Comparing accuracy of Machine Learning algorithms for the prediction of Housing prices in Boston Massachusetts

Abigail Tata
American University
At0427a@american.edu

Abstract As the housing market fluctuates up and down throughout the years, there are some factors that stick with the houses and are a big determiner of what the house might be worth. For example, we can assume that the more rooms there are in a house, the more expensive that house would be. There has been extensive research throughout the years in hopes of learning the factors of what determines a houses price and how heavily that factor affects the price of the house. As we learn about these factors, we want to learn the correlation of the housing price and if there is a good method that helps us have a good accuracy of the prediction that we are trying to see. Methods that can help us in learning the accuracy of the predictions of housing prices would be Machine learning Algorithms. Machine Learning algorithms are a good and effective way to be able to see the accuracy of the prediction we are trying to make for determining a housing price.

Keywords: Decision Tree, Multiple Linear Regression, Random Forest

1. Introduction

The housing market has been going up and down for decades. There are multiple reasons as to why the housing prices have been inconsistent, Inflation and the economy can have big roles towards determining the houses worth. but there are also multiple other factors that can contribute to why a house is sold at a certain price. In this paper we will be discussing the housing market and if there is an accurate way of predicting what the cost of a house would be. We will be looking at machine learning algorithms and focus on how they work with the data and if they create an accurate prediction. We will then look at the predictions of each

algorithm and see which one had the best accuracy percentage.

2. The data set, modifications, and variables

A good data set that looks at the housing prices and other factors would be the Boston Housing data set. This is a well-known data set in the data science community which looks at the US census data from the 1970s and shows information about the Boston suburbs. The data set goes into great depth in explaining the circumstances of the houses, inside the home and the around the of Boston that occupies the area of the home. Some of these variables include the crime rate, rooms in the house, ages of the occupants, access to highways, and the most important of what we are going to be discussing, the median value of the occupied homes. The data set is relatively small including 506 rows and 14 columns.

Modifications

Within this project we will be using Google Collaboratory which is practically the program Python. In Google Collaboratory there are some built in datasets that everyone can use. One of these datasets is the Boston housing data set. In order for us to start looking at this data set we need to do so much needed modifications for it to meet our standards.

The first fix that we are going to have to do to the data set is include the MEDV data variable. The MEDV data column is our most important column because it stands for the median home value, which is how we are going to be able to make predictions. When you look at the built in Boston housing data set in python (Google Collaboratory) you will notice that the data set has 13 columns instead of 14 and does not include the MEDV data set. If you go to other programs, for example, R, you will notice that the built in Boston housing data set contains MEDV, unlike python.

In order to fix this, we have to grab the original data set and set it to a data frame X. Then we will have to target the Boston data set would this specific column “MEDV” and set that to a data frame Y. After that, we will concatenate the two as x and y and this will combine the two into a new data frame with both the original and the MEDV column together. This makes the data set go from the original data set of 13 variables to 14.

The next step we will be taking is eliminating a variable that is considered to be problematic and unethical. The variable within the data set that describes the racial makeup of a population where the column "B" is the proportion of African Americans per town. The data showed that if there was a neighborhood with a high volume of black Americans, the price of the houses went down. "In spite of the lack of any systematic evidence supporting the self-segregation hypothesis, it is difficult to dismiss. The problem lies in the fact that it is virtually impossible to determine conclusively the role of self-segregation as long as traces of white community antagonism toward black efforts to leave the ghetto remain" (Kain and Quigley). It is important to know that the demographics in the 1970s were quite different than they are today. the black population was around 22%, which is significantly higher compared to other minority groups (Latinx 2.8%)1. Today the minority groups in Boston are a quarter African American, half white, and the rest being a mix of other races2 This is an outdated data set which can have racist interpretations, especially when historical context is not considered. Looking at all of this information, we then decide to take out the ethical problem and code to drop the column.
We take out the specific column "B" by using the code df.drop. In brackets you list the column "B" and then once you display the data frame it shows the new data set that does not contain the problematic variable and displays 13 columns instead of the previous 14.

The Variables

Now that we have the data set up and ready to work with, we can now look at the variables that we will be using for the rest of the project. These include:

CRIM - per capita crime rate by town
ZN - proportion of residential land zoned for lots over 25,000 sq.ft.

