Electricity Load and Peak Forecasting: Feature Engineering, Probabilistic LightGBM and Temporal Hierarchies

Nicolò Rubattu[1], Gabriele Maroni[1], and Giorgio Corani[1]

[1] Dalle Molle Institute for Artificial Intelligence (IDSIA), USI-SUPSI,
CH-6962, Lugano, Switzerland
{nicolo.rubattu, gabriele.maroni, giorgio.corani}@idsia.ch

Abstract. We describe our experience in developing a predictive model that placed a high position in the BigDEAL Challenge 2022, an energy competition of load and peak forecasting. We present a novel procedure for feature engineering and feature selection, based on cluster permutation of temperatures and calendar variables. We adopted gradient boosting of trees and we enhanced its capabilities with trend modeling and distributional forecasts. We also included an approach to forecasts combination known as temporal hierarchies, which further improves the accuracy.

Keywords: Load Forecasting · Feature engineering · Gradient Boosting · Hierarchical Forecasting · Forecast Reconciliation

1 Introduction

Load forecasting is the problem of predicting the future profile of power demand, while *peak forecasting* is the problem of predicting the maximum (e.g. daily) value of demand and the time of its occurrence. Peak forecasting is important because often decisions are made based on the forecast of the peak rather than on the forecast of the entire load profile.

In this work, we present an approach that successfully competed in the BigDEAL Challenge 2022, which was about energy load and peak forecasting. The competition was held in October-December 2022; 121 contestants took part, divided into 78 teams. The forecasts were assessed using different indicators and the competition was split into a qualifying match and a final match. We achieved the 3^{rd} position in the qualifying match, gaining access to the final match, where we ended 6^{th} [16].

For the qualifying match, we used Gradient Boosting (GB) of trees, coupled with an original method for feature engineering and feature selection. For the final match, we developed a more sophisticated approach. In particular, we adopted a recent probabilistic version of LightGBM [28] and used temporal hierarchies [3] in order to improve the forecasts by combining predictions at different temporal scales. Even though the competition only scored the point forecasts, our approach is probabilistic and thus quantifies the uncertainty of the forecasts. This is indeed needed to support decision-making.

We present our approach in this paper, which is organized as follows. In Sec. 2.1 an outlook of long-term load forecasting and our motivations are given. We introduce Gradient Boosting (GB) of trees and probabilistic extensions in Sec. 2.2. We present our approach for feature engineering for load forecasting in Sec. 2.3, and feature selection in Sec. 2.4. Temporal hierarchies are presented in Sec. 2.5. In Sec. 3 we detailed review our pipeline with technical insights, and competition results. We end this work with a critical conclusion in Sec. 4.

2 Methodology

2.1 Long-term Load Forecasting

Load forecasting is the problem of predicting the electricity demand of the next H time steps, denoted by $[y_{T+1}, \dots, y_{T+H}]$. When the order of magnitude of H is a few hundred or more, we talk about *long-term* forecasting. For instance, forecasting a year ahead at an hourly scale implies producing $24 \times 365 = 8760$ forecasts. Classical forecasting strategies [4] condition the forecast on the last observations of the time series. However, this is not viable for long-term forecasting, since in this case y_{T+H} is independent of y_T. Long-term forecasting is better addressed as a regression problem, adopting a rich set of explanatory variables (*features*) regarding calendar effects, temperatures, etc. [7]. This approach allows adopting regression methods such as Gradient Boosting (GB) of trees [11], which is indeed successful in long-term energy forecasting [32].

2.2 Gradient Boosting and Distributional forecasts

In fact, GB achieved top positions in the Global Energy Forecasting Competitions (GEFCom) of 2012, 2014, and 2017 [18–20], in the M5 forecasting competition [23], and competitions on tabular data [6]. The most popular implementations are XGBoost, LightGBM, and CatBoost.

GB can be trained with different loss functions besides the traditional least squares. For instance, GB trained to perform quantile regression won the GEFCom2014 probabilistic competition [12]. Yet, even quantile regression only returns point forecasts without a predictive distribution. It is possible to train different GB models, one for each desired quantile; but if the predicted quantiles cross, the predictive distribution is invalid [29, 30]. The recent versions of probabilistic GB of trees constitute a sounder approach [9, 27, 28] to probabilistic forecasting. In this work, we adopt the LightGBM extended model of März et al. [28], which returns the moments of the predictive distribution.

A successful implementation of GB requires anyway paying attention to some possible issues. For instance, GB is generally unable to model a long-term trend. If the time series is trendy, it is recommended to detrend it, fit the GB model, and then add the predicted trend to the GB forecast [34]. Another preprocessing step that is sometimes helpful is a logarithmic transformation which stabilizes the variance of the target time series [31]. Moreover, GB is subject to overfitting. The DART algorithm [33] solves the problem by introducing Dropout regularization analogously to Neural Networks.

2.3 Feature Engineering

The exogenous variables that are frequently used in load forecasting are related to calendars and temperatures.

Calendar features Calendar variables allow to capture the seasonal patterns. They are commonly modeled by categorical variables. For example, the day of the week is represented by a categorical variable with seven levels. Holidays are represented by a binary variable: 1 for holidays and 0 for non-holiday.

Lagged and rolling temperatures Temperature impacts energy consumption, by driving the use of heating, ventilation, and air conditioning (HVAC) systems. However, there is generally a delay between the change in temperature and the change in energy consumption. We thus consider the lagged hourly temperatures:

$$T(t-h), \qquad h = 1, 2, \ldots, L \tag{1}$$

where L is the maximum lag; and the rolling temperature's statistics:

$$T_f^w(t) = f(T(t-1), \ldots, T(t-w)) \tag{2}$$

where $f(\cdot)$ is some statistical function and w indicates the width of the window of past values of hourly temperatures considered. For example, the moving average of the last 24 hours of temperature values is $T_{avg}^{24}(t) = \frac{1}{24}\sum_{h=1}^{24} T(t-h)$.

