Abstract

Electricity supply must be matched with demand at all times. This helps reduce the chances of problems with load frequency control and the chances of electricity blackouts. To gain a better understanding of the load that is likely to be required over the next 24 hours, estimations under uncertainty are needed. This is especially difficult in a decentralized electricity market with many micro-producers who are not under central control.

In this paper, we investigate the impact of 11 offline learning and five online learning algorithms to predict the electricity demand profile over the next 24 hours. We achieve this through integration within the long-term agent-based model, ElecSim. These algorithms include multilayer perceptron (MLP) neural networks, support vector regression and linear regression. By predicting the electricity demand profile over the next 24 hours, we can simulate the predictions made for a day-ahead market. Once we have made these predictions, we sample from the residual distributions and perturb the electricity market demand using the simulation, ElecSim. This enables us to understand the impact of errors on the long-term dynamics of a decentralized electricity market.

Our results show that we are able to reduce the mean absolute error by 30% using an online algorithm when compared to the best offline model, whilst reducing the required tendered national grid reserve required. Such a reduction allows for a smaller required tendered grid reserve, saving costs and emissions. We also show that large errors in prediction accuracy have a disproportionate error on investments made over a 17-year time frame, as well as electricity mix.

Keywords: Long-Term Energy Modelling, Online learning, Machine learning, Market investment, Climate Change, Investment Decisions, Machine Learning, Forecasting, Electricity Demand

1. Introduction

The integration of higher proportions of intermittent renewable energy sources (IRES) in the electricity grid will mean that the forecasting of electricity demand will become increasingly important and challenging. Examples of IRES are solar panels and wind turbines, which fluctuate in terms of power output based on localized wind speed and solar irradiance. However, as supply must meet demand at all times and the fact that IRES are less predictable than dispatchable energy sources such as coal and combined-cycle gas turbines (CCGTs) this means extra attention must be made in predicting future demand if we wish to keep, or better reduce, the current frequency of blackouts [1]. A dispatchable source is one that can be turned on and off by human control and therefore, able to adjust output just in time, at a moment convenient for the grid.

Typically, peaker plants, such as reciprocal gas engines, are used to fill fluctuations in demand, that had not been previously planned for. Specifically, peaker plants meet the peaks in demand where other cheaper options are at full capacity. These peaker plants are typically expensive to run and have higher greenhouse gas emissions than their non-peaker counterparts [2]. Whilst peaker plants are also dispatchable plants, not all dispatchable plants are peaker plants. For example coal, which is a dispatchable plant, is run as a base load plant, due to its inability to deal with the fluctuating conditions required of a peaker plant.

To reduce reliance on peaker plants, it is helpful to know how much electricity demand there will be in the future so that more efficient plants can be used to meet this expected demand. This is so that these more efficient plants can be brought up to speed at a time suitable to match the demand. Forecasting a day into the future is especially useful in decentralized electricity markets which have day-ahead markets. Decentralized electricity markets are ones where electricity is provided by multiple generation companies, as opposed to a centralized source, such as a government. To aid in this prediction, machine learning and statistical techniques have been used to accurately predict demand based on several different factors and data sources [3], such as weather [4], day of the week [5] and holidays [6].

Various studies have looked at predicting electricity demand at various horizons [7, 8, 9]. However, the impact of poor demand predictions on the long-term electricity mix has been studied to a lesser degree.

In this paper, we compare several machine learning and statistical techniques to predict the energy demand for each hour over the next 24-hour horizon. We chose to predict over the next 24 hours to simulate a day-ahead market, which is often seen in decentralized electricity markets. Our approach, however, could be utilized for differing time horizons. In addition to this, we use our long-term agent-based model, ElecSim [10, 11], to simulate the impact of different forecasting methods on long-term investments, power plant usage and carbon emissions for the years 2018 through 2035 in the United Kingdom. Our approach is generalizable to any country through parametrization of the ElecSim simulation.

As part of our work, we utilize online learning methods to improve the accuracy of our predictions. Online learning methods can learn from novel data while maintaining what was learnt from previous data. Online learning is useful for non-stationary datasets, and time-series data wher e recalculation of a model would take a prohibitive amount of time. Offline learning methods, however, must be retrained every time new data is added. Online approaches are constantly updated and do not require significant pauses while the offline training is being re-run.

We trial different algorithms and train different models for different times of the year. Specifically, we train different models for the different seasons. We also split weekdays and train both weekends and holidays together. This is due to the fact that holidays and weekend exhibit similar load profiles due to the reduction in industry electricity use and an increase in domestic. This enables a model to become good at a specific subset of the data which share similar patterns, as opposed to having to generalize to all of the data. Examples of the algorithms used are linear regression, lasso regression, random forests, support vector regression, MLP neural network, box-cox transformation linear regression and the passive aggressive model.

We expect a-priori that online algorithms will outperform the offline approach. This is due to the fact that the demand time-series is non-stationary, and thus changes sufficiently over time. In terms of the models, we presume that the machine learning algorithms, such as neural networks, support vector regression and random forests will outperform the statistical methods such as linear regression, lasso regression and box-cox transformation regression. We expect this due to the fact that machine learning has been shown to be able to learn more complex feature representations than statistical methods [7]. In addition, our previous work has shown that the random forest was able to outperform neural networks, support vector regression and long short term memory neural networks (LSTM) [12].

However, it should be noted, that such a-priori intuiton, is no substitute for analytical evidence and can (and has) been shown to be wrong in the past, due to imperfect knowledge of the data and understanding of some of the black box models, such as neural networks.

Using online and offline methods we take the residuals and fit a variety of distributions to them. We choose the distribution with the lowest sum of squared estimate of errors (SSE). SSE was chosen as the metric to ensure that both positive and negative errors were treated equally, as well as ensuring that large errors were penalized more than smaller errors. We fit over 80 different distributions, which include the Johnson Bounded distribution, the uniform distribution and the gamma distribution. The distribution that best fits the respective residuals is then used and sampled from to adjust the demand in the ElecSim model. We then observe the differences in carbon emissions, and which types of power plants were both invested in and utilized, with each of the different statistical and machine learning methods. To the best of our knowledge, this is the most comprehensive evaluation of online learning techniques to the application of day-ahead load forecasting as well as assessing the impacts of the errors that these models produce on the long-term electricity market dynamics.

We show that online learning has a significant impact on reducing the error for predicting electricity consumption a day ahead when compared to traditional offline learning techniques, such as MLP neural networks, linear regression, extra trees regression and support vector regression (SVR), which are models used in the literature [1, 13, 14]. For a full list of algorithms used in this work see Table 1.

We show that the forecasting algorithm has a non-negligible impact on carbon emissions and use of coal, onshore, photovoltaics, reciprocal gas engines and CCGT. Specifically, the amount of coal, photovoltaics, and reciprocal gas used from 2018 to 2035 was proportional to the median absolute error, while both onshore and offshore wind are inversely proportional to the median absolute error. Total investments in coal, offshore and photovoltaics are proportional to the median absolute error, while investments in CCGT, onshore and reciprocal gas engines are inversely proportional.

The contributions of this work are:

1. The evaluation of different online and offline learning models to forecast the electricity demand profile 24 hours ahead.
2. Evaluation of poor predictive ability on the long-term electricity market in the UK through the perturbation of demand in the ElecSim simulation.

In Section 2, we review the literature, including other uses of online learning algorithms and an application to electricity markets. We introduce the dynamics of the ElecSim simulation as well as the methods used in Section 3. We demonstrate the methodology undertaken in Section 4. In Section 5 we demonstrate our results, followed by a discussion in Section 6. We conclude our work in Section 7.

2. Literature Review

Multiple papers have looked at demand-side forecasting [7]. These include both artificial intelligence [15, 16, 17] and statistical techniques [8, 18]. To the best of our knowledge, the impact of online learning has been discussed with less frequency. In addition to this, our research models the impact of the performance of different algorithms on investments made, electricity sources dispatched and carbon emissions over a 17 year period. To model this, we use the long-term electricity market agent-based model, ElecSim.

2.1. *Offline learning*

Multiple electricity demand forecasting studies have been undertaken for offline learning [14, 19, 20]. Studies have been undertaken using both smart meter data, as well as with aggregated demand, similar to the work in this paper. Smart meters are a type of energy meter installed in each house, which monitor electricity usage at short intervals, such as every 15 or 30 minutes.

Fard *et al.* propose a new hybrid forecasting method based on the wavelet transform, autoregressive integrated moving average (ARIMA) and artificial neural network (ANN) [21]. The ARIMA model is utilized to capture the linear component of the time series, with the residuals containing the non-linear components. The non-linear components are decomposed using the discrete wavelet transform, which finds the sub-frequencies. These residuals are then used to train an ANN to predict the future residuals. Finally, the ARIMA and ANNs outputs are summed. They show that through this method they are able to improve their predictive ability.

Humeau *et al.* compare MLPs, SVRs and linear regression at predicting smart meter data [22]. They aggregate different households and observe which models work the best at each aggregate level. They find that linear regression outperforms both MLP and SVR when forecasting individual households. After aggregating over 32 households, SVR outperforms linear regression.

