University of Technology Sydney

Robert Kell

Donald Turton

Advanced Data Analytics Algorithms AT2

Practical Workplace-Related Data Analytics Project

Introduction

The topic of investigation for this Practical Machine Learning Project concerns Australian electricity prices per megawatt. It is the goal of this project to discover those variables that are the most significant predictors of the price of electricity, build two machine learning models sophisticated enough to predict electricity price 30min into the future, and select the most accurate model based on the root mean square error measurement metric. The two machine learning models selected for the project include a Random Forest Regressor and an XGBoost Regressor, given that this is not a classification task. There have been three different model iterations produced for each of the two regressors selected (six models in total); the first model has no hyperparameter tuning (only the amount of trees has been defined), the second employs random search hyperparameter tuning, and the third model is a model with cross-fold validation hyperparameter tuning. As a result of modelling the data with two different regressors and multiple iterations of each regressor, the variable importance will be different. So, this project report will present graphs which display how variable importance changes between each regressor and each regressors iteration. Each model iteration has been compared based on the root mean square error (RMSE) and the final regressor iteration selected for deployment will be chosen based on the lowed RMSE score. Finally, this project report will discuss the ethical considerations of the project such as data integrity, data security and intellectual property concerns.

Exploration

Data/feature engineering

The data for the project was sourced from a previous project that one of the group members had worked on. The data set has 18 variables, 17 when you exclude the dependent variable, price. See the table below for a breakdown of the variables in the data set. There are four main treatments that we applied to the data set before modelling. The first was removal of specific variables. We removed the three generic variables of “date_keep”, “time_keep” and “index” which were not considered to be features. We then considered further feature engineering.

Feature engineering was important because we are trying to predict price 30min into the future. In real-time, both “RRP” (price variable) and “TOTALDEMAND” (demand variable) data are received at exactly the same time. So, in order to predict price one moment ahead of real-time, we had to shift the demand variable forward one moment in time, so as to say, the dependent variable price at time (T) is regressed against a demand variable observation at time (T-1).

Aside from this data treatment, it was decided that we would include a one hour & 2 hour moving average for the price and demand variables. Therefore, in order to do so, we had to first create a ‘Pricelag” variable (lagging the price variable by one observation means that you can use historical prices as a predictor of future prices). After this treatment, the demand and price variables has lagged equivalents; ‘Pricelag’ and ‘TOTALDEMAND’. Therefore, we applied a moving average of 1hr (the last 2 observations) and 2hrs (the last 4 observations) and created 4 new variables.

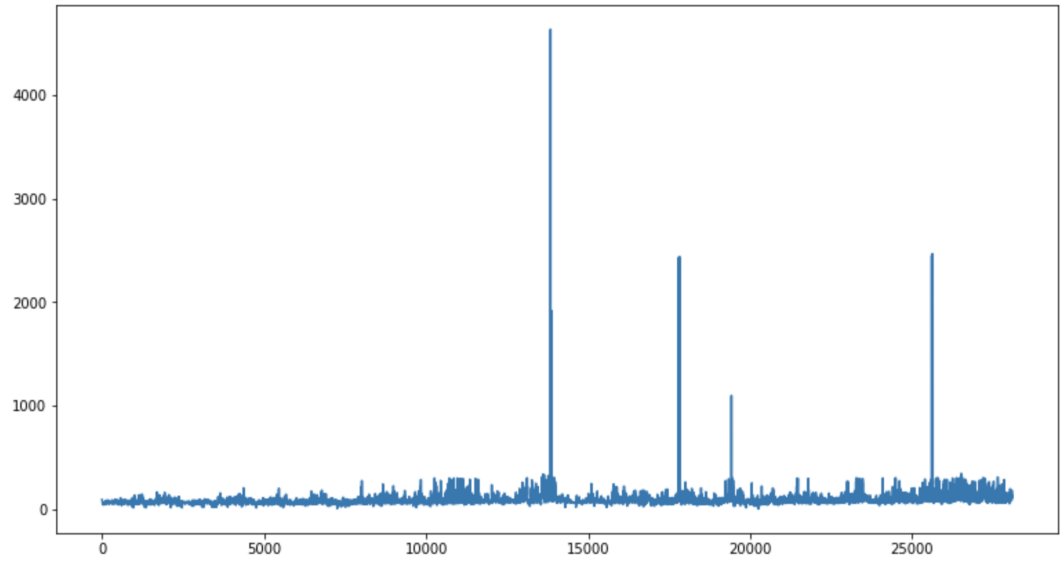
Variable	**Removed from data set**	**Feature Engineered**	**Structure**	**Structure Adjustment**
Year.x	No	No	Int	N/A
Month	No	No	Int	N/A
Day	No	No	Int	N/A
TOTALDEMAND	No	Yes	Float	N/A
RRP	No	No	Float	N/A
date_keep	Yes	No	Object	N/A

time_keep	Yes	No	Object	N/A
Daily.global.solar.exposure..MJ.m	No	No	Float	N/A
Minimum.temperature..Degree.C	No	No	Float	N/A
Maximum.temperature..Degree.C	No	No	Float	N/A
Rainfall.amount..millimetres.	No	No	Float	N/A
Season	No	No	Int	Category
price_mov_avg_1hr	No	Yes	Float	N/A
price_mov_avg_2hr	No	Yes	Float	N/A
demand_mov_avg_1hr	No	Yes	Float	N/A
demand_mov_avg_2hr	No	Yes	Float	N/A
segment_day	No	No	Int	Category
week_of_year	No	No	Int	Category
day_of_week	No	No	Int	Category
segment_30min	No	No	Int	Category
Pricelag	No	Yes	Float	N/A
index	Yes	No	Int	N/A

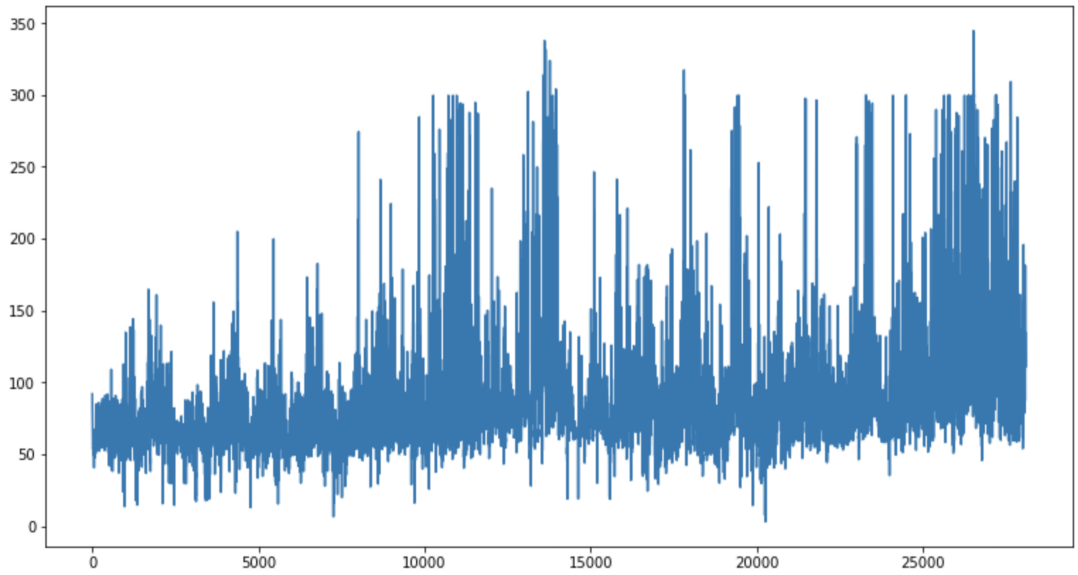
In terms of the adjustment if the structure of the variables, 5 variables were adjusted; the season, segment_day, week_of_year, day_of_week and sement_30min variables. The reasoning for structuring the variables in this format is based on a factorial being a better representation of the data, i.e., each factor is its own variable in the model.