Aggregated indicators of temperature Aggregated features can capture the long-term effect of temperature on energy load. They can be expressed as $\tilde{T}_f^g(t)$ where g is the aggregation period and $f(\cdot)$ is the aggregation function. These features include, for example, the daily maximum and minimum values of the temperature or the monthly standard deviation of the temperature.

In this paper, we propose additional aggregation functions (Tab. 1) borrowed from signal processing [10,36], which to the best of our knowledge have not yet been used in energy forecasting. They should be computed on the time series of temperature, and provide insights about the variability and shape within the aggregation period. For example, the crest factor measures the peak-to-average ratio of a signal; a high daily crest factor corresponds to large variations of temperature during the day, which generally increase energy demand; a low daily crest factor corresponds to stable temperatures during the day, which generally decreases energy demand.

2.4 Feature Selection

Feature engineering generates a large set of features, after which feature selection is needed [22, 25]. We perform feature selection based on hierarchical clustering and pairwise correlation of the features. The core of our approach is Permutation Feature Importance (PFI), which measures the drop in performance when a feature is randomly shuffled [5]. The size of the drop in performance shows how much the model relies on that feature for prediction. PFI is

Table 1. Signal Processing features for load forecasting.

RMS	$x_{RMS} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2}$
Peak value	$x_p = \max(\lvert x_i \rvert)$
Crest factor	$x_{crest} = \frac{x_p}{x_{RMS}}$
Impulse factor	$x_{if} = \frac{x_p}{\frac{1}{N}\sum_{i=1}^{N} \lvert x_i \rvert}$
Margin factor	$x_{mf} = \frac{x_p}{\left(\sum_{i=1}^{N} \lvert x_i \rvert^{1/2}\right)^2}$
Shape factor	$x_{sf} = \frac{x_{RMS}}{\frac{1}{N}\sum_{i=1}^{N} \lvert x_i \rvert}$
Peak to peak value	$x_{pp} = \max(x_i) - \min(x_i)$

Algorithm 1 Permutation Feature Importance

```
Require: A trained model and recorded score $s$ on an evaluation dataset.
  for feature $x_j, j = 1, \dots, d$ do
    for each repetition $k, k = 1, \dots, K$ do
      Randomly shuffle column $j$ of the original evaluation set.
      Compute the new score $s_{k,j}$ of the model on the perturbed set.
    end for
    Compute the importance of feature $x_j$ as $I_j = s - \frac{1}{K}\sum_{k=1}^{K} s_{k,j}$
  end for
```

appealing since it can be applied to any model; it is easy to implement (Algorithm 1); it can measure feature importance on the metric of the competition; it can be computed out-of-sample.

However, shuffling a single feature can produce unrealistic results if features are dependent. Furthermore, correlated features share importance, therefore their relevance may be underestimated (substitution effect, [15]).

Clustered Permutation Feature Importance To solve such issues we propose a novel approach, which we call Clustered Permutation Feature Importance (CPFI). The method works as follows.

At first, groups of highly correlated features are identified by applying hierarchical clustering on the correlation matrix of the features. For that, a measure of dependence between each feature pair is computed using a correlation index, Pearson's or Spearman's for instance. Then, all the variables of the same cluster are shuffled, and the subsequent performance drop is computed. The more orthogonal the information contained in different clusters, the more reliable the estimate of importance. Finally, non-informative feature clusters are dropped. Also, only one or few features can be selected from each relevant cluster based on some measure of explanation with respect to the target, or some expert advice. We propose a criterion for informativeness in Sec. 3.

2.5 Temporal hierarchies

As a further tool to improve forecasting accuracy, we consider temporal hierarchies [3]. For instance, assume that we want to generate forecasts at the hourly scale (referred to as the *bottom* level). A temporal hierarchy creates and combines forecasts also at coarser temporal scales (e.g., 2-hourly and 4-hourly), referred to as the *upper* levels. The smoothness of the upper time series enables enhanced modeling of long-term patterns. This process generally improves forecasting accuracy at all levels [3, 24].

A temporal hierarchy works as follows. First, forecasts are independently created at different temporal scales (*base forecasts*). For instance, Fig. 1 shows a temporal hierarchy aimed at forecasting 4-hours ahead. It contains 4 forecasts computed at hourly frequency ($\hat{h}_1, ..., \hat{h}_4$, bottom level); two forecasts computed at 2-hour frequency ($\hat{h}_{12}, ..., \hat{h}_{34}$, intermediate level); one forecast computed at 4-hour frequency ($\hat{h}_{1234}$, top level). Generally, the base forecasts do not sum up correctly and they are referred to as *incoherent*. For instance: $\hat{h}_{12} \neq \hat{h}_1 + \hat{h}_2$, $\hat{h}_{34} \neq \hat{h}_3 + \hat{h}_4$, etc. *Reconciliation* [35] is the process of adjusting the base forecast so that they become *coherent*, i.e., they sum up correctly. The reconciled forecasts are denoted with a tilde and thus in the example of Fig. 1 after reconciliation, we have: $\tilde{h}_{12} = \tilde{h}_1 + \tilde{h}_2$, $\tilde{h}_{34} = \tilde{h}_3 + \tilde{h}_4$, $\tilde{h}_{1234} = \tilde{h}_{12} + \tilde{h}_{34}$.

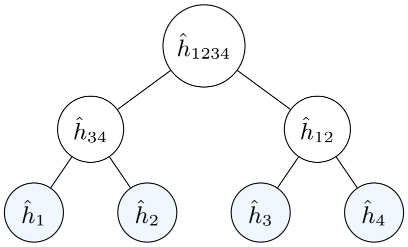


Fig. 1. Temporal hierarchy for forecasting 4-hours ahead, using hourly forecasts (bottom level), 2-hourly forecasts, and 4-hourly forecasts.

Temporal hierarchies require the mean and the variance of the base forecasts. The original algorithm [3] provides only the reconciled point forecast, while the approach of [8] yields also a reconciled predictive distribution.

