Forecasting hourly power demand with Amazon SageMaker and AutoGluon

Machine Learning Engineer Nanodegree Capstone Report

April 14th, 2023

Elaborated by:

Javier Castillo Peña

I. Definition

Project Overview

In Peru, electricity bill for free users includes mainly charges for generation, transmission, distribution, and other regulated fees. Free users are large consumers (with a maximum annual demand over 200 kW) which contract directly with a power generator the price of their electricity, while the prices for transmission and distribution are established by the national regulator.

Transmission charges may represent about 20% of the monthly electricity bill of a free user. This toll fee for using the transmission system is determined by a regulated price multiplied by the user's power demand (in kW), also known as “coincident peak demand”, at the time of the monthly national electric system's peak demand measured in 15-minute periods. This methodology by the national regulator aims to incentivize the reduction of consumption during peak hours from 5:00 pm to 11:00 pm.

In order to optimize their energy bill, free users should improve their consumption behavior by reducing their “coincident peak demand”. Besides, other “peak shaving” strategies include using locally produced electricity or energy battery storage to reduce demand from the electricity grid. In any case, predicting national power demand can help to apply those strategies to reduce “coincident peak demand” and the respective transmission charges.

Although national demand data is published by the independent system operator (COES), no open-source tool to predict national power demand is available in the market. The aim of this project is to build a machine learning model to forecast Peru’s power demand to help consumers to optimize their “coincident peak demand” and reduce their electricity bill.

Problem Statement

The problem to be solved in this project is to forecast hourly power’s demand of the Peruvian national electric system (SEIN) using time series historical data. By predicting power demand, a free user (large consumer) can make better decisions such as optimizing their “coincident peak demand” at the time of the monthly national electric system's peak demand from 5:00 pm to 11:00 pm and, therefore, reduce their transmission charges that can represent about 20% of their electricity bill.

To solve the problem, a machine learning model is built to forecast one-day ahead hourly power demand using historical data, Amazon SageMaker[1] platform and the AutoGluon[2] framework which performs advanced data processing, deep learning, and multi-layer model ensemble methods.

The project is developed following the next steps:

1. Data extraction, transformation, and load.
2. Exploratory data analysis (EDA).
3. Building of a benchmark model.
4. Model training with raw data using AutoGluon.
5. Feature engineering.
6. Model training using AutoGluon with new features (time-varying covariates).
7. Benchmark of models.

Metrics

Based on the characteristics of the forecasting problem, the performance of the model is measured using the RMSE metric, also known as RMSD[3]. The objective is to build a machine learning model with the lowest RMSE that helps free users to make decisions on reducing their "coincident peak demand" on peak hours from 5:00 pm to 11:00 pm.

$$\mathrm{RMSD} = \sqrt{\frac{\sum_{t=1}^{n} (\hat{y}_t - y_t)^2}{n}}.$$

Fig. 1 RMSE formula. Source: Wikipedia

[1] https://aws.amazon.com/sagemaker
[2] https://auto.gluon.ai/stable/index.html
[3] https://en.wikipedia.org/wiki/Root-mean-square_deviation

II. Analysis

Data Exploration

Power demand data of the national electric system (SEIN) is publicly available by the national system operator (COES) and includes the following:

1. "Demand": Power demand of the national electric system (SEIN) in MW registered in 15-minute periods and reported monthly. This data is collected from meters and used to determine the monthly peak demand.
   Source: https://www.coes.org.pe/Portal/portalinformacion/demanda?indicador=maxima

2. "SEIN demand": Power demand of the national electric system (SEIN) in MW registered in 30-minute periods and reported daily the next day of operations. Data collected from SCADA systems.
   Source: https://www.coes.org.pe/Portal/PostOperacion/Reportes/Ieod

3. "Total demand": Total power dispatch data published in 30-minute periods. This data collected from SCADA systems includes power dispatch from COES and non-COES generators.
   Source: https://www.coes.org.pe/Portal/portalinformacion/demanda

Figure 2 shows the differences among "demand", "SEIN demand", and "total demand" in MW.

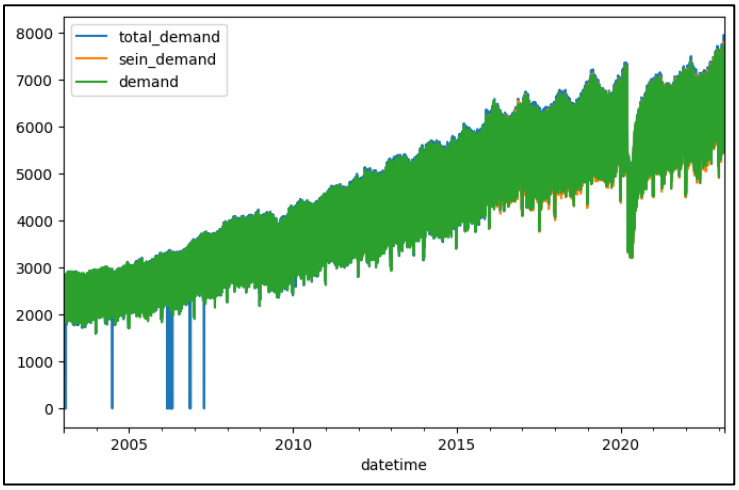

Fig. 2 Comparison of demand data.

Although monthly meter data ("demand") is used to determine peak demand, to forecast one-day ahead hourly demand, data collected from SCADA systems ("SEIN demand") was chosen because it is available daily. Despite a low deviation of 70.1592 MW (calculated in terms of RMSE) between meter and SCADA measurements, the selection of the monthly peak day can still be affected. As shown in Figure 3, 60% of monthly peak days from January 2016 to February 2023 were the same when evaluated using “SEIN demand” data.

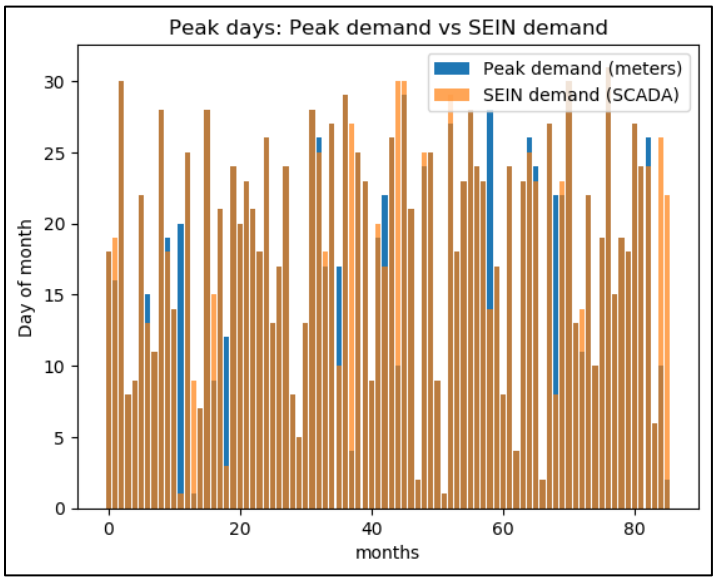

Fig. 3 Monthly peak demand days from January 2016 to February 2023.

