UDACITY

< Back to Machine Learning Engineer Nanodegree

# Predicting Boston Housing Prices

REVIEW

CODE REVIEW

HISTORY

## Requires Changes

7 SPECIFICATIONS REQUIRE CHANGES

Great work as a first submission..!!

You have understood most of the concepts well. There are still a few misconceptions that still need some clarity and hence I urge you to dive deeper. Read the materials that I have provided and I am sure you will gain some more insight. Keep up the good work. 😃

Thanks and do rate my review.

Happy Learning..!!

### Data Exploration

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

Good work implementing the code in NumPy. 😃

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

variable.

You are correct with the relationships here..!! But you need to justify each relationship as well. 😃

e.g.
Higher LSTAT means lower MEDV. But why? Probably because as the number of low class workers in a locality increase, the locality becomes less desirable and hence the prices of houses in that locality decrease.

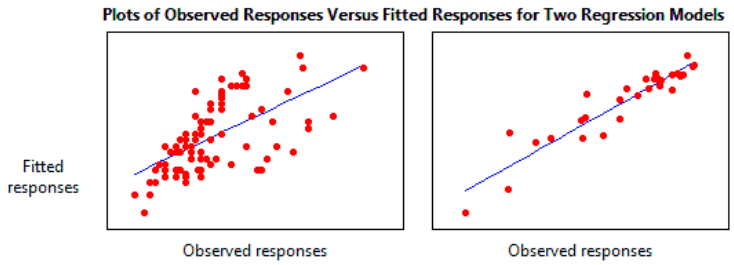
## Developing a Model

**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.**

Good work here..!! I also want you to explain R^2 score as a measure of goodness of fit here. 😃

Here is something that should help you gain intuition on how R^2 score is a measure of goodness of fit:

Plotting fitted values by observed values graphically illustrates different R-squared values for regression models.




The regression model on the left accounts for 38.0% of the variance while the one on the right accounts for 87.4%. The more variance that is accounted for by the regression model the closer the data points will fall to the fitted regression line. Theoretically, if a model could explain 100% of the variance, the fitted values would always equal the observed values and, therefore, all the data points would fall on the fitted regression line.

Some more info for you to dive deeper: link

**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.**

You are correct with the explanation of the ratio in which we split our data. I want you to answer one more question here:

- Why do we split our data into train and test?

Here are a few points that should help you answer this question:

- Suppose you use all your data for training. What do you think will happen in this scenario?
- Your model might overfit or underfit or it might be a good fit as well. If your model is underfitting you will know that from your train scores (which will be low). But if your model is overfitting, your train scores will be high. By first look it would seem you have done a great job in training, but most probably you have not. It is only after you test your model with the test set that you will know the real performance of your model. Using part of train set to test will always give good results if you are

performance of your model. Using part of train set to test will always give good results if you are getting good train score (since your model is biased towards that data)

- That's why you always need a test set. But while training you cannot use your test set performance to improve your model, because that would make your model biased towards the test set. Hence you need to do cross-validation.

Further let me define the concepts of underfitting, overfitting and the concept of being a good fit:

- UNDERFITTING: In this scenario, the model is not able to learn from the training (it is too simple) and hence it doesnot perform well on the test set (it wont perform well on train data as well).
- OVERFITTING: In this scenario, the model is over-complex and is trying to just memorize the train data, hence it doesnot generalize well on the test data.
- GOOD FIT: In this scenario, the model is neither too simple nor too complex, it is able to learn from the training data and hence it generalizes well to test data as well.

If you further want to know why we split our data, here is a good explanation: link

## 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.**

Great work..!! You understand how to check the performance of a model using learning curves. 😃

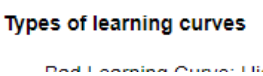
Here is a blog that explains learning curve: link

**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.**

There is one more thing that you need to explain here:

- What visual cues in the graph justify your conclusions for `max_depth=1` scenario?

