UDACITY

‹ Return to "Machine Learning Engineer Nanodegree" in the classroom

DISCUSS ON STUDENT HUB

Predicting Boston Housing Prices

REVIEW

CODE REVIEW

HISTORY

Requires Changes

2 SPECIFICATIONS REQUIRE CHANGES

Hi,

It's a pleasure to review your project, this is a very good submission. Just a few more details and you are done here.

Keep up the good work 🙌 and count on us!

Best regards,

Data Exploration

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

Great job using Numpy here!!! It is always important to have a basic understanding of our data set before diving into the problem. Remember that you should always explore your data before attempting to train a model on it.

A "dump" classifier that only predicts the average, would have as output the value of 454,342.94 for all houses.

TIP

It is always important to be aware of the tools you use. For example, the Pandas Series.std () will give you different results by default:

```
>>> pd.Series([7,20,22,22]).std()
7.2284161474004804
>>> np.std([7,20,22,22])
6.2599920127744575
```

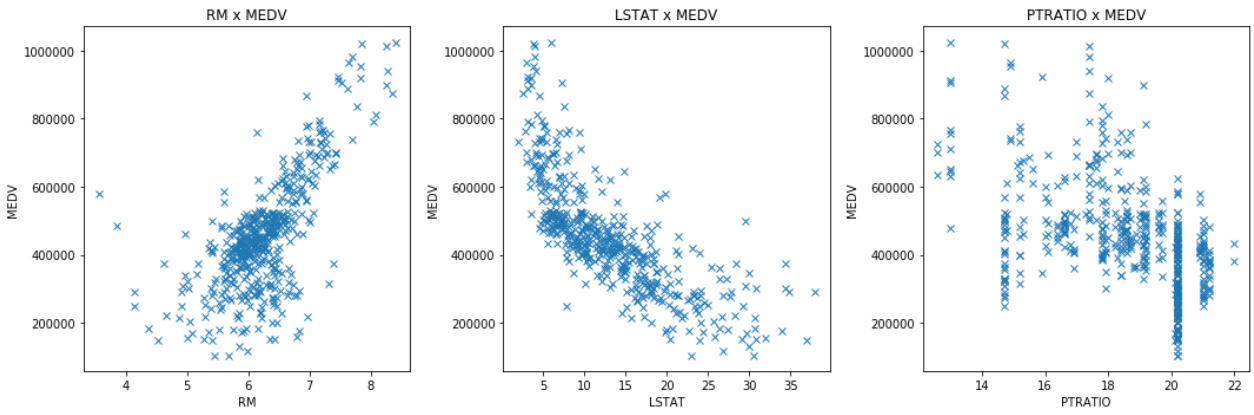
Student correctly justifies how each feature correlates with an increase or decrease in the target variable.

Excellent intuition!

A good machine learning engineer should always validate their intuition with more data exploration.

Here's a simple code snippet you can run in the Data Exploration section of the notebook to confirm your intuition about the data:

```
import matplotlib.pyplot as plt
plt.figure(figsize=(15, 5))
for i, col in enumerate(features.columns):
    plt.subplot(1, 3, i+1)
    plt.plot(data[col], prices, 'x')
    plt.title('%s x MEDV' % col)
    plt.xlabel(col)
    plt.ylabel('MEDV')
```



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.

Perfect! A high R^2 score may indicate a strong correlation between the predicted value and the true value.

Also, evaluating any model on a single metric can be deceiving. I recommend reading more about R^2 caveats. You will learn that the R^2 score won't tell you everything you need to know about your model's performance.
Link: https://en.wikipedia.org/wiki/Coefficient_of_determination#Caveats

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.

OK, great!

We would have no way to check for overfitting if we used the entire dataset only for training.

Remember, our goal is to build a model that generalizes well on any new unseen data. Splitting the dataset doesn't automatically guarantee the model won't overfit.

Here is the helpful link to a discussion:
https://www.researchgate.net/post/What_is_the_best_way_to_divide_a_dataset_into_training_and_test_sets

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.

Please, review this part:

We can clearly see that the curve has converged to its optimal score, so more data is not necessary.

In practice, collecting and labeling more data can often be time consuming and expensive, so it's a good idea to avoid having to collect more data.

TIP

However, note that some algorithms (like deep learning or XGboosting) can make use of more and more data to improve their performance.

Great post: https://machinelearningmastery.com/much-training-data-required-machine-learning/

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 have a pretty good understanding of bias/variance tradeoffs.

