Machine Learning Group Assignment Report

Team Members
Tolga Arslan 20302472, Oliver Kraus 20304856, Sai Kaushik Mudigonda 20303676

1. Introduction

Airbnb is a website that allows individuals to advertise their property for short-term stays like vacations. The site has come under criticism for driving up rent prices in Ireland, which is experiencing a housing crisis. In this project we create a dataset from the airbnb listings across Ireland. We then use features extracted from the website to predict the listing prices with various regression models. This data could be compared to data from the Irish rental market to further investigate the influence airbnb has on the Irish rental market.

2. Dataset and Features

We obtain our dataset from web scraping the airbnb website by using the Selenium library. The airbnb website shows 20 advertisements per page and a maximum of 15 pages per search radius. This means we can extract a maximum of 300 advertisements per search area from the website. To automatically obtain a sufficiently large dataset for all of Ireland we therefore decided to run a search for all 32 Irish counties on the airbnb website (including Northern Ireland). For each listing on the airbnb website, we extract the following features: Title of the listing, average rating, number of ratings, superhost status of the host (title for the highest rated hosts on the airbnb website), location of the property, description of the property, number of rooms the property has and price per night. We hypothesize that these are the most important pieces of information from a listing. After completion of the web scraping step we had 32 datasets (one for each Irish county). These datasets were merged to one dataset which finally contained 8 columns and 8267 rows (some counties have less than 300 listings).

	TITLE	AVERAGE RATING	NUMBER OF REVIEWS	SUPERHOST	LOCATION	DESCRIPTION	NUMBER OF ROOMS	PRICE PER NIGHT
0	Dunadry Townhouse - Close To Airport & Motorway	4.56	(56 reviews)	Superhost	Muckamore, Co Antrim, United Kingdom	Entire residential home hosted by Gary	4 guests · 2 bedrooms · 3 beds · 1 bathroom	€42

Figure 1: Sample of the original dataset which was extracted from the airbnb website

We then continued with preprocessing the dataset. The first step was to remove duplicates from the dataset, since we noticed that some counties contained search results from different counties, even though we restricted the search area to a specific county on the airbnb website. We identified duplicates based on the title of the listing. This reduced the number of rows to 4845. Also, we removed the title of the listing from the dataset after this step, as it is not relevant for predicting the price of a property. We removed the non-price variables in the column caused by the error of the data due to the shift between columns and updated it with the price variable in the same row. In the "number of reviews" and "price per night" columns, we've removed the "," used to separate numeric values and all other non-numeric characters (like €). Some listings did not have any reviews and therefore not an average rating (both values are “none” in the original dataset). For these cases we set the number of reviews to 0 and set the average rating to the average rating across all listings (which is 4.87). The “superhost” column was converted to boolean values. We replaced the original “location” column with a “county” column by extracting the county name from the original location column and then assigned an unique integer identifier to each county. The introduction of this feature allows us to detect regional differences in the listing prices. The “number of rooms” feature was split up into the four new features “guests”, “bedrooms”, “beds” and “bathrooms”, to make it easier to interpret for a machine learning algorithm. From the “description” column we extracted whether a

listing is a studio or not (boolean value). Some listings on airbnb are just a single room while others are full studios and we assume that this has an influence on the listing price. We finally converted the data types in the columns to numeric types (float64 and int64). By shuffling our data, we made sure the data points were evenly distributed and can later apply train-test splits.

	AVERAGE RATING	NUMBER OF REVIEWS	SUPERHOST	COUNTY	STUDIO	GUESTS	BEDROOMS	BEDS	BATHROOMS	PRICE
0	4.750000	4	0	26	0	7	4	0	4	105

Figure 2: Sample of the dataset after the preprocessing step

After completing the preprocessing we explored the obtained datasets. In Figure 3 we can see that the number of listings vary greatly by county. This makes it hard to compare the prices based on the county. Also, we can observe in Figure 4 that the average price by county varies greatly. Interestingly, Dublin has the lowest average price.