Quilumba *et al.* also apply machine learning techniques to individual households' electricity consumption by aggregation [21]. To achieve this aggregation, they use k-means clustering to aggregate the households to improve their forecasting ability. The authors also use a neural network based model for forecasting, and show that the number of optimum clusters for forecasting is dependent on the data, with three clusters optimal for a particular dataset, and four for another.

In our previous work we evaluate the performance of ANNs, random forests, SVR and long short-term memory neural networks [3]. We utilize smart meter data, and cluster by household using the k-means clustering algorithm to aggregate groups of demand. We find that through this clustering we are able to reduce the error, with the random forest performing the best.

2.2. *Online learning*

There have been several studies in diverse applications on the use of online machine learning to predict time-series data, however, to the best of our knowledge there are limited examples where this is applied to electricity markets. In our work, we trial a different set of algorithms to our problem. Due to time constraints, we do not trial the additional techniques discussed in this literature review within our paper.

Johansson *et al.* apply online machine learning algorithms for heat demand forecasting [23]. They find that their demand predictions display robust behaviour within acceptable error margins. They find that artificial neural networks (ANNs) provide the best forecasting ability of the standard algorithms and can handle data outside of the training set. Johansson *et al.*, however, do not look at the long-term effects of different algorithms on their application.

Baram *et al.* combine an ensemble of learners by developing an active-learning master algorithm [24]. To achieve this, they propose a simple maximum entropy criterion that provides effective estimates in realistic settings. Their active-learning master algorithm is empirically shown to, in some cases, outperform the best algorithm on a range of classification problems.

Schmitt *et al* also extends on existing algorithms through an extension of the FLORA algorithm in [25, 26]. The FLORA algorithm generates a rule-based model, which has the ability to make binary decisions. Their FLORA-MC enhances the FLORA algorithm for multi-classification and numerical input values. They use this algorithm for an ambient computing application. Ambient computing is where computing and communication merges into everyday life. They find that their model outperforms traditional offline learners by orders of magnitude.

Similarly to us, Pindoriya *et al.* trial several different machine learning methods such as adaptive wavelet neural network (AWNN). They find that AWNN has good prediction properties when compared to other forecasting techniques such as wavelet-ARIMA, MLP and radial basis function (RBF) neural networks as well as the fuzzy neural network (FNN).

Goncalves Da Silva *et al.* show the effect of prediction accuracy on local electricity markets [27]. To this end, they compare forecasting of groups of consumers in comparison to single individuals. They trial the use of the Seasonal-Naïve and Holt-Winters algorithms and look at the effect that the errors have on trading in an intra-day electricity market of consumers and prosumers. They found that with a photovoltaic penetration of 50%, over 10% of the total generation capacity was uncapitalized and roughly 10, 25 and 28% of the total traded volume were unnecessary buys, demand imbalances and unnecessary sells respectively. This represents energy that the participant has no control. Uncapitalized generation capacity is where a participant could have produced energy, however, it was not sold on the market. Our work, however, focuses on a national electricity market, as opposed to a local market.

3. Material

3.1. Machine Learning

Machine learning is a methodology for finding and describing structural patterns in data [28]. Offline learning models are trained with the data available at a single point in time. With non-stationary data where underlying distributions change, the model must be retrained at periodic intervals, determined by how quickly the model goes out of step with the true data. With online learning, the model is able to retrain every time a new data point becomes available, without having to retrain the entire model. This makes these models good for time-series data which exhibit moderate to significant non-stationary properties.

3.2. Online learning

Examples of online learning algorithms are Passive Aggressive (PA) Regressor [29], Linear Regression, Box-Cox Regressor [30], K-Neighbors Regressor [31] and MLP regressor [32]. For our work, we trial the stated algorithms, in addition to a host of offline learning techniques. The offline techniques trialled were Lasso regression [33], ridge regression [34], Elastic Net [35], Least Angle Regression [36], Extra Trees Regressor [36], Random Forest Regressor [37], AdaBoost Regressor [38], Gradient Boosting Regressor [39] and SVR [40]. We chose the boosting and random forest techniques due to previous successes of these algorithms when applied to electricity demand forecasting [12]. We trialled the additional algorithms due to availability of these algorithms using the scikit-learn package and online learning package, Creme [41, 42].

3.3. Linear regression models

Linear regression is a linear approach to modelling the relationship between a dependent variable and one or more independent variables. Linear regressions can be used for both online and offline learning. In this work, we used them for both online and offline learning. Linear regression models are often fitted using the least squares approach. The least squares approach minimizes the sum of the squares of the residuals.

Other methods for fitting linear regressions are by minimizing a penalized version of the least squares cost function, such as in ridge and lasso regression [33, 34]. Ridge regression is a useful approach for mitigating the problem of multicollinearity in linear regression. Multicollinearity is where one predictor variable can be linearly predicted from the others with a high degree of accuracy. This phenomenon often occurs in models with a large number of parameters.

In ridge regression, the OLS loss function is augmented so that we not only minimize the sum of squared residuals but also penalized the size of parameter estimates, in order to shrink them towards zero:

$$L_{ridge}(\hat{\beta}) = \sum_{i=1}^{n}(y_i - x_i'\hat{\beta})^2 + \lambda \sum_{j=1}^{m}\hat{\beta}_j^2 = ||y - X\hat{\beta}||^2 + \lambda||\hat{\beta}||^2. \tag{1}$$

Where λ is the regularization penalty which can be chosen through cross-validation, or the value that minimizes the cross-validated sum of squared residuals, for instance. n is the number of observations of the response variable, Y, with a linear combination of m predictor variables, X, and we solve for $\hat{\beta}$, where $\hat{\beta}$ are the OLS parameter estimates.

Lasso is a linear regression technique which performs both variable selection and regularization. It is a type of regression that uses shrinkage. Shrinkage is where data values are shrunk towards a central point, such as the mean. The lasso model encourages models with fewer parameters. This enables the selection of models with fewer numbers of parameters, or automate the process of variable selection.

Under Lasso the loss is defined as:

$$L_{lasso}(\hat{\beta}) = \sum_{i=1}^{n}(y_i - x_i'\hat{\beta})^2 + \lambda \sum_{j=1}^{m} |\hat{\beta}_j|. \tag{2}$$

The only difference between lasso and ridge regression is the penalty term.

Elastic net is a regularization regression that linearly combines the penalties of the lasso and ridge methods. Specifically, Elastic Net aims to minimize the following loss function:

$$L_{enet}(\hat{\beta}) = \frac{\sum_{i=1}^{n}(y_i - x_i'\hat{\beta})^2}{2n} + \lambda(\frac{1-\alpha}{2}\sum_{j=1}^{m}\hat{\beta}_j^2 + \alpha\sum_{j=1}^{m}|\hat{\beta}_j|), \tag{3}$$

where α is the mixing parameter between ridge ($\alpha = 0$) and lasso ($\alpha = 1$). The two parameters λ and α can be tuned.

Least Angle Regression (LARS) provides a mean of producing an estimate of which variables to include in a linear regression, as well as their coefficients.

3.4. Decision tree-based algorithms

The decision tree is a model which goes from observations to output using simple decision rules inferred from data features [43]. To build a regression tree, recursive binary splitting is used on the training data. Recursive binary splitting is a greedy top-down algorithm used to minimize the residual sum of squares. The RSS, in the case of a partitioned feature space with M partitions, is given by:

$$RSS = \sum_{m=1}^{M}\sum_{i \in R_m}(y - \hat{y}_{R_m})^2. \tag{4}$$

Where y is the value to be predicted, $\hat{y}$ is the predicted value for partition R_m.

Beginning at the top of the tree, a split is made into two branches. This split is carried out multiple times and the split is chosen that minimizes the current RSS. To obtain the best sequence of subtrees cost complexity, pruning is used as a function of α. α is a tuning parameter that balances the depth of the tree and the fit to the training data. This parameter can be tuned using cross-validation.

The AdaBoost training process selects only the features of a model known to improve the predictive power of the model [38]. By doing this, the dimensionality of the model is reduced and can improve compute time. This can be used in conjunction with multiple different models. In our paper, we utilized the decision tree based algorithm with AdaBoost.

Random Forests are an ensemble learning method for classification and regression [37]. Ensemble learning methods use multiple learning algorithms to obtain better predictive performance. They work by constructing multiple decision trees at training time, and outputting the predicted value that is the mode of the predictions of the individual trees.

To ensure that the individual decision trees within a Random Forest are not correlated, bagging is used to sample from the data. Bagging is the process of randomly sampling with replacement of the training set and fitting the trees. This has the benefit of reducing the variance of the model without increasing the bias.

Random Forests differ in one way from this bagging procedure. Namely, using a modified tree learning algorithm that selects, at each candidate split in the learning process, a random subset of the features, known as feature bagging. Feature bagging is undertaken due to the fact that some predictors with a high predictive ability may be selected many times by the individual trees, leading to a highly correlated Random Forest.

