1. An introduction to the CSV dataset

The dataset, Ireland House Price.csv, is a sales dataset on the Irish property market. This dataset contains information on different real estate properties, including price, location, total area of house, and other house characteristics. According to a given backgroud, potential Irish homebuyers have always been sensitive to the size, location and insulation of properties, resulting in different rates of property purchase in four local authorities in Dublin City: DCC, Fingal, Dun Laoghaire and South Dublin. The purpose of this analysis is to figure out that what factors influence potential Irish home buyers and predict which property is more likely to be sold using models. Therefore, 'buying or not buying' is first defined as the target variable, and then the 'price-per-sqft-$' is defined as the second target variable since by looking at other characteristics of this dataset, those may affect the price of a house per square foot.

2. Preliminary Exploratory Data Analysis (EDA)

2.1 Data structure and overview

The dataset has 12 columns with 13320 rows of data, including 3 columns of float64 type, 1 column of int64 type, and 8 columns of object type.

2.2 Preliminary data cleaning

Duplicate Values and Missing Values

After checking, it was found that there were no duplicate values in the dataset, but 945 missing values. Finally, all the missing values were deleted for the following reasons: It is assumed that after deleting all missing values, 796 rows will be reduced noting the total number of lines containing missing values is 796, however, the total number of rows is 13320. Considering that this missing value rows only account for about 5.9% of the total data, and the value of missing value, such as bath and size, cannot be reasonably speculated, what's more, its value may be related to other characteristics, as well as the price per square foot, so the changes are possible to lose the original law of the data through a simple filling method, so the 796 rows were finally discarded, leaving 12524 rows of data.

Data Normalization

Normalizing the data for each column of attributes. First checked the unique values of each categorical row and keep only numbers and decimal points for each numerical column for subsequent calculation. Then, adjusted the information in the DataFrame. For example, converting ‘availability’ to a uniform datetime format (YYYY-MM-DD) ; changing ‘buy or not buying’ to ‘buy_state’; change ‘price-per-sqft-$' to ‘price(€/sqtf)’ and keep both values two decimal places; replacing the range value in ‘total_sqft’ with the average, and finally convert each column to the float64 type, etc.

Processing of Outliers

Shapiro-Wilk test was used to check the distribution of the data, and it was found that none of the clomun belonged to normal distribution. Therefore, the Quartile method (suitable for non-normal distribution) was selected to detect the outliers of this dataset. The skewness and kurtosis of the data were calculated to inspect the deviation degree of each column. Then, It was found that the skewness of the kurtosis of most columns was high, so it’s resonable to consider threshold region as 3 times IQR. However, subsequent model tests showed that the accuracy of the model in this case was lower than that at a common 1.5 times, so 1.5 times was still selected for subsequent analysis. The results of the detected outliers by Quartile method (with threhold=1.5) are as follows:

Column	***Outliers***
total_sqft	1031 rows
price(€/sqtf)	1181 rows
number of bedrooms	681 rows
bath	807 rows
balcony	0 rows

Table 1 Outliers for numeric columns

For outliers, I replaced the outliers that are beyond the threshold with the maximum value of the threshold, and replace the outliers that are below the minimum value with the minimum value of the threshold. To compare the impact of handling outliers on model, the data before and after processing are stored in two DataFrames——"df" and "df_cleaned".

Steps such as data cleaning, normalization, and handling outliers are critical in any machine learning project (e.g., handling missing values, standardization, and outlier treatment) (Zhang & Zhong, 2017), after processing the data through the above steps, the data becomes cleaner and standardized, which is convenient for subsequent calculation and prediction.

Descriptive statistical analysis

Metric	***total_sqft***	***price(€/sqtf)***
Mean	1445.8	722.0
Standard Deviation	609.6	366.9
Minimum	5.0	30.4
Maximum	3260.0	1812.8
Q1 (25%)	1100.0	481.4
Q2 (50%)	1260.0	612.1
Q3(75%)	1640	814.3

Table 2 Descriptive statistical characteristics of continuous variables

It can be seen that the minimum value displayed is unreasonable due to the existence of some outliers.

