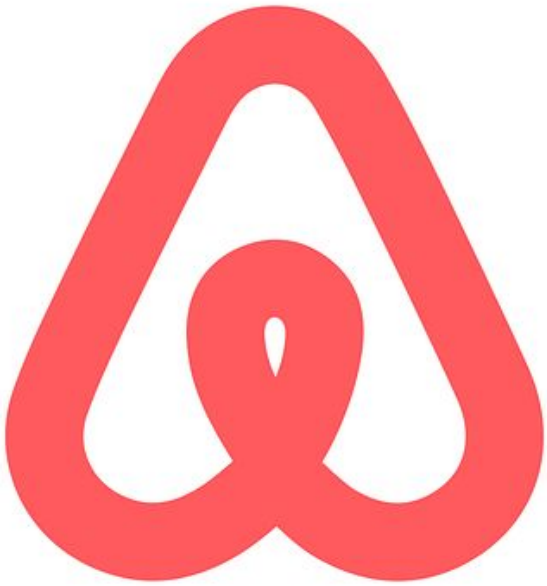

COMP5310 Assignment 2

Sydney AirBnB Analysis

Name: Nahian-Al Hasan
Student ID: 460430331
Unikey: nhas9102

Setup

Airbnb is an online marketplace that connects people who want to rent out their homes with people who are looking for accommodations in that locale. It currently covers more than 81,000 cities and 191 countries worldwide [1]. Since its inception in 2008, AirBnB has seen global success by disrupting the hospitality industry as more and more travelers decide to use AirBnB as their primary accommodation provider. In Australia, no city is more popular than Sydney in terms of the number of AirBnB listings. According to [2], there was a peak of approximately 55,000 AirBnB listings in New South Wales alone at the beginning of 2019. In [3], a report showed that Sydney hosts had an average monthly income of AUD 2,965 in 2019. Therefore, it might not seem like such a bad prospect to rent out one's empty apartment in Sydney. However, it is quite difficult for hosts to assess the true value and the demand of their homes. This project aims to discover insights regarding the AirBnB rental landscape in Sydney and answer the following questions -

(a) *Which factors influence how in-demand an AirBnB accommodation is in Sydney?*

(b) *Can we determine a fairly spot-on daily price for a new accommodation that fits into its specific market environment and competitors in Sydney?*

This analysis is expected to help aspiring Sydney AirBnB hosts to evaluate the value and opportunity costs of listing a new accommodation.

Formally, we have two research statements, and a subset of research statements for the first one:

(a)

(i) Null Hypothesis (H_0) - AirBnB listing prices are not affected by the measure of their distance from the New South Wales shoreline.
Alternate Hypothesis (H_1) - AirBnB listing prices are affected by the measure of their distance from the New South Wales shoreline.

(ii) Null Hypothesis (H_0) - AirBnB listing prices are not affected by the measure of their proximity to transport services.
Alternate Hypothesis (H_1) - AirBnB listing prices are affected by the measure of their proximity to transport services.

(b) Given that we produce two machine learning models - the XGBoost Regressor and the Linear Regressor (benchmark model), and the XGB Regressor has a higher r-squared score across 100 folds of cross validation:

Null Hypothesis (H_0) - The Linear Regressor and XGBRegressor perform similarly and there is no significant difference in prediction power between the two.
Alternate Hypothesis (H_1) - The XGB Regressor is a better predictor than the Linear Regressor

For **(a)** - the statistical significance of selected features (based on domain knowledge) are tested using a two-tailed T-test to determine the p-value of each feature with a 95% confidence interval. Later, we will also peer into feature importances for our XGBRegressor model to find out which features produce the most *gain* and *weight.* For **(b)** - we attempt to build and test two different models for predicting the price of AirBnB listings. Each model is evaluated using the r-square value obtained from predicting the values on a subset (test) of the dataset that is not used for training the model. We then perform a student's T-test across the r-squared scores obtained using 100-fold cross validation of both models, and attempt to statistically prove that one model outperforms the other.

Approach

The Sydney AirBnB dataset consists of multiple categories of data. The data has been divided into three different categories: **(a)** *Reviews* **(b)** *Listings* **(c)** *Calendar Entries*. The data that is being used was last updated on 16th March, 2020 and can be located at the URL, Inside Airbnb. Adding data to the debate [4]. It contains information about all listings and reviews of accommodations in Sydney from 2015 to present, and was compiled by Murray Cox, an independent digital storyteller and technologist [5]. In addition to the AirBnB dataset, I have also extracted all NSW transport service locations from OpenData, which can be found at the URL, Public Transport - Location Facilities and Operators [6].

Validation of Research Questions

For both research questions, some preliminary tasks needed to be performed before we could validate our research questions. In particular, data cleaning and exploratory data analysis (EDA) was performed for data mining and validation. In addition to data cleaning and EDA, a feature engineering study was also

performed based on domain knowledge to create *i) Proximity to transport services* and *ii) Shoreline/Coastline distance* data points for each listing.
It was established that proximity played a key role to cause variance in prices per neighbourhood. *Figures 1 and 2* (**Appendix A**) show that prices vary with distinct patterns within neighbourhoods due to distances from train stations or bus stops. This is also further corroborated by the fact that descriptions for most listings specifically highlight the time to walk or travel to the nearest station or bus stop. As such, we derive the research statement **(a, i)**.
Despite not having been covered in the first stage of the assignment, the *Shoreline/Coastline distance* for each listing was considered for stage 2 of the assignment. This attribute was derived using the *Basemap* python library in conjunction with the latitude and longitude for each listing and calculating the nearest Euclidean distance to the NSW shoreline. This attribute was inspired because of the price heat map generated for all the data points in *Figure 3* (**Appendix B**). One who has lived in Sydney would assume that Sydney is a popular getaway destination because of its myriad of beaches laid across the eastern coastline. As such, AirBnB listings closer to beaches seem to have a higher price range. This discovery leads us to derive the research statement **(a, ii)**.
Finally, the need to justify the performances of our proposed models derives the research statement **(b)**.

Features
In addition to the previous features, we also include the features obtained from our dataset as described in Stage 1. To reiterate, let's gloss over some of the features we are considering for our studies. The listings data set was *143.2 MiB* in size and contained *39,670* entries and *106* features. For our initial study for both research questions, we choose a subset of the original 106 features. Let's imagine we are in the shoes of someone who'd like to offer their home. Fixed features of our property include its rooms, size, and location. We also can decide on how we want to be listed: with a picture or not, how many minimum nights we want a guest to stay, whether we are instantly bookable, how we handle cancellations, etc. But we can neither be a "super host", nor do we have any reviews yet to show - although they can be very important for setting a price. So, let's focus only on features we can influence -
'id', 'space', 'description', 'host_has_profile_pic', 'neighbourhood_cleansed', 'latitude', 'longitude', 'property_type', 'room_type', 'accommodates', 'bathrooms', 'bedrooms', 'bed_type', 'amenities', 'square_feet', 'price', 'cleaning_fee', 'security_deposit', 'extra_people', 'guests_included', 'minimum_nights', 'instant_bookable', 'is_business_travel_ready', and ***'cancellation_policy'***. In addition to these features, we also perform a count based analysis on the occurrences of popular amenities in the description of each listing. Some popular amenities such as TV, Air Conditioning, Balcony, Laptop friendly workspace and Free Parking were added as features to the data set. However, after data cleaning (discussed in stage 1), and limiting prices to only AUD 1000, the dataset has 38903 listings in Sydney.