ExtraTrees adds one further step of randomization [36]. ExtraTrees stands for extremely randomized trees. There are two main differences between ExtraTrees and Random Forests. Namely, each tree is trained using the whole learning sample (and not a bootstrap sample), and the top-down splitting in the tree learner is randomized. That is, instead of computing an optimal cut-point for each feature, a random cut-point is selected from a uniform distribution. The split that yields the highest score is then chosen to split the node.

3.5. Gradient Boosting

Gradient boosting is also an ensemble model [39]. Gradient boosting optimizes a cost-function over function space by iteratively choosing a function that points in the negative gradient descent direction, known as a gradient descent method.

3.6. Support vector regression

Support vector regression is an algorithm which finds a hyperplane and decision boundary to map an input do-

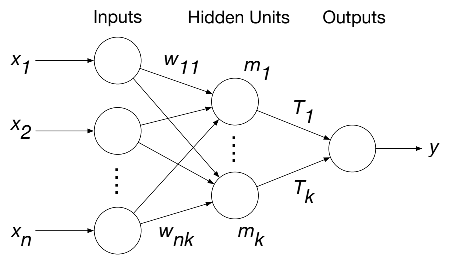


Figure 1: A three-layer feed forward neural network.

main to an output [40]. The hyperplane is chosen by minimizing the error within a certain tolerance.

Suppose we have the training set: $(x_1, y_1), \ldots, (x_i, y_i), \ldots, (x_n, y_n)$, where x_i is the input, and y_i is the output value of x_i. Support Vector Regression solves an optimization problem [44, 14], under given parameters $C > 0$ and $\varepsilon > 0$, the form of support vector regression is [45]:

$$\min_{\omega,b,\xi,\xi^*} \frac{1}{2}\omega^T\omega + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \tag{5}$$

subject to

$$\begin{aligned} y_i - (\omega^T\phi(x_i) + b) &\leq \varepsilon + \xi_i^*, \\ (\omega^T\phi(x_i) + b) - y_i &\leq \varepsilon + \xi_i, \\ \xi_i, \xi_i^* &\geq 0, i = 1, \ldots, n \end{aligned} \tag{6}$$

x_i is mapped to a higher dimensional space using the function ϕ. The ε-insensitive tube $(\omega^T\phi(x_i) + b) - y_i \leq \varepsilon$ is a range in which errors are permitted. ξ_i and ξ_i^* are slack variables which allow errors for data points which fall outside of ε. This enables the optimization to take into account the fact that data does not always fall within the ε range [46].

The constant $C > 0$ determines the trade-off between the flatness of the support vector function. ω is the model fit by the SVR. The parameters which control regression quality are the cost of error C, the width of the tube ε, and the mapping function ϕ [44, 14].

3.7. K-Neighbors Regressor

K-Neighbors regression is a non-parametric method used for regression [31]. The input consists of a new data point, and the algorithm finds the k closest training examples in the feature space. The output is the average value of the k nearest neighbours.

3.8. Multilayer Perceptron

A neural network can be used in both offline and online cases. In this work, we used them for both online and offline.

Artificial Neural Networks are a model which can model non-linear relationships between input and output data [47]. A popular neural network is a feed-forward MLP. Fig. 1 shows a three-layer feed-forward neural network with a single output unit, k hidden units, n input units. w_{ij} is the connection weight from the i^{th} input unit to the j^{th} hidden unit, and T_j is the connecting weight from the j^{th} hidden unit to the output unit [48]. These weights transform the input variables in the first layer to the output variable in the final layer using the training data.

For a univariate time series forecasting problem, suppose we have N observations $y_1, y_2, \ldots, y_N$ in the training set, and m observations in the test set, $y_{N+1}, y_{N+2}, \ldots, y_{N+m}$. In the test set and we are required to predict m periods ahead [48].

The training patterns are as follows:

$$y_{p+m} = f_W(y_p, y_{p-1}, \ldots, y_1) \tag{7}$$

$$y_{p+m+1} = f_W(y_{p+1}, y_p, \ldots, y_2) \tag{8}$$

$$\vdots$$

$$y_N = f_W(y_{N-m}, y_{N-m-1}, \ldots, y_{N-m-p+1}) \tag{9}$$

where $f_W(\cdot)$ represents the MLP network and W are the weights. For brevity we omit W. The training patterns use previous time-series points, for example, $y_p, y_{p-1}, \ldots, y_1$ as the time series is univariate. That is, we only have the time series in which we can draw inferences from. In addition, these time series points are correlated, and therefore provide information that can be used to predict the next time point.

The m testing patterns are

$$y_{N+1} = f_W(y_{N+1-m}, y_{N-m}, \ldots, y_{N-m-p+2}) \tag{10}$$

$$y_{N+2} = f_W(y_{N+2-m}, y_{N-m+1}, \ldots, y_{N-m-p+3}) \tag{11}$$

$$\vdots$$

$$y_{N+m} = f_W(y_N, y_{N-1}, \ldots, y_{N-p+1}). \tag{12}$$

The training objective is to minimize the overall predictive mean sum of squared estimate of errors (SSE) by adjusting the connection weights. For this network structure the SSE can be written as:

$$SSE = \sum_{i=p+m}^{N}(y_i - \hat{y}_i) \tag{13}$$

where $\hat{y}_i$ is the prediction from the network. The number of input nodes corresponds to the number of lagged observations. Having too few or too many input nodes can affect the predictive ability of the neural network [48].

It is also possible to vary the hyperparameter, the number of input units. Typically, various different configurations of units are trialled, with the best configuration being used in production. The weights W in f_W are trained using a process called backpropagation, which uses labelled data and gradient descent to update and optimize the weights.

3.9. *Online Algorithms*

In this Section we discuss the algorithms which were used exclusively for online learning in this work.

3.10. *Box-Cox regressor*

In this subsection, we discuss the Box-Cox regressor. Ordinary least square is a method for estimating the unknown parameters in a linear regression model. It estimates these unknown parameters by the principle of least squares. Specifically, it minimizes the sum of the squares of the differences between the observed variables and those predicted by the linear function.

The ordinary least squares regression assumes a normal distribution of residuals. However, when this is not the case, the Box-Cox Regression may be useful [30]. It transforms the dependent variable using the Box-Cox Transformation function and employs maximum likelihood estimation to determine the optimal level of the power parameter lambda. The Box-Cox Regression requires that no dependent variable has any negative values.

3.11. *Passive-Aggressive regressor*

The goal of the Passive-Aggressive (PA) algorithm is to change itself as little as possible to correct for any mistakes and low-confidence predictions it encounters [29]. Specifically, with each example PA solves the following optimisation [49]:

$$\boldsymbol{w}_{t+1} \leftarrow argmin \frac{1}{2} \|\boldsymbol{w}_t - \boldsymbol{w}\|^2 \tag{14}$$

$$s.t. \;\; y_i(\boldsymbol{w} \cdot \boldsymbol{x}_t) \geq 1. \tag{15}$$

Where x_t is the input data and y_i the output data, and w_t are the weights for the PA algorithm. Updates occur when the inner product does not exceed a fixed confidence margin - i.e., $y_i(\boldsymbol{w} \cdot \boldsymbol{x}_t) \geq 1$. The closed-form update for all examples is as follows:

$$\boldsymbol{w}_{t+1} \leftarrow \boldsymbol{w}_t + \alpha_t y_t \boldsymbol{x}_t \tag{16}$$

where

$$\alpha_t = max\left\{\frac{1 - y_t(\boldsymbol{w}_t \cdot \boldsymbol{x}_t)}{\|\boldsymbol{x}_t\|^2}, 0\right\}. \tag{17}$$

a_t is derived from a derivation process which uses the Lagrange multiplier. For full details of the derivation see [29].

3.12. *Long-term Energy Market Model*

In order to test the impact of the different residual distributions, we used the ElecSim simulation developed by Kell *et al.*, ElecSim [10, 11]. ElecSim is an agent-based model which mimics the behaviour of decentralized electricity markets. In this paper, we parametrized the model with data of the United Kingdom in 2018. This enabled us to create a digital twin of the UK electricity market and project forward. The data used for this parametrization included power plants in operation in 2018 and the funds available to the generation companies [50, 51].

ElecSim is made up of six fundamental components: 1) power plant data; 2) scenario data; 3) the time-steps of the algorithm; 4) the power exchange; 5) the investment algorithm and 6) the generation companies (GenCos) as agents. ElecSim uses a subset of representative days of electricity demand, solar irradiance and wind speed to approximate a full year. In this context, representative days are a subset of days which, when scaled up, represent an entire year [11]. We show how these components interact in Figure 2 [10]. Namely, electricity demand is matched with the supply provided by power plants through the use of a spot market. Generator companies invest in power plants based upon information provided by the data sources and expectation of the data provided by the configuration file.

ElecSim uses a configuration file which details the scenario which can be set by the user. This includes electricity demand, carbon price and fuel prices. The data sources parametrize the ElecSim simulation to make a digital twin of a particular country, including information such as wind capacity and power plants in operation. Generation Companies own and invest in power plants. These power plants are then matched to electricity demand using a spot market.