Response Variable Exploration

The response variable Energy Price “RRP” exploration resulted in the finding of some outliers. It was decided to replace these observations with the median of the energy price. Leaving these values could negatively influence the performance of the models.

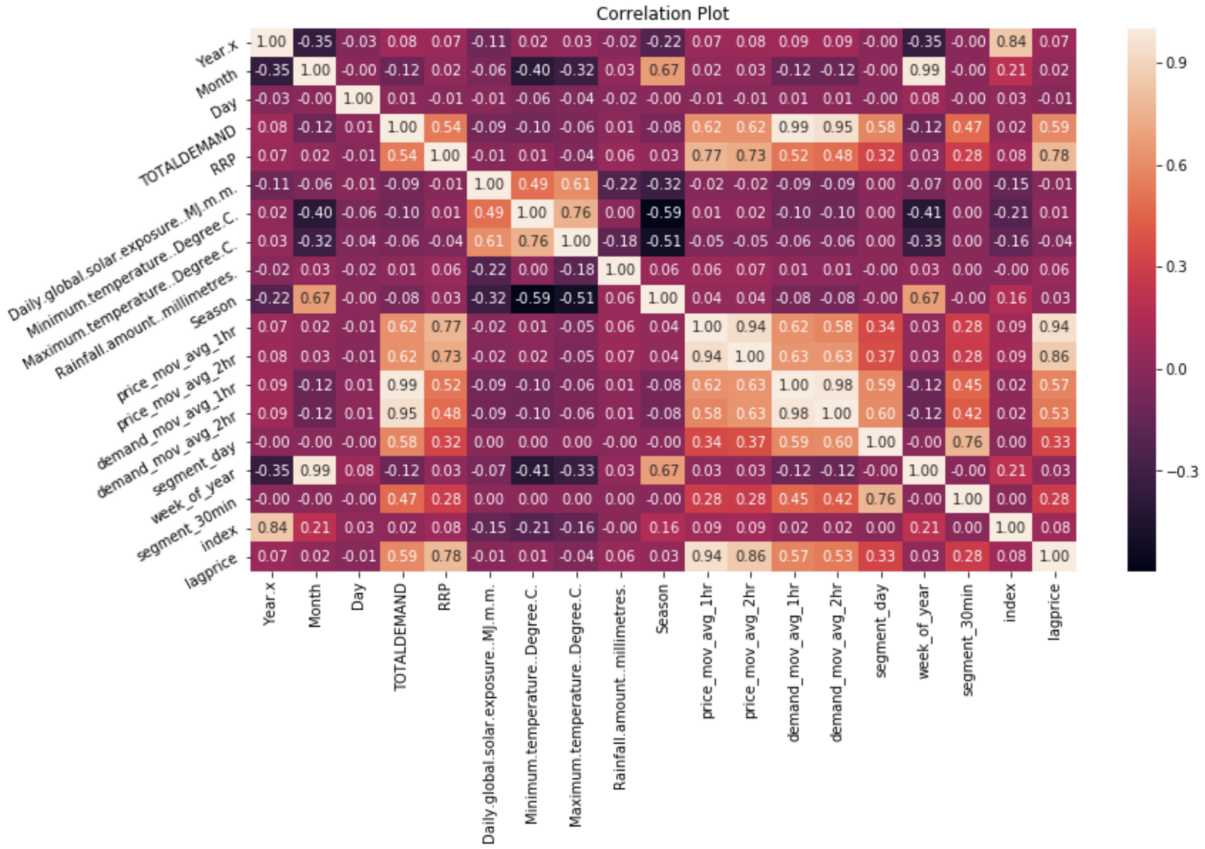


As it can be appreciated below, by replacing these values it’s easier to identify a trend and is expected to have a better performance of the models as the skewness is is diminished.

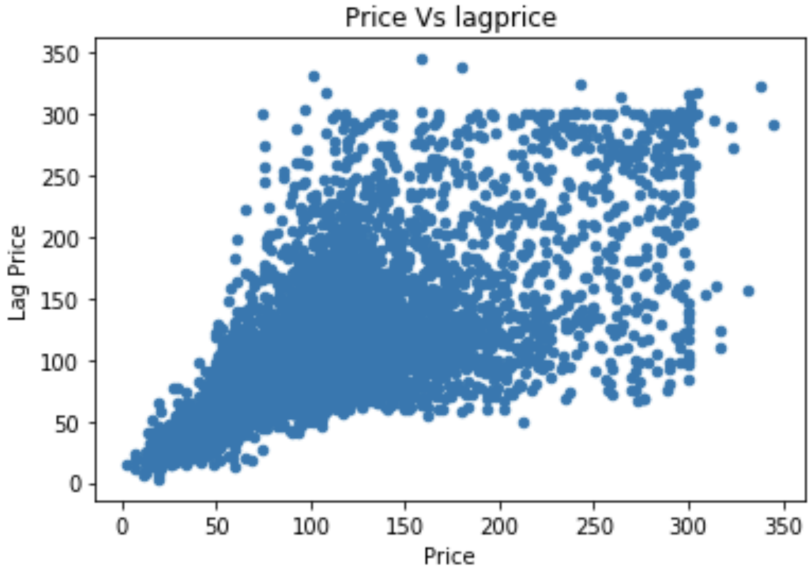


Correlation with Response Variable

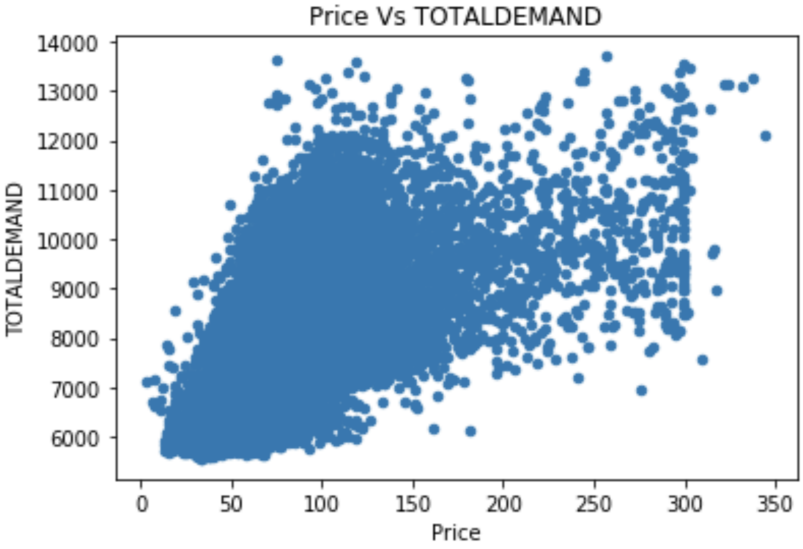
A correlation plot was constructed to identify which variables could be the most important features capable of explaining the behaviour of the energy price. As it can be appreciated below Lagprice, TOTALDEMAND, Price moving average 1h and Price moving average 2h are the highest correlated variables with Energy Price "RRP". These high correlated plots are expected to have relevant importance in the models which will be confirmed using "variable importance" plots.



