What Makes a House Worth More?

Project Option: Problem

Jake McCoy s1106263
Theo Lavandier s1103617

Course: Data Mining
Group 135
Teachers: Tom Hesks, Tom Claassen

January, 2023

Index

Summary

We want to be able to predict the average price of a neighbourhood in California based on the characteristics of the given census block. We will be using the California Housing dataset which is obtained through the StatLib repository (but this dataset was originally derived from the 1990 U.S census).

We will be using two different algorithms (in python) a ***Regression Decision Tree*** and a ***KNN Model*** to help predict the housing price of a given census block. Both algorithms will help see what attributes affect an average housing price more or less than another. We will be using ***K-Fold cross validation*** to train both these given algorithms so we make sure we are using these algorithms to their fullest capability.

Application Domain and the Research Problem

Application Domain

When selling a house most people tend to go to a housing agent (also known as a Realtor) for assistance in all aspects of selling a house. This could be advertisement, paper work assistance/guidance and more importantly the appraisal of a house.

Appraisals can be a very time consuming process since they have to go to the house and view it themselves documenting all problems/concerns a person may have when looking to buy a house (and what aspects may improve a house's overall price). The neighbourhood also plays a large role in the pricing of a given house and can usually give us a rough estimate of how much a house may sell for (Since a house will more likely be worth more if all the surrounding houses are also worth a lot) based on its neighbourhood.

We want to help be able to predict a house's price in a given neighbourhood based on specific attributes of that neighbourhood. More importantly be able to see what particular attributes may affect the price more than another so we know what houses are worth more. So we can correctly estimate a house's price before an in depth appraisal occurs to help speed up the process of selling one's house.

Research Problem

There are many attributes to take into account when trying to estimate the price of a neighbourhood like Medium income, Population, House Age, Average Bedrooms etc. We want to see which of these attributes will affect a house's price (Either positively or negatively) for example, if a neighbourhood has a High Medium Income and a low house age then the houses in that area are worth more (This is just an example of what relations we are looking for).

In having so many different attributes we need to use some algorithms as we cannot see patterns in 8-dimensional data as easily as others when trying to determine an average housing price for a given area.

Previous Attempts

There have been some attempts to use some Machine Learning and Neural Networks to determine housing prices before like in the paper ‘House price prediction using Machine Learning and Neural Networks’.

> Varma, A. *et al.* (2018) *House price prediction using Machine Learning and Neural Networks*, *IEEE Xplore*. IEEE. Available at: https://ieeexplore.ieee.org/abstract/document/8473231 (Accessed: January 3, 2023).

The difference between what we are trying to do and the paper cited above is that the system created is to help people find houses in areas they want to purchase based on a list of desirable attributes. Whilst we want to simply predict a house's price for a real estate agent to get an idea of what a house sells for.

Data Set and Preprocessing

Data Set

The data set was obtained from the StatLib repository (Which you can access at https://www.dcc.fc.up.pt/~ltorgo/Regression/cal_housing.html). This data set was derived from the 1990 U.S. census with each data point representing a neighbourhood ‘block’ typically between 600 to 3000 people depending on the area. There are many attributes in the data (With 8 continuous numerical attributes) and below I will describe each data point so we make sure we understand what attributes we will be looking at:

- MedInc: Median income of the census block
- HouseAge Medium age of a house in the census block
- AveRooms Average number of rooms per household in the census block
- AveBedrms Average number of bedrooms per household in the census block
- Population Population of the entire census block
- AveOccup Average number of people residing in a household
- Latitude Block group latitude position
- Longitude Block group longitude position

These are all the attributes that will be analysing to see what has the greatest effect on a neighbourhood's price to help us predict a house's value before a indepth appraisal can be done by a realtor.

We should also note that in the dataset we have 935 values that are collapsed to 5.000001, but in reality it’s supposed to be higher than this. This means we can’t accurately see a neighbourhood's average price above 5 million which will limit our observations on houses above this price.

