Predicting Boston Housing Prices

1) Statistical Analysis and Data Exploration

Number of Houses	Number of Features	Minimum Housing Price	Maxiumum Price
506	13	$5000	$50000

Mean Price	Median Price	Standard Deviation
$22,5382	$21,200	$9,188

2) Evaluating Model Performance

For measuring the error of the model an Mean Squared Error (MSE) measurement was used. MSE gives us the average sum of the squared differences between the predicted result and actual result. [1] Squaring the differences gives us two things; first it doesn't matter whether the differences are negative or positive as squaring makes them all positive, which makes it a better option than just looking at the average error. Second squaring punishes differences that are larger more, this can be both good and bad as outliers can have a very large effect on the MSE. [2] In our case since there don't seem to be any extreme outliers (all data points are within 3 standard deviations) so squaring the error is appropriate and a better score than mean absolute error.

We have split the data into test and training data so that we are able to evaluate our model, specifically helping make sure we avoid overfitting. We use the training data to help our model 'learn' what its parameters should be. We will then test this model on the test data to see how well it performs. This helps us find out how well what our model learned on the training set generalizes to new data it is asked to predict on. Without this split we could end up overfitting our model to the training data and not knowing if our model is able to generalize for future predictions. [2]

Grid Search is used to evaluate a list of potential parameters and parameter combinations to see which one gives the best performance. Unlike Randomized

Parameter Optimization, where different parameter combinations are randomly tried over a distribution of values, Grid Search uses a specified set of parameter combinations and values to explore through. [2]

Cross Validation breaks the dataset up into multiple pieces to help maximize the data that is being used for both testing and training. In the case of K-Fold Cross Validation the dataset is broken up into K different sets and you run K different experiments. In each of these experiments one of the sets is held back as testing data and then the rest of the data is used as training data. After running all of these you average the results from running each of the K experiments. In our case we use the scikit-learn default of 3 folds. [3]

We combine Cross Validation and Grid Search to help us experiment with a large number of parameter options across a the whole dataset. This helps us to come up with a well tuned model, while still having it split up with a training and test set so that we can prevent overfitting.

3) Analyzing Model Performance

At approximately a training size of 50 the test error levels off somewhat, this is fairly consistent for all decision tree depths. You can see this being the same in figures 1 through 10 as in all of them the error on the test dataset levels off after the training size hits 50. It still continues to improve with more training data, however at a much slower and diminishing rate. Training error increases with the size of the training dataset, and this affect decreases as the model complexity grows. This makes sense as a less complex model will have a harder time fitting to a larger, more complex, dataset.

The higher the tree depth (increase in complexity) the more the error in the training set decreases. You can see looking at figures 1 through 10 in sequence and seeing the training error continue to improve with each. However, as mentioned earlier, with some variance the test error levels off sharply around a training set of 50 with much smaller improvement as the training dataset grows. This is consistent through all model complexities.

There doesn't seem to be overfitting in the models with regards to the tree size comparing 1 and 10. The training error goes down with more depth, however the test error goes down as well - indicating that the more complex model still

predicts the test data better and does not seem to suffer from overfitting to the training data (see figures 1 through 10).

Looking at the model complexity output we see that the around a depth of 5 the test error levels off where, with some variance, the error stays the same (see figure 11). This is backed up by looking at the graphs of the learning curves for models 1 - 10 (figures 1 through 10). From depths 1 to 4 (figures 1 through 4) the test error improves with each increase in depth indicating that we are under-fit and our bias is too high. Each increase in complexity show an improvement in prediction accuracy so our model is inadequate early on.

From depths 4-10 (figures 4 through 10) there is little to no improvement in test error, despite the improvement in training error. There doesn't seem to be overfitting with depths 4 - 10 as the test error doesn't get worse, however the additional depths add complexity, and increases the amount of data we need to generalize, without bringing improvement in prediction. For our purposes, using a model with depth 4 is the best option as we have the lowest complexity while still keeping the model accuracy the same. Note, if more data was gathered this could change and we would need to re-evaluate.

4) Model Prediction

See Table 2 for input data.

With the given input data the predicted house price is $21,629.74 using a model with a depth of 4. This is close to both the mean ($22,530) and median ($21,200) house price and well within the standard deviation of the prices ($9,188) and well within the min ($5,000) and max ($50,000) of the dataset. Intuitively this gives us an idea that our predicted price is reasonable as it is similar to our existing data.

Additionally K Nearest Neighbours (KNN) was run which gives us the closest datapoints to our input point. [4] The average price of these KNN points was $21,520 which is very close to our predicted price. This gives us even more confidence in the prediction of our model.