Proposed Model and Learning Techniques
For both studies, we propose using the Machine Learning Regression model approach for predictions. Regression analysis is a form of predictive modelling technique which investigates the relationship between a dependent (target) and independent variable (s) (predictor). For all our models, our dependent variable is the *price*, and our independent variables are the rest of our features.
(a) For our first study, we fit a **Multi Variable Ordinary Least Squares (OLS) Linear Regressor** on all our data points. We use the *statsmodel* Python API to obtain the regression summary afterwards. The statsmodel API provides us with a few key information for answering our first research question. In addition to calculating the coefficient for each feature, the API produces the T-statistic of each coefficient and tells us how significant each coefficient is. The results are discussed later in the Results section of our analysis.
(b) For our second study, we perform a few more additional steps before building our models. The first step was to perform feature selection to search for the best features for our model. The second step was to build a benchmark model and an improved model using exhaustive parameter tuning and 10 fold cross-validation. Lastly, we compared our models using a T-test on the r-squared scores from 100 fold cross-validation. For this study, our benchmark model was a **Multi Variable OLS Linear Regressor**, and our improved model was the **XGBoost Regressor**.
XGBoost stands for eXtreme Gradient Boosting. XGBoost is an implementation of gradient boosted decision trees designed for speed and performance. It is an implementation of gradient boosting machines created by Tianqi Chen, now with contributions from many developers. It belongs to a broader

collection of tools under the umbrella of the Distributed Machine Learning Community or DMLC [7]. The idea of boosting came out of the idea of whether a weak learner can be modified to become better. A weak hypothesis or weak learner is defined as one whose performance is at least slightly better than random chance. Hypothesis boosting was the idea of filtering observations, leaving those observations that the weak learner can handle and focusing on developing new weak learners to handle the remaining difficult observations.

Feature Selection

For feature selection, we used the **Backward Elimination Method**. We attempted to create a multivariable OLS linear regressor by eliminating features iteratively, and building a model with a reduced set of features in each iteration. The elimination of the features is decided based on the T-statistic and the p-value of each feature based on a 95% confidence interval. This is performed until all the remaining features in the final model have a p-value of less than 0.05, i.e. we can accept all features to have a coefficient higher than 0 with a 95% confidence interval. However, eliminating the low p-value features unfortunately reduced the adjusted r-squared score of the OLS model for each iteration - 0.540, 0.537 and 0.537 respectively. This implied that reducing the features caused our subsequent models to capture lesser variance in the data after each iteration. As such, the feature selection process was dismissed and all the features were kept. In total, there were *34* features selected for our models. Categorical columns were one-hot coded before being fed into the model.

Model Tuning and Parameter Selection

Both models were built using a similar process. The training data is first split into a training and testing set at 80:20 ratio. For model tuning, the models were trained using 10-fold cross validation on the training set, and for parameter selection, we performed an exhaustive search over proposed parameters. Both of these were done in conjunction using *Sklearn's GridSearchCV* api utility. The grid search command performs an exhaustive search over the proposed parameters, and then calculates the r-squared value across 10-folds of cross-validation, and ranks the best model based on the mean r-squared value for every parameter combination. The models are then trained with the best parameters on the entire training set and evaluated on the testing set.

For our OLS model, the parameter search space was - `{'fit_intercept':[True,False], 'normalize':[True,False]}` and the best parameters detected were `{'fit_intercept': False, 'normalize': True}`. And for our XGBRegressor model, our parameter search space was `{'n_estimators': [200, 250], 'learning_rate': [0.01, 0.05, 0.1], 'max_depth': [7, 8, 9], 'colsample_bytree': [0.6, 0.7, 1], 'gamma': [0.0, 0.1, 0.2]}` and the best parameters detected were `{'colsample_bytree': 0.6, 'gamma': 0.2, 'learning_rate': 0.05, 'max_depth': 9, 'n_estimators': 200}`. The full output of the grid search computations can be found in **Appendix F**.

Results

For study **(a)**, a summary of the relevant results from the OLS Regression is provided in the table below.

feature	coef	std err	t-statistic	P>|t|
proximity	0.0100	0.000	22.827	0.000
Coastline_Dist	-429.2069	12.301	-34.893	0.000

From our results, the P-value for a two-tailed T-Test on our Coastline_Dist feature shows that it is close to 0.000. This means that with a 95% confidence interval, we can say that the null hypothesis can be rejected.

An interpretation of H_0 and H_1 from the values above can be inferred as follows. The H_0 states that the coefficient for the Coastline_Dist feature is 0, i.e. the prices regressed using Coastline_Dist will be unaffected by any change in Coastline_Dist. However, H_1 states that the coefficient is *-429.2069*, which means - that for every 1 unit increase in distance from the coast line, the price drops by 429 dollars. The t-statistic calculated using the Standard Error measures to *-34.893*, which gives us a P-value close to

0.000. Using this p-value, we can reject the null hypothesis. The similar argument also holds for the proximity data. The feature importance based on information gain and weight (total number of splits on a feature across the ensemble of trees) for each feature for our XGBRegressor is also shown in **Appendix D**. These figures show that Coastline Dist produces relatively high information gain and weight, whereas proximity only has very high weight (indicative of the low coefficient value in the OLS summary).

For study **(b)**, we go on to compare our Linear and XGB Regressors to conclude if the XGB Regressor is performing better than its counterpart. The evaluation of each model after selecting the best parameters is summarised in the table below:

	Training R-squared (%)	Test R-squared (%)	Training uncertainty at 95% confidence interval (AUD)	Test uncertainty at 95% confidence interval (AUD)
Linear Regressor	53.99	53.88	219.2	213.9
XGB Regressor	84.29	65.06	128.1	186.2

The residual plots for each model is provided in **Appendix E**. Critical analysis of the numbers above suggest that the XGB Regressor has a better 'goodness of fit' based on the R-squared values, and also has lower standard error on both the training and testing data, which leads to a smaller range of uncertainty as well. We can therefore claim that our XGB Regressor outperforms our standard OLS Linear Regressor benchmark.
In order to prove this claim, we perform a one-tailed T-Test on the r-squared value across a 100 fold cross-validation strategy on the entire dataset. We reject the null hypothesis as p-value = 1.3247202608446335e-16, when the alternative says that XGBRegressor is the better predictor.

Limitations
Despite a relatively high R-squared value for our better model - XGB Regressor, the uncertainty range of our model even after cross validation does not promise much value. Given that the median price range for Sydney AirbBnB listings is approximately 125 AUD, an uncertainty range of 186 AUD cripples the usefulness of our model. Additionally, a stark difference of 19% in R-squared values between training and testing set indicate overfitting the model on the training data set despite cross-validation. It turns out that the price is dependent not only on geography, size, and features. It stands to reason that the quality of presentation (e.g. pictures), availability, the number and content of reviews, communication (e.g. acceptance rate, host response time) or status (whether or not the host is a super host) might have a substantial influence too. These influences along with the high dimensionality of our data allows a high variance in our data set, thus disabling our model to generalise well to the data. Possible improvements include using a deep neural network, which are known to adapt well to high dimensional data, and perhaps including features we discarded from our study as well.

Conclusion

This extensive study has provided a valuable opportunity to explore the depths of Data Analytics and Data Science, their principles and fundamentals overall. It taught me that domain knowledge is particularly very important for a Data Analyst. I was also quite fond of learning about the statistical tests and measures used to validate the performance of Machine Learning models, as well as the validity and truthfulness of research statements. I believe these skills are a necessary toolkit in R&D and in industry (particularly for A/B testing) to provide justified reports and conclusions. Additionally, my skills with Python, pandas, sklearn, statsmodels, etc. has been hugely upgraded due to the undertaking of this study.

The purpose of this analysis was to recommend a price to a "rookie" without any reviews or status. With this in mind, we might say that we can't recommend an exact price, but rather a range. As such, we could recommend our solution to the problem as somewhat of a gateway to more improved solutions.