---

[1] Crangle, Colleen. "Inadvertent Racism in Popular Data Science Data Set." *LinkedIn*, LinkedIn, 21 Sept. 2018, https://www.linkedin.com/pulse/its-time-retire-boston-housing-dataset-colleen-e-crangle/. ttps://www.linkedin.com/pulse/its-time-retire-boston-housing-dataset-colleen-e-crangle/

[2]https://www.google.com/search?client=safari&rls=en&q=boston+demographics&ie=UTF-8&oe=UTF-8

INDUS - proportion of non-retail business acres per town.
CHAS - Charles River dummy variable (1 if tract bounds river; 0 otherwise)
NOX - nitric oxides concentration (parts per 10 million)
RM - average number of rooms per dwelling
AGE - proportion of owner-occupied units built prior to 1940
DIS - weighted distances to five Boston employment centers
RAD - index of accessibility to radial highways
TAX - full-value property-tax rate per $10,000
PTRATIO - pupil-teacher ratio by town
LSTAT - % lower status of the population
MEDV - Median value of owner-occupied homes in $1000's

3. Research Goal

What we are trying to do in this paper is look at a few different machine learning algorithms and have them predict the housing prices through looking at the characteristics that the given house provides. Once they create a prediction, they will compare the number to the actual price of the house. They then provide an accuracy of how well that given algorithm did at predicting the prices of the houses in the data set. The goal of this paper is to compare a Multiple Linear Regression, Random Forest, and a Decision Tree model to see which is better at predicting the housing prices in the Boston area. We want to know which model has the best accuracy of predicting the housing prices and what those numbers are

4. Models and Hypothesis

In order for us to be able to work on this goal, it is important for us to understand what these machine learning algorithms are and what they can do to help us in predicting the accuracy score of the housing prices.

> Decision Trees are models that predict through learning decision rules[3]. The decision tree asks a question and based on the answers it splits into branches[4] it eventually finds an answer. They are easy to interpret[5]

---

[3] 1.10. Decision Trees." *Scikit*, https://scikit-learn.org/stable/modules/tree.html.
Ampadu25.00, HAHyacinth, et al. "Decision Trees." *AI Pool*, https://ai-pool.com/a/s/decision-trees.

[4] "Decision Trees: An Overview

[5] "1.10. Decision Trees." *Scikit*, https://scikit-learn.org/stable/modules/tree.html.

Random Forests are constructed from decision trees. The forest is trained through bagging or bootstrap aggregation and produces a result from the predictions of the many decision trees. Increasing the number of trees increases the accuracy of the outcome. Good for large data sets. Harder to interpret[6]
Multiple Linear Regression is used in statistics to predict an outcome based on multiple variables. Constructed from an assumption of a relationship of linearity between both independent and dependent variables[7]

5. Related concepts

We decided that we wanted to look at relevant work so that we could get tips on how to write the code. We also wanted to get insight of the end results so that we could make grounds for creating a hypothesis. We found related code on Kaggle by an individual whose username is "SUBHRADEEP" that talked about predicting housing prices in an area called King County. This page brought up the prediction of housing prices through Random Forests and Decision Trees and Multiple Linear Regressions.

In this code, the individual provided an in-depth explanation into many of the processes we were looking for in order to continue our project. They explained the process of retrieving the data set, setting it up into a data frame for it to be easily interpreted, the correlations of the housing price and the other given characteristics, and code the algorithms with the data. Lastly, and most importantly, they explained how we could retrieve the algorithms information and creating it into a clean and neat data table for us to be able to interpret for ourselves. Eventually, it provided results that stated that the Random Forest model was the most successful and had the highest accuracy of the three.[8]

---

[6] "Introduction to Random Forest in Machine Learning." *Section*, https://www.section.io/engineering-education/introduction-to-random-forest-in-machine-learning/.

[7] GoCardless. "Multiple Linear Regression (MLR) Definition." *United Kingdom*, https://gocardless.com/en-us/guides/posts/multiple-linear-regression-mlr-definition/.