And the distribution bar charts of the categorical variables and interval numerical variables associated with the Irish property market dataset are below:

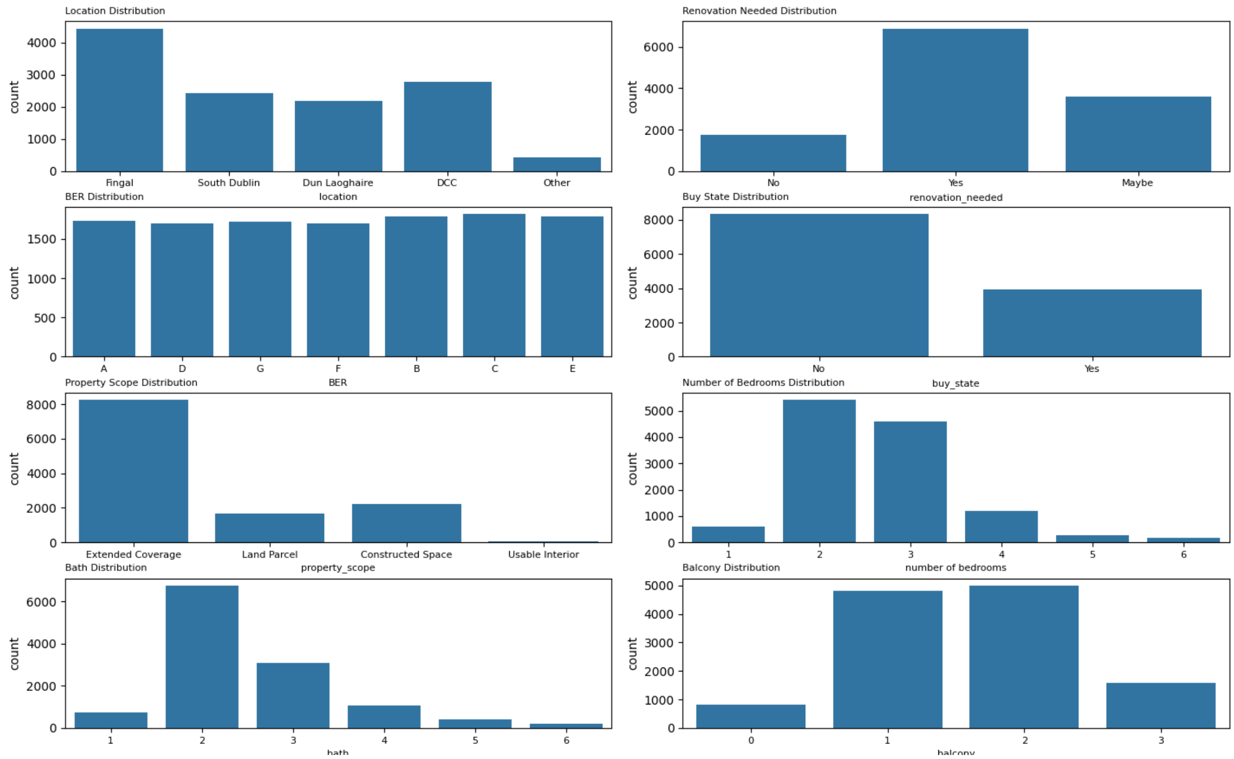

Fig 1 Distribution bar charts of the categorical and interval variables

The bar chart shows that:

Location distribution: the majority of the properties are located in Fingal and south Dublin, with a smaller number in Dun Laoghaire. And there is a category marked "other" indicates sites that do not belong to the four main local authorities.

BER (Building Energy Grade) distribution: there is a fairly balanced distribution of different energy grades (A to G). The highest amounts belong to BER C.

Renovation needs distribution: a large proportion of properties are marked "Yes", indicating that renovations are needed, while “Maybe” has a second largest proportion, indicating that most of these properties are in poor condition and need to be renovated.

Buy state distribution: most houses have not yet been purchased.

Property scope distribution: "extended coverage" is the most common property feature, followed by "constructed space", with fewer properties in “land parcel”.

Number of bedrooms distribution: most properties have 2 or 3 bedrooms, followed by 1 and 4 bedrooms, and fewer properties have 5 or 6 bedrooms.

Balcony distribution: Most properties have less than 1 or 2 balconies, and few properties have more balconies.

