Predicting the Cost of Housing

Joe Dobrowolski
Northeastern University, Boston, MA
dobrowolski.j@northeastern.edu

Abstract

In this project, the goal was to try to get the highest accuracy in predictions based on data in the Boston dataset using linear regression. Linear Regression models were trained on features based on correlation thresholds to see which features had the biggest influence. Initial observations made LSTAT and RM seem like the best candidates, however when comparing results, using those features alone was only second best to using all of the data. Results were further improved when removing all rows with MEDV of 50, as any values above this was truncated to 50 anyways, and results were improved further by removing a few rows containing some outliers. In this reduced dataset, the score went up from about .54 to .85. When testing the model that was trained on this reduced data but adding back in rows with MEDV of 50 and truncating the predictions, the improvement was less drastic, but still noticeably improved to .64, however adding the outliers back in seemed to generally negate this improvement.

Introduction

Prices of housing can vary greatly, even within a single region depending on several factors. It can be useful to make estimates on the median housing prices in a given area for tasks such as appraising a home or looking for areas that may fall within a limited budget. Using census data, it may be possible to predict the values of homes in a given area based on the features that identify that area. While there may be many available homes in a given region, the value of homes that are currently owned is also important, either when considering moving to the area, or when appraising nearby homes so that they aren't overvalued or undervalued compared to neighbors or based on other surrounding features.

While it may seem simple on the surface to look at certain features and intuitively associate them with housing prices, reality becomes a lot more complicated. The dataset used here is the Boston Housing Prices dataset, which contains 14 features, including the median value of the houses that are owned and is the number that I want to predict throughout these experiments.

My goal is to see what the most important features are in predicting home values, as regardless of the age of this dataset, this can still be used today if the same techniques are applied to newer datasets. My aim is to focus on how correlated each feature is to the value and how much this correlation matters when making predictions. I also want to see if better results can be found by removing specific rows from our training data.

Background

The Boston Housing Prices dataset is widely used as an introduction to Linear Regression as the small size of only 506 rows makes it easier to run on a less powerful machine and having no null values makes it great for having access to all data included. That said, there are problems with having such a small dataset: removing rows could result in removing a significant number of entries that are no longer considered and may bias the model towards what the data is shaped towards when removing these points.

There are additional flaws with the dataset that should be addressed as well. The most obvious is the hard limit of the MEDV data (MEDV = median housing value in 1000s) at 50 – analyzing the data reveals a significant portion of the values at exactly 50, which is a bad sign for making estimates as we don't know how much over 50 the actual results are which may skew the predictions to shoot lower than the true values. This also appears to be true with age being capped at 100. The data itself is also quite old at this point – good luck finding a closet in Boston for under $50k now, and determining housing prices value based on the portion of "blacks" by town would likely be a controversial choice, to put it lightly. There is also some incorrect data that is off by up to 22.62% from the true values, however correcting these mistakes does not seem to have any significant impact on the results (Cantaro, M. 2020). Rented homes are also

left out (Cantaro, M. 2020) which may make a difference in a modern setting, especially in Boston, where a large amount of the residents are living in apartments that are being rented compared to when this data was collected nearly 50 years ago. All that said, it is worth analyzing this data now as the techniques involved in making the predictions may still be applied to more modern data for similar purposes.

As mentioned earlier, this data is often handled by linear regression, so it is worth discussing exactly what linear regression is and why it is a good choice for this dataset. Breaking down the name, linear means it is based on a line, and a regression algorithm handles continues results, as opposed to classification algorithms which may predict a value that is part of a set of specific categorical data. In the context of this data, a continuous prediction might lead to a value like "22.3" while a similar prediction being classified may result in something like "20-30k range". More specifically defined, linear regression can be written with this formula:

$$y = w_1 x + w_0$$

where y is the prediction and the w values (Russell, P. and Norvig, S. 1995) are what we learn based off of the training data. This formula mirrors the formula for a line, y = mx + b, so what this essentially accomplishes is finding a line that best fits the general data and can be used to make a guess along the x axis the predicts a result on the line. There are a couple techniques to measure the error also referred to as the loss, which is what is used to adjust the line learned by the training data. One of these methods is the mean absolute error, which is the average of the distances from each true value to the line (Russell, P. and Norvig, S. 1995).