Bibliography

[1] “Airbnb: Advantages and Disadvantages,” Investopedia, 2020. [Online]. Available: https://www.investopedia.com/articles/personal-finance/032814/pros-and-cons-using-airbnb.asp. [Accessed: 11-Apr-2020].
[2] Radoslaw Panczak and T. Sigler, “Ever wondered how many Airbnbs Australia has and where they all are? We have the answers,” The Conversation, 12-Feb-2020. [Online]. Available: https://theconversation.com/ever-wondered-how-many-airbnbs-australia-has-and-where-they-all-are-we-have-the-answers-129003. [Accessed: 11-Apr-2020].
[3] The Flawed Consumer, “How much can you earn from Airbnb in Australia? - The Flawed Consumer,” The Flawed Consumer, 15-Mar-2019. [Online]. Available: https://www.theflawedconsumer.com/how-much-money-can-you-earn-from-airbnb-in-australia/. [Accessed: 11-Apr-2020].
[4] “Inside Airbnb. Adding data to the debate.,” Inside Airbnb, 2020. [Online]. Available: http://insideairbnb.com/get-the-data.html. [Accessed: 11-Apr-2020].
[5] “Inside Airbnb. Adding data to the debate.,” Inside Airbnb, 2014. [Online]. Available: http://insideairbnb.com/behind.html. [Accessed: 11-Apr-2020].
[6] “Public Transport - Location Facilities and Operators | TfNSW Open Data Hub and Developer Portal,” Nsw.gov.au, 2020. [Online]. Available: https://opendata.transport.nsw.gov.au/dataset/public-transport-location-facilities-and-operators. [Accessed: 11-Apr-2020].
[7] https://www.facebook.com/MachineLearningMastery, “A Gentle Introduction to XGBoost for Applied Machine Learning,” Machine Learning Mastery, 16-Aug-2016. [Online]. Available: https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/. [Accessed: 23-May-2020].

