NANYANG TECHNOLOGICAL UNIVERSITY

FINAL YEAR PROJECT REPORT

SCE16-0523

Electricity Price Forecasting using Ensemble Neural Networks

Supervisor: Ast/P Bo An

Examiner: Ast/P Lana Obraztsova

Author: Nainan Abhay George

Submitted in Partial Fulfillment of the Requirements
for the degree of Bachelor of Engineering in Compuer Science

School of Computer Science and Engineering
November 2017

Abstract

Computational Intelligence models are the newest family of models to tackle the research problem of Electricity Price Forecasting (EPF). This family of models consists of feed-forward, recurrent(RNN), and fuzzy neural networks. Since these forecasting models are non-linear and can train on sequences of input vectors, they learn more complex functions and predict more accurately than multi-agent, supply-demand, and statistical family of models.

This research project tackles the *Complatt SmartWatt* competition in 13th International Conference on the European Energy Market, EEM2016. A subset of the competition involves predicting one day-ahead (D+1) hourly electricity prices (euros/Mw) for the Iberian Electricity Market. The project evaluates different RNN architectures to minimize the Mean-Average Error (MAE) of the D+1 Electricity Prices.

The project successfully creates an Ensemble model that ranks 5th out of 45th with an MAE of 2.85, thus performing as well as existing state-of-the-art prediction models devised by other conference participants. The two final models were Long Short-Term Memory(LSTM)and Extreme Gradient Boosting (XGBoost) using endogenous (price) and exogenous(time-of-day, temperature) features. The two key innovations are creating a novel model evaluation metric called *Shift-Day MAE* and based on each models*Shift-Day MAE* performance, using the *Feature Weighted Linear Stacking* technique to build an Ensemble model out of the two existing models. The ensemble model is passed the predictions of the two existing models, as well as the time of the day at which the price is predicted. The ensemble learns the times-of-days when prices spike or change direction, and accordingly weighs the input predictions.

Lastly, it releases an EPF data-engineering and testing framework called *Lightening.* This command-line-utility addresses the research community's concern about a universal testing ground for different EPF models by letting researchers customize training and testing data by changing configurations inside a JSON file, and submitting their model through CLI for automated evaluation.

Keywords: *Electricity Price Forecasting, Long Short-Term Memory Neural Networks, Gated Recurrent Unit, Non-linear Autoregressive Networks, Extreme Gradient Boosting,Universal EPF Testing Framework*

Acknowledgements

I would like to express my sincerest gratitude to to the following people for their guidance and support throughout the research project.

Nanyang Assistant Professor Bo An, for his consistent support and valuable insights into the research problems tackled in this report.

Assistant Professor Lana Obraztsova for her time and attention in the perusal of this report, and the constructive feedback provided on improving the report.

Contents

List of Tables

List of Figures

1 Introduction and Problem Statement

Electricity Price Forecasting (EPF) is a cross-domain research area that has grown in significance since the early 1990s. It draws on tools and techniques from fields as diverse as Econometrics, Game Theory, Machine Learning, and Financial Mathematics to forecast daily, weekly, and monthly price levels and trends.Research on EPF, especially using non-stationary computational intelligence methods such as Machine Learning is increasingly appearing in leading general forecasting journals such as the International Journal of Forecasting.

1.1 Significance of Research Problem

Electricity is a non-storable commodity. Power transmission systems require a constant balancing of electricity production and consumption, since any oversupply above some threshold will cause transmission wires to break down. The transmission corporation thus heavily penalizes the producers if they oversupply. Consequntly producers heavily slash prices to clear out their supply since turning off production is expensive. This creates significant volatility in electricity prices. Price forecasting has generated strong commercial interests from state-run electricity utilities,power portfolio-managers, and electricity derivatives traders. With price volatility an order of magnitude higher than financial or commodity markets, volume and price hedging are no longer good risk-management practices, but are necessities to prevent significant losses or bankruptcy. Accurate demand and price forecasting is thus critical for producers to maintain profitability.

1.2 Problem Statement

The 13th European Energy Market Conference, EEM 2016, hosted the *Complatt SmartWatt* Electricity Price Forecasting Competition in 2016. Participants are to build prediction models that forecast the hourly spot price of the Iberian Electricity Market, MIBEL, on a daily rolling basis, for the 24 hours of the 5 days ahead i.e. (D+1) to (D+5). This project focuses solely on day-ahead (D+1) forecasts, which can be defined as follows:

Given a Time Series Vector of previous hourly prices:

$$x_{t-m}^{t} = [x_{t-m}, x_{t-m+1}, ..., x_t]$$

generate a model that maps this vector to the 24 future hourly prices:

$$x_{t+24}^{t} = f(x_{t-m}^{t}, \theta)$$

such that the mean-absolute error (MAE) cost function is minimized:

$$\arg\min_{\theta} \frac{\sum_{i=1}^{24} |x_{i,pred} - x_{i,true}|}{24}$$

1.3 Competition Goal and Model Evaluation Procedure

The Mean Average Error (MAE) of the top five submissions range from 2.35 to 3.00, and thus we establish the baseline as $MAE = 3.00$ as the aim is to break into the top 5 ranks. Ultimately the project builds an ensemble neural network combining multi-layer Long Short-Term Neural Networks with an Extreme Gradient Boosted model to create an ensemble model with an MAE of 2.85. This successfully places the model amongst the top five submissions.

Given that models performance can vary across different market regimes, comprehensive testing of model performance is required. This project uses the 10-Fold Cross Validation to calculate the average model performance. It should be assumed that any model performance data mentioned in the remainder of the report is the average performance data extracted after performing 10-Fold Cross Validation. The cross-validation estimate of prediction error for *K-fold* cross-validation is given as follows:

$$CV(\hat{f}) = \frac{1}{N} \sum_{i=1}^{N} L(y_i, \hat{f}^{-K(i)}(x_i)) \tag{1}$$

where $\hat{f}^{-K(i)}(x)$ is the prediction function fitted with the *Kth* part, and L is the loss function of the model i.e. MAE in our case.

1.4 Model Baselines and Approach Summary

The competition data-set contains hourly endogenous(price), and exogenous(solar, hydro-electric electricity supply) data for two years. Feature extraction is performed on the data-set. For exogenous variables, Principal Component Analysis is used to fundamentally understand the relationship between price and factor such as solar and hydro-electric, and also reduce the number of dimensions. tatistical features comprise of Mean, Historical Volatility, Variance, Skewness, Kurtosis, Turning Points, and Shannon Entropy. Wavelet-denoised features are underutilized in financial time-series denoising, and thus can be considered a research novelty.

Simple LSTM models are trained on these three feature sets to establish the baseline. The aim is to iteratively optimize these models by tuning the hyperparameters and the features themselves, and lastly create ensembles to beat the baseline models, and reach the competition goal of MAE ¡ 3.00. This is a very iterative process, hence the section titled *Experimentation, Results, and Analysis* is structured to show the reader the problems with a model, and how the problem is dealt with to arrive at a satisfactory model.

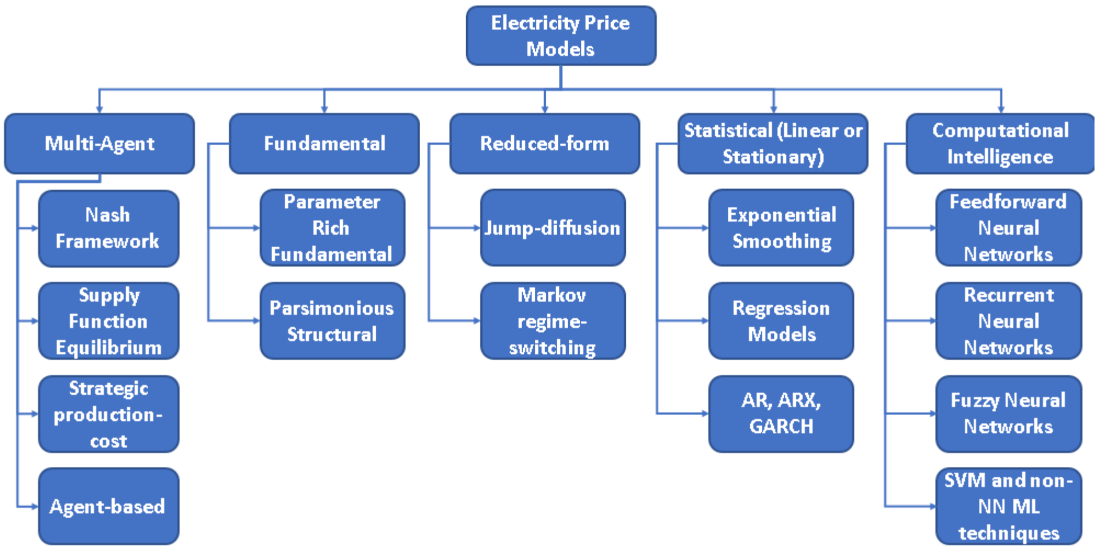
After establishing the baseline models for each feature-set, the following models are fitted to the feature-sets: LSTM, GRU, RNN, XGBoost, and NARX. The best model for each data-set is then selected. However, this is not sufficient to reach the goal.

Model optimization is done by input-replication and using brute-force to find the optimal number of LSTM layers, as well as the optimum look-back period to train the LSTM. Then, the research novelty: *shift-day-MAE* is introduced to better understand the model performance. The input training vectors are modified to minimize the shift-day MAE loss function, instead of the conventional MAE loss function. Lastly, an ensemble layer using LSTM and XGBoost is created using the *linear-weighted stacking technique*, which looks at the time-of-day of the forecast interval, and accordingly weights the outputs of LSTM and XGBoost. This ensemble model achieves an MAE of 2.85, beating all baseline models, and reaching the research goal of MAE ¡ 2.85.

1.5 Building an Open Source EPF Testing Framework

Weron[45] argues heavily about inconsistency in the use of datasets, software implementations of multi-parameter models. Differnt EPF competitions use different time-frames and metrics such as Mean-Absolute-Error, Mean-Absolute-Percentage-Error, or Root-Mean-Square-Error to evaluate models. For example, Misiorek et. al claim that Markov Regime-Switching models are poor forecasters[47], but in the same year, Kosater and Mosler[27] praise its accuracy, while testing it in a different market and for a longer duration.

Lightening is a Python based data-engineering and testing command-line utility created by the author to automate the error-prone and time-consuming nature of setting up testing and training data, and feeding it to the model. *Lightening* uses JSON files as configuration files to store data engineering and model testing parameters. Users can simply specify the file location of the model, the date-range needed for testing, the price series that is needed for training and testing, and metrics such as MAE, RMSE, that are to be evaluated, among other parameters. *Lightening* automatically creates the training and test cases and stores them as *Numpy* objects. This allows the researcher to independently build the model, and simply specify the training shape in the *Lightening* configuration file. *Lightening* will then automatically call the model, train it, test it, and print a report on the model performance.