3 Experiments

The BigDEAL Challenge 2022 was divided in a qualifying match and a final match. The *qualifying match* provided hourly load and hourly temperature statistics (mean, median, min, max) of four weather stations for the period 2002-2006; see Fig. 2 for an example. It required forecasting the year 2007 given the actual temperatures. This is referred to as *ex-post* setting. The *final match* provided three years (2015-2017) of hourly load of three U.S. local distribution companies (LDC), and hourly temperatures from six weather stations. The forecasted (*ex-ante* setting) 1-day ahead temperatures for 2018 were released on a rolling basis, two months at a time. The forecasts for these periods were to be delivered, in a total of six consecutive rounds. Both matches required forecasts at hourly scale for the 24h, the values of the peak for each day, and its time of occurrence (i.e. a discrete number between 1 and 24).

The qualifying match served as a support for participants to validate their forecasting approach. Its ex-post setting is optimistic as *actual* temperatures for the forecasting horizon are used. The ex-ante setting of the final match instead represents a realistic scenario as forecasts are obtained from *forecasted* temperatures. In the literature, the comparison of the two settings is used to measure the effectiveness of the forecasting models [21], i.e. the influence of the forecast errors is isolated in the input variables.

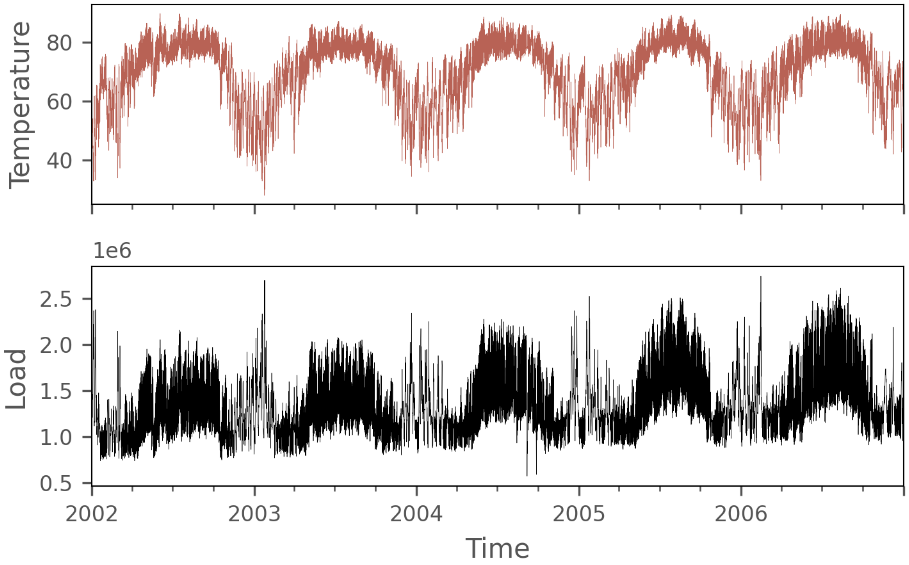


Fig. 2. Load (MW) and temperature T$_{avg}$ (°F) of the qualifying match of the BigDEAL challenge. For readability, we show data aggregated over 12 hours.

3.1 Performance measures

The organizers evaluated the forecasts of each match with three different tracks.

In the qualifying match the hourly forecasts ($24 \times 365 = 8760$) were scored using the *Mean Absolute Percentage Error* (MAPE):

$$\text{MAPE} = \frac{1}{H} \sum_{t=T+1}^{T+H} \frac{|y_t - \hat{y}_t|}{|y_t|} \times 100, \tag{3}$$

where y_t and $\hat{y}_t$ denote the actual and the forecasted value for time t. The second metric was the *Magnitude* (M); it is the MAPE between the actual and forecasted daily peak values (i.e., it refers to 365 forecasts with a one-year horizon). We recall that MAPE has been criticized in the forecasting literature: it penalizes over-estimation errors more than under-estimation ones [2] and it is numerically unstable when dealing with values close to 0. To score the prediction of peak hours the organizers used a third metric, called *Timing* (T), which computes the *Mean Absolute Error*. For example, if the actual peak is at 6 pm, and the forecasted peak time is at 8 pm, the error for that day is $|6-8| = 2$.

The final match scored the forecasts using *Magnitude* (M) and *Timing* (T), plus an additional metric called *Shape* (S). However, the definition of Timing was modified introducing a non-uniform cost for the error. Let us denote by $\mathcal{T}_d$ and $\hat{\mathcal{T}}_d$ the actual and the forecasted peak hour for a day d. Timing was then defined as:

$$\mathrm{T} = \frac{1}{|days|} \sum_{d \,\mathrm{in}\, days} w(\mathcal{T}_d, \hat{\mathcal{T}}_d), \text{with}$$

$$w(\mathcal{T}_d, \hat{\mathcal{T}}_d) = \begin{cases} |\mathcal{T}_d - \hat{\mathcal{T}}_d|, & \text{if } |\mathcal{T}_d - \hat{\mathcal{T}}_d| = 1, \\ 2|\mathcal{T}_d - \hat{\mathcal{T}}_d|, & \text{if } 2 \leq |\mathcal{T}_d - \hat{\mathcal{T}}_d| \leq 4, \\ 10, & \text{if } |\mathcal{T}_d - \hat{\mathcal{T}}_d| \geq 5 \end{cases} \tag{4}$$

Shape (S) scored the shape of the forecast around the peak. To compute it, the 24h load forecasts of a day are normalized by the peak forecast of that day, and the same is done for the actual load. Then the sum of absolute errors during the 5-hour peak period (actual peak hour ± 2 hours) of every day is calculated. We denote by $\bar{y}_d$ and $\bar{\hat{y}}_d$ the normalized actual and forecasted load for a day d; $\bar{y}_d = \frac{y_d}{\max y_d}$, $\bar{\hat{y}}_d = \frac{\hat{y}_d}{\max \hat{y}_d}$. Shape is defined as:

$$\mathrm{S} = \frac{1}{|days|} \sum_{d \,\mathrm{in}\, days} \sum_{t \,\mathrm{in}\, \{\mathcal{T}_d, \mathcal{T}_d \pm 1, \mathcal{T}_d \pm 2\}} |\bar{y}_d(t) - \bar{\hat{y}}_d(t)| \tag{5}$$

