FINAL PROJECT PAPER

Project Title:

Predicting Housing Prices using Publicly Available Datasets
and Machine Learning

Course: Data Science in Python

Word count: 1944 words

Number of Pages: 8 pages

Authors: Nguyen Le, Zachary Pao, Ashtosh Bhandari

Table of Contents

1. INTRODUCTION

Property appraisal has always been one of the most taxing tasks because of the low level of certainty in the valuations, especially in a rapidly changing property market nowadays. Valuing properties is a challenging job since it is a fairly imprecise discipline due to reliance on the unique features of the real estate property such as its location, size, or property type. Achieving consistency in real estate appraisal is difficult; hence why, we wanted to build a solution that is capable of predicting the house prices better, and perhaps with greater consistency and precision, than individual appraisers.

For our project, we will be building a Machine Learning model to predict house prices in Melbourne using 'Melbourne Housing Snapshot' dataset that can be found on Kaggle[1]. The dataset consists of metrics such as number of rooms, land size, type of housing, and others for each suburb in Melbourne. The ultimate goal of the project is to build a prediction engine that would perform this task in place of a human with greater accuracy.

We initially wanted to work with a dataset that represents housing in Australia; however, it was too large to handle, so we chose Melbourne as the location of investigation for our project instead. To explore this issue further, we will work with scikit-learn, a machine learning library used in Python, to perform models such as K-Nearest Neighbors and Decision Trees. For the parameters for our approach, we will vary k for k-nearest neighbors to maximize the accuracy and, similarly, the maximum depth for decision trees. I will discuss the methodology in more detail further into the research paper.

2. BACKGROUND INFORMATION

2.1 PANDAS DATAFRAMES

2.1.1 What is pandas?

pandas is a Python library that is typically used in data analysis, cleaning, and manipulation[2]. The pandas module provides all kinds of functionality for dealing with tables of data. It allows us to analyze our input data and draw out conclusions using statistics. It can also clean datasets for us, which means to make them readable and relevant. Using the pandas module, we can gather a lot of information about the data we use such as correlation between columns of the dataset or finding the max value. One of the fundamental objects that we will be interacting with pandas is the DataFrames object, which is a highly annotated array with mixed data types[3].

2.1.2 Working with pandas DataFrames

pandas DataFrames also have functionality for dealing with more messy data, with missing values, and different data types mixed in the same object. The data that is imported to DataFrames often are either .tsv files or .csv files. For our project, the dataset we will be using is imported as a .csv (Comma Separated Values) file in the ZIP format. .csv files are easily readable as text as they contain our data values in a table separated by commas. DataFrames has a lot of

[1] www.kaggle.com. (n.d.). *Melbourne Housing Snapshot*. [online] Available at: https://www.kaggle.com/datasets/dansbecker/melbourne-housing-snapshot.

[2] www.w3schools.com. (n.d.). *Pandas Tutorial*. [online] Available at: https://www.w3schools.com/python/pandas/default.asp.

[3] Gold, K. (2022). *DS 110 Lecture 11 — Pandas*. [online] learn.bu.edu

useful properties such as loc property, which lets us retrieve and look up values in our data by index and column[4]. It also has methods such as .drop(), which is part of cleaning and processing our data for the calculations.

2.2 MACHINE LEARNING WITH SCIKIT-LEARN

2.2.1 What is scikit-learn?

scikit-learn is a Python library that is used in the fields of Machine Learning. It offers many algorithms and methods that is used in Machine Learning such as K-Nearest Neighbors or Decision Trees. Like pandas, it's a library that can be used for data processing; however, it is more specific to Machine Learning, and thus it includes everything from data manipulation to processing metrics.

2.2.2 Splitting Data for the ML Algorithm

Data splitting is very important in Machine Learning. It is done to avoid overfitting, which means when the classifier model memorizes the data pattern in the training dataset but fails to generalize to unseen examples. Thus, the model is useful only to its training data but does not perform accurately in any other dataset, and therefore, fails to predict future observations reliably[5]. In general, most classifiers have this issue. Hence, the performed experiment should not evaluate solely on the training data, or the results won't give a sense of how well the model does on genuinely new data points. Therefore, the data needs to be split into training and testing data. It's best to do this blindly and randomly, because the distributions of training and testing data should match. The training set is the portion of the data that is used to train the model. Thus, our classifier model will learn from this data set. The testing set is a part of the data that is tested in the final classifier model and is compared against the previous sets of data. Thus, the testing set acts as a point of evaluation for the final algorithm[6]. Scikit-learn's train_test_split() function performs exactly this task of separating out randomly selected examples for testing.