Electricity Price Models
Multi-Agent
Nash Framework
Supply Function Equilibrium
Strategic production-cost
Agent-based
Fundamental
Parameter Rich Fundamental
Parsimonious Structural
Reduced-form
Jump-diffusion
Markov regime-switching
Statistical (Linear or Stationary)
Exponential Smoothing
Regression Models
AR, ARX, GARCH
Computational Intelligence
Feedforward Neural Networks
Recurrent Neural Networks
Fuzzy Neural Networks
SVM and non-NN ML techniques

Figure 1: EPF Model Taxonomy

2 Literature Review and Model Comparison

Broadly, EPF models can be split into five orthogonal models, namely Multi-agent, Fundamentals, Reduced-form, Statistical, and Computational Intelligence based methods[45]. The literature review briefly describes the methods, strengths, and weaknesses of the major families of models, before investigating deeper into the Computational Intelligence class of models. The approaches used by participants in the EEM2016 competition are discussed under the relevant sections. Unfortunately, there are very few EEM2016 competition specific publications, hence the review focuses more on understanding the frequency, market environment, and forecasting problems each family excels at.

2.1 Multi-agent Models

Multi-agent models are simulation based models that mimic a heterogenous set of agents making decisions in an electricity market environment with a bound set of parameters and rules [43]. They allow for extensive configuration and testing of an environment, and produce rich data-sets for understanding agent interactions and their impact on prices [44]. However, they are prone to modelling risk, given the extensive number of assumptions made in defining agent behaviour. Another key problem is the size of the ”market” that the simulations fail to recreate [15]. It is difficult to map the agent behaviours from the simulation to real-world entities and use that to predict real-world prices [40].

2.2 Fundamental Models

Fundamental models attempt to model the physical and economic functions that exist between raw materials, alternative electricity supplies, and consumer demand. Pure fundamental models fall into parameter-rich models, and parsimonious structure models. Parameter-rich models are best reflected in the models of Johnsen(2001) who presents a supply-demand framework for Norwegian electricity markets that was succesful in medium-term trend prediction, but was unsuccesful on a shorter timeframe, and especially inaccurate in predicting hourly prices. [20]. Parsimonious Structural model researchers focus on building simpler models to try and fit empirically seen supply and demand price curves. Kanamura and Ohashi (2008) create hockey-stick supply curves using Box-Cox and Ornstein-Uhlenbeck processes, that is able to account for price volatility and discontinuous price jumps [21]. Coulon and Howison(2009) build stochastic demand functions by empirically observing the movement of bids and asks in the electricty price order-book in order to predict jumps in prices [10]. These models are more successful than parameter-rich models because they can accurately predict trends on a weekly basis.

The major strengths of fundamental models over multi-agent models is the fact that they have clear techniques that help with price forecasting. However, the key challenges are that the forecasting time-frame was largely monthly, and has very recently been able to forecast accurately on a weekly timeframe. Nonetheless, supply and demand modelling is ineffective on an intraday basis, failing to capture intra-day seasonality and regime changes [4]. Another concern is the set of assumptions that are built into these models, especially through formulae that map fundamental parameters to electricity prices. These formulae are also subjected to regime-changes, and pose significant modelling risk [5].

2.3 Statistical Models

Statistical models comprise econometric, and technical analysis techniques to forecast prices relying largely on previous values of price and other correlated variables such as electricity load, and solar or hydro-electric output.[12]. These range from more intuitive models such as similar-day models, which compare price action of the forecasting interval to price action seen previously, to the more complex autoregressive models such as Seasonal ARIMA, which try to identify seasonality across different frequencies (daily, monthly) and regress previous prices to find a function to predict the next set of prices[9].

Regression models are the de-facto model for modelling auto-correlation [8]. Regression models are used by feedforward neural networks with linear activation functions, that find the optimal weights for a sequence of past inputs to predict future prices [37]Assuming a linear relationship between input and output variables:

$$P_t = \mathbf{W}\mathbf{X_t} + \epsilon_t \tag{2}$$

Kim, Yu, and Song(2002) utilize wavelet decomposition coupled with regression on a sequence of previous prices to predict intra-day trends and prices with far better accuracy than the supply-demand function models discussed previously [24]. They attribute the accuracy to the smoothening of the signal by wavelet decomposition, whilst retaining the scale of local structures such as peaks and troughs. This final year project follows from their lead in using wavelet decomposition, but aims to build non-linear models from them. It is hypothesized that the succesful denoising seen by using wavelet decomposition will let neural network models better learning the seasonality of the models. Another set of statistical models that have seen exceptional success are ARIMA and GARCH-based models as evaluated by Koopman, Ooms, and Carnero in 2007 [26]. The authors succesfully demonstrated the ability of ARIMA and GARCH models to capture yearly, monthly, and daily cycles in spot prices. These auto-regressive techniques were enhanced by Bunn et. al (2008) when they used ARX models i.e. autoregressive models with exogenous variables to predict 24 hours ahead electricity prices in the British markets [22]. ARX models serve as the inspiration for our use of NARX Neurasl Networks or Non-Linear Autoregressive Neural Networks with Exogenous variables. Bunn concludes that his ARX models that "incorporate time-varying coefficients and exogenous models exhibit the best predictive performances amongst other alternatives" [22].

The succes of statistical models in delivering EPF accuracy by just price data is what makes it a valuable forecasting tool [47]. This is because other models families so far have depended strongly on exogenous inputs, the prices or levels of which do not change at the speed of the prices. Additionally, if we make the assumption the electricity traders are informed participants just as in financial markets, then, the information derived from exogenous variables could already be incorporated in the price [29]. The weaknesses of statistical models is reflected in the way they lose accuracy during electricity price spikes. Weron and Misiorek in 2008 point out the presence of a reset-period for linear models where they only regain accuracy once markets have stabilized and a proper trend has formed again [46].

2.4 Neural Network Models

Neural Network models are the most exciting family of EPF models largely due to their ability to learn complex trends and patterns in datasets, and replicate any linear or non-linear function given a sufficient number of hidden layers [19]. Feedforward Neural Networks are the simplest Neural Networks and simply perform linear Regression using least-mean-square methods to find the optimal set of weights. Adding *hidden layers* or a set of nodes that are neither input nor output nodes allows them to further learn non-linear relationships. Complex feedforward architectures have been used by Huang (2005) [17], Keynia and Amjady(2008) [23], and Shafie-Khah et. al. (2011) to moderate success in 24 to 120 day ahead price prediction [38].

Since Feedforward Neural Networks cannot produce more than one set of values, they are not able to learn functions that predict a sequence of values [18]. Recurrent Neural Networks incorporate loops into its layers, so that each time a new input is provided, the layers enter a new state, even if the same input is provided multiple times. This allows RNNs to pick up temporal features [36]. For example, near key support levels of financial instrument prices, RNNs are able to learn that prices can rebound from those supports, because it learns from price action before and after the price reaches the levels. Feedforward neural networks are shown to predict continued declines more often, because they are not learning information based on how the price is fluctuating during the last few inputs [25].

Anbazhagan and Kumarappan(2013) show that pure RNNs using price data are able to perform better than ARIMA, wavelet-ARIMA, Feedforward, and other linear statistical models [1]. However, Sharma and Srinivasan(2013) who combine statistical methods with RNNs note rapid performance degradation once the forecasting interval is extended to more than a few days [39]. This is because of the *vanishing gradient problem* where the error gradient that has been derived using information from some input at time $t - q$, degrades exponentially from $t - q$ to t. Created by Lin,Horne,Tino, and Giles(1996) NARX (Non-Linear Autoregressive with Exogenous Variables) models are shown to overcome the vanishing gradient problem, and this is shown in this research report, where NARX performance degradation is much smoother as the forecasting interval is increased [28]. Chaabane(2014) shows how NAR, (NARX, but without exogenous variables), are able to show better accuracy than RNNs incorporating exogenous variables, for week-ahead forecasts [6]. Azadeh et al. (2013) use LSTMs which are similar to NARX in that they overcome the vanishing gradient problem by choosing which information to learn and which to forget [2]. The performance of their stacked LSTM models is comparable to NARX models.

2.5 Models of Competition Participants

Specific to the competition, Moreira, Bessa, and Gama [30] use a quantile regression and neural network framework to compute a prediction interval in which they believe the *true* value will lie. Their baseline model is the Linear Quantile Regression, upon which they build the Quantile Regression Boosting, Quantile Regression Forests, and Quantile Regression Neural Networks. Interestingly, their base model (Linear Quantile Regression) is the third most succesful model out of the six models built, with an average RMSE of approximately 7 percent for the 24-hr prediction interval. As expected, the Quantile Regression Neural Networks outperform linear regression, but not by a large margin, as they report an RMSE of 6.3 percent.

Goncalves and Saraiva[34] investigate the impact of secondary power sources called *feed-in generation* and hourly prices through a parsimonous-structural model belonging to the fundamental family of EPF models. They find strong evidence that feed-in generators begin supplying electricity when the prices reach higher price intervals, i.e. they find evidence about generators ability to time the market. This could be because such generators need to sell at the relatively higher price bracket to break even in operating costs. They further show that this sudden inflow from such generators rapidly pushes the supply curve to the right, and is partly responsible for volatility since they are inducing price jumps. However, they do not apply the algorithms to the competition, and thus do not provide comparable prediction scores.

Freitas and Lagarto[13] focus on incorporating the cyclicality of electricity prices and electricity production (exogenous sources) to predict prices more accurately. They rely on the field of *circular statistics* that models observations as directions through angles. They then use the *Von Mises Distribution* to predict the maximum day-ahead price interval of the next 24-hrs. Additionally, they show a correlation between the hours where the maximum electricity falls, and the hours where hydro-electric power supply is maximized.

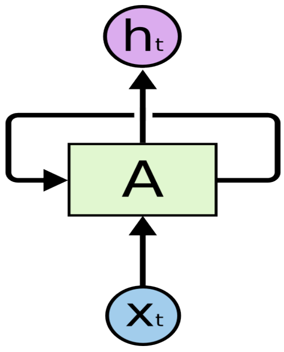


Figure 2: Loops in RNNs allow for information persistence
[33]

3 Long Short-Term Memory Neural Networks

Traditional neural networks do not have *persistence* in the sense that during price prediction the output of each input vector is dependent solely on that input vector and not what vectors were passed to it before. During model training, traditional feed-forward neural networks update their weight layers based on the input vector itself, not the relation between the input vector and the previous input vector.

3.1 Recurrent Neural Networks

Recurrent neural networks solve this problem by introducing loops within their layers, and this allows information (not just weights) from an input vector to persist even when another input vector is introduced at another time-step

Unfortunately, this persistence is fleeting, and while conventional RNNs are able to harness persistence or context in the near past, retaining context from a long-time ago is empirically shown to be limited(Hochreiter 1991) (Bengio, et. al 1994). This makes traditional RNNs unsuitable for EPF, given that seasonality can be seen on monthly-levels. For example, models predicting downward weekly trends as we transition into the holiday period in December, should not forget about the seasonal up-trend in electricity prices that the winter season creates in the Northern Hemisphere, due to increased need for heating, and limited output from solar farms. This sort of context awareness could enhance the predictive power of our models.

