CMT307 Coursework 2 Group Project

Group Number	21
Project Title	Energy Usage Prediction
Supervisor	Deviprabha Surendran
Word Count	4044
	23047708 23101655 23068901 1869948 23108615 23109900 23091913 22121504

Table of Contents

Figures

1. Introduction

In this project, the challenge presented by the "ASHRAE - Great Energy Predictor III" competition is being tackled. The project focus is on developing a tool to evaluate the value of energy efficiency upgrades in energy savings. This involves constructing counterfactual models to compare post-upgrade energy consumption against modelled values for the original building, thereby estimating retrofit savings.

The dataset provided for this project encompasses building metadata, weather data, and energy consumption records. The main challenge is to develop predictive models capable of forecasting energy usage for multiple buildings.

2. Literature Review

In the published literature, it is evident that energy demand and energy load prediction is a popular modelling problem. However, the problem varies widely in scope and scale. For example, Mounter *et al*. [1] implements three machine learning techniques on data from one academic building at Teeside University, while Li *et al*. [2] implements several neural networks to model annual energy consumption per building for 59 residential buildings in China.

Two literature reviews [3] [4] give general conclusions about common machine learning techniques for urban building energy performance and building energy consumption and performance, respectively. In both articles, artificial neural networks (ANNs) and support vector machines (SVMs) were found to be the most used techniques. Seyedzadeh *et al.* [4] note that SVMs surpass ANNs for energy load forecasting and is appropriate for training sets with limited samples. Both reports noted the importance of meteorological data as inputs. Fathi *et al.* [3] identify occupancy data as highly useful in modelling energy consumption of individual buildings. Beyond the top two methods, Gaussian distributed regression and random forests were popular [3][4]. The further focus will be on ANN and SVM.

2.1 Artificial Neural Networks (ANN)

Simple neural models seem to work adequately for energy forecasting. Kadri *et al.* [5] implement an ANN on time series data to forecast daily peak demand and find that 5 previous input days and 10 neurons in the hidden layer produced the lowest mean absolute error. Ebeltagi *et al.* [6] implement ANNs for predicting residential building consumption and determined the optimal structure for the network was two hidden layers of 10 and 16 neurons respectively. Seyedzadeh *et al.* [7] train a recurrent neural network on two energy consumption databases and find that one hidden layer to be best for all cases.

2.2 Support Vector Machines (SVM)

Li *et al*. [2] found support vector machines (SVM) to perform better than 3 ANN models. The structural risk minimisation (SRM) principle of SVM is considered superior to the empirical risk minimisation principle (ERM) of ANNs, as SRM is concerned with minimising the upper bound of generalisation error over mere training error [2]. SVM therefore prevents over-fitting by improving generalisation when compared to ANNs [8]. SVM prediction accuracy is improved significantly by parameter optimisation and by judicious choice of kernel function. The most common choice of kernel found in relevant literature is radial basic function (RBF) [8]-[10]. These common trends inform the basic setup for the model implementation.

Li *et al*. [2] found GRNN to be the best performing model of 3 ANN models in their study (though SVM performed best). Seyedzadeh *et al.* [7] conclude that SVM is a better choice, especially when the data is simple, while neural networks are faster to train and use.

3. Description of the Task and General Data

The dataset provided for this project comprises building metadata, weather data, and energy consumption records. It contains hourly meter readings of energy consumption during the year 2016 for 1449 buildings across 16 different sites worldwide. For the analysis, three files are provided: "train.csv", "building_meta.csv", and "weather_train.csv".

3.1 Train.csv

In the Train.csv dataset, each of the 20,216,100 entries corresponds to a specific building's energy consumption measurement. Each column serves a specific purpose, detailed below:

- Building_id: A unique identifier, linking to additional building information found in the building_meta.csv.
- Meter: Codes meter type, having 0 for electricity, 1 for chilled water, 2 for steam, and 3 for hot water. Not all buildings possess all meter types. Steam energy usage is much higher than other types.

- Timestamp: Denotes the time when the measurement was recorded.
- Meter_reading: Indicates the energy consumption in kWh or its equivalent and is the target variable. It is essential to note that this data may contain measurement errors, which could introduce a baseline level of modelling error..

3.2 Building_meta.csv

In the building_meta.csv dataset, each of the 1,449 entries provides additional information about each building:

- Site_id: Links data to weather_train.csv.
- Building_id: Links data to train.csv.
- Primary_use: Indicates the primary category of activities conducted within the building, based on EnergyStar property type definitions.
- Square_feet: Gives the gross floor area of the building, indicating size and implying energy requirements.
- Year_built: Gives the year of construction, where older buildings may be less energy efficient.
- Floor_count: Specifies the number of floors in the building.