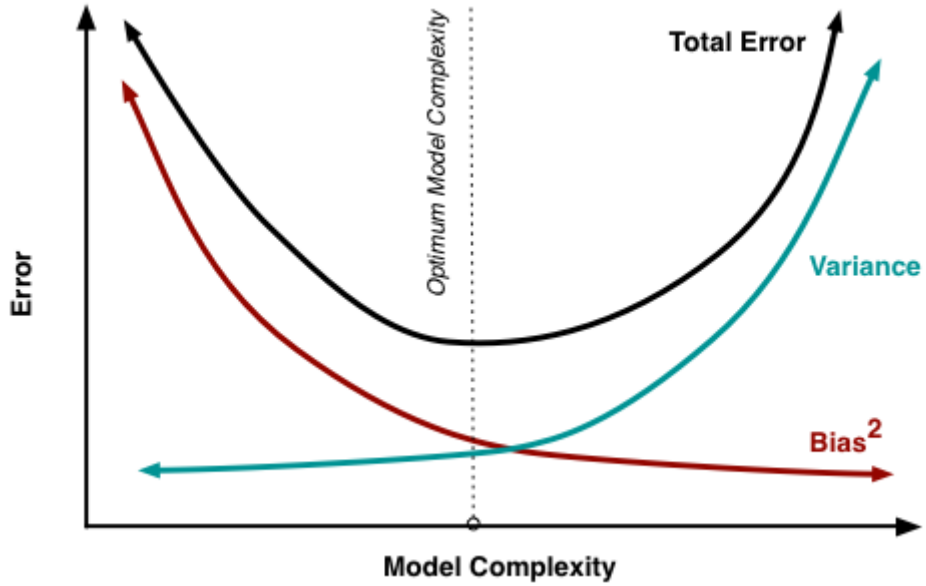
In this link here a good example of Underfitting vs. Overfitting with a different view: http://scikit-learn.org/stable/auto_examples/model_selection/plot_underfitting_overfitting.html

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

All good!

As we are looking for the highest validation score(which is what gridSearch searches for). And we are also looking for a good bias / variance tradeoff (with close training and validation scores while the training score is high).

Check out this image:

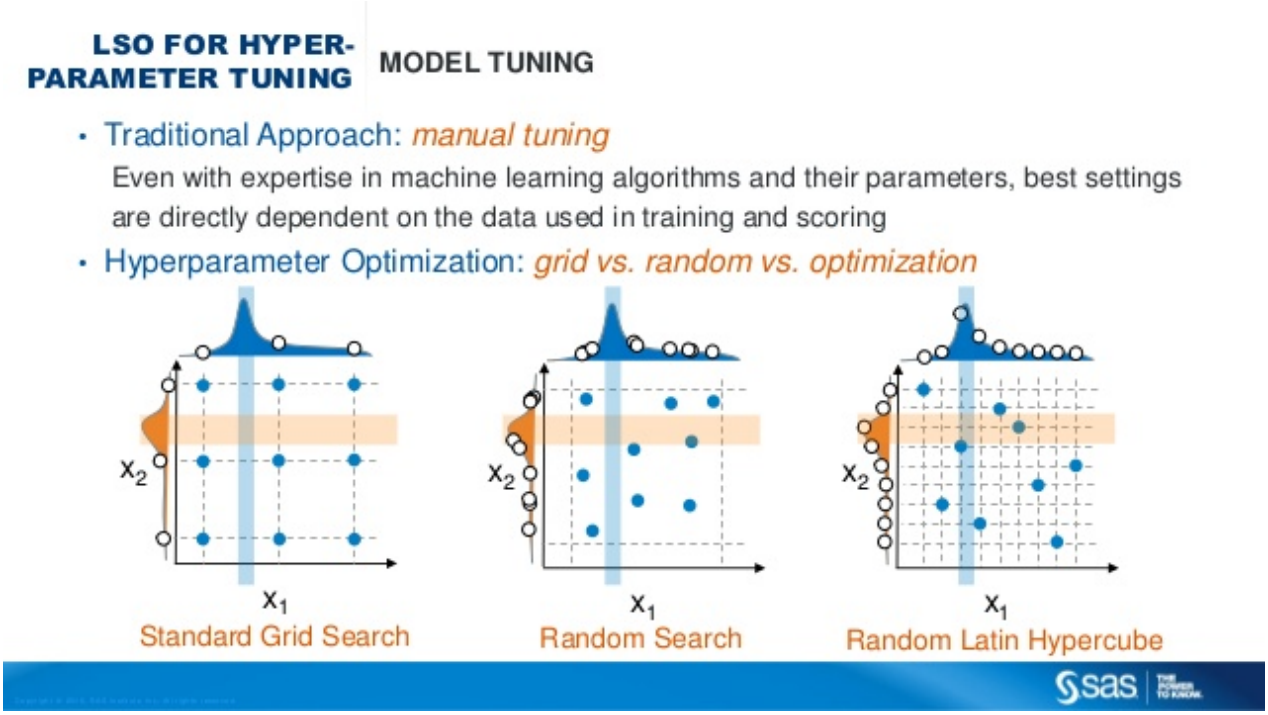


Evaluating Model Performance

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

Great discussion!