Long Short-Term Neural Networks (LSTMs), created by Hochreiter and Schmidhuber in 1997, have a looped structure of neural networks similar to RNNs, but instead of one neural network layer with an activation function, LSTMs have

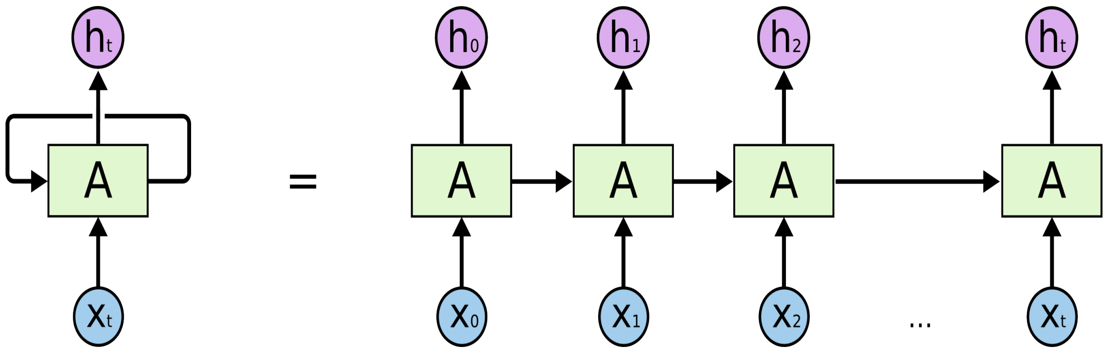


Figure 3: Equivalent Structure of Rolled RNN
[33]

four neural networks that act as *gates* to control the *cell state* of the LSTM. An LSTM cell has three gates, which are composed of a *sigmoid* neural network with an activation function (this generates the output) and a multiplication operator (this controls how much each cell in the loop can contribute to the cell-state)

The first gate is called the *Forget Gate* and uses the equation below to decide the information that needs to be discarded from the cell state. This is similar to the human mind prioritizing what is critical to learn, and what is irrelevant or just noise.

3.2 Long Short-Term Neural Networks

$$f_t = \sigma(W_f) \cdot [h_{t-1}, x_t] + b_f) \tag{3}$$

f_t ranges from 0 to 1. The *forget gate* evaluates the previous output h_{t-1} and the current input x_t to decide how much to forget.

The second gate is called the *input gate layer* and is composed of two neural network layers, namely the *sigmoid* gate and the *tanh* gate. The *sigmoid* gate selects the values of the neural network cell that need to be updated[33]:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{4}$$

Next, the *tanh* gate is used to create a new set of values to be added to the state:

$$\widetilde{C} = tanh(W_i \cdot [h_{t-1}, x_t] + b_C) \tag{5}$$

So far, the two gates have selected the amount of information to remember in the cell state, and the amount of new information ($\widetilde{C_t}$) to be added to the cell state. The cell state is updated through[33]:

$$C = f_t * C_{t-1} + i_t * \widetilde{C_t} \tag{6}$$

Lastly, having updated the cell state, the cell is required to produce an output, which is dependent on the cell state. Like conventional neural networks, the cell state is passed through an activation function, called *tanh* which normalizes the state to within [-1,1]. The normalized cell state is used as a weight to be multiplied by the input x_t to produce h_t:

$$h_t = \sigma(W_o[h_{t-1}, x_t] + b_o) * tanh(C_t)[33] \tag{7}$$

4 Wavelet Feature Extraction

4.1 How do Frequency-based Denoising Methods Work?

Chong Tan has provided a thorough mathematical foundation to understand the mechanisms of wavelet decomposition, and its use in denoising [41]. Denoising time-series data is important for the model to better learn price seasonality across different frequencies, without trying to learn and predict noise.This is especially important for EPF given the inherent market volatility, and spikes and jumps in prices.

Traditionally, Fourier Transforms have been the dominant analytical tool for frequency domain analysis, especially it's discrete version, known as the Fast Fourier Transform (FFT). The Fourier expansion is given by:

$$F(\omega) = \int_{-\infty}^{\infty} f(t)e^{-jt\omega}\, dt \tag{8}$$

They belong to the family of frequency thresholding techniques, which break down a function into its frequency co-efficients, remove frequencies higher than a particular threshold, and inverse the transform to produce the denoised dataset.

This frequency thresholding can be *hard-thresholding*:

$$\upsilon(\alpha) \quad \upsilon(\alpha) \geq \mu \tag{9a}$$

$$0 \quad \upsilon(\alpha) < \mu \tag{9b}$$

or *soft-thresholding*

$$\upsilon(\alpha) - sign(\upsilon(\alpha))\mu \quad \upsilon(\alpha) \geq \mu \tag{10a}$$

$$0 \quad \upsilon(\alpha) < \mu \tag{10b}$$

Where the *sign* function is an activation function that extracts the sign of the input function, but other activation functions can also be used. Empirical research suggests that Stein's Unbiased Risk Estimate (SURE) method gives an optimal μ of:

$$\mu = \sqrt{}(2\ln(n\log_2(n))) \tag{11}$$

where n is the length of the electricity price signal[41].

4.2 Why use Wavelet Transforms for Denoising?

Fourier Transforms have two major limitations with respect to this data-set:

1. Fourier Transforms can only provide frequency resolution, not time resolution, as a result, it cannot provide information on how the spectrum changes relative to time. This is a strong limitation given how regime-changes in trends happen on an intra-day, and intra-week basis.
2. Fourier Transforms assume that the input-signal is stationary, but electricity prices are not stationary and show significant auto-correlation as seen in the diagrams below[41].

Wavelets can be defined as functions whose outputs yield an average of zero:

$$\int_{-\infty}^{\infty} \psi(t)\, dt \tag{12}$$

Essentially, Wavelet decompositions attempt to show some time-series function as a superposition of a set of basis functions. These basis functions are small waves located across different time-periods in the series, and thus provides information on both the amplitude of the function as well as time. As a result, decompositions can provide information about how the time-series is changing relative to time, which lets it better capture seasonality in the data.

4.3 Mathematics behind Continuous and Discrete Wavelet Transforms

The Fourier transform equation shows the output's dependency on solely the frequency of the electricity prices. Continuous Wavelet Transforms (CWT) creates functions dependent on scale and translation parameters as seen in:

$$\gamma(s, \tau) = \int_{-\infty}^{\infty} \Psi_{s,\tau}(t)\, dt \tag{13}$$

The output at time t, is influenced by the scale and transmission, as seen in the basis function:

$$\Psi_{s,\tau}(t) = \frac{1}{s\sqrt{s}}\psi(t-\tau) \tag{14}$$

The scale parameter is the amplitude of the signal at a particular point of time in the time-series domain. It plays a similar role to the frequency parameter of the FFT. The translation parameter translates the basis function across different time periods, thereby accounting for time-based information. Qualitatively, the wavelet transforms calculate the correlation between the input and the basis function, and thus the scale output is expected to be maximal when the input signal is most similar to the basis function at that point of time. If an input signal is specifically concentrated at particular points of the time-series, then its coefficients will be far higher than that of a weaker input signal that is weakly correlated at multiple periods in the time-series. Filtering on the basis of amplitude removes low amplitude noise, and works similarly to the equation shown for FFT denoising[41].

While CWT allows for overlapping wavelets or basis functions, DWT decomposes a signal into orthogonal wavelets. Additionally, DWT is far less computational expensive than DWT, and is the preferred implementation in statistical computing toolboxes. Discrete wavelets are represented as:

$$\psi_{j,k}(t) = \frac{1}{s_0^j \, sqrt(s_0^j)} \phi(\tau - k\tau_0 s_0^j) \int_{-\infty}^{\infty} \Psi_{s,\tau}(t)\, dt \tag{15}$$

where j and k are integers, s_0 ¿ 1 is a fixed dilation step and the translation factor τ_0 depends on the dilation step

Using DWT, a time-series signal $f(t)$ can be decomposed as:

$$f(t) = \sum i = 1^k \, \lambda_{j-1}(k)\varphi(2^{j-1}t - k) + \sum i = 1^k \, \gamma_{j-1}(k)\psi(2^{j-1}t - k) \tag{16}$$

where the scaling function is:

$$\sum i = 1^k \, \lambda_{j-1}(k)\varphi(2^{j-1}t - k) \tag{17}$$

and the wavelet function is:

$$\sum i = 1^k \, \gamma_{j-1}(k)\psi(2^{j-1}t - k) \tag{18}$$

During decomposition of a signal using DWT, a high-pass and low-pass filter are applied to the signal, where the low-pass removes the higher freuquency components of the signal, or technically the lower scale components of the signal since DWT depends on scale and translation, not frequency. The filtered signal is then downsampled. To retrieve the denoised signal, the filtered signal is upsampled by

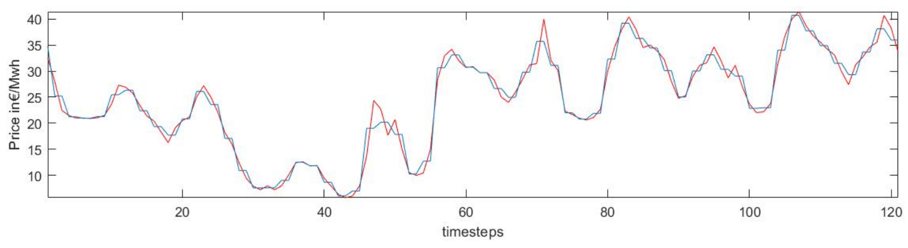


Figure 4: Haar Level 1 using DWT and Soft Thresholding

the same factor as it was downsampled, and the inverse of the filtering is applied, to yield the denoised signal. After each filtering step, a time-series signal $x(t)$ can be represented as[41]:

$$x(t) = A_1(t) + D_1(t) \tag{19}$$

where A_1 is the *approximation* of the original signal i.e. the denoised signal, and D_1 is the coefficients of the frequency. Often one level of denoising is insufficient, as a result, multiple levels of denoising take place. This can be represented through the following equations:

$$\begin{aligned} x(t) =& A_1(t) + D_1(t) && \text{(20a)}\\ =& A_2(t) + D_2(t) + D_1(t) && \text{(20b)}\\ =& A_3(t) + D_3(t) + D_2(t) + D_1(t) && \text{(20c)}\\ =& A_n(t) + D_n(t) + D_{n-1}(t) + ... + D_1(t) && \text{(20d)} \end{aligned}$$

4.4 Feature Quality of Different Wavelets

The graphs in Figures 2-7 show the different ways in which a 1-week, or 1 20-hour time-series signal can be denoised. Visually, it is easy to make out which wavelets perform the best smoothing, while retaining local features such as peaks or troughs at the same scale. The red line represents the original signal, while the blue line represents the denoised signal.

Based on the literature review, the DWT denoising methods applied use soft-thresholding, because the threshold is dependent upon localized features. This is important because otherwise intra-day seasonality will be denoised because it is not as significant as inter-month seasonality.