The highest correlated variable was “lagprice” which was expected as it is the historic data of the energy prices. We can appreciate a positive relationship between them in the following figure.



Total Demand was also evaluated. Below a scatter plot demonstrates a positive relationship with Energy Price. This suggests that we can expect an increase in Price when the Demand increments.



Day of the week prices

An exploration to determine the prices for each day of the week was made where it was found that the highest prices could be Tuesdays and the lowest price on Sundays. Which intuitively could be explained by the demand.

Day of week	min	mean	max	std
Monday	17.14	84.221278	303.96	35.272261
Tuesday	14.46	86.714777	344.61	38.761274
Wednesday	14.63	88.041570	337.72	38.173997
Thursday	20.11	87.750868	317.28	36.992757
Friday	12.93	87.164587	331.58	36.070937
Saturday	6.48	77.082146	299.60	25.750350
Sunday	2.98	72.940628	299.60	27.794134

Test/train set split

Given that the data is time series in nature and we are trying to resolve a time series related problem, we can not use classical cross-validation procedures to split the data. Since time series data is contiguous, you cannot split the data randomly because it is ordered and destroying the order will likely ruin the correlation present in the time axis, as described by Monnier (2018). So it was decided that we would split the data at a 70/30 ratio between test and train – the first 19,656 observations were to be the training set and the remaining the testing set.

Algorithm selection

The Random Forest was selected as one of the two algorithms for the comparative study in order to predict electricity price based on the fact that it is one of the most accurate machine learning algorithms in existence, it can train very quickly and efficiently on large data sets, and it generates a highly unbiased estimate of the total error in the data set as the forest gets larger and learn on each tree, as explained by Radenkovic (2011). The Random Forest can be used for both classification and regression tasks, but for this project we will employ the regression algorithm which uses the squared error to calculated the purity measure. Given the pros of the random forest, it was appropriate to select another tree-based algorithm to include in the comparative study, the XGBoost algorithm.

The XGBoost algorithm employs what is know as gradient boosting and then pushes the absolute limits of this boosting for the most optimal results. Some of the benefits of the XGBoost algorithm include that it regularizes the data in order to prevent over-fitting, it is the best tree-based algorithm when it comes to pruning trees, it can support parallel processing, and it has the inbuilt feature to impute missing values, as described by Kumar (2019). For these reasons we have opted to include this algorithm in the comparative study in order to find the best tree-based algorithmic method for predicting electricity price.

Hyperparameter tuning

Hyperparameter tuning is extremely important when discussing the optimization of algorithms with close to ten model hyperparameters. Moreover, there are going to be more optimal combinations of hyperparameters than other in terms of finding the model with the best measurement metric RMSE. There are three methods that we have employed to train the two different regressors, which means that we have designed and tested 6 different models, 3 different models for each regressor. The methods are as follows;

- A model with no hyperparameter tuning, only specifying the number of trees (1000)
- A model with random search cross validation
- A model with grid search cross validation

Mandava (2018) described grid search as a dictionary of all possible hyperparameter combinations and then trains the data on every selection to find the best performing selection, this is in contrast with the random search by evaluating a specified amount of uniformly random points in the ‘hyperparameter dictionary’ and select the one with the best performance.

Methodology

As previously explained, each model (Random Forest and XGBoost) was iterated three times with different parameters. The objective was to improve the models using hyperparameter tuning. The first iteration was performed with default settings which was used as the baseline model. For the second iteration, hyperparameter tuning was performed with Random Grid Search Cross-Validation. For the third iteration Grid Search Cross-Validation was implemented.

To be able to create the models the data set was first split into training and test sets with a 70/30 ratio (refer to Test/Train split section). Each implemented model was trained and tested using the same data sets and evaluated under the same criteria.

The models were run using Google Colab GPU, results and estimated time are subject to this instance capacity.

1. Random Forest Baseline model.

The first model to be tested was Random Forest without any hyperparameter tuning. This model is to be considered the baseline for the classification models tested in this project as it used the default parameters which later compared with the tuned models results. This model was trained using the "Train Set" and then tested with the "Test Set". RMSE and were calculated for performance measurement and comparison purposes. Actual values versus prediction and future importance plots are reported for comparison purposes.

1.1. Random Forest Hyper Parameter Tuning

Hyperparameter tuning was then performed over Random Forest Baseline model using Cross-Validation Random Search and Cross-Validation Grid search. This tuning refers to settings specification for the implementation of the model. The hyperparameters tuned on this model were:

Hyperparameter	**Description**
Max Depth	The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure
Max Features	The number of features to consider when looking for the best split If auto: max_features=n_features. If sqrt :max_features=sqrt(n_features)
Min Sample Leaf	The minimum number of samples required to be at a leaf node
Min Sample Split	The minimum number of samples required to split an internal node
Number of Estimators	Number of times to go through the modeling cycle described

(Sklearn 2019)

The defined Grid search for both methods (Random and Grid) for the Random Forest model was based on the following parameters:

Hyperparamter	**Possible Values**
Bootstrap	true or false
Max Depth	All combinations from 10 to 110 in increment of 11
Max Features	'auto' or 'sqrt'
Min Sample Leaf	1, 2, 4
Min Sample Split	2, 5, 10
Number of Estimators	All combinations from 200 to 2000 in increment of 10

1.2. Random Forest Random Grid Search: This method tests random combinations from the parameter grid specified for the model. The purpose of this method is to verify if Hyperparameter tuning is able to improve the model performance. Then a model was trained using the random combinations of the grid parameters and tested over the Test Set.

Random Grid Search method allows to determine the best parameters to fit the model under a low period of time (compared to testing all the possible combinations like it's done in Grid Search), by choosing random combination of parameters on the specified grid, test them and compare the RMSE with the other iterations.

The best parameter combination was then used to replicate the model and predict Energy Prices. These predictions were then compared with the "Test Set" to measure the model RMSE. The following table reports the best parameters chosen by the Random Forest Random Grid Search

Hyperparamter	Best Value
Bootstrap	True
Max Depth	90
Max Features	sqrt
Min Sample Leaf	4
Min Sample Split	2
Number of Estimators	2000

1.3. Random Forest "Grid Search": This method is similar to the "Randomized Grid Search" with the differential characteristic that these method testes every single combination of the values form the specified hyperparameter grid. This method results in the best parameter combination from all the possible combination from the grid. The trade off for testing all the combinations is a higher computational time.

Predictions were also calculated using this hyperparameter selection and an RMSE and were also estimated. The following table reports the best parameters chosen by the Random Forest Grid Search

Hyperparamter	**Best Value**
Bootstrap	True
Max Depth	110
Max Features	3
Min Sample Leaf	4
Min Sample Split	10
Number of Estimators	200

2. XGBoost Baseline model.

A baseline model for the XGBoost was performed in which default Hyperparameters were used for its training. This model was also trained and test with the same partition data set as the Random Forest. Prediction were calculated and RMSE and Accuracy were obtained based on these predictions.

