A logo for college computing

Description automatically generated

**Assessment Cover Page**

|  |  |
| --- | --- |
| *Suzanne Cotter* |  |
| *Sbs24020* |  |
| *Machine Learning* |  |
| *CA1* |  |
| *28th April, 2024* |  |
| *28th April, 2024* |  |

**Declaration**

By submitting this assessment, I confirm that I have read the CCT policy on academic misconduct and understand the implications of submitting work that is not my own or does not appropriately reference material taken from a third party or other source.

I declare it to be my own work and that all material from third parties has been appropriately referenced.

I further confirm that this work has not previously been submitted for assessment by myself or someone else in CCT College Dublin or any other higher education institution.

Contents

[Introduction 1](#_Toc165183129)

[Data Characterization and Preprocessing: 1](#_Toc165183130)

[Analysis of the Two models 3](#_Toc165183131)

[What is the primary purpose of hyperparameter tuning in machine learning? 3](#_Toc165183132)

[Conclusion 3](#_Toc165183133)

[References 4](#_Toc165183134)

## Introduction

Predicting house prices and understanding which housing characteristics are the most important is vital in real estate markets, urban planning and personal finance decision-making.

#### Description of the Problem Domain:

When predicting housing prices, the primary challenge lies in accurately estimating the value of a property based on various features and characteristics. This task is complex due to the many factors influencing housing prices, including location, size, condition economic indicators, and demographic factors. These factors often interact with each other in nonlinear ways, further complicating the prediction process.

Key challenges in predicting housing prices accurately include:

* Data Variability: Housing markets differ a lot from place to place, even within the same city. For example, a house in one neighbourhood might be more expensive than a similar house in another area. Predictive models need to understand these differences while still being useful in different places.
* Feature Selection: Figuring out which details about a house are most important for predicting its price is vital. However, it's not always easy to know which details matter the most or how they work together.
* Data Quality: Making sure that the information we have about houses is accurate and reliable is important for making good predictions. Sometimes, the data we have might be missing or wrong, and that can mess up our predictions if we don't handle it carefully.
* Model Complexity: Predictive models can be simple or complex. Simple models are easy to understand but might miss some important details. Complex models can capture a lot of detail, but they might also make mistakes if they're too complicated.
* Market Dynamics: Housing markets change over time because of things like the economy, government rules, and how people live. Predictive models need to be able to adapt to these changes so they can keep making good predictions.

The Housing Prices dataset is a collection of data about houses in the USA found [here](https://www.kaggle.com/datasets/yasserh/housing-prices-dataset).

The dataset consists of 545 rows, with each row representing a different house, and 13 columns representing different features or attributes of the houses.

It is a perfect subject for this project due to the following reasons:

* It is relevant in the real world as it is about people’s homes and describes features such as size and number of bedrooms.
* It has a good range of features, both numerical, Boolean and categorical which make it perfect for testing and training prediction models on.
* It is large enough and complex enough to showcase various machine learning techniques and offers challenges like categorical variables and feature engineering opportunities.

## Data Characterization and Preprocessing:

Features in the dataset include the following:

* Numeric features: These include different types of variables like 'area (the area of the house in square feet), 'bedrooms' (the number of bedrooms), and 'price' (the price at which the house was sold).
* Categorical features: There is one variable ‘furnishingstatus’
* Boolean features: These include ‘basement’ and ‘hotwaterheating’.

It's important to understand the structure of the dataset, including the number of rows and columns, the data types of each column (numeric, categorical, etc.), and the distribution of values within each column.

#### Data Preprocessing Steps:

* Handling Missing Values: Check for missing values in the dataset. In this case there were none.
* We check for duplicates and again there are none.
* We use one-hot encoding to convert the ‘furnishingstatus’ into 3 Boolean columns which machine learning algorithms can understand and the other Booleans into 1 column each.

Choice of Algorithms

For the prediction of housing prices, which involves predicting a continuous target variable (the price of a house), regression approaches are more suitable than classification approaches.