3.3 Weather_train.csv

The weather_train.csv dataset, with 139,773 lines, contains meteorological data recorded by a station close to each site:

- Site_id: Links to building_meta.csv.
- Timestamp: Indicates the specific moment of time when the measurement was taken.
- Air_temperature: Gives the temperature in Celsius at time of measurement.
- Cloud_coverage: Indicates the portion of the sky covered by clouds, measured in oktas.
- Dew_temperature: Denotes the temperature in Celsius at which air becomes saturated with moisture.
- Precip_depth_1_hr: Gives the amount of precipitation recorded, in millimetres, within a one-hour window.
- Sea_level_pressure: Indicates atmospheric pressure measured at sea level, in millibars or hectopascals.
- Wind_direction: Denotes the compass direction from which wind is blowing, in degrees (0-360).
- Wind_speed: Gives the wind speed in metres per second.

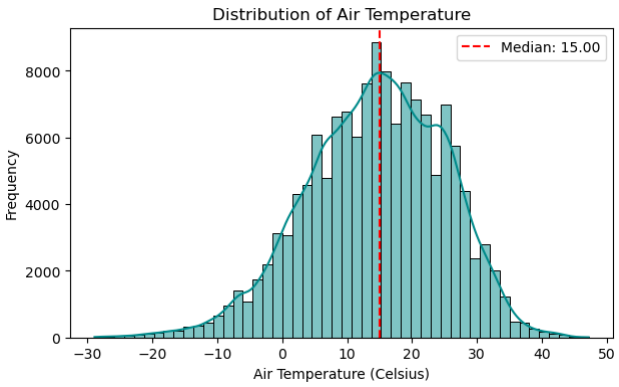

Figure 1 | Distribution of Air Temperature

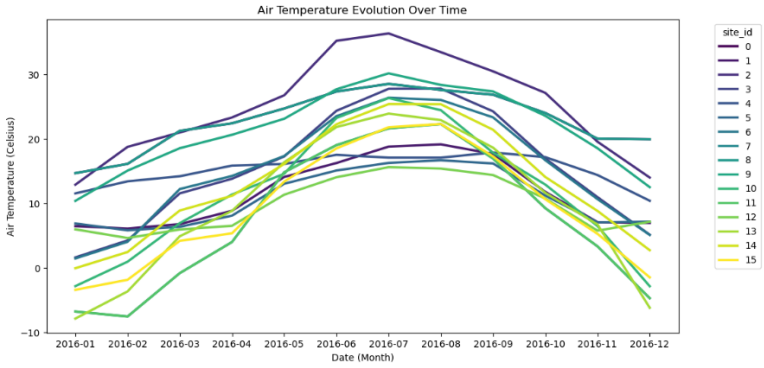

Figure 2 | Air Temperature Evolution Over Time per Site.

Figure 1 shows a histogram of the Air Temperature, showing a distribution that approximates a normal one with median 15°C, while Figure 2 shows for each site how has the average monthly air temperature evolved during the year. All sites show a higher average air temperature during the middle months of the year, suggesting that sites are in the Northern Hemisphere. However, some sites, such as Site 0, show considerably warmer air temperature, suggesting these sites have a different climate.

The weather data provided suffers significantly from missing values. While site_id and timestamp have no missing values, following are presented the number of missing values in each of the other columns, along with the percentage of total data points they represent:

- Air_temperature: 55 (0.04%)
- Cloud_coverage: 69,173 (49.48%)
- Dew_temperature: 113 (0.08%)
- Precip_depth_1_hr: 50,289 (35.98%)

- Sea_level_pressure: 10,618 (7.02%)
- Wind_direction: 6,268 (7.59%)
- Wind_speed: 304 (0.22%)

Moreover, several sites have incomplete weather data, with metrics entirely missing:

- Site 1 lacks data for precip_depth_1_hr and cloud_coverage,
- Site 5 lacks data for precip_depth_1_hr and sea_level_pressure,
- Sites 7 and 11 lack data for cloud_coverage,
- Site 12 lacks data for precip_depth_1_hr, and
- Site 15 is missing all weather data for March 2016

Furthermore, the heatmap of weather data presented in Figure 3 indicates minimal correlation between different parameters, except for measurements related to the same phenomena (e.g., air and dew temperature).

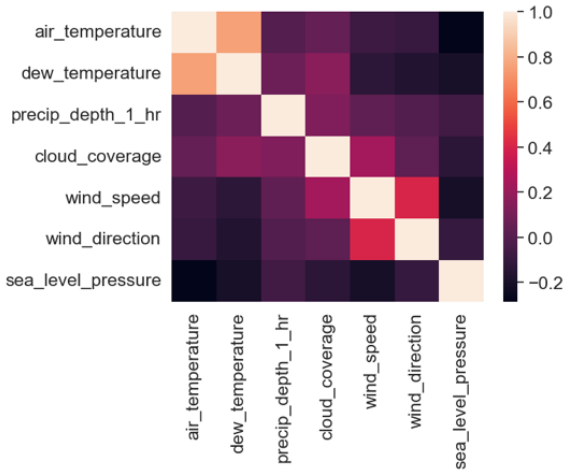

Figure 3 | Heatmap of Weather Parameters

3.4 Combined Data and Exploratory Data Analysis

To enable further exploratory data analysis, the three previously described datasets were combined by using site_id and building_id as keys. Moreover, the whole dataset was split in train and test having 70% in the first group and 30% in the second one. This split shuffled the data, so there is no time sequence in data points for either set.