Here are a few points on bias-variance tradeoff that should help you answer this question:



- Bad Learning Curve: High Bias
  - When training and testing errors converge and are high
    - No matter how much data we feed the model, the model cannot represent the underlying relationship and has high systematic errors
    - Poor fit
    - Poor generalization
- Bad Learning Curve: High Variance
  - When there is a large gap between the errors
    - Require data to improve
    - Can simplify the model with fewer or less complex features
- Ideal Learning Curve
  - Model that generalizes to new data
  - Testing and training learning curves converge at similar values
  - Smaller the gap, the better our model generalizes

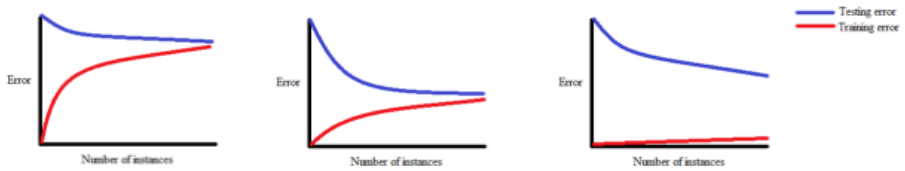



Image #1 (high bias) Image #2 (ideal) Image #3 (high variance)

Some more info for you to dive deeper: link

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

Wow..!! Even I believe the same. 😃

## Evaluating Model Performance

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

There is a slight misconception here:

- by evaluating the model with the grid: all the combination of hyper parameters.

The grid is not just the combination of hyper-parameters, it is a combination of hyper-parameters and their probable values.

Here are a few points on gridSearch that should help you clarify the concept:

- It is an algorithm with the help of which we can tune hyper-parameters of a model. We pass the hyper-parameters to tune, the possible values for each hyper-parameter and a performance metric as input to the grid search algorithm. The algorithm will then place all the possible hyper-parameter combination in a grid and then find the performance of the model for each combination against

some cross-validation set. Then it outputs the hyper-parameter combination that gives the best result

result.

Here is the official sklearn page on gridSearch: link

Here is another great answer on how gridSearch works: link

**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.**

There is a slight misconception in your answer here:

- k-fold cross-validation split the train data set to k pieces and train the model k times with k th portion of the data being validation dataset and the rest k-1 portion being train data

We donot apply k-fold on the whole dataset. Instead we first split our data into train and test and then apply k-fold on the train set. That is how cross-validation works.

Here are a few more points on k-fold so that you gain some more intuition:

- There is a `huge` difference between `testing` and `cross-validation`.
- K-fold is a cross-validation technique and not a testing technique.
- Suppose you use all your data for training. What do you think will happen in this scenario?
- Your model might overfit or underfit or it might be a good fit as well. If your model is underfitting you will know that from your train scores (which will be low). But if your model is overfitting, your train scores will be high. By first look it would seem you have done a great job in training, but most probably you have not. It is only after you test your model with the test set that you will know the real performance of your model. Using part of train set to test will always give good results if you are getting good train score (since your model is biased towards that data)
- That's why you always need a test set. But while training you cannot use your test set performance to improve your model, because that would make your model biased towards the test set. Hence you need to do cross-validation.
- Using normal cross-validation also has its disadvantages, since you use up a part of your training data. Hence, here k-fold comes to the rescue.

Here is some more info on k-fold CV: link

You can check the difference between cross-validation and testing here: link

**Student correctly implements the `fit_model` function in code.**

Good work here..!! 😃

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

Correct..!! 😃

**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.**

Here I want you to explain a little bit more. 😃

The main idea here is to explain your predictions to a person who has not seen the data or doesnot know what features you have selected to reach these predictions.

You can follow the below steps to answer this question:

- First try to compare the predictions to the descriptive statistics that you calculated in first TODO (min, max, mean etc.)
- Then compare the predictions for each client with each other.
- Finally dive deeper into the features for each client and then justify if the prices are reasonable.

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

Great discussion..!! I agree with you. 😃

RESUBMIT

DOWNLOAD PROJECT

Learn the best practices for revising and resubmitting your project.

RETURN TO PATH

Rate this review

Student FAQ