Appendix A - Proximity Transport Services and Income in $ by District

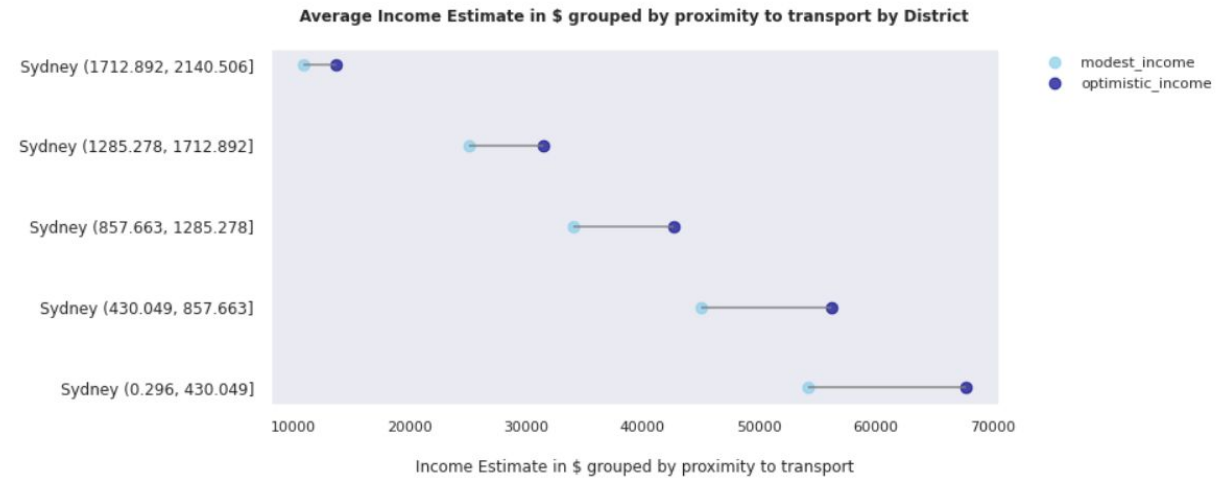

Figure 1: Income varying with distance for Sydney CBD district

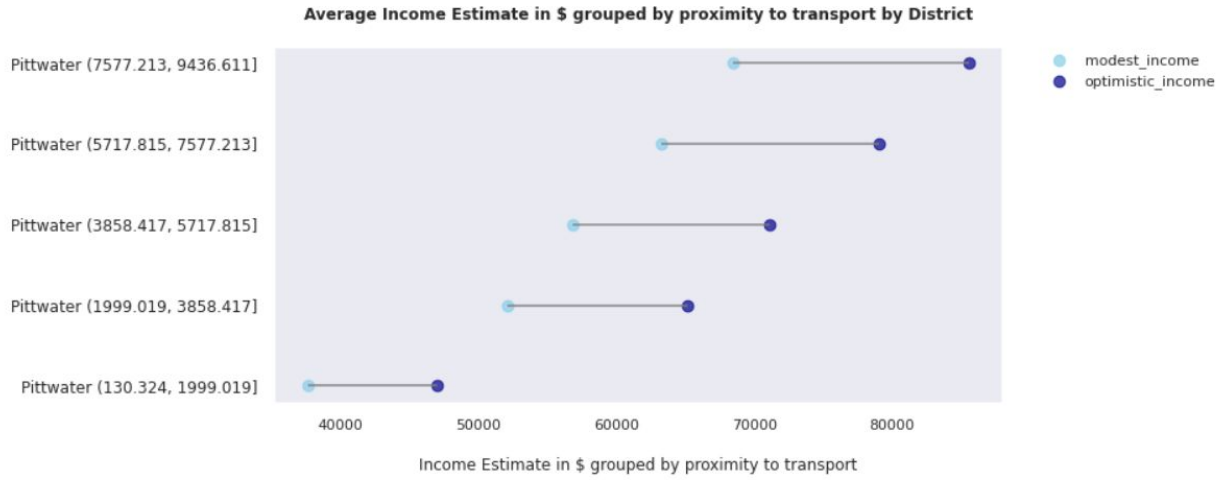

Figure 2: Income varying with distance for Pittwater district

Appendix B - Price heat map by latitude and longitude

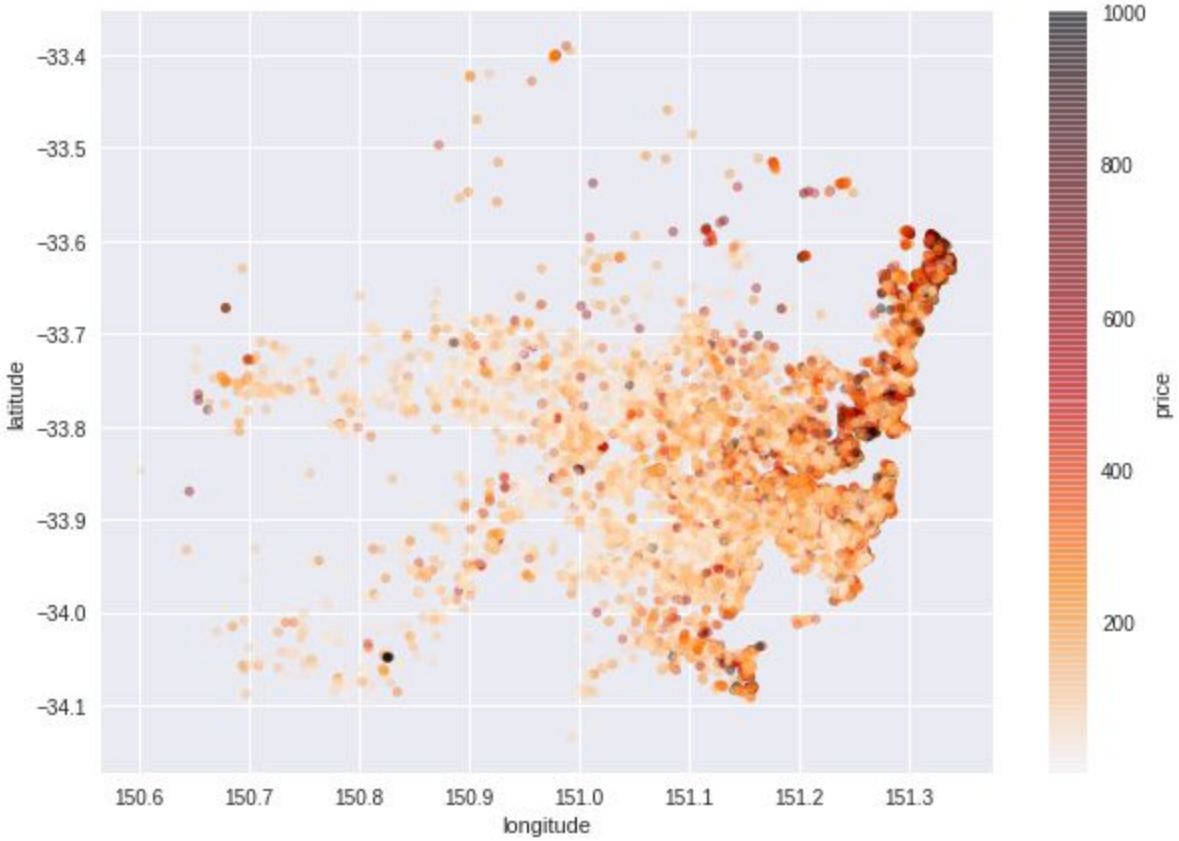

Figure 3: Price (AUD) heat map by latitude and longitude

Appendix D - Feature Importance Plots from XGB Regressor

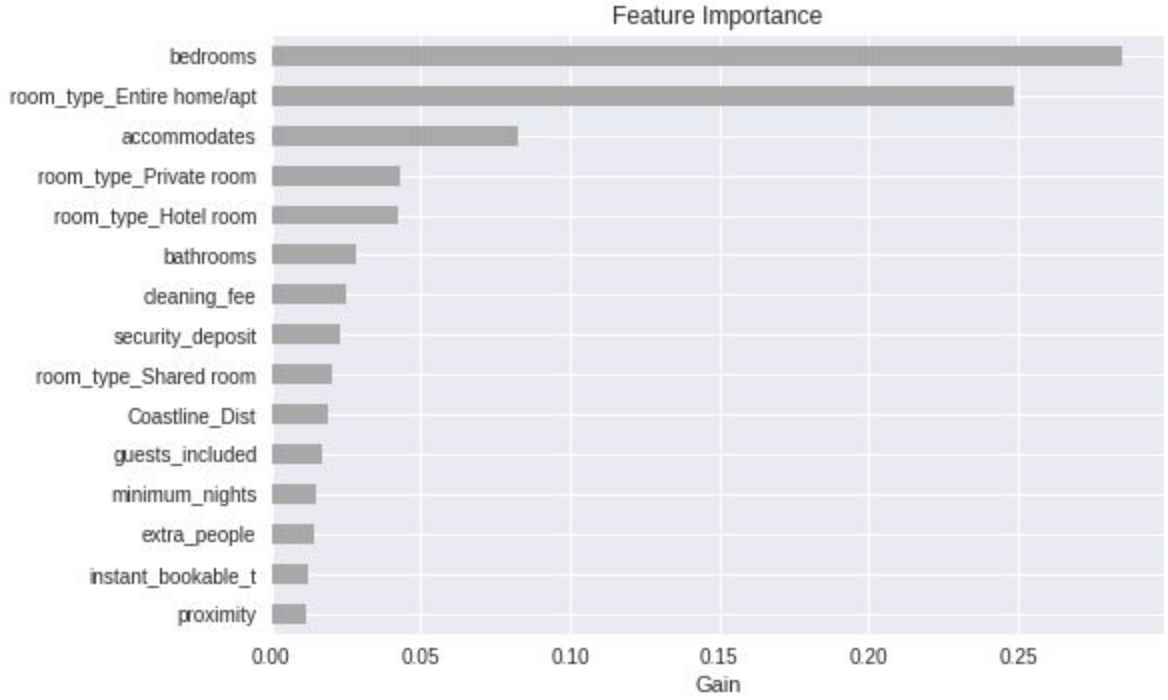

Figure 4: Top 15 features by information gain

Appendix E - Residual Plots for training data (blue) and test data (green)

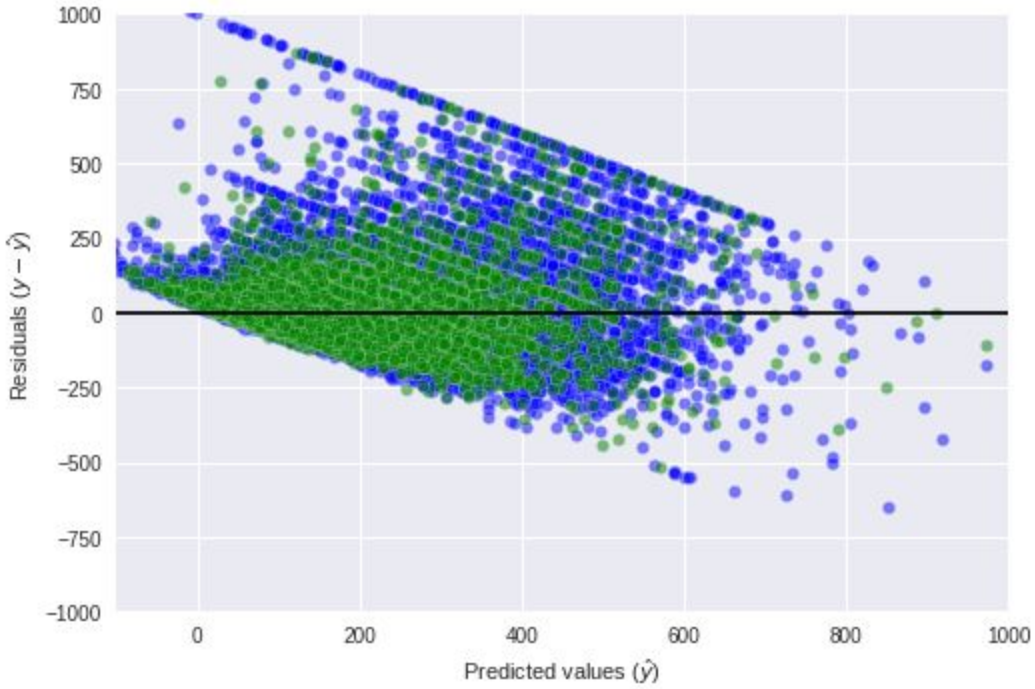

Figure 6: Residual plot for OLS Linear Regressor Model (AUD)

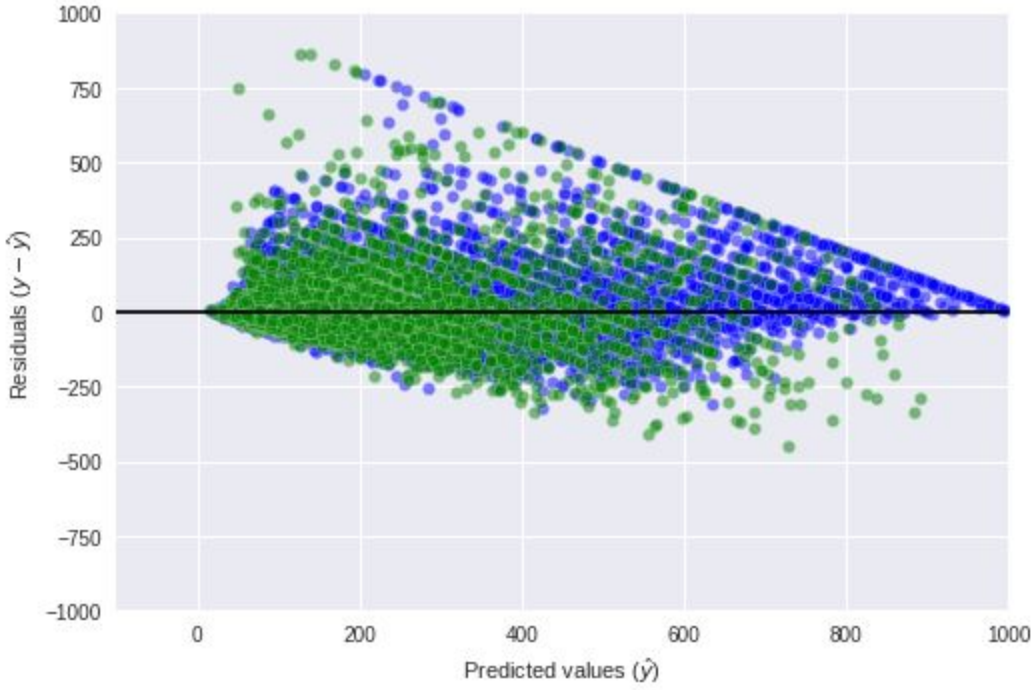

Figure 7: Residual plot for XGB Regressor Model (AUD)

