A Data Science Approach to Short-Term Energy Demand Forecast

Louis Long (z5162359) Mark Budrewicz (z5353932) Hao Chiou (z5131909)
Charlet Jeyapaul (z5375906)

07/10/2023

Contents

1 Abstract

2 Literature review

3 Material and Methods

3.1 Software

Our analysis and modelling were conducted using the R programming language, known for its proficiency in statistical analysis and data manipulation. We leveraged several essential R packages to streamline data management and analysis, empowering us to build robust predictive models. Some of the key R packages we used include:

- **tidyverse:** This comprehensive package suite equipped us with a powerful set of tools for data manipulation, visualization, and analysis. Its flexibility and integration with other packages were essential for our data preparation and exploration.
- **lubridate:** Essential for working with date and time data, a crucial aspect of time series analysis. This package allowed us to handle temporal information efficiently and accurately.
- **data.table:** We harnessed the efficiency of the data.table package for data manipulation tasks, particularly when dealing with large datasets. Its speed and concise syntax were instrumental in aggregating and selecting columns.
- **crayon:** Used for adding colourful text to our visualizations, enhancing their interpretability, and making our reports more engaging.
- **zoo:** As a pivotal package for handling time series data, zoo played a crucial role in our analysis. It provided the tools needed to process, manipulate, and visualize time-based data structures.
- **tsibble:** Facilitating the management of time series data, tsibble served as the foundation for our time-based operations, ensuring consistency and accuracy in our analyses.
- **gridExtra, grid, patchwork:** These packages enabled us to create complex grid layouts for our visualizations, enhancing the presentation of our findings and insights.
- **GGally:** This package expanded our ggplot2-based visualizations by offering additional plot types and features, enriching our data exploration.

3.2 Data Acquisition

Our analysis relied on hourly records of critical variables, including electricity demand, temperature, rainfall, solar exposure, and forecasted demand. We efficiently imported these data sources into R using the `fread` function from the data.table package, ensuring a rapid and seamless data import process.

3.3 Pre-processing Steps

- **Handling Duplicates:** We prioritized data integrity by identifying and meticulously removing duplicate records from the total electricity demand dataset.
- **Date Matrix Creation:** To standardize time intervals in the total demand dataset, we crafted a date matrix with 5-minute intervals. Missing values within this dataset were expertly interpolated using the 'na.approx' function.
- **Temperature Data:** Suspicious temperature values below -23°C were logically removed from the dataset, as they were considered unrealistic for New South Wales. The temperature dataset was further polished by interpolation.

- **Forecast Data:** We efficiently extracted initial and final forecast data from the forecast demand dataset. Time interval inconsistencies were addressed, and missing values were filled using a meticulous down-fill approach.
- **Aggregation:** In alignment with our modelling approach, we conducted hourly aggregation on the prepared datasets to harmonize temporal resolutions.

3.4 Feature Engineering

- **Date-related Features:** Our analysis benefitted from a range of date-related features, including season, season indicators (summer and winter), day/night indicators, and days of the week. We also classified days based on business days (weekdays) and weekends, enriching our dataset with temporal insights.
- **Temperature-related Features:** Various temperature-related features were artfully engineered to capture temperature extremes and categorize them into hot/cold day/night and extreme categories, providing valuable context for our modelling.

3.5 Data Integration

- **Rainfall and Solar Exposure:** To enhance our dataset's comprehensiveness, we seamlessly integrated rainfall and solar exposure data. We accomplished this by joining the datasets using a date-based key and expertly interpolating any missing values.
- **Public Holidays:** The inclusion of public holiday data for New South Wales enriched our dataset with additional contextual factors, offering valuable insights for our analysis.

3.6 Modelling Methods

We implemented three different predictive models to forecast electricity demand:

Lasso Regression:

- To address overfitting and improve model interpretability, we applied Lasso regression with cross-validation to select the best regularization parameter (lambda).
- The selected model was trained on the training data and evaluated using Mean Absolute Error (MAE) and Mean Squared Error (MSE) on the test data.

Support Vector Regression (SVR):

- We explored SVR models with various hyperparameters (epsilon and kernel) to capture non-linear relationships in the data.
- Multiple SVR models were trained and evaluated on the test data, with performance metrics reported.

Random Forest:

- Hyperparameter tuning was performed for Random Forest models, including parameters like the number of trees, mtrices, and sample rate.
- We trained two Random Forest models, one with all features and one with a reduced feature set based on variance importance.
- Model performance was assessed using test data, and predictions were compared with actual demand values.

3.7 Model Selection and Evaluation

We compared the performance of our predictive models on a holdout dataset:

- **AEMO Benchmark:** We benchmarked our models against the Australian Energy Market Operator (AEMO) forecast to evaluate their performance in a real-world context.
- **Lasso Model:** We evaluated the performance of our Lasso regression model, comparing its predictions with actual demand values using MAE and MSE.
- **Support Vector Regression:** We assessed the SVR models' performance, considering various combinations of hyperparameters, and compared them to actual demand.
- **Random Forest:** The Random Forest models' performance was evaluated using test data, and the predictions were compared with actual demand values.

3.8 Data Exploration

Model Evaluation:

Our evaluation process was meticulous, aiming to gauge the predictive accuracy of our models. We employed well-established error metrics, including Mean Absolute Error (MAE) and Mean Squared Error (MSE), to assess the performance of our predictive models rigorously. These metrics served as valuable guides in the selection of the most suitable model.

Key EDA Visualizations:

In addition to model evaluation, we conducted a comprehensive Exploratory Data Analysis (EDA) that unveiled essential insights into our dataset. Some of the pivotal plots used in our EDA included:

- **Boxplot of Hourly Total Demand by Month:** This visualization shed light on the monthly variations in electricity demand, offering valuable insights.
- **Line Chart of Forecast vs. Actual (2010 & 2021):** A comparison of forecasted demand against actual demand for the years 2010 and 2021, aiding in the assessment of predictive accuracy.
- **Line Charts of Total Demand vs. Temperature:** These visualizations illustrated the relationship between electricity demand and temperature fluctuations, uncovering crucial patterns.
- **Total Demand vs. Temperature Pair plot:** An analytical tool that revealed intricate correlations between total demand and temperature.
- **Multivariate pair plot:** A multidimensional exploration of data relationships, helping us identify complex dependencies.
- **Lagged Correlations:** Insights into how past values of variables correlated with future electricity demand, providing a deeper understanding of temporal dependencies.

These visualizations played a pivotal role in our analysis by helping us uncover insights, identify outliers, and validate the effectiveness of our modelling approach. In the subsequent 'EDA Section,' we will delve deeper into the insights gained from our exploratory data analysis, which involved the creation of various plots and visualizations to uncover patterns and relationships within the data.