2.1. XGBoost Hyper Parameter Tuning

Randomized Grid Search and Grid Search were applied to find the best Hyperparameters. The methodology applied for XGB models was the same as the one implemented for the Random Forest, thus will not be detailed explained. The hyperparameters tuned on this model were:

Hyperparamter	**Description**
Subsample (Eta)	Subsample ratio of the training instances. (Fixed to 1 because is a time series)
Number of Estimators	Number times to go through the modeling cycle described
Min Child Weight	minimum number of instances needed to be in each node. The larger min_child_weight is, the more conservative the algorithm will be.
Max Depth	Maximum depth of a tree. Increasing this value will make the model more complex and more likely to overfit.
Learning Rate	Step size shrinkage used in update to prevent overfitting.
Gamma (Min Split loss)	Minimum loss reduction required to make a further partition on a leaf node of the tree. The larger gamma is, the more conservative the algorithm will be.

(Aarhay Jain 2016)

The defined Grid search for both methods (Random and Grid) for the XGBoost model was based on the following possible parameters:

Hyperparamter	**Possible Values**
Subsample	1
Number of Estimators	100, 200, 500, 1000
Min Child Weight (Default = 1)	1, 2, 4
Max Depth (Default 6)	4, 6, 8
Learning Rate (Default = 0.3)	0.01, 0.05, 0.1
Gamma (Default 0)	0.1, 0.2, 0.3, 0.4, 0.5

2.2. XGB Random Grid Search

This method tested random hyperparameters combination from the defined grid. The best combination found with this method was used to make Energy Price predictions. RMSE and were estimated for this model and compared to the results from the baseline method. The following table reports the best parameters chosen by the XGB Random Grid Search.

Hyperparamter	Value
Subsample	1
Number of Estimators	1000
Min Child Weight	4
Max Depth	4
Learning Rate	0.01
Gamma	0.4

2.3. XGB Grid Search

Every single combination from the grid was tested and the best combination was used to create a model and make predictions for the Energy Price. RMSE were calculated. The following table reports the best parameters chosen by the XGB Grid Search.

Hyperparamter	Value
Subsample	1
Number of Estimators	1000
Min Child Weight	4
Max Depth	4
Learning Rate	0.01
Gamma	0.0

Evaluation

Given that this project has been designed as a comparative study of two tree-based regressors, the models will be compared based on the root mean square error measure (RMSE). Moreover, we will compare the results, graphically, of the predictions that each model iteration makes against the testing data set. Finally, we will compare the variable importance measures of each model iteration to examine how each model values the variables differently.

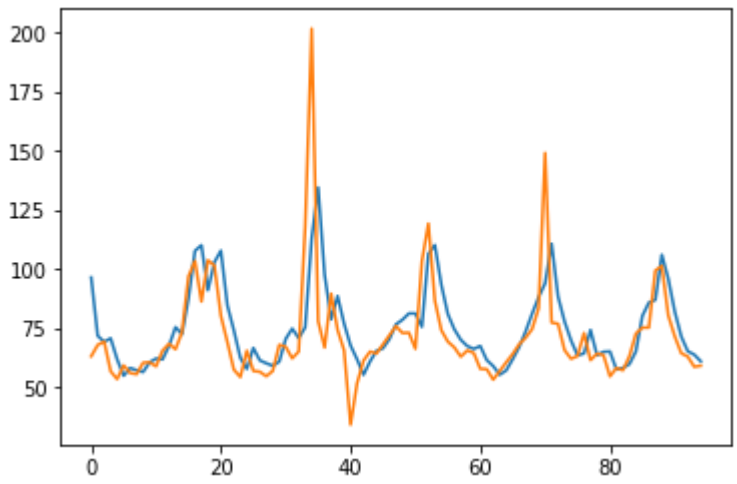
1.1 Baseline Random Forest Model

1.1.1 RMSE

The RMSE for the Baseline Random Forest model returned a measure of 27.17. This measure on its own has little to no value but it will act as a benchmark for the RMSE measures returned by two other model iterations of the Random Forest.

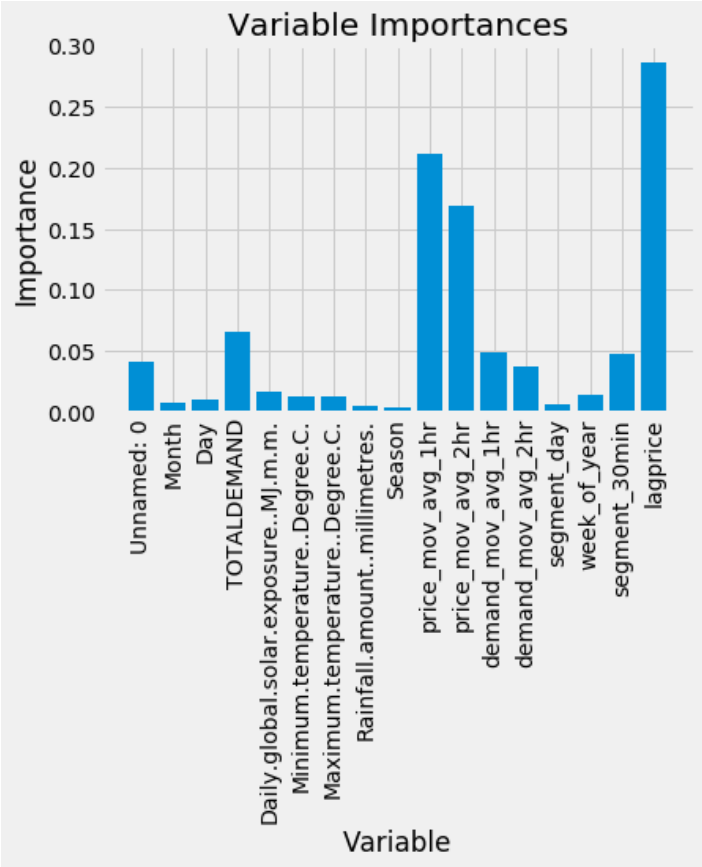
1.1.2 Test set versus prediction

As can be seen in the graph below, two lines are plotted. The orange line is the predicted electricity price and the blue line is the test set electricity price. As we can see, the predicted electricity price 30 minutes into the future is doing a reasonable job. It is modelling the general behaviour of the daily peaks and troughs quite well, however, it does seem to lag the test set price observations when the price decreases from each peak.



1.1.3 Variable Importance

The variable importance of the Baseline Random Forest is displayed below. Since Random Forests split data based upon the node purity, the mean square error of a variable in the model when using regression, each variable is therefore ranked based upon this purity measure. As we can see, the top three most important variables in the Baseline Random Forest Model are 'lag price', 'price_mov_avg_1hr' and 'price_mov_avg_2hr'. It is not a surprise that historical price is an excellent predictor of future price movements.