Preprocessing

Incorrect/Inaccurate Data

Our first step in preprocessing the data is to remove all data points that have an average housing price greater than 5.000001. This is because we can't accurately tell how much an attribute will affect the price due to the fact we don't know it's true price (We only can tell that it's worth more than 5 million). Removing these data points removes 935 values out of the total 20650 ($\approx 4.5\%$ of the data) which will help us be able to identify the attributes that affect the true price of a neighbourhood.

It is important we do this first since we want these data points to have no influence at all towards the other data points (especially during standardisation!). If we removed these data points after standardisation we won't have correctly standardised data since we have removed a portion of our data that may affect what the mean is.

We implemented this (in python) simply by looping through all of our data to see which housing prices were above 5 (million dollars) and saving those indexes which we deleted. You can see this in Cell 3 of the housing_prices_predictions.ipynb file

Standardization

Standardization is necessary for us since we have 8 different attributes and thousands of data points so making sure all our data has been set to a standard in each of our attributes means all of our data is in harmony (that they are all correct and are written in the same way so our algorithm doesn't get confused).

It was a fairly simple choice on how we were to implement this by using '*sklearn.preprocessing*' package '*StandardScaler*' which you can see how this is done in the housing_prices_predictions.ipynb file in Cell 3.

Sampling

Sampling is also part of our preprocessing since processing the entire set of data becomes time consuming and requires more computational power inorder to complete any kind of processing. We settled on sampling without replacement over sampling with replacement for the simple reason that it requires less computational power/resources to replace data we take out. This allows us to take larger samples of our data and still view it as multiple individual samples.

Sampling was simple to implement using '*sklearn.utlis*' package '*random*' which selects a given number of random integers (defined by the attribute n_samples) from a given range (defined by the attribute n_population). You can see this sampling implemented in the housing_prices_predictions.ipynb file in Cell 3

Approach/Method

Our Approach can be broken up into 4 simple steps (which each will get its own title about what we do within each step):

1. General analysis of the data (Graphs and PCA)
2. Creating prediction models (Decision Tree and KNN)
3. Trying to find the best parameters for the models (By analysing the efficiency of each model with different parameters, using R2 score and Mean Absolute Error)
4. Compare our two models, their qualities and their defaults (Accuracy, MAE, R2 Score and more methods that we need to find since this is a regression analysis)

General Analysis of the Data

PCA

Principal Component Analysis is a very useful technique when trying to reduce high dimensionality data but still holding trends and patterns we want to analyse. Especially since we are trying to see what attributes lead to an increase in a house's price (More or less than others). There are some downsides to PCA when trying to analyse data:

- Can only discover linear relations since the principal components are linear combinations of features.
- Loss of information can occur since we are compressing the data which can result in the loss of certain patterns we may see with other methods.

This was fairly simple to implement since the data is already standardized by using '*sklearn.decomposition*' with the package '*PCA*'. We focus on the first 3 PC's by setting n_components = 3 when initialising PCA(). You can see how this is implemented in the housing_prices_predictions.ipynb file in Cell 4. In figures 1 and 2 in Cell 4 you can see bar graphs of the explained variance ratio of each of the 3 PC's (and in Figure 1 you can also see the cumulative explained variance).

We can print out the values of each Principal Component to see what attributes affect the variance in each component the most, which we have done in Cell 5 in the housing_prices_predictions.ipynb file. We will quickly talk about each PC and what attributes are affecting this and why this might be the case.

Principle Component 1

In PC1 we can see that the most variance lies in the last two attributes which are Latitude and Longitude (-0.6865, 0.6849). This makes sense since the placement of a house can play a big role in its worth e.g. houses along the beach are more desirable therefore more valuable than houses inland. The combination of these two points makes a coordinate for an area which shows us the placement of a house plays a big role in the variance of pricing.

Principle Component 2

In PC2 the largest value we have is -0.9829 (Since a PC is a vector the negative symbol indicates a direction and not lesser of a value) which is the AveOccup. This may be because more people living in an area may represent a poorer area e.g. people who live in big apartment blocks in comparison to stand alone houses.

