Report: Boston housing price prediction using decision tree regression

In this project, decision tree regression was used to predict Boston housing price. The project was completed based on the provided python code template, numpy and sklearn packages. The detailed explanation of the code is given below.

1) Statistical Analysis and Data Exploration

- Number of data points (houses)? 506
- Number of features? 13
- Minimum and maximum housing prices? 5.0, 50.0
- Mean and median Boston housing prices? 22.53, 21.2
- Standard deviation? 9.19

2) Evaluating Model Performance

- Which measure of model performance is best to use for predicting Boston housing data and analyzing the errors? Why do you think this measurement most appropriate? Why might the other measurements not be appropriate here?

  Mean squared error (MSE) is used as the performance metric to predict housing price.

  Cross-validation using MSE performance metric tends to choose the model that produces more small or media errors rather than the model produces few large errors. This is because a few squared large errors tend to be larger than even many squared small errors. On the contrary, mean absolute error (MAE) prefers model with few errors but tolerate some large error. For housing price prediction, as a realtor, I try to be as accurate as possible, and I can't tolerate large errors. Therefore, MSE is a better metric to use for this application.

- Why is it important to split the Boston housing data into training and testing data? What happens if you do not do this?

  The purpose of training/testing split is to keep the model trained on training data being independent of testing data. If split is not done, we are using training data to evaluate model performance. The testing error will be greatly underestimated.

- What does grid search do and why might you want to use it?

  Gridsearch is a nice method provided in sklearn to loop through all combination of hyper-parameters to select the one set of parameters that minimizes the validation error. The selected parameters are considered optimal and used to predict test data.

- Why is cross validation useful and why might we use it with grid search?

  In case of scarce data, splitting training/testing data results in insufficient data to train and evaluate performance. Cross validation is useful in the case by resampling data and training models many times to estimate prediction error.

  We use it with grid search since we want to do cross validation with different combination of hyper-parameters and determine the optimal set of hyper-parameters.

3) Analyzing Model Performance

- Look at all learning curve graphs provided. What is the general trend of training and testing error as training size increases?

  Training error increases and testing error decreases with increasing training size. This is due to the fact that more training data is harder to fit (increase training error) but yields more accurate model (decrease testing error).

- Look at the learning curves for the decision tree regressor with max depth 1 and 10 (first and last learning curve graphs). When the model is fully trained does it suffer from either high bias/underfitting or high variance/overfitting?

  With max depth 1, the training error and testing error are very close and high. It indicates that the model suffers from high bias.

  With max depth 10, the training error is almost zero, and testing error is high. It indicates that the model suffers from high variance.

- Look at the model complexity graph. How do the training and test error relate to increasing model complexity? Based on this relationship, which model (max depth) best generalizes the dataset and why?

  With increasing model complexity, the training error decreases and testing error first decreases and then stabilized. Max depth = 6 best generalizes the dataset because it yields the lowest testing error.

4) Model Prediction

- Model makes predicted housing price with detailed model parameters (max depth) reported using grid search. Note due to the small randomization of the code it is recommended to run the program several times to identify the most common/reasonable price/model complexity.

  The predicted housing price is 20.968 with max depth of 5. I ran the program several times and max depth of 5 is the most frequent.

- Compare prediction to earlier statistics and make a case if you think it is a valid model.

  Use the feature vector x to find k=10 nearest neighbors, the average housing price of the 10 nearest neighbors is quite close to the predicted price. This indicates that the model is valid.