FORECASTING DAY AHEAD WHOLESALE ELECTRICITY PRICES FOR THE IRISH MARKET USING MACHINE LEARNING ALGORITHMS

Higher Diploma in Data Science and Analytics 2017

Kevin O'Mahony

Abstract

The electricity supply industry is undergoing a process of deregulation world-wide which has turned electricity into a tradable commodity. This has spawned extensive research into electricity price prediction as short term forecasting has been a key concern for all market participants.

This project investigates the use of machine learning techniques in order to perform one day ahead predictions of Irish wholesale electricity prices. Support Vector Machines and Neural Networks are the two main models examined. We also look at how Random Forests, Gradient Boosting and ExtraTree Regressor perform by comparison. These models are selected because they have produced good prediction accuracies in other fields and can also handle the non-linear, non-stationary nature of electricity price time series data.

Acknowledgements

I would like to acknowledge my supervisor, Dr. Laura Climent for her assistance throughout the project, Adrian O'Connor for providing second reader feedback and Jane O'Flynn and Conor Lynch of Nimbus for their help.

This document represents the authors views and not those of CIT or Nimbus. It is presented in partial fulfilment of the Higher Diploma in Data Science and Analytics, May 2017.

Contents

Tables

Figures

1 Introduction

Over the last 20 years there has been a world-wide move to deregulate the electricity supply industry. As a result, electricity has become similar to other tradeable commodities and the forecasting of electricity prices has become an area of significant research. Electricity is fundamentally different from most other commodities in that it can't be stored, its production typically depends on costly plant start-up/shutdown procedures and its price movement within a day can be very volatile with extreme price spikes (order of magnitude increases/decreases). Short term forecasting ability is key to all market participants.

Electricity price forecasting has long been considered a difficult prediction task because of the volatile nature of its typical time series. Some of the most promising recent results in predicting electricity prices have come from using models in the machine learning domain. For this study, we select two of the most popular models: Neural Networks (NN) and Support Vector Machine. These models are selected on the basis that they can effectively handle the non-linear nature of our data set, cope with our non-stationary time series data and offer some of the best prediction accuracies seen in the field. There are also a wide variety of libraries available that implement these models in R and Python.

The client (Nimbus) is developing novel forecasting models loosely based on existing time series techniques. Therefore, there is little benefit in looking at traditional time series models (ARMA, ARIMA, Exponential smoothing, etc.) when the client has already improved on the typical forecasting accuracy levels possible with these methods.

The Single Electricity Market (SEM) is the wholesale electricity market operating in Ireland. All electricity generators above 10MW must sell their output through SEM. Suppliers buy electricity from SEM and sell it on to their end users, the retail subscribers. The Single Electricity Market Operator (SEMO) oversees the operation and administration of the market. SEMO is a joint venture between EirGrid plc in the Republic and SONI Limited in Northern Ireland.

Wholesale electricity price data and estimates are available from the SEMO web site. We have three years (2014 – 2016) worth of price and load time series data. The price data consists of a combination of SEMO estimated and final settlement prices. We also have the corresponding three years' worth of metrological data (wind speed and temperature) collected at four different sites in Ireland. Our goal is to build a model using a combination of these data series to allows us generate day-ahead predictions of wholesale electricity prices.

2 Related Research

2.1 Neural Networks

NNs are biologically inspired models consisting of neuron type processing nodes arranged in layers. The input for each node is connected to the outputs of nodes in the previous layer. Each node executes a function, known as its activation function on the sum of its inputs. The activation function dictates what if any output is generated by the node and fed through to its connected nodes. The links between nodes act much like the axons connecting neurons

in the human brain. The links are assigned varying weights to encode knowledge during a training phase conceptually mirroring an infant's early acquisition of knowledge. A minimal network will consist of one or more input nodes, an output layer of one or more nodes and zero or more internal layers referred to as hidden layers. NN operation is divided into two phases: the training and prediction phases. For the training phase, the network is presented with input to drive changes in the weights in order to match the required known output in a supervised learning mode. When the network weights are fixed from the training phase by reaching a predefined number of iterations or an acceptable error criteria we can present unseen input to the network and expect to get our predicted output.

The origins of Neural Networks dates back to the work of McCulloch and Pitts [1] in the 1940's when they first proposed the concept of an artificial neuron. It was very simple: the neuron's inputs could be either zero or one and the output could be a zero or one. All inputs where summed. Each input could be either add one into the sum or subtract one from the sum. The final sum would then need to be greater than some defined threshold before the output was one otherwise the output was zero. With this simple arrangement, most of the basic Boolean logic gates could be constructed along with many more operations like delay gates etc. using one or more neurons.

It was twenty years later before Rosenblatt's invented the perceptron [2]. This was originally designed as a hardware project for image recognition consisting of a photocell array connected to neurons with mechanically variable weights controlled by motors. It was able to be trained to identify objects held up to the array of photocells however it fell short of being a global classifier model. Rosenblatt's single layer perceptron model whilst a major step forward was only capable of learning linear separable patterns. Minsky and Papert's work [3] showed up the limitations of the perceptron, where they presented a mathematical proof that it was not capable of making global generalisations. They also claimed a multilayer NN would have the same limitations. Their work generated doubts about the possible capabilities of NNs and led to a period of reduced interest and research activity in NNs that lasted for almost 20 years.

The 1980's brought renewed interest when the back-propagation algorithm for training an NN was discovered. Several researchers independently discovered the back-propagation algorithm however Rumalhart et al. [4] remains the most influential paper. This proved to be a very efficient ANN training method and it meant that NNs could now be used for solving a wider range of prediction problems than before.

A general evaluation in 1998 of the state of research into NN [5] drew two conclusions, "First, NNs, when they are effectively implemented and validated, show potential for forecasting and prediction. Second, a significant portion of the NN research in forecasting and prediction lacks validity". Several of the studies they reviewed suffered from validation or implementation problems which cast doubts over their results. Of the 'clean' NN studies they reviewed 86% of them outperformed alternative models. (Regression, kNN, Discriminant Analysis, ARIMA). The majority of the NNs reviewed used back propagation.

By the early 2000's we can see NNs starting to gain traction in electricity price prediction research. The Californian market for Jan – Sept. 1999 was studied [6] using a short term forecast model with a four week training period. A novel committee machine of neural networks [7] was used to forecast pricing in the 2002 New England market to compensate for the possible errors in the input-output mapping of a single neural network. Rather than use the average of the outputs of the networks, weights were derived from the input and

historical data for combining each output. A one to six hour ahead forecasting study of the Victoria market [8] for the period 2000 – 2003 achieved average MAPE[1] figures of 9.75% for the one hour and 20.03% for the six hour ahead predictions. Multiple-input, multiple-output models where considered by Gareta et al [9] when predicting day ahead hourly prices. Conejo et al. [10] performed day ahead prediction study comparing time series (ARIMA, dynamic regression and transfer function), neural network and wavelet techniques with 2002 Pennsylvania–New Jersey–Maryland (PJM) data. Their neural network was outperformed by the other methods for all sampled season. In contrast the same market [11] was studied using a modified Levenberg-Marquardt algorithm (LM is a technique to minimize non-linear functions) for the same year. It performed better than other models: ARIMA, Dynamic Regression, Transfer Functions, combined ARIMA/Wavelet model and achieved forecasting MAPEs ranging 4 – 13% depending on day of week and season used. LM was also used in a study to predict week ahead prices in the Spanish and Californian markets [12] using a three layered feed forward NN. MAPEs of 9% for the Spanish and 3% for the Californian tests were achieved. An interesting cascaded neural network configuration was used [13] to first predict day ahead load and then using the load prediction along with other external variables as input to a second NN to predict day ahead price values. An average MAPE of 6% was achieved using California market data.

A 2009 evaluation of the main methodologies used in electricity price prediction [14] showed that for the 25 NNs studies reviewed, the most popular choice among researchers at the time was the feed forward architecture along with back propagation as the learning algorithm. Neupane et al. [15] demonstrated how focusing on input feature selection with a simple NN configuration produced excellent results. They used three different markets: New York, Australia and Spain and achieved average MAPEs: 4.18%, 8.17% and 5.76%. Radial basis functions (RBF) have found use as activation functions in neural networks hidden layers. A 2011 study [16] on the New South Wales (Australia) market used a RBF neural network with day-of-week used as an extra input. The RBF NN performed better than comparable results generated from classical moving average, Holt-Winters and a feed forward neural network.

Hybrid model research has combined ANN models with other techniques such as Fuzzy Logic, Wavelet Transforms, Particle Swarm [17] and Genetic Algorithms [18]. Catalao et al. [19] used a wavelet transform (WT) with a hybrid of ANN and fuzzy logic. The WT was first used to decompose the price series into a set of simpler component series. The predicted values of these constitutive series were generated using the ANN/fuzzy logic combination and used as input to the inverse WT to create the final forecasted prices. Average MAPE of 6.5% was achieved using data from the Spanish market.

It can be seen from the research literature that the best price prediction errors achieved using ANNs are in the region of 5% for short term forecasting. There doesn't appear to be any particular combination of neural network configuration, activation function or hybrid pairing that consistently gives the best forecasting results.