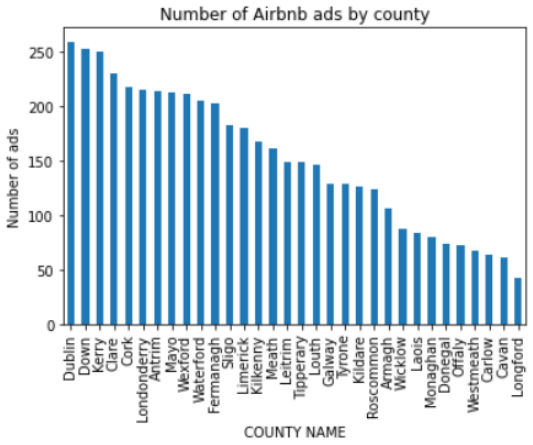


Figure 3: Number of listings by county

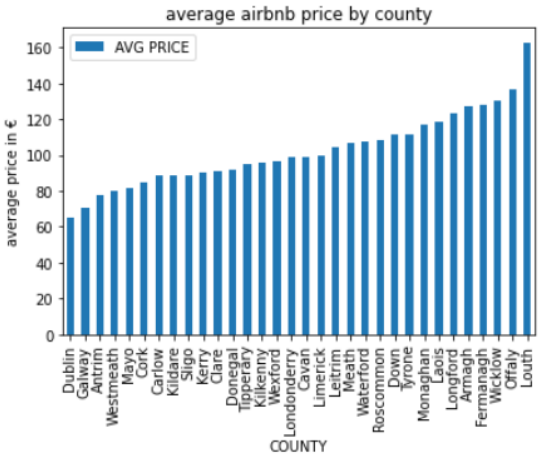


Figure 4: Average airbnb price by county

Figure 5: Correlation Heat Map

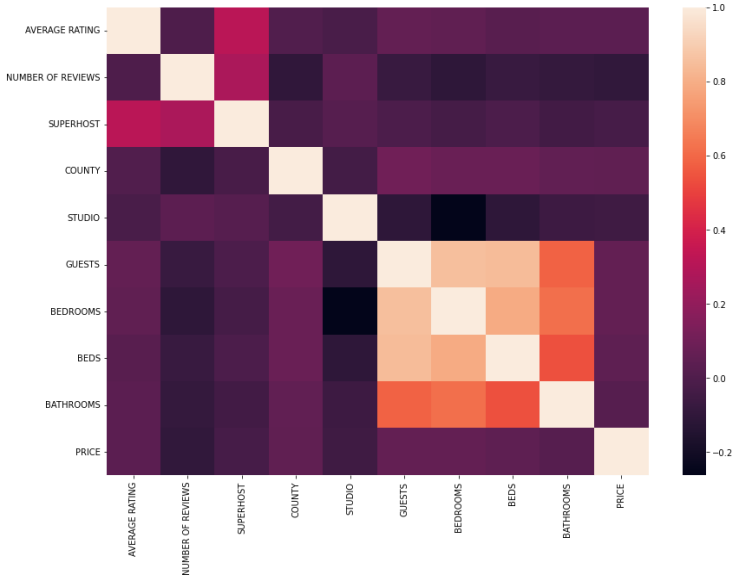


Figure 5 shows that there is generally no high correlation between the values. Most importantly, there is no high correlation between the price of a property and any other feature that we used. This will make it hard for us to create an accurate model. There is a high correlation between the number of guests and the number of bedrooms/beds, but this will not help us with the prediction of the property prices.

3. Methods

Linear Regression

By applying Linear Regression on our data, we measured how our data responded to training with the linear approach. The linear approach did not produce a successful result on our data. In addition to this approach, we removed 3 columns from our dataset and retrained our model according to the importance of the features we obtained with XGBoost. In addition to these, we included our approach where we observed the effect of polynomial degrees(degree 2 to 5) on our model. We have explained all these in detail in the discussion section of this report.

Lasso and Ridge Regression

We added Lasso and Ridge regression since we had more than 2 features and were also adding polynomial regression. It would lead the model to overfit. So through lasso and ridge regression, we can penalize this error.

- Lasso Regression:
  Lasso Regression performs L1 regularization, it adds a penalty which is equal to the sum of the absolute value of the coefficients. The alpha value here is calculated by 1/C. Lasso regression helps in reducing the effect of less prominent features and making them closer to zero. So we feel that this is better for our model than ridge regression.

- Ridge Regression:
  Ridge Regression performs L2 regularization, it adds a penalty which is equal to the square of magnitude of the coefficients. In Ridge regression, alpha value is calculated by 1/2C. This type of regression works similar to lasso regression wrt the lambda value.

kNN

Another simple model we use is kNN regression. It is different from the linear regression models, because it does not try to fit a line through the data, but rather makes predictions based on the k nearest training data points. A kNN model can therefore predict different “shapes” in the data than a regression model. The standard distance metric to calculate the nearest training data points is the euclidean metric. The value of the datapoint can then be predicted by taking the weighted mean of the values from the k nearest data points. The weighting of the nearest neighbours can be modified, but the standard is uniform weighting. This means that all neighbours have the same impact on the prediction of the data point. The only hyperparameter that has to be tuned is therefore k.

