UDACITY

Back to Machine Learning Engineer Nanodegree

Predicting Boston Housing Prices

REVIEW

CODE REVIEW

HISTORY

SHARE YOUR ACCOMPLISHMENT!

Meets Specifications

Hello Learner,

Your work has been outstanding so far and I am so glad to have reviewed this document. It was really easy to go through and enjoy looking at such maturity from a student such early into this Nanodegree.

Keep it up and goodluck for the future projects. 😎

Data Exploration

All requested statistics for the Boston Housing dataset are accurately calculated. Student correctly leverages NumPy functionality to obtain these results.

Excellent work! You successfully implemented NumPy to calculate all the requested statistics from the dataset.

Here is a link to get some performance tips for NumPy.

Here is a course in udacity actually teaching us how to use NumPy in more depth: intro to data analysis watch it if you want to learn more about it.

Here is a website which provide lots of example work about most common used numpy function

Here are a few things to note. Usually you can get a good speed-up just by thinking about what your memory is doing. In NumPy, it's really easy to accidentally do a copy or cast that you really didn't need, and have that be your bottleneck. The thing to do is to be sure you initialize your data structures in the types and containers you want from the start and not move them around (like passing them through functions constantly). If you have some relative sparsity to your problem, use sparse matrices, unless you are changing the structure of your matrix in which case you should be careful which sparse matrix implementation you choose (there is at least one that is good for when you have to change structure a lot). Try to do everything in-place, and always use Numpy functions instead of generic Python (e.g. max() instead of np.amax(), the former will cast your array into a Python list)

Student correctly justifies how each feature correlates with an increase or decrease in the target variable.

Great explanation here. You have a good understanding of the concepts so far.

Developing a Model

Student correctly identifies whether the hypothetical model successfully captures the variation of the target variable based on the model's R^2 score.
The performance metric is correctly implemented in code.

Very well, good job here. Here is a document for better understanding.

Student provides a valid reason for why a dataset is split into training and testing subsets for a model. Training and testing split is correctly implemented in code.

Well done here! You explained this concept really well.
This page from Amazon gives really ample information about why a dataset is to be split into a training and evaluation set.

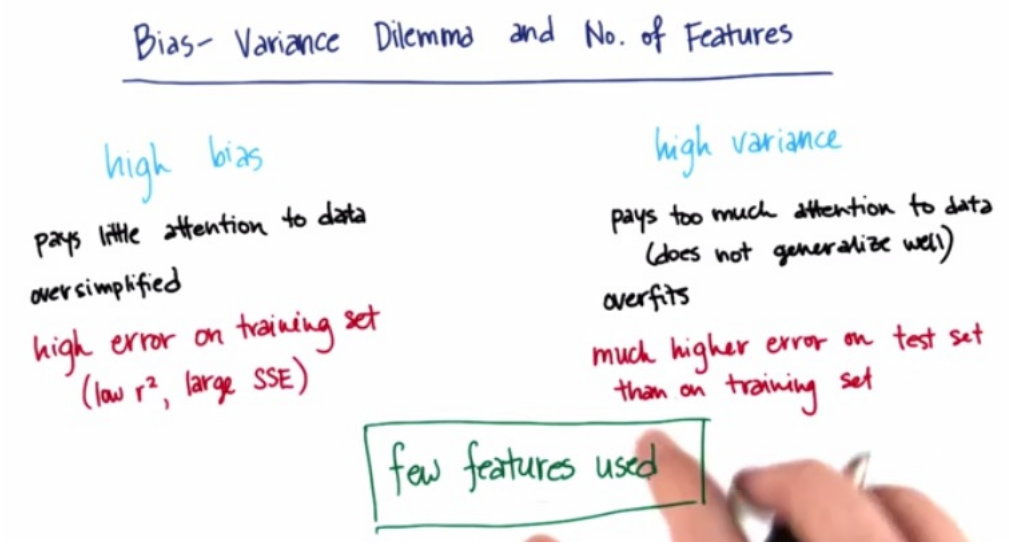
Analyzing Model Performance

Student correctly identifies the trend of both the training and testing curves from the graph as more training points are added. Discussion is made as to whether additional training points would benefit the model.

Good job!

Student correctly identifies whether the model at a max depth of 1 and a max depth of 10 suffer from either high bias or high variance, with justification using the complexity curves graph.

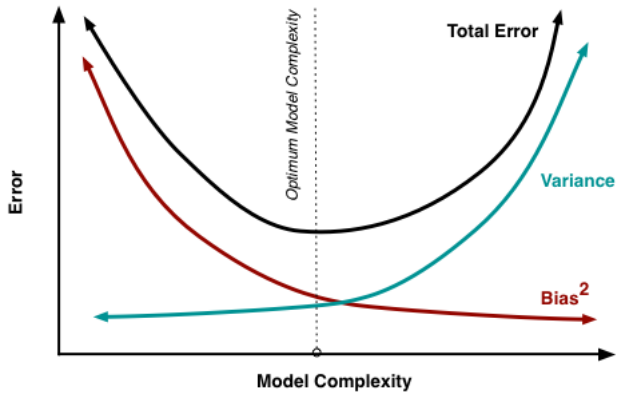
You're right. Have a look here for reference:



Student picks a best-guess optimal model with reasonable justification using the model complexity graph.

A well thought out answer. Keep it up!

Check out the following diagram, which amazingly summarises the concept of bias-variance tradeoff in Machine Learning:



Evaluating Model Performance

Student correctly describes the grid search technique and how it can be applied to a learning algorithm.

Hands down one of the best explanations for this answer yet! Keep it up!

Great work yet again. Here is a link if you want to go more in-depth to study this.

Another very powerful parameter tuning algorithm is RandomizedSearchCV. In contrast with GridSearchCV, not all parameters are tried out, but rather a fixed number of parameter settings is sampled from the specified distributions.
One particular advantage of RandomizedSearchCV is that it is much faster than GridSearchCV, and it is theoretically proven to find models that are as good; or even better than grid search.

Student correctly describes the k-fold cross-validation technique and discusses the benefits of its application when used with grid search when optimizing a model.

Good job!

Student correctly implements the `fit_model` function in code.

Stellar work!

Student reports the optimal model and compares this model to the one they chose earlier.

Student reports the predicted selling price for the three clients listed in the provided table. Discussion is made for each of the three predictions as to whether these prices are reasonable given the data and the earlier calculated descriptive statistics.

Very well!

Student thoroughly discusses whether the model should or should not be used in a real-world setting.

Very thorough and logical!

DOWNLOAD PROJECT

RETURN TO PATH

Rate this review

Student FAQ