2.2.3 Data Validation

Despite splitting the data helps mitigate the effects of overfitting, it is still possible for it to happen when we try to update our parameters to see if any improvements can be made. Thus, we need to split the training data yet again, splitting off a portion of the training data to be validation data, allowing us to go back and forth trying to improve your performance without worrying that test set knowledge affecting it. This is called cross-validation. Cross-validation set is a dataset of examples used to change learning process parameters. This dataset ranks the model's accuracy and helps with model selection[7]. Scikit-learn's cross_val_score() function performs exactly this.

2.2.4 K-Nearest Neighbors

To find the correct classification of a point, its neighbors is consulted. K-Nearest Neighbors is a classification method for estimating the likelihood that a data point will become a member of a group based on what group the data points nearest to it belong to[8]. Euclidean distance is typically used to determine this.

---

[4] Gold, K. (2022). *DS 110 Lecture 11 — Pandas*. [online] learn.bu.edu

[5] Gold, K. (2022). *DS 110 Lecture 20 — Introduction to Supervised Machine Learning*. [online] learn.bu.edu

[6] SearchEnterpriseAI. (n.d.). *What is data splitting and why is it important?* [online] Available at: https://www.techtarget.com/searchenterpriseai/definition/data-splitting.

[7] Gold, K. (2022). *DS 110 Lecture 20 — Introduction to Supervised Machine Learning*. [online] learn.bu.edu

[8] Joby, A. (2021). *What Is K-Nearest Neighbor? An ML Algorithm to Classify Data*. [online] learn.g2.com. Available at: https://learn.g2.com/k-nearest-neighbor.

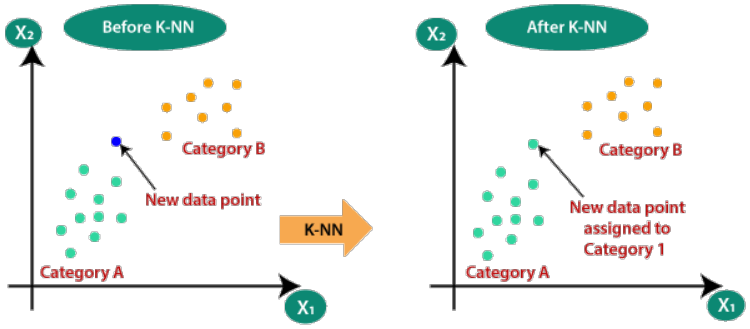


Figure 2.2.4: Illustration of KNN algorithm in action for Machine Learning (Source: Javatpoint[9])

2.2.5 Working with Decision Trees

Decision trees is a ML method that builds a classifier in response to data that is distinct from the data. The final form of this classifier is a tree. The interior nodes of the tree store yes or no questions, and the leaves of the tree store classifications[10].

When classifying, the algorithm starts at the top, asks the first question, and moves left or right as a result of the answer being yes or no. The questions continue until a leaf is reached, and that is the classification. There could be many more leaves than possible classifications, which means there can be multiple ways to output the same classification.

Decision trees have the advantage of being inspectable and transparent compared to other methods. It's easy to say why something was classified.

3. EXPERIMENTAL METHODOLOGY

3.1 Independent Variables

Independent variable of the experiment refers to what will be changed/updated in the experiment to create our results. Our dataset consists of a variety of metrics, but since we are trying to predict a property valuation, our model will be using **number of rooms, land size, number of cars that can be parked, number of bathrooms, and number of bedrooms** for each house in each suburb in Melbourne as a point of reference to calculate the predicted price for those houses.