Scoring the Predictive Distribution While the competition only assessed the point forecasts, we also scored the distributional forecasts obtained from our probabilistic models. In particular, we compared the probabilistic forecast of our GB model (based on [28]) with those obtained after the application of the temporal hierarchy. We scored the predictive distributions of the model using the *Continuous Ranked Probability Score* (CRPS) [14]. Let us denote by $\hat{\mathbf{F}}$ the predictive cumulative distribution function and by y the actual value:

$$\mathrm{CRPS}(\hat{\mathbf{F}}, y) = \int_{-\infty}^{\infty} (\hat{\mathbf{F}}(x) - \mathbb{1}(x \geq y))^2 \, dx \tag{6}$$

With Gaussian $\hat{\mathbf{F}}$, the integral can be computed in closed form [14]. We then scored the prediction intervals using the *Interval Score* (IS) [13]. Let us denote by $1-\alpha$ the nominal coverage of the interval (assumed 0.9 in this paper), by $\mathbf{l}$ and $\mathbf{u}$ its lower and upper bound. We thus computed with the models, for each hour, a 90% prediction interval and the score:

$$\mathrm{IS}(\mathbf{l}, \mathbf{u}, y) = (\mathbf{u} - \mathbf{l}) + \frac{2}{\alpha}(\mathbf{l} - y)\mathbb{1}(y < \mathbf{l}) + \frac{2}{\alpha}(y - \mathbf{u})\mathbb{1}(y > \mathbf{u})\,. \tag{7}$$

We also report the proportion of cases in which the interval $(\mathbf{l}, \mathbf{u})$ contains y.

Skill score Let m_{origin} and m_{new} be the results obtained by two different models on a certain metric to be minimized. We denote the positive or negative percentage improvement by the Skill score defined as:

$$\text{Skill}_{\%}(m_{origin}, m_{new}) = \frac{m_{origin} - m_{new}}{(m_{origin} + m_{new})/2} \times 100 \tag{8}$$

3.2 Qualifying Match

Here, we detail the building blocks of our implementation.

Baseline We started by modeling essential calendar features (`Year`, `Month`, `Week`, `Day`, `Weekday`, `Hour`) and temperatures at the current time ($\texttt{T}_{avg}$, $\texttt{T}_{med}$, $\texttt{T}_{min}$, $\texttt{T}_{max}$). We applied a logarithmic transformation to the target variable to stabilize its variance. Moreover, since the target variable has a long-term increasing trend, we performed detrending. We fitted a Linear Regression (LR) model ($y_i = \beta_0 + \beta_1 x_i$, where x_i are progressive time indices with $i = 1, \ldots, T$) to the training data. We then subtracted the linear trend before fitting the LightGBM model. At prediction, we added the extrapolated trend to the out-of-sample predictions, followed by an exponential transformation, to obtain the final forecast. With detrending: the residuals have a mean of 0, otherwise, they are severely biased; we reduced the MAPE (H) of the baseline model from 6.18 to 4.81.

Cross-validation We used time series cross-validation to evaluate the performance of each model, hyper-parameter tuning, and feature selection. The size of the time window is typically chosen equal to the size of the test set on which the final prediction is to be made. Hence, for the qualifying phase, the years 2004, 2005, and 2006 were used as out-of-sample folds.

Feature engineering Feature engineering was performed *incrementally* by adding related feature blocks one step at a time. We found the following features to be predictive for this competition:

- Additional calendar features: `Holiday`, `Holiday name`, `Weekend`, `Week of month`, `Season`, `Day of year`, `Days since last / until next holiday`. We transform the `Holiday name` string feature with label encoding.
- Lagged hourly temperatures: for each temperature variable ($\texttt{T}_{avg}$, $\texttt{T}_{med}$, $\texttt{T}_{min}$, $\texttt{T}_{max}$) lagged hourly temperatures were incorporated into the model, ranging from a minimum lag of 1 hour to a maximum lag of 48 hours, for a total of 192 new features.
- Temperature-based rolling statistics: for each temperature variable, and 4 different values of window widths (3 hours, 1 day, 1 week, 1 month), 5 statistical functions (*mean, max, min, median, std*) were computed, for a total of 80 new features.

- Aggregated temperature statistics: for each temperature variable, for 2 different aggregation periods (Year-Month-Day, Month-Hour), 11 aggregation functions (*mean, max, min, median*, and centered *RMS, crest factor, peak value, impulse factor, margin factor, shape factor, peak to peak value*) coupled with the differences between the current temperature values and the aggregated values were computed for a total of $88 \times 2 = 176$ new features. For example, we denote by $\tilde{T}^{Year,Month,Day}_{max}(t)$ the daily maximum, where Year-Month-Day is the aggregation period, to be read from left to right.