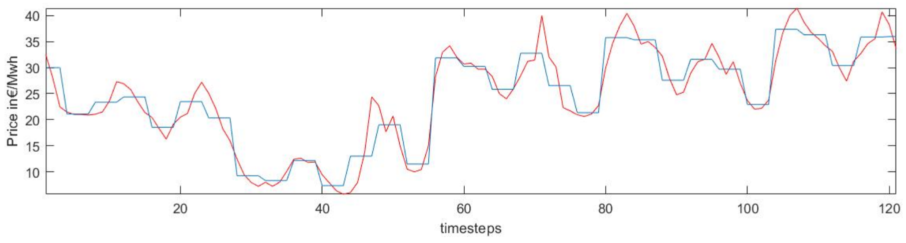


Figure 5: Haar Level 2 using DWT and Soft Thresholding

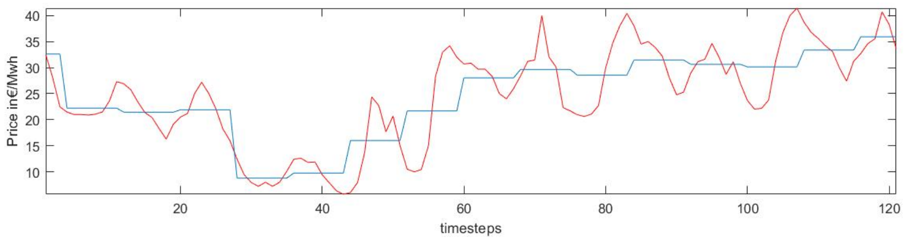


Figure 6: Haar Level 3 using DWT and Soft Thresholding

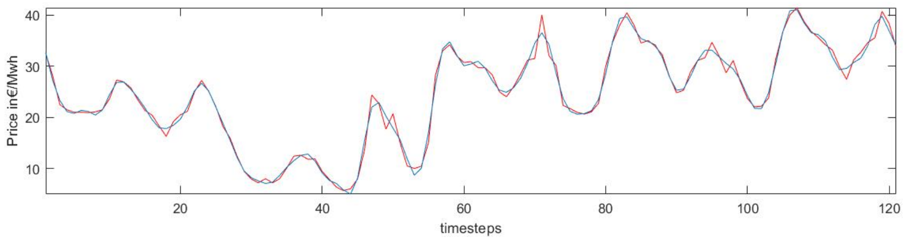


Figure 7: Db40 Level 1 using DWT and Soft Thresholding

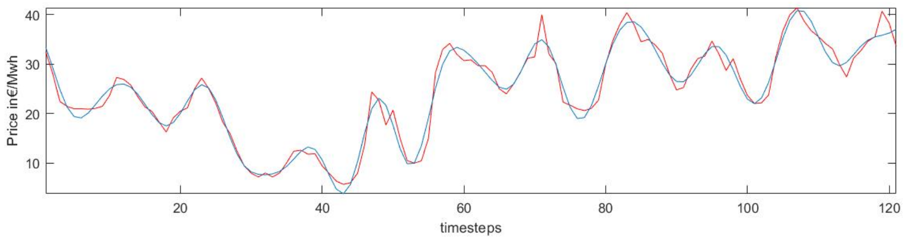


Figure 8: Db40 Level 2 using DWT and Soft Thresholding

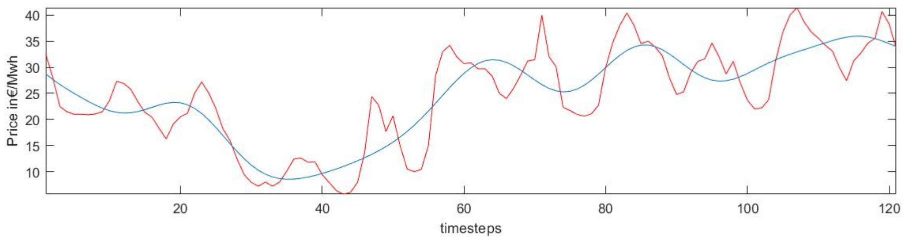


Figure 9: Db40 Level 3 using DWT and Soft Thresholding

Volatility comparisons of the different times series are given in the table below. Db40 wavelets in general do not decrease volatility as much as Haar wavelets. This can be seen through the graphs, as Db40 smoothly approximates local maxima and minima for level 1 and 2, and provides a smooth trend-line for level 3, whereas Haar wavelets model the signal as a discrete function. Despite being a discrete function, Haar wavelets at level 1 and 2 do not lose information about local features, and are able to reduce volatility while preserving features of the time-series. Level 4 and further are not shown because the level of denoising begins to remove intraday seasonality.

Level	StdDev Haar	StdDev Db40
1	13.31	15.39
2	13.94	15.10
3	12.45	13.28

Table 1: Signal Standard Deviation after Denoising

The input signal, along with the denoised signal are passed to a 2-layer vanilla LSTM Neural Network model, as an input vector $\vec{x}_{t-1}$ of $sequence\_length = 24$. All input signals are normalized before training. The model is calibrated to predict day-ahead prices. The models and their MAE scores are shown in the table below. Since different starting weights can lead to different error rates during the testing phase, each model is retrained for $n = 30$ times, and the mean MAE is noted below. Although the MAE scores shown below are far from the range of 2.35 - 3.00 required to place amongst the top 5 results, it is a clear indication of which wavelet methods should be used in more complex models.

Level	Haar	Db40
1	8.97	10.9
2	8.37	13.6
3	9.106	12.6

Table 2: Different Wavelet Model MAE after 20 Epoch training through Vanilla LSTM

5 Feature Engineering and Analysis

Feature engineering and analysis work separately approached two categories of features, namely exogenous features dealing with how correlated environment variables such as solar and hydro-electricity production is related to energy prices, and statistical features which are extracted from the time-series of the price data itself.

The following table lists the properties of the exogenous variables considered in this competition:

Energy Demand:

1. Spain and Portugal Electricity Consumption (Hourly)

Day-Ahead (+24hr) Meteorological Data:

1. Wind Speed (Quarter-hourly)
2. Wind Direction (Quarter-hourly)
3. Temperature (Quarter-hourly)
4. Irradiance (Quarter-hourly)
5. Precipitation (Quarter-hourly)

Energy Generation:

1. Hydro-power Generation (Hourly)
2. Electricity Import/Export (Hourly)
3. Wind Power Generation (Hourly)
4. Photo-voltaic Generation(Hourly)

The variable categories are similar to the datasets used by [28] and [22] when they applied linear statistical models. Bunn [22] indicates that consumer demand variabls and multi-source supply variables are effective in training NARX models.

5.1 Fitting Simplest Model to establish Baseline

A comparison of an autoregressive 2-layer vanilla LTSM model against an autoregressive 2-layer LSTM model that also incorporates the above variables is given in the table below:

AR-LSTM	ARX LSTM	ARX Pchng LSTM
8.78	8.55	6.83

Table 3: MAE comparison of AR and ARX models after 20 Epoch Training through Vanilla LSTM

The MAE scores of AR-LSTM are in line with AR-LSTM using Wavelet Denoising methods. Contrary to expectations however are the ARX-LSTM scores, which only offer a marginal improvement over LSTMs. This could be due to the model using normalized raw levels of exogenous variables. Drawing from [11] where pre-processing techniques for financial time-series data is discussed, these levels are converted to percentage changes relative to the previous hour. The ARX-Pchng model trains on the enhanced dataset, and the score improvements are clear. However, the model is still not in line with results generated by EEM2016 competition parctipants [31], where the expected MAE using solely ARX-based models is in the vicinity of 5.00.

To further leverage information present in the exogenous variables, Principal Component Analysis is used to achieve dimensionality reduction. This is important given the entire data-set has merely 8700 data-points comprising 1-year worth of hourly price and exogenous data. Using a traditional 80/20 testing-training split yields close to 6960 training sets. Consequently, the LSTMs fail to learn more complex non-linear relationships. This will be highlighted later on during the evaluation of complex models. Specifically, predicting price transitions between two days sees accuracy drop significantly for the first few predictions of the new. However, the model should learn that a new day has started, given that we supply hour-of-day as an exogenous variable.

5.2 Principal Component Analysis of Exogenous Variables

Applying PCA to the exogenous variable dataset, it is found that the previous price, natural-gas based generation, coal-based generation, and text-hour are the most important features:

The exogenous variable data is folded into two PCA components, and the distribution of price levels based on the two components are plotted below:

The better the clustering of the different bins, the more effective the PCA folding has been. Unfortunately, while the yellow bins are able to be clustered

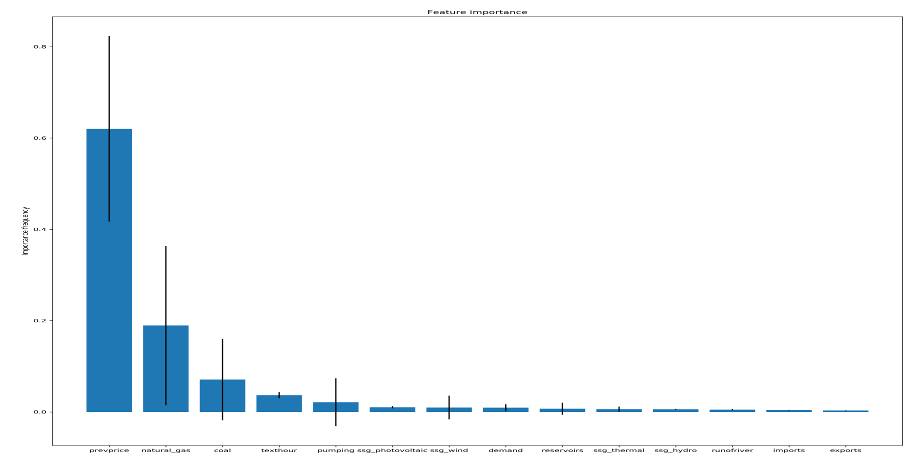


Figure 10: Feature Importance based on PCA Analyis

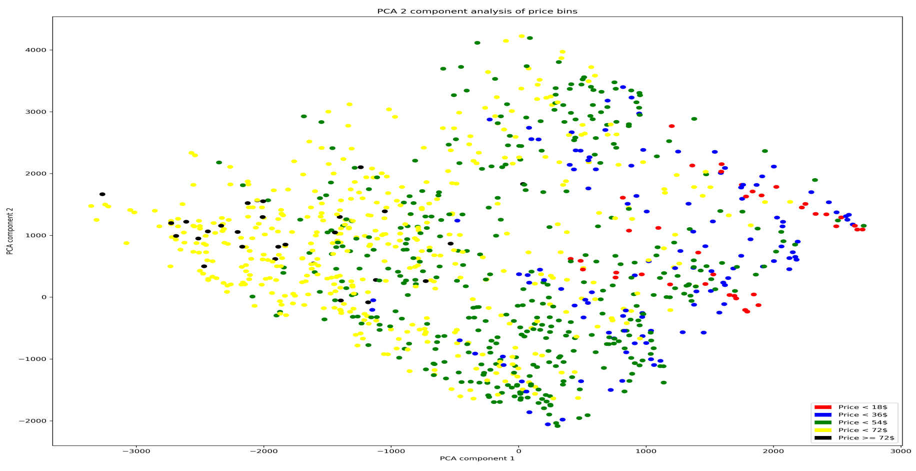


Figure 11: Price bin distribution based on PCA values

well to the right of the grid, it is difficult to separate the blue and red. To ensure better clustering, the last five exogenous variables are dropped, and the PCA folding is done in three dimensions. The resultants PCA components are then used as input vectors to LSTM models. The MAE scores are shown below:

Model	**MAE Score**
AR	8.78
ARX	8.55
ARX Pchng	6.83
ARX Pchng PCA2	5.13
ARX Pchng PCA3	4.24

Table 4: PCA analysis significantly improves ARX feature based LSTM model prediction

The improvement is noteworthy as it yields an MAE of 4.24, placing the model performance among the top 30 competitors, and surpassing the expectations by [16].