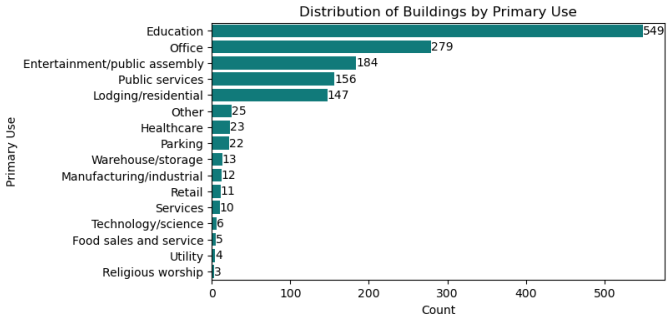

Figure 4 | Distribution of Buildings by Primary Use

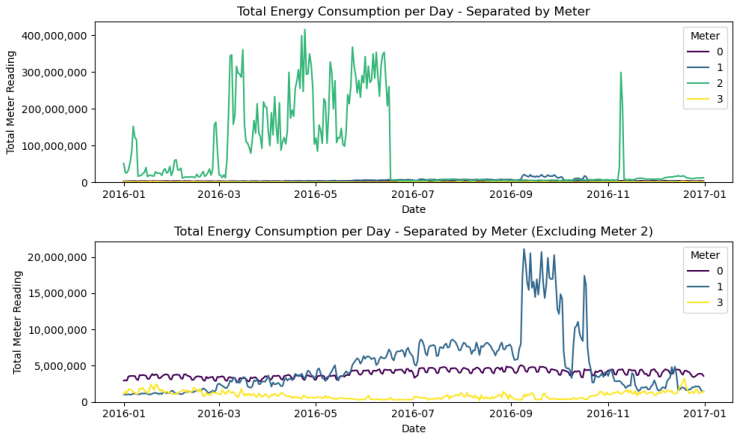

Figure 5 | Energy Consumption Over Time per Meter

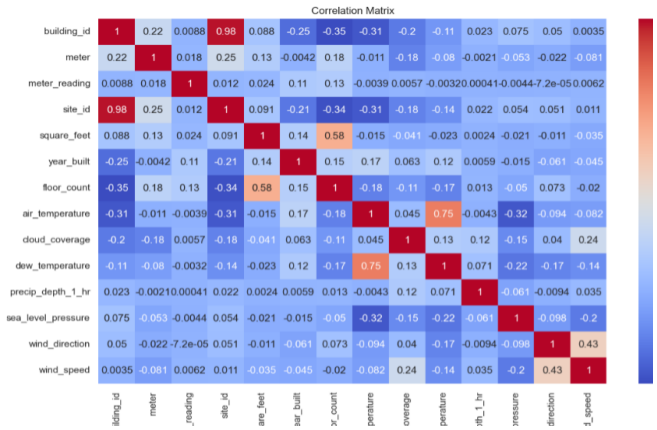

Figure 6 | Correlation Matrix of All Variables.

The analysis began with assessing the primary use of buildings. Out of 1,449 buildings examined (Figure 4), the predominant type is education.

Figure 5 highlights distinct consumption trends by meter type. Meter 2, representing steam, exhibits periodic high and low consumption. In contrast, meters 0 (electricity) and 3 (hot water) remain at a consistent level. Meter 1, chilled water, peaks between September and November when steam demand is lowest.

A correlation matrix for all variables is presented in Figure 6. The high correlation between Site ID and Building ID is due to their role as unique building identifiers. Floor count and square footage are positively correlated, as a taller building is often a larger building. The primary use of the buildings has a strong positive relation to the electricity use.

Energy use correlates with average temperature, though notably shifting around the midpoint of the year, corresponding with a decline in steam consumption (Figure 7). This aligns with reduced steam usage during warmer months. However, the prevalence of steam may complicate modelling energy consumption solely based on temperature, despite an initial relationship.

Analysing daily consumption from a time series standpoint reveals positive autocorrelation between current and previous readings. However, since the final model is based on shuffled data, time series information is deemed less relevant.

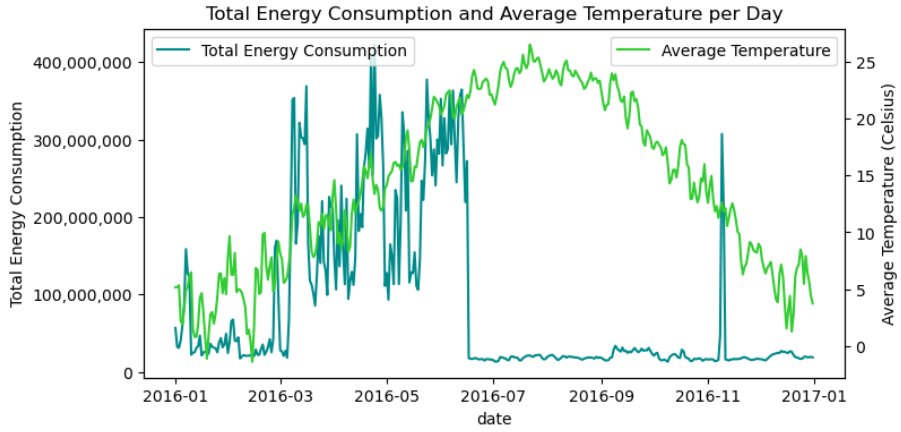

Figure 7 | Total Energy Consumption and Average Temperature per Day.

