P1: Predicting Boston Housing Prices

1) Statistical Analysis and Data Exploration

- Number of data points (houses): 506
- Number of features (per house): 13
- Minimum house price: 5.0
- Maximum house price: 50.0
- Mean Boston housing price: 22.53
- Median Boston housing price: 21.2
- Standard deviation of Boston housing prices: 9.19

2) Evaluating Model Performance

- We are using regression to predict the price of a house, the reason we selected regression was because the data is continuous, while trying to predict the price of a given house it's not all that important to predict the price exactly, rather a close approximation is enough. By selecting a regression model we are limiting ourselves to metrics that measure regression model performance, for example Mean Absolute Error and Mean Squared Error metrics, further because we like to highlight larger errors vs. smaller errors as discussed above Mean Squared Error was selected for Boston Housing project.

- The Boston housing data was split into training and testing sub sets, we need independent set of data (testing data, in this case it’s set to 0.3 or 30%) to validate the results trained on training sub set, without testing data there is no way to know if the model will generalize for data that the model has not yet seen. For example without testing data we might overfit the model and have no way of knowing this.

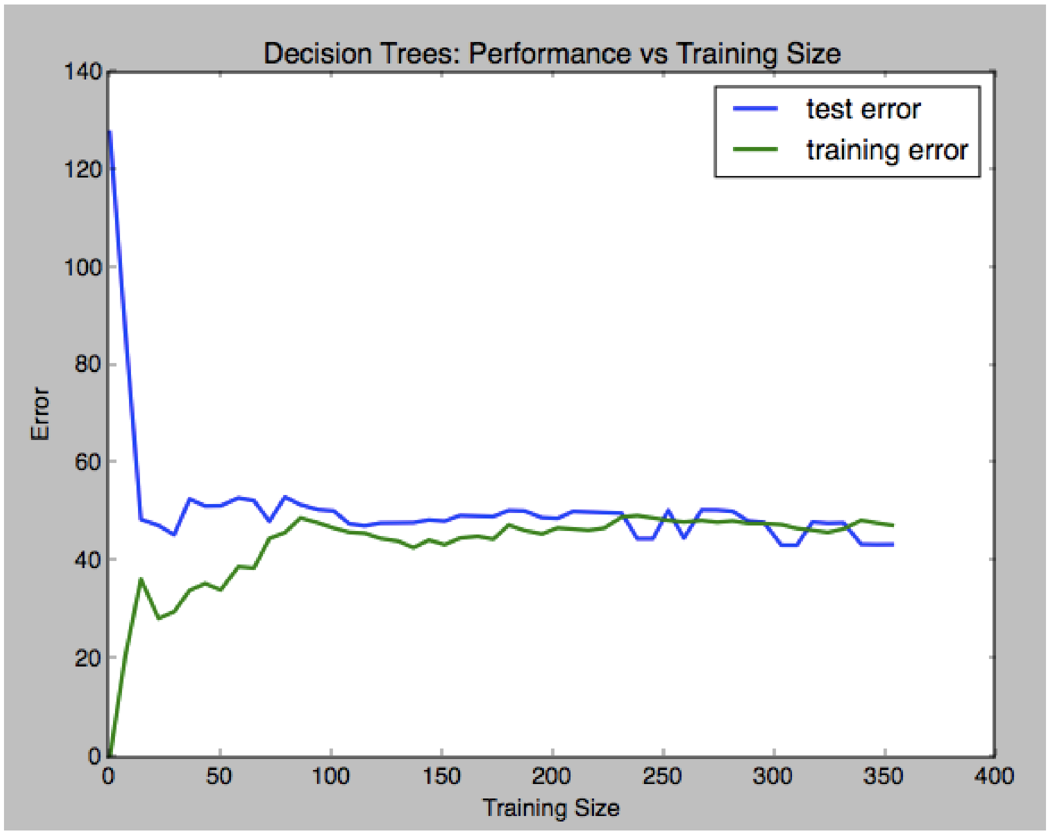
- Grid search systematically searches over all parameters given (for example max_depth) to find the best model (by using given Mean Squared Error metric in this case) to fit the data given, in this case grid search checks 10 regression prediction models from max_depth of 1 to 10 (also can be referred to as model complexity) to find the best model.

