UDACITY

PROJECT

Predicting Boston Housing Prices

A part of the Machine Learning Engineer Nanodegree Program

PROJECT REVIEW

CODE REVIEW

NOTES

SHARE YOUR ACCOMPLISHMENT!

Requires Changes

4 SPECIFICATIONS REQUIRE CHANGES

Data Exploration

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

Well done on getting the statistics using `numpy`. For most real world projects, getting statistics of the data is the first step to understand the data before subsequent feature engineering can take place. We may also examine if there are any missing values and outliers.

There are other statistical measures supported by `numpy`. For example, we can look at the percentile of the data:

```
first_quartile = np.percentile(prices, 25)
third_quartile = np.percentile(prices, 75)
```

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

You are on the right track. To make your answer to Q2 perfect, can you elaborate on your reasoning behind your conclusions please?

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.**

With this high R^2 value we may say that the model has done a good job to capture variation of the data.

However, as a cautious note, we only have five points here, and it may be hard to draw conclusion that is statistically significant.

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.

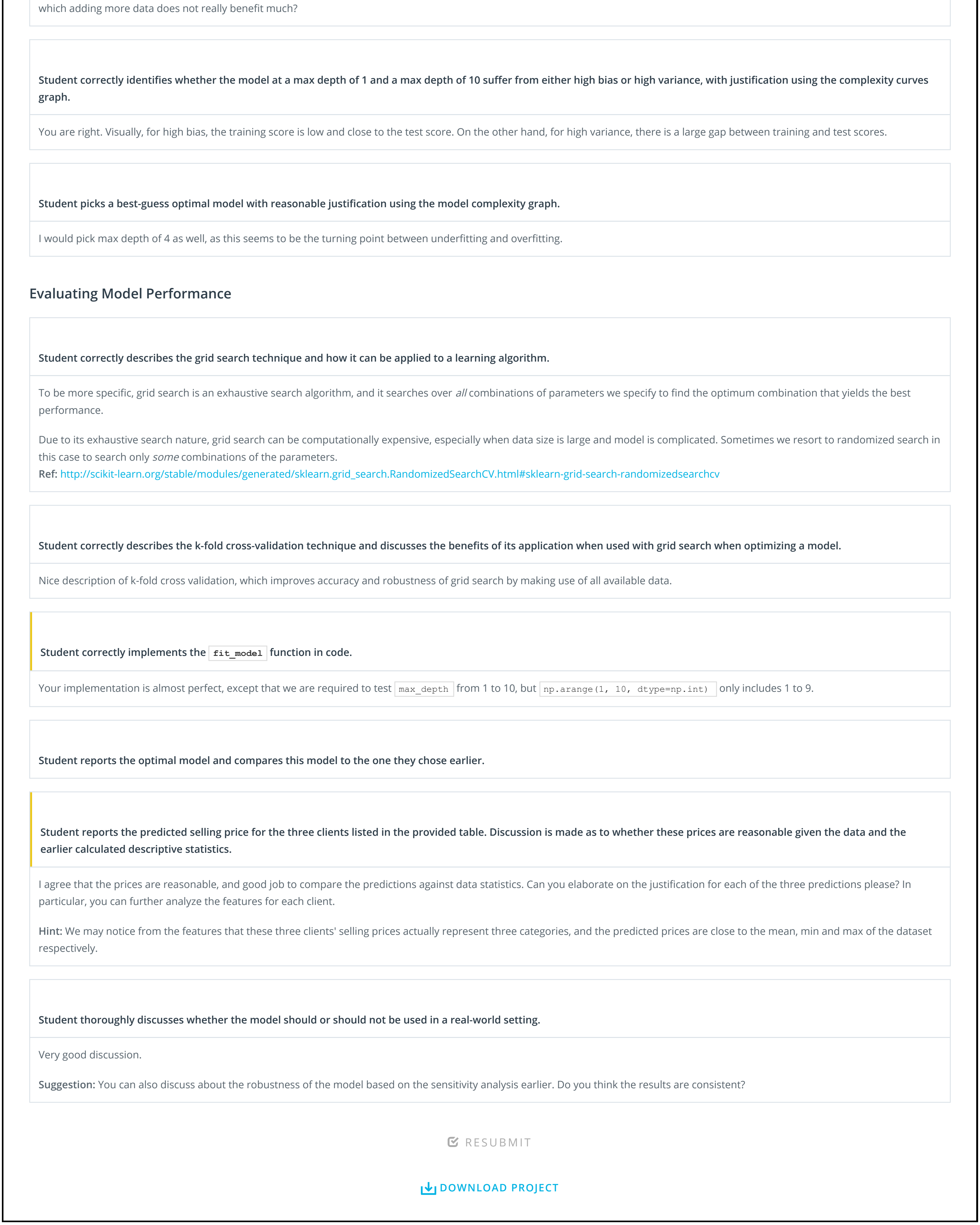
Well done on splitting the data, and you have successfully captured the key benefit of doing it.

Without a testing set being reserved, there is no data to validate the model to ensure it is working well before being used in the general context. Therefore we need to train the model using training subset and test it over the testing subset to confirm whether the predictions being made on *unseen* data are correct.

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.

I would not really agree that more data is always beneficial. You may note that there are actually two phases in the testing score with different rates of change. So the benefit of adding more data differs in these two phases. I suggest you to elaborate the two phases separately. Some questions to answer are, which phase benefits more from more data, and is there a limit beyond



which adding more data does not really benefit much?

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 are right. Visually, for high bias, the training score is low and close to the test score. On the other hand, for high variance, there is a large gap between training and test scores.

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

I would pick max depth of 4 as well, as this seems to be the turning point between underfitting and overfitting.

Evaluating Model Performance

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

To be more specific, grid search is an exhaustive search algorithm, and it searches over *all* combinations of parameters we specify to find the optimum combination that yields the best performance.

Due to its exhaustive search nature, grid search can be computationally expensive, especially when data size is large and model is complicated. Sometimes we resort to randomized search in this case to search only *some* combinations of the parameters.
Ref: http://scikit-learn.org/stable/modules/generated/sklearn.grid_search.RandomizedSearchCV.html#sklearn-grid-search-randomizedsearchcv

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.

Nice description of k-fold cross validation, which improves accuracy and robustness of grid search by making use of all available data.

Student correctly implements the `fit_model` function in code.

Your implementation is almost perfect, except that we are required to test `max_depth` from 1 to 10, but `np.arange(1, 10, dtype=np.int)` only includes 1 to 9.

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 as to whether these prices are reasonable given the data and the earlier calculated descriptive statistics.

I agree that the prices are reasonable, and good job to compare the predictions against data statistics. Can you elaborate on the justification for each of the three predictions please? In particular, you can further analyze the features for each client.

Hint: We may notice from the features that these three clients' selling prices actually represent three categories, and the predicted prices are close to the mean, min and max of the dataset respectively.

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

Very good discussion.

Suggestion: You can also discuss about the robustness of the model based on the sensitivity analysis earlier. Do you think the results are consistent?

Best practices for your project resubmission

Ben shares 5 helpful tips to get you through revising and resubmitting your project.

▶ Watch Video (3:01)

Have a question about your review? Email us at review-support@udacity.com and include the link to this review.

RETURN TO PATH

Rate this review

Student FAQ