4. Pre-Processing

Given prevailing trends in the literature, SVM and ANN emerge as promising candidates for this modelling problem. Accordingly, preprocessing and preparation were undertaken to align with the specific requirements of these methods. The output dataset (temp_df) from the data prep stage was split into train-validation-test with an 80%-10%-10% ratio, following data preparation.

4.1 Data Preparation

This pre-processing stage uses the combined dataset, as per section 3.4.

At a first glance, the descriptive statistics showed some of the anomalies highlighted during EDA. There were several features with NaN values, negative values (i.e., negative mm of rain per hour), as well as large variations shown in the four number summaries for most of the features.

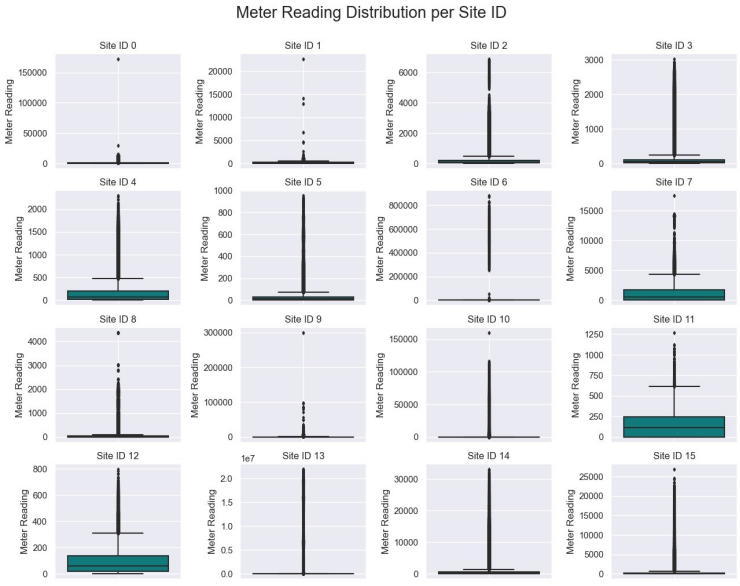

Figure 8 | Energy Consumption Boxplots per Site.

Based on EDA recommendations, the following features were dropped: year_built, floor_count and building_id had very little value as a predictor. Both year_built and floor_count have many missing values, 60% and 83.7% respectively. Data imputation on this many null values would introduce error in the model, so these features were dropped.

Hourly precipitation had values of 0.0 throughout its entire interquartile range. This represents 71% of the total rows; moreover, 19.1% of these were Null and only 9.64% were non-zero values, so this feature was removed from the dataset.

The target variable showed the largest variance between its maximum value (2million+ kWh) and its 3rd quartile (~268 kWh), skewing the data significantly. Looking at the EDA, it became clear that this variation came from site 13, as can be observed in Figure 8. The range of consumption (millions of kWh) for site 13 is significantly larger than other sites. In addition, its air temperature profile displayed irregular patterns, with temperature peaking between 8pm and midnight. Site 13 was labelled "Anonymous" [11], so it was not possible to investigate further, due to lack of geographical information. The rows for site 13 were dropped from the dataset due to these irregularities

From descriptive statistics, it can be observed that there were rows with 0 kWh which seemed unrealistic. Since the consumption is hourly, only those rows with energy consumption of 1kWh or above were kept.

According to "The Building Data Genome Project 2" [11] the units of energy consumption were converted to kWh from its original units; however, an error reported in the Kaggle competition discussion forum [12] found that the electric meters for site 0 were still in kBTU, these rows were converted to kWh during pre-processing.

The target variable was found to have a right skewed distribution. This can lead to unreliable models or invalid results [13]. To address this, a log10 transformation was used to address the right skewness [14].

The hourly consumption profiles per site show irregular patterns for sites 8, 6 and 10. These sites were also anonymous, so information to explain this divergence [11], is not present. In addition, the air-temperature profile looked abnormal for sites 8 and 10, with unusually low or high temperature peaks at unusual times. Therefore, the rows for these sites were removed.

From the site-wise hourly consumption and hourly air-temperature profiles it became apparent that the time zone (i.e., timestamp feature) was likely set to UTC for all sites, since the only typical pattern was observed only for the UK based sites. Information on the names of each site was found in "The Building Data Genome Project 2" [11] and the time zone from for each was taken from NIST [15]. This information was used to map the names to the site IDs and to find their respective time zones during pre-processing.