The market runs a merit-order dispatch model, and bids are made by the power plant's short-run marginal cost (SRMC). A merit-order dispatch model is one which dispatches the cheapest electricity generators first. SRMC is the cost it takes to dispatch a single MWh of electricity and does not include capital costs. Investment in power plants is based upon a net present value

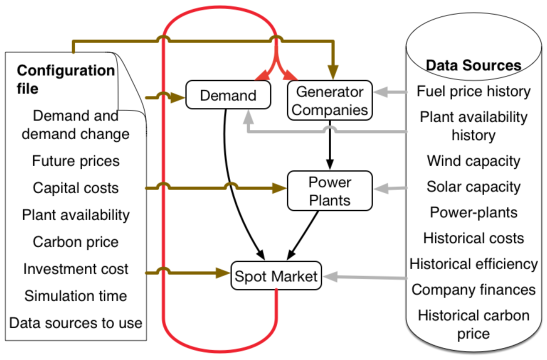


Figure 2: System overview of ElecSim [10].

(NPV) calculation. NPV is the difference between the present value of cash inflows and the present value of cash outflows over a period of time. This is shown in Equation 18, where t is the year of the cash flow, i is the discount rate, N is the total number of years, or lifetime of power plant, and R_t is the net cash flow of the year t:

$$NPV(t, N) = \sum_{t=0}^{N} \frac{R_t}{(1+t)^t}. \tag{18}$$

Each of the Generator Companies (GenCos) estimate the yearly income for each prospective power plant by running a merit-order dispatch electricity market simulation ten years into the future. However, it is true that the expected cost of electricity ten years into the future is particularly challenging to predict. We, therefore, use a reference scenario projected by the UK Government Department for Business and Industrial Strategy (BEIS), and use the predicted costs of electricity calibrated by Kell *et al* [11, 52]. The agents predict the future carbon price by using a linear regression model.

4. Methods

4.1. Data preparation

Similarly to our previous work in [3], we selected a number of calendar attributes and demand data from the GB National Grid Status dataset provided by the electricity market settlement company Elexon, and the University of Sheffield [53]. This dataset contained data between the years 2011-2018 for the United Kingdom. The calendar attributes used as predictors to the models were hour, month, day of the week, day of the month and year. These attributes allow us to account for the periodicity of the data within each day, month and year.

It is also the case that electricity demand on a public holiday which falls on a weekday is dissimilar to load behaviours of ordinary weekdays [15]. We, therefore, marked each holiday day to allow the model to account for this.

As demand data is highly correlated with historical demand, we lagged the input demand data. In this context, the lagged data is where we provide data of previous time steps at the input. For example, for predicting $t + 1$, we use n inputs: $t, t-1, t-2, \ldots, t-n$. This enabled us to take into account correlations on previous days, weeks and the previous month. Specifically, we used the previous 28 hours before the time step to be predicted for the previous 1st, 2nd, 7th and 30th day. We chose this as we believe that the previous two days were the most relevant to the day to be predicted, as well as the weekday of the previous week and the previous month. We chose the previous 28 hours to account for a full day, plus an additional 4 hours to account for the previous day's correlation with the day to be predicted. We could have increased the number of days provided to the algorithm. However, due to time and computational constraints, we used our previously described intuition for lagged data selection. The number of lagged inputs to trial increases exponentially with each additional day added, therefore making the problem intractable when also trialling such a high number of algorithms and hyperparameters.

In addition to this, we marked each of the days with their respective seven seasons. These seasons were defined by the National Grid Short Term Operating Reserve (STOR) Market Information Report [54]. These differ from the traditional four seasons by splitting autumn into two further seasons, and winter into three seasons. Finally, to predict a full 24-hours ahead, we used 24 different models, 1 for each hour of the day.

The data is standardized and normalized using min-max scaling between -1 and 1 before training and predicting with the model. This is due to the fact that the inputs such as day of the week, hour of day are significantly smaller than that of demand. Therefore, the demand will influence the result more due to its larger value. However, this does not necessarily mean that demand has greater predictive power.

4.2. Algorithm Tuning

To find the optimum hyperparameters, cross-validation is used. As this time-series data were correlated in the time-domain, we took the first six years of data (2011-2017) for training and tested on the remaining year of data (2017-2018).

Each machine learning algorithm has a different set of parameters to tune. To tune the parameters in this paper, we used a grid search method. Grid search is a brute force approach that trials each combination of parameters at our choosing; however, for our search space was small enough to make other approaches not worth the additional effort.

Tables 1 and 2 display each of the models and respective parameters that were used in the grid search. Table 1 shows the offline machine learning methods, whereas Table 2 displays the online machine learning methods. Each of the parameters within the columns “Values” are trialled with every other parameter.

Whilst there is room to increase the total number of parameters, due to the exponential nature of grid-search, we chose a smaller subset of hyperparameters, and a larger number of regressor types. Specifically, with neural networks, there is a possibility to extend the number of layers as well as the number of neurons, to use a technique called deep learning. Deep learning is a class of neural networks that use multiple layers to extract higher levels of features from the input. For this paper, however, we decided to trial a large number of different models, instead of a large number of different configurations for neural networks.

4.3. *Prediction Residuals in ElecSim*

Each of the previously mentioned models trialled will have a certain degree of errors. Prediction residuals are the difference between the estimated and actual values. We collect the prediction residuals to form a distribution for each of the models. We then trial 80 different closed-form distributions to see which of the distributions best fits the residuals from each of the models. These 80 distributions were chosen due to their implementation in scikit-learn [41].

Once each of the prediction residual distributions are fit with a sensible closed-form distribution, we sample from this new distribution and perturb the demand for the electricity market at each time step within ElecSim.

By perturbing the market by the residuals, we can observe what the effects are of incorrect predictions of demand in an electricity market using the long-term electricity market model, ElecSim. We are able to understand the differences that prediction residuals have on long-term investment decisions as well as generators utilized.

5. Results

In this Section, we detail the accuracy of the algorithms and statistical models to predict 24 hours ahead for the day-ahead market. In addition to this, we display the impact of the errors on electricity generation investment and electricity mix from the years 2018 to 2035 using the agent-based model ElecSim.

5.1. *Offline Machine Learning*

To generate these results, we use a training set to train the data, and a test set to see how well each algorithm performs on the testing data. That is, how well the algorithm can predict data it is yet to see. In our case, the training data was from 2011 to 2017, and the testing data was from 2017 to 2018.

Figure 3 displays the mean absolute error of each of the offline statistical and machine learning models on a log scale. It can be seen that the different models have varying degrees of success. The least accurate models were linear regression, the MLP (MLP) model and the Least Angle Regression (LARS). These all have mean absolute errors over 10,000MWh. This error would be prohibitively high in practice; the max tendered national grid reserve is 6,000MWh, while the average tendered national grid reserve is 2,000MWh [54].

A number of models perform well, with a low mean absolute error. These include the Lasso, gradient Boosting Regressor and K-neighbours regressor. The best model, similar to [3], was the decision tree-based model, Extra Trees Regressor, with a mean absolute error of $1,604$MWh. This level is well within the average national grid reserve of 2,000MWh.

Table **??** displays different metrics for measuring the accuracy of the offline machine learning techniques. These include Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) and the Mean R-Squared. The results largely compliment each other. With high error values in one metric correlating to high error metrics in others for each of the estimators.

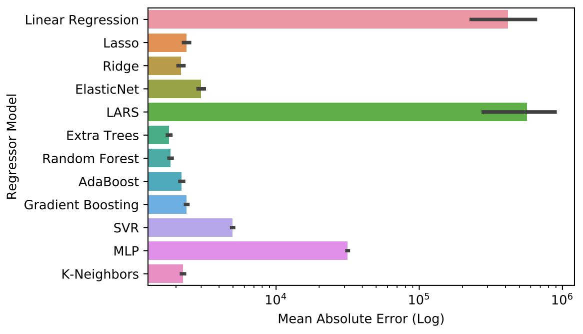


Figure 3: Offline models mean absolute error comparison, with 95% confidence interval for 5 runs of each model.