3.2 Dependent Variables

Dependent variable of the experiment refers to the variable being measured, which in this experiment is the **predicted house price and the accuracy score.** The predicted prices, which is computed using the independent variables, will be compared to an arbitrary value (which as of now is 1'100'000) and a Boolean variable would be returned if the predicted price meets this value. Then the accuracy scores will also be measured for the KNN and Decision Tree to see how well each model performed in predicting the pricing.

---

[9] JavaTpoint (n.d.). *K-Nearest Neighbor(KNN) Algorithm for Machine Learning - Javatpoint*. [online] www.javatpoint.com. Available at: https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning.

[10] Gold, K. (2022). *DS 110 Lecture 21 — Decision Trees*. [online] learn.bu.edu

3.3 Controlled Variables

What was kept constant throughout the experiment

Variable	Description	Specifications (if applicable)
Integrated Development Environment (IDE)	We will be running the project on a single web IDE for Python	**IDE:** Google Colaboratory (Colab)
Programming Language to write the experiment	Python	**Version:** Python 3.7 (default for Colab)
k value for K-Nearest Neighbors	Different k values are tested to prevent overfitting of data	Incremented in loop
max depth for Decision Tree	Controls how deep the tree can grow	max depth = 3

3.4 Experimental Procedure

The procedure for the experiment is as follows:

1. Load in the dataset
2. Clean the data using .dropna() method
3. Splitting data into training and test data
4. Seaborn Data Visualization
5. Performing K-Nearest Neighbor
6. Determining the cross-validation score for the KNN algorithm
7. Creating Decision Tree and cross-validating it

4. EXPERIMENTAL RESULTS

4.1 Data Collection and Processing

Below is the Boolean output returned of the predicted prices as compared to an expected arbitrary price value (as of now it is set to 1'100'000)

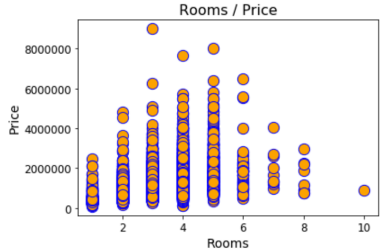
```
0         True
1        False
2         True
3        False
4         True
        ...
13575     True
13576    False
13577     True
13578     True
13579     True
Name: Price, Length: 13518, dtype: bool
```

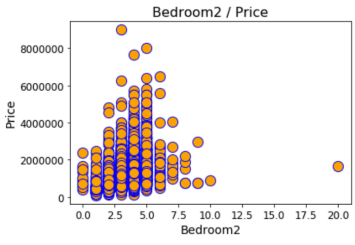
In addition to that, we also have a table of the accuracy scores for each classifier in the last ran version of the classifier models, which to determines how well each model performed in predicting the pricing.

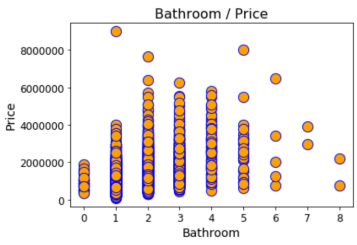
	K-Nearest Neighbor	Decision Tree
Accuracy Score	0.6823855755894591	0.6693786982248521

4.2 Visualization of Collected Data

Seaborn is a Python library that is used for data visualization. Here, we used Seaborn to visualize our data on a graph so that it is easier to see the trends in the data.







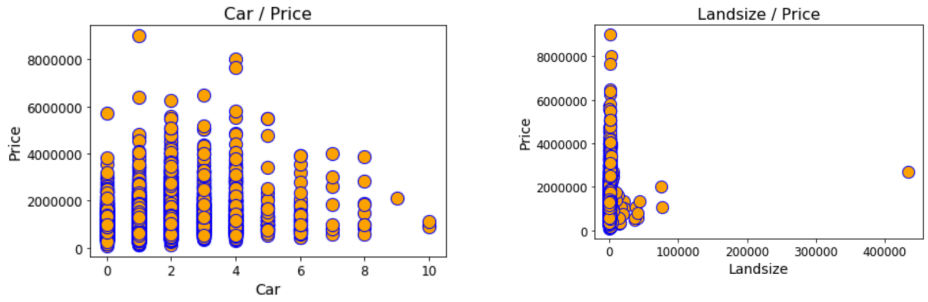


Figure 4.2.1: Seaborn Visualization of Correlation between Price and Other Metrics

Below is the decision tree created from our classifier model. With this visualization, we can understand why certain attributes used in property valuation was classified.

