UDACITY

Back to Machine Learning Engineer Nanodegree

Predicting Boston Housing Prices

REVIEW

HISTORY

Meets Specifications

Great work..!!

You have read the previous reviewers comments and improved your answers. I would still urge you to dive deeper and read all the materials that I have provided. I am sure there is something more to learn. Keep up the good work. 😃

Thanks and do rate my review.

Happy Learning..!!

Some extra 5 minute reads on regression:

- Basics of Linear Regression
- Advanced regression Techniques

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.

Wow..!! You understand the features and their relationships well. 😃

Developing a Model

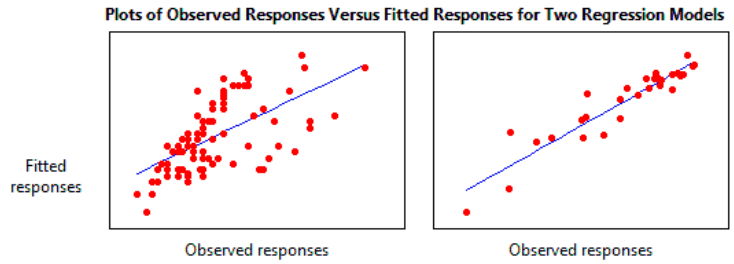
Student correctly identifies whether the hypothetical model successfully captures the variation of the target variable based on the model's R^2 score.
The performance metric is correctly implemented in code.

Good work..!! 😃

What I want you to understand is how R^2 score is a measure of goodness of fit. Think about the distance between the actual point (Y) and the predicted point (Y') on the regression line.

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.

Good work..!! You understand why we split our data into train and test. 😃

Here are a few additional points on why we split:

- 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.

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.

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

Here is another blog that explains learning curves in depth: 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.

Great..!! You understand bias-variance tradeoff well. It is a very important concept in machine learning and hence I urge you to dive deeper.

Here is some more info on bias-variance tradeoff:

Types of learning curves

- 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

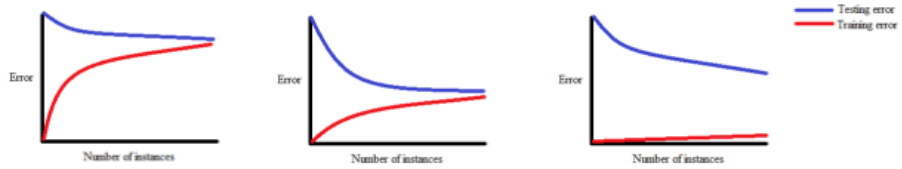


Image #1 (high bias) Image #2 (ideal) Image #3 (high variance)

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.

Awesome..!! You understand gridSearch well. 😃

Here are a few additional points on gridSearch so that you don't miss anything:

- 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.

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.

Great work explaining k-fold cross-validation..!! 😃

Here are a few additional points on k-fold to always keep in mind:

- 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 implementation..!! 😃

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.

Good work here..!! 😃

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

Great discussion..!! I agree with your points. 😃

DOWNLOAD PROJECT

RETURN TO PATH