Appendix F - Grid Search output

Linear Regressor -
Grid search mean and stdev:

-15737166176773607424.000 (+/-9442299706064163635 2.000) for {'copy_X': True, 'fit_intercept': True, 'normalize': True}
0.536 (+/-0.040) for {'copy_X': True, 'fit_intercept': True, 'normalize': False}
0.536 (+/-0.040) for {'copy_X': True, 'fit_intercept': False, 'normalize': True}
0.536 (+/-0.040) for {'copy_X': True, 'fit_intercept': False, 'normalize': False}

Linear Regressor best params:
{'copy_X': True, 'fit_intercept': False, 'normalize': True}

Linear Regressor r-squared on training data:
0.5399022219614744

Linear Regressor r-squared on test data:
0.5388431919502075

XGB Regressor -

Grid search mean and stdev:

0.600 (+/-0.026) for {'colsample_bytree': 0.6, 'gamma': 0.0, 'learning_rate': 0.01, 'max_depth': 7, 'n_estimators': 200}
0.629 (+/-0.025) for {'colsample_bytree': 0.6, 'gamma': 0.0, 'learning_rate': 0.01, 'max_depth': 7, 'n_estimators': 250}
0.605 (+/-0.025) for {'colsample_bytree': 0.6, 'gamma': 0.0, 'learning_rate': 0.01, 'max_depth': 8, 'n_estimators': 200}
0.634 (+/-0.024) for {'colsample_bytree': 0.6, 'gamma': 0.0, 'learning_rate': 0.01, 'max_depth': 8, 'n_estimators': 250}
0.608 (+/-0.025) for {'colsample_bytree': 0.6, 'gamma': 0.0, 'learning_rate': 0.01, 'max_depth': 9, 'n_estimators': 200}
0.636 (+/-0.024) for {'colsample_bytree': 0.6, 'gamma': 0.0, 'learning_rate': 0.01, 'max_depth': 9, 'n_estimators': 250}
0.660 (+/-0.023) for {'colsample_bytree': 0.6, 'gamma': 0.0, 'learning_rate': 0.05, 'max_depth': 7, 'n_estimators': 200}
0.660 (+/-0.023) for {'colsample_bytree': 0.6, 'gamma': 0.0, 'learning_rate': 0.05, 'max_depth': 7, 'n_estimators': 250}
0.661 (+/-0.025) for {'colsample_bytree': 0.6, 'gamma': 0.0, 'learning_rate': 0.05, 'max_depth': 8, 'n_estimators': 200}
0.661 (+/-0.026) for {'colsample_bytree': 0.6, 'gamma': 0.0, 'learning_rate': 0.05, 'max_depth': 8, 'n_estimators': 250}
0.661 (+/-0.023) for {'colsample_bytree': 0.6, 'gamma': 0.0, 'learning_rate': 0.05, 'max_depth': 9, 'n_estimators': 200}
0.661 (+/-0.023) for {'colsample_bytree': 0.6, 'gamma': 0.0, 'learning_rate': 0.05, 'max_depth': 9, 'n_estimators': 250}
0.657 (+/-0.028) for {'colsample_bytree': 0.6, 'gamma': 0.0, 'learning_rate': 0.1, 'max_depth': 7, 'n_estimators': 200}
0.656 (+/-0.028) for {'colsample_bytree': 0.6, 'gamma': 0.0, 'learning_rate': 0.1, 'max_depth': 7, 'n_estimators': 250}
0.657 (+/-0.024) for {'colsample_bytree': 0.6, 'gamma': 0.0, 'learning_rate': 0.1, 'max_depth': 8, 'n_estimators': 200}
0.656 (+/-0.023) for {'colsample_bytree': 0.6, 'gamma': 0.0, 'learning_rate': 0.1, 'max_depth': 8, 'n_estimators': 250}
0.654 (+/-0.025) for {'colsample_bytree': 0.6, 'gamma': 0.0, 'learning_rate': 0.1, 'max_depth': 9, 'n_estimators': 200}
0.653 (+/-0.025) for {'colsample_bytree': 0.6, 'gamma': 0.0, 'learning_rate': 0.1, 'max_depth': 9, 'n_estimators': 250}
0.600 (+/-0.026) for {'colsample_bytree': 0.6, 'gamma': 0.1, 'learning_rate': 0.01, 'max_depth': 7, 'n_estimators': 200}
0.629 (+/-0.025) for {'colsample_bytree': 0.6, 'gamma': 0.1, 'learning_rate': 0.01, 'max_depth': 7, 'n_estimators': 250}
0.605 (+/-0.025) for {'colsample_bytree': 0.6, 'gamma': 0.1, 'learning_rate': 0.01, 'max_depth': 8, 'n_estimators': 200}
0.634 (+/-0.024) for {'colsample_bytree': 0.6, 'gamma': 0.1, 'learning_rate': 0.01, 'max_depth': 8, 'n_estimators': 250}
0.608 (+/-0.025) for {'colsample_bytree': 0.6, 'gamma': 0.1, 'learning_rate': 0.01, 'max_depth': 9, 'n_estimators': 200}
0.636 (+/-0.024) for {'colsample_bytree': 0.6, 'gamma': 0.1, 'learning_rate': 0.01, 'max_depth': 9, 'n_estimators': 250}
0.660 (+/-0.023) for {'colsample_bytree': 0.6, 'gamma': 0.1, 'learning_rate': 0.05, 'max_depth': 7, 'n_estimators': 200}