The original demand data is published in Excel reports for each day of operation; therefore, some scripts to download, extract, transform, and load the data into a single dataset were developed. The merged dataset contains the following fields:

1. **'datetime'**: Date and time in "yyyy-mm-dd hh:mm:ss" format and 30-minute periods.
2. **'demand'**: Power demand of the national electric system (SEIN) in MW collected from meters. Data from January 2003 to February 2023.
3. **'sein_demand'**: Power demand of the national electric system (SEIN) in MW collected from SCADA systems. Data from January 2016 to February 2023.

Exploratory Visualization

The power demand of the national electric system was analyzed as it is shown in the following figures. First, Figure 4 shows the distribution of power demand, while in Figure 5 and Figure 6, it can be observed

that power demand has been increasing since 2003, however, it was drastically reduced during COVID lockdown in 2020.

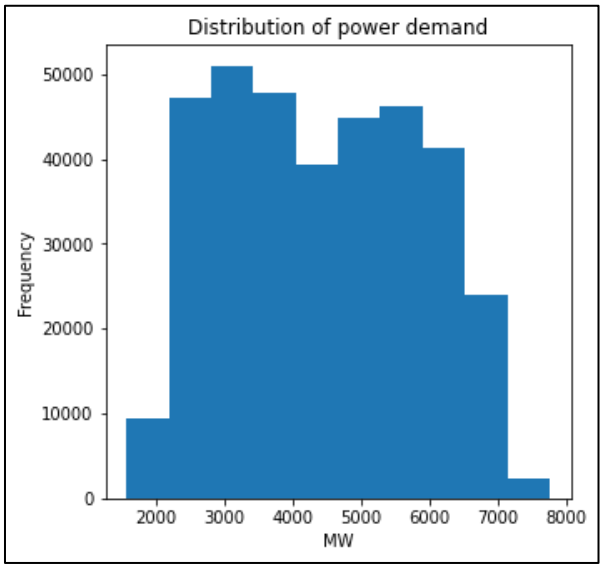

Fig. 4 Histogram for power demand in MW.

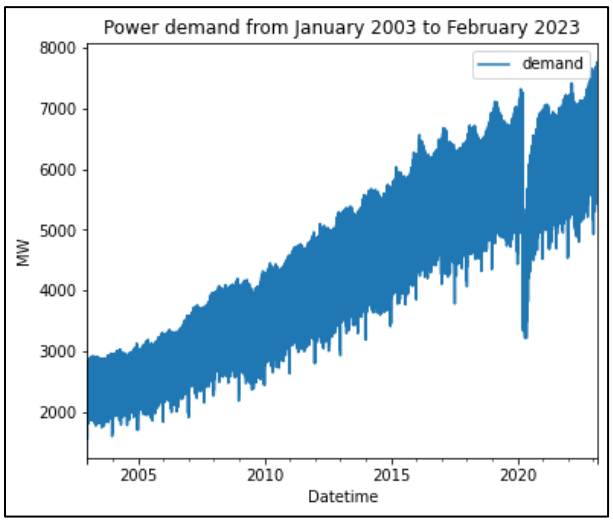

Fig. 5 Power demand from January 2003 to February 2023 in MW.

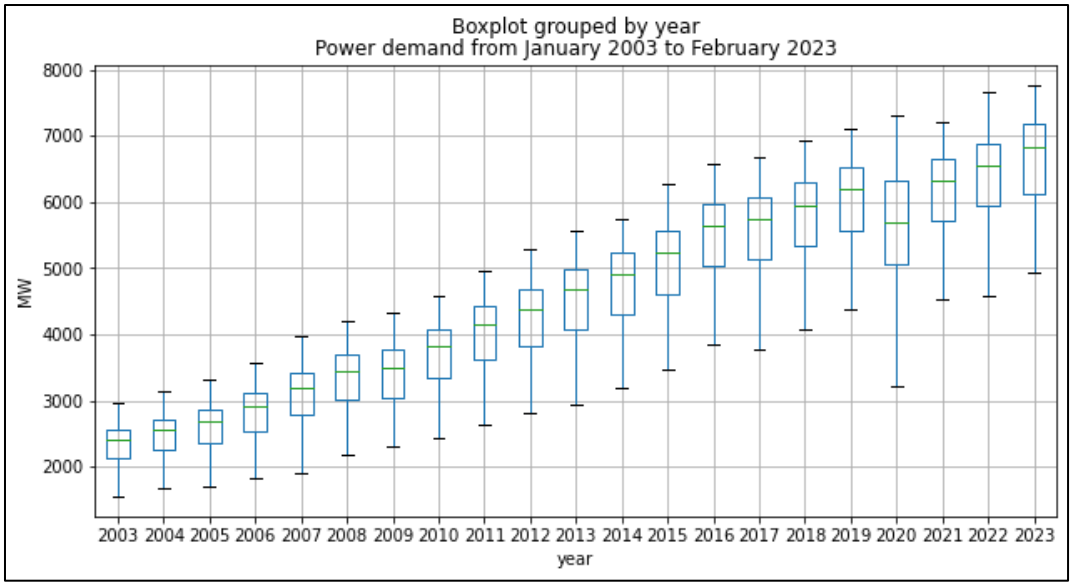

Fig. 6 Power demand by year from January 2003 to February 2023 in MW.

Figure 7 shows the behavior of power demand during the year, where higher demand is required during spring and summer periods from September to March. Besides, as it was expected, demand decreases on weekend days (Saturdays and Sundays), as shown in Figure 8.

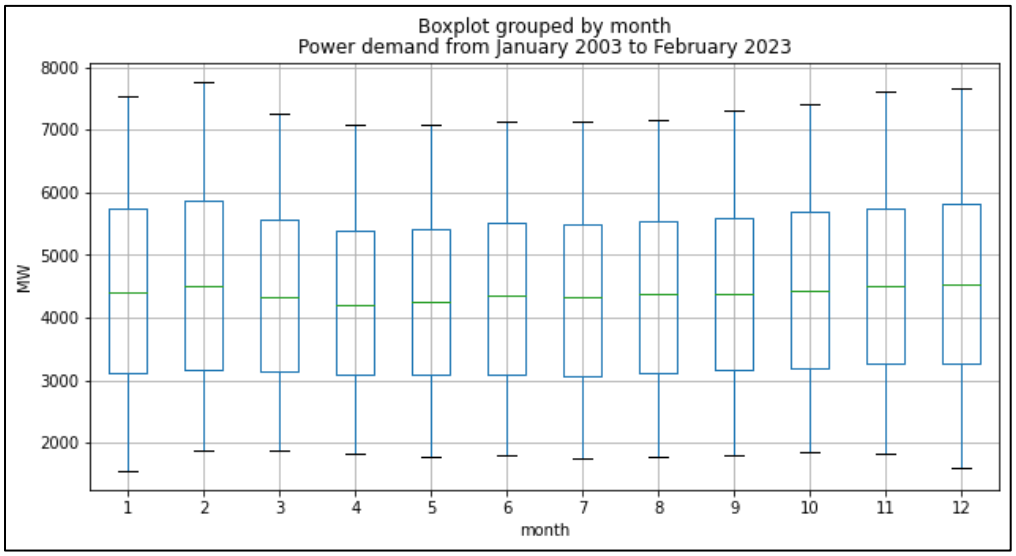

Fig. 7 Power demand by month in MW.