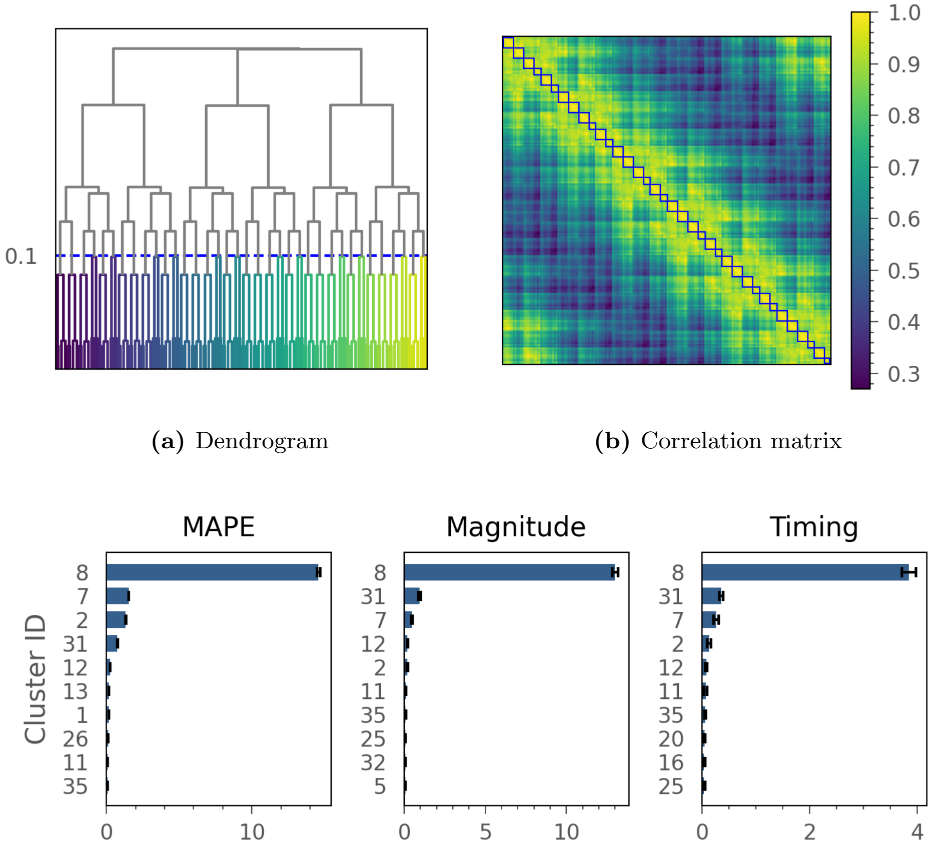
Feature selection To evaluate our feature selection strategy, we carried out multiple experiments. First, we assessed the model performance without any feature selection (experiment **a**). Then, we applied the feature selection strategy described in Sec. 2.4 after completing all feature engineering, on the entire set of features added to the baseline model (experiment **b**). Finally, we performed *step-by-step* feature selection whenever we added a new block of features to the model, i.e. after adding lagged variables, after adding rolling variables, and so on (experiment **c**).

Cluster permutations were executed 100 times, and mean values and standard deviations of performance drops were calculated against all the out-of-sample folds. We consider a cluster of features *informative* if the importance value fall within three standard deviations of the mean, above 0. Results are presented in Tab. 2. Specifically, the columns for MAPE, Magnitude, and Timing present the results based on the respective competition metrics, whereas columns **a**, **b**, and **c** correspond to the 3 experimental strategies employed. It is important to note that, unlike experiment **a**, where the results were obtained in a single training run, the results for experiments **b** and **c** were derived from three different training runs, each one maximizing the metric of interest.

Table 2. Out-of-fold qualification results with feature selection methods.

	MAPE (H)			Magnitude			Timing		
	a	**b**	**c**	**a**	**b**	**c**	**a**	**b**	**c**
Baseline	4.81	-	-	4.43	-	-	1.42	-	-
Calendar	4.83	-	4.78	4.48	-	4.46	1.39	-	1.39
Lags	3.33	-	3.24	3.29	-	3.20	0.94	-	0.92
Roll lags	3.28	-	3.16	3.22	-	3.20	1.06	-	0.94
Agg stats	3.24	3.16	3.09	3.21	3.10	3.08	0.91	0.95	0.91

For illustration purposes, in Fig. 3, we present the feature selection results obtained after incorporating lagged hourly temperatures into the model. Fig. 3a presents the dendrogram obtained from hierarchical clustering computed on the Spearman correlation matrix, which is shown in Fig. 3b. With a threshold value of 0.1, we identified 36 clusters. The cluster rankings that maximize, respectively, the performance of MAPE, Magnitude, and Timing are visible in Fig. 3c. For all the metrics, cluster 8 proved to be the most significant, followed by clusters 31, 7, 2, and 12. This suggests that most informative lags are at t-$\{1, 2, 3, 4, 5, 6\}$, t-$\{11, 12\}$, and t-$\{25, 26\}$. Tab. 3 shows the clusters associated feature set.



(a) Dendrogram

(b) Correlation matrix

(c) CPFIs

Fig. 3. Hierarchical clustering (threshold of 0.1) (a) and Spearman's correlation matrix (b). The blue squares highlight the 36 clusters. In (c) Clustered Permutation Feature Importance (CPFI) values are reported for each track.

Hyper-parameter optimization We used the Optuna framework [1] to tune the learning control parameters of LightGBM, primarily: the max number of leaves in one tree, the minimal number of data in one leaf, L1 and L2 regularization, bagging and feature fractions, the number of estimators, and the learning rate. The parameters are optimized for the best cross-validation performance, also considering the standard deviation of the different folds. Optuna implements

Table 3. Clustered Permutation Feature Importance: Top-5 clusters of lagged temperatures that maximize performance indicators.

Cluster ID	Feature Set
8	$\mathtt{T}_{avg,med,min}(t-1)$, $\mathtt{T}_{avg,med,min}(t-2)$
31	$\mathtt{T}_{avg,med,min}(t-11)$, $\mathtt{T}_{avg,med,min}(t-12)$
7	$\mathtt{T}_{avg,med,min}(t-3)$, $\mathtt{T}_{avg}(t-4)$
2	$\mathtt{T}_{avg,med,min}(t-5)$, $\mathtt{T}_{avg,med}(t-6)$
12	$\mathtt{T}_{max}(t-1)$, $\mathtt{T}_{max}(t-2)$, $\mathtt{T}_{max}(t-25)$, $\mathtt{T}_{max}(t-26)$

time-budget optimization which was useful given the short deadlines of the competition.

Results Our team was named “swissknife”; as reported in Tab. 4, we ranked 8^{th} on the hourly forecast (H), 3^{rd} on the Magnitude (M), 3^{rd} on the Timing (T).

Table 4. Leaderboard of Qualifying Match [17].