Regressor Type	**Parameters**	**Values**	**Parameters**	**Values**	**Parameters**	**Values**
Linear	N/A	N/A				
Lasso	N/A	N/A				
Elastic Net	N/A	N/A				
Least-Angle	N/A	N/A				
Extra Trees	# Estimators	[16, 32]				
Random Forest	# Estimators	[16, 32]				
AdaBoost	# Estimators	[16, 32]				
Gradient Boosting	# Estimators	[16, 32]	learning rate	[0.8, 1.0]		
Support Vector	Kernel	[linear, rbf]	C	[1, 10]	Gamma	[0.001, 0.0001]
MLP	Activation function	[tanh, relu]	hidden layer sizes	[1, 50]	Alpha	[0.00005, 0.0005]
K-Neighbours	# Neighbours	[5, 20, 50]				

Table 1: Hyperparameters for offline machine learning regression algorithms

Regressor Type	**Parameters**	**Values**	**Parameters**	**Values**	**Parameters**	**Values**
Linear	N/A	N/A				
Box-Cox	Power	[0.1, 0.05, 0.01]				
MLP	Hidden layer sizes	[(10, 50, 100), (10), (20), (50), (10, 50)]				
Passive Aggressive	C	[0.1, 1, 2]	Fit intercept?	[True, False]	Max iterations	[1, 10, 100, 1000]

Table 2: Hyperparameters for online machine learning regression algorithms

Estimator	Mean Fit Time	Mean Score Time	Mean MSE	Mean RMSE	Mean MAE	Mean R-Squared
LinearRegression	57	4.84	9444808.95	3073.24	2249.34	0.81
Lasso	787.97	3.13	9446957.22	3073.59	2249.55	0.81
Ridge	26.99	5.03	9444701.58	3073.22	2252.23	0.81
ElasticNet	54.49	6.13	31139231.96	5580.25	4628.64	0.36
llars	46.85	8	10164815.45	3188.23	2333.31	0.79
ExtraTreesRegressor	9321.52	58.06	5562579.53	2358.51	1605	0.89
RandomForestRegressor	16567.46	13.99	5882618.89	2425.41	1646.29	0.88
AdaBoostRegressor	8897.55	26.9	18551963.36	4307.2	3544.49	0.62
GradientBoostingRegressor	6417.62	8.16	6744402.87	2597	1833.62	0.86
SVR	19170.82	5221.66	[illegible]	[illegible]	[illegible]	-0.05
KNeighborsRegressor	118.89	15215.87	[illegible]	[illegible]	[illegible]	0.79

Figure 4 displays the distribution of the best offline machine result (Extra Trees Regressor). It can be seen that the max tendered national grid reserve falls well above the 5% and 95% percentiles. However, there are occasions where the errors are greater than the maximum tendered national grid reserve. In addition, the majority of the time, the model's predictions fall within the average available tendered national grid reserve.

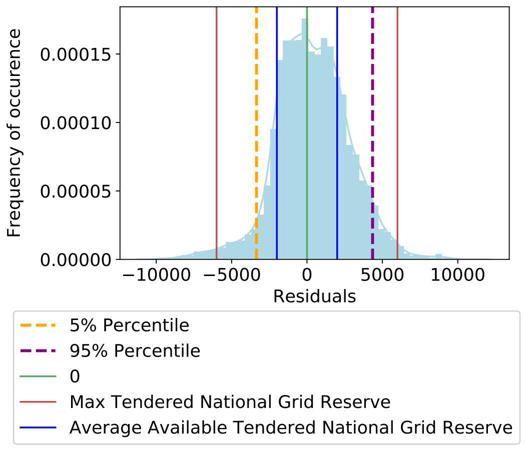


Figure 4: Best offline machine learning algorithm (Extra Trees Regressor) distribution.

Figures 5 and 6 display the time taken to train the model and time taken to sample from the model versus the absolute error respectively for the offline algorithms. Multiple fits are trialled for each parameter type for each model. The error bars indicate the results of multiple cross-validations.

It can be seen from Figure 5 that the time to fit varies significantly between algorithms and parameter choices. The MLP consistently takes a long time to fit, when compared to the other algorithms and performs relatively poorly in terms of MAE. There are many models, such as the random forest regressor, and extra trees regressors which perform well, however, take a long time to fit, especially when compared to the K-Nearest neighbours.

For a small deterioration in MAE it is possible to decrease the time it takes to train the model significantly. For example, by using the K-Nearest neighbours or support vector regression (SVR).

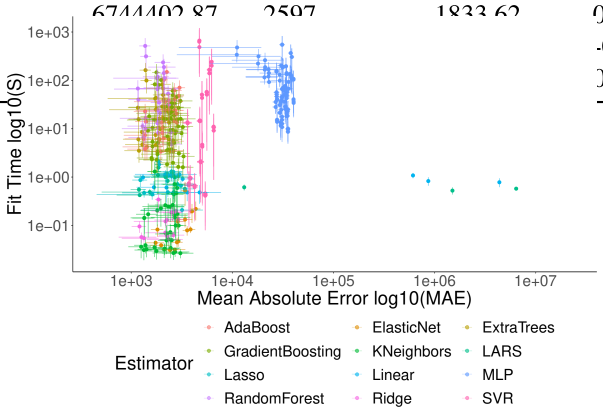


Figure 5: Time taken to train the offline models versus mean absolute error. Error bars display standard deviation between points.

The scoring time, displayed in Figure 6, also displays a large variation between model types. For instance, the MLP regressor takes a shorter time to sample predictions when compared to the K-Neighbors algorithm and support vector regression. It is possible to have a cluster of algorithms with low sample times and low mean absolute errors. However, often a trade-off is required, with a fast prediction time requiring a longer training time and vice-versa.

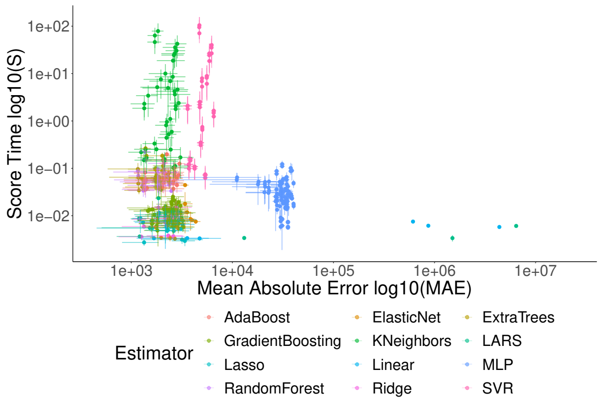


Figure 6: Time taken to score the offline models versus mean absolute error. Error bars display standard deviation between points.

5.2. *Online Machine Learning*

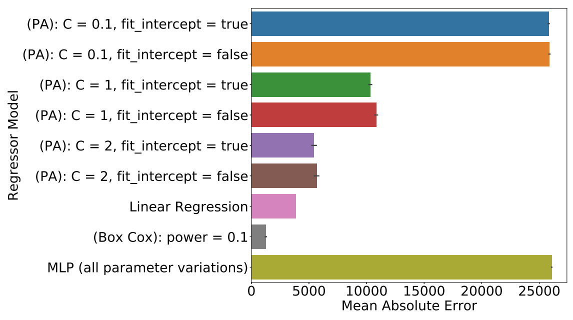


Figure 7: Comparison of mean absolute errors (MAE) for different online regressor models. MLP results for all parameters are shown in a single barchart due to the very similar MAEs for the differing hyperparameters.

To see if we can improve on the predictions, we utilize an online machine learning approach. If we are successful, we should be able to reduce the national grid reserves, reducing cost and emissions.

Figure 7 displays the comparison of mean absolute errors for the different trialled online regressor models. To produce this graph, we showed various hyperparameter trials. Where the hyperparameters had the same results, we removed them. For the MLP, we aggregated all hyperparameters, due to the similar nature of the predictions.

It can be seen that the best performing model was the Box-Cox regressor, with an MAE of 1100. This is an improvement of over 30% on the best offline model. The other models perform less well. However, it can be seen that the linear regression model improves significantly for the online case when compared to the offline case. The passive aggressive (PA) model improve significantly with the varying parameters, and the MLP performs poorly in all cases.

Figure 8 displays the best online model. We can see a significant improvement over the best online model distribution, shown in Figure 4. We remain within the max tendered national grid reserve for 98.9% of the time, and the average available tendered national grid reserve is close to the 5% and 95% percentiles.

Figure 9 displays the residuals for a model with poor predictive ability, the passive aggressive regressor. It displays a large period of time of prediction errors at -20,000MWh, and often falls outside of the national grid reserve. These results demonstrate the importance of trying a multitude of different models and parameters to improve prediction accuracy.

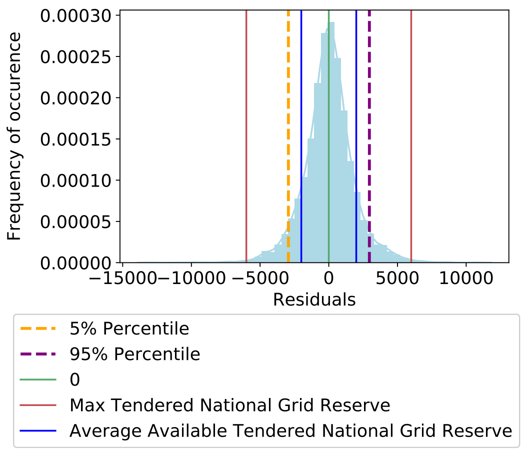


Figure 8: Best online model (Box-Cox Regressor) distribution.

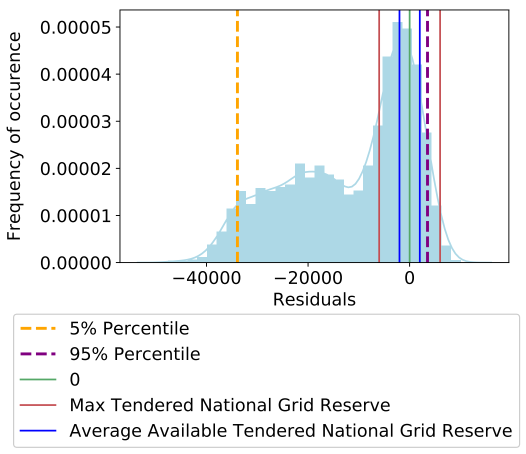


Figure 9: Online machine learning algorithm distribution. (Passive Aggressive Regressor (C=0.1, fit intercept = true, maximum iterations = 1000, shuffle = false, tolerance = 0.001), chosen as it was the worst result for the passive aggressive model.