Footnotes:

[1] http://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_squared_error.html#sklearn.metrics.mean_squared_error
[2] https://en.wikipedia.org/wiki/Mean_squared_error
[3] https://www.udacity.com/course/viewer#!/c-ud725-nd/l-5406799334/e-2948348839/m-2990338635
[4] http://scikit-learn.org/stable/modules/neighbors.html#finding-the-nearest-neighbors

Figures and Tables:

Table 1 - Input Parameters:

CRIM	Per capita crime rate by town
ZN	Proportion of residential land zoned for lots over 25,000 sq.ft.
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 centres
RAD	Index of accessibility to radial highways
TAX	Full-value property-tax rate per $10,000
PTRATIO	Pupil-teacher ratio by town
B	1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town
LSTAT	% lower status of the population

Table 2 - Data for Prediction:

CRIM	11.95
ZN	0.00
INDUS	18.100
CHAS	0
NOX	0.6590
RM	5.6090
AGE	90.00
DIS	1.385
RAD	24
TAX	680.0
PTRATIO	20.20
B	332.09
LSTAT	12.13

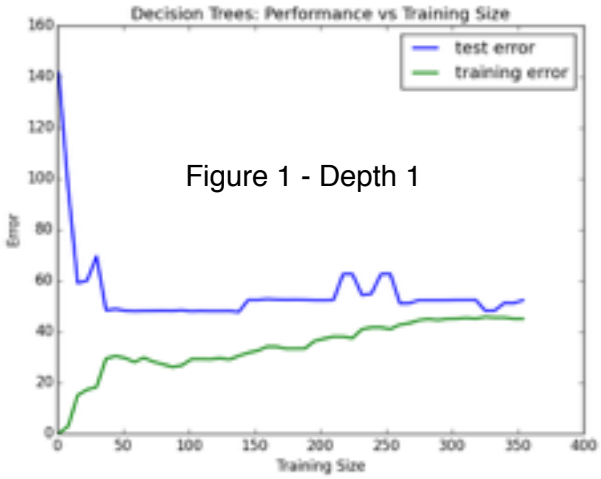

Figure 1 - Depth 1

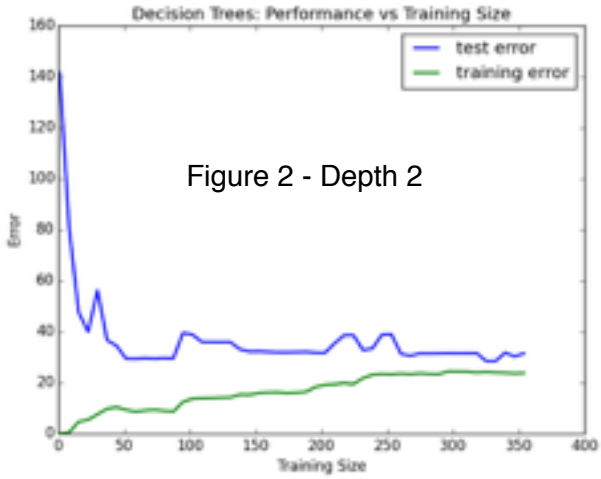

Figure 2 - Depth 2

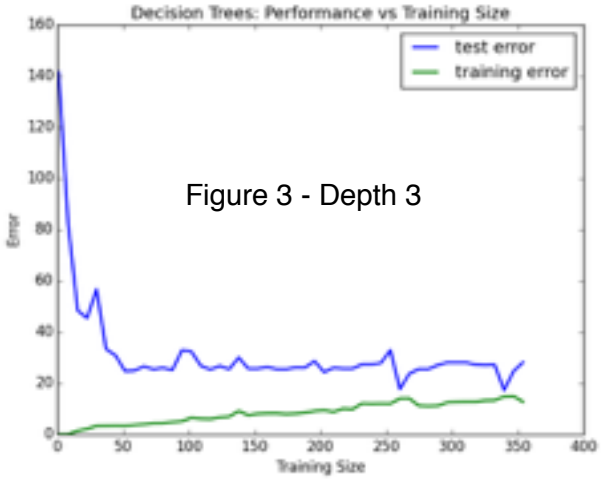

Figure 3 - Depth 3

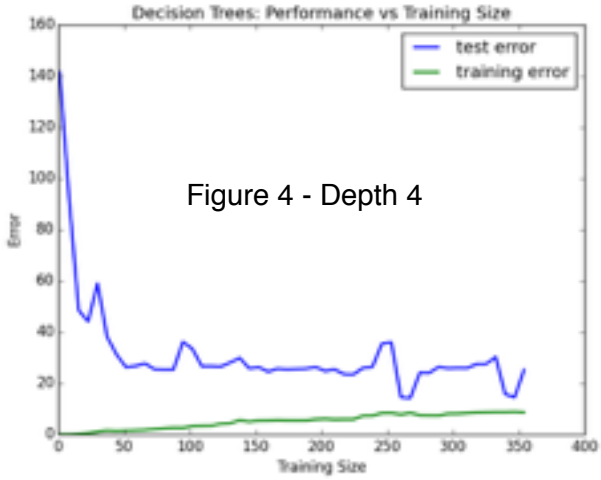

Figure 4 - Depth 4

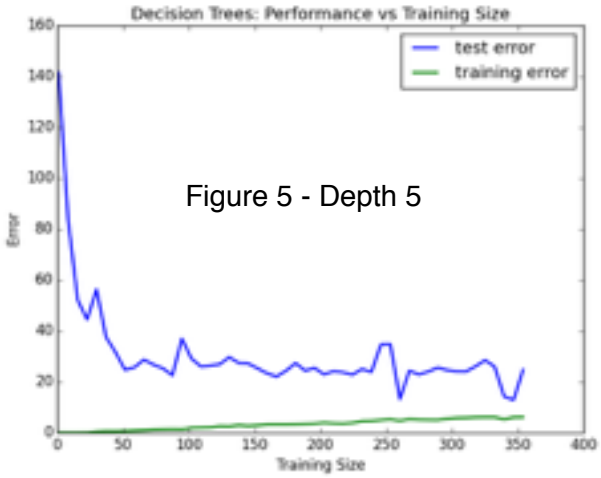

Figure 5 - Depth 5

Figure 6 - Depth 6

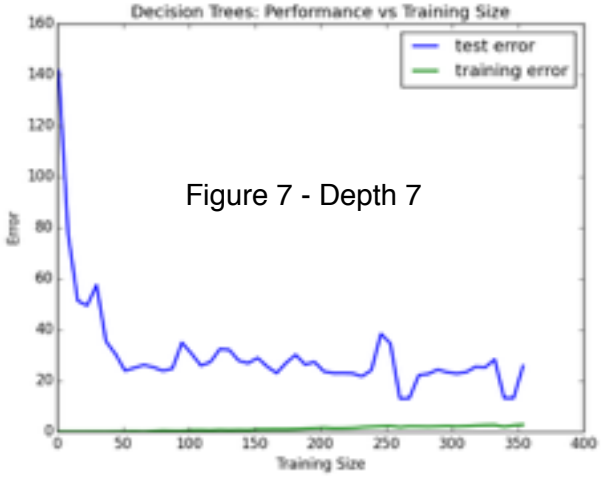

Figure 7 - Depth 7

Figure 8 - Depth 8

Figure 9 - Depth 9

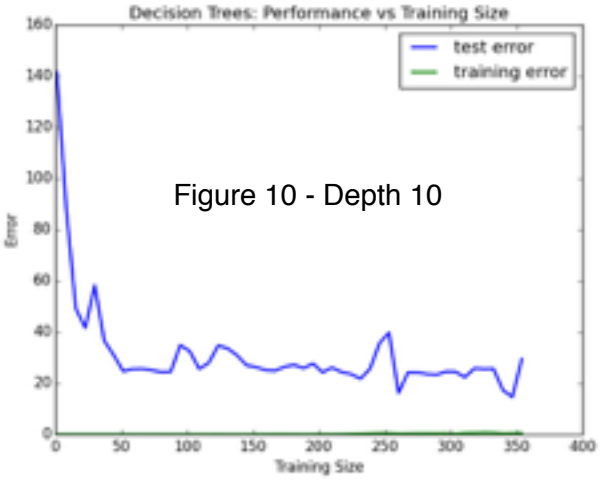

Figure 10 - Depth 10

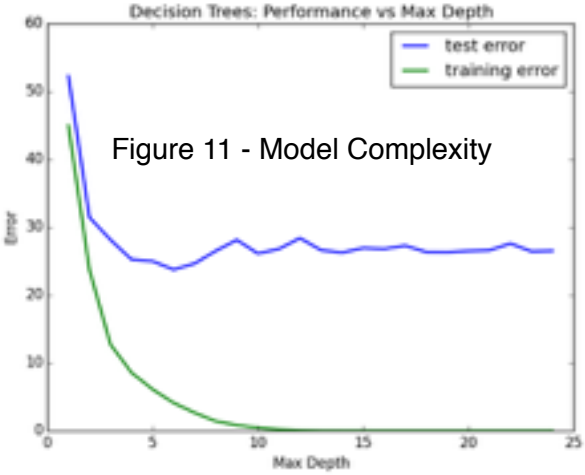

Figure 11 - Model Complexity