4 Exploratory Data Analysis

5 Analysis and Results

5.1 Modelling setup

A few pre-processing steps were required before performing the demand forecast modelling, which included defining additional features, removing data where lagged variables do not exist, and splitting the dataset into training, test and holdout sets.

As our models will prioritise predictive performance over interpretability, it is important to include any potential features which may help in more accurately predicting energy demand. However, potential issues such as model bias or overfitting may arise, so an appropriate experimental setup was implemented to ensure the final model selected would minimise these occurrences. Some additional features include:

- Dummy variables for **hour of day** (denoted `HOUR_1`, `HOUR_2`, ..., `HOUR_23`, with midnight being the baseline hour)
- Dummy variables for **day of week** (denoted `weekday_2`, `weekday_3`, ..., `weekday_7`, with Monday being the baseline day of week)
- Dummy variables for **month of year** (denoted `MONTH_2`, `MONTH_3`, ..., `MONTH_12`, with January being the baseline month)
- **Lagged demand**, which includes demand from the previous five periods as well as the corresponding demand 24 hour ago. From the data exploration section, these periods demonstrated the highest correlation with present demand

To account for the inclusion of lagged demand, the range of the dataset had to be adjusted slightly to remove any rows which had missing lagged demand due to constraints in the date range of the data. Thus, the first 24 hours of the dataset was removed for modelling. From an overall perspective, this should have minimal impact on the trained models when compared with the overall coverage of data available.

Finally, the final dataset was split into three main sets: training, test, and holdout. This was done in order to fairly evaluate the performance of each model.

The training set would be used to determine the model based on the model type and selected hyperparameters if present. The test set would then be used to evaluate which set of hyperparameters for a particular model type has the best predictive performance. Finally, the holdout set is used to simulate a production environment where the model will be used to evaluate data after the model was created.

It is important to note that unlike non time series related modelling which would involve random sampling at a particular ratio, the split of training, test and holdout sets here are determined by set time periods due to the inherent correlation between adjacent data points.

The table below summarises the split used in our modelling section. Due to the large time range present and granularity of our data, there were no concerns on insufficient coverage across any sets. However, it was important to include at least a full year's worth of data in each step to account for potential seasonality within a calendar year.

Table 1: Split between modelling sets

Model.set	Datetime.start	Datetime.end
Training	2010-01-02 00:00:00	2018-07-31 23:00:00
Test	2018-08-01 00:00:00	2020-07-31 23:00:00
Holdout	2020-08-01 00:00:00	2022-08-01 00:00:00

5.2 LASSO

The first model that was trialed was the lasso model, which includes a shrinkage hyper-parameter `lambda` to determine the penalty of additional features to prevent over-fitting.

$$y_i = \sum_j x_{ij}\beta_j + \lambda \sum_j |\beta_j|$$

In order to ensure that the shrinkage feature works as intended, the input data was first scaled using the `scale` function. In this way, certain coefficients would not be penalised harder for having naturally larger numbers over others such that each variable would be equally considered for having their coefficients reduced.

The code snippet below shows the list of `lambda` considered. The key design decision was to ensure that the `lambdas` considered spanned across several orders of magnitude.

```
##  [1]  0.01000000  0.01258925  0.01584893  0.01995262  0.02511886  0.03162278
##  [7]  0.03981072  0.05011872  0.06309573  0.07943282  0.10000000  0.12589254
## [13]  0.15848932  0.19952623  0.25118864  0.31622777  0.39810717  0.50118723
## [19]  0.63095734  0.79432823  1.00000000  1.25892541  1.58489319  1.99526231
## [25]  2.51188643  3.16227766  3.98107171  5.01187234  6.30957344  7.94328235
## [31] 10.00000000
```

The process of deciding which lambda to consider was achieved using cross-validation. Thus, both the training and test sets defined in the previous section was used here. As there was a large span of historical data for cross-validation, the number of folds chosen of five was relatively low.

From the results of cross-validation in the chart below, it can be seen that there is a clear upward trend in MSE as lambda increases. This shows that penalising coefficients of variables has had an adverse effect on predictive performance. Thus, it made sense to choose the smallest lambda from the set of considered values for our final lasso regression model.

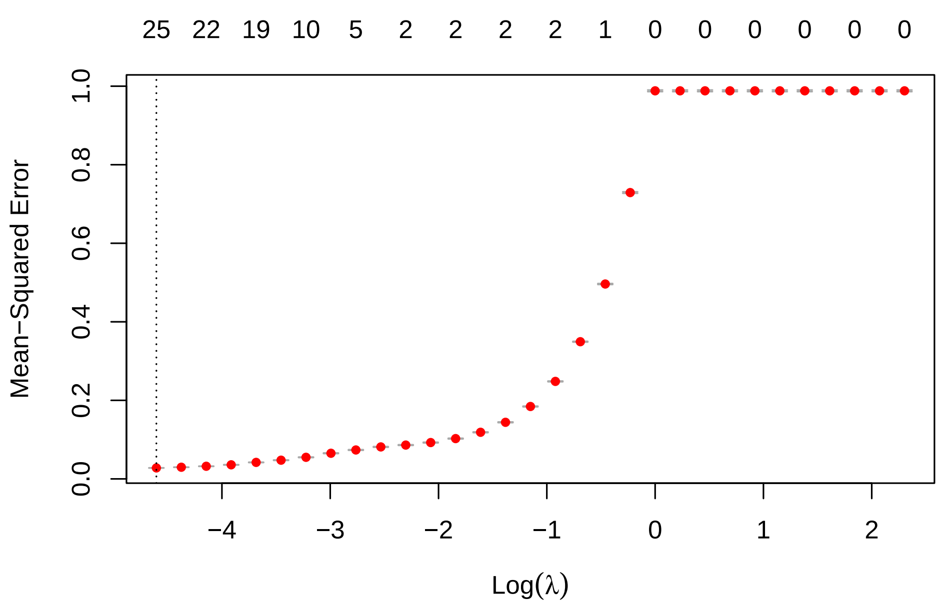


Figure 1: Effect of lambda on MSE

When inspecting the coefficients, it is interesting to see some variables which are known determinants of demand to have shrunk to zero such as temperature. However, this can be explained by the fact that other variables captured such as lagged demand and hour of the day captures these more effectively, resulting in a superior predictive model.

