time series data

- **Ordinal encoding:** Each value in a category is represented by a number.
- **One hot encoding:** This is a binary representation of each value, where each value is represented by a vector of length n, where n is the number of possible values.
- **Trigonometric encoding:** Mapping the cyclic feature onto a unit circle.

2.2. GBoost & XGBoost Algorithms

Boosting algorithms are powerful machine learning techniques that iteratively combine weak learners to create a strong ensemble model.

Gradient Boosting creates an ensemble model from numerous weak predictors, usually Regression Trees, which are

added in a stage wise fashion with each new tree focusing on the pseudo-residuals of the previous tree.

$$F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \rho_m h_m(\mathbf{x}) \tag{1}$$

$$r_{mi} = \left[\frac{\partial L\left(y_i, F(\mathbf{x})\right)}{\partial F(\mathbf{x})}\right]_{F(\mathbf{x})=F_{m-1}(\mathbf{x})} \tag{2}$$

XGBoost enhances GBoost by incorporating additional regularization techniques, such as L1 and L2 regularization, to control overfitting. It computes the second order derivative to gather more information to make a better approximation about the direction of the maximum decrease (in the loss function).

$$\mathrm{Obj}(\Theta) = \sum_{i=1}^{n} \ell\left(y_i, \hat{y}_i\right) + \sum_{k=1}^{K} \Omega\left(f_k\right) \tag{3}$$

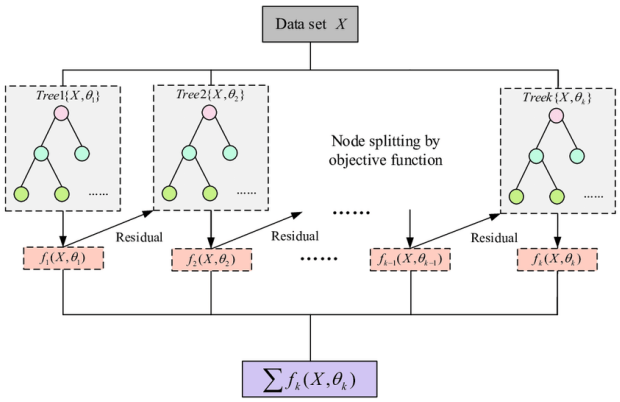

Figure 2. Gradient boosted regression trees

2.3. Optimizing Learning Rates

In this project, we would modify learning rate of XGBoost algorithm.

- **Fixed Learning Rate:** This is the simplest approach in which the learning rate remains constant throughout training. It is easy to implement it but it leads to slow convergence (if learning rate is low) or even might lead to divergence (if learning rate is high).
- **Simulated Annealing:** Here, the learning rate gradually decreases over time. Initially, this method explores a wider range of solutions (higher learning rate) and then gradually settles into a more refined search (lower learning rates). This helps escape local minima and converge towards global optima.
- **Adam (Adaptive Moment Estimation):** This method combines the advantages of both adaptive learning rates and momentum. It dynamically adjusts the learning rate for each parameter based on the first and second moments of their gradients.

3. Intended Experiments

3.1. Data Set

We intend on using the Hourly Energy Consumption Dataset. This dataset contains the hourly consumption of energy in megawatts(MW) for over past 10 years.

3.2. Trigonometric Encoding

The efficiency and simplicity of sine-cosine embeddings makes it a good candidate for capturing the periodicity/cyclic dependency of time, date, month etc. The cyclic relation and proximity of 23:59 and 00:00 can be captured by sine-cosine embedding.

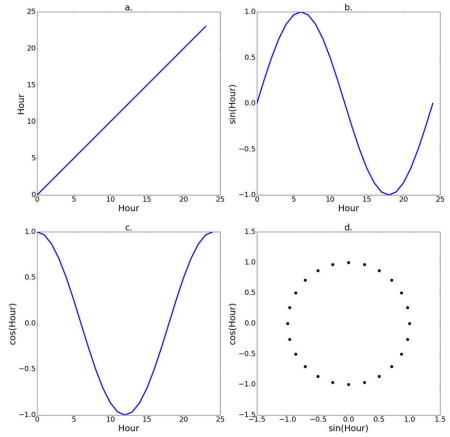

Figure 3. sine-cosine encoding

3.3. GBoost and XGBoost

XGBoost and GBoost have the potential to perform well on time series datasets, but one needs to be careful to select appropriate hyperparameters, including learning rate, the maximum tree depth or maximum number of tree nodes and maybe pruning of branches and possibly incorporate domain knowledge to achieve optimal performance. Time series data often exhibit non-linear patterns and complex relationships between variables. GBoost and XGBoost are based on decision trees, which inherently have the ability to model non-linear relationships between features and the target variable by recursively splitting the feature space into regions, and each split can be based on any feature and any threshold, allowing for the capture of complex and non-linear decision boundaries.

3.4. Learning rate improvement

We aim to enhance the learning rate of XGBoost for time series prediction by customizing the training process to incorporate various algorithmic enhancements. This involves customizing the learning rate using fixed Learning rate, Simulated annealing and ADAM within XGBoost. Each method offers its own trade-off in terms of simplicity, performance and adaptability, thereby requiring experimentation to determine the most effective strategy for the time series prediction task.

4. References

- XGBoost: Introduction to XGBoost Algorithm in Machine Learning, Analytics Vidhya
- XGBoost: A Scalable Tree Boosting System
- XGBoost Documentation
- Hourly Energy Consumption Dataset, Kaggle
- Handling cyclical features, such as hours in a day, for machine learning pipelines with Python example, Self-idian Academy
- A Comparative Analysis of XGBoost: Candice BentéjacAnna Csörgő, Gonzalo Martínez-Muñoz
- A Proof of Local Convergence for the Adam Optimizer: Sebastian Bock, Martin Weiß