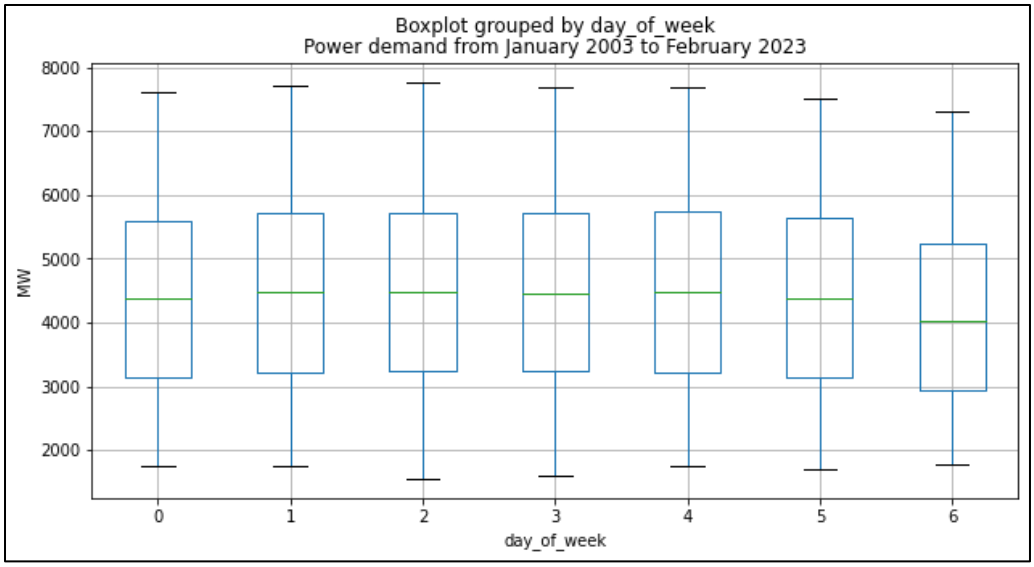

Fig. 8 Power demand by day of the week in MW.

When exploring data during a day in Figure 9, it can be observed how the demand changes during the day and increases on peak hours.

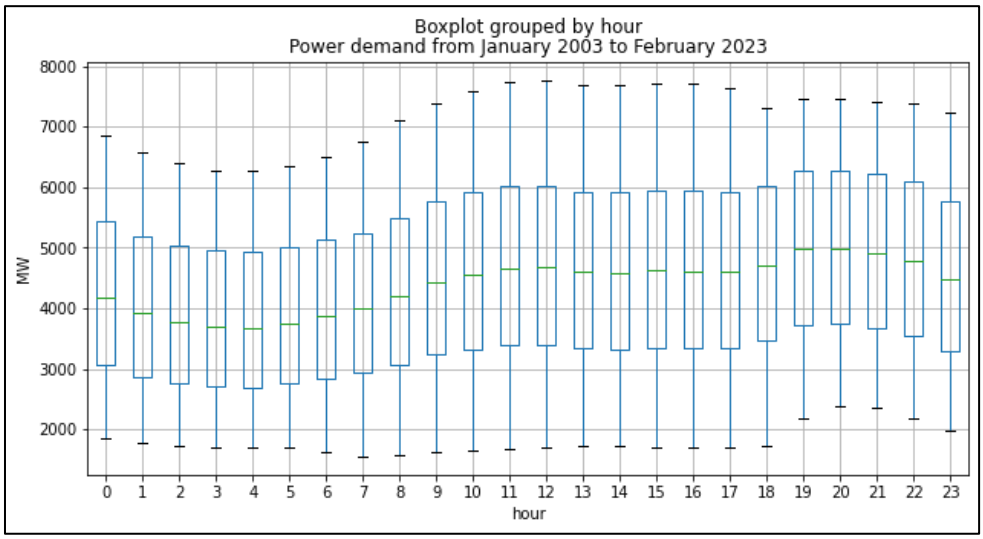

Fig. 9 Power demand by hour in MW.

The behavior of power demand on peak hours was also analyzed, in order to observe historical data patterns. Figure 10 shows that peak demand occurs more frequently in the last week of the month, on

Wednesday, and at 19:00 hours. However, peak demand has also occasionally occurred on Saturday, before 18:00 hours or after 21:00 hours.

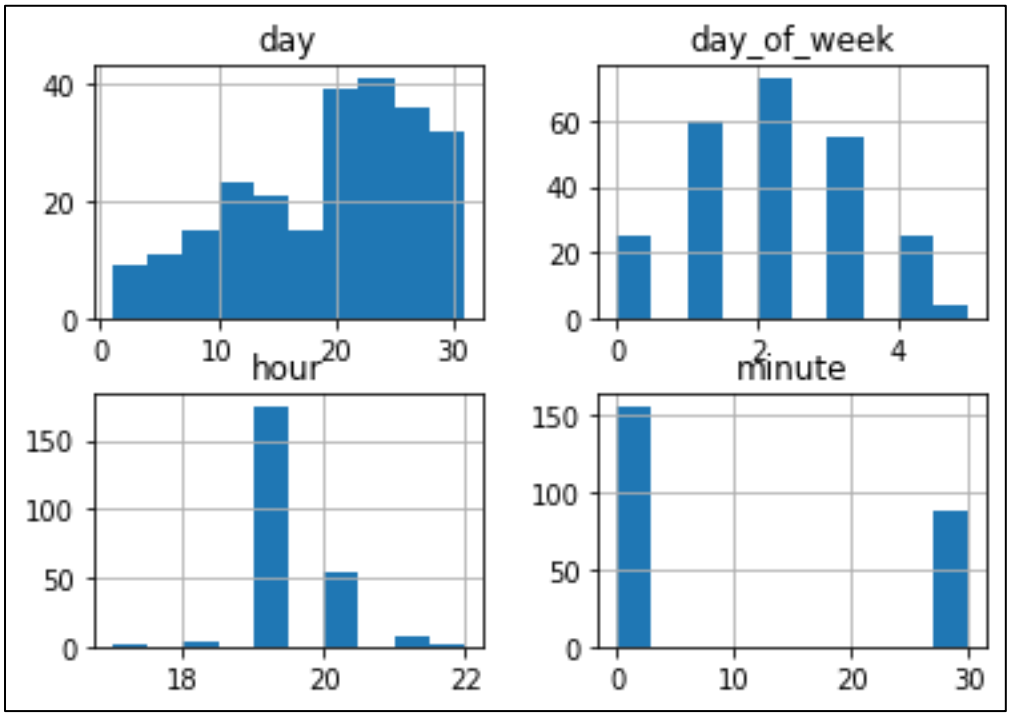

Fig. 10 Histograms of power demand during peak hours.