[1] MAPE is the Mean Absolute Percent Error and is the most commonly quoted measure of forecasting error in the literature:

$$MAPE = \frac{1}{n}\left(\sum \frac{|Actual - Forecast|}{|Actual|}\right) X\ 100$$

2.2 Support Vector Machines

Support Vector Machines (SVM) [20] are used for both non-linear classification and regression problems. This is achieved by transforming the input data into a higher dimensional feature space resulting in a linear problem that can be more easily solved. The mapping is achieved via a kernel function. Popular non-linear kernel functions include sigmoid, radial basis and polynomial.

SVMs make use of the structural risk minimization concept. This attempts to balance a models complexity with its success in fitting training data in order to avoid overfitting. This is in contrast to empirical risk minimization which essentially boils down to using the model with the best fit of the training data but which may perform very poorly on test data. Neural networks employ empirical risk minimization.

The basic concepts in SVM can be understood by considering the binary classification problem. In the case of two linearly separable classes as shown in Fig. 1 there are multiple dividing lines that could be used to separate the classes.

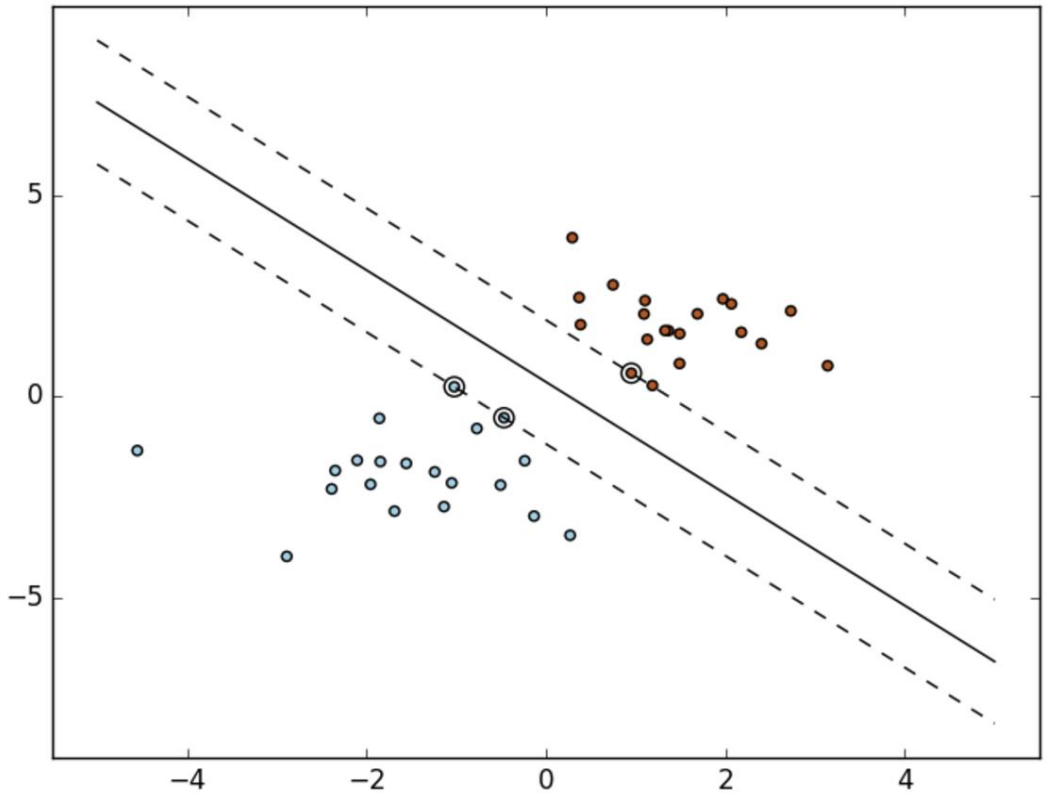
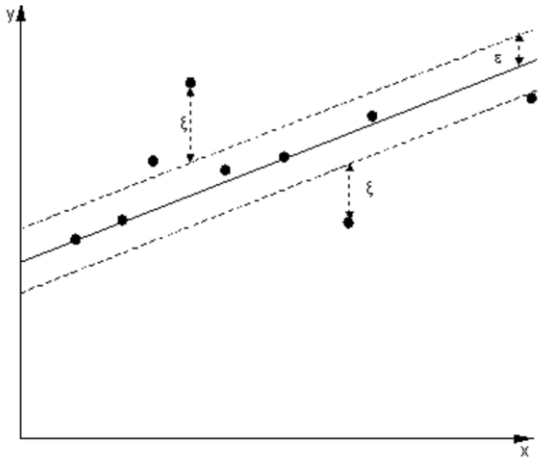


Figure 1. *Linearly separable classes with MMH and Support Vectors*

SVM picks the dividing line by searching for the maximum margin hyperplane (MMH) i.e. the plane which separates the two classes while minimizing error and maximising the separation between the classes. This increases the chances that future unseen data will also be separable by the same plane thereby reducing the likelihood of overfitting. The selection of the hyperplane is solved by means of quadratic programming. The support vectors are the data points closest to the MMH, the circled data points above. Each class must have a

minimum of one support vector. This approach can be generalised to N-dimensions including non-linear surfaces generated by the kernel functions.

For regression problems, SVM use is referred to as Support Vector Regression (SVR). Here a loss function that ignores errors up to a fixed maximum deviation between predicted and actual values is introduced. This is known as the epsilon, ε-insensitive loss function. SVR attempts to fit as many data points as it can within this deviation. Points lying outside this zone are penalized relative to their distance, ξ from the zone. Points within the deviation are not penalized. Figure 2 below shows an example of a linear regression function with the ε-insensitive zone.



.

Figure 2*. Linear regression function*

The problem then evaluates to finding a function f(x) with at most ε-deviations from all the training data and is as 'flat' as possible, i.e. as simple as possible. For a linear function above:

$$f(x) = \langle w, x \rangle + b \qquad \textbf{(1)}$$

- where $\langle w, x \rangle$ represents the dot product, flatness means selecting small *w.*

One way to do this is to minimize the Euclidean norm i.e. $\|w\|^2$. This can then be expressed as a convex optimization problem, requiring:

$$\textit{minimizing} \quad \tfrac{1}{2}\|w\|^2 \qquad \textbf{(2)}$$

$$\textit{subject to} \quad \begin{cases} y - \langle w, x \rangle - b \leq \varepsilon \\ \langle w, x \rangle + b - y \leq \varepsilon \end{cases} \qquad \textbf{(3)}$$

For real world scenarios this may not be possible, e.g. there may be noise in the training data so the idea of slack variables ξ as shown above, are introduced which allows for errors to exist. This changes the problem to minimizing the term from (2) and including the sum of all slack error variables and accounting for ξ in the constraints:

$$\textit{minimizing} \quad \tfrac{1}{2}\|w\|^2 + C \sum \xi \qquad \textbf{(4)}$$

$$\text{subject to} \quad \begin{cases} y - \langle w, x \rangle - b \leq \varepsilon + \xi \\ \langle w, x \rangle + b - y \leq \varepsilon + \xi \\ \xi \geq 0 \end{cases} \qquad \textbf{(5)}$$

Here the constant C > 0 determines the trade-off between the flatness of the function and the degree to which deviations greater than ε are allowed.

For a non-linear data set SVR first maps the data to high-dimensional feature space as in the classification problem, using a kernel transformation and then proceeds as before. The mathematical detail is beyond the scope of this study.

There are several recent examples of studies of electricity price prediction using SVR. [21] used daily prices from New England 2003 – 2010 to predict average daily prices and found SVR to outperform a back propagation 3 layer NN, achieving an average MAPE of 8%. Australia, New South Wales regional price data from 2002 was examined [22]. Forecasts where for 7 days into the future and yielded an MAE[2] of 28. This relatively poor figure was not improved with the inclusion of intraday estimated demand as input features. SVR models [23] have been used to combine diverse factors like oil and natural gas prices to predict German electricity price from 2010 to 2012. It generated a best average MSE[3] of 38 for 2012.

Hybrid solutions have been used to optimize SVR model parameters. A Particle Swarm Optimization process [24] was used to customise the cost penalty and the maximum deviation parameters for a SVR model applied to the Italian, New York and New England markets for 2004 – 2005. MAPE ranged from 4.6% to 9.2% for New York and New England while the recently created Italian market showed much greater volatility and MAPE values ranged from 16% to 4%. This study also included useful empirical formulae for the cost penalty and maximum deviation based on characteristics of the training data.

The remainder of this study is organised as follows. Section 3 details the data set and exploratory analysis performed. Section 4 discussed the experimental setup used to forecast our day ahead prices and the results achieved. Section 5 presents the conclusions of the study followed by suggestions for future work.

3 Methodology

Next, we describe the data set and perform a preliminary exploratory analysis.

3.1 Data Set

The electricity price data is available from the SEMO web site [25]. Metrological data was downloaded from the Met Eireann web site. The data set consists of three years (2014 – 2016) of price, load and metrological data as detailed below.