2.4 Correlation Analysis

Since the data contains various types of columns, and the correlation matrix is usually used to measure the linear relationship between continuous variables, directly calculating the correlation matrix may lead to errors, so all the categorical variables in “df_cleaned” were first labeled and these newly generated columns are stored. At the same time, the original “df” data containing outliers was also encoded for comparing the results of df and df_clean against the subsequent model tests. Because the data of some attributes did not conform to the normal distribution, the spearman coorrelation will be more resonable. Based on the encoded DataFrame, the spearman correlation coefficient and p-value were calculated:

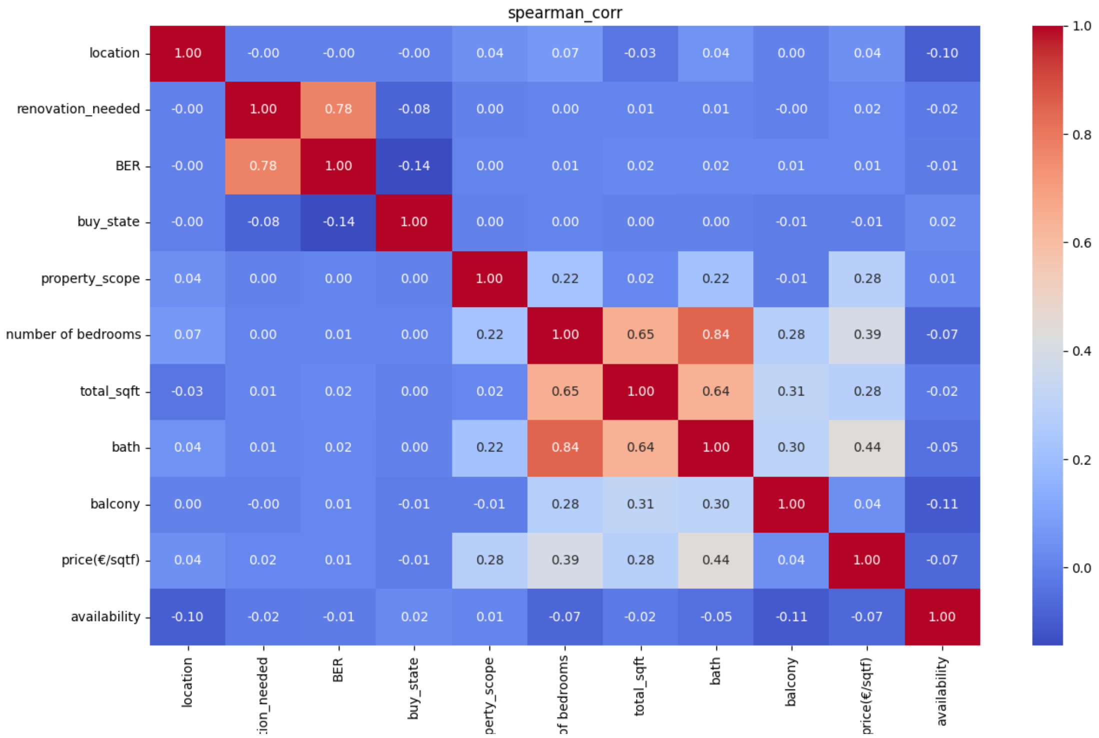

Fig 2 Heat map of spearman correlation between variables

This heatmap shows that there is no strong correlation between the target variables (“buy_state” & “price(€/sqtf)”) and other variables that are highly correlated with the target variables were also not found, which indicates that other features of the target variable may not have an obvious monotonic relationship, but a more complex relationship. At the same time, it is found that the correlation coefficient between “bath” and “number of bedrooms” is 0.84, indicating that the two exist collinearity. Highly correlated features might make it difficult for the model to distinguish the independent contribution of each feature, thus affecting the interpretation and stability of the model. So keep this in mind when using models later.

2.4 Feature Importance Analysis

In order to further understand the influencing factors of target variables, the random forest model was used again to check the importance of each feature *(Chandrashekar & Sahin, 2014)*, and the results are as follows:

Important Features for Buy_state	*df*	*df_cleaned*
Location	0.158993	0.160442
Total_sqft	0.244474	0.241995
Balcony	0.054318	0.056370
Price(€/sqft)	0.314252	0.308689
BER	0.102001	0.107709

Table 3 Selected Important Features of "buy_state"

Important Features for price(€/sqtf)	*df*	*df_cleaned*
Location	0.069238	0.196634
Total_sqft	0.307499	0.252712
Balcony	0.054814	0.113759
Bath	0.054814	
BER	0.080654	0.166114
Number of bedrooms	0.092607	
Property_scope	0.289467	0.131612

Table 4 Selected Important Features of "price(€/sqtf)"

Important Features for renovation_needed	*df*	*df_cleaned*
BER	1.0	1.0

Table 5 Selected Important Features of “renovation_needed”

The important features of “buy_state” screened from the data containing outliers and the data without outliers have little difference, but the important features of “price(€/sqtf)” have changed greatly. This is because although tree models (such as random forest and XGBoost) are relatively robust to outliers, they may still select segmentation points near outliers for features with dense outliers, which affects importance assessment. After cleaning outliers, the model is more focused on the distribution of most data points, and the feature importance is more reflective of major trends. Especially in the case of large data noise, the importance of cleaning is usually more accurate. At the same time, it is also found that “BER” has a high forecasting ability for “renovation_needed” before and after data cleaning, which can be verified by subsequent analysis.