Table 2: Resulting coefficients after shrinkage (part 1)

Variable	Coefficient
TEMPERATURE	.
HOT_DAY	0.0093
HOT_NIGHT	.
COLD_DAY	.
COLD_NIGHT	.
EXTREME_HOT_DAY	.
EXTREME_HOT_NIGHT	.
EXTREME_COLD_DAY	.
EXTREME_COLD_NIGHT	.
PUBLIC_HOLIDAY	0.0018
RAINFALL	.
SOLAR_EXPOSURE	.
HOUR_1	-0.0291
HOUR_2	-0.0283
HOUR_3	.
HOUR_4	0.0152
HOUR_5	0.0563
HOUR_6	0.0817
HOUR_7	0.044
HOUR_8	0.0017
HOUR_9	.
HOUR_10	.
HOUR_11	.
HOUR_12	.
HOUR_13	0.0039
HOUR_14	0.0068
HOUR_15	0.0179
HOUR_16	0.0284
HOUR_17	0.04
HOUR_18	0.0092
HOUR_19	-0.018
HOUR_20	-0.02
HOUR_21	-0.0117
HOUR_22	.
HOUR_23	-0.0088
weekday_2	.
weekday_3	.
weekday_4	.
weekday_5	.
weekday_6	-0.0278

Table 3: Resulting coefficients after shrinkage (part 2)

	Variable	Coefficient
41	weekday_7	-0.0152
42	MONTH_2	.
43	MONTH_3	.
44	MONTH_4	.
45	MONTH_5	.
46	MONTH_6	.
47	MONTH_7	.
48	MONTH_8	.
49	MONTH_9	.
50	MONTH_10	.
51	MONTH_11	.
52	MONTH_12	.
53	demand_lag_1	0.9863
54	demand_lag_2	.
55	demand_lag_3	-0.1714
56	demand_lag_4	.
57	demand_lag_5	.
58	demand_lag_24	0.1351

A quick inspection of the lambda pathway plot reaffirms this, with recent demand being the strongest indicators of future demand. In comparison, the lasso model found that most other coefficients were not as significant, and were therefore shrunk to zero or kept at a very low magnitude.

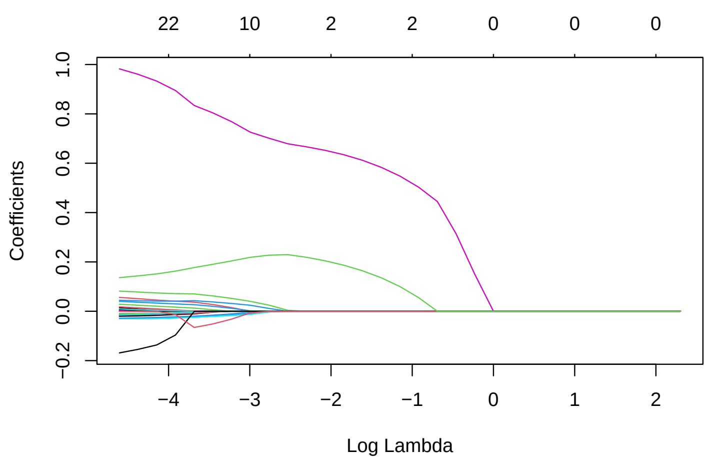


Figure 2: Lambda pathway plot

5.3 Support vector regression (SVR)

Support vector regressions uses support vectors (lines) to help define data points. Similar to lasso regression, the variables first had to be scaled to ensure direct comparisons when separating data points of different classes.

Another consideration when modelling using SVRs is constraints surrounding computational resources. In particular, kernel machines are sensitive to the amount of training data used. Using the full training dataset here resulted in unreasonably long computation times for hyperparamter tuning and large amounts of memory required to store the models. To reduce the impact on computational resources, the training data was cut to the start of **August 2014** onwards.

In addition, the set of features considered in the model was more targeted in SVRs for the same reason. From guidance in the data exploration section, the features used in this model included temperature and the lagged demands established.

Both of these adjustments significantly improved runtime with minimal impacts on the predictive performance of the model.

There were a couple hyperparameters that were considered for SVRs in this project.

- Kernel: algorithm used for pattern analysis. A simple linear kernel was considered, as well as a more complex radial kernel to account for non-linear relationships
- Epsilon: penalisations for errors. The larger the epsilon, the larger the penalisation from errors.

The table below shows the full set of SVR models considered in this project.

Table 4: List of SVR models

Model ID	Epsilon	Kernel
1	0.0	linear
2	0.0	radial
3	0.5	linear
4	0.5	radial
5	1.0	linear
6	1.0	radial

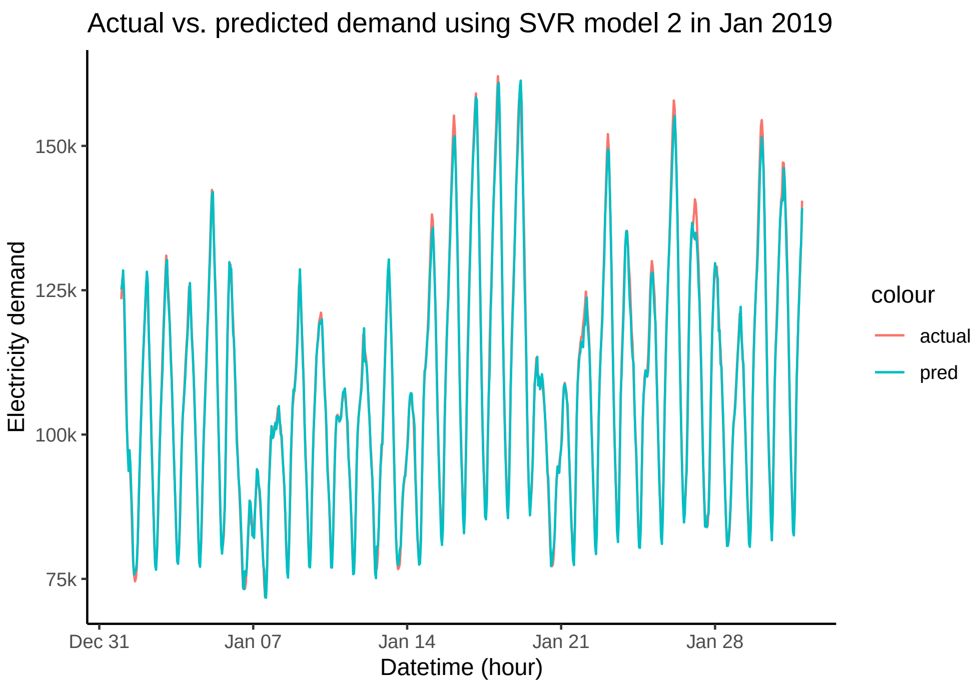
The tuning process involves fitting a model with all the hyperparameter combinations shown above using training data and evaluating their performances using the test data. These metrics would be used as the final SVR model to consider.