Algorithms and Techniques

Accurately forecasting power demand can be a challenging task that requires a combination of machine learning skills and high-quality tools. This project leverages the features provided by AutoGluon, which is a machine learning library designed to develop high-performing models.

AutoGluon TimeSeries is a module of AutoGluon that is specifically designed for time series forecasting tasks. AutoGluon TimeSeries automates the process of developing accurate and robust time series models. It provides a range of features, including automatic feature engineering, hyperparameter tuning, and ensembling, to optimize model performance.

The algorithms available to fit with AutoGluon[4] include simple baselines (Naive, SeasonalNaive), statistical models (ARIMA, ETS, Theta), tree-based models XGBoost, LightGBM and CatBoost wrapped by

[4] https://auto.gluon.ai/stable/tutorials/timeseries/forecasting-quick-start.html

AutoGluonTabular, a deep learning model DeepAR, and a weighted ensemble combining that combines predictions of these models.

The following steps describe the process to train a model with AutoGluon TimeSeries:

1. Data preparation
2. Loading time series data as a **TimeSeriesDataFrame**
3. Training time series models with **TimeSeriesPredictor.fit**
4. Generating forecasts with **TimeSeriesPredictor.predict**
5. Evaluating the performance of different models

Benchmark

A simple naïve model is used as a benchmark model, where the most recent known value is used as the predicted value at the same time the next day. In this manner, the previous day power demand at a specific time is used as the predicted value for the next day demand at the same time. In order to compare the performance of the benchmark model and the machine learning models, a RMSE of 337.7162 MW was calculated on the same test dataset. The aim of this project is to surpass the results obtained by the naïve model. Figure 11 and Figure 12 show predictions of the benchmark model for a one-month and 6-month periods, respectively.

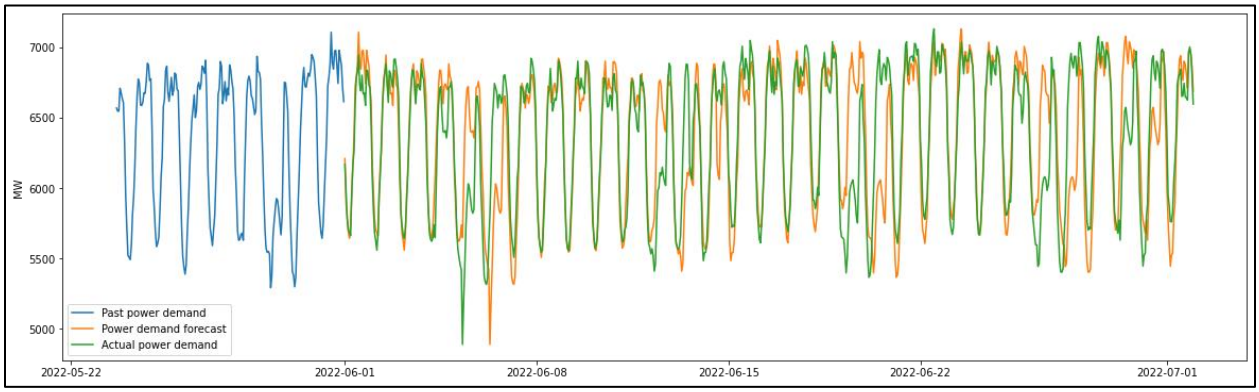

Fig. 11 Predictions of benchmark model for June 2022.

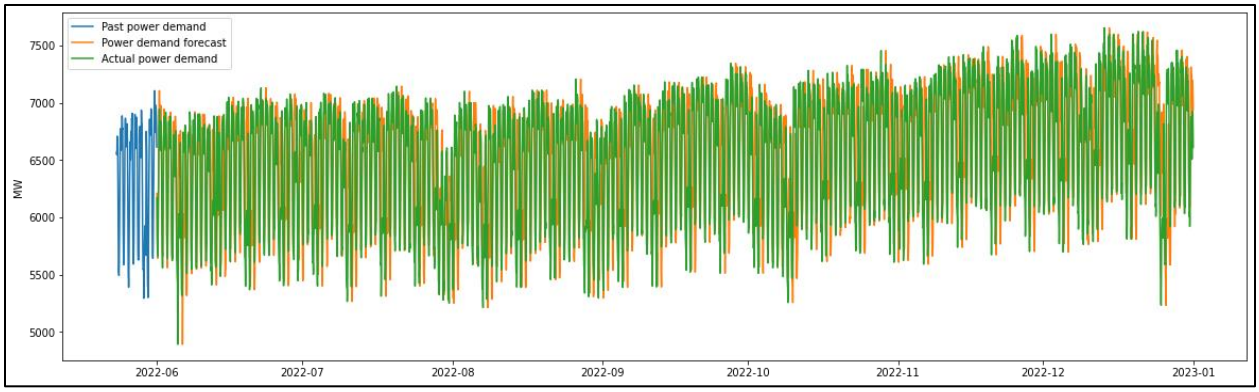

Fig. 12 Predictions of benchmark model from June to December 2022.

III. Methodology

Data Preprocessing

As described above, the original time series data contains data of power demand registered in 30-minute periods. The preprocessing process included data cleaning and feature engineering. In order to enhance the machine learning model, feature engineering played a key role to increase power demand forecasting accuracy.

Some missing values were detected for three days of power demand data ("SEIN demand") on 2019-07-31, 2020-11-04, and 2020-12-25. Two strategies were analyzed to replace missing data, either the mean between values from the previous and the next week or the mean between values from the previous and the next year. For instance, missing demand on 2019-07-31 and 2020-11-04 were replaced with the mean between values from the previous and the next week, as shown in Figure 13 and Figure 14.

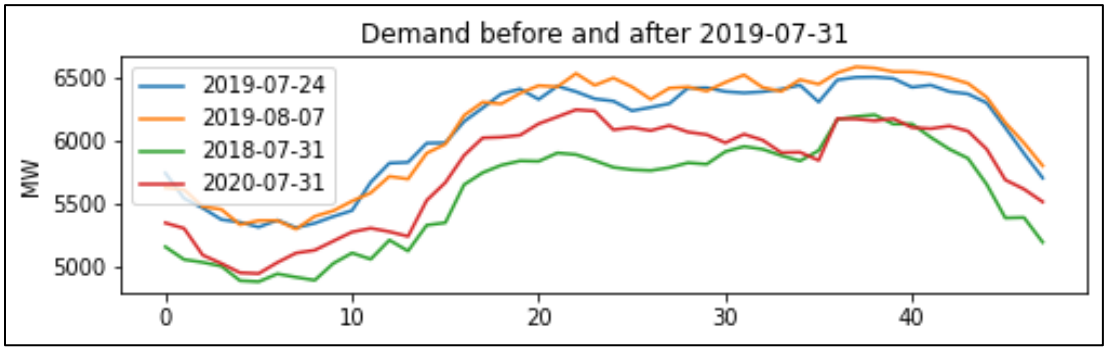

Fig. 13 Demand data to replace missing values on 2019-07-31.