[2] MAE is the Mean Absolute Error defined as $\frac{1}{n}\sum_{i=1}^{n}|Forecast_i - Actual_i|$

[3] MSE is the Mean Squared Error defined as $\frac{1}{n}\sum_{i=1}^{n}(Forecast_i - Actual_i)^2$

- **Date** = Date of price and load data.
- **Slot =** Half hour slot of day, numbered from 1 – 48.
- **EA price** = First Ex Ante estimate price. Single measurement per half hour. Generated by SEMO.
- **EA2 price** = Second Ex Ante estimate price. Single measurement per half hour. Generated by SEMO.
- **WD1 price** = Within day estimate price. Single measurement per half hour. Generated by SEMO.
- **EP1 price** = Ex. Post indicative settlement price. Issued 1 day after trading day (TD+1). Single measurement per half hour. Generated by SEMO.
- **EP2 price** = Ex. Post final settlement price. Issued 4 days after trading day (TD+4). Single measurement per half hour. Provided by SEMO.
- **Load** = Electricity demand in MW. Single measurement per hour. Provided by SEMO.
- **Wind (Athenry)** = Wind speed (in m/s) measured in Athenry. Single measurement per hour. Provided by Met Eireann
- **Wind (Valentia)** = Wind speed (in m/s) measured in Valentia. Single measurement per hour. Provided by Met Eireann
- **Temperature (Cork Airport)** = Temperature (in degrees Celsius) measured at Cork Airport. Single measurement per hours. Provided by Met Eireann.
- **Temperature (Dublin Airport)** = Temperature (in degrees Celsius) measured at Dublin Airport. Single measurement per hours. Provided by Met Eireann.

The Ex. Ante price estimates are generated by a SEMO prediction model that incorporates metrological data. We will need to standardise time slots across the price/load/metrological data to half hourly periods. The only option here is to duplicate the load and metrological hour figure for the two corresponding half hour periods of the price data.

3.2 Exploratory data analysis

3.2.1 Data summary

We have 52,544 rows in our data set covering the time period 01/01/2014 to 31/12/2016. Each day is broken up into 48 slots. Each slot has an associated EA, EA2, WD1, EP1 and EP2 price data. Summary statistics for the complete three-year period are shown in Table 1 below.

Table 1. *Price data summary statistics (2014 – 2016)*

	EA	**EA2**	**WD1**	**EP1**	**EP2**
Mean	46.83	49.06	49.09	49.59	49.72
StdDev	24.29	24.92	27.72	29.27	30.22
Min	-100.00	-100.00	-100.00	-53.53	0.01
Max	617.92	1000.00	541.87	956.83	955.38

The estimated features: EA, EA2, WD1 and EP1 can be seen to vary widely on how close to the actual, EP2 price's range. Mean and standard deviation are all relatively close to each other which suggests the range difference are due to a small number of extreme estimated values.

Table 2. *Metrological data summary statistics (2014 – 2016)*

	Cork_Temp *(celsius)*	**Dublin_Temp** *(celsius)*	**Wind_Athenry** *(meters/second)*	**Wind_Valentia** *(meters/second)*
Mean	9.93	9.80	3.83	5.02
StdDev	4.33	4.89	2.18	2.98
Min	-2.30	-7.70	0.00	0.51
Max	23.80	26.00	14.92	24.69

The next three plots illustrate how volatile actual electricity prices are over single days, weeks and months.

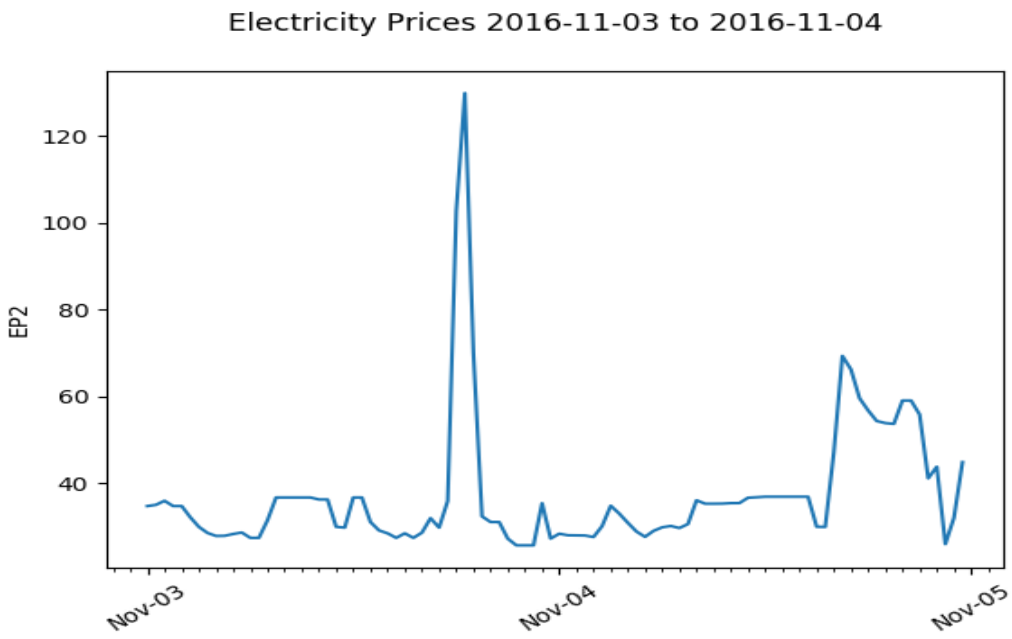


Figure 3. *Daily Electricity Prices*

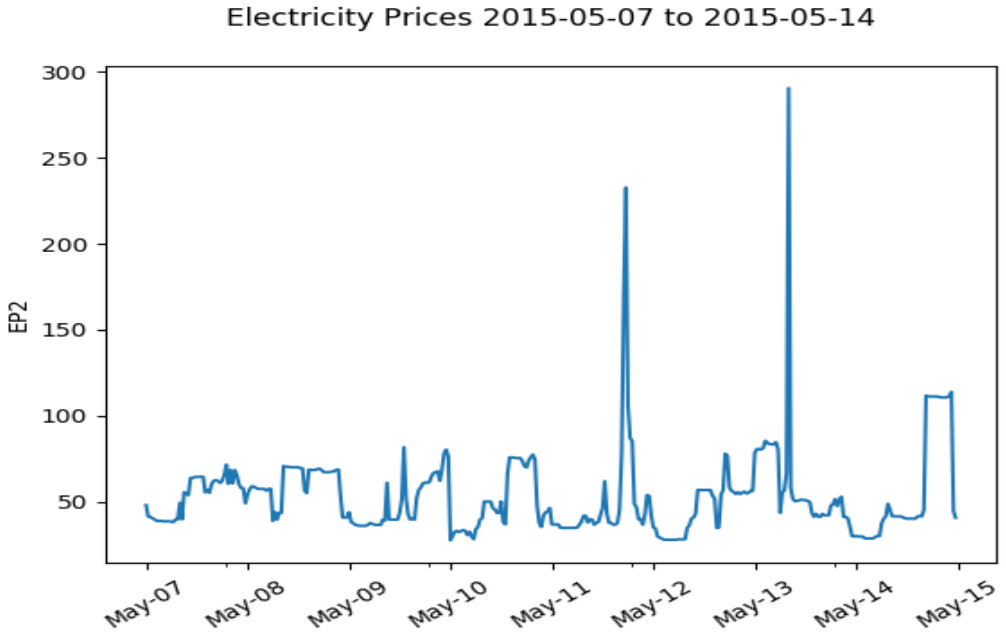


Figure 4. *Electricity prices over one week*

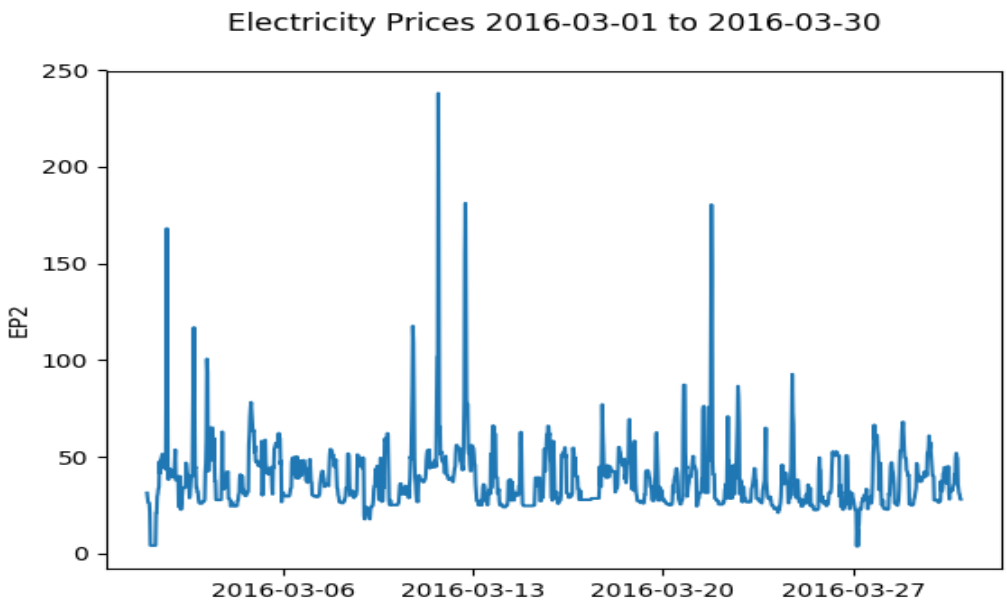


Figure 5. *Electricity prices over one month*

3.2.2 Univariate Profiling

From the histograms of the price information we can see that they do not have normal distributions, are highly positively skewed with a large degree of kurtosis. The load data appears to have a bimodal distribution with two distinct peaks.