Team	Rank H.	Team	Rank M.	Team	Rank T.
X-Mines	1	Amperon	1	RandomForecast	1
Amperon	2	Team SGEM KIT	2	Amperon	2
Yike Li	3	swissknife	3	swissknife	3
peaky-finders	4	peaky-finders	4	freshlobster	4
KIT-IAI	5	KIT-IAI	5	peaky-finders	5
Overfitters	6	EnergyHACker	6	*Recency Benchmark*	
BelindaTrotta	7	BelindaTrotta	7	X-Mines	6
swissknife	8	Overfitters	8	BrisDF	7
Recency Benchmark		VinayakSharma	9	BelindaTrotta	8
RandomForecast	9	SheenJavan	10	KIT-IAI	9
Team SGEM KIT	10	...		SheenJavan	10
...		*Recency Benchmark*	13	...	
Tao’s Vanilla Benchmark	27	*Tao’s Vanilla Benchmark*	25	*Tao’s Vanilla Benchmark*	30

3.3 Final Match

For the final match, we followed the same pipeline tuned in the qualification phase, with the exception of target transformation, which was not required as the target variable was already stationary. Additionally, three LDC loads were required to be forecasted (`LDC1, LDC2, LDC3`), and the temperature variables come from six weather stations (`T1, T2, T3, T4, T5, T6`), without aggregate statistics and geographical references. To further enhance performance, we incorporated several techniques, including DART, probabilistic LightGBM, and temporal hierarchies.

Feature selection The most important lagged temperatures were found at time t-$\{1,2,3,4,5\}$, and t-$\{10,11,12\}$, and the most important rolling lag temperatures were found with $w = \{3\,hours,\, 1\,day\}$. Fig. 4 shows that within the six weather stations, temperatures {T1, T2, T5} better explain LDC1. Analogously, LDC2 is better explained by {T3, T4}, and LDC3 by {T5, T6, T1}. To save space, we do not present the Out-of-fold Top-20 features for LDC2 and LDC3 in this paper, but the results are in line with those of LDC1. Hence, even if according to the guidelines of the competition it was not necessary to rely on the location of the data, our method nicely handles datasets with multiple weather stations.

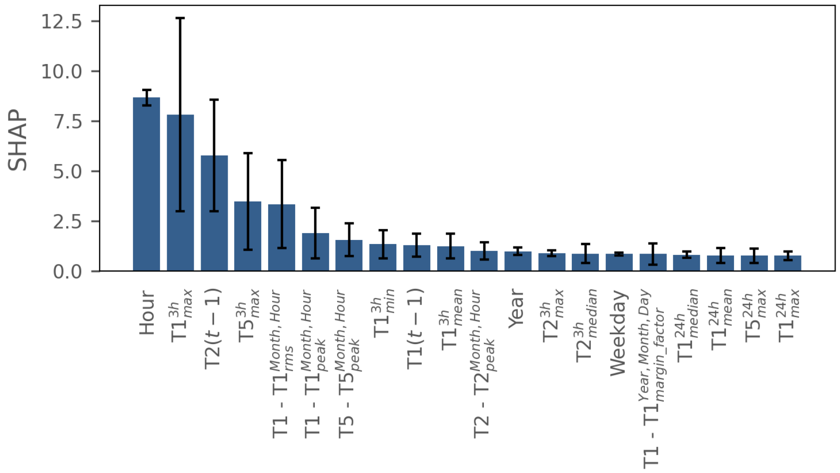


Fig. 4. Out-of-fold Top-20 Features Importance obtained after the last incremental step of feature engineering (*aggregated features*) and feature selection, for LDC1 at Round 1. On the y-axis, we reported SHAP (SHapley Additive exPlanations) [26] values of the LightGBM model.

Regularization Using Dropout, the DART booster reduced the overfitting that affects LightGBM with the standard booster. It also reduced the prediction error, but training became slower since it required more boosting iterations. We tested DART on the qualification data only when it was over. With 30'000 iterations, the MAPE (H) went from 3.24 to 2.83, and the Magnitude from 3.21 to 3.09. Hence, we included DART in the final match models.

Temporal hierarchies We built temporal hierarchies by summing the hourly load and temperatures at the following scales: *2-hours, 4-hours, 6-hours*, and *12-hours*. We trained an independent probabilistic LightGBM-LSS [28] model at each time scale. The model minimizes the Negative Log-Likelihood loss function. Gaussian distributional base forecasts were obtained at each temporal scale for the same forecasting horizon H. We implemented probabilistic reconciliation as formulated in [8]. In Tab. 5 (load profile) and Tab. 6 (peak), we compare base and reconciled forecasts, using skill scores ($S_{\%}$); in Fig. 5

we show some forecasts. Temporal hierarchy improves only slightly the point forecasts, but more importantly the predictive distribution, with a skill score of about 5% on CRPS and 10% on IS. We also tested 1-day aggregation without further improvement for the bottom time series. As the previous feature importance analyses showed, past values close to the conditioning time are the most important variables for prediction. We came to the explanation that a high-scale aggregation (empirically greater than 1 day) makes these variables vanish. Instead, small hierarchies also improved peaks, as shown in Tab. 6 and Fig. 5. Given the availability, the metrics we present for the final match refer to actual competition values of Round 1-5 (Jan-Oct 2018).

Results We placed 6^{th} (M), 6^{th} (T), and 7^{th} (S) [16], see Tab. 7.

Table 5. Reconciliation metrics for the *load profiles*; base ($\hat{y}$) and reconciled ($\tilde{y}$) forecasts, with skill scores ($S_{\%}$). Temporal hierarchy for forecasting using hourly (bottom level), 2-hourly, 4-hourly, 6-hourly, and 12-hourly aggregations.