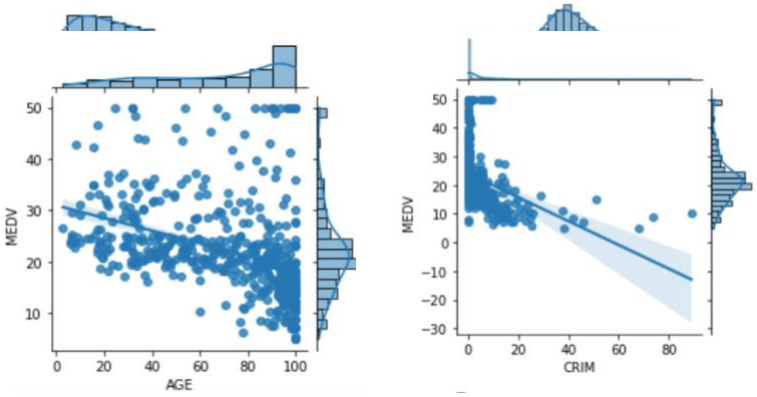
[8] subhradeep88, SUBHRADEEP. "House Price Predict: Decision Tree,Random Forest." *Kaggle*, Kaggle, 25 Aug. 2020, https://www.kaggle.com/subhradeep88/house-price-predict-decision-tree-random-forest/notebook.

6. Hypothesis

We have now analyzed our data, learned what each algorithm is and what they're used for, and we have looked at relevant work. We can now come up with a hypothesis of which model might have the highest accuracy at predicting housing prices. Out of the three methods, I think that Random Forests will perform the best. Random forests are used for big data sets, just like the King County data set which contains around 21 columns and around 21,000 rows. One decision tree won't be able to produce good results due to the simplicity of the model. I feel that the Boston data set is big enough and complex enough for it to be the most successful with a Random Forest model.

7. Correlation

Although not a necessity towards our project, we can consider it to be helpful for us to look at scatterplots and see the correlation that variables have with one another. We will be focusing on the variable MEDV which is the housing prices and the other variables or characteristics and see how the two interact with one another. We will be looking at if the scatter plot is widely spread out or go along the line and if the line goes up or down (positive correlation and negative correlation).



These are four examples from our code, the rest are provided within our code on Google Collaboratory. As we look at the scatterplots of all variables, we can see a trend that a large majority of the scatterplots seem to have a negative correlation, or moving downwards. Some of the scatter plots have the negative correlation but instead of the dots being closely together in a straight line (high correlation) they are still negative, but spread out, meaning that theres a low correlation. Only a select few happen to have a positive

correlation and those for example are self explanatory relations like the more rooms a house has, the more expensive the house happens to be.

8. Experiment

The experiment will consist of the set up of training the data and then coding the three algorithms.

Training the data

We start by looking at the data and splitting it into a 80/20 ratio where 80 percent of the data is trained and the remaining 20 percent of the data will be tested. We will train the models using the training set and then apply the model to the testing set to receive the final results. To code this, we put X(independent variables) and Y(target variables) variables and set them both to a training and testing variable. We then put the testing size to .2 (20% of the data that we are going to train) and then set the testing set to a random state. A random state makes sure that no matter how many times we test the data, the results will be the same. If we did not contain random state code, every time we run the code it will produce a different number.

The Algorithms codes

Multiple Linear regression:

```
[12] mlr = LinearRegression()
     mlr.fit(X_train,y_train)
     mlr_score = mlr.score(X_test,y_test)
     pred_mlr = mlr.predict(X_test)
```

In this code, we set the linear regression code to MLR for Multiple Linear Regression. We then fit the training data which is presented on line two, then test the data which shows the score on line three. The fourth line is an optional line that states that the prediction score is the X_Test. On line three we have the prediction score which is X-Test and the target score which is Y_Test. the Multiple Linear Regression score on line three.

Decision Tree:

```
[13] tr_regressor = DecisionTreeRegressor(random_state=0)
     tr_regressor.fit(X_train,y_train)
     tr_regressor.score(X_test,y_test)
     pred_tr = tr_regressor.predict(X_test)
     decision_score=tr_regressor.score(X_test,y_test)
```

In this code we set the decision tree regressor to tr_Regressor and put a random state of 0 (a seed that I chose at random), so we receive the same results every time. We then fit the training data which is seen on line two and then test the data which shows a score which is coded on line three. Line three codes from the prediction score (X_Test)and then the target score (Y_Test_) to create the Decision Tree score. Line five is the same as line three but it is labeled.

