Linear Regression Task

All the relevant libraries were loaded along with the dataset. The following columns were dropped. “House_ID” as it provided no predictive value for the regression model and in some cases leads to overfitting. “Month” was dropped because the same information was already available in “season”. According to Galvão et al, (2009) Variance Inflation Factor (VIF) “indicates the degree by which the variance of b^k is increased (‘inflated’) with respect to the variance that would result if the x variables were uncorrelated.” It is a measure of the amount of multicollinearity in regression analysis. For this reason, “Bill_Amount” was dropped as it displayed a score of 15,7. I also dropped “Daily_Average_Consumption_KWh” as it would end up leaking data about the monthly average, leading to overfitting. When included we saw RMSE around 61 and an average r^2 of around 95, however this was a result of overfitting and likely would cause the model to not perform well on data not in the training set.

The first values of the dataset were checked, along with the dtypes and number of non-null values. This was to get a feel for the dataset and understand how much data was missing. Later, I checked % of nan values in the dataset. Most of the data had at most 5% missing. Only “Type_of_heating” is missing a large chunk of 35%. Duplicate values were tested, none were found.

Several misspelt words were present in all the datasets categorical columns. To remedy this, rapidfuzz library was integrated into the project. This allowed us to use a concept called Levenshtein distance, which is the minimum number of single-character edits needed to change a string into another. Thus, we can create a similarity score between the misspelt strings in the dataset and our list of correctly spelt strings. Using this score we can return the closest match (Shuwandy et al, 2020). Using this approach the spelling of all words was corrected. All strings were lowercased for consistency.

I can see that NumOccupants is inputted as datatype object. This information would be beneficial as a number. After correcting spelling all words (particularly the word “five”) was converted to its number equivalent. The columns dtype was changed to int32. All the remaining datatypes were correct.

Outliers were identified and removed, using z-score. Z-scores are statistical models which quantify the distance between data points and the mean of the dataset. A z-score above 3 is treated as an outlier. Instead of dropping outlier instances, I chose to turn them into nan values and impute their values later. This was so that I would have more entries to train my dataset on.

In QLD the minimum size of a property to build a house needs to be 450 m^2 so all properties smaller than that had its value replaced with 450. Its upper bound was set to 5000 to try remove outliers. Negative readings in “Monthly_Consumption_kWh” and “Age_of_Building” are impossible and were replaced with 0 (Brisbane City Council,n.d.).

Multiple graphs, scatterplots and histograms, were used to identify linear relationships and determine attribute skewness and determine any linear relationship with our label. All that was found was that most attributes were bell shaped with minimal skewness. This helped determine which attributes to impute with the mean and which to impute with the median. This gets covered later in the pipeline section.

Correlation matrixes were used to identify multicollinearity amongst attributes. This was paired with VIF scores for each attribute. The results found was that ‘Daily_Average_Consumption_kWh’ and ‘Bill_Amount’ displayed multicollinearity and needed to be removed. Beyond these not many attributes displayed much correlation to the target variable. This was the first indication of a poor dataset. The remaining attributes were within an acceptable range and did not show much correlation.

The dataset was split into 2 sections. With 80% used to train the data and 20% used to test it. This test section is purposefully kept separate from the model so that we can accurately test our results

A custom ratio pipeline was made to create a column detailing the ratio between “NumOccupants” and “Area_sq_ft”. It will tell the model how many occupants are living on a property of a certain size. Another pipeline called “age_bin” was also created to bin houses based on their age, labeling houses were ‘new’, ‘mid’, and ‘old’. This could potentially help the model if determine if older buildings used more electricity or less, ultimately bettering prediction.

A “bell_pipeline” was created to impute values for attributes with no skewness, being ‘Temperature_Average’ and ‘NumOccupants’. Since these attributes have no skewness and not much missing data, maximum of 5% nan values, imputing the mean won’t negatively affect the model. A categorical pipeline was created and applied to "Season", "Tariff_Type", and "Renewable_Energy_Installed". Values were imputed using the most frequent categorical attribute. This is because not much data was missing. These are also nominal attributes, so OneHotEncoder was used to create a sparse matrix, making it easier for regression models to make predictions.

The same thing was done for ordinal attributes using ord_pipeline, being applied to “Insulation_Quality”. Since “Type_of_heating” has 35% missing I rather chose to fill in all unknown values as a new category “unknown” and then apply nominal encoding. Filling in the mean would likely provide garbage readings and could potentially worsen the model. Other alternatives were converting the column to numerical attributes and using KNN to impute values, yet it provided no notable return and merely added complexity.

A default pipeline would be used on all remaining numerical attributes and would apply median to “Area_sq_ft”, due to its right skewness.

Numerical attributes were scaled using StandardScaler(). This centres the data and helps the regression algorithms that assume a standard normal distribution.

In terms of choosing a model, multiple models were tried. Their implementations are shown in the notebook file. However, XGBRegressor is the best choice. XGBRegressor can pick up on non-linear relationships in datasets due to it being an ensemble of decision trees. It can capture complexities in non-linear datasets.

XGBRegressor can pick up on trends which linear regression can’t. Although its results did not differ much from the linear, ridge or lasso regression it is likely that the model’s poor performance stems from the lack of quality data. The similar performance between linear, ridge and lasso indicate there is unlikely to be overfitting issues but rather data quality issues.

References

Galvão, R.K.H., & Araújo, M.C.U. (2009). 3.05 - Variable Selection. In S. D. Brown, R. Tauler, & B. Walczak (Eds.), *Comprehensive chemometrics* (pp. 233-283). Elsevier. https://doi.org/10.1016/B978-044452701-1.00075-2

Shuwandy, M. L., Zaidan, B. B., Zaidan, A. A., Albahri, A. S., Alamoodi, A. H., Albahri, O. S., Alazab, M. (2020). mHealth Authentication Approach Based 3D Touchscreen and Microphone Sensors for Real-Time Remote Healthcare Monitoring System: Comprehensive Review, Open Issues and Methodological Aspects. *Computer Science Review, 38*, 100300. https://doi.org/10.1016/j.cosrev.2020.100300

Brisbane City Council. (n.d.). *New house*. Retrieved May 16, 2025, from https://www.brisbane.qld.gov.au/building-and-planning/getting-started-on-your-project/residential-projects/new-house#:~:text=Maximum%20site%20cover,less%20than%20200%20square%20metres