$$\text{Absolute value loss: } L_1(y, \hat{y}) = |y - \hat{y}|$$

However, there is a more popular option that ends up punishing outliers as they get further out by squaring the error, called Means Squared Error (Russell, P. and Norvig, S. 1995):

$$\text{Squared error loss: } L_2(y, \hat{y}) = (y - \hat{y})^2$$

According to documentation for sklearn, the squared error is used to train the built in linear regression model[1]. One thing that is crucial to having a successful linear regression based prediction is the correlation – essentially how closely the features resemble a line on a scatter plot. A correlation of 1 means that every point falls exactly in line with each other, while 0 is closer to randomness. That said, looking at the data can help identify patterns that may not yield a high correlation linearly, but show distinct patterns that are not random, such as curves and clusters. Other methods can use a polynomial formula to fit "wavy" data more closely, however this is a risk of overfitting where using a simple line may in fact be the better predictor. I will go into more detail in the other sections, but the linear model works well with this data set as LSTAT and RM have a fairly high linear correlation to MEDV.

The pseudocode for my testing algorithms is as follows:

```
1.  Def train_and_test_model(data, list of thresholds)
2.  {
3.      results = []
4.      For threshold in thresholds do:
5.      {
6.          Get_correlated_data(data, threshold)
7.          X = data.non_target_features
8.          y = data.target
9.
10.         X_train, X_test, y_train, y_test = split(X, y)
11.         LinearModel.train(X_train, y_train)
12.
13.         predictions = LinearModel.predict(x_test)
14.         result = get_scores(predictions, y_test)
15.         results.append(result)
16.     }
17.     return results
18. }
```

Related Work

As mentioned previously, using a classifier may have been another valid method for obtaining similar results, but rather than predicting a value based on the test data, it would assign it to a range where that data may fall. Depending on how large the error is with a continuous prediction, this may result in higher "accuracy" scores. This would be useful for homebuyers, as they are not typically looking for a very precise dollar amount, but a general range of prices in their budget when considering areas to move into. I chose not to experiment with this method because for the purpose of this project, I wanted to try to get the most accurate price predictions and the results are likely more interesting than categorizing the data ranges when detecting smaller changes based on trimming the test data.

Another approach may have been trying nonlinear regression, using polynomials to get curves that may better fit the data. In addition to time constraints and lack of expertise, one reason that I did not pursue this method is that the bulk of the features did not have any apparent strong correlation to MEDV. I did note that there appeared to be a slight curve in the LSTAT correlation data, however this curve was not enough that polynomial regression would have helped much, and the RM feature was strongly linear, it seemed to make sense to stick with linear for now, but would further explore different regression models if I were to repeat this project.

Mentioned later in this paper, I would also like to explore other more hidden correlations of features that may separately have no strong correlation with MEDV, but combined with other features have a much stronger correlation. Working with these compound features proved to be something that either didn't work here or an area I would need to learn

more about before diving deeper than the experiment I did attempt.

Finally, other models within sklearn may have been useful, but ultimately I did not have the time to try these after changing my research topic. SGDRegressor is able to use gradient decent, which finds the lowest "path" to find the best fit line. This may have given better results, but I did not have the chance to get too deep into it.

Project Description

To begin this project, the first step is to take a closer look at the data provided. Fitting every scatter plot between every pair of features would end up being too small to be useful here, so in Figure 1, I have included a visual of some selected features plotted against MEDV. This data is shown after removing some data, which is elaborated on in the next paragraphs.

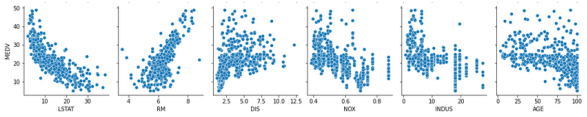


Figure 1 - Scatterplots for LSTAT, RM, DIS, NOX, INDUS, and AGE against MEDV from Data2

Looking at this, LSTAT and RM seem to have a pretty apparent linear correlation, though LSTAT does seem to have a slight curve, it is not significant for these purposes. Using .corr() on the data to get the correlation data, this is shown with these two features having -0.74 and 0.69 correlation repectively – some good candidates for features to hone in on. This full dataset with all rows will be our Data1 dataset.

As highlighted earlier, there are a few additional aspects to consider, namely the hard limit on values for MEDV and AGE. We will make two additional datasets: Data2 which removes any rows where MEDV is equal to 50 and Data3, which removes any rows where AGE is equal to 100 from Data2.

