Forecasting Locational Marginal Prices

By Leveraging A Combination of Machine Learning & Statistical Models

Kabitri Chattopadhyay, April 2025

Introduction

What is Locational Marginal Price (LMP)?

LMP represents the cost of delivering the next increment of electricity at a specific location in the grid.

Why forecast LMP?

It helps market participants to optimize bids, manage risk, and improve decision-making.

Project objective

Find a powerful combination of ML and statistical models to improve LMP forecasting.

No exogenous variables are used; the focus is solely on time-series dynamics and auto-correlations in historical LMP data to determine how much can be achieved without external inputs.

Data overview

LMP data from CAlifornia Independent System Operator (CAISO) from Jan 1, 2020, to March 3, 2025, covering 3 price zones: NP-15, SP-15, and ZP-26.

No exogenous variables, like weather data or grid data like load, congestion, generation mix, is used.

Data Preprocessing

- Handling missing data with interpolation
- Handling missing timestamp
- Outlier adjustment
- Feature selection based on correlation analysis

Model selection

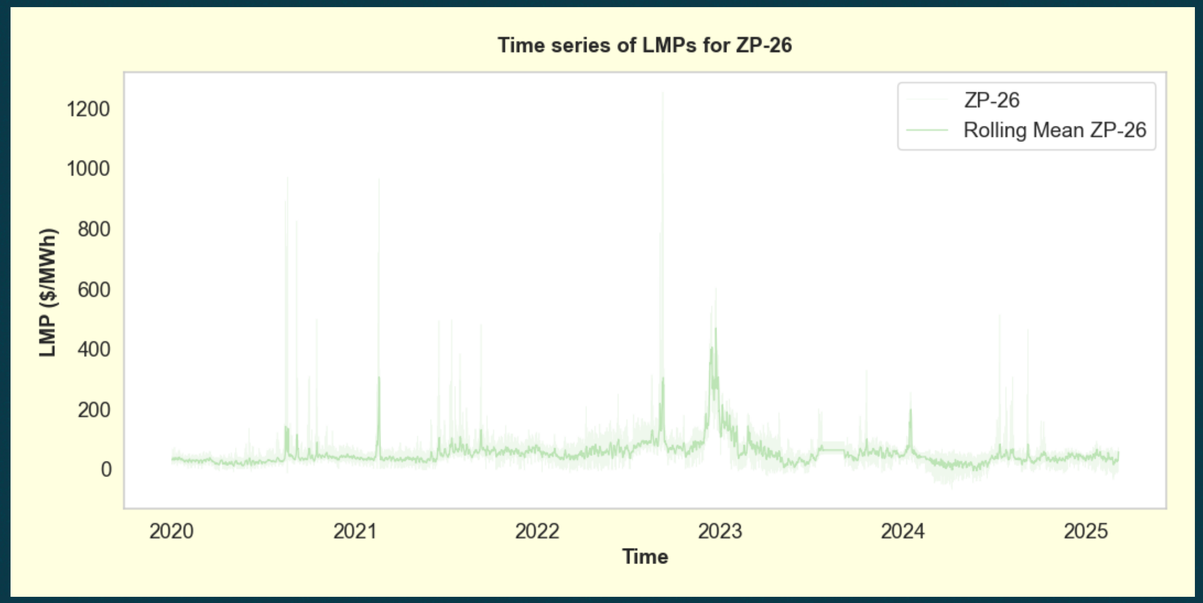
Machine Learning (ML) models

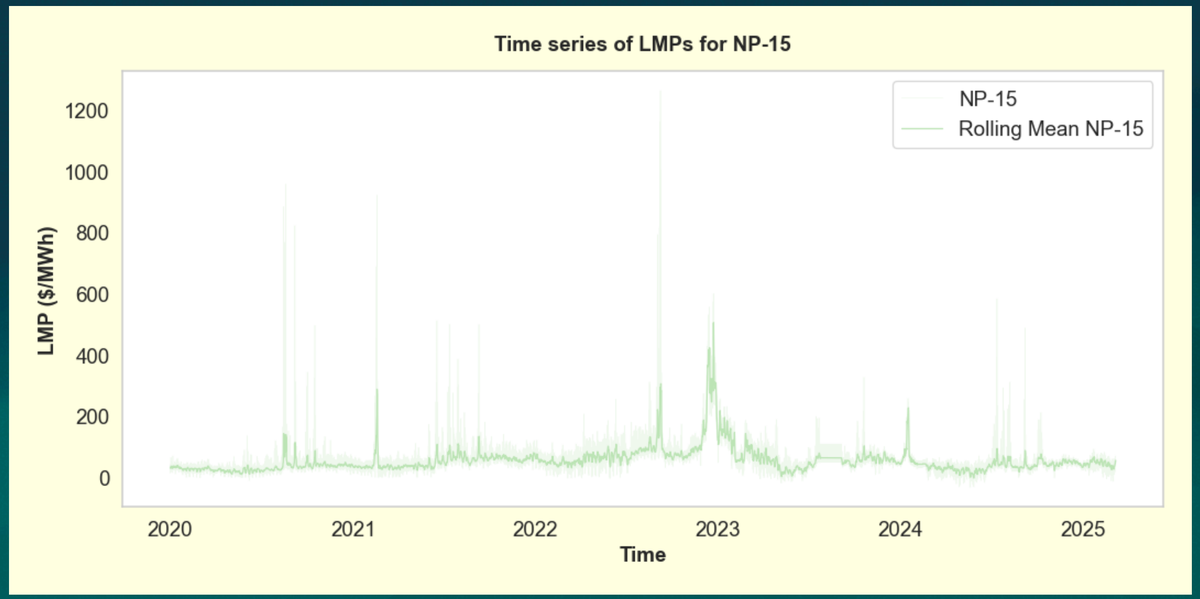
Two tree-based ML models, RandomForest and XGBoost, are implemented that use multiple decision trees to make predictions, combining their outputs to enhance accuracy and reduce overfitting. Both models have built-in cross-validation capabilities, allowing easy evaluation and hyper parameter tuning.

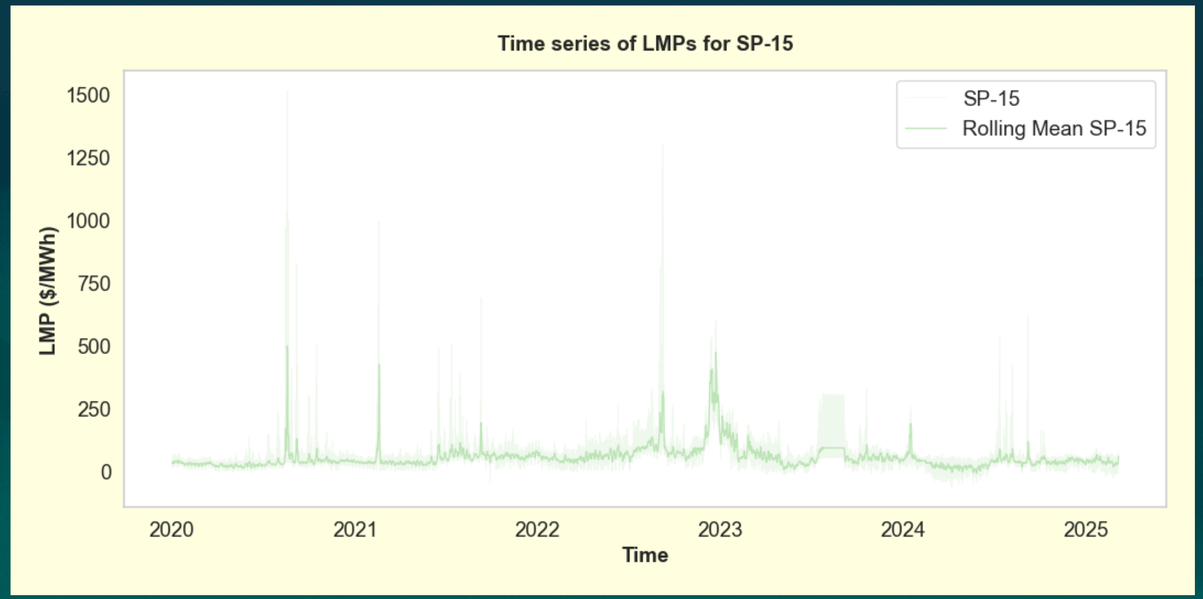
Statistical models

ARIMA (AutoRegressive Integrated Moving Average) is a statistical model describing autocorrelation in time-series data. GARCH (Generalized Autoregressive Conditional Heteroskedasticity) is a model used to describe the volatility clustering in time-series data. Both models are combined to capture volatility in LMP data.