Figure 6. *Price and load distributions*

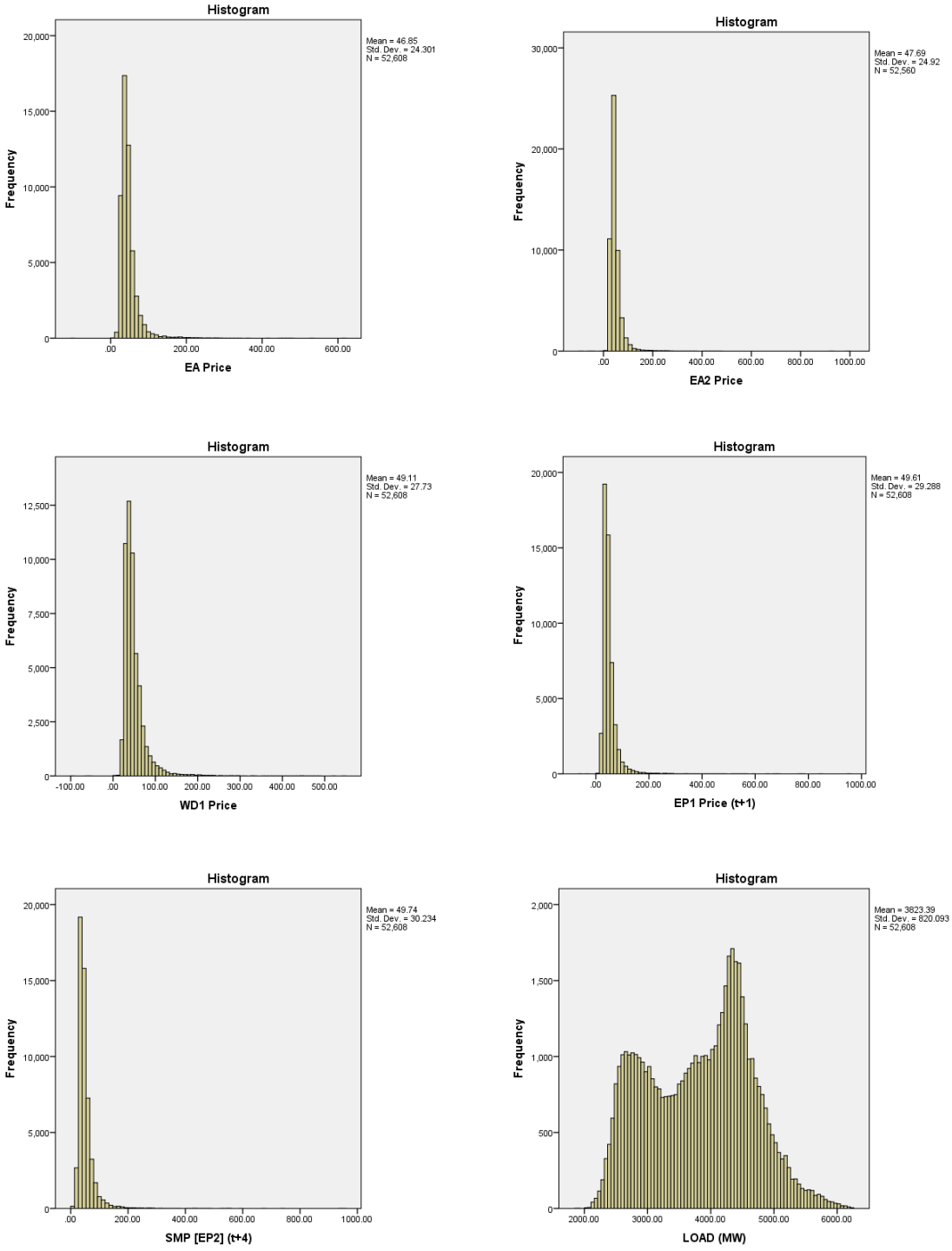
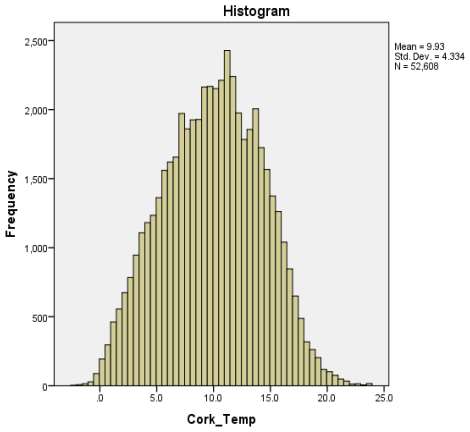
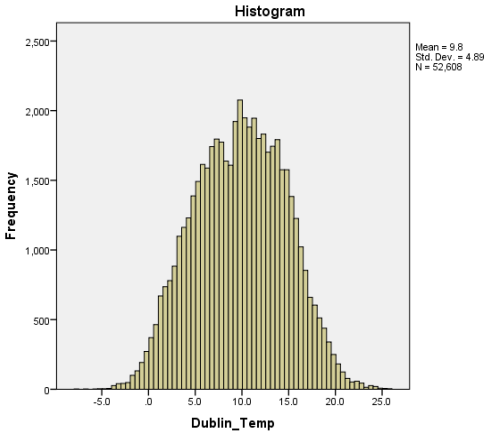


Figure 7. *Temperature distributions*





The wind data is highly skewed to the right as seen in Figure 8 below.

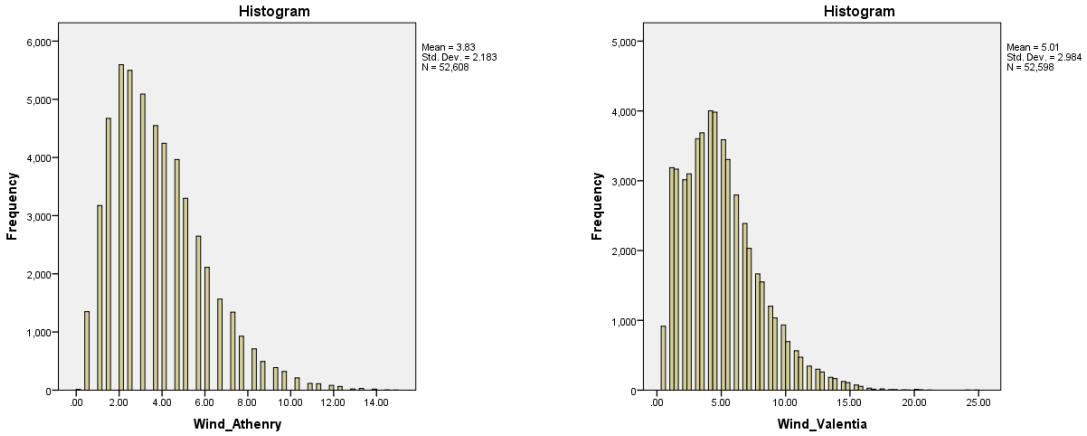


Figure 8. *Wind data distribution*

3.2.3 Bivariate Profiling

We used Pearson and Spearman's correlation coefficients to assess possible relationships between our predictors (EA, EA2, WD1, EP1, Load, Wind, Temp) and response variables (EP2). Looking at the scatterplots of these indicates we are not dealing with linear relationships. While Pearson is intended to describe linear relationships between normally distributed data, Spearman's ranking coefficient is less strict on linearity and also less sensitive to outliers. It is useful to see the results of Pearson versus Spearman. We can see that Spearman's values are greater than Pearson and this can indicate that we have non-linear monotonic components in the relationships. We get strong correlations between EA-EP2, EA2-EP2, LOAD-EP2 and WD1-EP2 looked at year-by-year and also taken over the complete three-year period. The highest correlation numbers are for EA-EP2 for both coefficients. (Correlation figures are included in the Appendix). We note that the EA estimate comes from a SEMO model that incorporates some metrological data. This may account for its strong correlation with the final settlement price, EP2.

The metrological data looks to be of limited value given its very low correlation numbers with EP2. This initially appears counter-intuitive, but if we consider the usage patterns of large commercial electricity subscribers (manufacturing, industrial usages, etc.) not being significantly influenced by weather patterns plus the fact that we already have metrological influences factored into the price from SEMO it becomes more understandable.