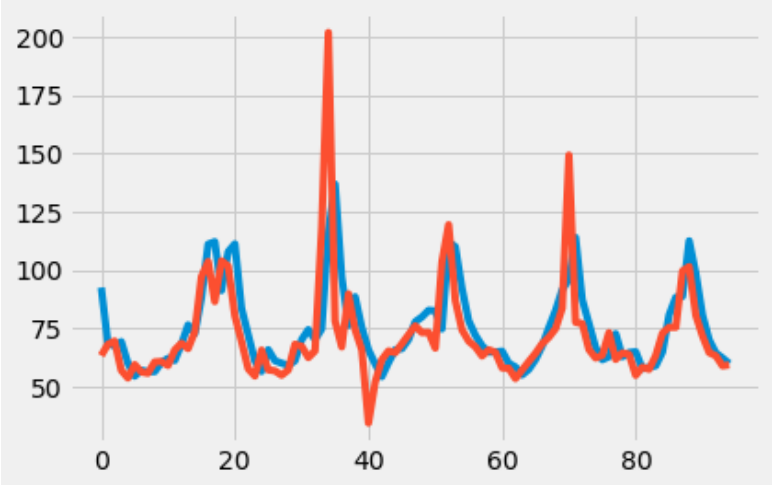
1.2 Random Grid Random Forest

1.2.1 RMSE

The RMSE for the Random Grid Search Random Forest was 27.09. This represents a decrease in the RMSE from the Baseline Random Forest of 0.08. It can therefore reasonably be stated that the randomised grid search for optimal hyperparameter combinations has successfully reduced the error in the model.

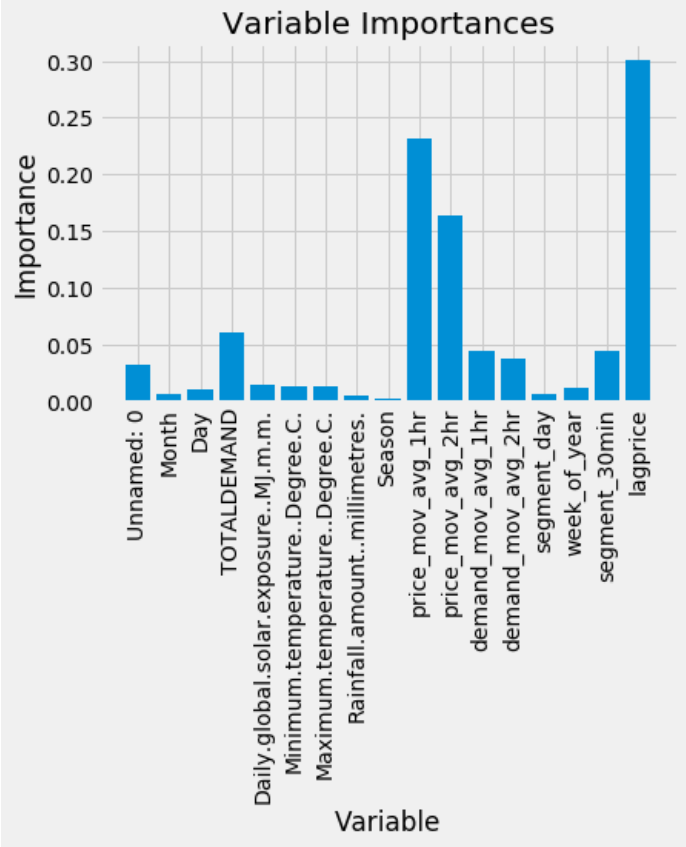
1.2.2 Test set versus prediction

Similarly, the table below represents the Random Grid Search Random Forest price predictions (red) and the test set price observations (blue). Visually, the graph looks bery similar to that produce by the Baseline Random Forest model. The model has again predicted far in excess of two of the peaks in the testing data set just as the Baseline Random Forest model has.



1.2.3 Variable Importance

The variable importance measures of the Random Grid Search Random Forest are displayed in the graph below. The case is similar to that of the Baseline Random Forest model as it has identified the same top three most important variables. It should be noted though that this model has reduced the importance of those variables with low importance (close to zero) in the Baseline Random Forest model closer to zero.



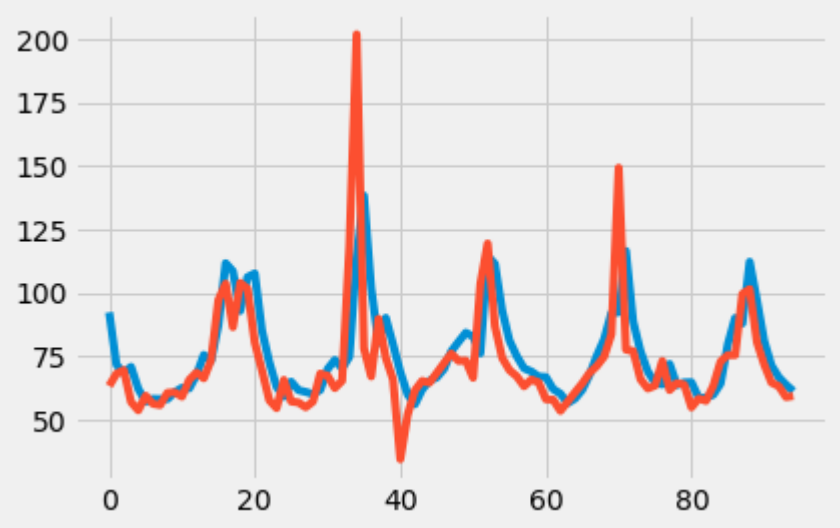
1.3 Grid Search Random Forest

1.3.1 RMSE

The RMSE of the Grid Search Random Forest model returned a measure of 27.29. In comparison to the Baseline Random Forest and the Random Grid Random Forest, this model performed the worst.

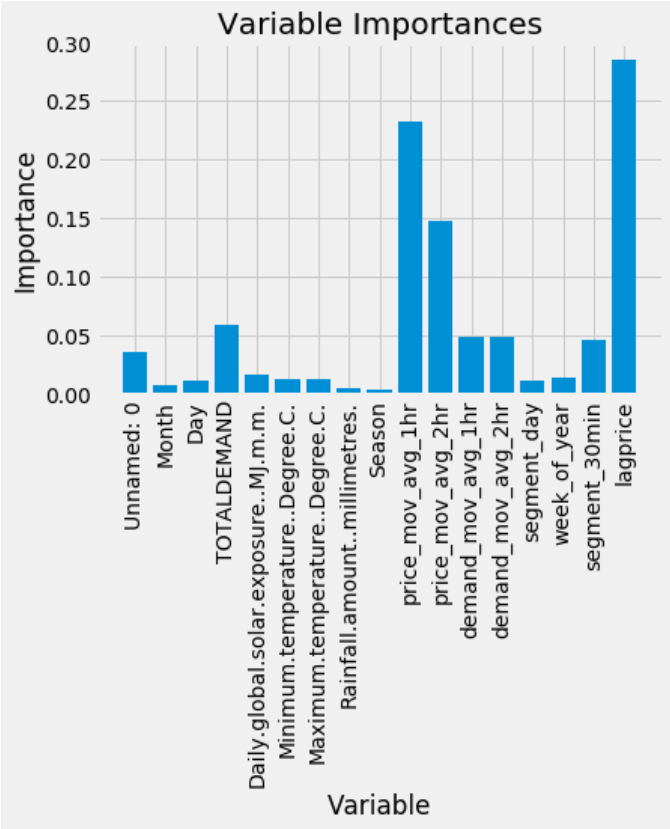
1.3.2 Test set versus prediction

Below is a graph of the predicted price values from the Grid Search Random Forest (red) versus the test set price observations (blue). It should be noted that again, this model presents visually similar to the Baseline Random Forest and the Random Grid Random Forest models. It predicts outlier price peaks for the same two daily peaks as the other two models. This could therefore represent an issue with the dataset preparation not hyperparameter tuning.