LMP time-series





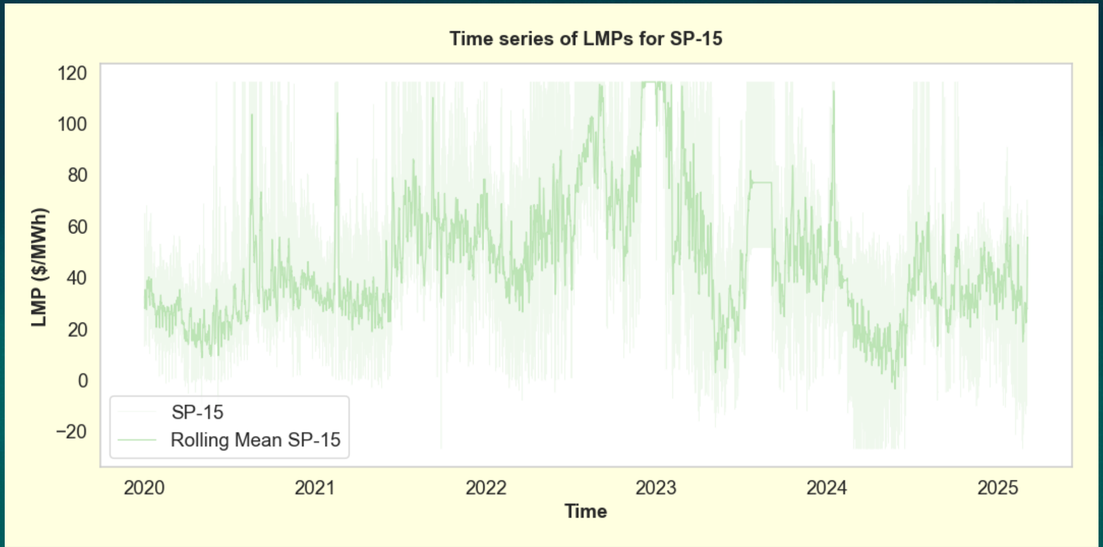


Outlier adjustment

Only extreme outliers adjusted without distorting overall data distribution. Two approaches tried.

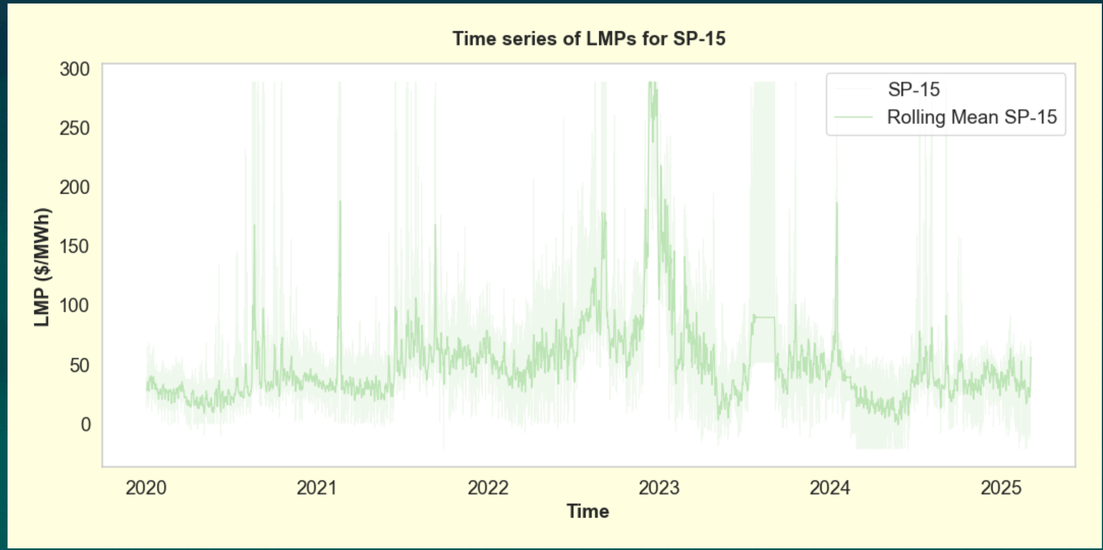
Inter-quartile Range (IQR)

Outlier detection using first (Q1) and third (Q3) quartiles, with outliers falling outside the range of Q1 - 1.5 IQR and Q3 + 1.5 IQR. No data loss occurs.

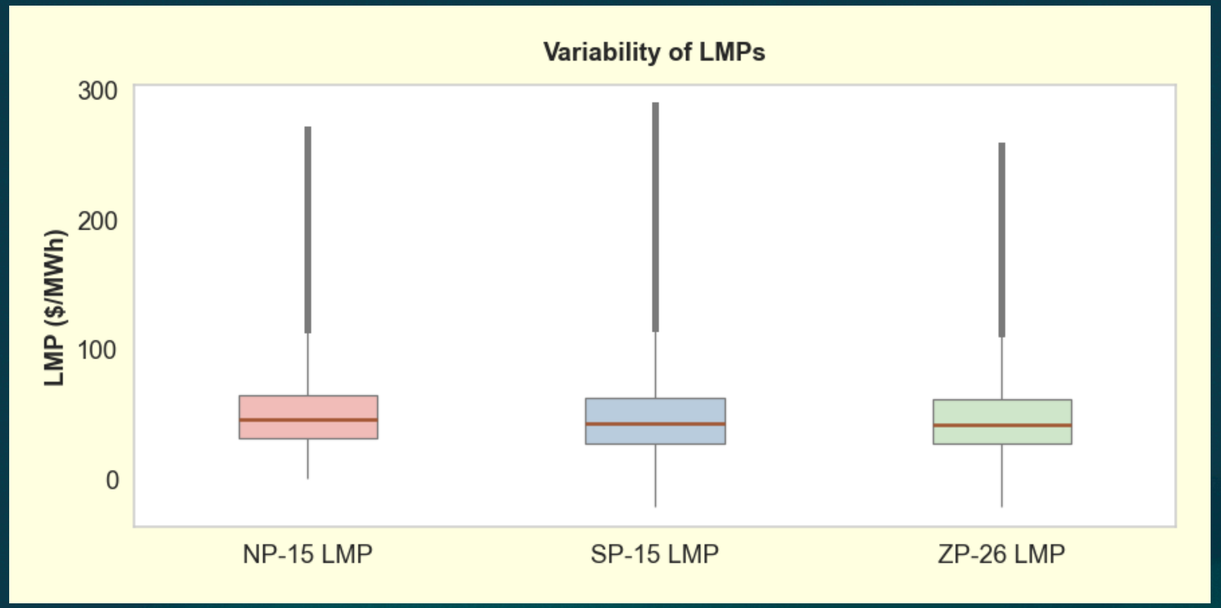


Winsorization

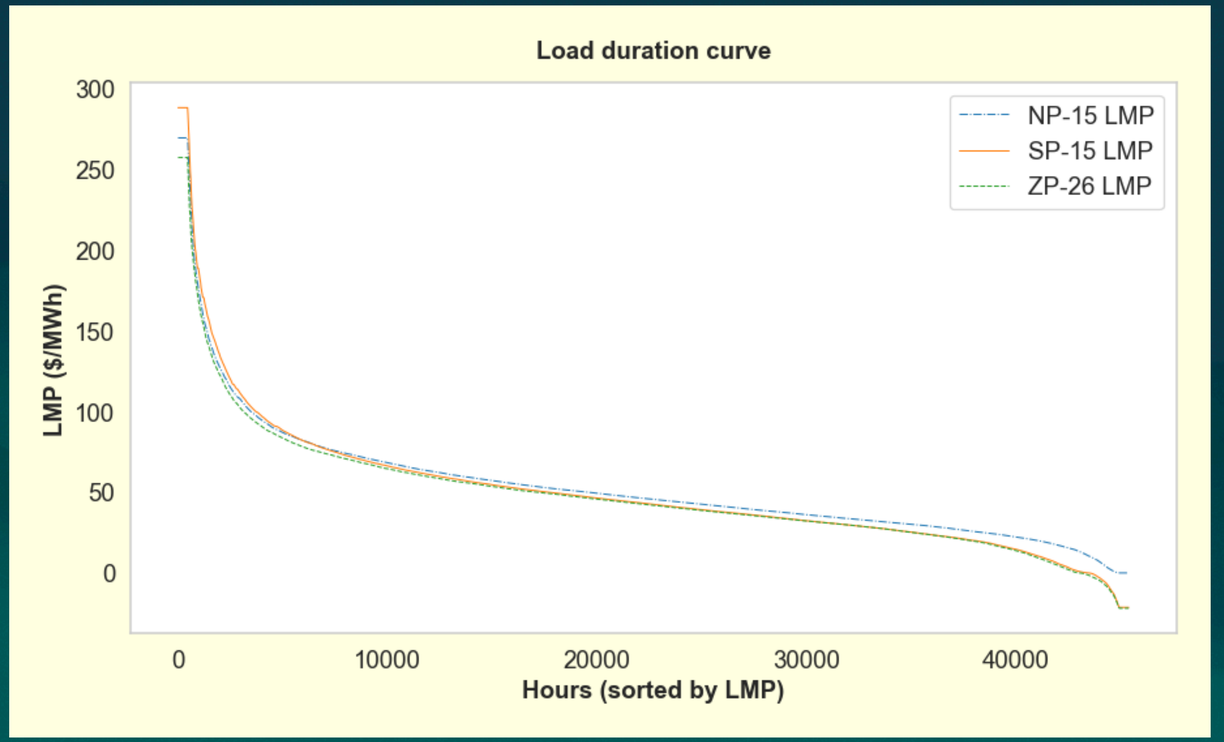
Replacing extreme values with nearest percentile-based threshold, usually 1st and 99th percentiles are used. No data loss occurs.