Table 5: SVR model performances

model_name	MAPE	MSE	RMSE	MAE	R-Squared
svr_1	1.90%	6,198,630	2,490	1,838	97.13%
svr_2	1.33%	2,952,711	1,718	1,263	98.63%
svr_3	2.34%	7,635,083	2,763	2,223	96.47%
svr_4	2.55%	8,493,121	2,914	2,362	96.07%
svr_5	5.57%	33,413,035	5,780	5,070	84.53%
svr_6	6.37%	45,642,602	6,756	5,819	78.87%

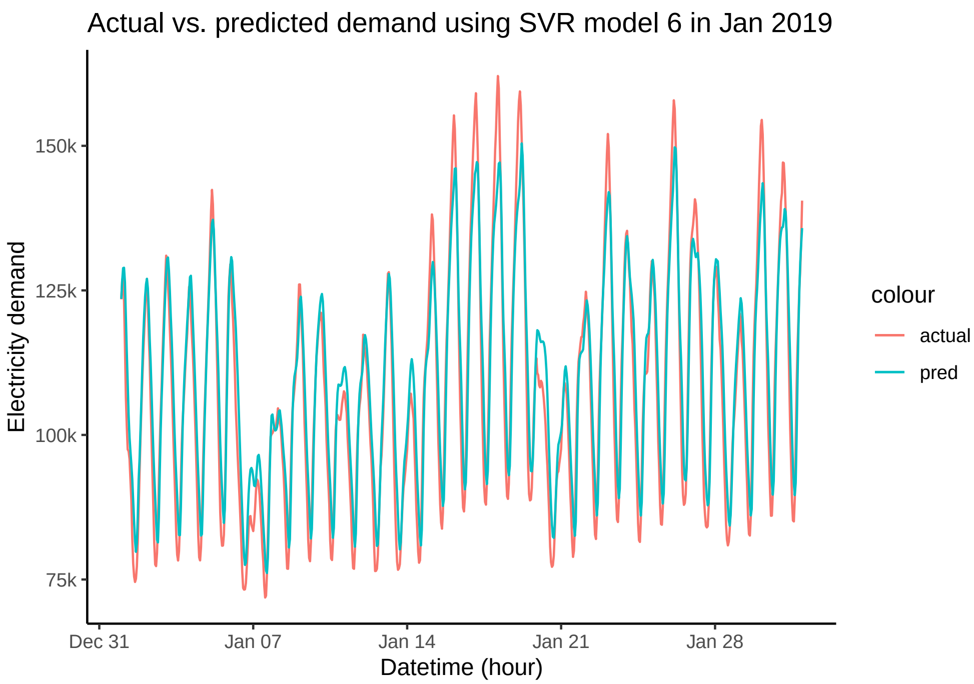
The table above shows that radial SVRs outperform linear SVRs, which can be explained by accounting for non-linear trends. Furthermore, models with lower epsilon values also attributed to better performance. Overall, the second SVR model (radial kernel with epsilon value of zero) performed best and was considered in the overall model selection process.

The charts below compare the fitted versus actual values for the best and worst SVR models from the tuning process:



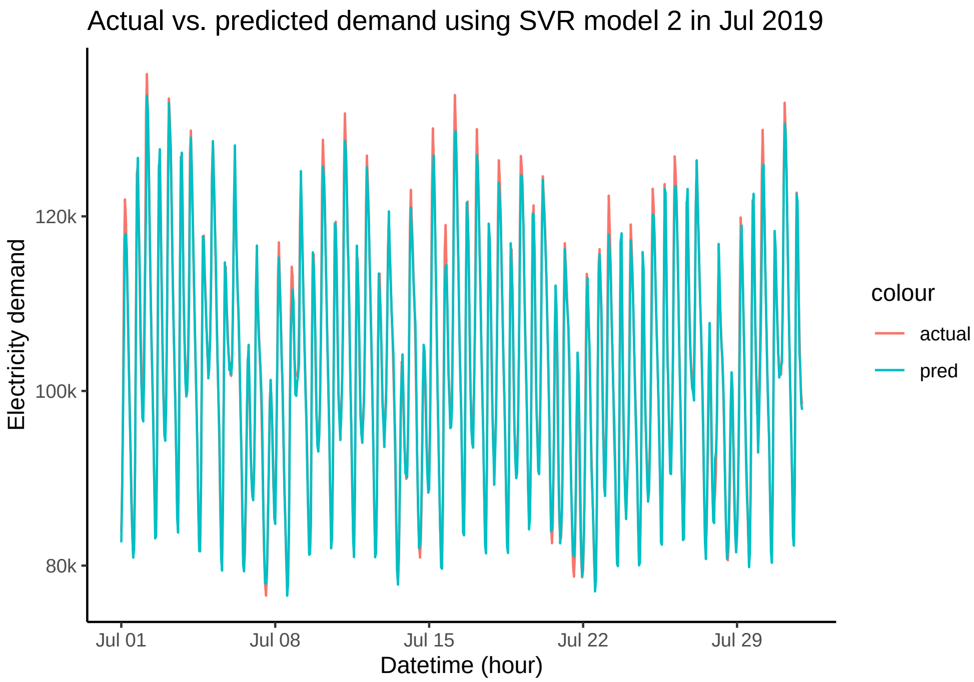
The chart above demonstrates that the best SVR model had strong forecasting performances throughout the month of Jan 2019.

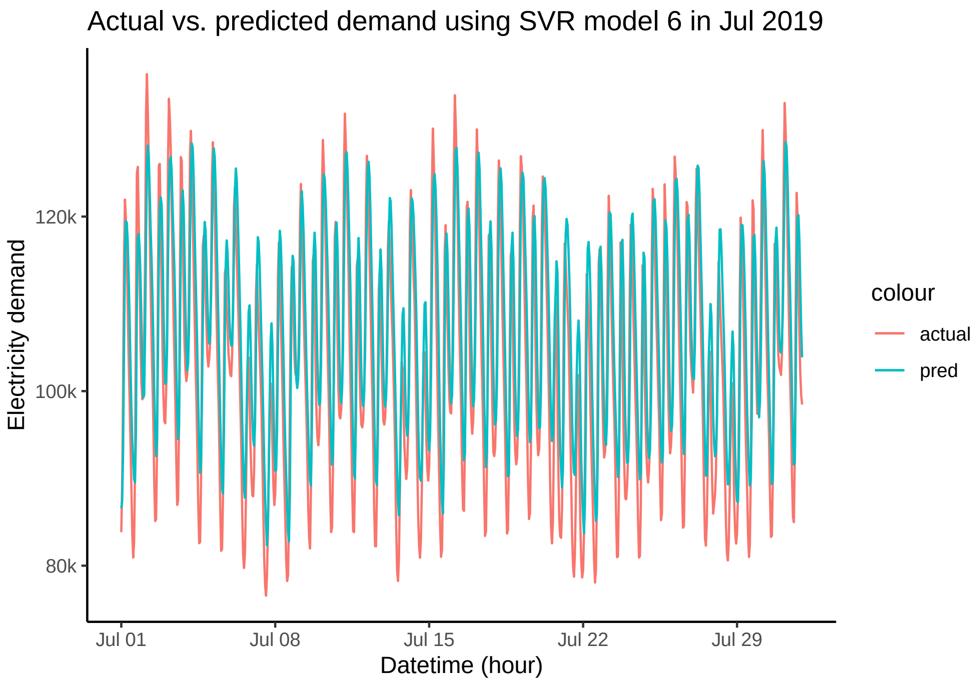
Meanwhile, the chart below, which used an SVR with a linear kernel and an epsilon value of one had difficulties estimating the peaks and troughs of each day, consistently underestimating the magnitudes in either directions.