#### Linear Regression Model with k-fold cross validation

1. Firstly, we start by setting up a model that can predict house prices based on features like the size and location of the house.
2. Cross-Validation: Instead of a single data split, use cross-validation to divide the data into folds. Train the model on some folds and test it on others to assess performance across different subsets of the data.
3. Calculate Metrics: calculate three different metrics for each fold:
   1. Mean Squared Error (MSE): Measures the average squared difference between predicted and actual prices.
   2. Mean Absolute Error (MAE): This also measures the average difference between predicted and actual prices, but it doesn't penalize large errors as heavily as MSE.
   3. R-squared Score (R2): Evaluates how well the model explains variation in the actual house prices. Higher values are better. (Mueller and Luca Massaron, 2021, p. 243-270)

kNN Regression

1. After splitting the data into two parts, one containing the features (X) and one containing the sale prices (y or target), we split it into training and testing sets to assess algorithm performance.
2. Standardizing Features: We make sure all the features (like house sizes and room numbers) are on the same scale, which helps the algorithm understand them better.
3. The k-Nearest Neighbours method looks at the houses closest to the one we're trying to predict and uses their prices to make a guess.
4. Training the Model: We teach our algorithm how to guess house prices by showing it examples from the training data.
5. Making Predictions: Once our algorithm has learned from the training data, we test it on new houses to see if it can predict their prices accurately.
6. Evaluating Performance: Finally, we check how well our algorithm did by comparing its predictions to the actual prices of the houses. We use the same metrics as above. (Mueller and Luca Massaron, 2021, p. 238-242)

## Analysis of the Two models

Why choose kNN and Linear Regression models?

kNN excels when the underlying relationship between features and target variable is non-linear and the data isn’t too large. Linear Regression assumes a linear relationship between features and target variable, which is often suitable for housing price prediction where some features may have a linear impact, i.e. some features have more impact that others on prices. The 2 algorithms therefore offer complementary approaches.

K-fold cross-validation allows for robust evaluation of the model performance as it trains and tests the models on multiple subsets of the data. This provides more reliable estimates of model performance compared to single splits.

We can tune the hyperparameters to optimise performance. For kNN we can change the number of neighbours (k) while with our linear regression we can increase the folds as well as change regularization parameters and feature transformations.

What is the primary purpose of hyperparameter tuning in machine learning?

Hyperparameter tuning is like adjusting the knobs on a machine learning model to make it work better. Just like tuning a guitar string to get the right sound, we tweak these settings to improve how well our model learns from data and predicts outcomes.

It helps to make our models smarter as the hyperparameters control how our model learns and behaves. By finding the best settings, we can make our model smarter and better at making predictions.

They help us avoid Underfitting where the model is too simple and Overfitting where the model is too complicated.

Every problem is unique, and what works well for one might not work for another. By tuning hyperparameters, we can adapt our model to different situations and datasets.

To find the best settings, we use techniques like Grid Search, Random Search, Bayesian Optimization, or even just trying different combinations until we find the one that works best. Overall, hyperparameter tuning is about finding the sweet spot that makes our model perform its best on the tasks we give it.

# Conclusion

There is a very good possibility that there is an issue with my code for kNN and Linear Regression introduced by the for loop to test the 3 training splits, .2, .25 and .3. Before I added the For loop, I was getting values around 0.62 to 0.64 as seen in the final piece of code of the GridSearchCV.

Scores of approximately 0.53 suggests that the models have moderate predictive power. They can capture some of the relationships between the independent and dependent variables but may not fully capture the complexity of the data.

After using GridSearchCV to tweak the model settings for both linear regression and KNN, we saw better performance in terms of how well the models predict outcomes. The linear regression model ended up with a slightly higher score of 0.64, compared to the KNN model's score of 0.62. However, both models did well in predicting outcomes.

The linear regression model shows consistent performance across different training split ratios, indicating robustness to changes in the size of the training dataset.

The KNN model shows improved performance with larger training datasets, suggesting that it benefits from more data for training. This is a common characteristic of KNN, as it relies on the local similarity of data points and can potentially capture more complex patterns with larger datasets.

# References

De Cock, D. (2023). *Ames Iowa: Alternative to the Boston Housing Data Set*. [online] Amstat.org. Available at: https://jse.amstat.org/v19n3/decock/DataDocumentation.txt.

Yasser H, M. (2022). *Housing Prices Dataset*. [online] www.kaggle.com. Available at: <https://www.kaggle.com/datasets/yasserh/housing-prices-dataset>.

Mueller, J. and Luca Massaron (2021). *Machine learning for dummies*. 2nd ed. Hoboken, New Jersey: John Wiley & Sons, Inc.