Variability analysis



Winsorization is applied prior to data analysis to ensure extreme values do not skew model training, allowing for a more accurate LMP forecast, better feature representation, and enhanced figure clarity.



Volatility in the data set

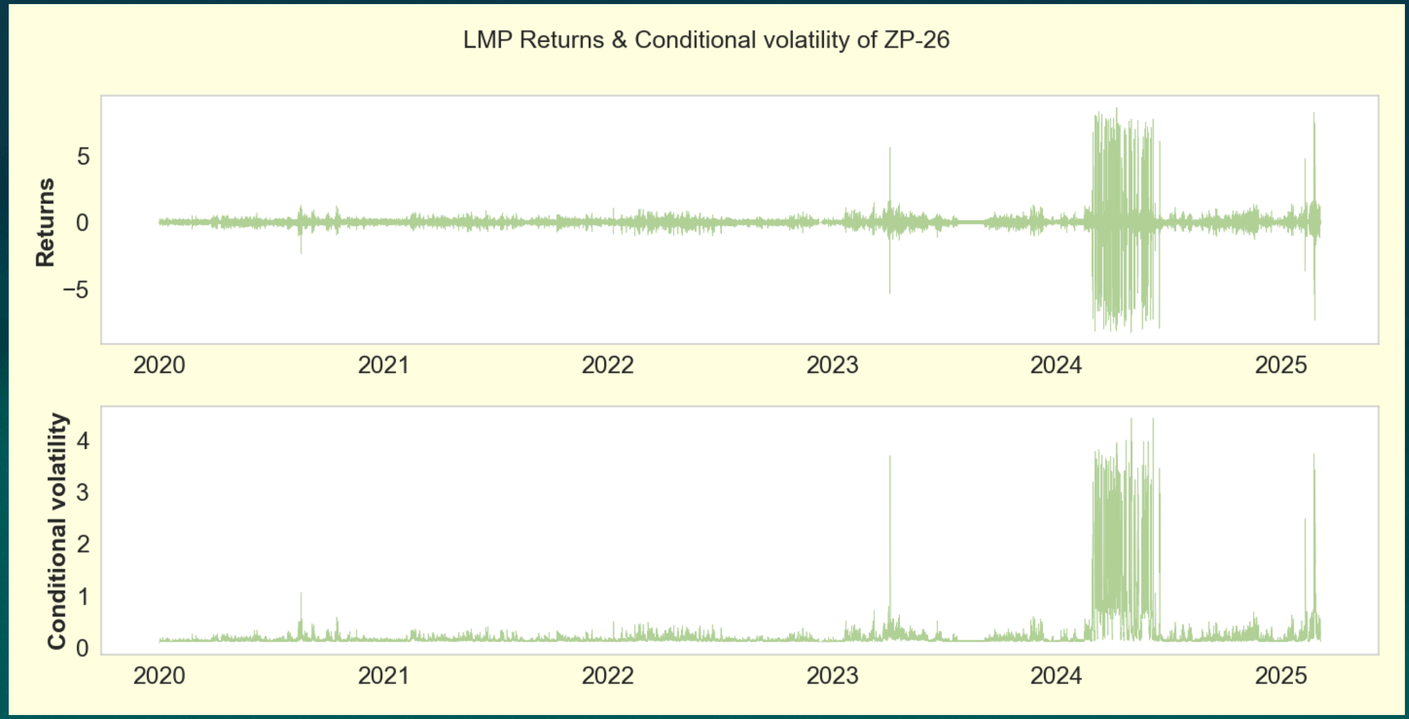
To assess and quantify the volatility in the data, LMP returns (r) are analyzed.

$$r = ln\left(\frac{P'_t}{P'_{t-1}}\right) \text{ with } P'_t = P_t + |min(P)| + \epsilon$$

The volatility of the LMP time-series follows its returns very closely and it is conditioned to the past values of itself and model errors - hence conditional volatility.

Exhibits key characteristics of volatility:

1. *Mean reversion*
2. *Volatility clustering*



Model implementation

Baseline model 👇

Simple RandomForest

Simple XGBoost

→

RandomForest with enhanced features

XGBoost with enhanced features

→

RandomForest with volatility

XGBoost with volatility

↓

Ensemble (combined) model

↓

Stacked (Ridge-fit combined) model

→

Stacked model with volatility

👆 Final model

Baseline models

A simple implementation of RandomForest and XGBoost with the following temporal features:

- hour (for diurnality), month (for seasonality), day-of-week (weekday-weekend effects)
- lag (for auto-correlation effect), rolling mean (over 24 hours)

Implementation & parameterization

```
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
gb_model = XGBRegressor(n_estimators=100, learning_rate=0.1, random_state=42)
```

Models with enhanced features

The simple models are further improved by introducing more features showcasing LMP temporal variability, with the following additional features:

- sine and cosine functions of hour and month
- rolling standard deviations (over 24 hours)

Hyperparameter tuning

```
# Hyperparameter Tuning for RandomForest
    param_grid_rf = {
        'n_estimators': [100, 200],
        'max_depth': [5,10]
    }
```

```
# Hyperparameter Tuning for XGBoost
    param_grid_gb = {
        'n_estimators': [100, 200],
        'learning_rate': [0.01, 0.1],
        'max_depth': [5,10]
    }
```

Models with volatility

Due to the inherent volatile nature of LMP, the data is further analyzed with statistical models, where, LMP log-returns are first used in ARIMA model, its residue to fed to the GARCH model to obtain conditional (time-varying) volatility, which is passed onto the ML models. New features include:

- ARIMA model residuals
- conditional volatility

Implementation

```
arima_model = ARIMA(ret, order=best_arima_order).fit()
arima = arima_model.resid.values

garch_model = arch_model(arima, mean='Zero', vol='Garch',
                         p=best_garch_order[0], q=best_garch_order[1]).fit(disp='off')

vol = garch_model.conditional_volatility
```

Combined models

Instead of using RandomForest and XGBoost separately, they are combined to yield results with the best of two:

Ensemble model

Prediction results from both RandomForest & XGBoost are combined using weighting factors derived from previous performance.