5.3 Statistical Feature Analysis

Fiorenzo and Weron [45] generate a variety of statistical features of the time-series data in order to compensate for the small training size. While features such as rolling-mean and volatility are folded into the time-series itself, explicitly specifying lets the model learn more hidden and complex relationships while also understanding the level of current prices relative to global and local maxima/minima. This research draws from [**?**] which use features such as turning points for dimensionality reduction and to anticipate changes in streaming data trends.

The rolling mean can be considered a basic method of smoothing the input signal, and used to determine the direction in which the signal is trending.

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n=[5,12,24]} x_i \tag{21}$$

Standard deviation is a prominent measure of volatility in the data-series. In fact, the standard-deviation of stock-price returns is used as the definition of volatility in option-pricing models of financial instruments.

$$\sigma(\boldsymbol{x}) = \frac{1}{n-1} \sqrt{\sum_{i=1}^{n=[5,12,24]} (x_i - \bar{x})^2} \tag{22}$$

Skewness can provide information on the dominant trend in the window period. When computing skewness, instead of raw hourly price data, percentage change of hourly prices are used. As a result, a positively skewed percentage-change window can indicate a strong up-trend, and vice-versa for a negative skew.

$$\gamma_1 = E\left[\left(\frac{\boldsymbol{X} - \mu}{\sigma}\right)^3\right] \tag{23}$$

Turning points are calculated to indicate to the model what the local landscape looks like. The lack of any turning points should ideally encourage sthe model to predict values in line with the trend, whereas multiple turning points can indicate volatility and likely change in trends.

$$(x_{i+1} - x_i) * (x_i - xi - 1) < 0 \tag{24}$$

In addition to standard deviation, historical volatility is calculated over an exponentially weighted moving average. It is suggested that exponential moving averages provide better information about trends when the time-series shows excessive volatility. Since EPF needs to account for intra-day seasonality, exponential moving averages are faster to react to change in scales. As a result, this measure of volatility needs to be considered in addition to standard deviations.

$$\sigma_n^2 = \lambda\sigma_{n-1}^2 + (1 - \lambda)x_{n-1}^2 \tag{25}$$

$$d^2y/dx^2 \tag{26}$$

Since first-order statistical features are not exceptionally helpful when it comes to actually forecasting very short-term prices [35], Panagiotelis, Anastasios and Smith, Michael conduct dimensionality reduction tests on a large set of statistical metrics and conclude that rate of change of the first and second order are important in determining the appropriate price level, once the model has a general idea of the trend in the forecasting time-period.

6 Experimentation Results and Discussion

The baseline model performance for statistical features, wavelet features, and exogenous features have been established in the feature engineering section. Comparsion of different models, short-listing of potential models and their optimization are detailed in this section:

6.1 Statistical Feature Based Model Performance

To find the baseline model performance for statistical feature based models, simple LSTMs are fitted over different feature-sets. The MAE results are compared to previous models using solely auto-regressive (AR) and AR with exogenous variables (ARX):

ARX-Pchng-LSTM	ARXPchng-PCA	AR-Statistical-LSTM
6.83	4.24	4.63

Table 5: MAE comparison of AR and ARX models after 20 Epoch Training through Vanilla LSTM

The improvement is significant, and already places another model within the top 35 out of 44 contestants. Since wavelet features are technically also statistical features, they can be combined with the above statistical features. The performance of an 8-parameter statistical feature model is presented below:

ARXPchng-PCA	AR-Statistical	AR-Statistical+Wavelet
4.24	4.63	4.47

Table 6: Incremental Advantage from Wavelet Function is limited

The accuracy improvement of incorporating wavelet features is not significant in this example, in contradiction to accuracy gains seen in [24], although [24] uses stochastic wavelet decomposition and works on a different electricity price dataset.

6.2 Feature Extraction using Neural Networks

Keynia and Amjady[23] take statistical feature extraction a step further by trying to predict future feature values for features such as historical volatility and standard deviation, rather than simply calculating the present values, and then

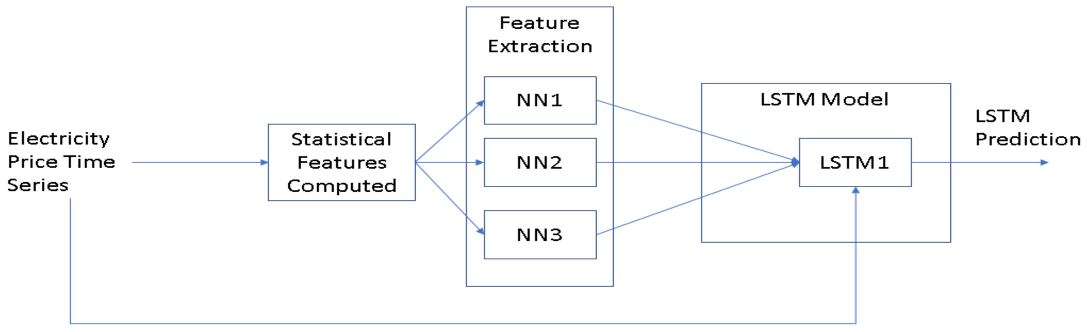


Figure 12: Statistical Feature Extraction Architecture

use them to forecast future prices. These statistical features can themselves be fed into neural networks and the final model can use predicted volatility and standard deviation to make better predictions of future prices. This is because information of the future price is embedded in the value of future volatility and future means.

To test this idea, a *cheating* model is created whose training and testing data consists of one step-look ahead into future volatility parameters. The model accuracy is extremely strong, but surprisingly still not enough to defeat the state-of-the-art forecasting models of the competition:

ARXPchng-PCA	AR-Statistical	AR-Lookahead-Statistical
4.24	4.63	3.48

Table 7: Comparing Lookahead Volatility Parameter Models to Existing Statistical Feature Models

Convinced by the potential gains in accuracy, we train four neural networks to predict future HV, Standard Deviation, Rate of RoC, and Turning Points. The outputs are then fed into the AR-Statistical model, but the new model performs worse than the original model:

ARXPchng-PCA	AR-Statistical	AR-Predicted-Statistical
4.24	4.63	7.84

Table 8: Statistical Feature Extraction using Neural Network Degrades Accuracy

So far AR-Statistical provides the best performance with an $MAE = 4.63$, followed by ARXPchng with an $MAE = 6.83$. Having compared different feature sets have not gotten the performance close to that of state-of-art models.

Advanced techniques such as input-replication and ensemble-models need to be used. Additionally, neural network architectures apart from LSTMs must also be considered.

6.3 Comparing GRU, LSTM, and Simple RNN

Using the aforementioned feature set of AR, ARX, and ARX-Pchng, the MAE scores of each model is calculated over $n = 40$ iterations and averaged to yield the following:

Model Type	AR	ARX	ARXPchng	ARXPchng-PCA
LSTM	8.78	8.55	6.83	4.24
GRU	8.82	8.23	7.12	4.92
SimpleRNN	9.55	10.83	8.33	6.92

Table 9: LSTMs clearly outperform RNNs but barely outperform GRU

Although it may seem that the LSTM is performing marginally better than the GRU, the standard deviations of both the LSTM and GRU MAE scores do not allow for a clear-cut decision on which cell structure befores better. As [https://arxiv.org/pdf/1412.3555v1.pdf] shows, GRUs are very similar to LSTMs in overcoming the *vanishing gradient problem*, which SimpleRNN cells are unable to do.

A key structural difference between the GRU and the LSTM is that the GRU has two instead of the LSTM's three gates. By merging the LSTM's *forget* and *input* gates into a singular *update* gate, GRUs can choose to focus on extremely short-term memory to generate an output. Essentially, GRUs can easily repeat the input layer, whereas LSTMs need to tune the weights of the *tanh* layers before it can reproduce the same results. Given that neither have any distinguishing features with respect to this medium-term price forecasting competition, the heuristic of Occam's Razor is applied, and only LSTM-based neural networks are considered for the remainder of the project. Additionally, sub-optimal models are dropped from comparison at this stage. Testing of new techniques is limited to ARPX-Pchng, ARPX-Pchng-PCA, and AR-Statistical, with AR-Statistical also combining wavelet features

6.4 Input Replication Enhances Model Learning

[32] notes that thinking of training sets in hourly frequency is conceptually incorrect. The paper suggests that in term of information, there are only 365 days worth of data, which can be thought of as 365 samples worth of information, since the competition requires participants to predicts 24-hour ahead. As a result, there are insufficient samples to learn intra-day seasonality of prices.

To increase the sample sizes, the input data is first split into the conventional testing and training datasets. Each 24 hour interval in the training dataset is scaled by a random scalar value in the range $[0.6, 1.4]$. The range parameters are chosen by looking at the standard deviation of the training dataset, and computing the standard percentage deviation. The range corresponds to roughly $\pm 2\sigma$. Note that the testing dataset is not duplicated. If duplicated into training, it is equivalent of giving the model lookahead capabilities. The number of replications and corresponding LSTM model performances are given below:

Replications	**ARXPchng**	**ARXPchng-PCA**	**AR-Statistical**
0	6.83	4.24	4.63
1	6.19	3.98	3.81
2	5.54	3.67	3.77
3	5.42	3.56	3.68

Table 10: Replication Improves Model Performance but marginal improvement decreases, and overfitting risk increases

The number of replications is selected be 2, because input replication can drastically increase the chance of overfitting, since LSTM cells are able to learn sequences, and can find the most similar output sequence to an input sequence during testing, and simply output that. As a result, it does not properly learn the functions and hidden relationships.