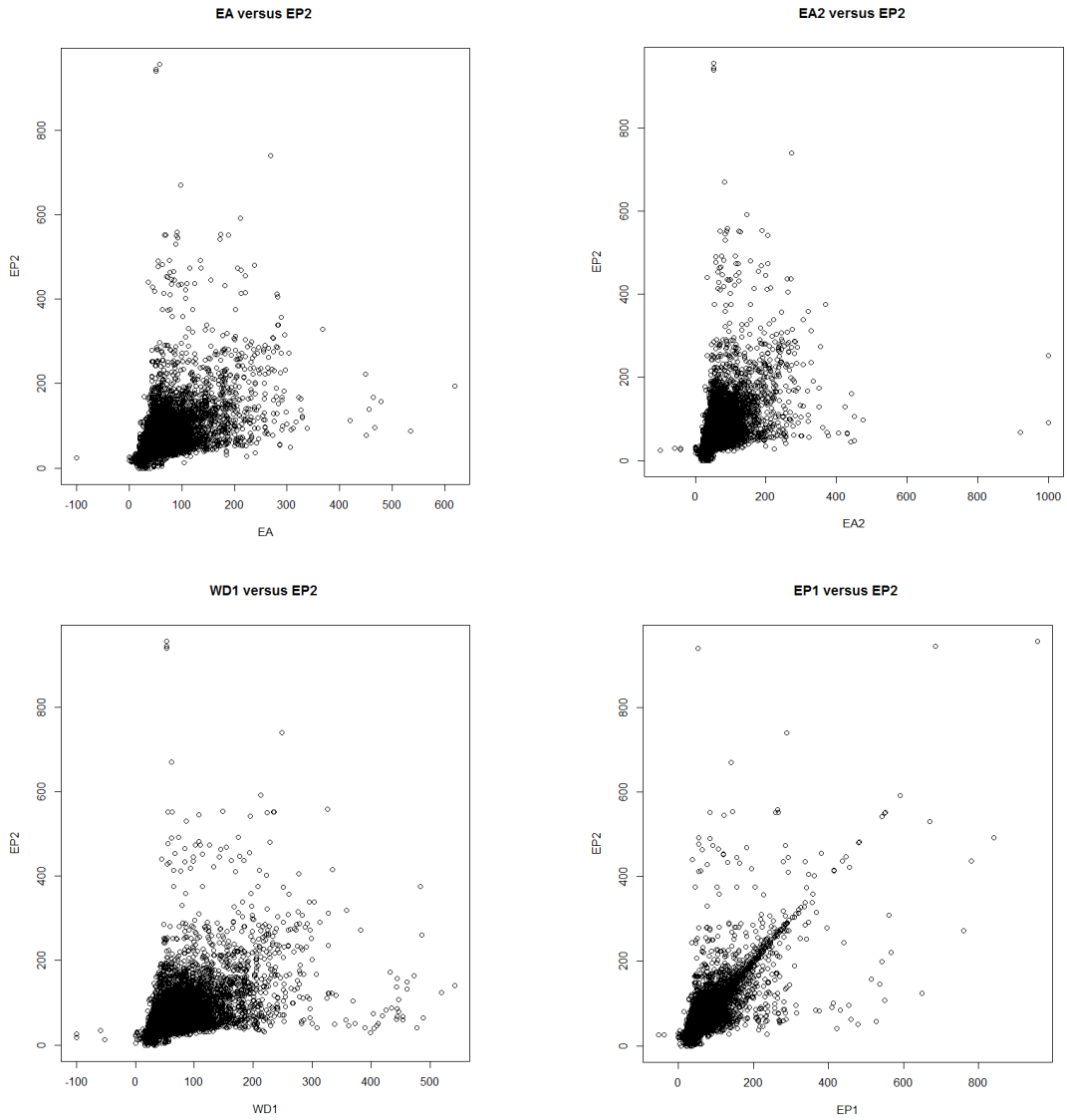


Figure 9. *Price bivariate studies*

3.2.4 Missing Data

We also observe a small number of missing price values. We will use the best estimate/actual value available for missing prices e.g. if EA value missing use EA2, EA2 missing use WD1, etc.

3.2.5 Outliers

We note the presence of extreme values in the price data. We cannot necessarily call these data points outliers as the electricity prices typically contain large spikes which represent real world price behavior. It is possible to have a zero or even negative wholesale electricity price. This represents the cost of production (restarting production process after outage) when demand/bids are low. For this reason no extreme values are removed from the price data series.

3.2.6 Data Transformations

We note from earlier that the price data is strongly skewed to the right. We may apply a $\log_{10}$ or natural log transform to the price data to provide a more normally distributed set. Because our input feature data set has a mixture of ranges we need to scale the data so that all features are weighed equally by our models. The standard approach here is to use mean standard deviation normalization, where the data is converted to have zero mean and unity standard deviation (Z-score normalization). We can also consider Min-Max normalisation which will retain the original distribution but transform it into a range of [0,1].

4 Empirical Evaluation

The requirement is to predict 48 prices for the 48 half hour slots for tomorrow based on our input feature data set. Our approach is to:

- perform feature selection based on feature importance derived from a Random Forest model
- tune model parameters using the same train/test cycle (1/1/2015 – 31/12/2015)
- train our machine learning models over a subset of the available data and then predict prices for the day after the last day of the training period (1/1/2016 – 25/12/2016)

Python was used to implement the evaluation using the Scikit-Learn machine learning library.

4.1 Feature Selection

Our input feature set consists of:

{ Slot, EA, EA2, WD1, Cork_Temp, Dublin_Temp, Wind_Athenry, Wind_Valentia }

The following features were added to the dataset in an attempt to capture some of the time related behaviour of the price data.

DayOfYear	DayOfMonth	DayOfWeek	WeekNumber	Month
(1 – 365)	(1 – 31)	(1 – 7)	(1 – 52)	(1 – 12)

The following time shifted features were added to capture price and load data from yesterday and the same day from last week.

EP2_SHIFT_7D	The EP2 price shift forward 7 days. This represents the time slot price for the same day/slot from last week.
DIFF_LOG_EP2_SHIFT_7D	The difference between the log10 of today's EP2 – log10 of yesterday's EP2 shifted forward 7 days.
LOAD_SHIFT_1D	The load shifted forward 1 day.
LOAD_SHIFT_7D	The load shifted forward 7 days.

We can see from the charts in 3.3.1 that the price has strong daily, weekly and monthly patterns. The Slot number may allow the models capture some of the daily cycles. Similarly,

the DayOfWeek and DayOfMonth may help capture weekly/monthly cycles. Identification of longer term seasonal patterns could be enabled with WeekNumber and Month features but our dataset and training period are probably too small for this to be significant.

With the addition of above features we now have 17 input features. We looked at their relative importance using a RandomForest model. This was more for curiosity than necessity as our feature set and data set size is relatively small and we could proceed as is. The RandomForest (RF) model tracks individual feature importance as part of its out-of-bag testing process. Running the complete feature set through a RF model using 200 training days and doing 20 iterations produced the following average importance figures.

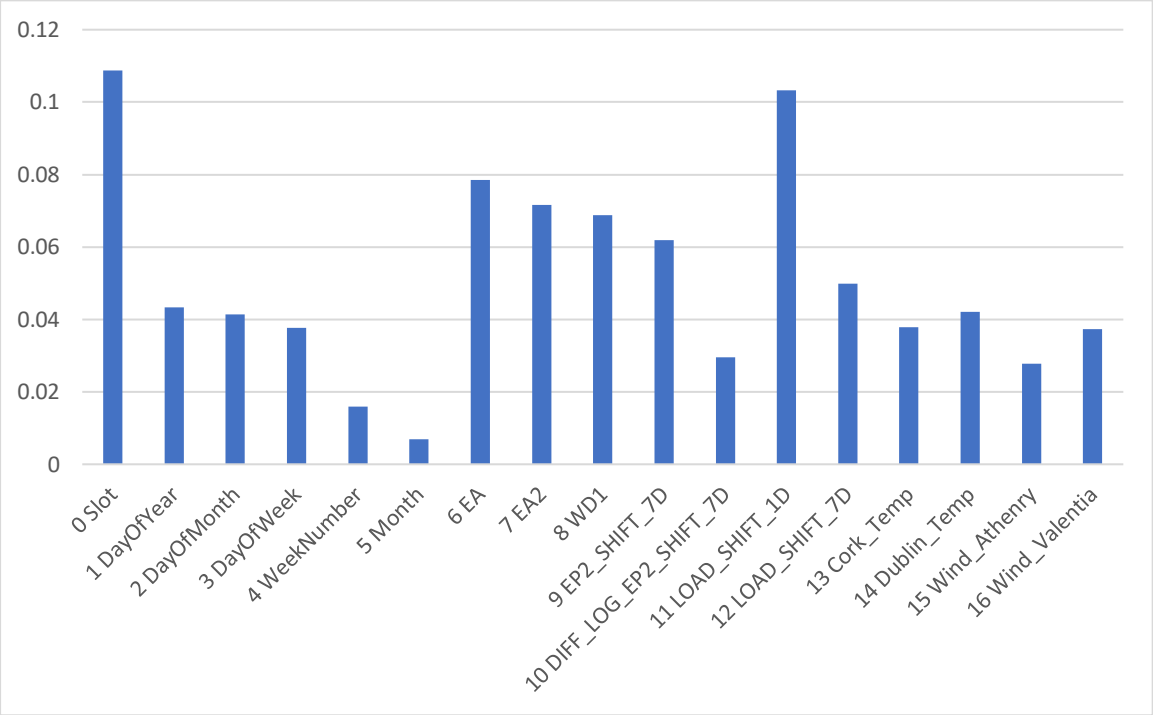


***Figure 10**. Feature Importance*

We can see the most prominent features are: Slot, EA, EA2, WD1, LOAD_SHIFT_1D. If we use 0.05 as the threshold for inclusion, we are left with:

***Table 3.** Reduced Feature Set*

Average importance of feature 0 **Slot**	0.108767010955
Average importance of feature 6 **EA**	0.0784367481515
Average importance of feature 7 **EA2**	0.0715487222997
Average importance of feature 8 **WD1**	0.0687146575462
Average importance of feature 10 **EP2_SHIFT_7D**	0.0619553179046
Average importance of feature 12 **LOAD_SHIFT_1D**	0.10325512281
Average importance of feature 13 **LOAD_SHIFT_7D**	0.0499160175521

Tests showed that there was only a slight loss of accuracy (< 0.5) by using the reduced feature set above.