1.3.3 Variable Importance

Below is a graph of the variable importance for the Grid Search Random Forest model. This is a very similar, if not the same, as the variable importance graphs previously presented. The top three most important variables have not changed. Interestingly, the differential between the ‘lagprice’ and the ‘price_mov_avg_1hr’ variables has become marginally smaller, which indicates that this model evaluates the purity of these two variables to be more similar than the previously discussed models.



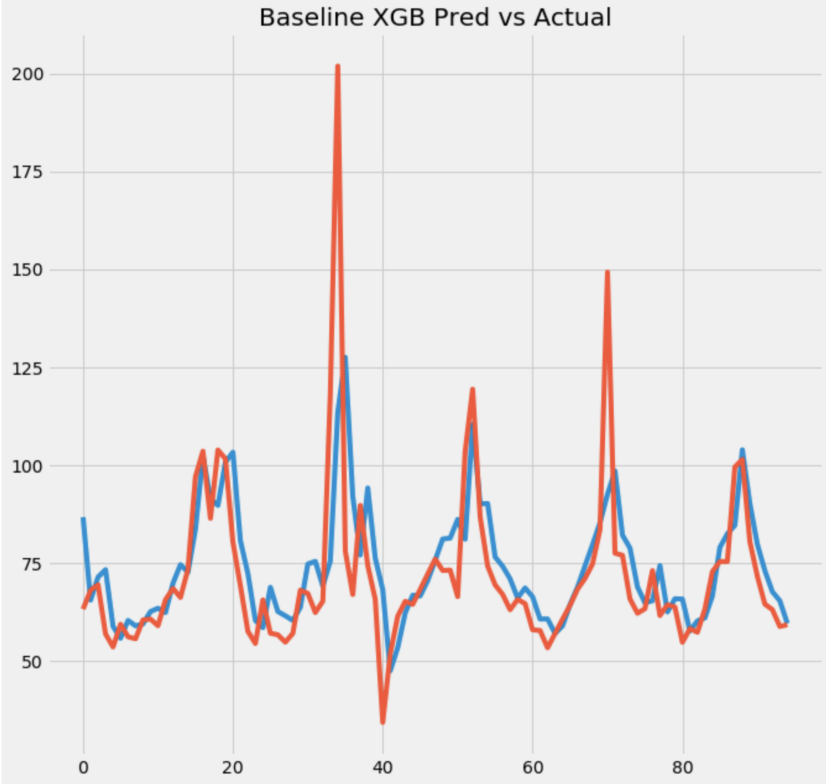
2.1 XGBoost - Baseline Model (no hyperparameter tuning)

2.1.1 RMSE

The RMSE for the Baseline XGBoost model returned a measure of 26.79. This measure will be used as a benchmark for the RMSE measures returned by two other model iterations of the XGB with hyperparameter tunning.

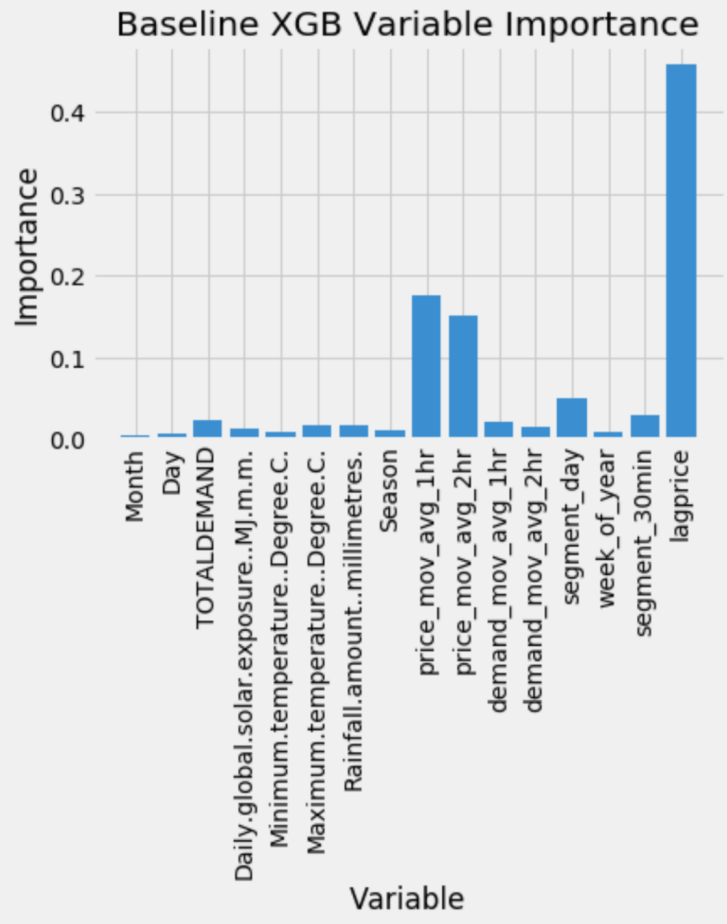
2.1.2 Test set versus prediction

As can be seen in the graph below, two lines are plotted indicating the behavior of the energy price. The blue line is the predicted electricity price and the orange line is the test set electricity price. As we can see, the predicted electricity price 30 minutes into the future is doing a reasonable job. Despite the fact that the graphics for the baseline model for both XGB and Random Forest looks quite similar, its apreciatable that XGB has make a better job at responding during price peaks.



2.1.3 Variable Importance

The variable importance of the Baseline XGB is displayed below. XGBoost follows the same logic as Random Forest, where it to split data based upon the node purity and ranked based upon this purity measure. As we can see, the top three most important variables in the Baseline Random Forest Model are 'lag price', 'price_mov_avg_1hr' and 'price_mov_avg_2hr'.