4.2 Data Pre-Processing

The first step was to address the outliers. Each numerical feature had a different interquartile range (IQR), especially those that we considered important such as air temperature. Winsorisation method for removing outliers was not a good option in this case, based on literature, the cap would need to be very small, and it was proven unable to improve the performance for a neural network model [19]. Because of this, the outliers had to be removed based on the target variable's IQR. Once the dataset was split, the indices of y_train (target) were used to remove outliers in x_train (features) to prevent further data leaks [17].

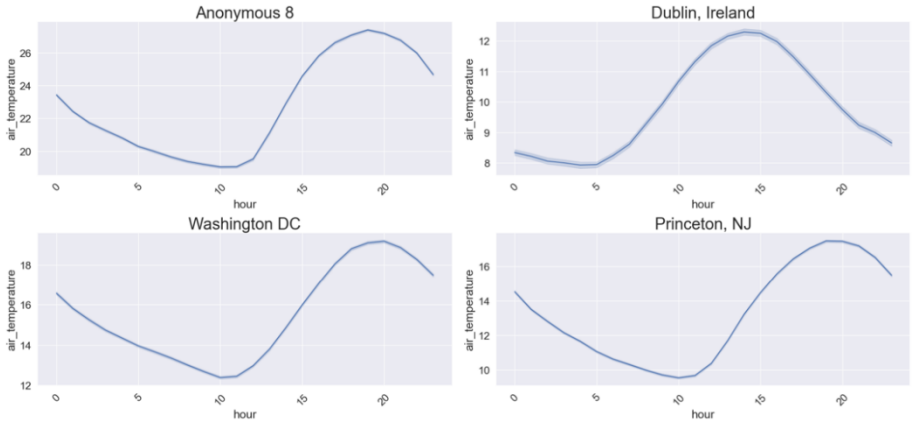

Figure 9 | Hourly Air Temperature (Celsius) Comparison Between Selected Sites.

As previously mentioned, during EDA patterns in energy consumption were observed in relation to time, as observed in Figure 9. (Note: Full grid comparison can be found in pre-processing code). Therefore, the timestamp feature was split into day of the week, month of the year and hour of day. Next, time zones were adjusted from UTC to each site's local time. In addition, meter types were mapped to their names [18]. These adjustments/additions were added separately to the tests and validation sets as well.

The pre-processing pipeline consist of data imputation of missing values, feature encoding and data standardisation. These steps were performed using scikit-learn's Pipeline and Column Transformer classes.

Looking into the proportion of missing values, most of them were found in the weather features. The missing values in air_temperature, dew_temperature and wind_speed added up to less than 2%, so these rows were dropped from the dataset. In addition, based on its very low correlation with the target variable as shown in section 3.3 (see EDA code), further features were dropped: cloud coverage and precipitation depth per hour (precip_depth_1_hr). These variables also had high percentage of missing values (see section 3.3). The feature wind_direction is less meaningful compared to wind_speed or air_temperature unless further geographical or building specific information is introduced [16], so this feature was also dropped. This left only 'sea_level_pressure' with around 7% of values to impute.

The imputer of choice was scikit-learn's IterativeImputer, which is a multivariate data imputation tool that considers all other features to estimate missing values of a given feature [20]. It uses the Iterative MICE approach which fits a regressor (estimator) to predict the missing values [22]. The estimator of choice to go with the Iterative Imputer is Random Forest Regressor, which according to literature, outperformed other data imputation methods compared [21][24].

Data Standardisation on numerical features was carried out using a mix of RobustScaler and MinMax scaler. The Robust scaler, being resilient to outliers, was used for features which still show moderate presence of outliers and the MinMax scaler for the remaining ones.

Categorical features were encoded using OneHotEncoder class, since it is a bivariate encoder, it avoids misunderstanding the data by implying an ordinal relationship between categories [23]. Also, since OneHotEncoder is interpreted as a binary vector it makes it easier for the models to learn [23].

In the pipeline, the parameters for data standardisation and imputation were estimated using the training set and these same parameters were used to transform the validation and test sets accordingly, to prevent data leakage.

5. Model Experimentation

The team initially chose to implement 2 models to compare the suggested methods of support vector machine (SVM) and artificial neural network (ANN) from literature. The SVM model immediately faced significant computation challenges and was abandoned in favour of focusing solely on the ANN.

Due to there being no automatic hyperparameter tuning for neural network layers, an experimental course of action needed to be developed. The plan was to execute in several rounds. The first round would explore activation functions. The following activation functions were tested: ReLU, ELU, SeLU, Sigmoid, and Tanh. The final two were the suggestions from the literature.

All models were built using a Keras sequential model and trained using the Adam optimizer with a loss of mean squared error (MSE). Each model was run for 10 epochs with callback to understand which model showed promise before the best would proceed to more fine-tuning. The models also tracked root mean squared error (RMSE) as a performance metric. The model was fit with batch size 512 to speed up an enhance generalisation. This format was applied to all subsequent rounds of testing.

