# Time Series Forecasting Report

**Virtual Internship**

**Yes Energy - Team 2**


Chenxin Huang

Mona Qi

Omkar Joshi

Weiyu Liu

Wenqin Li (Wendy)

Zhikang Qin (Tony)

**University of Canterbury**

## 1. INTRODUCTION

Yes Energy is a leading energy market data platform with ISO/RTO data, LMP prices, FTR auction results, transmission and generation outages, real-time generation and flow data, and load and weather forecasts. They are looking to develop a product for electricity price prediction. In this report, the team will use time series models to make predictions, including data cleaning, feature engineering, feature selection, and model evaluation. The goal is to achieve an accurate prediction algorithm for electricity price forecasting.

## 2. DATASET AND PREPROCESSING

### 2.1 Overview and Objectives

Data from the past three and a half years were provided, including electricity prices, electricity demand, electricity generation, and weather. A time series algorithm is required to predict future electricity prices.

### 2.2 Data Cleaning and Preparation

All data provided were standardized to a half-hourly basis. Linear interpolation was applied to replace the missing values. The overall data structure, including mean, median, Q1, and Q3, remained mostly the same. The data files were combined into one file eventually. All the models were run based on the file named "tokyo_electricity_final_dataset_half_hourly.csv" in the modeling process.

Table 1. Summary for final dataset

| | Min | Q1 (25th) | Mean | Median | Q3 (75th) | Max |
|---|---|---|---|---|---|---|
| actual_electricity_price | 0.01 | 9.36 | 17.19 | 13.42 | 20.01 | 252.00 |
| electricity_demand | 1855.00 | 2664.00 | 3194.92 | 3116.00 | 3610.00 | 5930.00 |
| solar_generation | 0.00 | 0.00 | 266.89 | 4.00 | 449.00 | 1635.33 |
| wind_generation | 0.00 | 4.00 | 9.08 | 7.00 | 13.00 | 41.00 |
| temperature(C) | -3.20 | 9.50 | 16.38 | 16.40 | 23.00 | 37.20 |
| cloud(8) | 0.00 | 2.00 | 4.96 | 6.00 | 7.00 | 8.00 |
| wind(mps) | 0.00 | 1.00 | 2.73 | 3.00 | 4.00 | 12.00 |
| wind_direction(360) | 0.00 | 130.00 | 201.44 | 180.00 | 320.00 | 360.00 |
| rain(mm) | 0.00 | 0.00 | 0.19 | 0.00 | 0.04 | 14.25 |
| humidity | 11.00 | 53.00 | 68.24 | 69.00 | 85.00 | 100.00 |
| radiation(jcm2) | 0.00 | 0.00 | 59.11 | 1.00 | 101.00 | 366.00 |

## 3. FEATURE ENGINEERING

Lag features refer to data values at one or more points in time prior to the current point in time. By creating lag features, models can use information from the past to predict the future. Rolling statistics refer to statistics calculated in a sliding window, such as mean, standard deviation, minimum, maximum, etc. This is useful for dealing with noise and short-term fluctuations. For example, in this task, Lag48 features are created to capture the cyclical changes of the previous day and the past week. The mean and standard deviation of Rolling Window 14 are also calculated to smooth out short-term fluctuations, highlight the long-term trend, and reflect the cyclical changes of the week.

Feature selection methods can identify the most important features for model prediction, remove redundant or unimportant features, and improve model performance and efficiency. In this section, Lasso methods were used. Lasso applies L1 regularization to constrain the sum of absolute values of regression coefficients to achieve the purpose of feature selection. Finally, 36 features were selected, as shown in the table 2:

Table 2: The selected features

| Lagging features | Rolling features | | Time features |
|---|---|---|---|
| | Rolling mean | Rolling variance | |
| price_act_lag48<br>electricity_demand_lag48<br>solar_generation_lag48<br>wind_generation_lag48<br>tempc_lag48<br>cloud8_lag48<br>windmps_lag48<br>wdir_lag48<br>rainmm_lag48<br>radjcm2_lag48<br>humid_lag48 | price_act_rolling_mean<br>electricity_demand_rolling_mean<br>solar_generation_rolling_mean<br>wind_generation_rolling_mean<br>tempc_rolling_mean<br>cloud8_rolling_mean<br>windmps_rolling_mean<br>wdir_rolling_mean<br>rainmm_rolling_mean<br>humid_rolling_mean<br>radjcm2_rolling_mean | price_act_rolling_std<br>electricity_demand_rolling_std<br>solar_generation_rolling_std<br>wind_generation_rolling_std<br>tempc_rolling_std<br>cloud8_rolling_std<br>wdir_rolling_std<br>humid_rolling_std<br>radjcm2_rolling_std | month<br>quarter<br>day_of_week<br>hour<br>minute |

## 4. MODEL SELECTIONS

Five models for time series analysis were developed during the training process. A lower RMSE indicates better performance in forecasting prices, so the model with the lowest RMSE will be selected. XGBoost performed the best and has been chosen for use on the test data. The other models that were tried will also be explained in the following section.