Finally, a final dataset, Data4, will be generated based on Data3 but removing a few notable outliers from some of the plots. Figure 2 shows a before and after of LSTAT/AGE when removing the two outlier rows from the dataset.

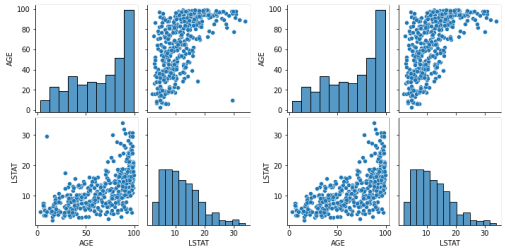


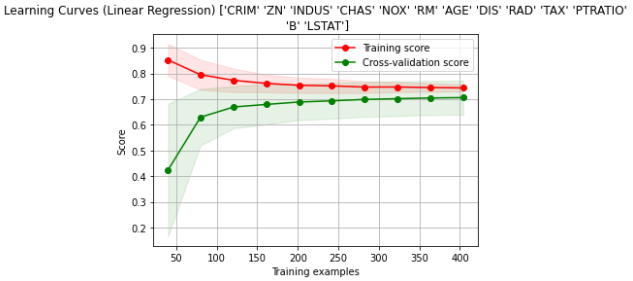
Figure 2 – Some outliers removed from Data3 to create Data4

Similar outlier removals were performed on LSTAT/RM and MEDV/RM removing a total of 6 rows from Data3, and about 60 rows total from Data1.

Each of these datasets will be test 8 times each, using features in each iteration that pass a threshold for the correlation data from 0.0 to 0.7 in increments of 0.1. For example, the test run at the .3 threshold will only include features with a correlation that is greater than .3 and won't consider the other features. The goal of these tests is to see which features help and hinder the predictions as well as see any changes that may arise from removing the rows from the previous section.

Experiments

With the four data sets and methods established in the previous section, I will now go over the results for each of these datasets and observations at each step.



	Threshold	Features	Score	Mean Abs Error	Mean Square Error
0	0	[CRIM, ZN, INDUS, CHAS, NOX, RM, AGE, DIS, RAD...	0.589222	3.84291	33.449
1	0.1	[CRIM, ZN, INDUS, CHAS, NOX, RM, AGE, DIS, RAD...	0.589222	3.84291	33.449
2	0.2	[CRIM, ZN, INDUS, NOX, RM, AGE, DIS, RAD, TAX,...	0.579849	3.83947	34.2123
3	0.3	[CRIM, ZN, INDUS, NOX, RM, AGE, RAD, TAX, PTRA...	0.528773	4.12438	38.3713
4	0.4	[INDUS, NOX, RM, TAX, PTRATIO, LSTAT]	0.476203	4.3945	42.6519
5	0.5	[RM, PTRATIO, LSTAT]	0.488164	4.40443	41.678
6	0.6	[RM, LSTAT]	0.540908	4.14244	37.3831
7	0.7	[LSTAT]	0.430957	4.86401	46.3363

Figure 3 - Results from Data1 and the learning curves from linear regression

For Data1, shown in Figure 3, the scores at each threshold revealed an interesting pattern. As expected, there is a rise in the score at threshold .6, which includes only the RM and LSTAT features that we determined were the most correlated to MEDV, however this is not the highest score – in fast, including all features resulted in the highest score, decreasing until the threshold of 0.4 and rising again to a peak at threshold of 0.6. This was a little surprising to me as I had expected the less correlated features to negatively affect the score, but it seems like this is not entirely the case as including some of the least correlated features improved predictions. I will address my thoughts as to why this may be in the Conclusion section of this paper, so for now I will continue discussing the results of the other data subsets.