6.5 Comparing LSTM Models with NARX Models

LSTMs are supposed to effectively model non-linear relationships, and one way of evaluating existing LSTM model performance is to compare them with Non-linear Auto-regressive Models with Exogenous Variables (NARX). Using the *Neural Networks Toolbox* in MATLAB, the same input vectors of $sequence-length = 24$ are fed to the following model, and its prediction accuracy is recorded:

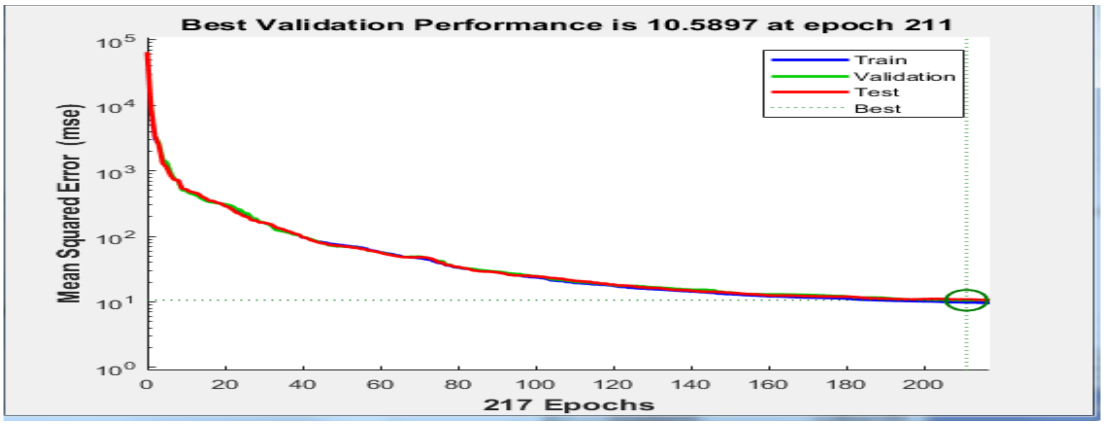


Figure 13: Training of Non-Recurrent NARX Model

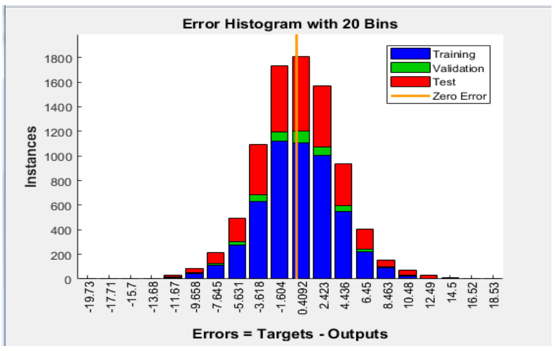
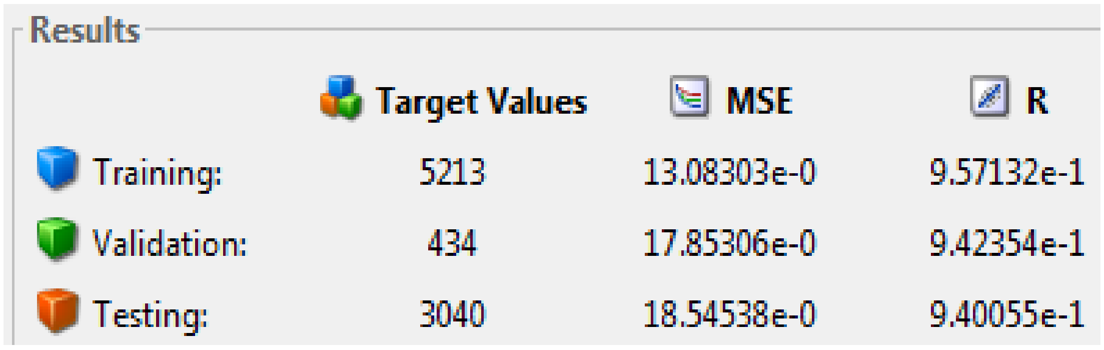


Figure 14: Average Error Histogram


Results

	Target Values	MSE	R
Training:	5213	13.08303e-0	9.57132e-1
Validation:	434	17.85306e-0	9.42354e-1
Testing:	3040	18.54538e-0	9.40055e-1

Figure 15: MAE of 4.24 is similar to LSTM performance

For this research problem, it is clear that LSTMs are able to approximate non-linear functions as well as specially configured NARX models provided in the Neural Netwokrs Toolbox. The NARX is updated using the Scaled Conjugate Gradient back-propagation technique.

6.6 Using Extreme Gradient Boosting (XGBoost)

Friedman's paper on Greedy Function Approximation discusses how to optimize the predictive power of boosted trees[14]. These techniques are enhanced by XGBoost to better solve supervised learning problems. Functions approximating Supervised learning problems are are optimized by using an *objective-function*:

$$Obj(\Theta) = Loss(\theta) + Regularization(\Theta)$$

The Loss() is a gauge of the model's prediction abilities ex. MAE, RMSE, or RMSPROP, while the regularization term controls how complex the model is. There exists a trade-off between the Loss() and Regularization Function known as the bias-variance tradeoff[7]

XGBoost is a greedy algorithm able to locate global optimums for *tree ensemble* models. Tree ensembles are collection of CART (Classification and Regression Trees) where each node represents a decision criteria. [10] gives evidence of how power utilities make decisions on the basis of changes in supply and demand information. This means CART models could be an effective way to arrive at accurate forecasts by generating a series of rules and decision criteria on the basis of exogenous variables.

Using XGBoost's $XGBRegressor()$ function with a replication level of 2, the model is trained on approximately 19,000 samples, and trained on 6000 samples. Out-of-box XGBoost models are surprisingly able to generates better results than LSTMs across most feature sets:

Model	**ARXPchng**	**ARXPchng-PCA**	**AR-Statistical**
XGBoost	3.88	3.35	4.29
LSTM	5.54	3.67	3.77

Table 11: XGBoost outperform LSTMs with minimal parameter tweaking

This is surprising, given that XGBRegressor() only accepts 2D arrays, i.e. it cannot train on a sequence of inputs. Additionally, XGBoost models performs worse on AR-Statistical features than LSTM models, but performs much better on ARX features than LSTM models. Perhaps this is due to CART being a better framework for price decision-making based on exogenous variables, than LSTMs.

It is clear that despite these efforts, the models are unable to breach the barrier of $MAE, \leq 3.00$. The next steps involve building an ensemble model that could combine XGBoost and LSTM predictions in an optimal manner. Before doing so, it is important to evaluate how well the models are learning, which is not the same as comparing prediction metrics. The next section discusses two novel methods that not been used in EPF research, that can estimate how well the models are actually learning complex functions.

7 Model Evaluation

As mentioned before, it is important to evaluate whether the models are actually learning complex functions, or are summing some noise to the input vector and generating an output. Often, poorly calibrated *n-step ahead* models predict values by simply adding noise to the price data of the input vector. In such cases, the performance linearly degrades with each-step ahead. Effective features are needed to help models understand the seasonal, time-dependent, and volatile nature of electricity prices.
A novel metric is defined and computed here namely *shift-day MAE* to better understand how the models are predicting outputs

7.1 Breakthrough using Shift-Day MAE

Shift-Day MAE involves creating test-sets of $sequence_length = 24$ where the first 12 hours are derived from Day 1 (20:00 to 08:00) and the next 12 hours from Day 2 (08:00 to 20:00). The hypothesis is that the model accuracy drops significantly when switching from one regime to another, and then adjusts back upwards once fully into another regime. Mornings usually represent a regime-shift because of massive load changes as society transitions from minimal demand to peak demand very quickly. This idea has its roots in [42] where the authors discuss Markov Regime-Switching models. The graph below plots the MAE for each hour during the shift-day interval, for all the testing data:

The graphs and tables present strong evidence of the model's inability to learn that a new day has started, despite the exogenous variable PCA indicating the informational value of the time-of-day parameter. A similar problem is faced by other competition participants, especially Moerira[31] who claims *For future*

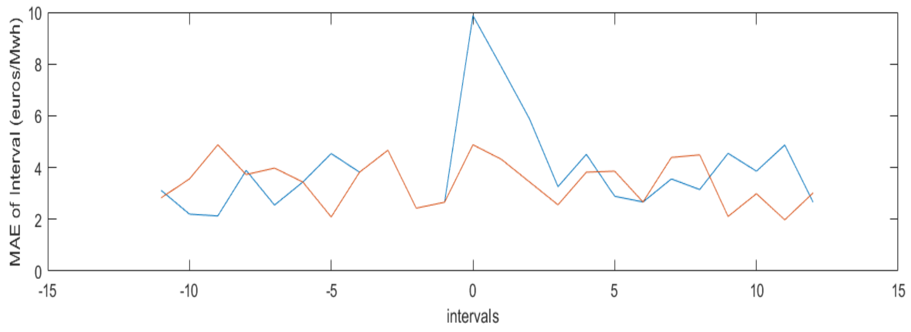


Figure 16: XGBoost(ARX) learns regime-shifts better than LSTM(Statistical)models

work, since the occurrence of extreme prices appears to have a strong impact on the quality of the prediction, the employment of a spike detection/forecast method may lead to improved results. This is corroborated by another competition participant Frietas[13] who states *...the growing trend of error over the time horizon can be partially explained by the forecast iterative strategy. The sharp increase between 18th and 21 st hours could be associated with the existence of many outliers in those hours.*

It is noticed that XGBoost performs significantly more consistently than LSTM based models, and is able to better identify regime switches. Looking at the error distribution graphs also suggest that XGBoost has thinner tails and narrower peaks. This lends credence to the idea of using ensemble models to improve accuracy.

7.2 Solution to Regime-Switch Problem

Having identified this problem, the frequency of Shift-Day input vectors is artificially increased by a factor of 1.2. This is to help both the XGBoost and LSTM models better anticipate and cope with sudden changes. As the models are already effective at trending time-periods, boosting Shift-Day input vector frequency should not have an adverse impact on their performance.

Model	**ARXPchng**	**ARXPchng-PCA**	**AR-Statistical**
XGBoost	3.53	3.12	4.09
LSTM	4.88	3.57	3.21

Table 12: Boosting frequency of Shift-Day input vector improves prediction performance

The graphs and tables present strong evidence of the model's inability to learn that a new day has started, despite the exogenous variable PCA indicating the informational value of the time-of-day parameter. It is noticed that XGBoost performs significantly more consistently than LSTM based models, and is able to better identify regime switches. Looking at the error distribution graphs also suggest that XGBoost has thinner tails and narrower peaks. This lends credence to the idea of using ensemble models to improve accuracy.

7.3 Brute Force Feature Batching

Ensemble predictions involve more than mere parameter combination. For example, combining statistical and exogenous features together and training either XGBoost or LSTM models on them, causes significant performance degradation:

Model	ARXPchng	ARXPchng-PCA	AR-Statistical	ARX+Statistical
XGBoost	3.53	3.12	4.09	5.32
LSTM	4.88	3.57	3.21	6.78

Table 13: Combining Statistical and Exogenous Parameters Degrades Prediction Performance

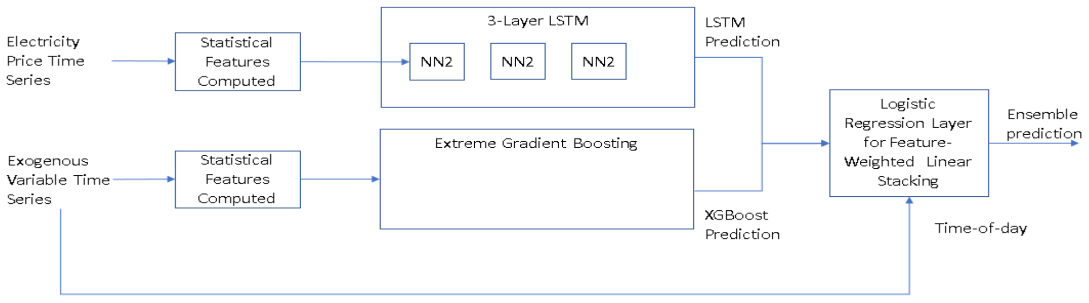
7.4 Feature Weighted Linear Stacking Ensemble Models

The Netflix Prize was an open competition for the best collaborative filtering algorithm to predict user ratings for films, based on previous ratings without any other information about the users or films, i.e. without the users or the films being identified except by numbers assigned for the contest[3]. The prize-winners recognized that some models work better for users who give very few recommendations, and other models work better for seasoned reviewers with a vast review history. Essentially, a *meta-feature* of the input vector can be used to decide which is the best model to run.

