# ACS341 Assignment: Predicting Household Energy Consumption

## *Task 1: Data Cleaning*

The first step that was taken in cleaning the data was eliminating unnecessary input features. Since the focus of this assignment is to predict a household's energy consumption, it makes sense to only choose the features of a household which draw power, and climate variables that have an impact on the usage of household appliances as the input variables for the model. Therefore the following features were left out and will not be used in our model as they do not contribute to a more accurate prediction of the energy consumption:

- Radon level: Radon is a harmful type of gas that appears in the atmosphere due to the decay of uranium in soil. Radon then moves through the air and goes into houses through holes or gaps found in the home's exterior. Radon then gradually builds up inside the house over time [1]. Therefore the levels of radon have no relevance when predicting a household's energy consumption.
- Visibility: Has no effect on energy consumption in any way.
- Pressure: Has little to no effect on energy consumption.
- Wind bearing: Wind speed does have an effect on energy consumption. However, the direction in which the wind is blowing at has no relevance.

The next step was handling missing/infinite/non-numeric values in the data set. This was done by going through each column, identifying the anomalous data points, and replacing them with the mean of the corresponding column.

The following step was handling outliers in the data set. This was done by producing scatter graphs of the data points for each feature/column in the data set. After the outliers have been identified, they would get replaced by the corresponding column's mean.

Handling missing data and outliers is necessary as the model can't be built when there are values that are non-numeric.

Finally the data was normalized (scaled and centred) using the standardization method. This was done by computing the means and standard deviations of each column and storing them in their own respective array. A nested for loop was then used to access every single data point in the array in-order to update it by subtracting the average value of the data point's corresponding column from the data point itself, and then dividing by the corresponding column's standard deviation. It is essential to normalize the data set since normalization prevents certain features from exerting dominance over the model, which would then lead to the model being biased.

## *Task 2: Linear Regression Model*

Figure 1 shows the linear regression model built using the processed data from task 1.

We can check which predictors have collinearity by determining the Variance Inflation Factors (VIF). VIF is a tool that can help determine the degree of collinearity for a given predictor [2]. Figure 2 is a snippet from the MATLAB command window which shows the results after computing the VIF.

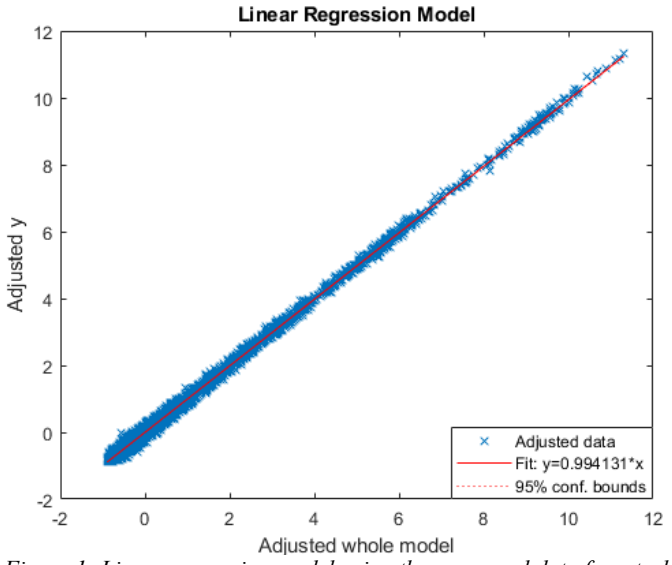


*Figure 1: Linear regression model using the processed data from task 1.*

```
Variance Inflation Factors (VIF):
  Columns 1 through 15

    3.0337    1.8292    1.0683    1.2465    1.4512    1.0424    1.0300    1.0571    1.0007    1.0007    1.0099    1.0004    1.0705    1.0366    1.0178

  Columns 16 through 23

    1.0550    3.0329  121.3582    1.4284  122.5430    1.8183    1.0207    1.1547
```

*Figure 2: VIF of the predictors used for the model in Figure 1.*

Typically, VIF values greater than 5 indicate that the level of collinearity is significant. However, since the norm for these predictors seems to lie between 1 – 2, any predictor exceeding this range will be removed. Therefore, the 1st, 17th, 18th, and 20th input variables will be removed. After removing all of these predictors, we can observe our final linear regression model in Figure 3.