	Threshold	Features	Score	Mean Abs Error	Mean Square Error
0	0	[CRIM, ZN, INDUS, CHAS, NOX, RM, AGE, DIS, RAD...	0.757694	3.13748	18.7357
1	0.1	[CRIM, ZN, INDUS, NOX, RM, AGE, DIS, RAD, TAX,...	0.759177	3.13297	18.6211
2	0.2	[CRIM, ZN, INDUS, NOX, RM, AGE, DIS, RAD, TAX,...	0.759177	3.13297	18.6211
3	0.3	[CRIM, ZN, INDUS, NOX, RM, AGE, DIS, RAD, TAX,...	0.759177	3.13297	18.6211
4	0.4	[CRIM, ZN, INDUS, NOX, RM, AGE, RAD, TAX, PTRA...	0.734549	3.30179	20.5254
5	0.5	[INDUS, NOX, RM, TAX, PTRATIO, LSTAT]	0.717841	3.41983	21.8173
6	0.6	[INDUS, RM, LSTAT]	0.646711	3.98964	27.3172
7	0.7	[LSTAT]	0.526573	4.3507	36.6066

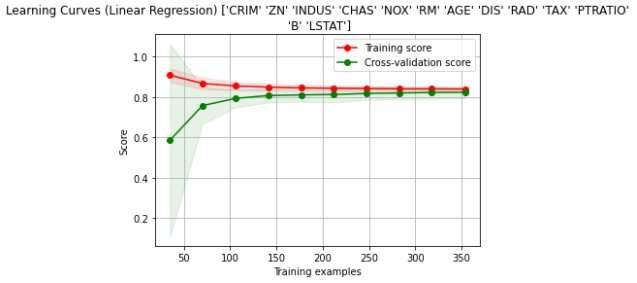
Figure 4 – Results from Data2, no rows where MEDV == 50

Figure 4 details the results from Data2, which excluded rows with an MEDV value of 50. A few interesting observations here: all of the scores at each threshold has increased by roughly 10~20%, but not linearly. Some of the correlation values also shifted, as evidenced by INDUS surpassing the .6 threshold. However, this did not improve predictions at this threshold, being the worst score after LSTAT alone in features used to make predictions. The winner this time was all features except for CHAS, resulting in a score of 0.759, a huge improvement over the highest score of .589 from Data1.

	Threshold	Features	Score	Mean Abs Error	Mean Square Error
0	0	[CRIM, ZN, INDUS, CHAS, NOX, RM, AGE, DIS, RAD...	0.832845	2.31691	10.6461
1	0.1	[CRIM, ZN, INDUS, NOX, RM, AGE, DIS, RAD, TAX,...	0.832941	2.31448	10.6399
2	0.2	[CRIM, ZN, INDUS, NOX, RM, AGE, DIS, RAD, TAX,...	0.832941	2.31448	10.6399
3	0.3	[CRIM, ZN, INDUS, NOX, RM, AGE, DIS, RAD, TAX,...	0.832941	2.31448	10.6399
4	0.4	[CRIM, INDUS, NOX, RM, AGE, RAD, TAX, PTRATIO,...	0.769656	2.92413	14.6706
5	0.5	[INDUS, NOX, RM, TAX, PTRATIO, LSTAT]	0.769028	2.93824	14.7105
6	0.6	[RM, LSTAT]	0.703336	3.31214	18.8945
7	0.7	[RM, LSTAT]	0.703336	3.31214	18.8945

Figure 5 – Data3, removed AGE == 100 from Data2

Moving on to Data3 shown in Figure 5, after removing AGE=1000 from Data2, we get our best scores yet. This time, the score seemed best using all features and dropped as we increased our correlation threshold, with the best score being .83 and the lowest at .7, which is still not a bad score.



	Threshold	Features	Score	Mean Abs Error	Mean Square Error
0	0	[CRIM, ZN, INDUS, CHAS, NOX, RM, AGE, DIS, RAD...	0.853518	2.10158	8.81732
1	0.1	[CRIM, ZN, INDUS, NOX, RM, AGE, DIS, RAD, TAX,...	0.85256	2.1351	8.87499
2	0.2	[CRIM, ZN, INDUS, NOX, RM, AGE, DIS, RAD, TAX,...	0.85256	2.1351	8.87499
3	0.3	[CRIM, ZN, INDUS, NOX, RM, AGE, DIS, RAD, TAX,...	0.85256	2.1351	8.87499
4	0.4	[CRIM, INDUS, NOX, RM, AGE, RAD, TAX, PTRATIO,...	0.795056	2.5773	12.3363
5	0.5	[INDUS, NOX, RM, TAX, PTRATIO, LSTAT]	0.798138	2.57015	12.1509
6	0.6	[RM, LSTAT]	0.719689	3.00704	16.873
7	0.7	[RM, LSTAT]	0.719689	3.00704	16.873

Figure 6 – Data4, removed outliers from Data3 along with the learning curves from linear regression

Finally, in Figure 6 shows the results of Data4 after removing some of the outlier rows from our data. This improved on the results from Data3 with a new high score of .85. Looking at the learning curves, compared to Figure 1, they begin converging much sooner now.