4.2 Training period

The general advice on training period is the longer the better. On average this is backed up by testing with various training periods from 10 – 800 days for both our models. However, there is a significant increase in training times for both models as you increase the amount of training days, e.g. SVR (using the RBF kernel) training time increases by an order of magnitude going from predicting 360 days using 100 days training to using 400 days training. We select 360 days as the training period for SVR models as there is minimal accuracy improvements with longer periods, it approximates to a year of data plus further increases in training times become an issue for testing.

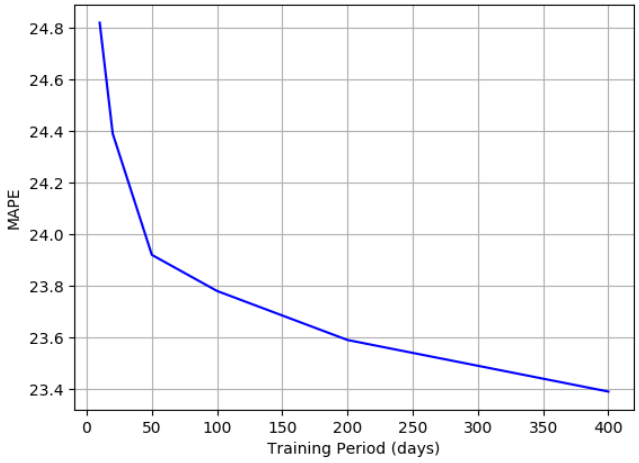


Figure 11. *Training period (days) versus MAPE for SVR*

For the Neural Network, we test accuracy improvements over a range of 10 – 400 training days. We see that MAPE improves slightly up to 100 training days and then increases thereafter. We therefore select 100 as out optimum number of training days for the neural network model.

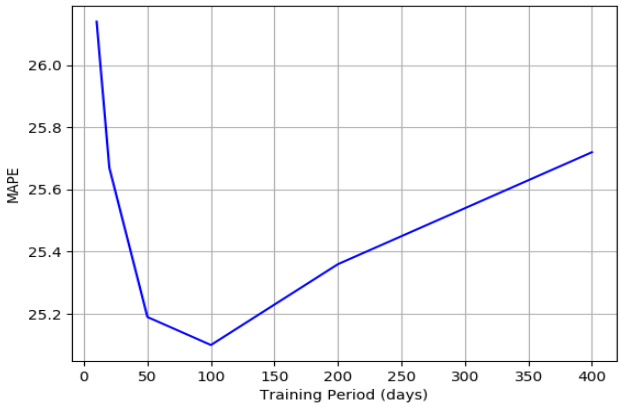


Figure 12. *Training period (days) versus MAPE for Neural Network*

4.3 Parameter Tuning

4.3.1 SVR

The two most important parameters for tuning a SVR common to all kernels are C, the penalty factor and ε, the width of the no penalty zone used in fitting the training data. C determines the trade-off between model complexity and error frequency. ε decides the level of accuracy of the model, set to zero would likely cause overfitting and set too large would produce inaccurate predictions. It has been shown to be directly linked to the level of noise in the input training data [26]. Parameter selection was performed using basic trial and error as we were not able to use cross validation methods with our time series data. The default choice of C=1 and $\varepsilon = 0.1$ proved to be the best for our application. Kernel selection was also done through experimentation and the radial basis function was found to give the best performance.

4.3.2 Neural Network

The first area to consider for NNs is the number of hidden layers and the number of neurons in each layer. It is considered that most non-linear functions can be approximated by a NN using just a single hidden layer. There are several rule-of-thumbs for selecting the number of neurons for the hidden layers. The danger is using excess neurons that results in overfitting the training data and poor performance on generalising to new unseen data. Some of the heuristics for picking the number of neurons include:

- For a three layer network with n input and m output neurons, the hidden layer would have sqrt(n∗m) neurons" [27]
- From [28] we have the number of hidden neurons as:
    - between the size of the input layer and the size of the output layer or
    - 2/3 the size of the input layer, plus the size of the output layer.
    - less than twice the size of the input layer.

Again, as N-fold cross validation was not available, trial and error was used with the above heuristics. The best performance was achieved using 13 neurons in one hidden layer.

The sigmoid activation function was used along with the stochastic gradient descent algorithm for weight optimisation. This was the best performing combination found from the range of options available in the scikit implementation.

The momentum parameter is the fraction of the previous weighting used when updating the gradient descent and is a value between 0 and 1. It is recommended [scikit ref] to tune the momentum parameter when using gradient descent for best performance. Manual trial and error indicated a value 0.3 produced the best trade-off between training time and performance.

The learning rate parameter describes the schedule used to update the weights. The default behaviour is to keep a constant learning rate. An adaptive learning is recommended to get the best out of the gradient descent algorithm. This involves keeping the rate constant until such time as two consecutive rounds of updates fail to decrease the training loss by the tolerance setting. At this point the scikit neural network implementation divides the learning rate by 5.

4.4 Testing

Cross fold validation is the main method of model testing. N-fold cross validation will split the data set into N-folds, N-1 folds being used for training and the N^{th} used for testing. Each fold will be used in turn as the test fold e.g. fold 5 might be used as test, with folds [1-4,6-10] used as training for a 10-fold test. This method can't be applied to our dataset as the price data is time order dependent. We can't train on future prices and then test on past data. Therefore, we use a walk forward style of train/testing. This involves training for specified number of days, predicting for the next day after the last day of training and then moving the training window forward one day and repeating the train/test cycle.

4.5 Results

We predict prices from 1/1/2016 to 25/12/2016 using a training cycle of 360 days for the support vector regressor (SVR) and 100 days for the neural network (NN). We add three other models for comparison: a Random Forest (RF), a Gradient Boosted model (GB) and the ExtraTreesRegressor (ExtraTree). Included also are the two ex. ante price forecasts generated by SEMO (SEMO-EA, SEMO-EA2). Note we used the period 1/1/2016 to 25/12/2016 for training the three new models. The RMSE[4] and MAPE were calculated for each model.

[4] RMSE the Root Mean Squared error is another popular measure of the accuracy of prediction models and is defined as $\sqrt{\frac{1}{n}\sum_{i=1}^{n}(Forecast_i - Actual_i)^2}$