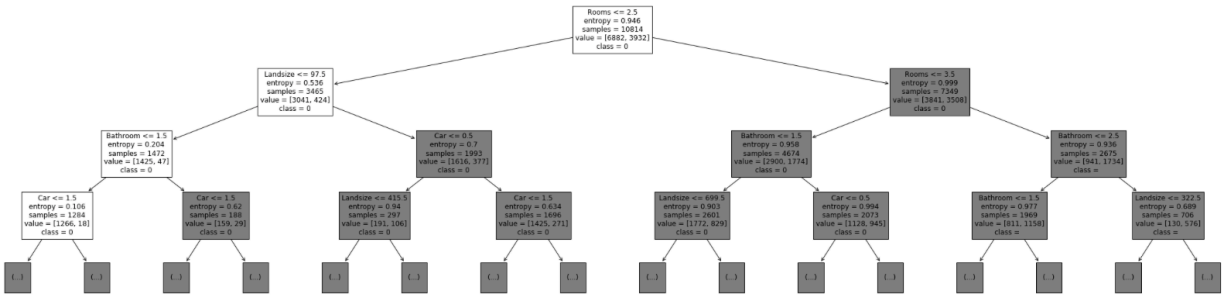


Figure 4.2.2: Decision Tree

4.3 Discussion of Results

From the visualizations graphed using the seaborn module (Fig 4.2.1.), we can see that there is a strong correlation between predicted price of the house and numbers of rooms, number of bathrooms, and number of cars that can be parked. On the other hand, there still somewhat is a correlation between the predicted price of the house and the number of bedrooms; however, the correlation here is not as strong as the one seen earlier. Moreover, looking at the land size in relation to predicted price graph, we can see that there is no correlation between the two metrics.

Turning our attention to the accuracy scores achieved by the K-Nearest Neighbor and the Decision Trees classifier models, we can see that their accuracy scores basically fall between 65-70%. This is an a fairly mediocre accuracy for both the classifier models in terms of figuring out a property valuation. However, at least this score confirms that the possibility of overfitting has been eliminated during the experiment as the score is not suspiciously too high, which makes our results most certainly more reliable.

5. CONCLUSION

To conclude, our classifiers have been somewhat accurate. Overall, the experiment seemed to work the way it was intended to, and there weren't many sources of error. There was, however, one issue with some of the metrics used in the classifier models that we had to drop during the data cleaning process after identifying the problem. The metrics Longitude and Latitude of the houses were dropped because if they were kept and used along with our other independent variables to predict our property valuations, the decisions in every node of the decision tree for some reason would only be based on only these two variables, and this would be a case of overfitting. Thus, the decision tree would not take any other variables into consideration for the price prediction if the longitude and latitude of the house was kept, hence, making our classifier model obsolete and overfitted.

6. WORKS CITED

Gold, K. (2022). *DS 110 Lecture 11 — Pandas*. [online] learn.bu.edu

Gold, K. (2022). *DS 110 Lecture 20 — Introduction to Supervised Machine Learning*. [online] learn.bu.edu

Gold, K. (2022). *DS 110 Lecture 21 — Decision Trees*. [online] learn.bu.edu

Joby, A. (2021). *What Is K-Nearest Neighbor? An ML Algorithm to Classify Data*. [online] learn.g2.com. Available at: https://learn.g2.com/k-nearest-neighbor.

JavaTpoint (n.d.). *K-Nearest Neighbor(KNN) Algorithm for Machine Learning - Javatpoint*. [online] www.javatpoint.com. Available at: https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning.

SearchEnterpriseAI. (n.d.). *What is data splitting and why is it important?* [online] Available at: https://www.techtarget.com/searchenterpriseai/definition/data-splitting.

www.kaggle.com. (n.d.). *Melbourne Housing Snapshot*. [online] Available at: https://www.kaggle.com/datasets/dansbecker/melbourne-housing-snapshot.

www.w3schools.com. (n.d.). *Pandas Tutorial*. [online] Available at: https://www.w3schools.com/python/pandas/default.asp.