3 Model Analysis, Prediction and Evaluation

3.1 Model Analysis, Prediction and Evaluation of “buy_state”

Model Training

Since the target variable “buy_state” is a binary (0-1) variable, Logistic Regression model, Decision Tree model, Random Forest model, Gradient Boosting model, SVM model, Naive Bayes and KNN model are used to study “buy_state” and the remaining 10 features variables. The results show that SVM model and GBM model perform well, their confusion matrix and classification report results are as follows:

SVM Results
Accuracy: 0.7126130
Confusion Matrix: [[2561   15] [1065   117]]

GBM Results
Accuracy:0.760777
Confusion Matrix: [[2574   2] [897   285]]

Table 6 Accuracy & Confusion Matrix

	precision	*recall*	*F1-score*	*support*
0	**0.71**	**0.99**	**0.83**	**2576**
1	**0.89**	**0.10**	**0.18**	**1182**
Accuracy			**0.71**	**3758**
Macro avg	**0.80**	**0.55**	**0.50**	**3758**
Weighted avg	**0.76**	**0.71**	**0.62**	**3758**

Table 7 Confusion Matrix & Classification Report of SVM

	precision	*recall*	*F1-score*	*support*
0	**0.74**	**1.00**	**0.85**	**2576**
1	**0.99**	**0.24**	**0.39**	**1182**
Accuracy			**0.76**	**3758**
Macro avg	**0.87**	**0.62**	**0.62**	**3758**
Weighted avg	**0.82**	**0.76**	**0.71**	**3758**

Table 8 Confusion Matrix & Classification Report of GBM

The results show that GBM outperforms SVM in accuracy, accuracy and weighted average, especially in the prediction ability of negative classes. However, both perform poorly on the recall rate of the positive class (class 1), possibly because the number of positive samples in the data is much smaller than the number of negative samples, resulting in a weak recognition ability of the model for the positive class. If you need to improve performance on the positive class, consider using more sophisticated techniques such as undersampling/oversampling or adjusting the threshold of the model, or using a model that specifically optimizes the positive class recall rate.

Model Optimization

Then, Trying to optimize data features by removing relatively repetitive variables with multicollinearity. After “bath” was deleted, predicted the residual variable and the target variable again. Then predicted again with removing another feature —“number of bedrooms” , and the two results showed that the interpretation degree of the model declined slightly. So I choose to retain these features.

Next, optimizing the GBM model. In the case of unbalanced data categories, the results after optimization by undersampling/oversampling and adjusting the decision threshold show that the GBM model before optimization has better performance, especially in the accuracy, negative class recall rate and negative class F1-score. Although the accuracy of the optimized GBM model has decreased, the Recall and F1-score of positive classes have improved, indicating that efforts have been made to improve the recognition ability of positive classes, but the overall performance is still inferior to that of the pre-optimized model. Therefore, the GBM model before optimization is still selected.

Model Evaluation

Accuracy of GBM model is 0.76, which is relatively high, indicating that the model can be correctly classified in most cases. This is an acceptable level of performance for most application scenarios. The Weighted F1-score of 0.71 is relatively high, which means that the model handles unbalanced classes (especially negative classes) well, taking into account precision and recall. At the same time, when the amount of data is large. If real-time requirements are high, or computing resources are limited, GBM training time may become a bottleneck. There are many other ways to optimize the model and prevent overfitting.

It should be noted that the new feature values input during prediction should be consistent with the feature order and dimension during training, and before predicting, the same preprocessing operations, such as encoding, standardization should be performed on the input new feature.