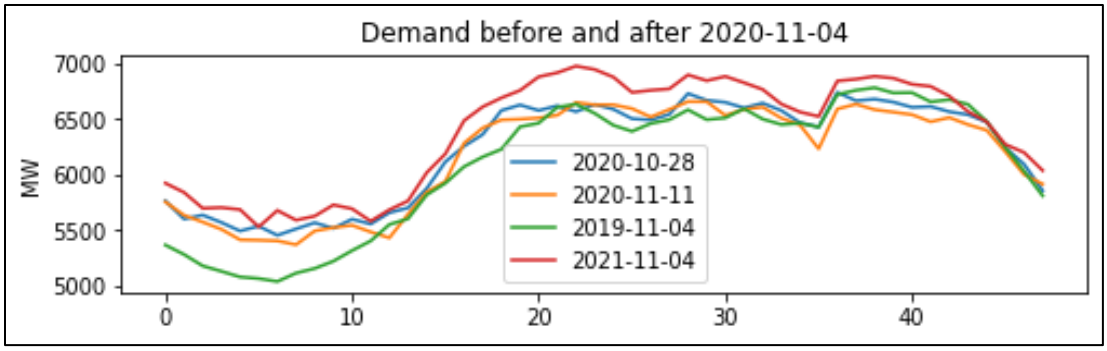

Fig. 14 Demand data to replace missing values on 2020-11-04.

Besides, in the case of missing demand on 2020-12-25, it was replaced with the mean between demand values on the same date in 2019 and 2021, as shown in Figure 15.

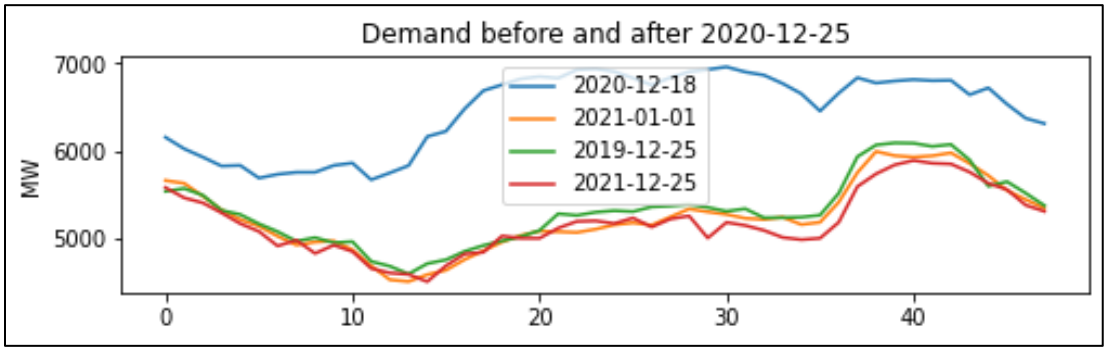

Fig. 15 Demand data to replace missing values on 2020-12-25.

Feature engineering is the process of selecting and transforming raw data into features that can be used to improve machine learning models' accuracy and generalization. First, AutoGluon recognizes intrinsic features from date and time, such as year, month, day, and hour. Besides, the following features were incorporated into the training data:

1. **'workday'** - Working day: Boolean feature that is True if date is a working day from Monday to Friday.
2. **'holiday'** - Holyday: Boolean feature that is True if date is a public holiday.
3. **'on_peak'** - Peak hours: Boolean feature that is True if power demand occurs during peak hours from Monday to Saturday, from 5:00 pm to 11:00 pm, excluding holidays.

4. **'season'** - Season: This feature classifies demand for 'summer', 'fall', 'winter', and 'spring'. These categories were transformed into dummy variables.
5. **'time_of_day'** - Time of the day: This feature classifies demand during 'night', 'morning', 'noon', and 'evening'. These categories were transformed into dummy variables.
6. **'on_covid'** - COVID period: It was observed a significant decrease in power demand during COVID lockdown in 2020. Therefore, a boolean feature was introduced for the instances between March and September 2020. FIG XX shows the behavior of power demand during that period.

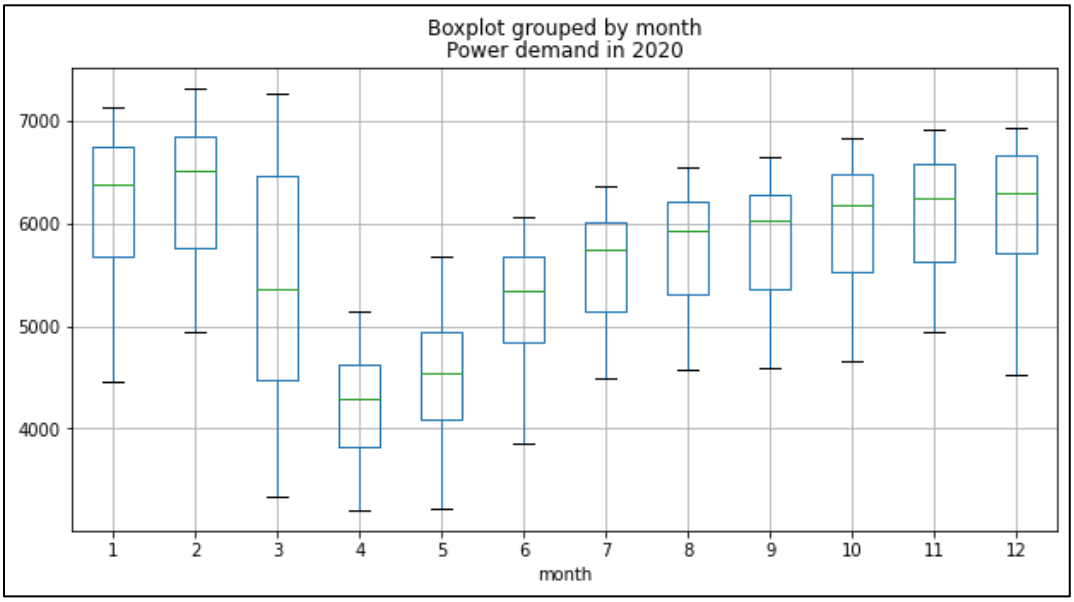

Fig. 16 Power demand during COVID lockdown in 2020.

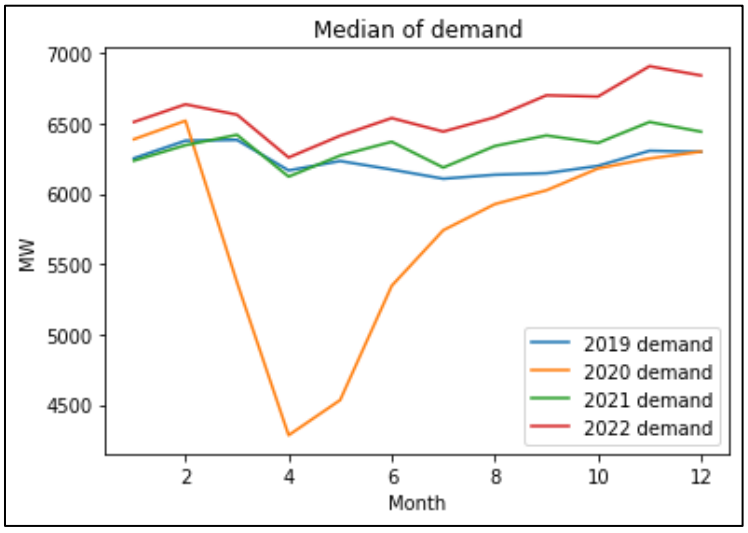

Fig. 17 Comparison of power demand before and after COVID lockdown.