There was one more additional test that didn't end up improving the results. I had tried combining TAX+RAD and DIS+NOX in Data2, as these features seemed to be highly correlated with each other. However, these results didn't improve and was all around very slightly worse than without, with a high score of .751.

	Threshold	Features	Score	Mean Abs Error	Mean Square Error
0	0	[CRIM, ZN, INDUS, CHAS, NOX, RM, AGE, DIS, RAD...	0.64012	4.08121	55.4914
1	0.1	[CRIM, ZN, INDUS, NOX, RM, AGE, DIS, RAD, TAX,...	0.63418	4.13118	56.4074
2	0.2	[CRIM, ZN, INDUS, NOX, RM, AGE, DIS, RAD, TAX,...	0.63418	4.13118	56.4074
3	0.3	[CRIM, ZN, INDUS, NOX, RM, AGE, DIS, RAD, TAX,...	0.63418	4.13118	56.4074
4	0.4	[CRIM, INDUS, NOX, RM, AGE, RAD, TAX, PTRATIO,...	0.59601	4.58568	62.2929
5	0.5	[INDUS, NOX, RM, TAX, PTRATIO, LSTAT]	0.595528	4.58293	62.3673
6	0.6	[RM, LSTAT]	0.614515	4.93823	59.4396
7	0.7	[RM, LSTAT]	0.614515	4.93823	59.4396

Figure 7 – Train on Data4, test on Data1 minus the outliers

With Data4 giving the best result, there was one final experiment I wanted to perform – train on this dataset, but use the rows where MEDV is 50 in the test set. To make the predictions more consistent, an additional change was needed where I needed to truncate the predictions to 50 for any prediction that exceeded this value. The results are in Figure 7. These results were not as good as just predicting on Data4, but predicting power was enhanced over just using the original data to train. Including the outliers as well reduced these improvements, but outliers are outliers for a reason and won't improve predictions necessarily either way. One thing to consider is that some of the 50+ values may have been outliers on their own or there may be a curve in the correlation that is hidden by the truncated data, but we can't know as we don't have accurate records of these.

Conclusion

Based on the results of my experiments, it appears that the conclusion would be that having all the data helps make better predictions – however, I am not ready to fully state this and further experiments would be needed to try to find out exactly why using the most highly correlated features alone often ended up with worse results than including the highly uncorrelated features included. One hypothesis I have is based on a little bit of my audio background. In audio, moving from a high bit depth (the number of bits used to describe the amplitude of the audio at a given point) to a lower bit depth introduces some quantization noise, where bits that aren't present in the lower bit audio is are rounded up or down to the nearest available bit. This can result in some

sound degradation, and in more extreme cases, the audio is rounded down to silence at a certain point. Adding random noise 1 bit in amplitude before converting, however, makes a slightly noisier result with the benefit that the randomness tends to center around the true amplitude, allowing these sounds that would be otherwise silenced still be audible at the expense of adding some noise. My guess is that something similar is happening here where the noise introduced by the non correlated data is actually helping to make a more accurate result.

My other hypothesis is that there are internal correlations between features not directly correlated to the MEDV value, however combined result in a new compound feature that has a stronger correlation. This is what I attempted to look at when combining a few highly correlated features, however this was not as thorough, and I lack the knowledge set at the moment in tackling this type of analysis and did not have the time to explore this much further after needing to change my topic very last minute.

If I were to continue further with these experiments, I may also try to look for more recent data and data that has more values, as I did have concerns that removing too many data points could result in a model biased towards my end goals or over fit data. I would also like to try different algorithms, as detailed in the Related Works section, and compare the results against the linear regression results.

References

Russell, S., and Norvig, P. *Artificial Intelligence: A Modern Approach*. Upper Saddle River: Prentice hall.

Cantaro, M. (2020, January 4). *What You Didn't Know About the Boston Housing Dataset*. https://towardsdatascience.com/things-you-didnt-know-about-the-boston-housing-dataset-2e87a6f960e8

[1] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html (used [1] because there is no author or anything else)