3.2 Model Analysis, Prediction and Evaluation of “price(€/sqtf)”

Model Training

Since the target variable is a continuous numerical variable, Linear Regression model, Decision Tree Regression model, Random Forest Regression model, Gradient Boosting Regression model and SVR model were used to study the “price(€/sqtf)” and the remaining 10 features in the two DataFrames. The results show that the model effect of the data after processing outliers is better than that of the unprocessed data, indicating that the previous processing of outliers is reasonable, so the processed data will be used for subsequent analysis. Moreover, among the applicable models, Gradient Boosting Regression model and Random Forest Regression model perform well, the RMSE and R^2 of the models are as follows:

```
Random Forest: {'RMSE': np.float64(217.37321682438014), 'R2': 0.44984913085432443}
Gradient Boosting: {'RMSE': np.float64(212.17148249994776), 'R2': 0.4758642769919711}
```

The RMSE of Gradient Boosting is smaller, indicating that its prediction error is smaller than that of random forest, it shows that Gradient Boosting performs better in the accuracy of prediction value. Gradient Boosting has a slightly higher R^2 than random forest, indicating that it can explain the variance of the data slightly better and has relatively strong predictive performance. From the evaluation indexes of these two models, Gradient Boosting is better than Random Forest in both RMSE and R^2. Therefore, Gradient Boosting is a more accurate regression model suitable for predictive tasks with these data.

Then, Trying to optimize data features by removing relatively repetitive variables with multicollinearity. After “bath” was deleted, predicted the residual variable and the target variable again. Then predicted again with removing another feature —“number of bedrooms” , and the two results showed that the interpretation degree of the model declined slightly.

```
Random Forest: {'RMSE': np.float64(220.70012684622753), 'R2': 0.4328800772521719}
Gradient Boosting: {'RMSE': np.float64(213.82792033811089), 'R2': 0.4676484010755593}
```

so I chose to retain these features.

Model Optimization

The two models that performed well were then optimized. RFE recursive feature screening method (select the 5 most important features), feature importance selection method (select the first five important features), PCA dimension reduction method (retain the 5 principal components), logarithm transformation method (logarithm transformation of total_sqft) were used to optimize the Gradient Boosting model and Random Forest model, and the results before and after optimization were compared. It was found that the logarithmic conversion of the Random Forest Regression model resulted in a slight increase in R^2 (from about 0.449 to about 0.450), while other optimizations did not effectively improve R^2 and reduce MSE of the two models, and even slightly decreased the model effect.

Then, the optimization method of logarithmic transformation was selected, and other methods are superimposed successively to further optimize the Random Forest model. However, the results show that after the first logarithmic transformation, the subsequent superimposed optimization weakens the explanatory power of the model.

Log Transformation + RFE + Feature Importance + PCA-MSE: 47726.3067, R2: 0.4443

The model was not optimized effectively.

Finally, the method of Stacking integration was attempted and below is the result:

Mean Squared Error (MSE): 42525.29928682212, and R-squared (R2): 0.5014141596578583

Model Evaluation

In summary, the Stacking integrated approach has the best effect. The Stacking model can explain about 50% of data variation, but MSE shows large errors. This indicates that the model has some capability, but may still need further optimization in some aspects. However, the Stacking method involves many layers of models, and the process of parameter tuning and model selection is complex.

It should be noted that the new feature values input during prediction should be consistent with the feature order and dimension during training, and before predicting, the same preprocessing operations, such as encoding, standardization should be performed on the input new feature.

3.3 Analytical validation of BER’s predictive ability to “renovation_needed”

KNN model, Naive Bayes model and GradientBoostin model are selected to verify the predictive ability of “BER” for “renovation_needed”. The results show that the accuracy and exact recall ratio f1 score are 1.0. Then the gradient rise model is selected for overfitting detection, and the performance of the training set and the test set are compared to determine whether the model is overfitting. If the training set performs significantly better than the test set (with higher accuracy and F1 scores), the model may be overfitting. On the contrary, if the performance of the two is similar, it indicates that the model has good generalization ability. The result printed that “Model seems to be generalizing well. No significant overfitting detected”.

References:

1. Han, J., Kamber, M., & Pei, J. (2011). *Data Mining: Concepts and Techniques* (3rd ed.). Elsevier.

2. Alpaydin, E. (2014). *Introduction to Machine Learning* (3rd ed.). MIT Press.

3. Chandrashekar, G., & Sahin, F. (2014). A Survey on Feature Selection Techniques. *Computer Science Review*, 14, 1-13.

4. Zhou, Z.-H. (2012). *Ensemble Methods: Foundations and Algorithms*. CRC Press.

5. Friedman, J., Hastie, T., & Tibshirani, R. (2001). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (2nd ed.). Springer.

6. James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An Introduction to Statistical Learning: with Applications in R*. Springer.