### 4.1 Informer

The Informer model was configured with specific parameters, including the input length (48 half-hour intervals), output length (1 interval), layers, and other hyperparameters to capture complex temporal patterns. The model was trained on the transformed training set and validated on the testing set. Only variables are highly correlation were considered (absolute correlations): electricity_demand (0.33), solar_generation (-0.14), temperature(C) (-0.13), radiation(jcm2) (-0.12). Predictions were made for the test set and recursively for future time steps to forecast electricity prices for the next three hours. The model's performance was evaluated using the Root Mean Squared Error (RMSE), and the results (3.105) demonstrated the model's capability to accurately predict short-term electricity prices.

### 4.2 XGboost

This project attempts to use XGBoost regression model to forecast Tokyo electricity price. First, the data file that has been processed with missing values is read and the date and time columns are converted into a uniform datetime format and set as indexes. We then use feature engineering to create multiple lagging and rolling features, such as lag1, lag7, rolling mean, and rolling standard deviation, while generating time features, including month, quarter, day of the week, hour, and minute. To simplify the model and prevent overfitting, we used Lasso regression model for feature selection. After feature selection, we divided the data set into a training set and a validation set, used the training set to train the XGBoost model, evaluated the model performance on the validation set, and measured the model's prediction accuracy by calculating the root mean square error (RMSE).

Finally, we apply the trained model on an independent test set and calculate RMSE on the test set to verify the model's generalization ability.

### 4.3 SARIMA

Firstly, we entified the seasonal component ` (P, D, Q, s) ` as ` (1, 1, 0, 24) ` based on the observed data pattern, indicating a yearly seasonality with no seasonal differencing and a periodicity of 24 hours.

Secondly, we determined the non-seasonal parameters ` (p, d, q) ` as ` (2, 1, 3) ` through iterative model fitting and evaluation. Starting with initial guesses and leveraging automated tools, we adjusted these parameters to minimize AIC and BIC values, ensuring the model's simplicity while capturing essential data dynamics.

Lastly, after fitting the SARIMA model with these parameters to the training data, we conducted diagnostic checks. This involved analysing residual plots and autocorrelation functions to validate the model's assumptions and confirm its adequacy in capturing the underlying time series patterns effectively.

### 4.4 GRU

In GRU model, initially data is loaded and pre-processed which includes parsing datetime, filtering out infinite values, scaling features and target variables using Robust Scaler. Lagged features and expanding window statistics are generated.

Mutual information (MI) regression is applied to select top features which are then split into training and test dataset. GRU (Gated Recurrent Unit) model is defined using a sequential model with one GRU layer containing 50 units and a ReLU activation function followed by Dense layer with a single output unit. The model is compiled with Adam optimizer and Mean Squared Error (MSE) loss function. It is trained for 7 epochs with a batch size of 32, using reshaped data input data.

GRU model is trained using selected features. Its performance is evaluated using Root Mean Squared Error (RMSE). Results are then plotted to compare the actual and predicted prices.

### 4.5 Random Forest Regression

In the random forest model, Streamlit was used to build the interactive interface. Regarding the model ontology, firstly, rolling statistical features with seasonal features were used and normalized, then feature selection was performed using ExtraTreesRegressor to select the most important features. And the hyperparameters of RandomForestRegressor are optimized using GridSearchCV before training the model. Finally, the optimal model is trained using the training set.

After the model training is completed, the trained model is used to predict the validation and test sets and the root mean square error (RMSE) is calculated. Then cross-validation is performed using TimeSeriesSplit to assess the stability of the model. After all metrics are calculated, predictions are made using the trained model. Future predictions are displayed (allowing the user to choose their own prediction time) with a download option (in csv format).

It also allows the user to upload the actual price data file, compare it with the prediction result and calculate the RMSE between the prediction result and the actual price to be displayed on the page. Also present the actual gap in a line graph (RMSE=4.6412)

## 5. EVALUATION

Table 3: Compare different models through RMSE based on June 2024 data

| Model | RMSE score |
|---|---|
| Informer | 3.105 |
| **XGboost** | **1.438** |
| SARIMA | 4.021 |
| GRU | 4.380 |
| Random Forest Regressor | 4.641 |

The table above shows that the XGBoost model has the lowest RMSE of 1.438 on the datasets. Therefore, the XGBoost model is recommended for predicting electricity prices.

## 6. CHALLENGES

### 6.1 Missing Data and Feature Synchronization

One of the primary challenges was managing missing data within the datasets. It was essential to address these gaps accurately to maintain the integrity of the time series analysis. Additionally, synchronizing features across different datasets was crucial to ensure consistency and reliability in the predictions.

### 6.2 Interpreting and Balancing Feature Importance

Another challenge was interpreting the importance of various features and balancing them effectively. This was necessary to enhance predictive accuracy while avoiding overfitting or underfitting the model. It required careful analysis and fine-tuning to determine which features most significantly impacted the electricity price predictions.

## 7. CONCLUSION

In summary, this report delved into predicting Tokyo's electricity prices using a variety of time series models applied to comprehensive data. Through meticulous data preprocessing and

feature engineering, including lag features and rolling statistics, the dataset was prepared for model training. The models evaluated, such as Informer, XGBoost, SARIMA, GRU, and Random Forest Regression, were assessed based on their Root Mean Squared Error (RMSE). Among these, the XGBoost model demonstrated exceptional accuracy with an RMSE based on June data of 1.4377, showcasing its effectiveness in forecasting electricity prices.

Challenges encountered included managing missing data and optimizing feature importance, highlighting areas for future refinement to meet the dynamic demands of Tokyo's electricity market.