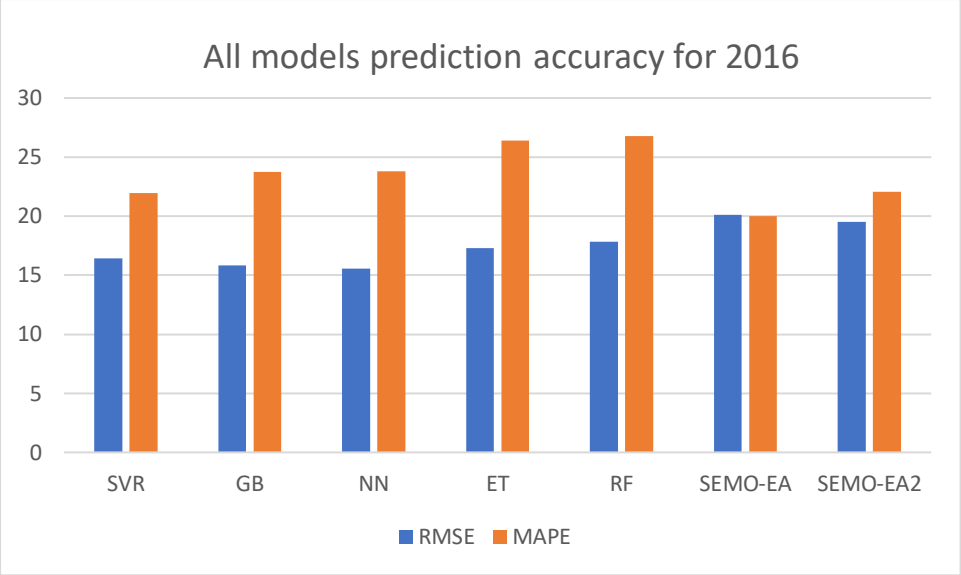


Figure 13. *All model prediction accuracy for 2016*

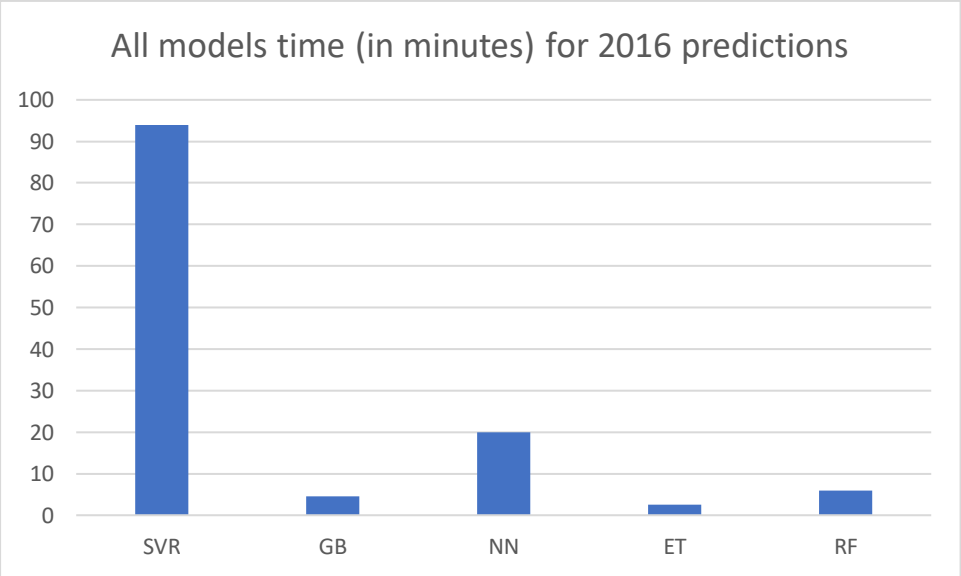


Figure 14. *All models time (in minutes) for 2016 predictions*

4.6 Interpretation

The results show that the support vector regression achieves the best MAPE amongst our models but takes over four times longer that the next highest execution time associated with the neural network model. The gradient boosted trees model comes in at second place with an order magnitude faster execution time. The GB and NN model's accuracy are very similar but we note the neural network has an execution time of 20 minutes. The extra trees regressor model has the fastest execution time at under 3 minutes but poor MAPE and RMSE figures.

The two price estimates generated by SEMO, EA and EA2 perform better than all our predictions bar the Support Vector which has an MAPE (21.97) value between the EA (20.0) and EA2 (22.04) figures. Data on runtime for generating EA and EA2 are not available.

Overall the prediction accuracy is poor. It is suspected that one of the reasons for this is the fact that our data set is made up of several forecasted features which register as being significant and correlated to the actual price but do not have a high predictive ability for day ahead price prediction. We also note that the price time series shows extreme volatility at the intraday and daily timeframes which makes price prediction especially difficult.

All tests were performed on the same hardware[5] so the wide range in execution times is also noteworthy. It is clear that training the support vector is very inefficient when compared to the other models with no outstanding improvement in prediction performance. Other kernels (polynomial and sigmoid) were used without any improvements in training times. There may be scope to improve on the SVR performance with a fully automated method of tuning its parameters. As expected the NN takes comparable long to train also with the ensemble models proving to be the most memory efficient and fastest to train.

5 Conclusion

We performed basic tuning, training and prediction on SEMO electricity price time series using two main models: a neural network and a support vector regressor. The support vector regressor gave the best accuracy figures with the worst execution time. Overall the prediction accuracy was poor and the training times for both NN and SVR were not suitable for anything close to a real-time application. Three other models: gradient boosted trees, random forest and extra tree regressors were also evaluated for comparison with our two main models. The gradient boosted model performs close to the SVR and NN with a much better execution time.

Electricity prices can behave very differently from one half hour to the next over the course of a full day. Very large intraday spikes in price that can occur at any time of the day. The evidence of this study shows it is very difficult to get usable predictions from a single model. Hybrid models have become popular for use with time series data in an attempt to first decompose or simplify the time series data into a set of more predictable data streams that are then used as input to multiple forecasting models. Some of the decompositions techniques used include wavelet transforms and seasonal ARIMA approaches frequently combined with NN or SVMs. These methods look promising based on accuracy levels achieved as reported in the research literature.

Another hybrid style discovered in the course of this study performed clustering of half hour slots that display similar price behaviour as a pre-processing step using self-organising maps with the output then being feed into multiple SVR models. Each of the SVR models could then be individually optimized.

[5] All tests run on laptop with dual core i7-6600 CPU @ 2.6 GHz, 2.81GHz, 16G solid state RAM under Windows 10.

Any further study with electricity price time series data and machine learning techniques would have to involve sourcing or implementing some means of performing automated tuning of model parameters. Time did not permit development of this functionality in this study but it would be essential in order to get the best available performance from our models.

The gradient boosted model achieved a MAPE figure as good as our best model with an order of magnitude faster training time. Therefore, this model may be worth further study for use on this dataset or any other electricity price time series.

6 Appendix A – Data tables

Table 4. *Spearman's Correlation Coefficients by individual years (2014 – 2016)*

						2014	2015	2016
EA	EA2	WD1	EP1	**EP2**		0.6625209	0.6316781	0.7312278
EA	**EA2**	WD1	EP1	**EP2**		0.6545279	0.6280589	0.7455329
EA	EA2	**WD1**	EP1	**EP2**		0.6471408	0.6095887	0.7041107

						2014	2015	2016
EA	EA2	WD1	**EP1**	**EP2**		0.8226924	0.7615168	0.8530156

						2014	2015	2016
Load	WS Athenry	WS Valentia	Temp. Cork	Temp. Dublin	**EP2**	0.6709379	0.5110448	0.5510058
Load	**WS Athenry**	WS Valentia	Temp. Cork	Temp. Dublin	**EP2**	-0.1267884	-0.2249783	-0.1559703
Load	WS Athenry	**WS Valentia**	Temp. Cork	Temp. Dublin	**EP2**	-0.1131185	-0.2613176	-0.1601656
Load	WS Athenry	WS Valentia	**Temp. Cork**	Temp. Dublin	**EP2**	-0.0304153	0.0734188	-0.0129225
Load	WS Athenry	WS Valentia	Temp. Cork	**Temp. Dublin**	**EP2**	-0.0119313	0.0734799	-0.0169361

						2014	2015	2016
WS Athenry	WS Valentia	Temp. Cork	Temp. Dublin	**Load**		0.1302110	0.1178707	0.1069367
WS Athenry	**WS Valentia**	Temp. Cork	Temp. Dublin	**Load**		0.1014226	0.0905732	0.0538684
WS Athenry	WS Valentia	**Temp. Cork**	Temp. Dublin	**Load**		-0.0485790	-0.0227416	-0.1044191
WS Athenry	WS Valentia	Temp. Cork	**Temp. Dublin**	**Load**		-0.0192488	-0.0032299	-0.0842355

Table 5. *Spearman's Correlation Coefficients for period (2014 – 2016)*

						2014 - 2016
EA	EA2	WD1	EP1	**EP2**		0.7132343
EA	**EA2**	WD1	EP1	**EP2**		0.7103159
EA	EA2	**WD1**	EP1	**EP2**		0.6888726

						2014 - 2016
EA	EA2	WD1	**EP1**	**EP2**		0.8327827

						2014 - 2016
Load	WS Athenry	WS Valentia	Temp. Cork	Temp. Dublin	**EP2**	0.5249414
Load	**WS Athenry**	WS Valentia	Temp. Cork	Temp. Dublin	**EP2**	-0.1483375
Load	WS Athenry	**WS Valentia**	Temp. Cork	Temp. Dublin	**EP2**	-0.1561005
Load	WS Athenry	WS Valentia	**Temp. Cork**	Temp. Dublin	**EP2**	0.01040464
Load	WS Athenry	WS Valentia	Temp. Cork	**Temp. Dublin**	**EP2**	0.01389712

						2014 - 2016
WS Athenry	WS Valentia	Temp. Cork	Temp. Dublin	**Load**		0.1167467
WS Athenry	**WS Valentia**	Temp. Cork	Temp. Dublin	**Load**		0.08087498
WS Athenry	WS Valentia	**Temp. Cork**	Temp. Dublin	**Load**		-0.0647729
WS Athenry	WS Valentia	Temp. Cork	**Temp. Dublin**	**Load**		-0.0385965

Table 6. *Pearson's Correlation Coefficients by individual year (2014 – 2016)*

						2014	2015	2016
EA	EA2	WD1	EP1	**EP2**		0.5533	0.5518	0.6244
EA	**EA2**	WD1	EP1	**EP2**		0.5487	0.5055	0.5667
EA	EA2	**WD1**	EP1	**EP2**		0.5605	0.5391	0.5649

						2014	2015	2016
EA	EA2	WD1	**EP1**	**EP2**		0.7998	0.7709	0.7789

						2014	2015	2016
Load	WS Athenry	WS Valentia	Temp. Cork	Temp. Dublin	**EP2**	0.5009	0.4516	0.4370
Load	**WS Athenry**	WS Valentia	Temp. Cork	Temp. Dublin	**EP2**	-0.118634	-0.1867152	-0.1309634
Load	WS Athenry	**WS Valentia**	Temp. Cork	Temp. Dublin	**EP2**	-0.07110487	-0.1799707	-0.1186908
Load	WS Athenry	WS Valentia	**Temp. Cork**	Temp. Dublin	**EP2**	-0.07300457	-0.0375616	-0.0479619
Load	WS Athenry	WS Valentia	Temp. Cork	**Temp. Dublin**	**EP2**	-0.06032596	-0.0334363	-0.0495827