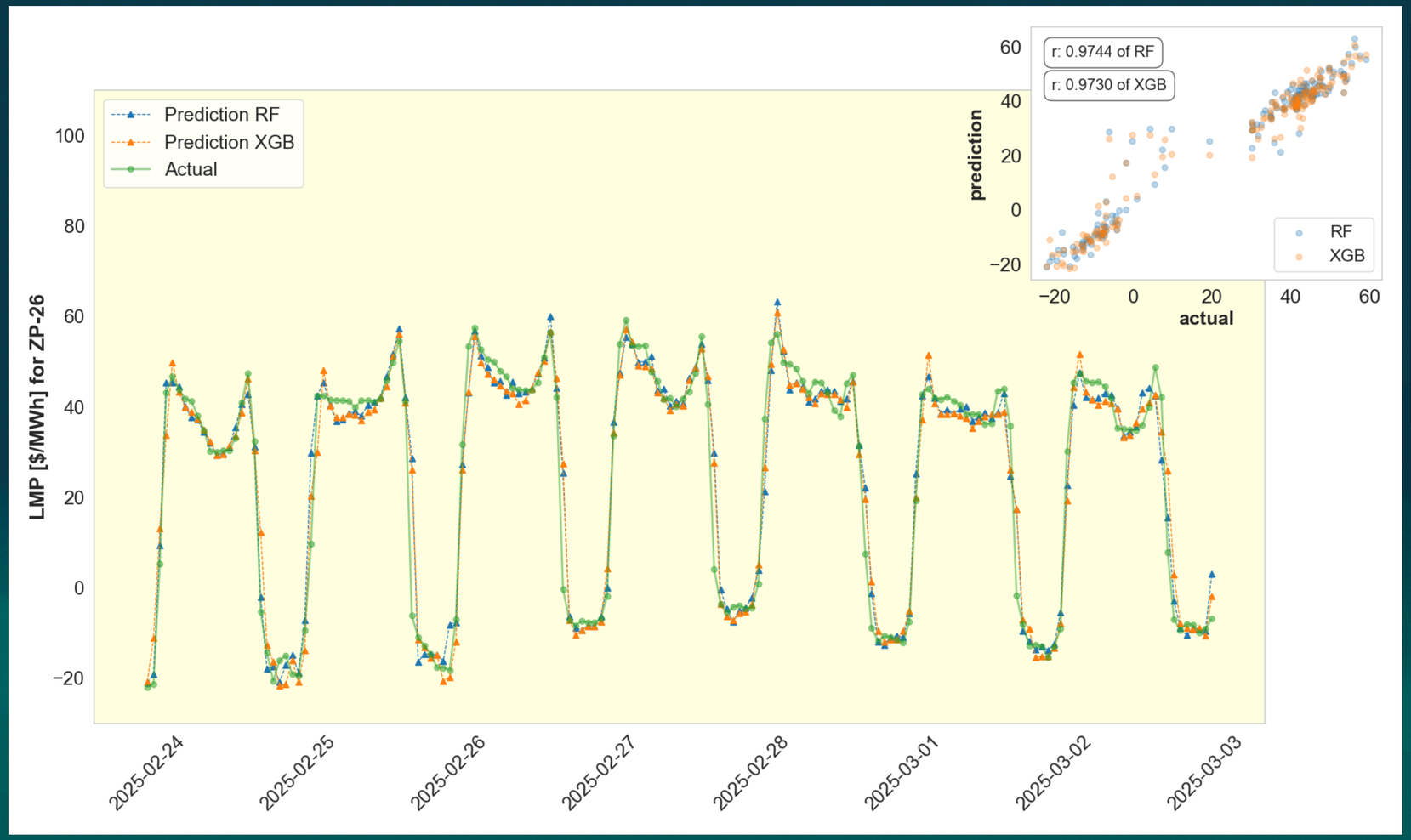
```
pred = (wf_rf * rf_pred) + (wf_gb * gb_pred)
```

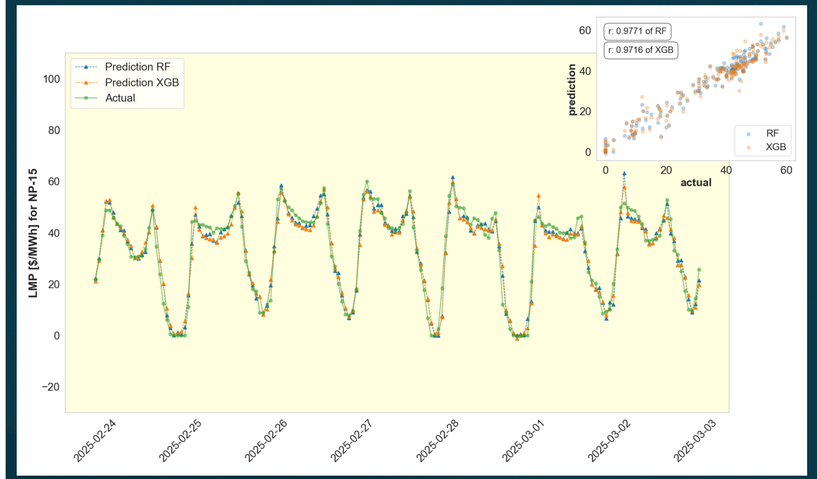
Stacked model

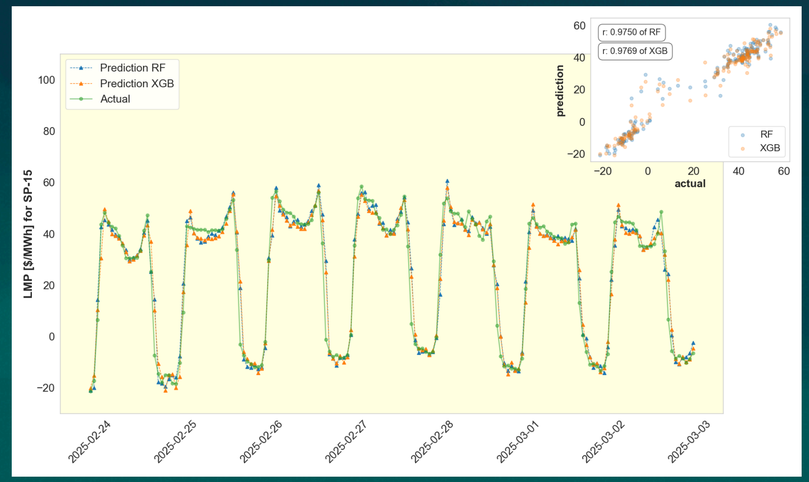
Predictions from the Random Forest and XGBoost models were used as meta-features and a Ridge regression model was then trained on them to learn the optimal linear combination for improved final forecasts.

```
meta_features.append([rf_pred, gb_pred])
meta_X = np.array(meta_features)
meta_y = np.array(actual_values)
ridge = Ridge(alpha=1.0)
ridge.fit(meta_X, meta_y)
stacked_prediction = ridge.predict(meta_X)
```