Dummy Regression

To compare results between our models, we also created a "DummyRegression" model that estimates the mean value. Since the model uses the mean value for prediction, we did not expect a successful result from this model. As we expected, we met with low success as a result of the model.

XGBoost

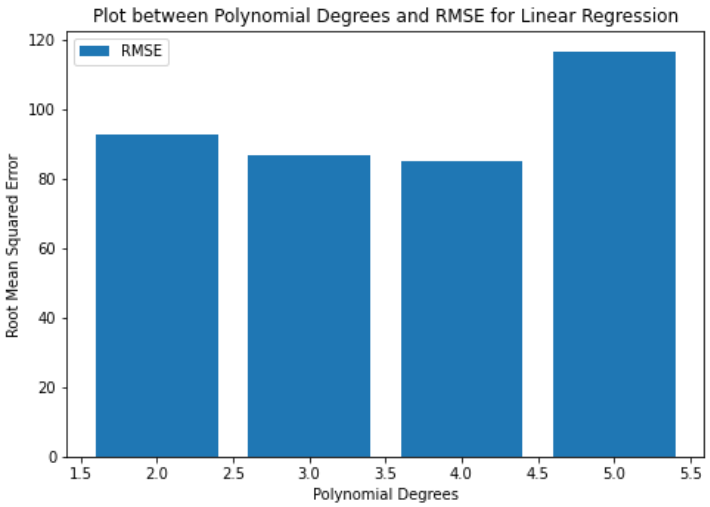
Using XGBRegressor, we first set up our model on the default parameter values. Afterwards, we thought that the optimum adjustment of the parameter values in our model could have a positive effect on our model, and we obtained the best parameter values for our model with GridSearchCV. In addition, with XGBoost, we determined the importance of the features in our dataset for our model and target value. We have applied the output from this approach to our other models to improve our results.

4. Experiments/Results/Discussion

Linear Regression

When we applied it to our Linear Regression model, which is one of the basic approaches, we obtained the RMSE value of 96.91 and the model score of 0.001. According to these scores, our result can be classified as unsuccessful. Low success is certainly due to many reasons, which we will explain in the relevant sections. Here, one of the reasons for obtaining an unsuccessful result is that the linear model does not completely surround the data set values.

When we remove the [“BEDS”, “SUPERHOST”, “STUDIO”] columns on the Linear Regression model, we see that the RMSE value decreased to 96.62 and our model score increased to 0.01.



Additionally, we extended our Linear Regression model by applying polynomial properties. Among the polynomial features we created in the range(2,6), we achieved the lowest rmse above grade 4 in terms of Model score and RMSE value. We got 83.88 as RMSE value and 0.26 as model score on degree 4. We didn't train our model for higher degrees as there was more probability of overfitting when using large and complex equations to represent our model and the time taken to train the model was also increasing exponentially.

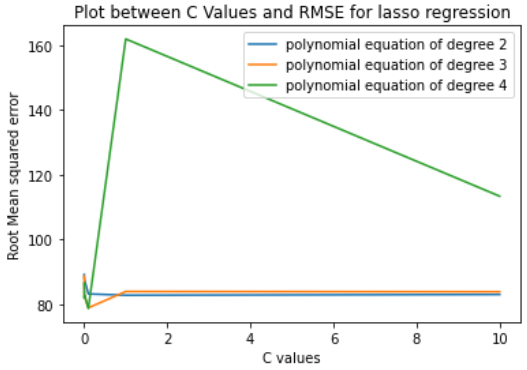
Figure 6: Polynomial Degrees and RMSE Values

Lasso Regression

From the XGBoost result, we got the importance value of all our features. So we tried removing the less important features and trained our model on it again. We removed columns like, studio, superhost, and beds. We trained our data on lasso regression with polynomial degrees from 2 to 5 and for each of these degrees, we evaluated our lasso regression with multiple values of c values to get the degree and c value combination for the least error. From the results we found that adding polynomial features with degree 4 to the data and training it with lasso regression at c value=0.01 gives the lowest error.

```
Lasso Regression at polynomial degree 4 and c: 0.01
MAE: 40.65825786640516
MSE: 6198.472232337564
RMSE: 78.73037680805017
```

Figure 7: plot between c values and rsme for lasso



Ridge Regression

We trained our ridge regression model with the same parameter as we did with the lasso regression. We had added polynomial features from degree 2 to 5 with different values of c. Without removing the features which were below the threshold, we got the

combination with least rmse at degree, 4 and c= 0.01 with results,

```
MAE: 40.40120843726776
MSE: 6362.608622238284
RMSE: 79.76596155151823
```