Figure 10 displays a comparison between the actual electricity consumption compared to the predictions. It can be seen that the Box-Cox model better predicts the actual electricity demand in most cases when compared to the best offline model, the Extra Trees regressor. The Extra Trees regressor often overestimates the demand, particularly during weekdays. Whilst the Box-Cox regressor more closely matches the actual results. During the weekend (between the hours of 120 and 168), the Extra Trees regressor performs better, particularly on the Saturday (between hours of 144 and 168).

Figures 11 and 12 display the mean absolute error versus test and training time respectively. In these graphs, a selection of models and parameter combina-

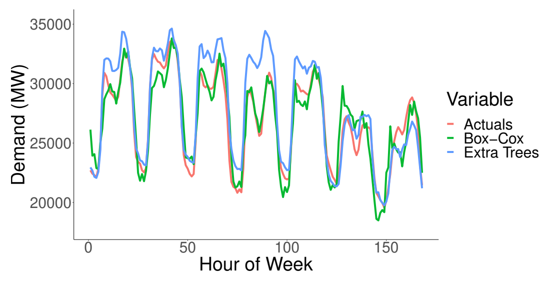


Figure 10: Best offline model compared to the best online model over a one week period.

tions are chosen.

Clear clusters can be seen between different types of models and parameter types. With the passive aggressive (PA) model performing the slowest for both training and testing. Different parameter combinations show different results in terms of mean absolute error.

The best performing model is the Box-Cox model, which is also the fastest to both train and test. The linear regression, which performs worse in terms of predictive performance, is as quick to train and test as the Box-Cox model. Additionally, the MLP is relatively quick to train and test when compared to the PA models.

It is noted that when compared to the offline models, the training time is a good indicator to the testing time. In other words, models that are fast to train are also fast to test and vice-versa.

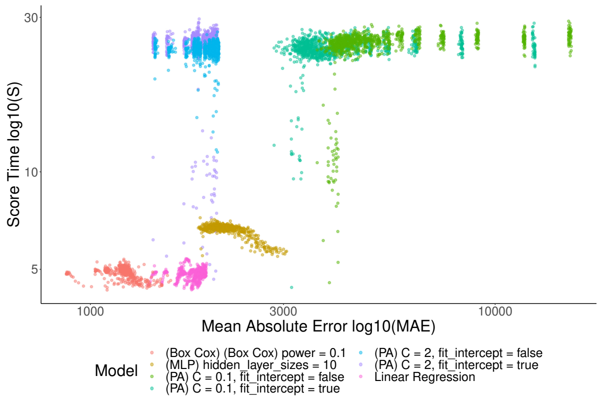


Figure 11: Time taken to test the online models versus mean absolute error.

5.3. *Scenario Comparison*

In this Section we explore the effect of these residuals on investments made and the electricity generation mix. To generate these graphs, we perturbed the exogenous demand in ElecSim by sampling from the best-fitting

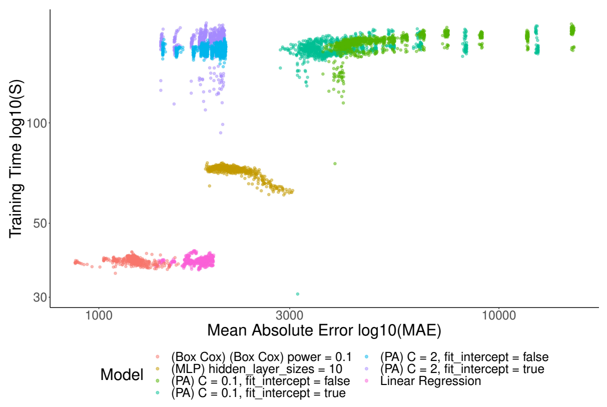


Figure 12: Time taken to train the online models versus mean absolute error.

distributions for the respective residuals of each of the online methods. We did this for all of the online learning algorithms displayed in Figure 7. We let the simulation run for 17 years from 2018 to 2035.

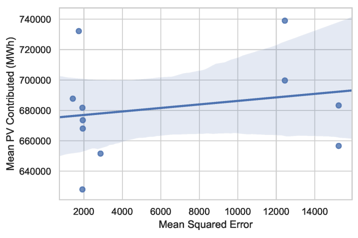
Running this simulation enabled us to see the effect on carbon emissions on the electricity grid over a long time period. For instance, does underestimating electricity demand mean that peaker power plants, such as reciprocal gas engines, are over utilized when other, less polluting power plants could be used?

5.3.1. *Mean Contributed Energy Generation*

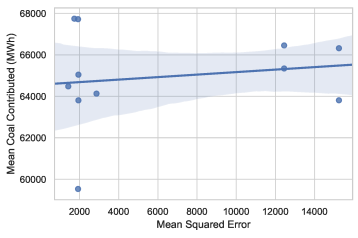
In this Section we display the mean electricity mix contributed by different electricity sources over the years 2018 to 2035.

Figure 13a displays the mean photovoltaic (PV) contributed between 2018 and 2035 vs. mean absolute error of the various online regressor models displayed in Figure 7. A positive correlation can be seen with PV contributed and mean absolute error. This is similar for coal and nuclear output, shown in Figures 13b and 13c respectively. However, as shown by Figure 13d, offshore wind reduces with mean absolute error. Figure 13e displays the mean reciprocal gas engine output vs mean absolute error between the same time period. Output for the reciprocal gas engine also increases with mean absolute error.

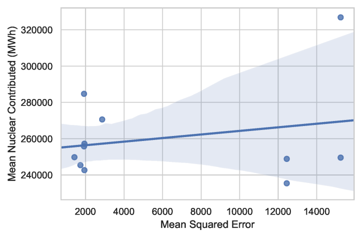
The reciprocal gas engine was expected to increase with times of high error. This is because, traditionally, reciprocal gas engines are peaker power plants. Peaker power plants provide power at times of peak demand, which cannot be covered by other plants due to them being at their maximum capacity level or out of service. It may also be the case, that with higher proportions of intermittent technologies, there is a larger need for these



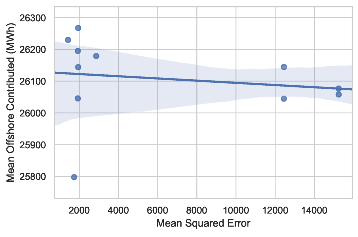
(a) Photovoltaic output.



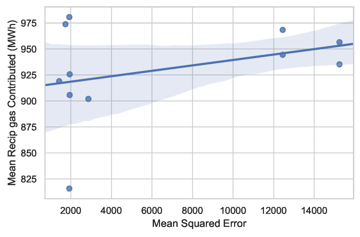
(b) Coal output.



(c) Nuclear output.



(d) Offshore output.



(e) Reciprocal gas engine output.

Figure 13: Mean outputs of various technologies vs. mean absolute error from 2018 to 2035 in ElecSim.

peaker power plants to fill in for times where there is a deficit in wind speed and solar irradiance.

It is hypothesized that coal and nuclear output increase to cover the predicted increased demands of the service. As these generation types are dispatchable, meaning operators can choose when they generate electricity, they are more likely to be used in times of higher predicted demand.

Photovoltaics may be used more with higher errors due to the times at which the errors were greatest. For example, during the day, where demand is higher, as is solar irradiance.

5.3.2. Total Energy Generation

In this Section, we detail the difference in total technologies invested in over the time period between 2018 to 2035, as predicted by ElecSim.

CCGT, onshore, and reciprocal gas engines are invested in less over the time period, as shown by Figures 14a, 14d, 15a respectively. While coal, offshore, nuclear and photovoltaics all exhibit increasing investments.

It is hypothesized that coal and nuclear increase in investment due to their dispatchable nature. While onshore, non-dispatchable by nature, become a less attractive investment when compared to the other technologies.

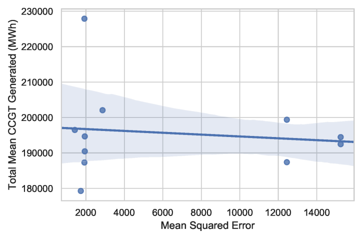
CCGT and reciprocal gas engines may have decreased in capacity over this time, due to the increase in coal. This could be because of the large consistent errors in prediction accuracy that meant that reciprocal gas engines were perceived to be less valuable.

Figure 15b shows an increase in relative mean carbon emitted with mean absolute error of the predictions residuals. The reason for an increase in relative carbon emitted could be due to the increased output of utility of the reciprocal gas engine, coal, and decrease in offshore output. Reciprocal gas engines are peaker plants and, along with coal, can be dispatched. By being dispatched, the errors in predictions of demand can be filled. It is therefore recommended that by improving the demand prediction algorithms, significant gains can be made in reducing carbon emissions.