7.5 Performance of Final Ensemble Model

Adapting the above idea to the EPF problem domain, *time-of-day* becomes a distinguishing *meta-feature*, because XGBoost trained on ARXPchng-PCA is able to predict more accurately during regime-switching, and LSTMs trained on AR-Statistical achieve good performance during trending time-periods. This ensemble technique is called *Feature-Weighted-Linear Stacking* and uses a linear regression layer comprising of the outputs through an XGBoost Layer, an LSTM layer, and *time-of-day*.

The final ensemble neural network design is shown below:



Electricity Price Time Series
Statistical Features Computed
3-Layer LSTM
NN2
NN2
NN2
LSTM Prediction
Exogenous Variable Time Series
Statistical Features Computed
Extreme Gradient Boosting
XGBoost Prediction
Logistic Regression Layer for Feature-Weighted Linear Stacking
Ensemble prediction
Time-of-day

Figure 17: Training of Non-Recurrent NARX Model

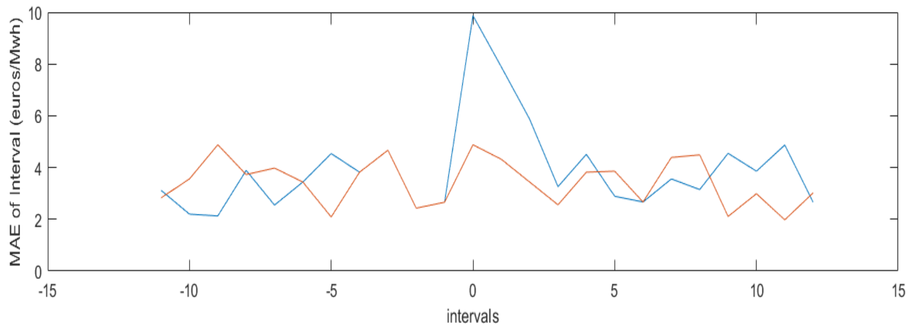


Figure 18: Ensemble outperforms both models during regime-changes with a lower σ_{MAE}

The resultant MAE is 2.85

Model	**MAE**
XGBoost-ARXPchng-PCA	3.35
LSTM-ARStatistical	3.77
Ensemble:	2.85

Table 14: Ensemble Model outperforms both XGBoost and LSTM models using Feature-Weighted-Linear-Stacking

Looking at the shift-MAE plot, we see that the ensemble model can better understand when the regime-switches, and provides a more stable performance:

8 Lightening - Open Source Testing Framework

Weron(2014)[45]argues that the variance in forecasting performance between a set of models belonging to the same family should be far less than what the EPF research community has published so far. He highlights the need for a solution that contains:

1. Standardized and indexable collection of EPF data-sets from different elec-

tricity markets

2. Robust error evaluation procedures

3. Statistical testing of out-performance claims

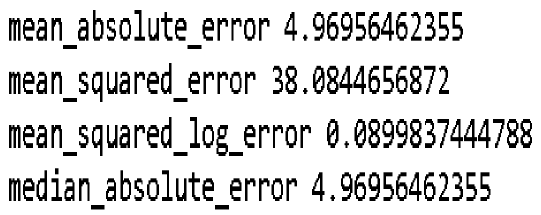
Lightening is a command-line utility that automates model testing and data-engineering by providing two JSON files that contain testing and training configuration. Presently, the data-engineering configuration file contains the following options:

```
{"params": {
    "ts_location": "X.csv",
    "column_header": [],
    "enforce_column_header":"False",
    "date_columns": [0],
    "date_format": "%Y%m%d",
    "missing_values_protocol":{
        "method":"ffill",
        "limit":"1"
    },
    "test_train_split":{
        "date_based":{
            "activate":"False",
            "train_start_date":"27/05/1996",
            "train_end_date":"27/05/1998",

            "test_start_date":"27/05/1996",
            "test_end_date":"27/05/1996"
        },
        "size_based":{
            "activate":"True",
            "train_float": 0.7
        },
        "shuffle":"False"
    },
    "input_sequence_length":0,
    "filtering":{
    },
    "drop_columns":[],
    "Y_info":{
        "out_of_csv":"False",
        "location":"X_previous_price.csv",
        "col_num":null,
        "col_name":"Price"
    },
    "autoregress":"True",
    "lag":1,
    "sequence_length":24,
    "is2DModel" : "False"
}}
```

Researchers can specify the location of the time-series file, and configure how they want the dataset to be parsed. Testing and training data can be divided by either specifying the date-times, or through the conventional 80/20 size or length based split. Researchers can next decide how to parse the target values, and once the testing and training data is loaded, can also configure they want to create input vector sequences or use auto-regression. Tedious operations such as data-reshaping, converting from data-frames to numpy arrays etc. are handled by Python scripts in the back-end. The final output from the data-engineering back-end are the testing and training data-sets saved as numpy arrays which can be loaded in later scripts.

Once the training and testing data is shaped properly, the researchers can work independently on their models, and train the models using the saved training and testing data-sets. Once the model has been built, it is saved as an *.h5* file on the disk. Next, the researcher will configure the testing JSON file:

```
{"test": {
    "test_model": {
      "model_loc":"my_model.h5"
    },

    "test_fn":{
      "file_loc":"abc.py",
      "fn_name":"abc()"
    },
    "isRegression":"True",
```



```
mean_absolute_error 4.96956462355
mean_squared_error 38.0844656872
mean_squared_log_error 0.0899837444788
median_absolute_error 4.96956462355
```

Figure 19: Lightening will automatically store accuracy metrics into a dictionary after each run

```
    "regression_metrics":["mean_absolute_error",
                "mean_squared_error",
                "mean_squared_log_error",
                "median_absolute_error"],

    "isClassification":"False",
    "classification_metrics":["accuracy_score",
                  "auc",
                  "average_precision_score",
                  "classification_report",
                  "confusion_matrix",
                  "f1_score",
                  "log_loss"],

    "use_close_loop_test":"True",
    "isSeparate":"False",
    "X_test_loc":"abc.csv",
    "X_test_filetype":"csv"
}}
```

The researcher can specify the problem type (classification or regression, the appropriate regression testing metrics, whether it would like forecast an interval using a closed loop, or whether the model outputs a sequence by itself. If the researcher wants to test the model on out-of-sample data, or another data-set (like in transfer learning), it can specify the location of the separate test files. The Python back-end will output information about the model. Lightening will automatically generate and save error distribution diagrams. The MAE distribution of one of the AR-Statistical models is seen below:

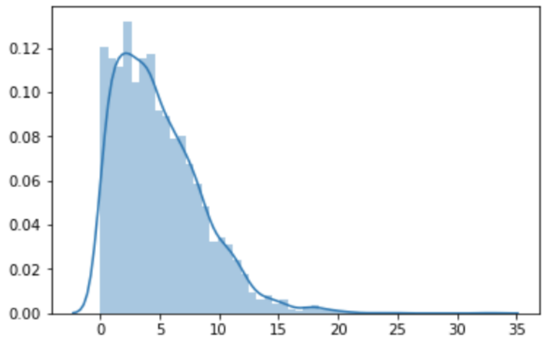


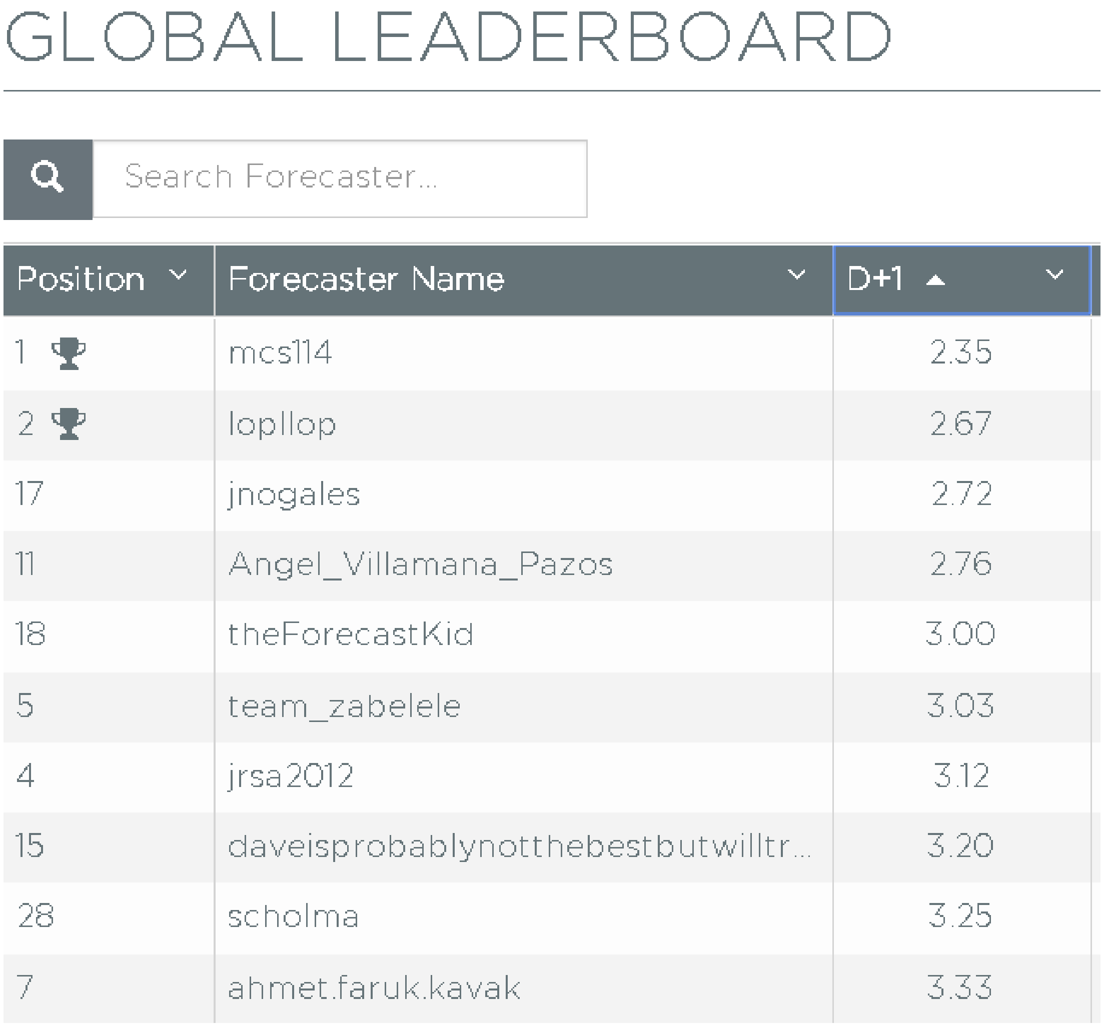
Figure 20: MAE Distribution of AR-Statistical LSTM

9 Conclusion

The research efforts in this report have successfully met the goal of building a neural-networks based Electricity-Price-Forecasting model that matches state-of-the-art methods presented in the original challenge. With an MAE of 2.85 on out-of-sample competition testing data, the model ranks 5th out of 45 participants, and narrowly misses out to competitor *jnogales* who had scored 2.72. Nogales is an active researcher in the EPF community, and makes use of complex Markov Regime Switching models that effectively forecast price jumps.