Implementation

The implementation process to develop a machine learning model with AutoGluon TimeSeries followed 5 stages: data preparation, data loading, training, prediction, and evaluation. The stages are described as follows:

1. **Data preparation.**

   The dataframe was split to use 80% of data for training and 20% for testing. The train dataset contains data from 2020-01-01 to 2022-05-31, while the test data from 2022-06-01 to 2022-12-31.

   As required by AutoGluon, each row of the data frame should contain a single observation ('datetime') of a single time series represented by:

   - unique ID of the time series ('item_id') as int or str. A single value "D1" was set for the entire dataset.
   - timestamp of the observation ('datetime') as a pandas.Timestamp or compatible format.
   - numeric value of the time series target ('sein_demand')

   A first model was fit only with the following features:

   - 'item_id', 'datetime'

   On the other hand, a second model was trained using the following time-varying covariates, in addition to the previous features:

   - 'workday', 'on_peak', 'holiday', 'on_covid', 'season_summer', 'season_fall','season_winter', 'season_spring', 'time_of_day_night', 'time_of_day_morning', 'time_of_day_noon', 'time_of_day_evening'

2. **Loading time series data as a TimeSeriesDataFrame.**

   Autogluon Timeseries requires the class TimeSeriesDataFrame to store the time series dataset, as follows:

```
train_data = TimeSeriesDataFrame.from_data_frame(
    train_data,
    id_column="item_id",
    timestamp_column="datetime"
)
test_data = TimeSeriesDataFrame.from_data_frame(
    test_data,
    id_column="item_id",
    timestamp_column="datetime"
)
```

3. Training time series models with TimeSeriesPredictor.fit.

AutogGluon requires a TimeSeriesPredictor object to be created to forecast future values of the time series. The target to forecast is stored in the column 'sein_demand' of the TimeSeriesDataFrame.

The forecast horizon is set with the prediction_length parameter. In this project, the dataset contains time series measured at 30-minute periods, so the prediction_length was set to 48 to train models that forecast up to 24 hours into the future.

Besides setting the path to save trained models, it was specified that AutoGluon should rank models according to the RMSE metric.

```
predictor = TimeSeriesPredictor(
    prediction_length=prediction_length,
    path="./AutogluonModels/ag-raw-1h-sein-demand",
    target="sein_demand",
    eval_metric="RMSE",
)
```

To fit the model, it was used the "medium_quality" presets and the training time was limited to 20 minutes (1200 seconds). The medium_quality presets define the models to fit such as simple baselines (Naive, SeasonalNaive), statistical models (ARIMA, ETS, Theta), tree-based models XGBoost, LightGBM and CatBoost wrapped by AutoGluonTabular, a deep learning model DeepAR, and a weighted ensemble combining these. By default, AutoGluon sets the validation set as the last prediction_length timesteps in train_data.

```
predictor.fit(
    train_data,
    presets="medium_quality",
    time_limit=1200,
)
```

4. Generating forecasts with TimeSeriesPredictor.predict.

Once the model was trained, the fitted TimeSeriesPredictorcan be used to forecast the future time series values. By default, AutoGluon makes forecasts using the model that had the best score. The forecast includes predictions for the next prediction_length timesteps, starting from the end of the time series in train_data. For instance, as train_ data includes demand from 2020-01-01 to 2022-05-31, AutoGluon will predict 48 timesteps i.e power demand on 2022-06-01 in 30-minute periods.

```
predictions = predictor.predict(train_data)
```

5. Evaluating the performance of different models.

The machine learning models were used to make forecast on the test period from 2022-06-01 to 2022-12-31 (5136 predictions). Finally. the models were evaluated using the RMSE metric.

Refinement

As mentioned in the Implementation section, in the first run, it was used only the 'datetime' feature. In the second run, in order to improve the performance of the model, covariates (time-varying features) were introduced since they may influence the target ('sein_demand'). The TimeSeriesPredictor was updated as follows for the second run:

```
known_covariates_names = ['workday', 'on_peak',
                          'holiday', 'on_covid', 'season_summer', 'season_fall',
                          'season_winter', 'season_spring', 'time_of_day_night',
                          'time_of_day_morning', 'time_of_day_noon', 'time_of_day_evening']


predictor = TimeSeriesPredictor(
    prediction_length=prediction_length,
    path="./AutogluonModels/ag-new-features-1h-sein-demand",
    target="sein_demand",
    eval_metric="RMSE",
    known_covariates_names=known_covariates_names,
)
```

IV. Results

Model Evaluation and Validation

The performance of the machine learning models was evaluated on validation and test data. The validation score in the first run was 89.923456 MW (RMSE), while the second model obtained a RMSE of 77.705239 MW. In both cases, it can be observed that the model with best performance was an ensemble model ("WeightedEnsemble"). Figure 18 and Figure 19 show the validation scores of different trained models in the first and second run, respectively.

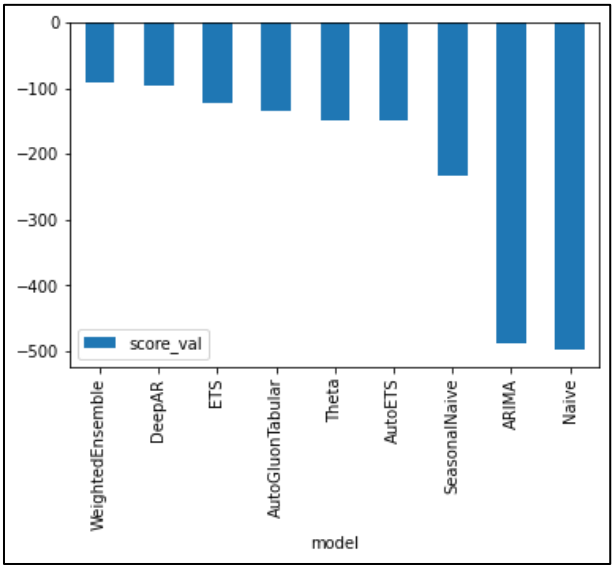

Fig. 18 Validation scores for the trained models in the first run.