0.660 (+/-0.023) for {'colsample_bytree': 0.6, 'gamma': 0.1, 'learning_rate': 0.05, 'max_depth': 7, 'n_estimators': 250}
0.661 (+/-0.025) for {'colsample_bytree': 0.6, 'gamma': 0.1, 'learning_rate': 0.05, 'max_depth': 8, 'n_estimators': 200}
0.661 (+/-0.026) for {'colsample_bytree': 0.6, 'gamma': 0.1, 'learning_rate': 0.05, 'max_depth': 8, 'n_estimators': 250}
0.661 (+/-0.023) for {'colsample_bytree': 0.6, 'gamma': 0.1, 'learning_rate': 0.05, 'max_depth': 9, 'n_estimators': 200}
0.661 (+/-0.023) for {'colsample_bytree': 0.6, 'gamma': 0.1, 'learning_rate': 0.05, 'max_depth': 9, 'n_estimators': 250}
0.657 (+/-0.028) for {'colsample_bytree': 0.6, 'gamma': 0.1, 'learning_rate': 0.1, 'max_depth': 7, 'n_estimators': 200}
0.656 (+/-0.028) for {'colsample_bytree': 0.6, 'gamma': 0.1, 'learning_rate': 0.1, 'max_depth': 7, 'n_estimators': 250}
0.657 (+/-0.024) for {'colsample_bytree': 0.6, 'gamma': 0.1, 'learning_rate': 0.1, 'max_depth': 8, 'n_estimators': 200}
0.656 (+/-0.024) for {'colsample_bytree': 0.6, 'gamma': 0.1, 'learning_rate': 0.1, 'max_depth': 8, 'n_estimators': 250}
0.654 (+/-0.026) for {'colsample_bytree': 0.6, 'gamma': 0.1, 'learning_rate': 0.1, 'max_depth': 9, 'n_estimators': 200}
0.653 (+/-0.026) for {'colsample_bytree': 0.6, 'gamma': 0.1, 'learning_rate': 0.1, 'max_depth': 9, 'n_estimators': 250}
0.600 (+/-0.026) for {'colsample_bytree': 0.6, 'gamma': 0.2, 'learning_rate': 0.01, 'max_depth': 7, 'n_estimators': 200}
0.629 (+/-0.025) for {'colsample_bytree': 0.6, 'gamma': 0.2, 'learning_rate': 0.01, 'max_depth': 7, 'n_estimators': 250}
0.605 (+/-0.025) for {'colsample_bytree': 0.6, 'gamma': 0.2, 'learning_rate': 0.01, 'max_depth': 8, 'n_estimators': 200}
0.634 (+/-0.024) for {'colsample_bytree': 0.6, 'gamma': 0.2, 'learning_rate': 0.01, 'max_depth': 8, 'n_estimators': 250}
0.608 (+/-0.025) for {'colsample_bytree': 0.6, 'gamma': 0.2, 'learning_rate': 0.01, 'max_depth': 9, 'n_estimators': 200}
0.636 (+/-0.024) for {'colsample_bytree': 0.6, 'gamma': 0.2, 'learning_rate': 0.01, 'max_depth': 9, 'n_estimators': 250}
0.660 (+/-0.023) for {'colsample_bytree': 0.6, 'gamma': 0.2, 'learning_rate': 0.05, 'max_depth': 7, 'n_estimators': 200}
0.660 (+/-0.023) for {'colsample_bytree': 0.6, 'gamma': 0.2, 'learning_rate': 0.05, 'max_depth': 7, 'n_estimators': 250}
0.661 (+/-0.026) for {'colsample_bytree': 0.6, 'gamma': 0.2, 'learning_rate': 0.05, 'max_depth': 8, 'n_estimators': 200}
0.661 (+/-0.026) for {'colsample_bytree': 0.6, 'gamma': 0.2, 'learning_rate': 0.05, 'max_depth': 8, 'n_estimators': 250}
0.661 (+/-0.023) for {'colsample_bytree': 0.6, 'gamma': 0.2, 'learning_rate': 0.05, 'max_depth': 9, 'n_estimators': 200}
0.661 (+/-0.023) for {'colsample_bytree': 0.6, 'gamma': 0.2, 'learning_rate': 0.05, 'max_depth': 9, 'n_estimators': 250}
0.657 (+/-0.028) for {'colsample_bytree': 0.6, 'gamma': 0.2, 'learning_rate': 0.1, 'max_depth': 7, 'n_estimators': 200}
0.656 (+/-0.028) for {'colsample_bytree': 0.6, 'gamma': 0.2, 'learning_rate': 0.1, 'max_depth': 7, 'n_estimators': 250}
0.657 (+/-0.024) for {'colsample_bytree': 0.6, 'gamma': 0.2, 'learning_rate': 0.1, 'max_depth': 8, 'n_estimators': 200}
0.656 (+/-0.024) for {'colsample_bytree': 0.6, 'gamma': 0.2, 'learning_rate': 0.1, 'max_depth': 8, 'n_estimators': 250}
0.655 (+/-0.026) for {'colsample_bytree': 0.6, 'gamma': 0.2, 'learning_rate': 0.1, 'max_depth': 9, 'n_estimators': 200}
0.654 (+/-0.026) for {'colsample_bytree': 0.6, 'gamma': 0.2, 'learning_rate': 0.1, 'max_depth': 9, 'n_estimators': 250}
0.600 (+/-0.026) for {'colsample_bytree': 0.7, 'gamma': 0.0, 'learning_rate': 0.01, 'max_depth': 7, 'n_estimators': 200}
0.628 (+/-0.024) for {'colsample_bytree': 0.7, 'gamma': 0.0, 'learning_rate': 0.01, 'max_depth': 7, 'n_estimators': 250}
0.605 (+/-0.024) for {'colsample_bytree': 0.7, 'gamma': 0.0, 'learning_rate': 0.01, 'max_depth': 8, 'n_estimators': 200}
0.633 (+/-0.023) for {'colsample_bytree': 0.7, 'gamma': 0.0, 'learning_rate': 0.01, 'max_depth': 8, 'n_estimators': 250}
0.607 (+/-0.025) for {'colsample_bytree': 0.7, 'gamma': 0.0, 'learning_rate': 0.01, 'max_depth': 9, 'n_estimators': 200}
0.635 (+/-0.024) for {'colsample_bytree': 0.7, 'gamma': 0.0, 'learning_rate': 0.01, 'max_depth': 9, 'n_estimators': 250}
0.658 (+/-0.023) for {'colsample_bytree': 0.7, 'gamma': 0.0, 'learning_rate': 0.05, 'max_depth': 7, 'n_estimators': 200}
0.658 (+/-0.023) for {'colsample_bytree': 0.7, 'gamma': 0.0, 'learning_rate': 0.05, 'max_depth': 7, 'n_estimators': 250}
0.660 (+/-0.024) for {'colsample_bytree': 0.7, 'gamma': 0.0, 'learning_rate': 0.05, 'max_depth': 8, 'n_estimators': 200}