Principle Component 3

In PC3 the first 3 attributes have the largest values (0.4532, -0.4707, 0.6087) which represent Medium Income, House Age and Average Rooms. These are all almost self explanatory in why they influence a house's value. People who make more money will live in different houses to people who make less money. An older house is usually not as valuable as a newer house (An extreme example would be comparing a 40 year old house to one made last year very rarely will the older house be worth more). The average rooms would also could indicate the size of a house since usually bigger houses are worth more (In a real world scenario)

Initial Graphs

First we simply plotted a scatter matrix to see if we could visually spot any trends/patterns that we may want to explore further in our report. In Cell 6 of the housing_prices_predictions.ipynb file we can see a couple of patterns of note. We can observe three things of note:

1. There appears to be some linear association with MedInc and MedPrice that the higher the MedInc the higher the MedPrice
2. There is some linear association between AveRooms and MedInc (The higher AveRooms the Higher the MedInc is)
3. Plotting Latitude and Longitude together plots where each data point lies in California. You can Look at Cell 7 of the provided file to see the data points plotted on an interactive map of california

We can further confirm points 1 and 2 made above by plotting a heat map correlation matrix (Figure 3 in the provided file) of the data using the seaborn library which you can look at in the output of Cell 5. We get a value of the linear association and for the MedPrice to MedInc a value of 0.65 which is the highest linear association we get. AveRooms to MedInc also has a value of 0.33 which isn't a super strong association but one we can't ignore.

From these initial graphs we can see there is some linear association between MedInc to MedPrice and MedInc to AveRooms. This makes sense when you think about it, the higher average income people are able to purchase more expensive homes and having more rooms is usually a good indicator of a higher income

.

Creating Prediction Models

K-Nearest Neighbours

A K-Nearest Neighbours model is a very simplistic idea, compare a new data entry to previously known data to predict its value. We chose this model since we wanted a lazy-learning model to see if we can make an accurate model simply by comparing it to known data points rather than how other models like a Regression Tree Model work (If we can use a simpler model that requires less computational power and training we should try to create it).

Regression Tree Model

Our second approach to making a model to predict a house's price will be using a Regression Tree Model. This will help us to be able to predict the value of houses which might not have a linear relationships (Something regression tree models are good at). Again our hope is that in training our regression tree model (Using our already preprocessed data) to predict housing prices we can help real estate agents predict a house's value based on its attributes before a proper appraisal can occur. Defining a decision tree is very easy using the package '*sklearn.tree*' and more importantly there packages '*DecisionTreeRegressor*' and 'plot_tree'. In Cell 8 of the housing_prices_predictions.ipynb.

Finding the best parameters for our models

Best Parameters for K-Nearest Neighbours Model

Number of Neighbours

The only parameter we pass to a KNN model is the value of K (The number of neighbours we will use to predict the given data point) which we can use K-Fold Cross-Validation to determine the optimal value for K (We chose this method since all the data is used for both training and testing purposes which helps us to avoid overfitting of our model since we have a more 'general' model).

In Cell 11 of the housing_prices_predictions.ipynb file you can see our implementation of K-Fold Cross-Validation which plots the error rate of different models with different amounts of K (Nearest Neighbours). From the output of this Cell we can see that 11 and 12 gives us the lowest error rate even if it's by only a small amount so we will set the model to consider the nearest 12 neighbours.

Best Parameters for Regression Tree Model

Maximum Depth

To find the optimal depth of our tree we use K-Fold Cross Validation so that all data is used for both training and testing purposes of our model (min_samples_split is set to its default at 2 currently). We try maximum depths from 1 to 20 (Which you can see in housing_prices_predictions.ipynb Cell 9 , Figure 7) to determine what depth gives us the least amount of errors. This turns out to be at a depth of 9 with an error rate of ≈0.27. Any smaller depth or higher depth ends up in more errors so the maximum depth will be set to 9.

Maximum Samples before Split

Now knowing our optimal depth we will use the same method above (K-Fold Cross Validation) but instead of trying depths from 1 to 20 we will try a range of different minimum number of sample splits (With the maximum depth now set to 9). These will be [2, 20, 40, 60, 80, 100, 120, 150, 200] (A large range of from the minimum to well above minimum). If you look at Cell 10 in the file and at Figure 8 you can see the minimum value split is our 5th value which for us is a value of 80.