The ensemble model presented is able to achieve this by a novel data-engineering technique and another novel ensemble model prediction technique. *Shift-Day MAE* is a metric that is not used in any EPF forecasting papers, and proves to be extremely useful in understanding what the model is learning. In this case, the model was not learning regime-shifts despite being provided with time-of-day information. Without *Shift-Day MAE*, it would not have been possible to understand how exactly the model could be further improved, and take the appropriate steps to boost the frequency of *Shift-Day* input vectors.

The second novelty lies in the use of Feature Weighted Linear Stacking method to build an ensemble model. What is noteworthy is that the idea was derived from the Netflix Prediction Challenge, which is a completely different domain, and then adapted to the price forecasting problem. Traditionally, either complex non-linear aggregators are used to decide how to combine multiple predictions, or a one-size-fits-all linear-regressor is used. Using a linear-regression layer which is also dependent on an exogenous feature allowed for the creation of a very simple but highly accurate aggregator. This is reflective of an insightful understanding of the nature of the EPF data, as opposed to simply applying models in a blanket fashion without regard to how to best use the existing features.


GLOBAL LEADERBOARD

Search Forecaster...

Position	Forecaster Name	D+1
1	mcs114	2.35
2	lopllop	2.67
17	jnogales	2.72
11	Angel_Villamana_Pazos	2.76
18	theForecastKid	3.00
5	team_zabelele	3.03
4	jrsa2012	3.12
15	daveisprobablynotthebestbutwilltr...	3.20
28	scholma	3.25
7	ahmet.faruk.kavak	3.33

Figure 21: The ensemble model's MAE score of 2.85 allows it to enter the top 5 (D+1) ranks

Lastly, the building of an in-the-works open source testing platform is an exciting chance to collaborate with the broader EPF community and help evaluate models on a standardized platform. Development of the platform will continue after the research project.

References

[1] S. Anbazhagan and N. Kumarappan, “Day-ahead deregulated electricity market price forecasting using recurrent neural network,” *IEEE Systems Journal*, vol. 7, no. 4, pp. 866–872, 2013.

[2] A. Azadeh, M. Moghaddam, M. Mahdi, and S. Seyedmahmoudi, “Optimum long-term electricity price forecasting in noisy and complex environments,” *Energy Sources, Part B: Economics, Planning, and Policy*, vol. 8, no. 3, pp. 235–244, 2013.

[3] J. Bennett, S. Lanning, and Netflix, “The netflix prize,” in *In KDD Cup and Workshop in conjunction with KDD*, 2007.

[4] M. Burger, B. Graeber, and G. Schindlmayr, *Managing energy risk: an integrated view on power and other energy markets.* Wiley, 2007.

[5] R. Carmona and M. Coulon, “A survey of commodity markets and structural models for electricity prices,” in *Quantitative energy finance: modeling, pricing, and hedging in energy and commodity markets.* Springer, 2014, pp. 167–202.

[6] N. Chaabane, “A novel auto-regressive fractionally integrated moving average-least-squares support vector machine model for electricity spot prices prediction,” *Journal of Applied Statistics*, vol. 41, no. 3, pp. 635–651, 2014.

[7] T. Chen and C. Guestrin, “Xgboost: A scalable tree boosting system,” in *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD ’16. New York, NY, USA: ACM, 2016, pp. 785–794. [Online]. Available: http://doi.acm.org/10.1145/2939672.2939785

[8] A. Conejo, J. E. Conteras, and P. M.A, “Forecasting electricity prices for a day-ahead pool-based electric energy market,” *International Journal of Forecasting*, vol. 21, no. 3, pp. 435–462, 2005.

[9] J. Contreras, R. Espinola, F. Nogales, and A. Conejo, “Arima models to predict next-day electricity prices,” *IEEE Transactions on Power Systems*, vol. 18, no. 3, pp. 1014–1020, 2003.

[10] M. Coulon and S. Howison, “Stochastic behavior of the electricity bid stack:f rom fundamental drivers to power prices,” *Journal of Energy Markets*, vol. 2, no. 1, pp. 1756–3615, 2009.

[11] F. X. Diebold, T. A. Gunther, and A. S. Tay, "Evaluating density forecasts with applications to financial risk management," *International Economic Review*, vol. 39, no. 4, pp. 863–883, 1998.

[12] J. Durbin and S. Koopman, *Time seroes analysis by state space methods.* Oxford University Press, 2001.

[13] D. Freitas, A. Martins, and J. Lagarto, "Modeling of cyclic events in electricity markets using circular statistical methods," in *2016 13th International Conference on the European Energy Market (EEM)*, June 2016, pp. 1–5.

[14] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Annals of Statistics*, vol. 29, pp. 1189–1232, 2000.

[15] E. Guerci, M. Rastegar, and S. Cincotti, "Agent-based modeling and simulation of competitive wholesale electricity markets," in *Handbook of power systems II-energy systems.* Springer, 2010, pp. 241–286.

[16] E. B. Hreinsson, "Electric load forecasting in a hydro- and renewable based power system," in *2016 13th International Conference on the European Energy Market (EEM)*, June 2016, pp. 1–6.

[17] C.-M. Huang, C.-J. Huang, and M.-L. Wang, "A particle swarm optimization to identifying the armax model for short-term load forecasting," *IEEE Transactions on Power Systems*, vol. 20, no. 2, pp. 1126–1133, 2005.

[18] R. Hyndman and G. Athanasopoulos, *Forecasting: principles and practice.* OTexts, 2013.

[19] A. Jain, J. Mao, and K. Mohiuddin, "Artificial neural networks: a tutorial," *Computer*, vol. 29, no. 3, pp. 31–44, 1996.

[20] T. Johnsen, "Demand, generation and price in the norweigan market for electric power," *Energy Economics*, vol. 23, no. 3, pp. 227–251, 2001.

[21] T. Kanamura and K. Ohashi, "A structural model for electricity prices with spikes: measurement of spike risk and optimal policies for hydropower plant operation," *Energy Economics*, vol. 29, no. 5, pp. 1010–1032, 2007.

[22] N. Karakatsani and D. Bunn, "Forecasting electricity prices: the impact of fundamentals and time-varying coefficients," *International Journal of Forecasting*, vol. 24, no. 4, pp. 764–785, 2008.

[23] F. Keynia and N. Amjady, "Electricity price forecasting with a new feature selection algorithm," *Journal of Energy Markets*, vol. 1, no. 4, pp. 47–63, 2008.

[24] C.-J. Kim, I.-K. Yu, and Y. H. Song, "Prediction of system marginal price of electricty using wavelet transform analysis," *Energy Conversation and Management*, vol. 43, no. 4, pp. 1839–1851, 2002.

[25] A. Konar, *Computational intelligence: principles, techniques, and applications.* Springer, 2005.

[26] S. Koopman, M. Ooms, and M. Carnero, "Periodic seasonal reg-afirma-garch models for daily electricity spot prices," *Journal of the American Statisical Association*, vol. 102, no. 477, pp. 16–27, 2007.

[27] P. Kosater and K. Mosler, "Can markov-regime switching models improve power price forecasts? evidence for german daily power prices," University of Cologne, Institute of Econometrics and Statistics, Discussion Papers in Econometrics and Statistics 1/05, 2005. [Online]. Available: https://EconPapers.repec.org/RePEc:zbw:ucdpse:105

[28] T. Lin, B. Horne, P. Tino, and C. Giles, "Learning long-term dependencies in narx recurrent neural networks," *IEEE Transactions on Neural Networks*, vol. 7, no. 6, pp. 1329–1337, 1996.

[29] J. Mcdonald, "A real-time implementation of short-term load forecasting for distribution power systems," *IEEE Transactions on Power Systems*, vol. 9, no. 2, pp. 988–994, 1994.

[30] R. Moreira, R. Bessa, and J. Gama, "Probabilistic forecasting of day-ahead electricity prices for the iberian electricity market," in *2016 13th International Conference on the European Energy Market (EEM)*, June 2016, pp. 1–5.

[31] ——, "Probabilistic forecasting of day-ahead electricity prices for the iberian electricity market," in *2016 13th International Conference on the European Energy Market (EEM)*, June 2016, pp. 1–5.

[32] F. Nogales, J. Contreras, A. Conejo, and R. Espinola, "Forecasting next-day electricity prices by time series models," *IEEE Transactions on Power Systems*, vol. 17, no. 2, pp. 342–348, 2002.

[33] C. Olah, "Understanding lstm networks." [Online]. Available: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[34] J. S. P. Goncalves, "Evaluation of the impact of the feed-in generation in the prices of the iberian electricity market in 2013," in *2016 13th International Conference on the European Energy Market (EEM)*, June 2016.

[35] A. Panagiotelis and M. Smith, "Bayesian density forecasting of intraday electricity prices using multivariate skew t distributions," *International Journal of Forecasting*, vol. 24, no. 4, pp. 710–727, 2008.

[36] L. Ruthkowski, *Computational intelligence: methods and techniques.* Springer, 2008.

[37] A. Schmutz and P. Elkuch, "Electricity price forecasting: application and experience in the european power markets," in *Proceedings of the 6th IAEE European Conference, Zurich.* IAEE, 2004, pp. 100–113.

[38] M. Shafie-khah, M. P. Moghaddam, and M. Sheikh-El-Eslami, "Price forecasting of day-ahead electricity markets using a hybrid forecast method," *Energy Conversion and Management*, vol. 52, no. 5, pp. 2165–2169, 2011.

[39] V. Sharma and D. Srinivasan, "A hybrid intelligent model based on recurrent neural networks and excitable dynamics for price prediction in deregulated electricity markets," *Engineering Applications of Articial Intelligence*, vol. 26, no. 5-6, pp. 1562–1574, 2013.

[40] J. Sun and T. Leigh, "Dynamic testing of wholesale power market designs: An open-source agent based framework," *Computational Economics*, vol. 30, no. 3, pp. 291–327, 2007.

[41] C. Tan, "Financial time series forecasting using improved wavelet neural network."

[42] I. Vahvilainen and T. Pyykkonen, "Stochastic factor model for electricity spot price–the case of the nordic market," *Energy Economics*, vol. 27, no. 2, pp. 351–367, 2005.

[43] M. Ventosa, A. Baillo, and M. Rivier, "Oligopolistic competition in power networks: a conjectured supply function approach," *IEEE Transactions on Power Systems*, vol. 17, no. 3, pp. 597–607, 2002.

[44] ——, “Electricity market modeling trends,” *Energy Policy*, vol. 33, no. 7, pp. 897–913, 2005.

[45] R. Weron, *Modeling and forecasting electricity loads and prices: a statistical approach.* Wiley, 2006.

[46] R. Weron and A. Misiorek, “Forecasting spot electricity prices with time series models,” in *IEEE Conference Proceedings - EEM05.* IEEE, 2005, pp. 133–141.

[47] ——, “Short-term electricity price forecasting with time series models: a review and evaluation,” *Complex Electricity Markets*, vol. 16, no. 3, pp. 231–254, 2006.