0.660 (+/-0.024) for {'colsample_bytree': 0.7, 'gamma': 0.0, 'learning_rate': 0.05, 'max_depth': 8, 'n_estimators': 250}
0.659 (+/-0.025) for {'colsample_bytree': 0.7, 'gamma': 0.0, 'learning_rate': 0.05, 'max_depth': 9, 'n_estimators': 200}
0.659 (+/-0.025) for {'colsample_bytree': 0.7, 'gamma': 0.0, 'learning_rate': 0.05, 'max_depth': 9, 'n_estimators': 250}
0.656 (+/-0.024) for {'colsample_bytree': 0.7, 'gamma': 0.0, 'learning_rate': 0.1, 'max_depth': 7, 'n_estimators': 200}
0.655 (+/-0.024) for {'colsample_bytree': 0.7, 'gamma': 0.0, 'learning_rate': 0.1, 'max_depth': 7, 'n_estimators': 250}
0.656 (+/-0.026) for {'colsample_bytree': 0.7, 'gamma': 0.0, 'learning_rate': 0.1, 'max_depth': 8, 'n_estimators': 200}
0.655 (+/-0.027) for {'colsample_bytree': 0.7, 'gamma': 0.0, 'learning_rate': 0.1, 'max_depth': 8, 'n_estimators': 250}
0.655 (+/-0.023) for {'colsample_bytree': 0.7, 'gamma': 0.0, 'learning_rate': 0.1, 'max_depth': 9, 'n_estimators': 200}
0.654 (+/-0.022) for {'colsample_bytree': 0.7, 'gamma': 0.0, 'learning_rate': 0.1, 'max_depth': 9, 'n_estimators': 250}
0.600 (+/-0.026) for {'colsample_bytree': 0.7, 'gamma': 0.1, 'learning_rate': 0.01, 'max_depth': 7, 'n_estimators': 200}
0.628 (+/-0.024) for {'colsample_bytree': 0.7, 'gamma': 0.1, 'learning_rate': 0.01, 'max_depth': 7, 'n_estimators': 250}
0.605 (+/-0.024) for {'colsample_bytree': 0.7, 'gamma': 0.1, 'learning_rate': 0.01, 'max_depth': 8, 'n_estimators': 200}
0.633 (+/-0.023) for {'colsample_bytree': 0.7, 'gamma': 0.1, 'learning_rate': 0.01, 'max_depth': 8, 'n_estimators': 250}
0.607 (+/-0.025) for {'colsample_bytree': 0.7, 'gamma': 0.1, 'learning_rate': 0.01, 'max_depth': 9, 'n_estimators': 200}
0.635 (+/-0.024) for {'colsample_bytree': 0.7, 'gamma': 0.1, 'learning_rate': 0.01, 'max_depth': 9, 'n_estimators': 250}
0.658 (+/-0.023) for {'colsample_bytree': 0.7, 'gamma': 0.1, 'learning_rate': 0.05, 'max_depth': 7, 'n_estimators': 200}
0.658 (+/-0.023) for {'colsample_bytree': 0.7, 'gamma': 0.1, 'learning_rate': 0.05, 'max_depth': 7, 'n_estimators': 250}
0.660 (+/-0.024) for {'colsample_bytree': 0.7, 'gamma': 0.1, 'learning_rate': 0.05, 'max_depth': 8, 'n_estimators': 200}
0.660 (+/-0.024) for {'colsample_bytree': 0.7, 'gamma': 0.1, 'learning_rate': 0.05, 'max_depth': 8, 'n_estimators': 250}
0.659 (+/-0.025) for {'colsample_bytree': 0.7, 'gamma': 0.1, 'learning_rate': 0.05, 'max_depth': 9, 'n_estimators': 200}
0.659 (+/-0.025) for {'colsample_bytree': 0.7, 'gamma': 0.1, 'learning_rate': 0.05, 'max_depth': 9, 'n_estimators': 250}
0.655 (+/-0.024) for {'colsample_bytree': 0.7, 'gamma': 0.1, 'learning_rate': 0.1, 'max_depth': 7, 'n_estimators': 200}
0.655 (+/-0.024) for {'colsample_bytree': 0.7, 'gamma': 0.1, 'learning_rate': 0.1, 'max_depth': 7, 'n_estimators': 250}
0.656 (+/-0.025) for {'colsample_bytree': 0.7, 'gamma': 0.1, 'learning_rate': 0.1, 'max_depth': 8, 'n_estimators': 200}
0.655 (+/-0.026) for {'colsample_bytree': 0.7, 'gamma': 0.1, 'learning_rate': 0.1, 'max_depth': 8, 'n_estimators': 250}
0.655 (+/-0.023) for {'colsample_bytree': 0.7, 'gamma': 0.1, 'learning_rate': 0.1, 'max_depth': 9, 'n_estimators': 200}
0.654 (+/-0.023) for {'colsample_bytree': 0.7, 'gamma': 0.1, 'learning_rate': 0.1, 'max_depth': 9, 'n_estimators': 250}
0.600 (+/-0.026) for {'colsample_bytree': 0.7, 'gamma': 0.2, 'learning_rate': 0.01, 'max_depth': 7, 'n_estimators': 200}
0.628 (+/-0.024) for {'colsample_bytree': 0.7, 'gamma': 0.2, 'learning_rate': 0.01, 'max_depth': 7, 'n_estimators': 250}
0.605 (+/-0.024) for {'colsample_bytree': 0.7, 'gamma': 0.2, 'learning_rate': 0.01, 'max_depth': 8, 'n_estimators': 200}
0.633 (+/-0.023) for {'colsample_bytree': 0.7, 'gamma': 0.2, 'learning_rate': 0.01, 'max_depth': 8, 'n_estimators': 250}
0.607 (+/-0.025) for {'colsample_bytree': 0.7, 'gamma': 0.2, 'learning_rate': 0.01, 'max_depth': 9, 'n_estimators': 200}
0.635 (+/-0.024) for {'colsample_bytree': 0.7, 'gamma': 0.2, 'learning_rate': 0.01, 'max_depth': 9, 'n_estimators': 250}
0.658 (+/-0.023) for {'colsample_bytree': 0.7, 'gamma': 0.2, 'learning_rate': 0.05, 'max_depth': 7, 'n_estimators': 200}
0.658 (+/-0.023) for {'colsample_bytree': 0.7, 'gamma': 0.2, 'learning_rate': 0.05, 'max_depth': 7, 'n_estimators': 250}
0.660 (+/-0.024) for {'colsample_bytree': 0.7, 'gamma': 0.2, 'learning_rate': 0.05, 'max_depth': 8, 'n_estimators': 200}
0.660 (+/-0.024) for {'colsample_bytree': 0.7, 'gamma': 0.2, 'learning_rate': 0.05, 'max_depth': 8, 'n_estimators': 250}
0.659 (+/-0.025) for {'colsample_bytree': 0.7, 'gamma': 0.2, 'learning_rate': 0.05, 'max_depth': 9, 'n_estimators': 200}