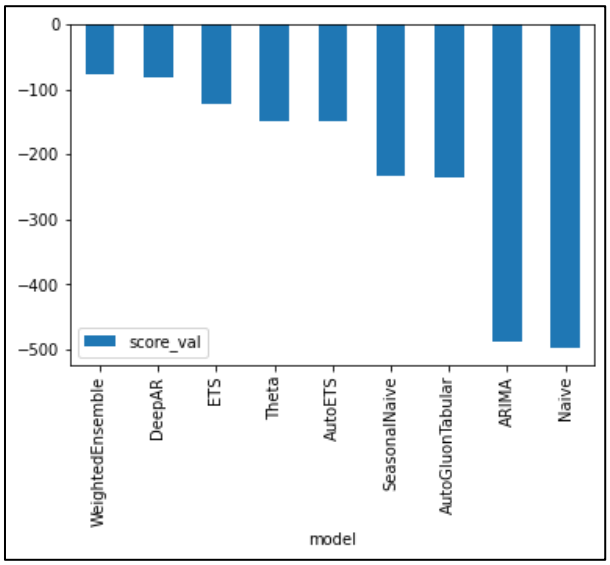

Fig. 19 Validation scores for the trained models in the second run.

When testing both models, the results show that the first run model obtained a RMSE of 137.2041 MW, while the model with feature engineering in the second run returned a RMSE of 111.4882 MW. The following figure summarizes the test scores:

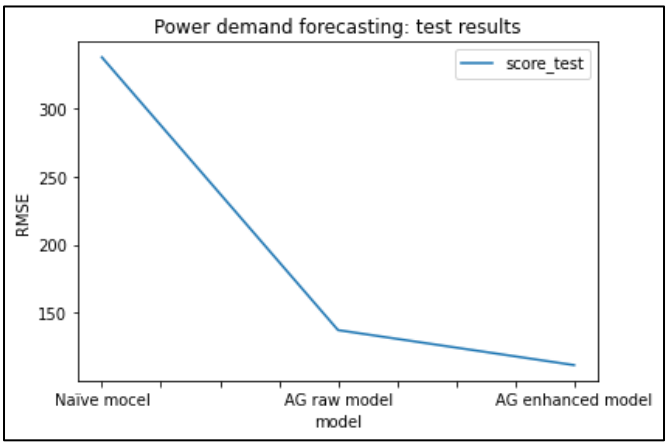

Fig. 20 Test score comparison for trained and benchmark models.

In addition, the following figures show the demand predictions of both models for June 2022.

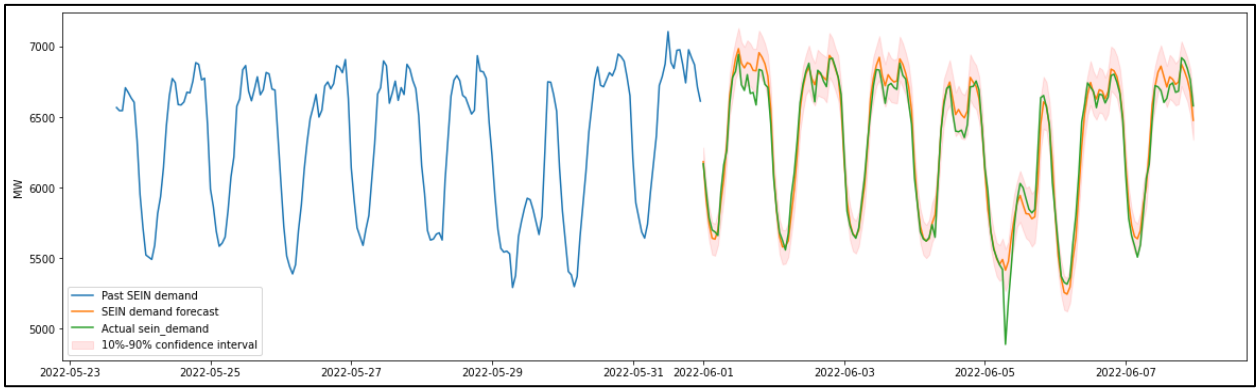

Fig. 21 Model 1: one-day ahead hourly demand forecasts for the first week of June 2022.

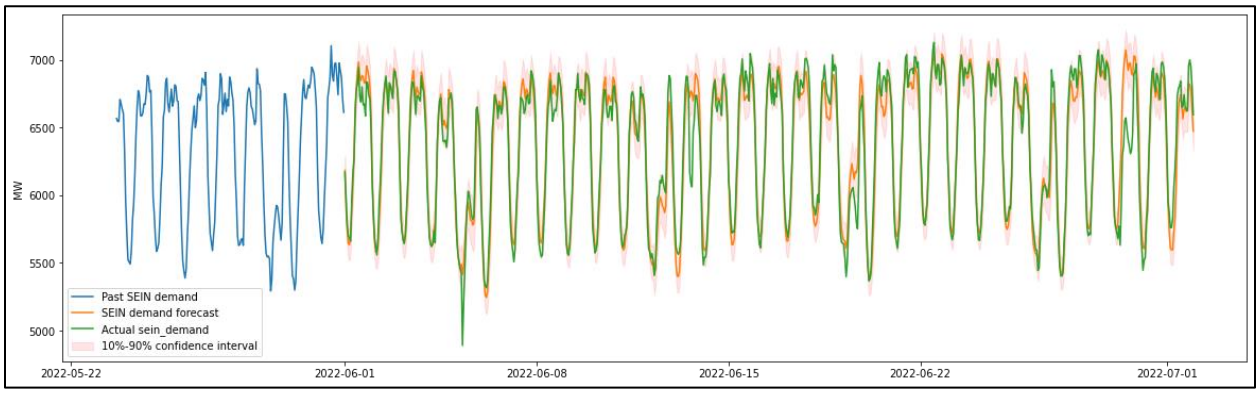

Fig. 22 Model 1: one-day ahead hourly demand forecasts for June 2022.

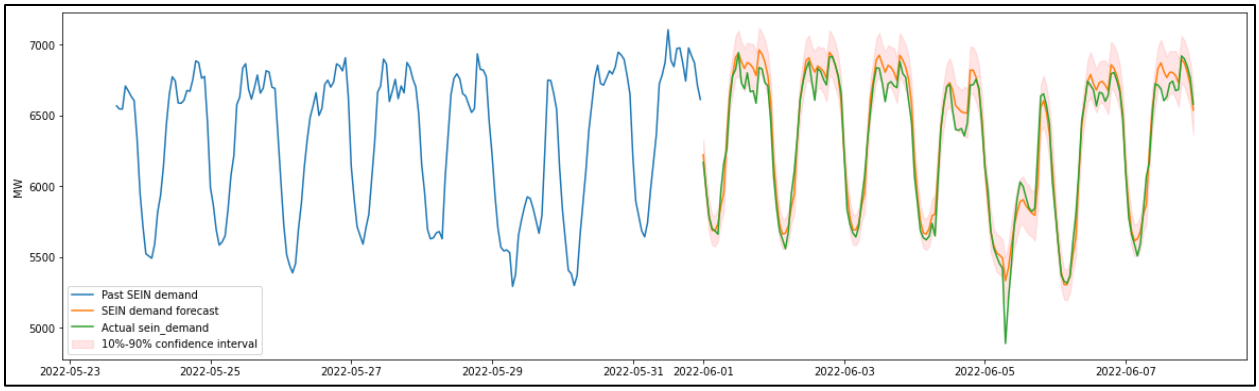

Fig. 23 Model 2: one-day ahead hourly demand forecasts for the first week of June 2022.