Actual vs. predicted demand using SVR model 6 in Jan 2019
Electricity demand
150k
125k
100k
75k
Dec 31
Jan 07
Jan 14
Jan 21
Jan 28
Datetime (hour)
colour
actual
pred

A similar trend can be seen in winter as well as seen in the two charts below for July 2019.





5.4 Random forest

Random forest regression is a machine learning algorithm that consists of many decision trees.

Similar to SVR, this section aims to find the best hyperparameters as well as perform some light feature selection. The hyperparamters considered for random forest regression include

- number of trees (100, 200, 300, 400, and 500 trees considered)
- mtries, which specifies the number of columns to randomly select at each level (3, 4, or 5)
- sample rate (ranged from 0.5 to 0.8)
- max depth of tree (ranged from 5 to 10)

The tuning process was performed using the `h2o` package in R. Using this package, a search algorithm can be specified to control the runtime of the tuning process. The search algorithm implemented here included:

- using a random discrete strategy when selecting hyperparameters
- set a max runtime of 600 seconds
- stop the process if the MSE has not improved after 5 models

The first model was tuned using all features available in the dataset. Afterwards, a second model was tuned by selecting variables which scored 0.01 or greater in the variance importance metric, which is a measure of the significance of each considered feature. By considering a smaller subset of features, the model may reduce over-fitting, leading to better predictive performance. The chart below shows the variance importance of the first model.

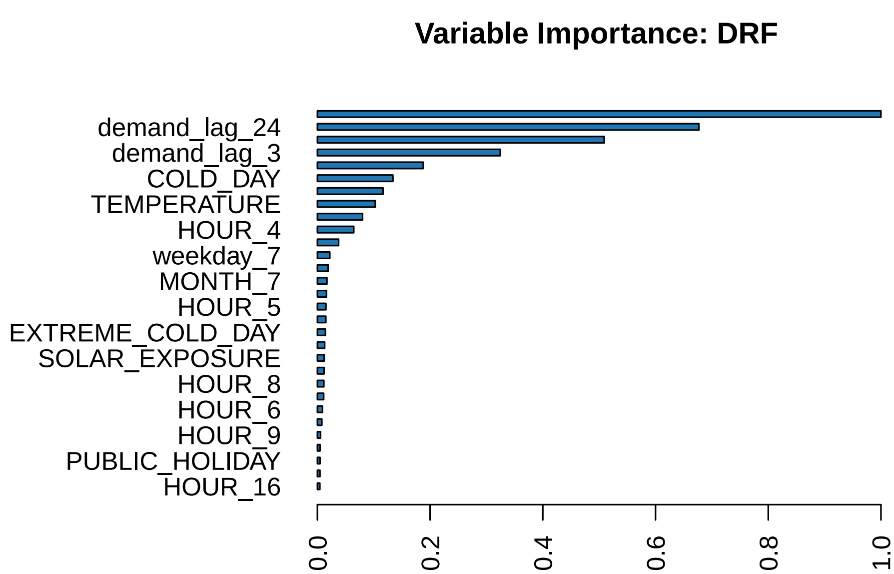


Figure 3: Variance Importance Plot considering all features

The most important measure according to the model was the demand 24 hours prior, which makes intuitive sense as they are highly correlated. Furthermore, temperature and whether it was a cold day also had high importance according to the model.

The second model had a reduction in number of features from 58 to 23 after considering their variance importance measures. The variance importance plot can be seen below.