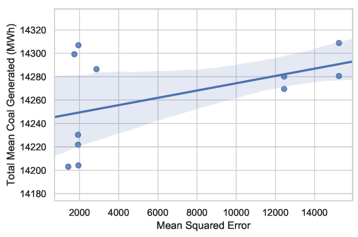
6. Discussion

From our results, it can be seen that different algorithms yield differing prediction accuracies. Online models can result in a decrease in 30% of prediction error on the best offline models. We calculated this by comparing the MAE for Extra Trees to the MAE for the Box-Cox regressor. We, therefore, recommend the use of online machine learning for predicting electricity demand in a day-ahead market.

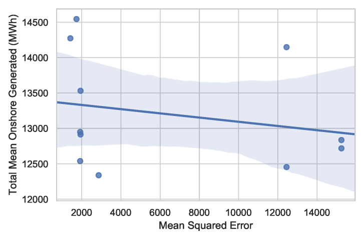
Similar to our assumptions, the online learning algorithms were able to outperform the offline models. This is due to the non-stationary nature of the data. An online method is able to use the most up-to-date knowledge of the complex system of energy demand. For instance, a certain year may have a particularly warm



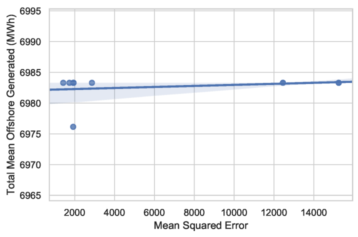
(a) Total CCGT.



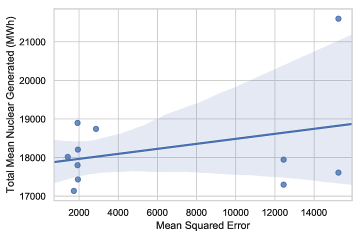
(b) Total Coal.



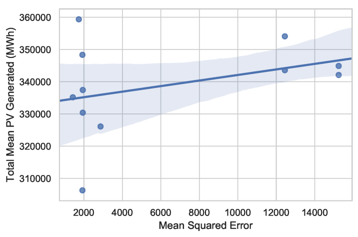
(c) Total Onshore.



(d) Total Offshore.

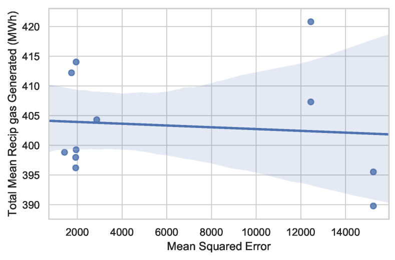


(e) Total nuclear.

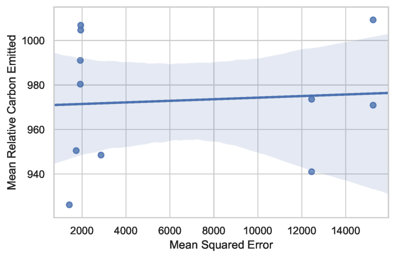


(f) Total photovoltaics.

Figure 14: Total technologies invested in vs. mean absolute error from 2018 to 2035 in ElecSim.



(a) Total Reciprocal gas engine.



(b) Mean carbon emitted.

Figure 15: a) Investments in reciprocal gas engine technologies vs. mean absolute error from 2018 to 2035 in ElecSim and d) mean carbon emissions between 2018 and 2035.

winter when compared to previous years, reducing the amount of electricity used for heating.

However, contrary to our assumptions, the online linear regression techniques outperformed the online machine learning techniques. This may be due to their simpler nature and ability to learn from a smaller subset of new data as opposed to relying on a large historic subset. For the offline models, the best performing algorithms were the decision tree approaches such as extra trees and random forests. This is a similar outcome to our previous work, which showed that the best performing method for demand forecasting were random forests [12]. Contrary to our assumptions, however, the lasso and ridge regression outperformed the machine learning techniques support vector regression and MLP. This may be due to the ability of feature selection by lasso and ridge regression, which only uses the most important features.

To the best of our knowledge, more work has been done using offline learning to predict electricity demand. This may be due to the additional complexity of running online algorithms, and a smaller number of available models to run in an online fashion.

In terms of computing power, finding the optimal input parameters, hyperparameters and models to use can be a large undertaking. This is due to the exponential growth of the number of choices that can be made. This can be an issue where accuracy is of importance, especially when the data changes over time, meaning it may be necessary to retest previous results. However, due to the financial and sustainability implications, we believe the trade-off between compute time and accuracy is balanced towards compute time. There are also large implications if the model were to break at a certain point in time. We, therefore, recommend the reliance on multiple well-performing models, as opposed to solely the best performing model at any one time.

For training time and prediction time, there is often a trade-off between training and predicting. For instance, the k-nearest neighbours is fast to train, but slow to sample from. Therefore stakeholders must make a decision based upon accuracy, speed of training and sampling.

Additionally, the impact on the broader electricity market has been shown to be significant. Principally, the investment behaviours of generation companies change as well as the dispatched electricity mix. The relative mean carbon emitted over this time period increases, due to an increase in the utilization of coal and reciprocal gas engines, at the expense of offshore wind.

7. Conclusion

In this paper, we evaluated 16 different machine learning and statistical models to predict electricity demand in the UK for the day-ahead market. Specifically, we used both online and offline algorithms to predict electricity demand 24 hours ahead. We compared the ability for the offline models: lasso regression, random forests, support vector regression, for both online and offline learning: linear regression, MLP and for just online learning: the Box-Cox transformation and the passive aggressive regressor, amongst others. The Box-Cox, as well as the passive aggressive regressors, were used as online learning algorithms, the MLP and linear regression were used as both, whereas the rest were used as offline learning algorithms.

We measured the errors and compared these to each model as well as the national grid reserve. We found that through the use of an online learning approach, we were able to significantly reduce error by 30% on the best offline algorithm. We were also able to reduce our errors to significantly below the national grid's mean and maximum tendered reserve, thus significantly reducing the chances of blackouts.

In addition to this, we took these errors and perturbed the electricity market of the agent-based model ElecSim. This enabled us to see the impact of different error distributions on the long-term electricity market, both in terms of investment and in terms of the electricity mix.

We observed that with an increase in prediction error, we get a higher proportion of electricity generated by coal, offshore, nuclear, reciprocal gas engines and photovoltaics. This could be due to the fact that more peaker plants are required to fill in for unexpected demand. In addition, a higher proportion of intermittent renewable energy sources leads to a higher use of peaker power plants to fill in the gaps of intermittency of wind and solar irradiance. However, by reducing the mean absolute error, we are able to significantly reduce the amount of reciprocal gas engines and coal usage.

In future work, we would like to trial a different selection of algorithms and statistical models and trial different inputs to the models, for instance, by providing the model with two months worth of historical data as dependent variables.

8. Funding Sources

This work was supported by the Engineering and Physical Sciences Research Council, Centre for Doctoral Training in Cloud Computing for Big Data [grant number EP/L015358/1].

References

[1] C. Lu, H. T. Wu, S. Vemuri, Neural network based short term load forecasting, IEEE Transactions on Power Systems 8 (1) (1993) 336–342. doi:10.1080/02533839.1995.9677697.

[2] A. Mahmood, M. N. Ullah, S. Razzaq, A. Basit, U. Mustafa, M. Naeem, N. Javaid, A new scheme for demand side management in future smart grid networks, Procedia Computer Science 32 (2014) 477–484.

[3] A. Kell, A. McGough, M. Forshaw, Segmenting residential smart meter data for short-Term load forecasting, in: e-Energy 2018 - Proceedings of the 9th ACM International Conference on Future Energy Systems, 2018.

[4] T. Hong, J. Wilson, J. Xie, A. Member, Long Term Probabilistic Load Forecasting and Normalization With Hourly Information 5 (1) (2014) 456–462.

[5] M. Al-Musaylh, R. Deo, J. Adamowski, Y. Li, Short-term electricity demand forecasting with MARS, SVR and ARIMA models using aggregated demand data in Queensland, Australia, Advanced Engineering Informatics 35 (November 2017) (2018) 1–16.

[6] P. Vrablecová, A. Bou Ezzeddine, V. Rozinajová, S. Šárik, A. K. Sangaiah, Smart grid load forecasting using online support vector regression, Computers & Electrical Engineering 0 (2017) 1–16. doi:10.1016/j.compeleceng.2017.07.006.

[7] A. K. Singh, Ibraheem, S. Khatoon, M. Muazzam, D. K. Chaturvedi, Load forecasting techniques and methodologies: A review, ICPCES 2012 - 2012 2nd International Conference on Power, Control and Embedded Systems (2012).

[8] S.-j. Huang, S. Member, K.-r. Shih, Short-Term Load Forecasting Via ARMA Model Identification Including Non-Gaussian 18 (2) (2003) 673–679.

[9] F. M. Andersen, H. V. Larsen, T. K. Boomsma, Long-term forecasting of hourly electricity load: Identification of consumption profiles and segmentation of customers, Energy Conversion and Management 68 (2013) 244–252.