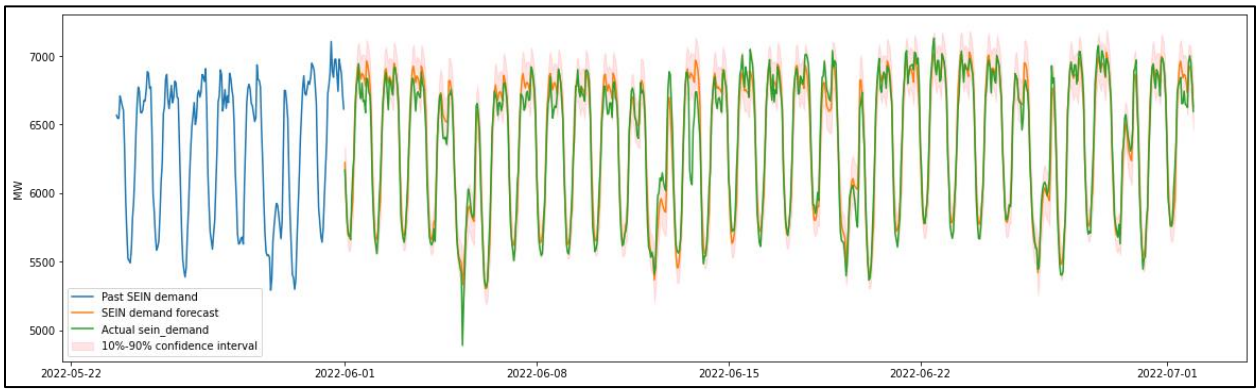

Fig. 24 Model 2: one-day ahead hourly demand forecasts for June 2022.

Justification

According to the results, the second model had the best performance when tested on 6-month data from June to December 2022. The RMSE obtained with this model is acceptable since the median of power demand was over 6,500 MW in 2022. Therefore, this model solves the problem by forecasting hourly power's demand of the Peruvian national electric system (SEIN) with significant accuracy.

The decision to choose the second model is justified based on its performance when tested on 6-month data from June to December 2022. The results indicate that the second model outperformed the other models and had the best performance on the test data. The evaluation metric used, RMSE, is a widely used measure to assess the accuracy of predictions, and the acceptable RMSE obtained by this model is a good indication of its predictive power. It is also essential to compare the RMSE value with the scale of the target variable. In this case, the median of power demand was over 6,500 MW in 2022, and the RMSE indicates that the model can make predictions that are close to the actual values of the power demand.

The second model's performance on the test data suggests that it can accurately forecast the hourly power demand of the national electric system with significant accuracy and solves the problem statement.

V. Conclusion

Predictions on peak hours

The machine learning model built with SageMaker and AuoGluon TimeSeries can predict one-day ahead hourly power demand on peak hours. Figure 25 shows the predictions made by the model on peak hours from June to December 2022.

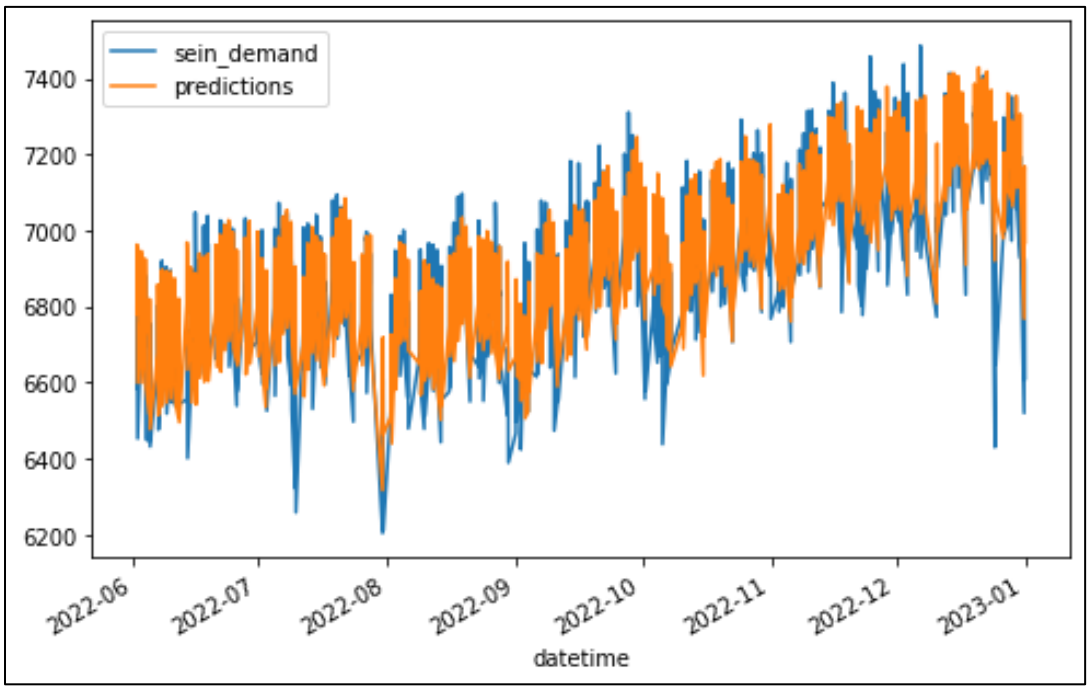

Fig. 25 One-day ahead hourly demand forecasts on peak hours.

Reflection

In this project, the main goal was to forecast hourly power demand of the national electric system using time series historical data to help free users make better decisions and reduce their transmission charges. The machine learning model was built using the AutoGluon framework on the Amazon SageMaker platform to forecast one-day ahead hourly power demand. The project followed a step-by-step approach from data extraction to model training and evaluation.

One aspect of the project that was particularly interesting was the feature engineering step. This is because it involved incorporating time-varying covariates into the model to improve its accuracy. This process shows the importance of selecting relevant features to improve the model's performance.

On the other hand, some aspects of the project that were challenging were the data extraction process and the exploratory data analysis (EDA). This is because it required significant time to download data from daily reports in Excel format, extract and transform relevant data, and finally load it into an AWS S3 bucket. Besides, it required a deep understanding of the data and the ability to extract meaningful insights that could guide the model development process. Nonetheless, data visualization techniques were leveraged to identify important patterns in the data that helped to build the model.

In conclusion, the project provided a comprehensive understanding of the end-to-end process involved in developing a machine learning model for a time series problem.

Improvement

One aspect of the current implementation that could be improved is the feature engineering step.

In this project, time-varying covariates were added to the model to improve its accuracy. One option to improve the model is to use an automated feature engineering process. This approach involves using algorithms to automatically generate features from the raw data. This can save time and improve the accuracy of the model by identifying complex relationships that may not be apparent through manual feature engineering.

In addition, incorporating new features from external factors such as weather data or economic indicators may influence in power demand.