Since the network structure had not been tested yet, the format of the model was a simple 57 node input layer, a 10 node hidden layer, a 16 node hidden layer and a 1 node output. Table 1 shows the results of the activation function tests.

Table 1: Activation Function Test Results.

Activation Function	MSE	RMSE
ReLU	0.1350	0.3674
ELU	0.1299	0.3604
SeLU	0.1324	0.3639
Sigmoid	1.4401	1.2000
Tanh	1.4353	1.1980

As evidenced by the lowest MSE and RMSE, ELU was the best-performing activation function.

Next came a test for number of nodes in the hidden layers. The initial model followed the structure proposed by literature: an input layer of 57 nodes to match the number of columns, a hidden layer of 10 nodes, a hidden layer of 16 nodes, an output to a single node to deliver the estimated meter reading. Experimentation was only performed on hidden layer parameters.

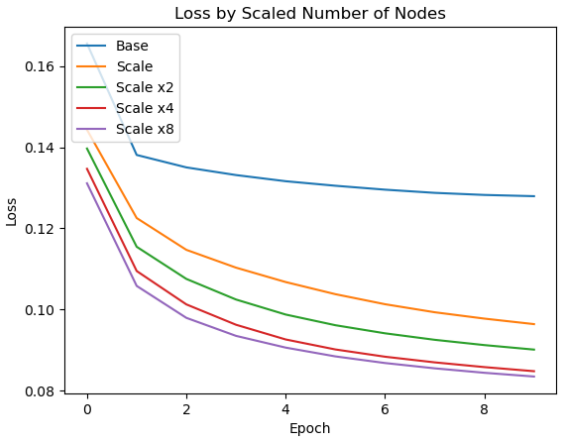

Figure 10 | Loss over Epochs for Number of Nodes.

The options tested were as follows: the initial structure (10-16), a structure 3.5 times scaled to match the number of inputs compared to the literature (35-57), two times the scale structure (70–114), four times scale structure (140-228), and an eight times scale structure (280-456).

After all models were run, it was found that the more nodes the more improvement there was. Although, as the models grew the improvements diminished. This is evidenced by the Scale x8 model being only marginally better than the Scale x4 as seen in Figure 10.

It was decided that the Scale x4 model would proceed to further tuning. It performed only minimally worse than Scale x8, with a loss of 0.0852

MSE on validation verses 0.0847 MSE on validation for Scale x8 but ran in much less time. Scale x4 fit 10 epochs in 9.7 minutes while Scale x8 fit 10 epochs in 15.3 minutes.

The original intent was to take the best performing model to proceed but the team weighted that the time saved by training a minimally worse model faster would give flexibility to explore more tuning options quicker.

Model Name	Training Time	Batch Size	Batch Normalisation	Dropout	Kernel Initialisation	Validation Loss	Validation RMSE
base512_Heunif	9m27s	512	No	No	HE Uniform	0.0826	0.2875
base_512	9m20s	512	No	No	No	0.0840	0.2898
base512_HEnorm	9m37s	512	No	No	HE Normal	0.0841	0.2900
base512_Glorotnorm	9m24s	512	No	No	Glorot Normal	0.0848	0.2912
base512_Glorotunif	9m40s	512	No	No	Glorot Uniform	0.0857	0.2927
base_bn	14m53s	512	Yes	No	No	0.0872	0.2952
base512_d01	10m23s	512	No	0.1	No	0.1009	0.3176
base512_d02	11m56s	512	No	0.2	No	0.1103	0.3322
base512_d01_bn	41m44s	512	Yes	0.1	No	0.1318	0.3630
base512_d02_bn	37m50s	512	Yes	0.2	No	0.1324	0.3639

In the second round of tuning the neural network, batch normalisation, dropout rates and different kernel initialisation options were tested on the base model from round one, the results of which are found in Table 2 above.

Batch normalisation

base_bn: Batch normalisation was added between each layer of the neural network. Validation loss and RMSE suffered compared to the base model, with respective increases of 0.0032 and 0.0054. Training time of the model increased by 62%.

Dropout Rates

base512_d01, base512_d01: Dropout rates, in which 10% or 20% of the data was dropped between each internal hidden layer to prevent overfitting were tested. While training times were largely unaffected, validation loss and validation RMSE worsened from the base model by an average of 0.0216 and 0.0350, respectively.

base512_d01_bn, base512_d01_bn: This effect was increased when batch normalisation was also added to the model, with a sharp increase in training times and the two worst validation losses and RMSE values recorded in testing.

Kernel Initialisation

base512_HEnorm, base512_Glorotnorm: These initialisers draw samples from truncated normal distributions based on input layer size (HEnormal) and input and output layer size (GlorotNormal). These models performed very similarly, with training times within 13 seconds of each other and a difference of 0.0007 and 0.0012 in validation loss and RMSE. Both models performed worse than the base model.

base512_Heunif, base512_Glorotunif: These initialisers draw samples from uniform distributions based on input layer size (HEUniform) and input and output layer size (GlorotUniform). HEUniform performed better than GlorotUniform, the only model to decrease validation loss and RMSE compared to the base model from round one testing.