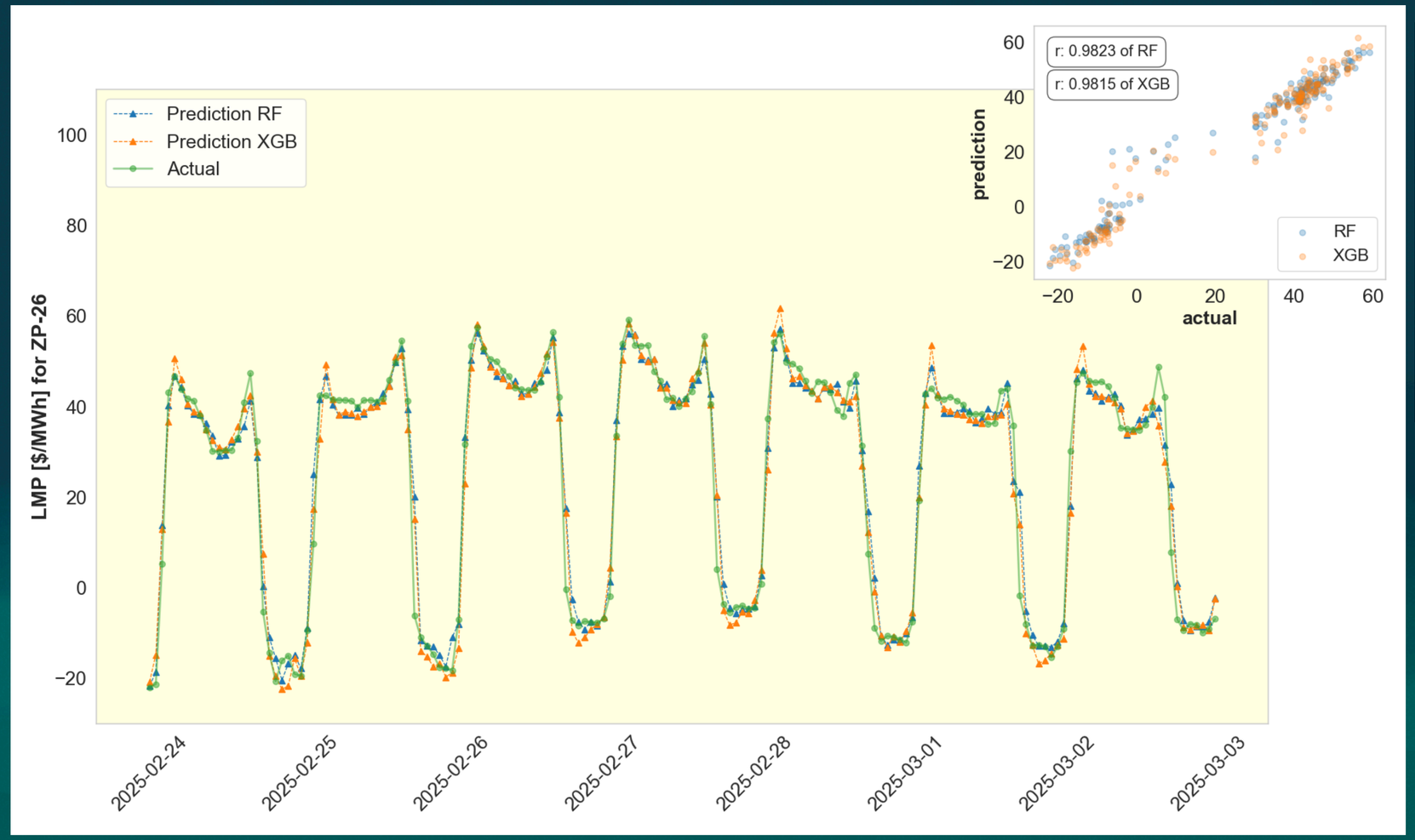
Simple models: Baseline

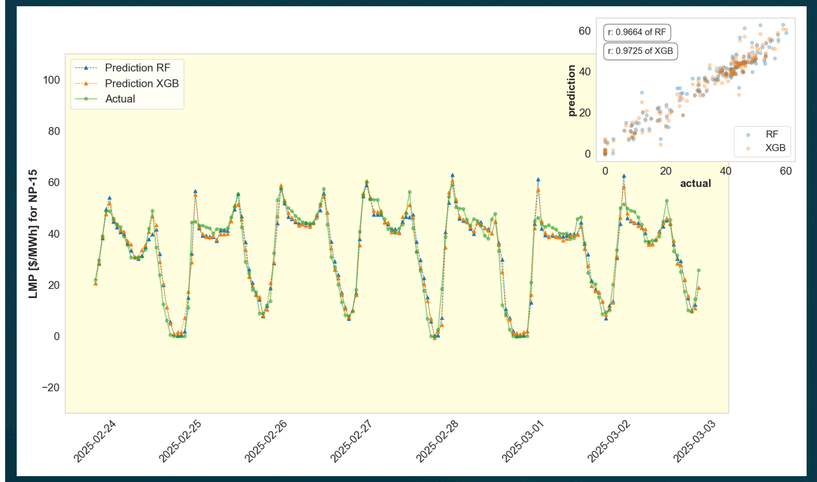


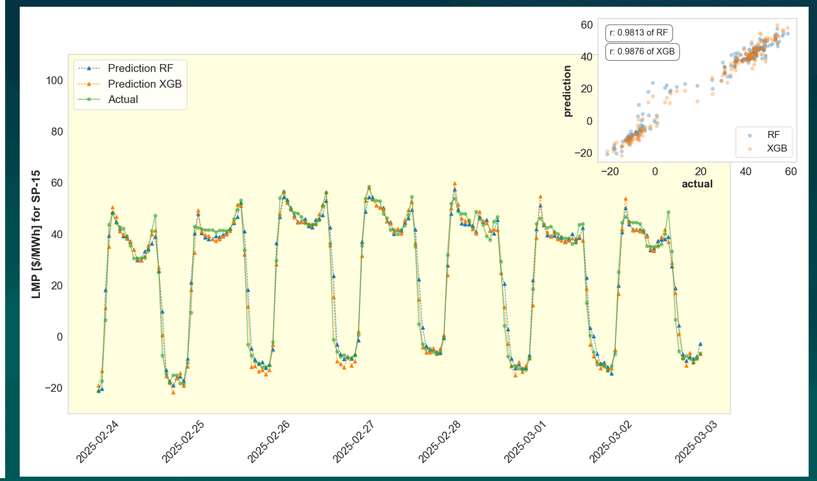




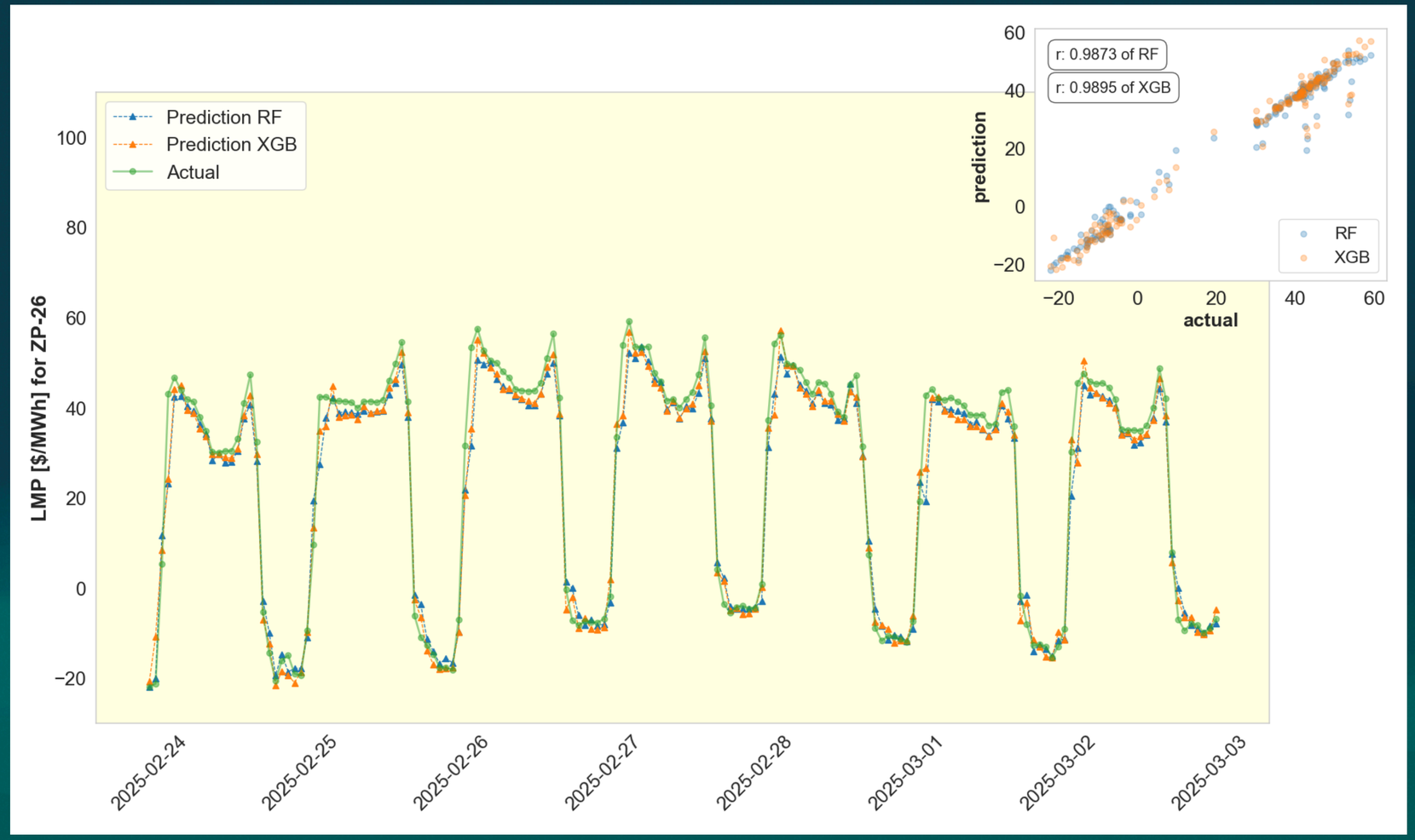
Models with enhanced features

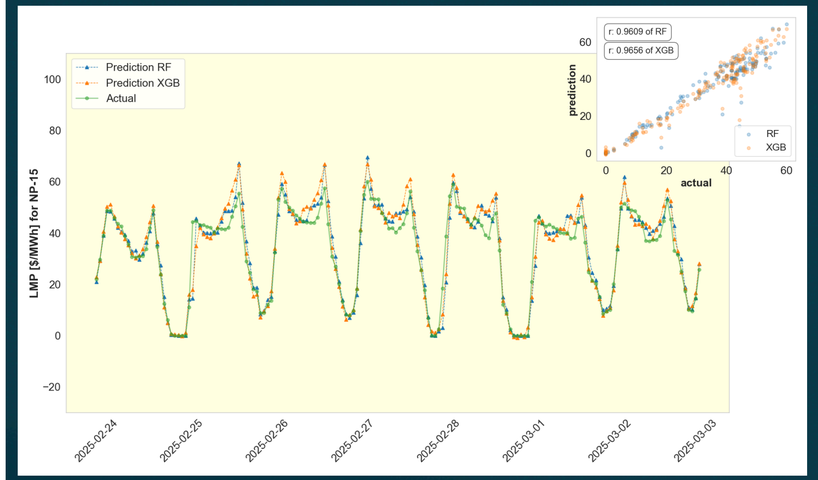


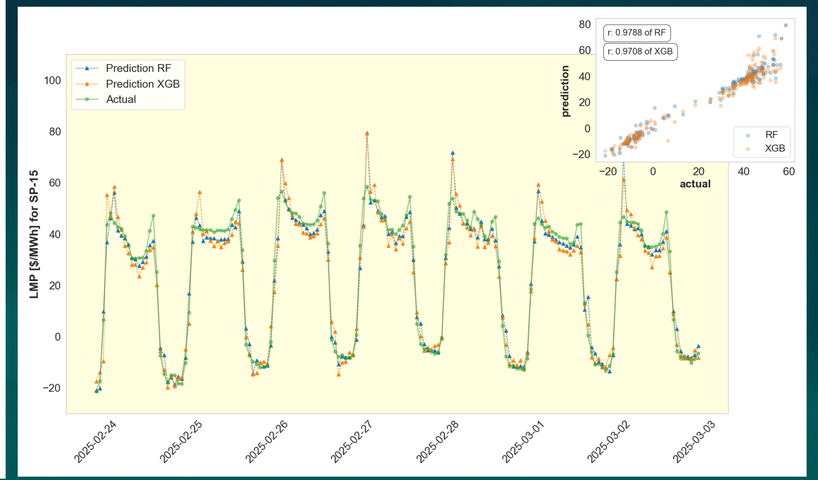




Models with volatility inclusion

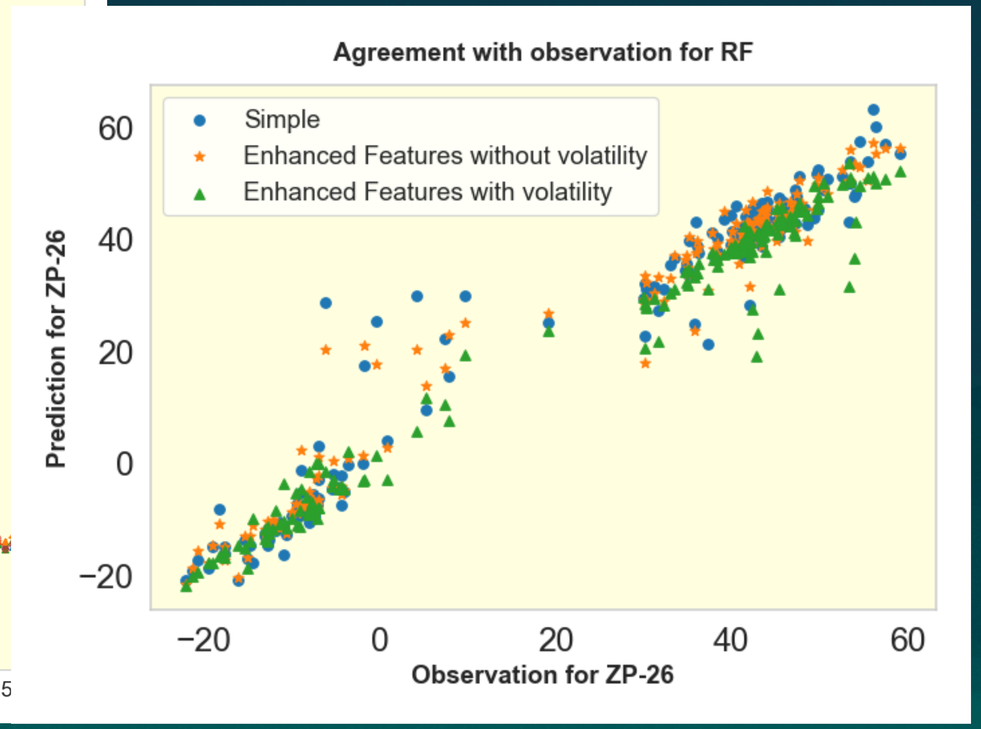






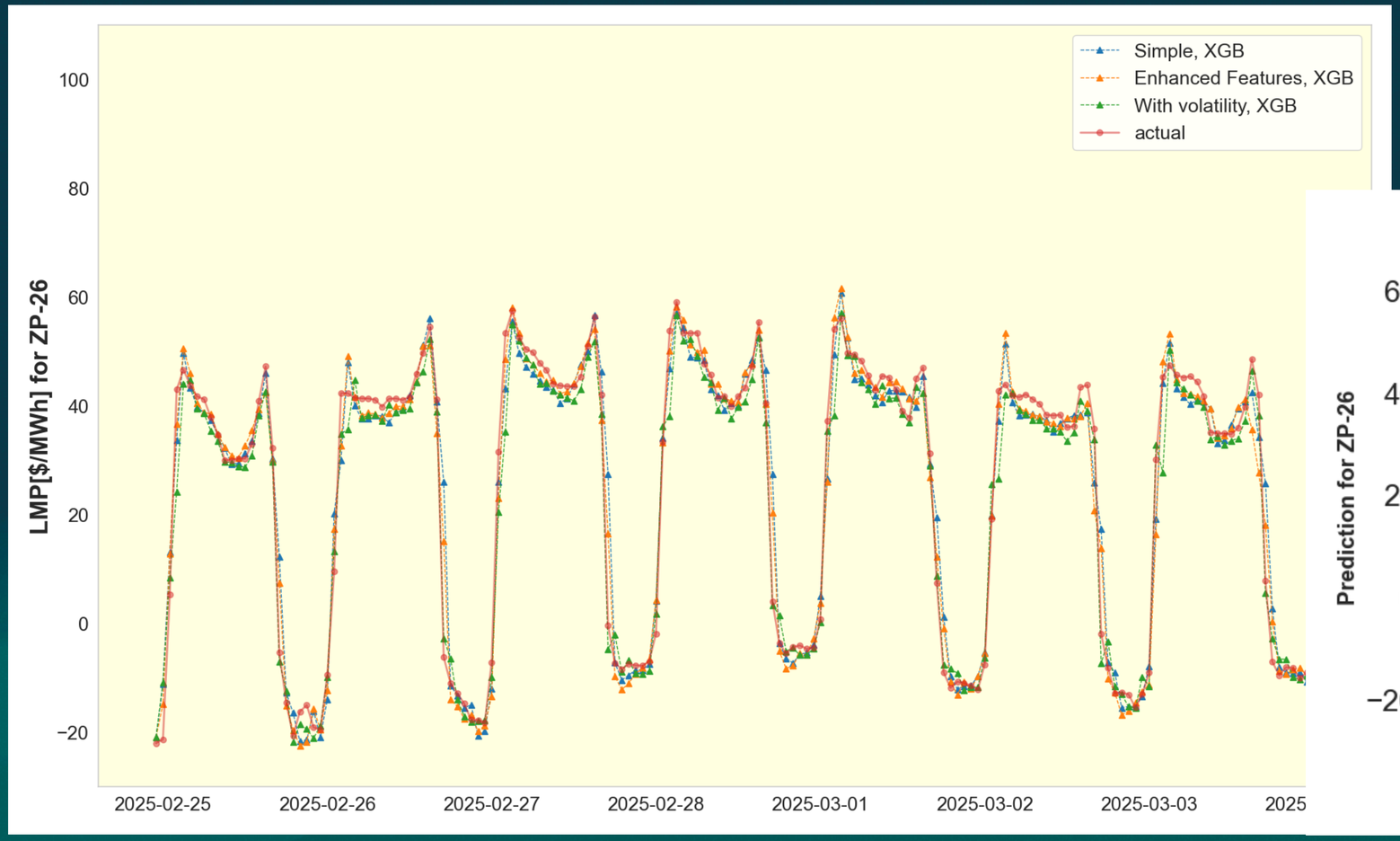
Model improvements

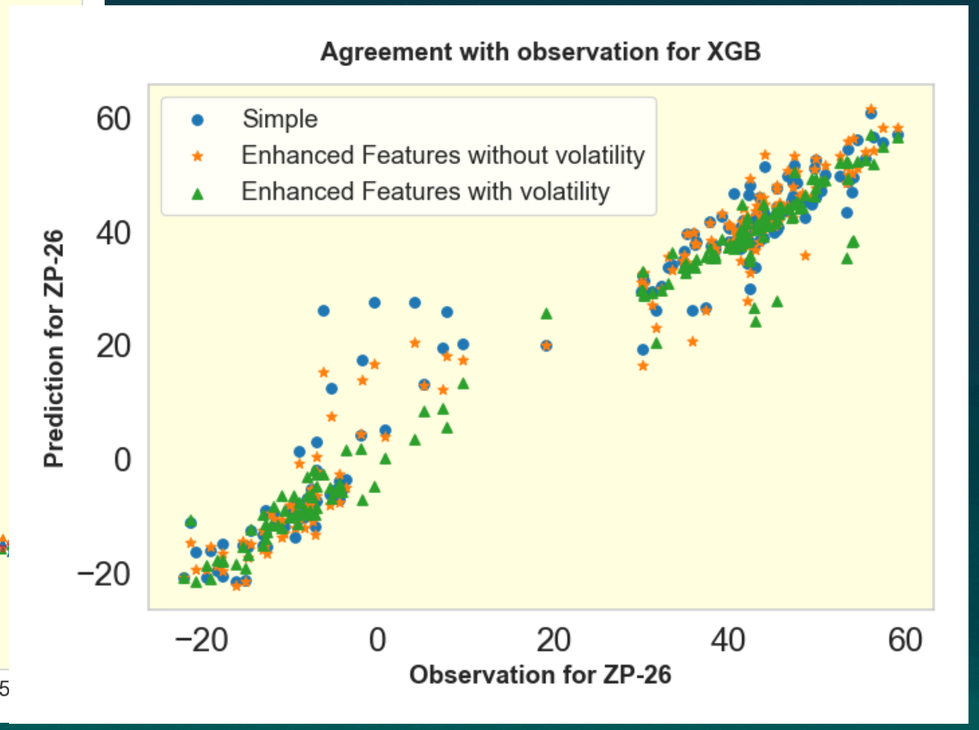
Use case: RandomForest



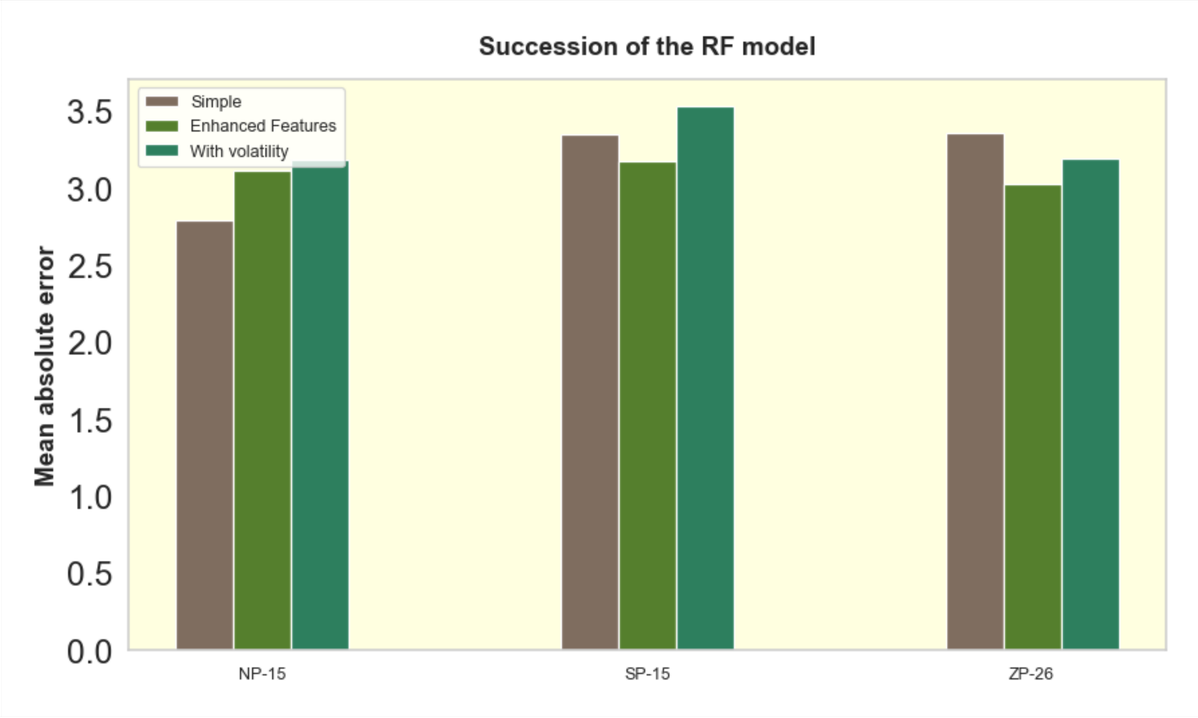
Model improvements

Use case: XGBoost