- Cross validation is a convenient way to splits the data into training and testing subsets for reasons discussed above. Additionally cross validation provides for selection of multiple validation (testing) and training sets, the data is randomly split into sections for each iteration, therefore we end up with multiple variations of testing and training sets, final accurity is determined by taking an average of each run. In this case train_test_split function is being used that splits the data into training and testing subsets dependent on parameters given including test_size, train_size and random_state for random sampling (random sampling is important to make sure the data is not ordered, for example we don't want to have one type of data for training and another type of data for testing, we want a mix of data for both testing and training). So given the discussion above we definitely want to use cross validation with grid search, GridSearchCV actually provides for this with parameter cv, an integer can be passed to cv to specify the number of folds to select, cv of None (defaults to 3-fold cross-validation) was selected for this project.

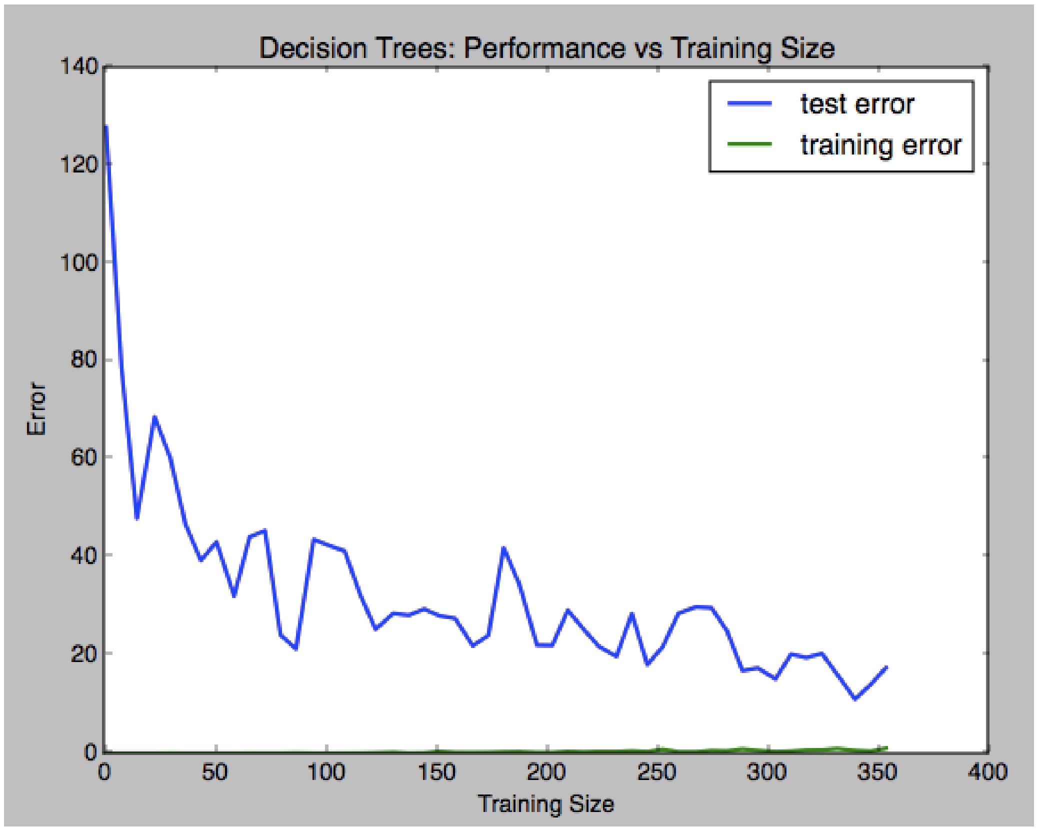
3) Analyzing Model Performance

- The general trend of training error as training size increases is up, as the training size increases the training error goes up. The general trend of testing error as training size increase is down, as the training size increases the testing error goes down.

- Looking at the learning curves of the decision tree regressor with max depth of 1 and 10. The learning curve of max depth of 1 (Graph 1) shows how the model is underfitting, the curve plateaus very quickly and does not improve with more training size, the model is just to simple to represent the data accurately (the model suffers from high bias). The learning curve of max depth of 10 (Graph 2) clearly shows that the model is overfitting (one can see how the training error is almost non-existent while the test error is significantly higher).