0.659 (+/-0.025) for {'colsample_bytree': 0.7, 'gamma': 0.2, 'learning_rate': 0.05, 'max_depth': 9, 'n_estimators': 250}
0.656 (+/-0.024) for {'colsample_bytree': 0.7, 'gamma': 0.2, 'learning_rate': 0.1, 'max_depth': 7, 'n_estimators': 200}
0.655 (+/-0.024) for {'colsample_bytree': 0.7, 'gamma': 0.2, 'learning_rate': 0.1, 'max_depth': 7, 'n_estimators': 250}
0.655 (+/-0.025) for {'colsample_bytree': 0.7, 'gamma': 0.2, 'learning_rate': 0.1, 'max_depth': 8, 'n_estimators': 200}
0.655 (+/-0.026) for {'colsample_bytree': 0.7, 'gamma': 0.2, 'learning_rate': 0.1, 'max_depth': 8, 'n_estimators': 250}
0.655 (+/-0.023) for {'colsample_bytree': 0.7, 'gamma': 0.2, 'learning_rate': 0.1, 'max_depth': 9, 'n_estimators': 200}
0.654 (+/-0.023) for {'colsample_bytree': 0.7, 'gamma': 0.2, 'learning_rate': 0.1, 'max_depth': 9, 'n_estimators': 250}
0.599 (+/-0.027) for {'colsample_bytree': 1, 'gamma': 0.0, 'learning_rate': 0.01, 'max_depth': 7, 'n_estimators': 200}
0.626 (+/-0.026) for {'colsample_bytree': 1, 'gamma': 0.0, 'learning_rate': 0.01, 'max_depth': 7, 'n_estimators': 250}
0.601 (+/-0.027) for {'colsample_bytree': 1, 'gamma': 0.0, 'learning_rate': 0.01, 'max_depth': 8, 'n_estimators': 200}
0.629 (+/-0.026) for {'colsample_bytree': 1, 'gamma': 0.0, 'learning_rate': 0.01, 'max_depth': 8, 'n_estimators': 250}
0.602 (+/-0.027) for {'colsample_bytree': 1, 'gamma': 0.0, 'learning_rate': 0.01, 'max_depth': 9, 'n_estimators': 200}
0.628 (+/-0.028) for {'colsample_bytree': 1, 'gamma': 0.0, 'learning_rate': 0.01, 'max_depth': 9, 'n_estimators': 250}
0.655 (+/-0.024) for {'colsample_bytree': 1, 'gamma': 0.0, 'learning_rate': 0.05, 'max_depth': 7, 'n_estimators': 200}
0.655 (+/-0.024) for {'colsample_bytree': 1, 'gamma': 0.0, 'learning_rate': 0.05, 'max_depth': 7, 'n_estimators': 250}
0.654 (+/-0.024) for {'colsample_bytree': 1, 'gamma': 0.0, 'learning_rate': 0.05, 'max_depth': 8, 'n_estimators': 200}
0.654 (+/-0.023) for {'colsample_bytree': 1, 'gamma': 0.0, 'learning_rate': 0.05, 'max_depth': 8, 'n_estimators': 250}
0.651 (+/-0.028) for {'colsample_bytree': 1, 'gamma': 0.0, 'learning_rate': 0.05, 'max_depth': 9, 'n_estimators': 200}
0.650 (+/-0.027) for {'colsample_bytree': 1, 'gamma': 0.0, 'learning_rate': 0.05, 'max_depth': 9, 'n_estimators': 250}
0.651 (+/-0.026) for {'colsample_bytree': 1, 'gamma': 0.0, 'learning_rate': 0.1, 'max_depth': 7, 'n_estimators': 200}
0.650 (+/-0.027) for {'colsample_bytree': 1, 'gamma': 0.0, 'learning_rate': 0.1, 'max_depth': 7, 'n_estimators': 250}
0.650 (+/-0.026) for {'colsample_bytree': 1, 'gamma': 0.0, 'learning_rate': 0.1, 'max_depth': 8, 'n_estimators': 200}
0.649 (+/-0.026) for {'colsample_bytree': 1, 'gamma': 0.0, 'learning_rate': 0.1, 'max_depth': 8, 'n_estimators': 250}
0.648 (+/-0.028) for {'colsample_bytree': 1, 'gamma': 0.0, 'learning_rate': 0.1, 'max_depth': 9, 'n_estimators': 200}
0.646 (+/-0.028) for {'colsample_bytree': 1, 'gamma': 0.0, 'learning_rate': 0.1, 'max_depth': 9, 'n_estimators': 250}
0.599 (+/-0.027) for {'colsample_bytree': 1, 'gamma': 0.1, 'learning_rate': 0.01, 'max_depth': 7, 'n_estimators': 200}
0.626 (+/-0.026) for {'colsample_bytree': 1, 'gamma': 0.1, 'learning_rate': 0.01, 'max_depth': 7, 'n_estimators': 250}
0.601 (+/-0.027) for {'colsample_bytree': 1, 'gamma': 0.1, 'learning_rate': 0.01, 'max_depth': 8, 'n_estimators': 200}
0.629 (+/-0.026) for {'colsample_bytree': 1, 'gamma': 0.1, 'learning_rate': 0.01, 'max_depth': 8, 'n_estimators': 250}
0.602 (+/-0.027) for {'colsample_bytree': 1, 'gamma': 0.1, 'learning_rate': 0.01, 'max_depth': 9, 'n_estimators': 200}
0.628 (+/-0.028) for {'colsample_bytree': 1, 'gamma': 0.1, 'learning_rate': 0.01, 'max_depth': 9, 'n_estimators': 250}
0.655 (+/-0.024) for {'colsample_bytree': 1, 'gamma': 0.1, 'learning_rate': 0.05, 'max_depth': 7, 'n_estimators': 200}
0.655 (+/-0.024) for {'colsample_bytree': 1, 'gamma': 0.1, 'learning_rate': 0.05, 'max_depth': 7, 'n_estimators': 250}
0.654 (+/-0.024) for {'colsample_bytree': 1, 'gamma': 0.1, 'learning_rate': 0.05, 'max_depth': 8, 'n_estimators': 200}
0.654 (+/-0.023) for {'colsample_bytree': 1, 'gamma': 0.1, 'learning_rate': 0.05, 'max_depth': 8, 'n_estimators': 250}
0.651 (+/-0.027) for {'colsample_bytree': 1, 'gamma': 0.1, 'learning_rate': 0.05, 'max_depth': 9, 'n_estimators': 200}
0.650 (+/-0.027) for {'colsample_bytree': 1, 'gamma': 0.1, 'learning_rate': 0.05, 'max_depth': 9, 'n_estimators': 250}
0.651 (+/-0.026) for {'colsample_bytree': 1, 'gamma': 0.1, 'learning_rate': 0.1, 'max_depth': 7, 'n_estimators': 200}
0.650 (+/-0.026) for {'colsample_bytree': 1, 'gamma': 0.1, 'learning_rate': 0.1, 'max_depth': 7, 'n_estimators': 250}
0.650 (+/-0.026) for {'colsample_bytree': 1, 'gamma': 0.1, 'learning_rate': 0.1, 'max_depth': 8, 'n_estimators': 200}
0.649 (+/-0.026) for {'colsample_bytree': 1, 'gamma': 0.1, 'learning_rate': 0.1, 'max_depth': 8, 'n_estimators': 250}
0.648 (+/-0.028) for {'colsample_bytree': 1, 'gamma': 0.1, 'learning_rate': 0.1, 'max_depth': 9, 'n_estimators': 200}
0.646 (+/-0.028) for {'colsample_bytree': 1, 'gamma': 0.1, 'learning_rate': 0.1, 'max_depth': 9, 'n_estimators': 250}
0.599 (+/-0.027) for {'colsample_bytree': 1, 'gamma': 0.2, 'learning_rate': 0.01, 'max_depth': 7, 'n_estimators': 200}
0.626 (+/-0.026) for {'colsample_bytree': 1, 'gamma': 0.2, 'learning_rate': 0.01, 'max_depth': 7, 'n_estimators': 250}
0.601 (+/-0.027) for {'colsample_bytree': 1, 'gamma': 0.2, 'learning_rate': 0.01, 'max_depth': 8, 'n_estimators': 200}
0.629 (+/-0.026) for {'colsample_bytree': 1, 'gamma': 0.2, 'learning_rate': 0.01, 'max_depth': 8, 'n_estimators': 250}
0.602 (+/-0.027) for {'colsample_bytree': 1, 'gamma': 0.2, 'learning_rate': 0.01, 'max_depth': 9, 'n_estimators': 200}
0.628 (+/-0.028) for {'colsample_bytree': 1, 'gamma': 0.2, 'learning_rate': 0.01, 'max_depth': 9, 'n_estimators': 250}
0.655 (+/-0.024) for {'colsample_bytree': 1, 'gamma': 0.2, 'learning_rate': 0.05, 'max_depth': 7, 'n_estimators': 200}
0.655 (+/-0.024) for {'colsample_bytree': 1, 'gamma': 0.2, 'learning_rate': 0.05, 'max_depth': 7, 'n_estimators': 250}
0.654 (+/-0.024) for {'colsample_bytree': 1, 'gamma': 0.2, 'learning_rate': 0.05, 'max_depth': 8, 'n_estimators': 200}
0.654 (+/-0.023) for {'colsample_bytree': 1, 'gamma': 0.2, 'learning_rate': 0.05, 'max_depth': 8, 'n_estimators': 250}
0.650 (+/-0.027) for {'colsample_bytree': 1, 'gamma': 0.2, 'learning_rate': 0.05, 'max_depth': 9, 'n_estimators': 200}
0.650 (+/-0.027) for {'colsample_bytree': 1, 'gamma': 0.2, 'learning_rate': 0.05, 'max_depth': 9, 'n_estimators': 250}
0.652 (+/-0.026) for {'colsample_bytree': 1, 'gamma': 0.2, 'learning_rate': 0.1, 'max_depth': 7, 'n_estimators': 200}
0.651 (+/-0.026) for {'colsample_bytree': 1, 'gamma': 0.2, 'learning_rate': 0.1, 'max_depth': 7, 'n_estimators': 250}
0.651 (+/-0.026) for {'colsample_bytree': 1, 'gamma': 0.2, 'learning_rate': 0.1, 'max_depth': 8, 'n_estimators': 200}
0.649 (+/-0.026) for {'colsample_bytree': 1, 'gamma': 0.2, 'learning_rate': 0.1, 'max_depth': 8, 'n_estimators': 250}
0.648 (+/-0.029) for {'colsample_bytree': 1, 'gamma': 0.2, 'learning_rate': 0.1, 'max_depth': 9, 'n_estimators': 200}
0.646 (+/-0.030) for {'colsample_bytree': 1, 'gamma': 0.2, 'learning_rate': 0.1, 'max_depth': 9, 'n_estimators': 250}

XGB Regressor best params:
{'colsample_bytree': 0.6, 'gamma': 0.2, 'learning_rate': 0.05, 'max_depth': 9, 'n_estimators': 200}

XGB Regressor r-squared on training data:
0.8429225555573611

XGB Regressor r-squared on test data:
0.6505580834182669