Final Model

Based on this testing, the final model was chosen to be the base model from round one testing with added HE-Uniform kernel initialisation. This model was run with all the same options as previous testing, with an increased epoch count of 100. The model had a validation loss and RMSE of 0.0645 and 0.2541, respectively. When evaluated on the test data, the model scored a RMSE of 0.2547 with a loss of 0.0649.

6. Results

The results of the final model have a MSE of 0.06719, a Mean Absolute Error (MAE) of 0.17833 and the R-squared (R2) of 0.86572. The difference between the MSE and the MAE implies that our model has minimal large errors, and the errors, in general, are relatively small. The difference is caused by the square in the MSE formula, which leads to a heavier penalty for larger errors. The R2 is close to 1, which shows that the different variables are equally well included in the prediction.

The residuals of the prediction are equally spread and centred around 0, as shown in Figure 11.

A closer look at the density graph of the residuals shows that our model is unbiased but predicts better for values around 2 (Figure 12). The mean of the text values is 1.963397, and the predicted value is 1.94951. Therefore, it is a logical consequence that the model predicts values better around the true mean. Examining sites individually shows that all of them perform similarly well. (See analysis in Code Appendix).

The Q-Q-plot (Figure 13) of the residuals shows that our error is normally distributed around the mean, which validates our model further. As it approaches the model's limits, it performs worse, which is expected after many outliers were removed from the training data.

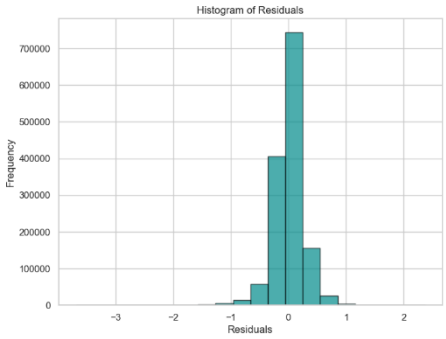

Figure 11 | Histogram of Residuals

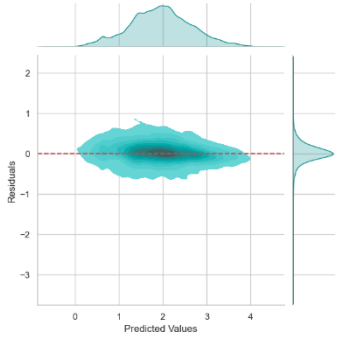

Figure 12 | Density Graph Residuals vs. Predicted Values

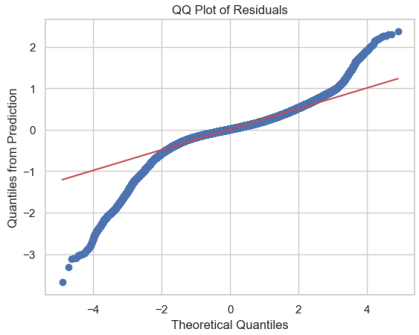

Figure 13 | Q-Q-Plot

7. Conclusion and Future Work

In conclusion, the team was able to produce an artificial neural network to predict the energy consumption of various buildings located across the globe. The model was optimised in choice of activation function, batch normalisation, node dropout, and kernel and kernel initialisation. There were trade-offs between training efficiency and model accuracy. The model performed reasonably well despite its simplicity.

As noted in the literature review, SVM works well with sparse data. Here we have almost too much data for SVM to run well. According to the scikit-learn documentation, the SVR implementation of SVM should only be used for up to 10^4 of rows, while our data was around 14 million. After a failed attempt at SVR using all the training data, tests of SVR models on subsets of the data (10^5 rows) performed reasonably well, although training took hours for each model before any parameter tuning. An ensemble approach of two hundred mini models was the only feasible way to create a SVR based model within reasonable computational time for this project. Training the ensemble with base models (no hyper-parameter tuning) took around eight hours and predictions took over two days to complete and resulted in a disappointing RMSE of 0.378. Potential future avenues to explore in the realm of SVR are ensemble learners for each site or any other ways to compress the data before attempting to model.

In the future it would be prudent to investigate potential non-meteorological input options for predications to determine if there is any improvement from including the additional information. Additionally, future work should consider a more exhaustive search of network parameters, including the impact of additional layers to the model. The team was disappointed to not have the opportunity to compare the support vector machine option with the artificial neural network and would encourage any future work to investigate comparing these two models.