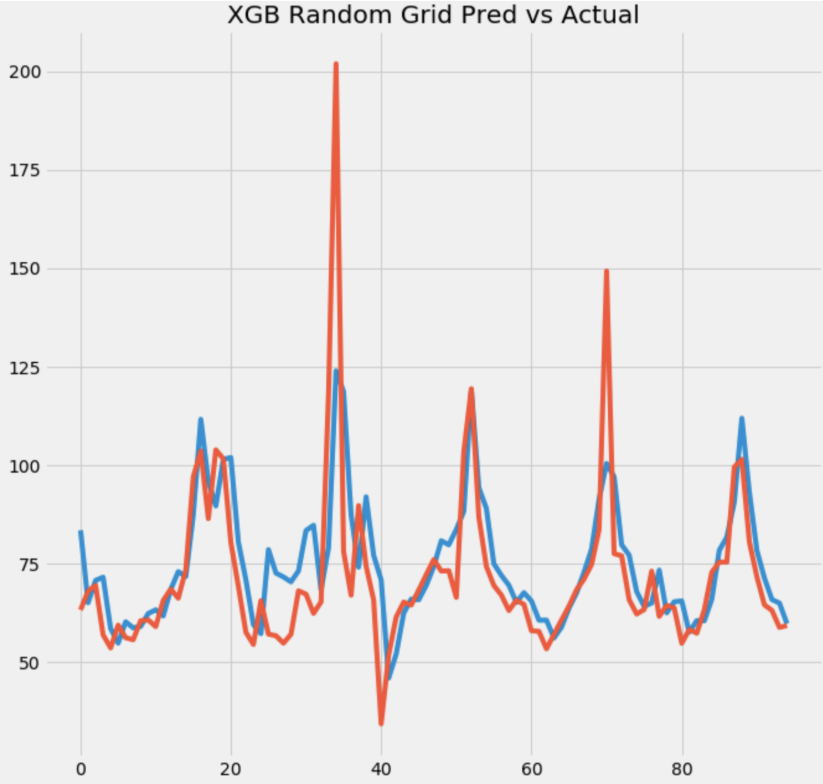
2.2 XGBoost - Random search cross validation

2.2.1 RMSE

The RMSE for the Random Grid Search XGBoost was 26.20. This represents a decrease in the RMSE from the Baseline Random Forest of 0.59. It can therefore reasonably be stated that the randomised grid search for optimal hyperparameter combinations has successfully reduced the error in the model.

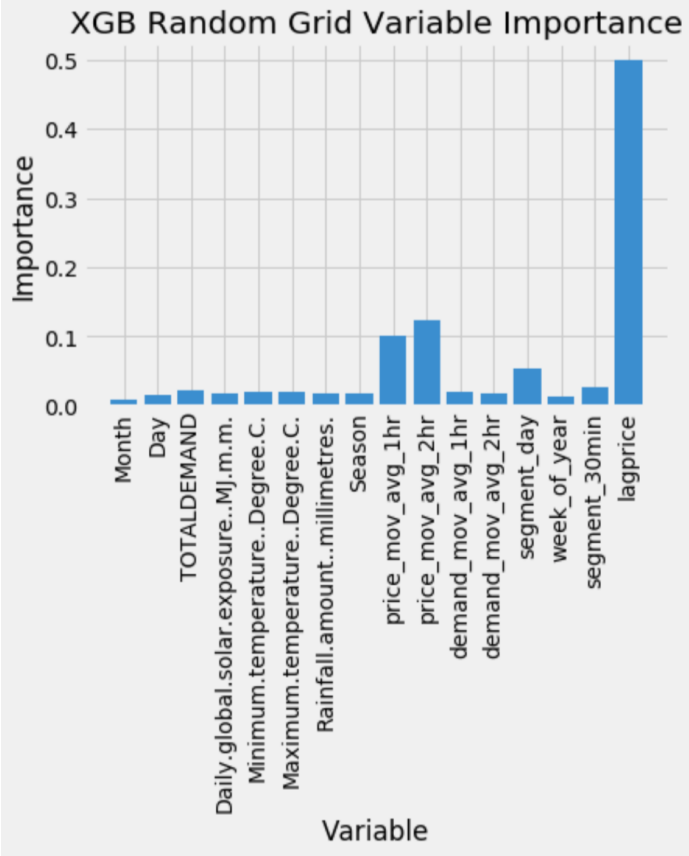
2.2.2 Test set versus prediction

Similarly, the table below represents the Random Grid Search XGBoost price predictions as the blue line and the test set price observations as the red line. Visually, the graph looks very similar to that produced by the Baseline XGB model.



2.2.3 Variable Importance

The variable importance measures of the Random Grid Search XGBoost are displayed in the graph below. The case is similar to that of the BaselineXGB model as it has identified the same top three most important variables. It should be noted though that this model has increased the importance of “lagprice” variable by 0.4 points and reduced the importance for price_mov_avg_1hr by 0.6 and price_mov_avg_2hr by 0.5 points.

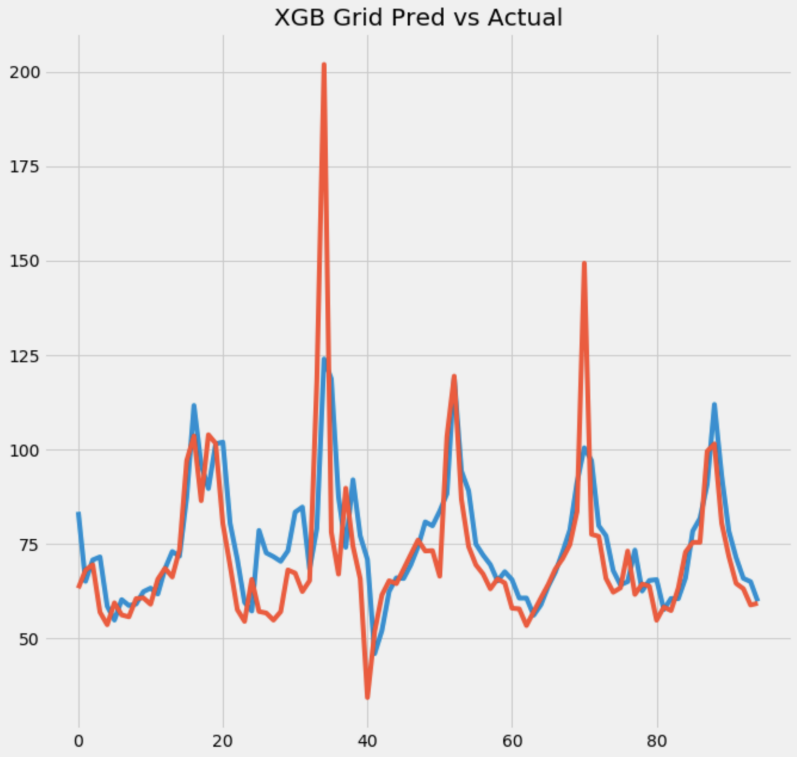

XGB Random Grid Variable Importance
Importance
0.5
0.4
0.3
0.2
0.1
0.0
Month
Day
TOTALDEMAND
Daily.global.solar.exposure..MJ.m.m.
Minimum.temperature..Degree.C.
Maximum.temperature..Degree.C.
Rainfall.amount..millimetres.
Season
price_mov_avg_1hr
price_mov_avg_2hr
demand_mov_avg_1hr
demand_mov_avg_2hr
segment_day
week_of_year
segment_30min
lagprice
Variable

