Project 1 - Predicting Boston Housing Prices
Model Evaluation & Validation Project
2015/11/18

Project Description

You want to be the best real estate agent out there. In order to compete with other agents in your area, you decide to use machine learning. You are going to use various statistical analysis tools to build the best model to predict the value of a given house. Your task is to find the best price your client can sell their house at. The best guess from a model is one that best generalizes the data.

For this assignment your client has a house with the following feature set: [11.95, 0.00, 18.100, 0, 0.6590, 5.6090, 90.00, 1.385, 24, 680.0, 20.20, 332.09, 12.13]. To get started, use the example scikit implementation. You will have to modify the code slightly to get the file up and running.

When you are done implementing the code please answer the following questions in a report with the appropriate sections provided.

> __Language and libraries__
> For this project, you will need to have the following software installed:
> * Python 2.7
> * NumPy
> * scikit-learn
>
> __Template code__
> Download boston_housing.zip, unzip it and open the boston_housing.py >
> template file. Follow the instructions to complete each step, and answer the questions in your report.

> __Deliverables__
> Report in PDF format
> Fully implemented Boston Housing Python code as boston_housing.py
> You can package these two files as a single zip and submit it.

Questions and Report Structure

1. Statistical Analysis and Data Exploration

Statistic	Value
Number of data points	506
Number of features	13
Min housing price	5.0
Max housing price	50.0
Mean housing price	22.5
Median housing price	21.2
Std dev of housing prices	9.2

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?
> We want to determine the gap between the __predicted__ outcome and the __actual__ value and then go with the measure that __minimizes__ this __gap__ in a __consistent__ manner. __Median Absolute Error__ had the lowest error values and is reflected quite well (IMO) in the chart below.
> ![alt text](https://github.com/CharlesCrous/Udacity_Machine-Learning-Nanodegree\P1_Predicting_Boston_Housing/actual_verus_predicted.png)
> Other performance metrics I tried included:
> * Explained Variance Score
> * R2 Score
> * Mean Absolute Error
> * Mean Squared Error (as well as it's root)

Why is it important to split the Boston housing data into training and testing data? What happens if you do not do this?
> Breaking the dataset into two separate datasets means that we can avoid __overfitting__ our model on the initial data used to __train__ our understanding of patterns within the data of interest and then have the opportunity to __test__ how well this model __generalizes (predicts)__ to data that the model hasn't seen before.

What does grid search do and why might you want to use it?
> Enables you to test multiple sets of parameters using K-fold cross validation.

Why is cross validation useful and why might we use it with grid search?
>Cross validation reduces the __variance__ in outcomes that might occur by relying on a single run of train/test split, since it ensures that the data capture in both the test and the training set reflect the proportional composition of the full data set (stratification).

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?
>

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?
>

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?
>

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.

>

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