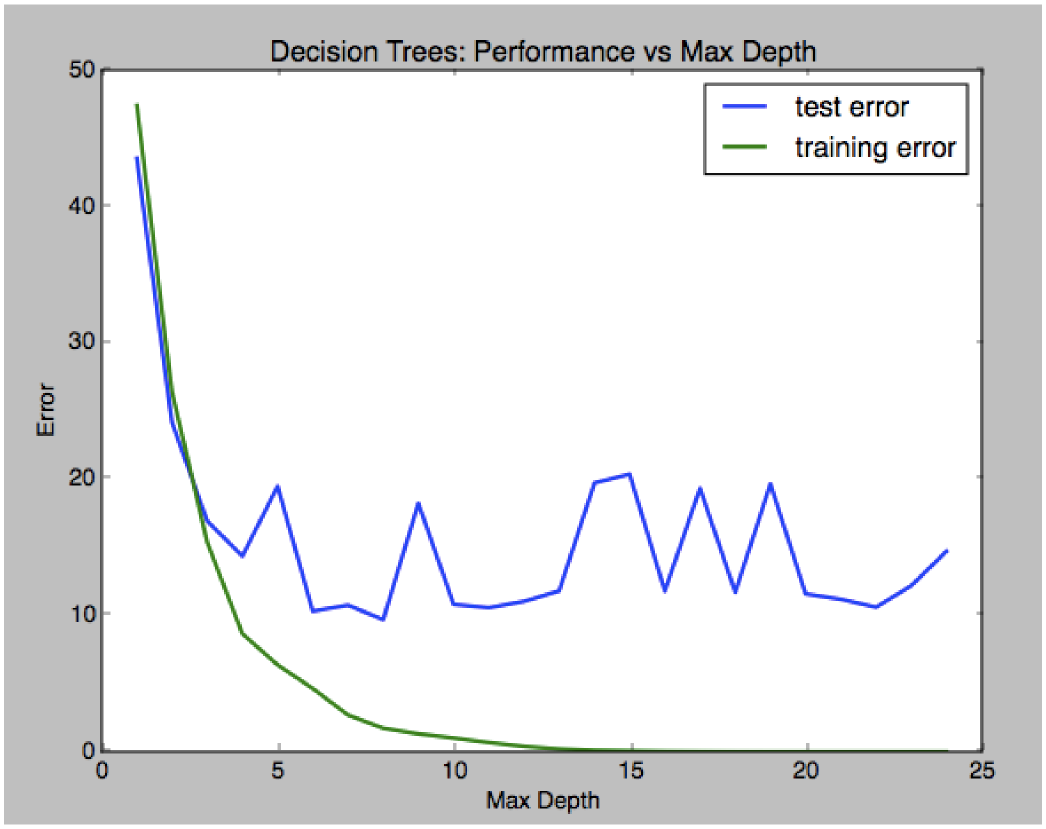


Graph 1



Graph 2

- Looking at the model complexity graph (Graph 3), training error decreases as the model increases in complexity (its max depth increase), while the test error also decreases as the max depth increases; however, the test error plateaus at about max depth of 4. In addition the training error and test error curves diverge quickly from each other after about max depth of 3 or 4, before max depth of 3 or 4 the two lines follow each other closely having about the same error rate. For this reason max depth of 4 model looks to generalize the data set the best. After max depth of 4 the models suffer from overfitting, and the test error no longer decreases.



Graph 3

4) Model Prediction

- The final model of max_depth = 4 predicts the price of the house to be 21.63.

- The max depth of 4 selected by grid search is close to the max depth of 4 that we selected looking at the complexity graph in the last section, therefore the max_depth of 4 selected by grid search seems reasonable. Looking at the learning curves of the decision tree regressor with max depth of 4, we can see that the error rate for testing dataset plateaus at around 15%. Additionally we can look at the mean (22.53) and the median (21.2) and the standard deviation (9.19) we calculated earlier for this dataset. The predicted price is well within one standard deviation from the mean, infact the predicted price is within 0.1 standard deviations from the mean.

  mean - predicted price = difference in price
  difference in price / standard deviation = x standard deviation
  22.53 - 21.63 = 0.9
  0.9 / 9.19 = 0.10 (rounded)

A 15% accuracy for the purposes of predicting Boston housing prices seems to be in the ballpark, precise enough for a quick estimate to identify a property of interest, but not precise enough to use as the actual price.