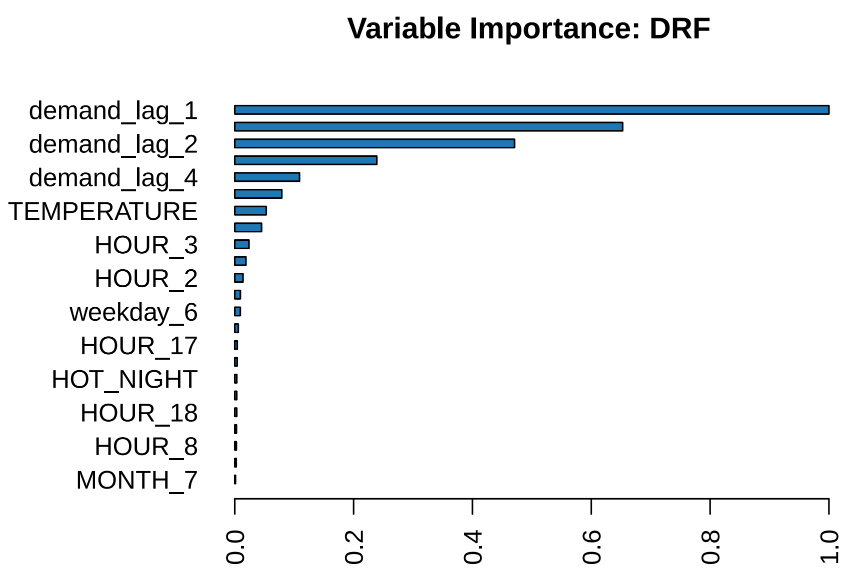


Figure 4: Variance Importance Plot after feature selection

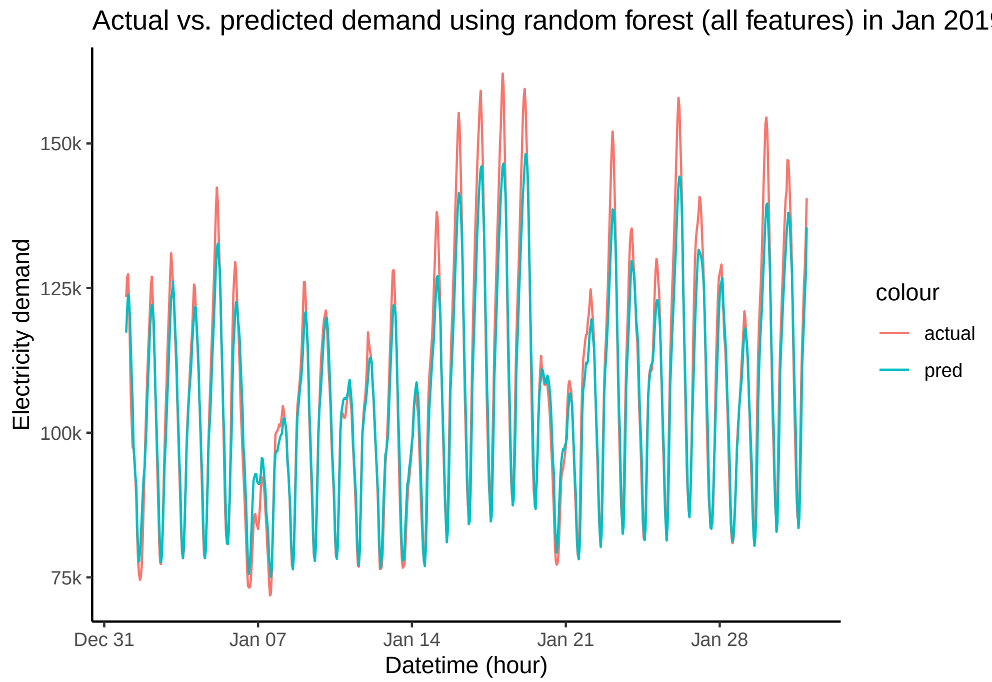
After performing feature selection, the top variables are recent demand values. This is consistent with what was found in the correlation analysis in the data exploration section.

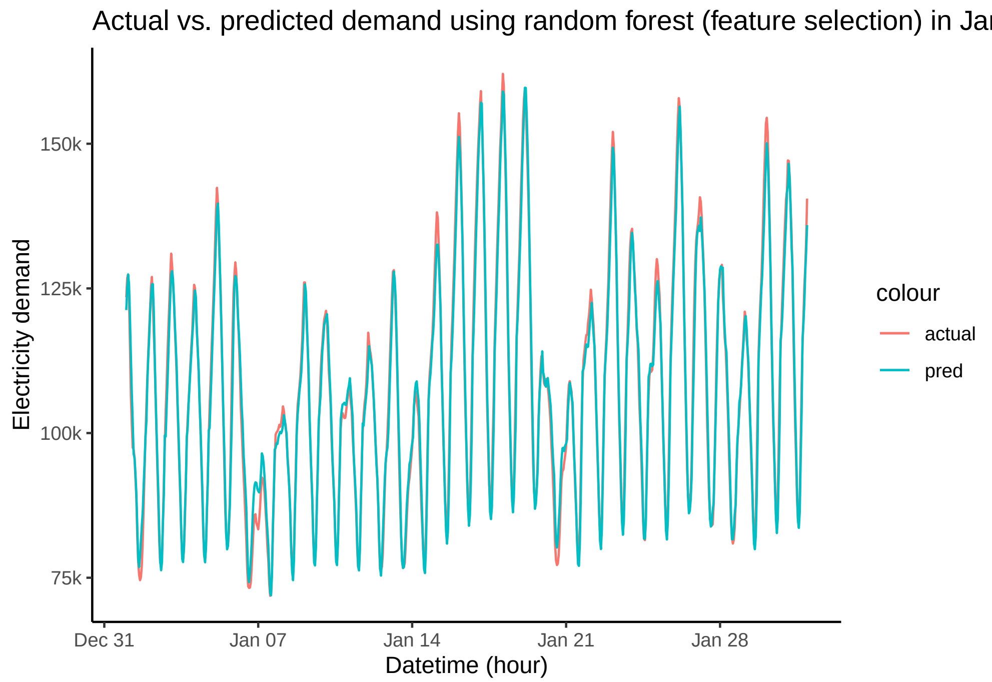
Evaluating the performances of the model with all features compared to the model after feature selection reveals that removing some features did improve the overall performance of the model. However in both cases, there may be some overfitting to the training data as the test set had a relatively significant reduction in performance compared to the training set.

Table 6: Random forest regression model performances

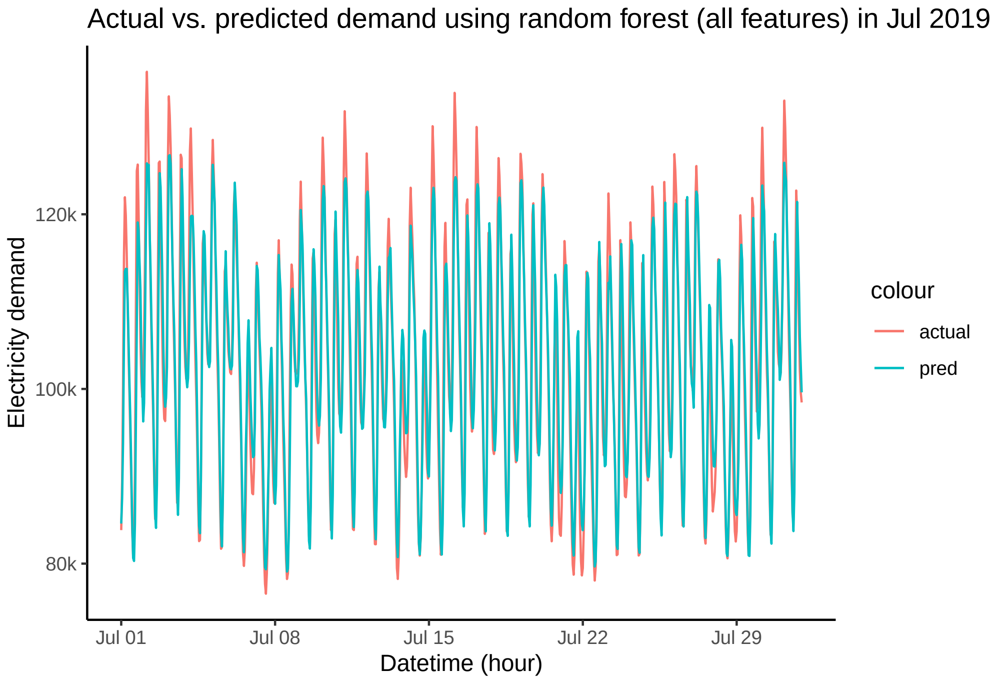
Model name	MAPE	MSE	RMSE	MAE	R-Squared
Training data (all features)	2.88%	13,495,453	3,674	2,796	94.48%
Test data (all features)	3.34%	16,709,373	4,088	3,152	92.26%
Training data (feature selection)	1.69%	4,870,574	2,207	1,658	98.01%
Test data (feature selection)	2.23%	7,868,561	2,805	2,116	96.36%