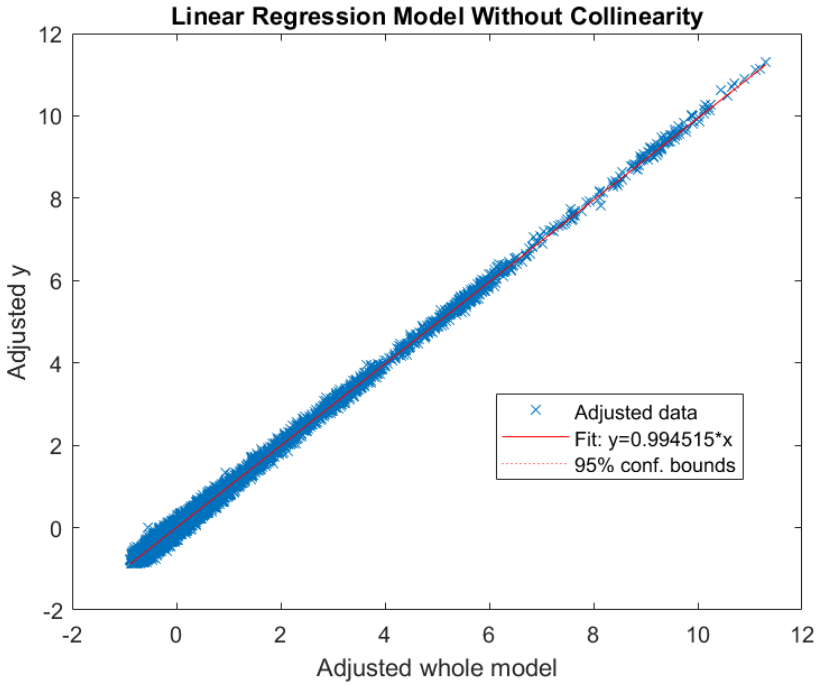


*Figure 3: Linear regression model after removing collinear predictors.*

It can be observed that both models are very similar, this can be explained by one or more input features having a very strong relationship with the output. Figure 4 shows a snippet of the performance metrics of the model in Figure 3.

Estimated Coefficients:

| | Estimate | SE | tStat | pValue |
|---|---|---|---|---|
| **(Intercept)** | -2.6114e-06 | 0.00035972 | -0.0072594 | 0.99421 |
| **x1** | 0.99451 | 0.00047119 | 2110.6 | 0 |
| **x2** | 0.0012876 | 0.00036975 | 3.4823 | 0.00049761 |
| **x3** | 0.0010293 | 0.00038374 | 2.6822 | 0.0073154 |
| **x4** | 0.0026442 | 0.00041872 | 6.3149 | 2.7258e-10 |

*Figure 4: Snippet of the performance metrics of the model in Figure 3.*

Here we can see that the first input feature has a t-statistic value of 2110.6. Such a large value indicates that the relationship between this predictor and the output is extremely significant. Therefore, this explains the similarity between the models in Figure 1 and Figure 3.
Tables 1 and 2 show the performance metrics of the models in Figures 1 and 3.

*Table 1: Performance metrics of the model before removing collinearity.*

| RMSE | $R^2$ | Adjusted $R^2$ | F-statistic | p-value |
|---|---|---|---|---|
| 0.0807 | 0.993 | 0.993 | 334,000 | 0 |

*Table 2: Performance metrics of the model after removing collinearity.*

| RMSE | $R^2$ | Adjusted $R^2$ | F-statistic | p-value |
|---|---|---|---|---|
| 0.0807 | 0.993 | 0.993 | 405,000 | 0 |

The near perfect $R^2$ value for both models indicate strong capabilities of both models at capturing the variance in the data points. Once again this is due to the predictor with the high t-statistic. Both models have identical performance metrics, however after removing collinearity, the model's F-statistic increases from 334,000 to 405,000. This indicates an increase in the overall significance of the model, which means that the input variables have become more significant in explaining the variation in the output.