Random Forest:

```
rf_regressor = RandomForestRegressor(n_estimators=10,random_state=0)
rf_regressor.fit(X_train,y_train)
rf_regressor.score(X_test,y_test)
rf_pred =rf_regressor.predict(X_test)
rf_score=rf_regressor.score(X_test,y_test)
```

Lastly, we are working with the Random Forest code where we set the Random Forest regressor to rf_regressor. We then fit the training set on line two and then on line three we score the testing data. As stated in the Decision Tree and Multiple Linear Regression code, we score by coding from the prediction score (X_Test)and then the target score (Y_Test_) to create the Random Forest Code. Line four is simply the prediction score and Line five as stated in Decision Tree is the same as line three but it's labeled.

9. Results

In order to retrieve the results we must print a code for us to see:

```
print("Multiple Linear Regression Model Score is ",round(mlr.score(X_test,y_test)*100))
print("Decision tree  Regression Model Score is ",round(tr_regressor.score(X_test,y_test)*100))
print("Random Forest Regression Model Score is ",round(rf_regressor.score(X_test,y_test)*100))

 models_score =pd.DataFrame({'Model':['Multiple Linear Regression','Decision Tree','Random forest Regression'],
                             'Score':[mlr_score,decision_score,rf_score]})
 models_score.sort_values(by='Score',ascending=False)
```

For each score we show the stated formula (all of them are on line three for each model). Once we code that we then multiply each one by 100 so we can receive a percentage instead of a decimal. We then put these models into a data frame for us to be able to easily view it and sort it by score so we can see from highest values to lowest values.

```
Multiple Linear Regression Model Score is  41
Decision tree  Regression Model Score is  27
Random Forest Regression Model Score is  47
```

	Model	Score
2	Random forest Regression	0.474226
0	Multiple Linear Regression	0.412986
1	Decision Tree	0.271576

The results show that Random Forest regression got a score or accuracy of 47%, Multiple Linear Regression got a score of 41% and Decision tree receiving a score of only 27%.

10 Conclusion

Our hypothesis was proven to be correct. Random Forest would produce the best scores (47%) which we believe is because of how it works with larger data sets. Multiple Linear Regression was not too far behind (41%). Although the first two didn't do well, not reaching an accuracy over 50%, The accuracy of the Decision Tree was practically half of the other two reaching only 27%. We can conclude that All of these models aren't good at producing accurate predictions about the housing prices, meaning that we would have to do more research in the future or create our own model that can create predict better and create a higher accuracy.

Work Cited

- Crangle, Colleen. "Inadvertent Racism in Popular Data Science Data Set." LinkedIn, LinkedIn, 21 Sept. 2018, https://www.linkedin.com/pulse/its-time-retire-boston-housing-dataset-colleen-e-crangle/.
- Lorenz, Jan, et al. "How Social Influence Can Undermine the Wisdom of Crowd Effect." PNAS, National Academy of Sciences, 31 May 2011, https://www.pnas.org/content/108/22/9020.
- "Decision Trees: An Overview." Aunalytics, 2 Nov. 2021, https://www.aunalytics.com/decision-trees-an-overview/.
- "Introduction to Random Forest in Machine Learning." Section, https://www.section.io/engineering-education/introduction-to-random-forest-in-machine-learning/.
- "1.10. Decision Trees." Scikit, https://scikit-learn.org/stable/modules/tree.html.

- Ampadu25.00, HAHyacinth, et al. “Decision Trees.” AI Pool, https://ai-pool.com/a/s/decision-trees.
- “Decision Tree vs. Random Forest - Which Algorithm Should You Use?” Analytics Vidhya, 12 May 2020, https://www.analyticsvidhya.com/blog/2020/05/decision-tree-vs-random-forest-algorithm/.
- GoCardless. “Multiple Linear Regression (MLR) Definition.” United Kingdom, https://gocardless.com/en-us/guides/posts/multiple-linear-regression-mlr-definition/.
- subhradeep88, SUBHRADEEP. “House Price Predict: Decision Tree,Random Forest.” *Kaggle*, Kaggle, 25 Aug. 2020, https://www.kaggle.com/subhradeep88/house-price-predict-decision-tree-random-forest/notebook.