8. References

[1] W. Mounter, C. Ogwumike, H. Dawood and N. Dawood, "Machine Learning and Data Segmentation for Building Energy Use Prediction—A Comparative Study," *Energies,* vol. 14, no. 18, 2021. doi: 10.3390/en14185947

[2] Q. Li, P. Ren and Q. Meng, "Prediction model for annual energy consumption of residential buildings," in *International Conference on Advances in Energy Engineering, Beijing, 2010*, Beijing, 2010. doi: 10.1109/ICAEE.2010.5557576

[3] S. Fathi, R. Srinivasan, A. Fenner and S. Fathi, "Machine learning applications in urban building energy performance forecasting: A systematic review," *Renewable and Sustainable Energy Reviews,* vol. 133, 2020. doi: 10.1016/j.rser.2020.110287

[4] S. Seyedzadeh, F. P. Rahimian, I. Glesk and M. Roper, "Machine learning for estimation of building energy consumption and performance: a review," *Visualization in Engineering,* vol. 6, no. 1, 2018. doi: 10.1186/s40327-018-0064-7

[5] A. Kadri and F. Mohammadi, "ANN Daily Peak Forecast for Peak Demand Charges Management," *IEEE Canadian Conference on Electrical and Computer Engineering (CCECE),* pp. 1-5, 2020. doi: 10.1109/CCECE47787.2020.9255778

[6] E. Elbeltagi and H. Wefki, "Predicting energy consumption for residential buildings using ANN through parametric modeling," *Energy Reports,* vol. 7, pp. 2534-2545, 2021. doi: 10.1016/j.egyr.2021.04.053

[7] S. Seyedzadeh, F. P. Rahimian and I. Glesk, "Tuning machine learning models for prediction of building energy loads," *Sustainable Cities and Society,* 47, 2019. doi: 10.1016/j.scs.2019.101484

[8] H. C. Jung, J. S. Kim and H. Heo, "Prediction of building energy consumption using an improved real coded genetic algorithm based least squares support vector machine approach," *Energy and Buildings,* vol. 90, pp. 76-84, 2015. doi: 10.1016/j.enbuild.2014.12.029

[9] Z. Ma, C. Ye, H. Li and W. Ma, "Applying support vector machines to predict building energy consumption in China," *Energy Procedia,* no. 152, pp. 780-786, 2018. doi: 10.1016/j.egypro.2018.09.245

[10] M. K. Shapi, N. A. Ramli and L. J. Awalin, "Energy consumption prediction by using machine learning for smart building: Case study in Malaysia," *Developments in the Built Environment,* vol. 5, 2021. doi: 10.1016/j.dibe.2020.100037

[11] C. Miller *et al*., "The Building Data Genome Project 2, energy meter data from the ASHRAE Great Energy Predictor III competition." *Sci Data* 7, 368, pp. 2, October 2020.

[12] S. Dane. "Correction to units for site 0 electric meter." Kaggle. https://www.kaggle.com/c/ashrae-energy-prediction/discussion/119261 (accessed 29 March 2024).

[13] K.S., Htoon, "Log-Transformation: Purpose and Interpretation," https://medium.com/@kyawsawhtoon/log-transformation-purpose-and-interpretation-9444b4b049c9 Medium. (accessed 24 March 2024).

[14] R. M. West, "Best practice in statistics: The use of log transformation." *Annals of Clinical Biochemistry,* vol. 59,3, pp. 162-163, May 2022, doi:10.1177/00045632211050531

[15] "Official U.S. Time." https://time.gov/ National Institute of Standards and Technology (NIST). (accessed 31 March 2024).

[16] S. Song, *et al.,* "Impact of Urban Morphology and Climate on Heating Energy Consumption of Buildings in Severe Cold Regions." *International journal of environmental research and public health* vol. 17,22 8354, pp. 24, November 2020, doi:10.3390/ijerph17228354

[17] Pedregosa *et al.*, "Common pitfalls and recommended practices." https://scikit-learn.org/stable/common_pitfalls.html scikit-learn: Machine Learning in Python. (accessed 31 March 2024).

[18] ASHRAE Great Energy Predictor III, https://www.kaggle.com/c/ashrae-energy-prediction/data (accessed 16 March 2024).

[19] S. Sharma, S. Chatterjee, "Winsorization for Robust Bayesian Neural Networks." *Entropy (Basel, Switzerland)* vol. 23,11 1546, pp. 36, Nov. 2021, doi:10.3390/e23111546

[20] Pedregosa *et al.* "Multivariate feature Imputation." https://scikit-learn.org/stable/modules/impute.html scikit-learn: Machine Learning In Python. (accessed: 21 March 2024).

[21] M.J. Azur, *et al.,* "Multiple imputation by chained equations: what is it and how does it work?" *International journal of methods in psychiatric research*, vol. 20,1, pp. 41-42, 2011, doi:10.1002/mpr.329

[22] Pedregosa *et al.* "Imputing missing values with variants of Iterative Imputer." https://scikit-learn.org/stable/auto_examples/impute/plot_iterative_imputer_variants_comparison.html scikit-learn: Machine Learning in Python. (accessed: 21 March 2024).

[23] S. Crêteur. "One-Hot-Encoding vs Integer Encoding." https://medium.com/geekculture/machine-learning-one-hot-encoding-vs-integer-encoding-f180eb831cf1 Medium. (accessed: 24 March 2024).

[24] J. Sebastian, A. Arndt, B. Felix. "A Benchmark for Data Imputation Methods." *Frontiers in Big Data*, vol. 4, pp. 14-15, July 2021, doi: 10.3389/fdata.2021.693674