With a smaller depth and a bigger minimum sample split value, our regression tree is more easily represented (see cell 13), while performing better than our decision tree with random parameters. It also allows better performance in terms of computation.

Comparison of Models

To compare our models we will use two different metrics for our evaluation, the first called R2 Score (R-Squared Score) and the Mean Absolute Error (MAE). We selected this since the MAE will tell us the average distance our model predicts to the actual value across all data points (A value of how well our model works). The R2 Score gives us a percentage score of how accurate our model is on a scale from 0 to 100 (A rating from 0 to 100 of how well our model works). By using these two comparison methods we first, get a value on how accurate our model is and secondly, a percentage value of how accurate our model is (Which we will represent as a percentage to 2 decimal places).

In the housing_prices_predictions.ipynb file we have implemented these evaluation techniques using the ‘sklearn.metrics’ package which does a lot of the heavy lifting for us in terms of the implementation in python. You can run Cell 15 in the file (After running all cells before) to generate the scores:

```
Regression Tree Model:
Mean absolute error:  0.43     (2 significant figures)
Mean absolute error:  0.4261080106505443
R2 Score:             61.0%  (2 significant figures)
R2 Score:             0.6097963831819633

K-Nearest Neighbours Model
Mean absolute error:  0.43     (2 significant figures)
Mean absolute error:  0.4318257913223141
R2 Score:             61.0%  (2 significant figures)
R2 Score:             0.6076240694086033
```

We can see that the values above are identical (when we look at the first 2 significant figures) at 0.43 for the MAE and 61% for the R2 Score. We printed them to their fullest float value that python as well to allow us to see how different they are. We can see the differences are negligible since we are talking such small fractions of a difference that the choice of an algorithm comes to fundamental differences in a Regression Tree Model and a K-Nearest Neighbour Model.

KNN Models are simple, Instance-based learning models that require no training. Whilst Regression Trees are effective when capturing non-linear relations and can be displayed visually a lot easier than a high dimensional KNN model like ours. In our opinion we would select the KNN Model for a few reasons:

1. Small changes in the data can really affect final estimations in a Tree model
2. KNN removes the worry for overfitting of the data
3. Requires no training before the model can be used
4. Are simple to understand

Representation of the prediction

To graphically represent the predictions of our two models, we have created a scatter plot representing the predicted values as well as the actual values. On this plot we have represented the residual error of each prediction with black lines. This allows us to see if the predictions of our models are good on random data points (see cell 14 of the jupyter notebook, figure 11).

Results Analysis

We managed to complete everything we wanted to in our report (Which you can read in the Summary). We are able to predict the average price of a neighbourhood in California based on the given data set using our KNN Model implemented in python. We were also able to identify features in a neighbourhood that help to improve the price of a neighbourhood (The Medium Income and the Average Number of Rooms). The only thing that happened we did not see coming at all was both models we would create and optimise would have an almost identical accuracy (accuracy here referring to the MAE and R2 Score). This ment our decision on a model came down to pure fundamentals of the two models rather than the outcome/accuracy of the models.

Future Improvements

We could consider different types of regression models (Linear Regression, Neural Networks, etc.) in search for another model that may outperform our two current models. More kinds of accuracy methods (More than MAE and R2 Score) may have also been insightful to see if there are any more differences in our models that our methods may have missed which may have led to a different conclusion when choosing the 2 algorithms. A more up to date dataset would also be helpful since with our current dataset we can predict neighbourhood values from 1990. With the housing crisis the world is currently in the neighbourhood prices may have vastly changed from what they were valued in 1990. In short:

1. Consider different regression models as well, like Neural Network which is known to be very efficient in a lot of regression tasks.
2. Use more accuracy methods for comparisons of models to have a more informed choice.
3. An up to date dataset of California so that we may accurately predict housing prices with current world trends.

Appendix

Jupyter Notebook File: housing_prices_predictions.ipynb