	MAPE			CRPS			$\text{IS}_{90\%}$			$\text{IC}_{90\%}$ (%)	
	$\hat{y}$	$\tilde{y}$	$\mathbf{S}_{\%}$	$\hat{y}$	$\tilde{y}$	$\mathbf{S}_{\%}$	$\hat{y}$	$\tilde{y}$	$\mathbf{S}_{\%}$	$\hat{y}$	$\tilde{y}$
LDC1	4.87	**4.84**	0.75	6.35	**6.03**	5.16	61.62	**55.01**	11.34	99.24	98.81
LDC2	5.02	**4.99**	0.52	10.92	**10.44**	4.49	101.35	**90.39**	11.43	99.07	98.49
LDC3	4.51	**4.5**	0.05	45.99	**43.84**	4.78	446.49	**398.37**	11.39	98.85	98.14

Table 6. Reconciliation metrics for the *peaks*; base ($\hat{y}$) and reconciled ($\tilde{y}$) forecasts, with skill scores ($S_{\%}$). Temporal hierarchy for forecasting using hourly (bottom level), 2-hourly, 4-hourly, 6-hourly, and 12-hourly aggregations.

	Magnitude			Timing			Shape			$\text{CRPS}_{\text{peak}}$		
	$\hat{y}$	$\tilde{y}$	$\mathbf{S}_{\%}$	$\hat{y}$	$\tilde{y}$	$\mathbf{S}_{\%}$	$\hat{y}$	$\tilde{y}$	$\mathbf{S}_{\%}$	$\hat{y}$	$\tilde{y}$	$\mathbf{S}_{\%}$
LDC1	4.97	**4.90**	1.34	1.22	**1.13**	7.93	0.088	**0.086**	2.16	8.33	**7.89**	5.46
LDC2	5.51	**5.48**	0.52	1.26	**1.23**	1.87	0.102	**0.101**	1.11	15.73	**15.13**	3.85
LDC3	4.83	**4.79**	0.95	1.19	**1.09**	8.80	0.079	**0.078**	1.56	60.83	**57.97**	4.82

4 Conclusion

We described our experience in an international energy forecasting competition. We introduced features borrowed from the literature of signal processing, a novel strategy for feature selection, and we pointed out the improvement that the DART booster allowed us to achieve over the traditional Gradient Boosting (GB) of trees. Furthermore, we adopted a recent probabilistic extension of LightGBM. A predictive distribution, instead of the point forecast solely, is of great impact because the decision-making processes can rely on the uncertainty

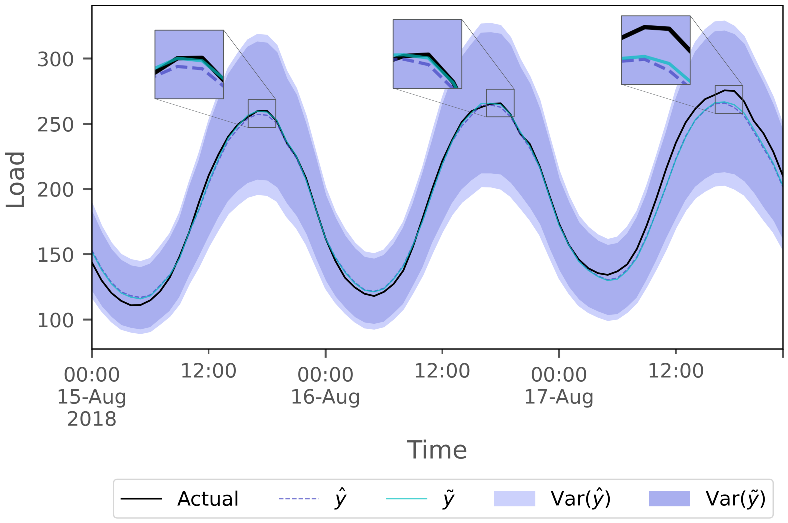


Fig. 5. Comparison of probabilistic forecasts, before and after the application of the temporal hierarchy. The temporal hierarchy slightly improves the point forecasts. It also shortens the prediction intervals without compromising their reliability. The sample refers to three days (15-17 Aug 2018) for `LDC1`.

Table 7. Leaderboard of Final Match [16].

Team	**Rank M.**	**Team**	**Rank T.**	**Team**	**Rank S.**
Amperon	1	KIT-IAI	1	KIT-IAI	1
Overfitters	2	Amperon	2	Amperon	2
peaky-finders	3	BelindaTrotta	3	Overfitters	3
Team SGEM KIT	4	Overfitters	4	X-mines	4
KIT-IAI	5	X-mines	5	SheenJavan	5
swissknife	6	swissknife	6	Rajnish Deo	6
Recency Benchmark	7	peaky-finders	7	swissknife	7
Energy HACker	8	Rajnish Deo	8	*Recency Benchmark*	8
Rajnish Deo	9	Team SGEM KIT	9	RandomForecast	8.5
X-mines	10	SheenJavan	10	Yike Li	8.5
...		...		peaky-finders	10
Tao's Vanilla Benchmark	17.5	*Recency Benchmark*	14	...	
		Tao's Vanilla Benchmark	18	*Tao's Vanilla Benchmark*	16

Team	**Final Rank**
Amperon	1
KIT-IAI	2
Overfitters	3
peaky-finders	4
X-mines	5
swissknife	6
Rajnish Deo	7
Team SGEM KIT	9
Recency Benchmark	10
...	
Tao's Vanilla Benchmark	14

inherent in the forecast. To the limits of our knowledge, these models have not yet been adopted in energy forecasting. Moreover, with distributional forecasts, we applied temporal hierarchies and further improved the results.

For future work, we intend to evaluate our method on other datasets and improve the capabilities of other models, specifically Deep learning models for energy forecasting.

Acknowledgments

Work partially funded by the Swiss National Science Foundation (grant 212164), and the ERA-NET Smart Energy Systems program (grant 883973, project Digicities).