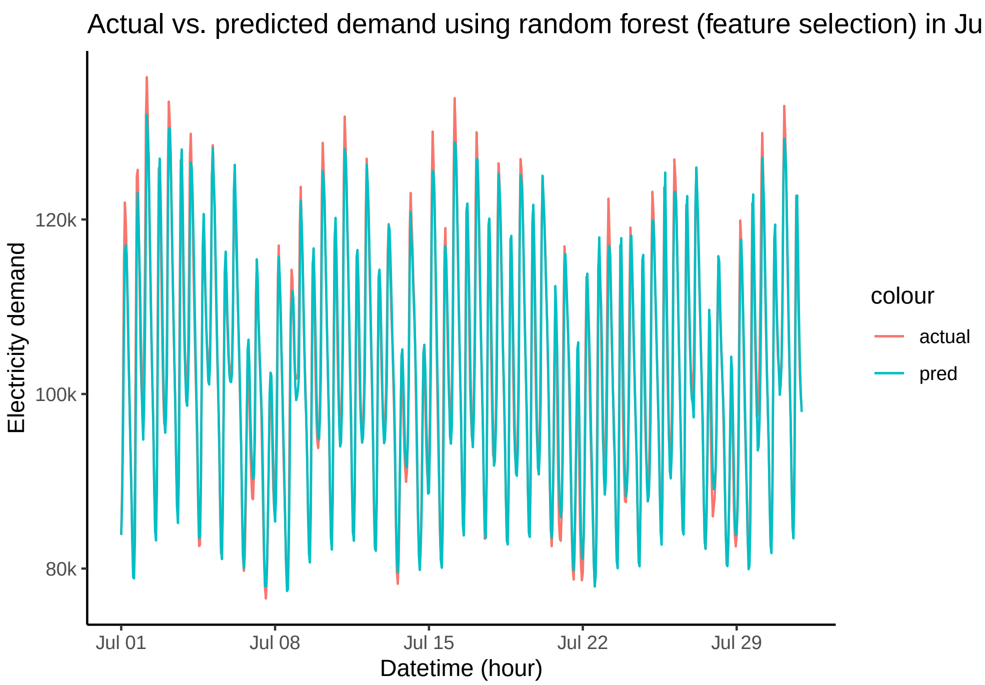
The plots below compare the actual versus fitted values in the test data. Using January 2019 data, the first random forest model had a particularly hard time forecasting the peaks of each day, consistently underestimating the true value. This issue is still present in days with high peak demands in the second random forest model, albeit to a lesser extent.





Again, a similar trend can be found when examining July 2019 fitted versus actual demand, with the feature selection model outperforming the model using all features.





5.5 Final model selection

The final model selection criteria is evaluated by the performance metrics using the holdout set. This is done to simulate a real world forecasting environment to predict unseen data. The AEMO demand forecast was also included as a benchmark to compare our models with.

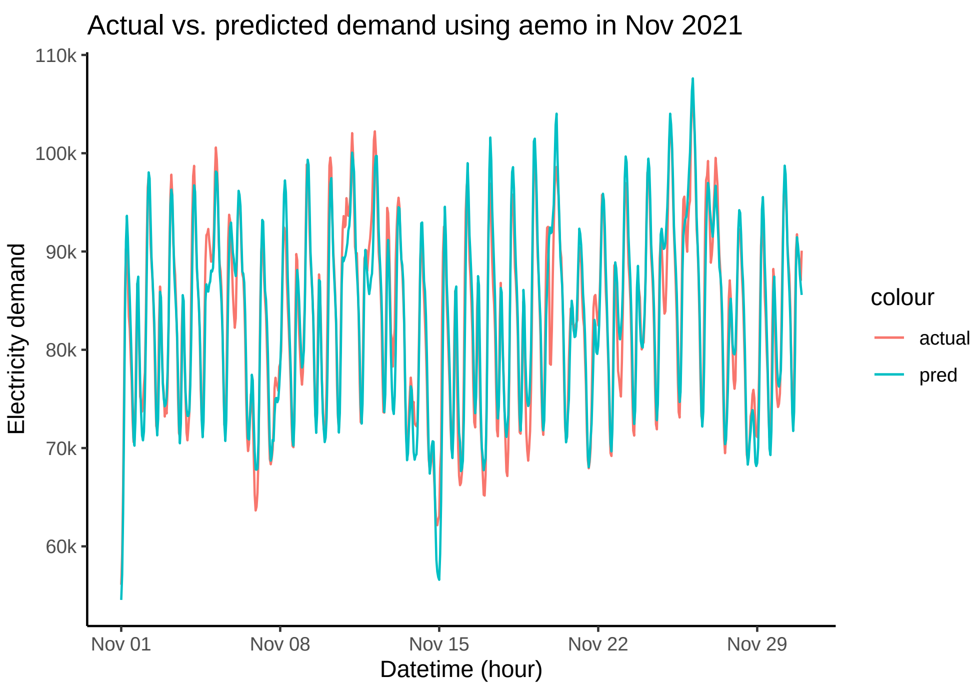
Table 7: Random forest regression model performances

Model name	MAPE	MSE	RMSE	MAE	R-Squared
aemo (benchmark)	3.30%	20,368,620	4,513	3,093	90.93%
lasso	2.38%	9,087,579	3,015	2,158	95.95%
svr	1.90%	5,928,964	2,435	1,684	97.36%
random forest	3.02%	13,631,166	3,692	2,645	93.93%

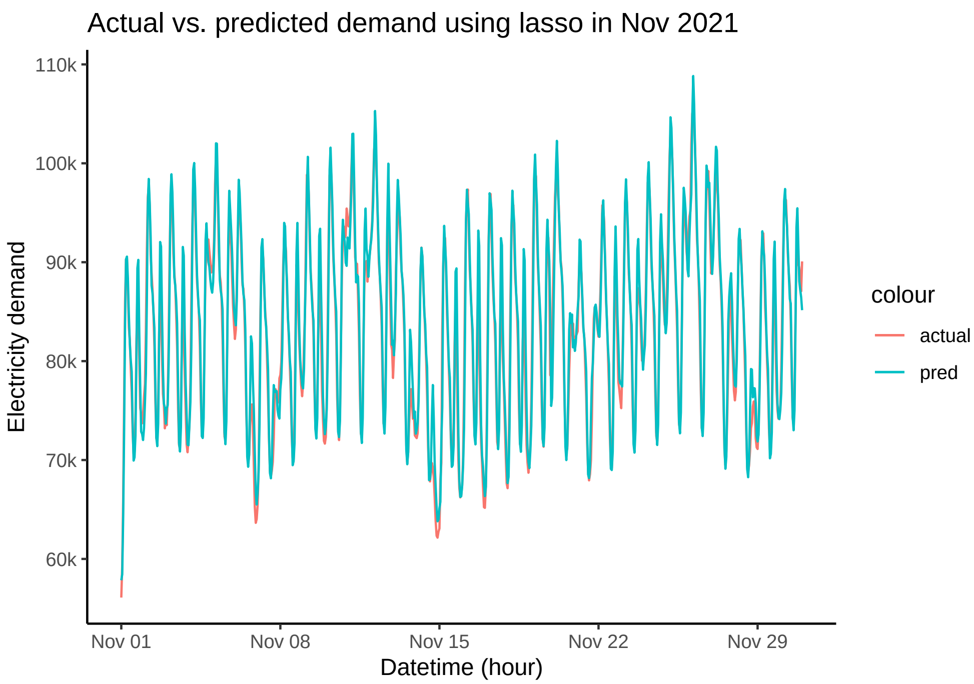
All three models considered in this report managed to out-perform the AEMO benchmark. This can be attributed to the way our models were set up to prioritise predictive performance, while AEMO's model equally considers performance and interpretability.