## *Task 3: Artificial Neural Network Model*

Figure 5 shows the Artificial Neural Network (ANN) diagram.

Multiple optimisation methods were experimented with, however the Levenberg-Marquardt method was chosen as it had the best performance metrics.

With regards to mitigating overfitting, the number of hidden layers (neurons) were set to 50. The number of neurons control the complexity of the model, which is in this case a moderate value, and therefore prevents the model from being too complex. A very complex model can lead to overfitting, therefore it is essential to choose a sensible number of neurons [3].

The input and output datasets were divided into testing and training sets. When training the model using a subset of the data and then testing its performance on another subset, the model's capabilities of generalizing to unseen data can be evaluated [4]. Therefore this stops overfitting by making sure that the model does not memorise the training data.

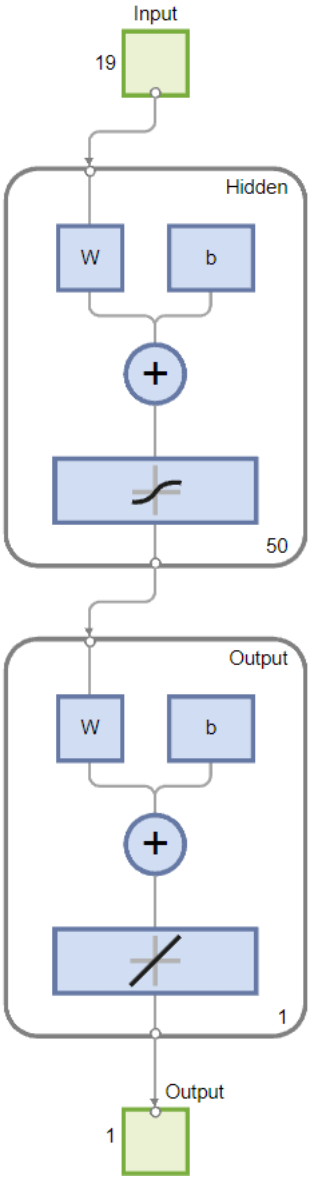


*Figure 5: ANN diagram.*

One issue with using the Levenberg- Marquardt method is that it might get trapped in a local minima [5]. It's not probable that this might happen, however if it does happen, it can lead to non-optimal results. Another issue is that the algorithm has high memory requirements. This is due to the algortihm storing and appending the Hessian matrix [6].

Table 3 shows the performance metrics of the ANN model, and Figure 6 shows the regression plot of the model's output.

*Table 3: Performance metrics of the ANN model.*

| RMSE | $R^2$ |
|---|---|
| 0.080911 | 0.99318 |

The ANN model and the linear regression model have near identical $R^2$ values. This means that both models are equally as good in capturing the variance in the data points. The RMSE is also near identical for both models which indicates that both models have the same level of prediction accuracy. Therefore, it is better to select the linear regression model when predicting a household's energy consumption as its easier to implement, less complex, more robust, and doesn't require as much memory and storage as the ANN model.