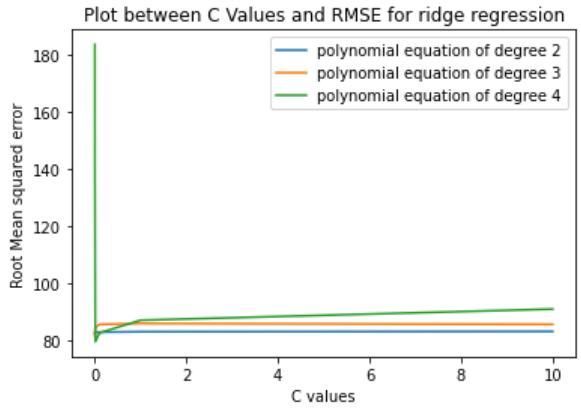


Figure 8: plot between c values and RMSE for ridge regression

In our case, both lasso and ridge regression are giving almost the same result for us.

kNN

We use five fold cross validation to select the hyperparameter k. We choose the negative mean squared error as a scoring method for the regression problem. The results are shown in Figure 9. We can see in the figure that the results of the model are bad for any parameter k.

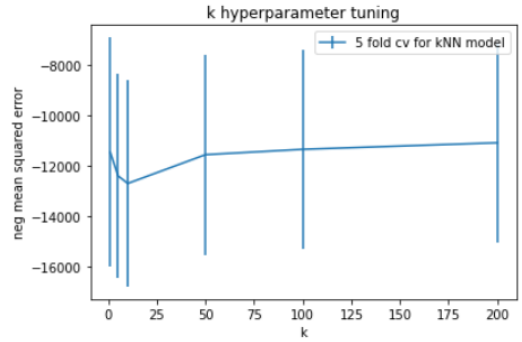


Figure 9: k hyperparameter tuning

The results are constantly bad, so it does not seem interesting to explore k values beyond 200. We choose a k value of 100 with uniform weighting which results in a RMSE of 136.3055.
In an attempt to improve the performance of the model we reduce the features according to the feature importance plot from the XGBoost model (features “beds”, “studio” and “superhost” are dropped).
Figure 10 shows that the results are equally bad for the kNN model with the reduced features. Again, it does not seem interesting to explore k values beyond 200 because the graph is nearly constant after k > 50. We again choose a k value of 200 with uniform weights. The RMSE of this model is 136.4272 which is nearly the same as for the previous model. No kNN model we tried was therefore able to adequately predict the data, since both their scores are worse than the dummy regression.

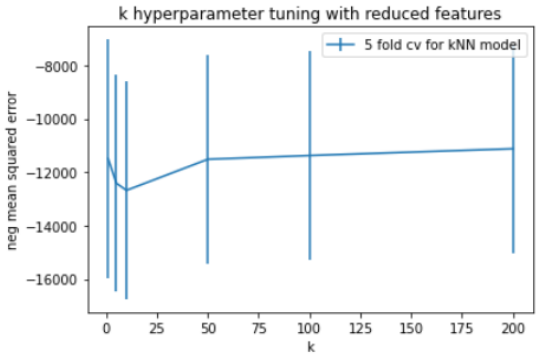


Figure 10: k hyperparameter tuning for reduced features

Dummy Regression

We created a "DummyRegression" model that estimates the mean value. As a result of the model, we got 97.25 for the RMSE value and -0.0001 for score value. From this we can see that the model fits our data extremely poorly.

XGBoost

```
parameters= {
    #"booster":["gblinear"],
    "n_estimators": np.arange(100, 450, 50),
    "objective":["reg:squarederror"],
    'colsample_bytree': [1,0.8,0.5],
    "min_child_weight":[1,3,5],
    'max_depth': np.arange(1,6,1),
    'alpha': [0,1],
    "gama":[0,0.3,0.5],
    "eta":[0.1,0.3],
    'lambda': [1,1.2],
    'subsample': [1,0.8,0.5,0.3,0.1]
}
```

We built the model on XGBoost to improve our results. In the model we developed using the default parameter values on XGBoost, we got the RMSE value of 78.73 and the model score of 0.34.