We can also apply random search or optimized search to find the best parameters. Here is an image showing the options:



Here is the sklearn guide for grid search: https://scikit-learn.org/stable/modules/grid_search.html#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 discussion here!

LINKS:

- What Is K-Fold Cross Validation? https://magoosh.com/data-science/k-fold-cross-validation/
- What is the difference between test set and validation set? https://stats.stackexchange.com/questions/19048/what-is-the-difference-between-test-set-and-validation-set

Student correctly implements the `fit_model` function in code.

Good use of the `range` function and the `random_state` parameter.

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

Great!!!

TIP

If you run the model multiple times, you might have different values for the depth.

To avoid this, you should set random_state for all functions that support this argument to ensure consistent reproducible results. The functions are:

- train_test_split
- ShuffleSplit
- DecisionTreeRegressor

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.

A good idea would be to compare these to the descriptive stats of the features. You can run:

```
features.describe()
```

Please justify whether you think the prices are reasonable or not based on statistics.

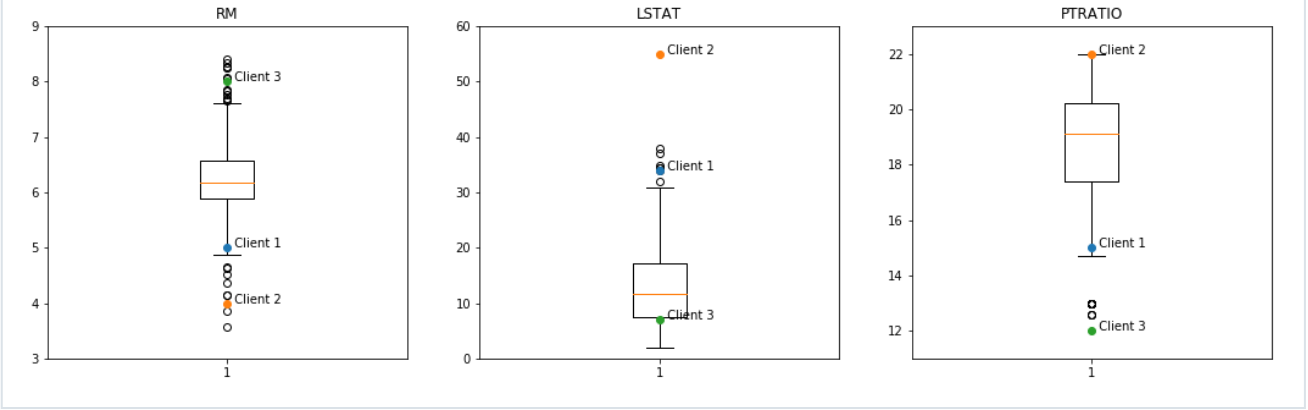
What is important here is that the different clients present different “categories” of selling prices.

1. The first client we could consider to be an “average” homeowner, based on their feature set, and so the price seems rather average (perhaps a bit low in Boston).
2. The second client would be considered a more “poor” home owner, which is indicated by the low selling price of the home.
3. The final client is an “expensive” home owner, based on the fact their house is large and they are quite wealthy (LSTAT).

I recommend plotting the clients over the data set to visualize how their homes compare to the market in the individual features. You can do this easily by running the following snippet:

```
import matplotlib.pyplot as plt
plt.figure(figsize=(15, 5))
for i, col in enumerate(features.columns):
    plt.subplot(1, 3, i+1)
    plt.boxplot(data[col])
    plt.title(col)
    for j in range(3):
        plt.plot(1, client_data[j][i], marker='o')
        plt.annotate('Client %s' % str(j+1), xy=(1, client_data[j][i]))
```

Here is the output:



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

Great!

RESUBMIT

DOWNLOAD PROJECT

Best practices for your project resubmission

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

Watch Video (3:01)

RETURN TO PATH

Rate this review