Interestingly, the random forest model continued to deteriorate in performance as the dataset moved away from the training data, showing some indication of overfitting to the training data. The SVR had the strongest performance overall with over 98% accuracy, and is the final model chosen for our project, being able to outperform aemo by +1.4% accuracy.

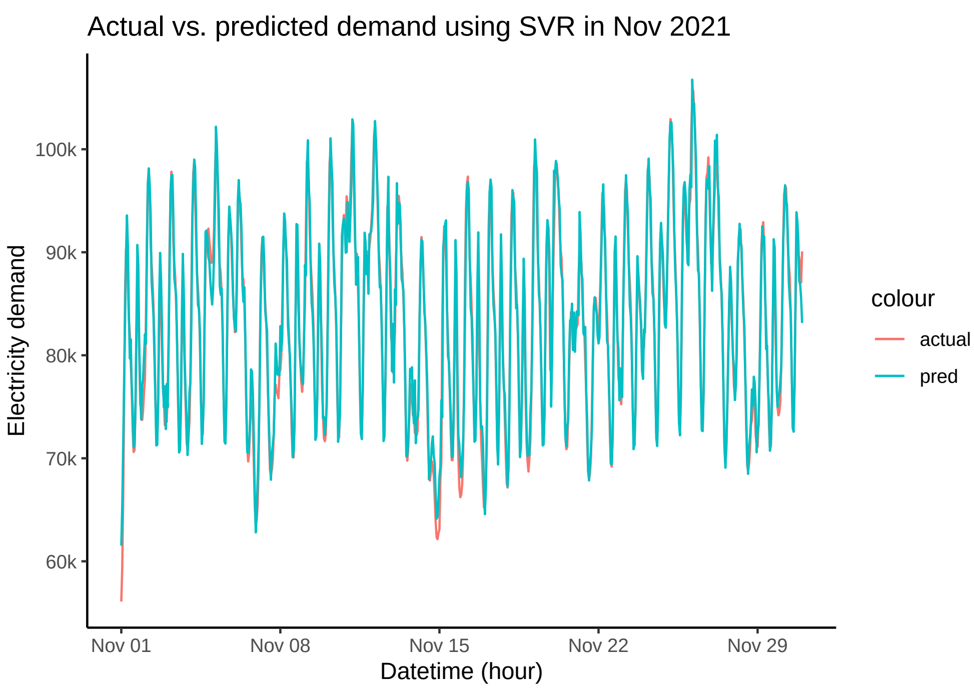
Taking a closer look at one of the months in the holdout set, it can be observed that most of the forecasting error in the aemo model occurs at the peaks and troughs of demand.



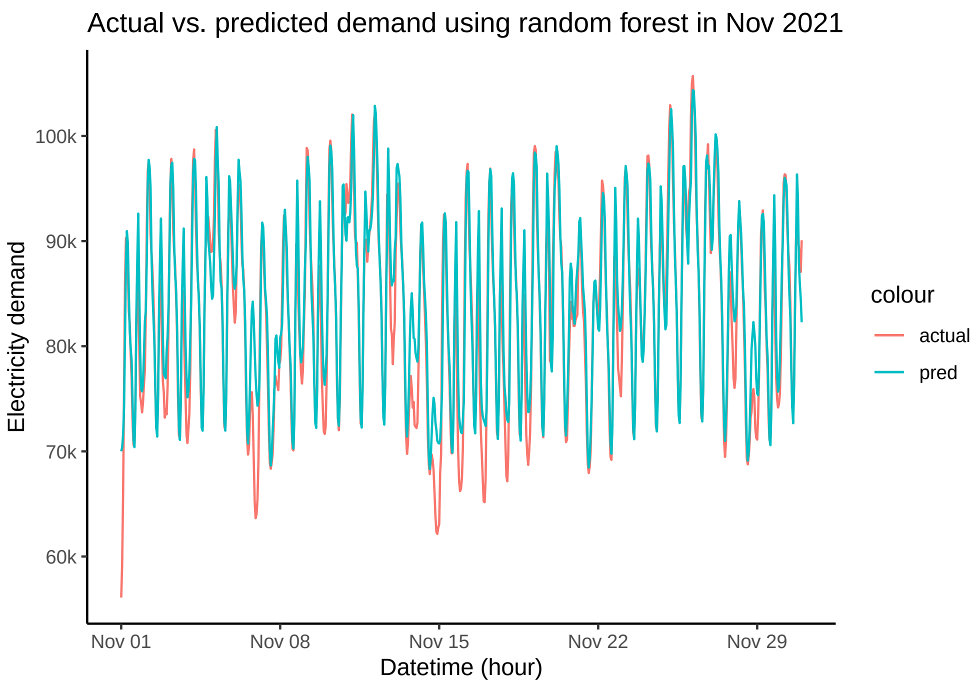
The lasso model fitted in this project shows a similar performance to the aemo model, with slightly more accurate predictions overall.



As reaffirmed by the performance metrics summary, the SVR also looks the most accurate when comparing with actuals in November 2021 as well.



Finally, the random forest model seemed to have the most trouble forecasting the troughs in November 2021, over-estimating energy demands in these periods overall.



6 Discussion

TODO: flesh this section out

Potential improvements to consider:

- More complex models (outside the scope of the team's experience): recurrent neural networks, anything else from literature review
- Greater computational resources: some tuning decisions in SVM and random forest were constrained by the runtime and memory size
- Multivariate analysis: being able to forecast multiple hours ahead at a given time
- Access to more data: hourly weather conditions like rainfall, solar exposure, wind speed, humidity

7 Conclusion

–

References

Appendix