References

1. T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama. Optuna: A next-generation hyperparameter optimization framework. In Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, pages 2623–2631, 2019.
2. J. S. Armstrong and F. Collopy. Error measures for generalizing about forecasting methods: Empirical comparisons. International journal of forecasting, 8(1):69–80, 1992.
3. G. Athanasopoulos, R. J. Hyndman, N. Kourentzes, and F. Petropoulos. Forecasting with temporal hierarchies. European Journal of Operational Research, 262(1):60–74, 2017.
4. G. Bontempi, S. Ben Taieb, and Y.-A. Le Borgne. Machine learning strategies for time series forecasting. Business Intelligence: Second European Summer School, eBISS 2012, Brussels, Belgium, July 15-21, 2012, Tutorial Lectures, pages 62–77, 2013.
5. L. Breiman. Random forests. Machine learning, 45:5–32, 2001.
6. H. Carlens. State of competitive machine learning in 2022. mlcontests.com/state-of-competitive-machine-learning-2022/, 2022. Accessed: 2023-04-01.
7. N. Charlton and C. Singleton. A refined parametric model for short term load forecasting. International Journal of Forecasting, 30(2):364–368, 2014.
8. G. Corani, D. Azzimonti, J. P. Augusto, and M. Zaffalon. Probabilistic reconciliation of hierarchical forecast via bayes' rule. In Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2020, Ghent, Belgium, September 14–18, 2020, Proceedings, Part III, pages 211–226. Springer, 2021.
9. T. Duan, A. Anand, D. Y. Ding, K. K. Thai, S. Basu, A. Ng, and A. Schuler. Ngboost: Natural gradient boosting for probabilistic prediction. In International Conference on Machine Learning, pages 2690–2700. PMLR, 2020.
10. H. Erişti, A. Uçar, and Y. Demir. Wavelet-based feature extraction and selection for classification of power system disturbances using support vector machines. Electric power systems research, 80(7):743–752, 2010.
11. J. H. Friedman. Stochastic gradient boosting. Computational statistics & data analysis, 38(4):367–378, 2002.
12. P. Gaillard, Y. Goude, and R. Nedellec. Additive models and robust aggregation for gefcom2014 probabilistic electric load and electricity price forecasting. International Journal of forecasting, 32(3):1038–1050, 2016.
13. T. Gneiting. Quantiles as optimal point forecasts. International Journal of forecasting, 27(2):197–207, 2011.
14. T. Gneiting, A. E. Raftery, A. H. Westveld, and T. Goldman. Calibrated probabilistic forecasting using ensemble model output statistics and minimum crps estimation. Monthly Weather Review, 133(5):1098–1118, 2005.
15. B. Gregorutti, B. Michel, and P. Saint-Pierre. Correlation and variable importance in random forests. Statistics and Computing, 27:659–678, 2017.
16. T. Hong. BigDeal Challenge 2022, Final Match. blog.drhongtao.com/2022/12/bigdeal-challenge-2022-final-leaderboard.html. Accessed: 2023-04-09.

17. T. Hong. BigDeal Challenge 2022, Qualifying Match. blog.drhongtao.com/2022/11/bigdeal-challenge-2022-qualifying-match.html. Accessed: 2023-04-09.
18. T. Hong, P. Pinson, and S. Fan. Global energy forecasting competition 2012, 2014.
19. T. Hong, P. Pinson, S. Fan, H. Zareipour, A. Troccoli, and R. J. Hyndman. Probabilistic energy forecasting: Global energy forecasting competition 2014 and beyond, 2016.
20. T. Hong, J. Xie, and J. Black. Global energy forecasting competition 2017: Hierarchical probabilistic load forecasting. International Journal of Forecasting, 35(4):1389–1399, 2019.
21. R. J. Hyndman and S. Fan. Density forecasting for long-term peak electricity demand. IEEE Transactions on Power Systems, 25(2):1142–1153, 2009.
22. G. James, D. Witten, T. Hastie, and R. Tibshirani. An introduction to statistical learning, 2013.
23. T. Januschowski, Y. Wang, K. Torkkola, T. Erkkilä, H. Hasson, and J. Gasthaus. Forecasting with trees. International Journal of Forecasting, 38(4):1473–1481, 2022.
24. N. Kourentzes and G. Athanasopoulos. Elucidate structure in intermittent demand series. European Journal of Operational Research, 288(1):141–152, 2021.
25. J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu. Feature selection: A data perspective. ACM computing surveys (CSUR), 50(6):1–45, 2017.
26. S. M. Lundberg and S.-I. Lee. A unified approach to interpreting model predictions. Advances in neural information processing systems, 30, 2017.
27. A. März. Xgboostlss–an extension of xgboost to probabilistic forecasting. arXiv preprint arXiv:1907.03178, 2019.
28. A. März and T. Kneib. Distributional gradient boosting machines. arXiv preprint arXiv:2204.00778, 2022.
29. N. Meinshausen and G. Ridgeway. Quantile regression forests. Journal of machine learning research, 7(6), 2006.
30. L. Nespoli and V. Medici. Multivariate boosted trees and applications to forecasting and control. Journal of Machine Learning Research, 23(246):1–47, 2022.
31. S. Smyl and N. G. Hua. Machine learning methods for gefcom2017 probabilistic load forecasting. International Journal of Forecasting, 35(4):1424–1431, 2019.
32. S. B. Taieb and R. J. Hyndman. A gradient boosting approach to the kaggle load forecasting competition. International journal of forecasting, 30(2):382–394, 2014.
33. R. K. Vinayak and R. Gilad-Bachrach. Dart: Dropouts meet multiple additive regression trees. In Artificial Intelligence and Statistics, pages 489–497. PMLR, 2015.
34. Y. Wang, S. Sun, X. Chen, X. Zeng, Y. Kong, J. Chen, Y. Guo, and T. Wang. Short-term load forecasting of industrial customers based on svmd and xgboost. International Journal of Electrical Power & Energy Systems, 129:106830, 2021.
35. S. L. Wickramasuriya, G. Athanasopoulos, and R. J. Hyndman. Optimal forecast reconciliation for hierarchical and grouped time series through trace minimization. Journal of the American Statistical Association, 114(526):804–819, 2019.
36. J. Yu. Bearing performance degradation assessment using locality preserving projections and gaussian mixture models. Mechanical Systems and Signal Processing, 25(7):2573–2588, 2011.