2.3 XGBoost - Grid Search

2.3.1 RMSE

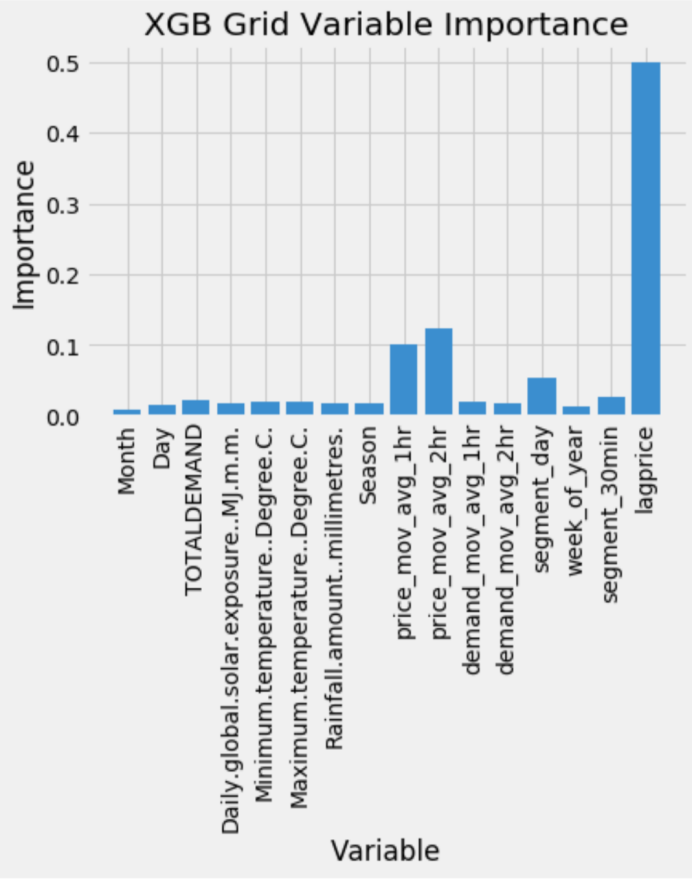
Despite the values for the hyperparameter selection on the grid where different. The RMSE for the Grid Search XGBoost was 26.20, the same result achieved on the Random Grid Search XGB.

2.3.2 Test set versus prediction



2.3.3 Variable Importance

Below is a graph of the variable importance for the Grid Search XGB model. The variable importance remained the same as in the Random Grid Search XGB model.



3.0 Comparison of performance measures

Model Performance Comparison

	RMSE	Time
Baseline RF	27.16	6 min
CV Random Search RF	27.08	31 min
CV Grid Search RF	27.27	1h 21min
Baseline XGB	26.79	1.42 sec
CV Random Search XGB	26.20	7 min
CV Grid Search XGB	26.20	59 min

Discussion

As it was expected the best performance was obtained with the XGBoost with Random Grid Search hyperparameter tuning. This model outperformed the Random Forest with Grid Search tuning by 1.73 RMSE points.

Despite both grid search for the XGB model returned the same results. The XGB Random Grid Search is selected as the best model performed. The RMSE result and the computational time required to achieve places this model as the most efficient and effective from all the tested models.

Given that this project has considered two models in the comparison study, there could be possible improvements on each model for future iterations of the prediction improvement. In terms of the Random Forest, we considered and tested two of the best approaches to optimising a Random Forest Model, so it is likely that we will not be able to optimise the model in this aspect. Possible future directions for improvement could be seen through;

- The use of a larger data set so that the model has a better opportunity to learn
- The inclusion of more features and/or better quality features
- The use of a different algorithm to model the data, such a time series forecasting model, e.g., ARIMA or Prophet.

Conclusion

This practical project has considered the problem of trying to predict electricity price, specifically, one period ahead of time (30 minutes). In order to do so, a comparison study has been designed to build, test and compare two tree-based regressors, a Random Forest and an XGBoost. Moreover, each regressor had three individual iterations; a model with no hyperparameter testing, a model with random grid search hyperparameter tuning, and a model with grid search hyperparameter tuning. Each model would be compared based on the RMSE measurement metric. The results of the comparison study conclude that the XGBoost algorithm was superior in predicting electricity price over the Random Forest. The Random Forest with Random Grid Search returned the lowest RMSE of the Random Forest Models. The XGBoost model with hyperparameter tuning also returned the lowest RMSE of the XGBoost model iterations. Comparing the two regressors, the XGBoost model with hyperparameter tuning produced the lowest RMSE out of all models in the entire study. Furthermore, it is recommended that future directions for such a project focus more time and energy into acquiring a larger data set, acquiring better quality features and/or engineering powerful variables, and perhaps implementing time series forecasting models such as ARIMA or Prophet. This project has successfully designed multiple iterations of two regressors in a comparison study in order to find and produce a model that is able to minimise predictive error.

Data Ethics

Having conducted and concluded this practical project, two ethical considerations have become apparent as a result of data analysis and experimentation. Those ethical considerations are transparency and privacy. One way in which the mitigation of ethical concerns and risks regarding transparency has been by keeping all of the project code publicly available on the project member's Github accounts. The fact that the Github repositories are publicly available mitigates risk of wrongful doing and invites the public to audit the project. As Saqr (2017) explains, in an ideal world, transparency should be applied to all development phases of an analysis of data. Therefore, it is at the conclusion of this project that the consideration for transparency the data analysis is in-line with current think on data transparency.

Data privacy was another key ethical concern for the project since the proliferation of data theft and reidentification has become more common in the digital era. The data for the project was sourced from one of the project team member's previous projects, and this data set was aggregated using publicly available data sets. Moreover, the data set, including the feature engineered variables, has been made publicly available on the project members' Github accounts. As Mooney & Pejaver (2018) argue, data that concerns the characteristics of people is the most at-risk type of data in relation to privacy concerns, but even a data set without said traits can still put the creator or aggregator of a data set at risk. Therefore, privacy risk could potentially stem from the fact that the data set has been made publicly available against the creators wishes, but this is not the case. It can therefore be stated that this project has made a reasonable attempt to mitigate privacy risk during the course of the project.

References

Aarshay Jain. 2016, 'Complete Guide to Parameter Tuning in XGBoost with codes in Python', Analytics Vidhya, <https://www.analyticsvidhya.com/blog/2016/03/complete-guide-parameter-tuning-xgboost-with-codes-python/>.

Kumar, D. 2019, Machine Learning Series: The XGBoost Algorithm in Python, O'Reilly, viewed 19 September 2019, <http://find.lib.uts.edu.au/search?N=0&Ntk=All&Ntx=matchallpartial&Ntt=benefits%20of%20XGBoost>.

Mandava, S. 2018, 'Cross Validation and HyperParameter Tuning in Python', *Medium*, weblog, Medium, 19 September 2018, viewed 15 September 2019, <https://medium.com/@mandava807/cross-validation-and-hyperparameter-tuning-in-python-65cfb80ee485>.

Monnier, S. 2018, 'Cross-validation tools for time series', *Medium,* weblog, Medium, 8 September 2018, viewed 20 September 2019, <https://medium.com/@samuel.monnier/cross-validation-tools-for-time-series-ffa1a5a09bf9>

Mooney, S. & Pejaver, V. 2018, 'Big Data in Public Health: Terminology, Machine Learning, and Privacy', *Annual Review of Public Health,* vol. 39, pp. 95-112.

Radenkovic, P. 2011, 'Random Forest', *3273/10,* powerpoint slides, University of Belgrade, Belgrade, viewed 16 September 2019, <http://home.etf.rs/~vm/os/dmsw/Random%20Forest.pptx>.

Saqr, M. 2017, 'Big data and the emerging ethical challenges', *International Journal of Health Sciences*, vol. 11, no. 4, pp. 1-2.

Sklearn DecisionTreeRegressor, sklearn, viewed Sept 16, 2019, <https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html>

XGBoost Parameters documentation, viewed Sept 17, 2019, <https://xgboost.readthedocs.io/en/latest/parameter.html>.