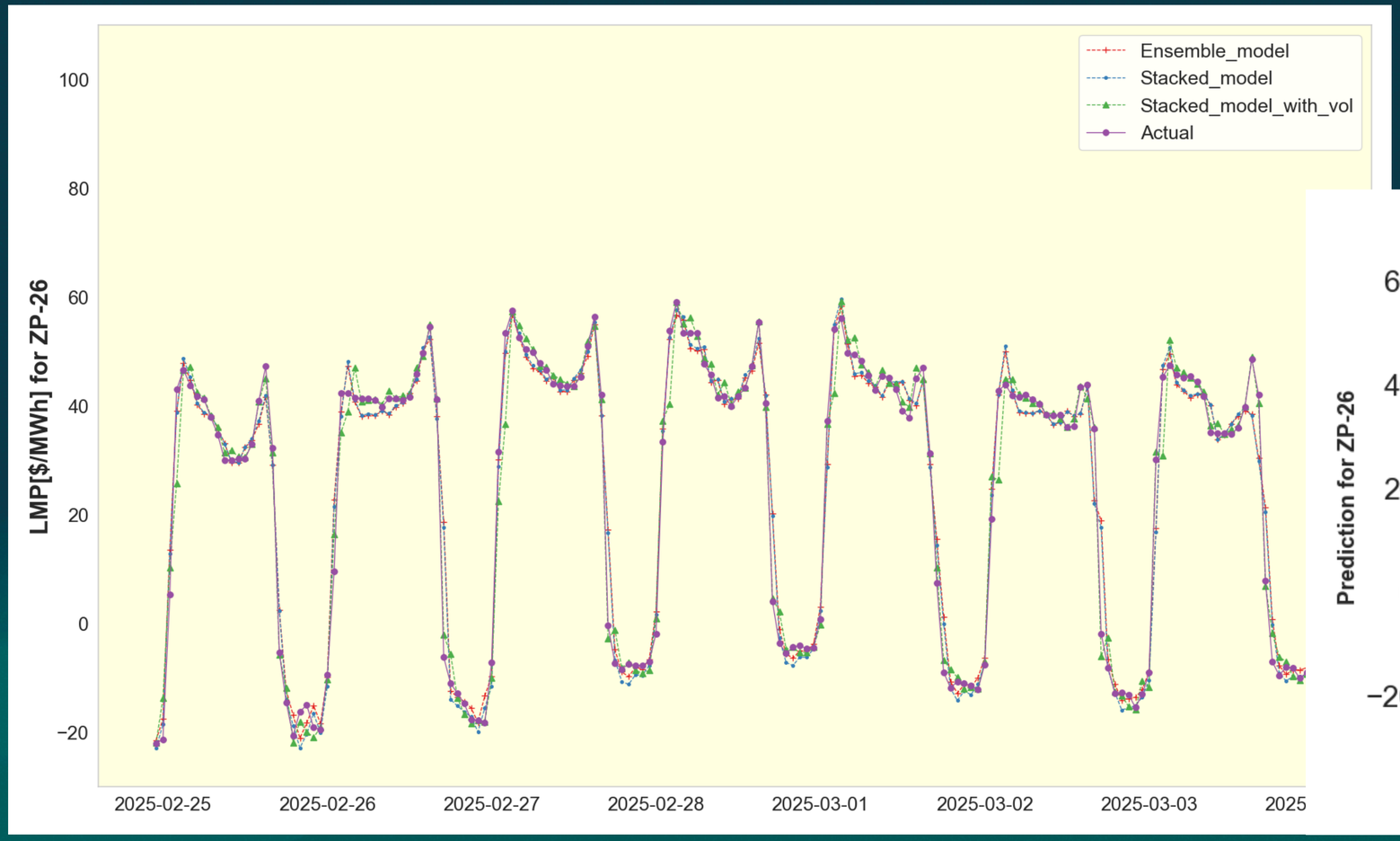


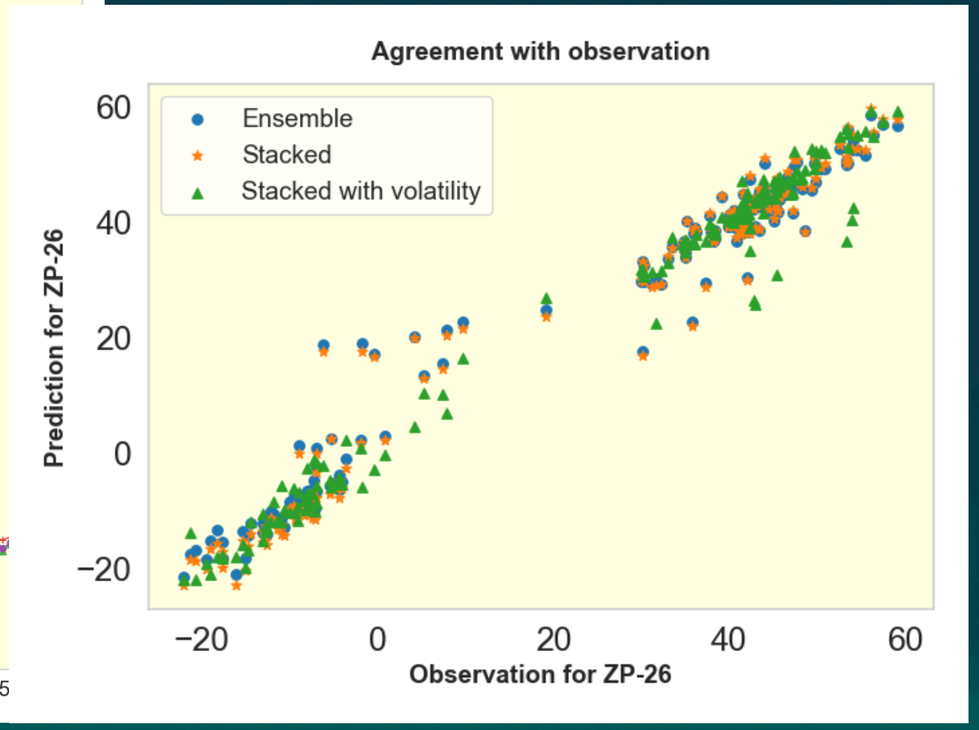


Performance analysis

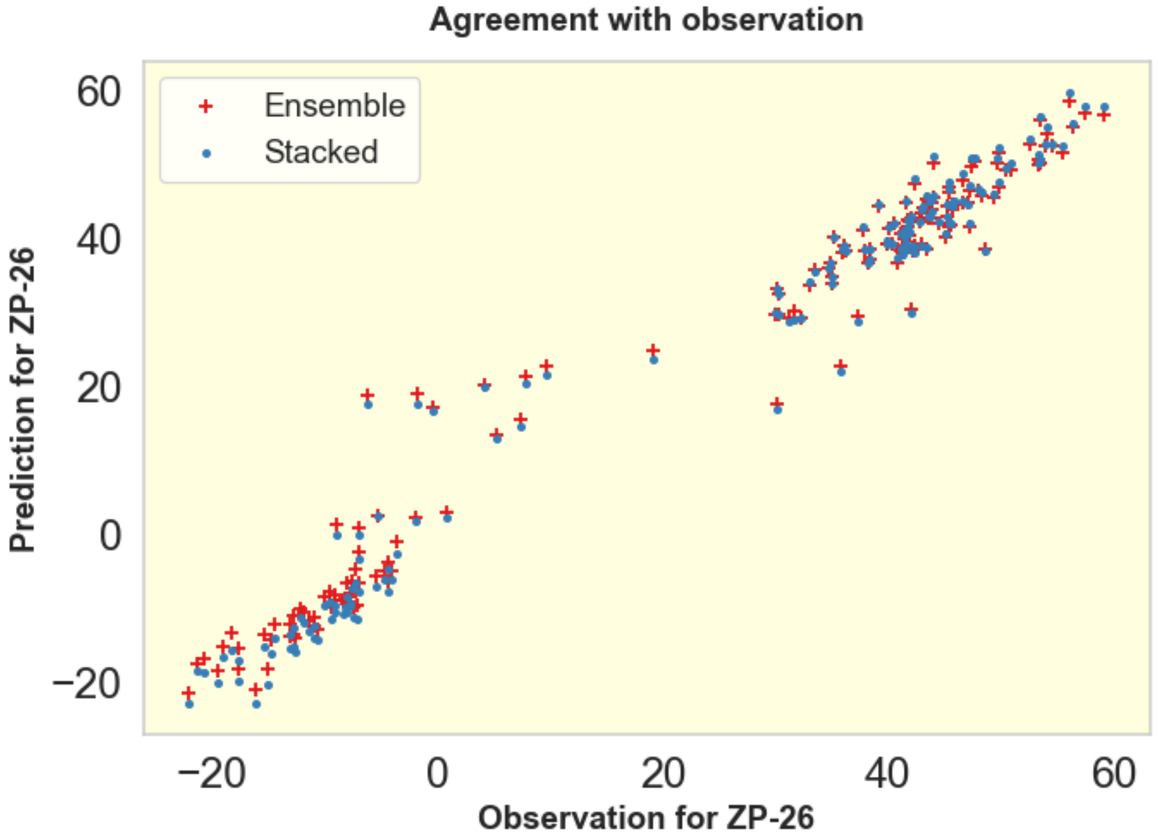


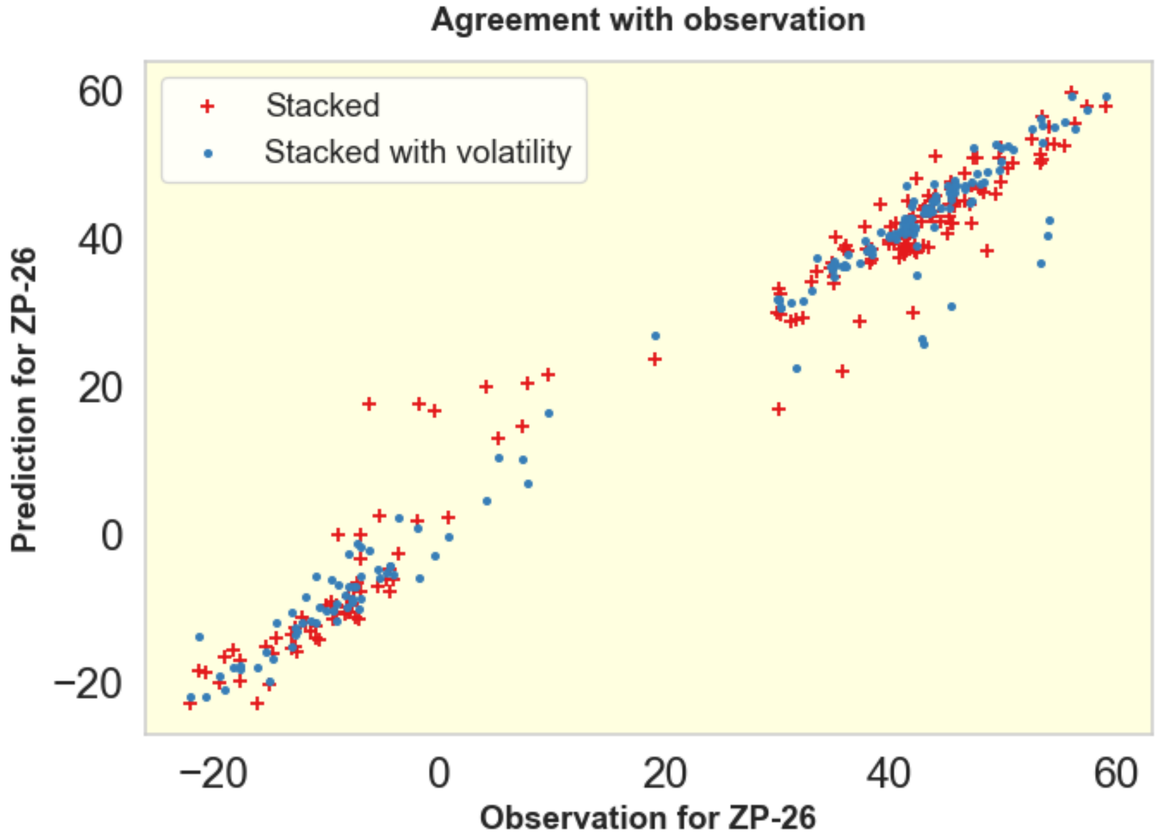
Results of combined models





Improvements with model successions





Performance analysis

Critical analysis

Given the different levels of temporal dynamics in each area, the model needs to be optimized separately to yield the best result. The model, so far, is optimized for ZP-26. In future, NP-15 and SP-15 will also be optimized for improved performance.

In the absence of any exogenous variable, like demand and renewable energy share, the model has the tendency to often misinterpret noise as volatility, which reflects poorly in the forecast. A deeper dive into defining the appropriate order of volatility is required to ensure smooth performance.

So far the model performance is only evaluated against a few error measures, like Mean Absolute Error, Mean Squared Error, Correlation coefficient. To extend the applicability of the model to stakeholders and other utility owners, a probabilistic forecast is preferred over point forecast. In such cases, a CRPS-score is often the more desired and useful measure of error.

Future work & outlook

In future, the models will be updated with inputs from exogenous variables like generation mix, demand, grid congestion and more.

Also, volatility will be further analyzed using other measures like realized volatility, realized semi-variance, realized bi-power variance etc,.

The models should also be used for other day-ahead electricity markets like PJM, ERCOT, MISO.

Exploring reinforcement learning and testing further ensemble learning approaches will be an important part of future endeavors.

Thank You

For questions, please send me an email at
kabitri.chattopadhyay@gmail.com