Here, optimum adjustment of parameter values could have given us a more effective result, we were aware of this. For this, we ran the values we defined for the parameters of XGBoost with GridSearchCV with feedback. You can see these values on the code block above. To put it more clearly, we narrowed the value ranges we defined at each step according to the effect on the model. Here, we saw that the RMSE value for which we built the model with the best parameters we obtained decreased partially, and the model score partially increased. The obtained RMSE value is 78.47 and the model score is 0.35. You can see the best parameter values obtained in the result block below.

```
Output:

Best:  {'alpha': 0, 'colsample_bytree': 1, 'eta': 0.1, 'gama': 0, 'lambda': 1,
'max_depth': 3, 'min_child_weight': 1, 'n_estimators': 200, 'objective':
'reg:squarederror', 'subsample': 1}
```

In addition, at the end of the XGBoost model, we created our feature_importance graph that shows the importance of the dataset columns on the model and target value. Based on this graph, we can say that [“GUESTS”, “COUNTY”, “NUMBER OF REVIEWS”] values have a very strong effect on the model, while the effect of [“BEDS”, “SUPERHOST”, “STUDIO”] values on the target value and the model is low. You can see the graph we got below.

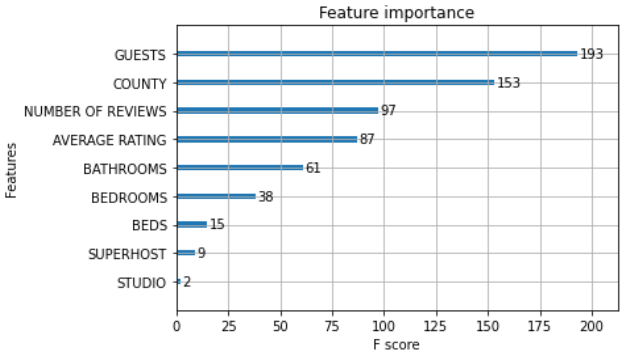


Figure11: Feature Importance

This transaction paved the way for a new approach. Reconstructing our models by removing the columns [“BEDS”, “SUPERHOST”, “STUDIO”], which we classified as insignificant, from our data set would have benefited us in terms of model success. We applied this approach to all of our models.

[“BEDS”, “SUPERHOST”, “STUDIO”] columns are removed from the data set, and the RMSE value we obtained in the XGBoost model we established is 79.66, and the model score is 0.33. Removing these columns caused a partial decrease in our model success. From this we can understand that these columns have a partially positive effect on the model.

5. Summary

None of our models could accurately predict the listing prices. The reason for this is the low quality of the scraped data and the selected features. We have seen that none of our features had a correlation to the listing prices. To improve the performance of our models we could therefore try to better extract the property type from the airbnb listings (like “single room”, “villa” or “town house”). This has probably a high impact on the price of a listing.
Even the amenities data which is available for each listing would be very helpful in the contribution of the price prediction for a listing. Another major drawback we had was the limit airbnb had while accessing different listings. We were only able to get a maximum of 300 listings per search. Had this been more, we would have collected more data and there might also have been some improvement in our models.

6. Contributions

Tolga Arslan, 20302472, T.A

- Report: Introduction, Dataset and Features, Linear Regression&XGBoost&Dummy Regression sections in Methods and Evaluation Titles, Summary
- Scraped 9 counties from airbnb website
- Combining of discrete data obtained for all counties
- Creation of combined datasets at all stages.
- Creating data cleaning and data preprocessing processes on the data set
- Worked with oliver and kaushik on web scraping code, data preprocessing and cleaning.
- Statistical approaches, correlation analysis in data set analysis/explanation part
- Application of Linear Regression, Baseline Models, XGBoost on the data set.
- Wrote code for data visualization(graphs, heatmap)
- In the improving results part, the construction of the XGboost model, the determination of the best parameters for XGBoost via GridSearchCV, the determination of the importance of the data set columns on the model and target value to apply to all models in our study.

Oliver Kraus, 20305836, O.K.

- Report: Introduction, Dataset and Features, kNN section in Methods and kNN section in evaluation, summary
- Scraped 11 counties from airbnb website

- Worked together with Tolga and Kaushik on the web scraping code, data preprocessing
- Wrote code for data visualization
- Wrote kNN model code
- Data cleaning and extracting the county column from raw data

Sai Kaushik Mudigonda, 20303676, S.K.M

- Report: Methods, Lasso and ridge regression in experiments section, graphs between c values and rmse.
- Scraped 12 counties data from airbnb website.
- Data cleaning, and extracting features like studio, bedrooms, beds, guests.
- Worked with oliver and tolga on web scraping code, data preprocessing and cleaning.
- Worked on polynomial features, lasso and ridge regression model's code.

7. Project Links

Web Scraping

https://colab.research.google.com/drive/1NPKicezn9_qTd5MUBWvuLQnN30blnSMP?usp=sharing

Models

https://colab.research.google.com/drive/1f5cr57Add2rNyidH-hgdjaeRE0Nskza7?usp=sharing