[10] A. Kell, M. Forshaw, A. S. Mcgough, ElecSim : Monte-Carlo Open-Source Agent-Based Model to Inform Policy for Long-Term Electricity Planning, The Tenth ACM International Conference on Future Energy Systems (ACM e-Energy) (2019) 556–565.

[11] A. J. M. Kell, M. Forshaw, A. S. McGough, Long-Term Electricity Market Agent Based Model Validation using Genetic Algorithm based Optimization, The Eleventh ACM International Conference on Future Energy Systems (e-Energy'20) (2020).

[12] A. Kell, A. S. Mcgough, M. Forshaw, Segmenting Residential Smart Meter Data for Short-Term Load Forecasting, e-Energy Conference (2018) 91–96.

[13] M. W. Ahmad, M. Mourshed, Y. Rezgui, Trees vs Neurons: Comparison between random forest and ANN for high-resolution prediction of building energy consumption, Energy and Buildings 147 (2017) 77–89.

[14] B.-j. Chen, M.-w. Chang, C.-j. Lin, Load Forecasting Using Support Vector Machines : A Study on EUNITE Competition 2001, IEEE Transactions on Power Systems 19 (4) (2004) 1821–1830.

[15] K.-h. Kim, H.-s. Youn, S. Member, Y.-c. Kang, Short-term load forecasting for special days in anomalous load conditions using neural networks, IEEE Transactions on Power Systems 15 (2) (2000) 559–565. doi:10.1109/59.867141.

[16] J. Nagi, S. K. Yap, S. K. Tiong, S. K. Ahmed, Electrical Power Load Forecasting using Hybrid Self-Organizing Maps and Support Vector Machines, The 2nd International Power Engineering optimization Conference (PEOCO) (June) (2008) 51 – 56.

[17] F. L. Quilumba, W.-j. Lee, H. Huang, D. Y. Wang, S. Member, R. L. Szabados, Using Smart Meter Data to Improve the Accuracy of Intraday Load Forecasting Considering Customer Behavior Similarities (2014) 1–8.

[18] H. Nguyen, C. K. Hansen, Short-term electricity load forecasting with Time Series Analysis, 2017 IEEE International Conference on Prognostics and Health Management (ICPHM) (2017) 214–221doi:10.1109/ICPHM.2017.7998331.
URL `http://ieeexplore.ieee.org/document/7998331/`

[19] G. Gross, F. Galiana, Short-term load forecasting, Proceedings of the IEEE 75 (12) (1987) 1558–1573. doi:10.1109/PROC.1987.13927.

[20] M. Ghofrani, M. Hassanzadeh, M. S. Fadali, Smart Meter Based Short-Term Load Forecasting for Residential Customers 13–17.

[21] A. K. Fard, M.-R. Akbari-Zadeh, A hybrid method based on wavelet, ANN and ARIMA model for short-term load forecasting, Journal of Experimental & Theoretical Artificial Intelligence 26 (2) (2014) 167–182. doi:10.1080/0952813X.2013.813976.

[22] S. Humeau, T. K. Wijaya, M. Vasirani, K. Aberer, Electricity load forecasting for residential customers: Exploiting aggregation and correlation between households, 2013 Sustainable Internet and ICT for Sustainability, SustainIT 2013 (2013). doi:10.1109/SustainIT.2013.6685208.

[23] C. Johansson, M. Bergkvist, D. Geysen, O. D. Somer, N. Lavesson, D. Vanhoudt, Operational Demand Forecasting in District Heating Systems Using Ensembles of Online Machine Learning Algorithms, Energy Procedia 116 (2017) 208–216.

[24] Y. Baram, R. El-Yaniv, K. Luz, Online Choice of Active Learning Algorithms, Proceedings, Twentieth International Conference on Machine Learning 1 (2003) 19–26.

[25] J. Schmitt, M. Hollick, C. Roos, R. Steinmetz, Adapting the user context in realtime: Tailoring online machine learning algorithms to ambient computing, Mobile Networks and Applications 13 (6) (2008) 583–598. doi:10.1007/s11036-008-0095-8.

[26] G. Widmer, Learning in the presence of concept drift and hidden contexts, Machine Learning 23 (1) (1996) 69–101. doi:10.1007/bf00116900.

[27] P. Goncalves Da Silva, D. Ilic, S. Karnouskos, The Impact of Smart Grid Prosumer Grouping on Forecasting Accuracy and Its Benefits for Local Electricity Market Trading, IEEE Transactions on Smart Grid 5 (1) (2014) 402–410. doi:10.1109/TSG.2013.2278868.

[28] I. H. Witten, E. Frank, M. a. Hall, Data Mining: Practical Machine Learning Tools and Techniques, 2011.

[29] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, Y. Singer, Online Passive-Aggressive Algorithms, Journal of Machine Learning Research (2006). doi:10.1201/b15810-63.

[30] G. E. P. Box, D. Cox, An Analysis of Transformations, Journal, Source Statistical, Royal Series, Society 26 (2) (1964) 211–252.

[31] E. Forgy, Cluster analysis of multivariate data: Efficiency versus interpretability of classification, Biometrics 21 (3) (1965) 768–769.

[32] G. E. Hinton, Connectionist learning procedures, Artificial Intelligence 40 (1-3) (1989) 185–234. doi:10.1016/0004-3702(89)90049-0.

[33] R. Tibshirani, Regression Shrinkage and Selection Via the Lasso, Journal of the Royal Statistical Society: Series B (Methodological) 58 (1) (1996) 267–288. doi:10.1111/j.2517-6161.1996.tb02080.x.

[34] P. Geladi, Measurement, regression and calibration, philip brown, oxford statistical science series. vol. 12, oxford science publications, oxford, 1993, no of pages: 224. price: £27.50. isbn 0 19-852245-2, Journal of chemometrics 8 (5) (1994) 371–372.

[35] J. Friedman, T. Hastie, R. Tibshirani, Regularization Paths for Generalized Linear Models via Coordinate Descent, Journal of Statistical Software 33 (1) (2010) 1–22. arXiv:0908.3817, doi:10.1016/j.expneurol.2008.01.011.

[36] B. Efron, T. Hastie, I. Johnstone, R. Tibshirani, Least angle regression, The Annals of Statistics 32 (2) (1988) 440–444. doi:10.1109/glocom.1988.25879.

[37] L. Breiman, Random forests, Machine Learning 45 (1) (2001) 5–32. arXiv:/dx.doi.org/10.1023%2FA%3A1010933404324, doi:10.1023/A:1010933404324.

[38] Y. Freund, R. E. Schapire, A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting, Journal of Computer and System Sciences 55 (1) (1997) 119–139. doi:10.1006/jcss.1997.1504.

[39] J. H. Friedman, Greedy Function Approximation: A Gradient Boosting Machine (316) 400.

[40] C. Cortes, V. Vapnik, Support-Vector Networks, Machine Learning 20 (3) (1995) 273–297. arXiv:arXiv:1011.1669v3, doi:10.1023/A:1022627411411.

[41] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in Python, Journal of Machine Learning Research 12 (2011) 2825–2830.

[42] Creme, https://pypi.org/project/creme/ (2019).

[43] J. R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[44] F. Shu, C. Luonan, Short-term load forecasting based on an adaptive hybrid method, Power Systems, IEEE Transactions on 21 (1) (2006) 392–401. doi:10.1109/TPWRS.2005.860944.

[45] H. Drucker, C. J. C. Burges, L. Kaufman, A. Smola, V. Vapnik, Support vector regression machines, Advances in Neural Information Processing Systems 1 (1997) 155–161. doi:10.1.1.10.4845.

[46] A. J. Smola, B. Schölkopf, A tutorial on support vector regression, Statistics and Computing 14 (3) (2004) 199–222. arXiv:arXiv:1011.1669v3, doi:10.1023/B:STCO.0000035301.49549.88.

[47] H. Akaike, A New Look at the Statistical Model Identification, IEEE Transactions on Automatic Control 19 (6) (1974) 716–723. arXiv:arXiv:1011.1669v3, doi:10.1109/TAC.1974.1100705.

[48] H.-T. Pao, Forecasting electricity market pricing using artificial

neural networks, Energy Conversion and Management 48 (3) (2007) 907–912. doi:10.1016/j.enconman.2006.08.016.

[49] J. Ma, L. K. Saul, S. Savage, G. M. Voelker, Identifying suspicious URLs: An application of large-scale online learning, Proceedings of the 26th International Conference On Machine Learning, ICML 2009 (2009) 681–688.

[50] Department for Business and Industrial Strategy, UK Government, Power stations in the united kingdom, may 2019, Digest of United Kingdom Energy Statistics (DUKES) (2019).

[51] Department for Business, Energy & Industrial Strategy, Companies house - gov.uk, UK Government (2019).

[52] Department for Business Energy & Industrial Strategy, Updated energy and emissions projections 2018, The Energy White Paper (April) (2019).

[53] Elexon portal and Sheffield University, G.b. national grid status, https://www.gridwatch.templar.co.uk/.

[54] National Grid, STOR Market Information Report (October) (2019) 0–11.