						2014	2015	2016
WS Athenry	WS Valentia	Temp. Cork	Temp. Dublin	**Load**		0.1094094	0.1050051	0.08530654
WS Athenry	**WS Valentia**	Temp. Cork	Temp. Dublin	**Load**		0.09318572	0.08304193	0.0347087
WS Athenry	WS Valentia	**Temp. Cork**	Temp. Dublin	**Load**		-0.0268596	-0.0341733	-0.0908528
WS Athenry	WS Valentia	Temp. Cork	**Temp. Dublin**	**Load**		0.00226837	-0.0037804	-0.0689811

Table 7. *Pearson Correlation Coefficient for period (2014 – 2016)*

						2014 - 2016
EA	EA2	WD1	EP1	**EP2**		0.5892785
EA	**EA2**	WD1	EP1	**EP2**		0.5635247
EA	EA2	**WD1**	EP1	**EP2**		0.5731585

						2014 - 2016
EA	EA2	WD1	**EP1**	**EP2**		0.7947925

						2014 - 2016
Load	WS Athenry	WS Valentia	Temp. Cork	Temp. Dublin	**EP2**	0.4368797
Load	**WS Athenry**	WS Valentia	Temp. Cork	Temp. Dublin	**EP2**	-0.131081
Load	WS Athenry	**WS Valentia**	Temp. Cork	Temp. Dublin	**EP2**	-0.1062432
Load	WS Athenry	WS Valentia	**Temp. Cork**	Temp. Dublin	**EP2**	-0.04711493
Load	WS Athenry	WS Valentia	Temp. Cork	**Temp. Dublin**	**EP2**	-0.04400337

						2014 - 2016
WS Athenry	WS Valentia	Temp. Cork	Temp. Dublin	**Load**		0.09754446
WS Athenry	**WS Valentia**	Temp. Cork	Temp. Dublin	**Load**		0.06898039
WS Athenry	WS Valentia	**Temp. Cork**	Temp. Dublin	**Load**		-0.0537402
WS Athenry	WS Valentia	Temp. Cork	**Temp. Dublin**	**Load**		-0.0260499

Table 8. *Feature Importance from Random Forest*

Average importance of feature 0 **Slot**	0.108767010955
Average importance of feature 1 **DayOfYear**	0.0432982890229
Average importance of feature 2 **DayOfMonth**	0.0413690630751
Average importance of feature 3 **DayOfWeek**	0.0375938352021
Average importance of feature 4 **WeekNumber**	0.0159390418847
Average importance of feature 5 **Month**	0.00690751752276
Average importance of feature 6 **EA**	0.0784367481515
Average importance of feature 7 **EA2**	0.0715487222997
Average importance of feature 8 **WD1**	0.0687146575462
Average importance of feature 9 **EP2_SHIFT_7D**	0.0619553179046
Average importance of feature 10 **DIFF_LOG_EP2_SHIFT_7D**	0.0296223042235
Average importance of feature 11 **LOAD_SHIFT_1D**	0.10325512281
Average importance of feature 12 **LOAD_SHIFT_7D**	0.0499160175521
Average importance of feature 13 **Cork_Temp**	0.037839197373
Average importance of feature 14 **Dublin_Temp**	0.0420325369534
Average importance of feature 15 **Wind_Athenry**	0.027746678703
Average importance of feature 16 **Wind_Valentia**	0.0372517823809

Table 9. *All Model Prediction results 2016*

	RMSE	**MAPE**	**TIME (m)**
SVR	16.42	21.97	94
GB	15.84	23.72	4.5
NN	15.58	23.77	20
ExtraTree	17.31	26.39	2.5
RF	17.85	26.79	6
SEMO-EA	20.09	20.00	NA
SEMO-EA2	19.50	22.04	NA

7 Bibliography

[1] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bull. Math. Biophys.*, vol. 5, no. 4, pp. 115–133, Dec. 1943.

[2] F. Rosenblatt, "Principles of Neurodynamics. Perceptrons and the Theory of brain mechanisms." Archives of General Psychiatry, vol. 7 (1962) pp. 218-219, 1961.

[3] M. Minsky and S. Papert, *Perceptrons : an introduction to computational geometry*. MIT Press, 1988.

[4] R. J. Rumelhart, David E., Hinton, Geoffery E., Williams, "Learning representations by back propagation of errors," *Nature*, vol. 323, 1986.

[5] M. Adya and F. Collopy, "How effective are neural networks at forecasting and prediction? A review and evaluation," *J. Forecast.*, vol. 17, no. 5–6, pp. 481–495, 1998.

[6] H. Y. Yamin, S. M. Shahidehpour, and Z. Li, "Adaptive short-term electricity price forecasting using artificial neural networks in the restructured power markets," *Int. J. Electr. Power Energy Syst.*, vol. 26, no. 8, pp. 571–581, 2004.

[7] J.-J. Guo and P. B. Luh, "Improving Market Clearing Price Prediction by Using a Committee Machine of Neural Networks," *IEEE Trans. Power Syst.*, vol. 19, no. 4, pp. 1867–1876, Nov. 2004.

[8] P. Mandal, T. Senjyu, and T. Funabashi, "Neural networks approach to forecast several hour ahead electricity prices and loads in deregulated market," *Energy Convers. Manag.*, vol. 47, no. 15–16, pp. 2128–2142, 2006.

[9] R. Gareta, L. M. Romeo, and A. Gil, "Forecasting of electricity prices with neural networks," *Energy Convers. Manag.*, vol. 47, no. 13–14, pp. 1770–1778, 2006.

[10] A. J. Conejo, J. Contreras, R. Espínola, and M. A. Plazas, "Forecasting electricity prices for a day-ahead pool-based electric energy market," *Int. J. Forecast.*, vol. 21, no. 3, pp. 435–462, 2005.

[11] V. Vahidinasab, S. Jadid, and A. Kazemi, "Day-ahead price forecasting in restructured power systems using artificial neural networks," *Electr. Power Syst. Res.*, vol. 78, no. 8, 2008.

[12] J. P. S. Catalão, S. J. P. S. Mariano, V. M. F. Mendes, and L. A. F. M. Ferreira, "Short-term electricity prices forecasting in a competitive market: A neural network approach," *Electr. Power Syst. Res.*, vol. 77, no. 10, pp. 1297–1304, 2007.

[13] P. S. Georgilakis, "Artificial Intelligence Solution To Electricity Price Forecasting Problem," *Appl. Artif. Intell.*, vol. 21, no. 8, pp. 707–727, 2007.

[14] S. K. Aggarwal, L. M. Saini, and A. Kumar, "Electricity price forecasting in deregulated markets: A review and evaluation," *Int. J. Electr. Power Energy Syst.*, vol. 31, no. 1, pp. 13–22, 2009.

[15] B. Neupane, K. S. Perera, Z. Aung, and W. L. Woon, "Artificial Neural Network-based Electricity Price Forecasting for Smart Grid Deployment," *IEEE*, 2013.

[16] N. K. Singh, M. Tripathy, and A. K. Singh, "A radial basis function neural network approach for multi-hour short term load-price forecasting with type of day parameter," in *2011 6th International Conference on Industrial and Information Systems*, 2011, pp. 316–321.

[17] Z. A. Bashir and M. E. El-Hawary, "Applying Wavelets to Short-Term Load Forecasting Using PSO-Based Neural Networks," *IEEE Trans. Power Syst.*, vol. 24, no. 1, pp. 20–27, Feb. 2009.

[18] D. Srinivasan, F. C. Yong, and A. C. Liew, "Electricity Price Forecasting Using Evolved Neural Networks," in *2007 International Conference on Intelligent Systems*

Applications to Power Systems, 2007, pp. 1–7.
[19] H. M. I. Pousinho, V. M. F. Mendes, and J. P. S. Catalão, "Short-term electricity prices forecasting in a competitive market by a hybrid intelligent approach," *Int. J. Electr. Power Energy Syst.*, vol. 39, no. 1, pp. 29–35, 2012.
[20] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A Training Algorithm for Optimal Margin Classiiers," *Proc. fifth Annu. Work. Comput. Learn. theory*, pp. 144–152, 1992.
[21] A. T. Mohamed and M. E. El-hawary, "Mid-Term Electricity Price Forecasting Using SVM," 2016.
[22] D. C. Sansom, T. Downs, and T. K. Saha, "Support vector machine based electricity price forecasting for electricity markets utilising projected assessment of system adequacy data," *IPEC 2003 - 6th Int. Power Eng. Conf.*, no. November, pp. 839–844, 2003.
[23] A. R.-K. and B. M. Ali Shiriz, Mohammad Afsharx and C. of E. zSNL/CIPCE, School of Electrical and Computer Engineering, "Electricity Price Forecasting Using Support Vector Machines by Considering Oil and Natural Gas Price Impacts," pp. 2–6, 2015.
[24] C. Gao, E. Bompard, R. Napoli, and H. Cheng, "Price forecast in the competitive electricity market by support vector machine," *Phys. A Stat. Mech. its Appl.*, vol. 382, no. 1, pp. 98–113, 2007.
[25] SEMO, "Home," *http://www.sem-o.com/*.
[26] V. Cherkassky and Y. Ma, "Practical selection of SVM parameters and noise estimation for SVM regression."
[27] T. Masters, *Practical neural network recipes in C++*. .
[28] J. Heaton, *Introduction to neural networks with Java*. Heaton Research, 2008.