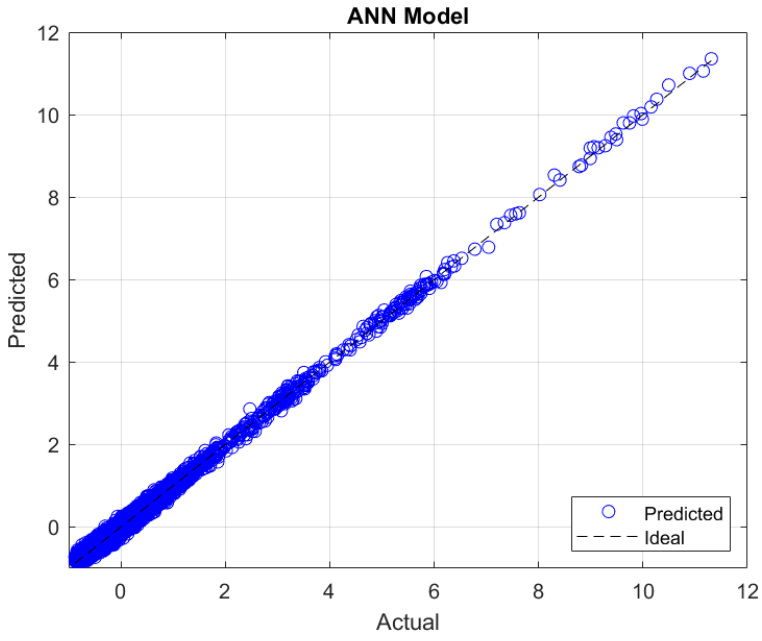


*Figure 6: ANN model regression plot.*

**Summary**
The aim of predicting a household's energy consumption makes it easier to transition to renewable sources of energy, which is in this case wind power. By developing a model that can predict the energy consumption, the client can therefore optimise their power distribution by utilising wind farms, and thus decreasing the reliance on fossil fuels. The use of energy predicting models can therefore lead to a reduced environmental impact, and savings on costs. However, creating such models requires the gathering of data of appliances that consume power within households. This may cause some ethical concerns due to privacy reasons, therefore the data gathering process needs to be secure and anonymous to reduce privacy risks. Machine learning models rely on the volume and quality of the data-sets. Although there is an abundance of energy consumption data, making sure that the data-sets are complete and accurate remains a challenge. The data obtained can vary significantly between each household, therefore identifying and handling these patterns in the data-sets in-order to build a model is challenging. Finally, transitioning to wind energy adds more complexity to the model due to their high variability. The machine learning models are then required to adapt to the changing levels of energy produced by the wind turbines, and then proceed to predict the energy required for households during these conditions.

## *References*

[1] "How does radon get into your home? | US EPA." US EPA. Accessed: Mar. 29, 2024. [Online]. Available: https://www.epa.gov/radon/how-does-radon-get-your-home#:~:text=It%20comes%20from%20the%20natural,where%20it%20can%20build%20up.

[2] "Variance inflation factor (VIF)." Investopedia. Accessed: Mar. 30, 2024. [Online]. Available: https://www.investopedia.com/terms/v/variance-inflation-factor.asp#:~:text=A%20variance%20inflation%20factor%20is,the%20inputs%20into%20the%20model.

[3] G. Malato. "How many neurons for a neural network? | Your Data Teacher." Your Data Teacher. Accessed: Apr. 8, 2024. [Online]. Available: https://www.yourdatateacher.com/2021/05/10/how-many-neurons-for-a-neural-network/#:~:text=A%20small%20number%20could%20produce,neurons%20that%20ensures%20good%20training.

[4] S. Khanna. "A comprehensive guide to train-test-validation split in 2024." Analytics Vidhya. Accessed: Apr. 8, 2024. [Online]. Available: https://www.analyticsvidhya.com/blog/2023/11/train-test-validation-split/#:~:text=The%20train-test-validation%20split%20helps%20assess%20how%20well%20a,to%20generalize%20to%20new%20instances.

[5] "Gradient descent based minimization algorithm that doesn't require initial guess to be near the global optimum." StackExchange. Accessed: Apr. 8, 2024. [Online]. Available: https://stats.stackexchange.com/questions/100045/gradient-descent-based-minimization-algorithm-that-doesnt-require-initial-guess

[6] B. Kumaraswamy. "Levenberg-marquardt algorithm" ScienceDirect, Accessed: Apr. 9, 2024. [Online]. Available: https://www.sciencedirect.com/topics/engineering/levenberg-marquardt-algorithm#:~:text=The%20Levenberg%E2%80%93Marquardt%20Method&text=%E2%88%92%20i%20k%20)%20%2C-,where%20%CE%BC%20k%20is%20a%20positive%20scalar%20called%20the%20damping,given%20by%20equation%20(23.7).