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

This is a very impressive submission. Just need a few minor adjustments and you will be golden, but also check out some of the other ideas presented in this review. One tip here would be that some of these topics are extremely important as you embark on your journey throughout your Machine Learning career and it will be well worth your time to get a great grasp on these topics before you dive deeper in. Keep up the hard work!!

### Data Exploration

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

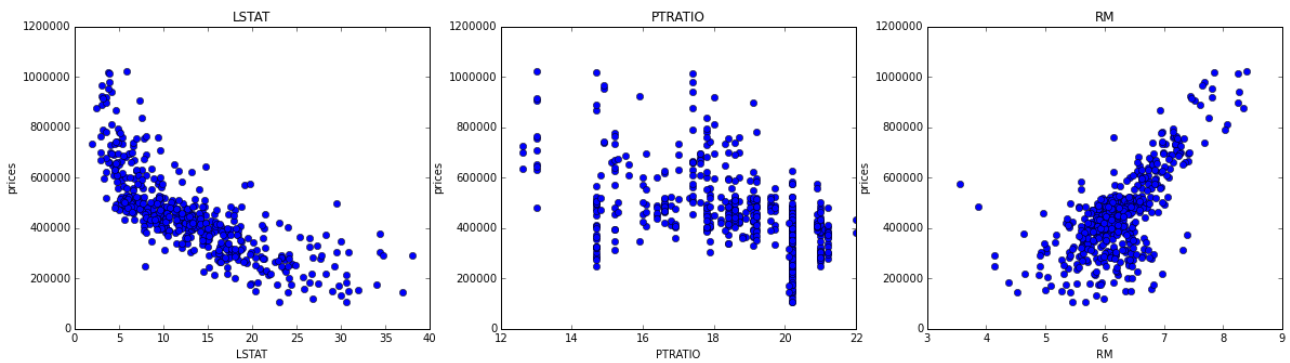
Although these are the correct ideas, for this section you should leverage NumPy functionality to obtain these results. Therefore you are actually using pandas `Series()` commands to receive these statistics in your implementation. Therefore I bet numpy has the same methods!! You have all these commented out.

```
minimum_price = np.min(prices)
....
```

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

Nice observations for the features in this dataset. As we can confirm these ideas by plotting each feature vs MEDV housing prices.

```
import matplotlib.pyplot as plt
plt.figure(figsize=(20, 5))
for i, col in enumerate(features.columns):
    plt.subplot(1, 3, i+1)
    plt.plot(data[col], prices, 'o')
    plt.title(col)
    plt.xlabel(col)
    plt.ylabel('prices')
```



## 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.**

Would recommend expanding a bit more. How does this compare to the optimal score? How do the 'true values' and 'predictions' compare? etc... Could also think about if more data points would allow us to be more confident in this model?

R-squared is a statistical measure of how close the data are to the fitted regression line. It is also known as the coefficient of determination, or the coefficient of multiple determination for multiple regression. The definition of R-squared is fairly straight-forward; it is the percentage of the response variable variation that is explained by a linear model. Or:

- R-squared = Explained variation / Total variation

R-squared is always between 0 and 100%:

- 0% indicates that the model explains none of the variability of the response data around its mean.
- 100% indicates that the model explains all the variability of the response data around its mean.

In general, the higher the R-squared, the better the model fits your data. So with a high value of 92.3% (0.923) we can clearly see that we have strong correlation between the true values and predictions.

**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.**

> "With less training data, the parameter estimates have greater variance. With less testing data, the performance statistic will have greater variance."

First off, you have "greater variance" twice here, is one of these suppose to be high bias? Also these comments aren't necessarily true. We can actually still see underfitting or overfitting if the training data is small or if the training data is large(as it depends on the type of model and the parameter values of the model). For example you can see that a decision tree overfits the training data with a small training set(shown in the next section). But a linear model like linear regression could underfit if we don't have enough data. But both both would be bad models.

Therefore just focus on *why* we actually need a testing set. What does it tell us about our model? What can we try and 'protect against' or 'detect' being able to test it with an independent dataset? etc...

Links

- Lecture
- (http://info.salford-systems.com/blog/bid/337783/Why-Data-Scientists-Split-Data-into-Train-and-Test)

## 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.**

You are correct with your comment of

> "adding more training points will not benefit this model"

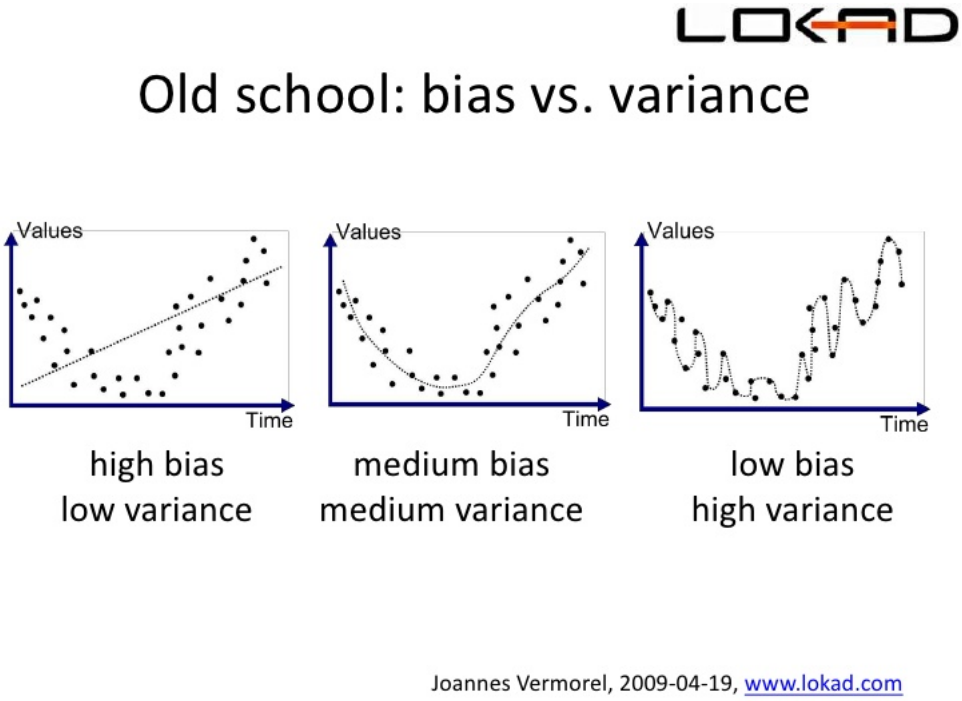
As in the beginning it is beneficial, but at the end if we look at the testing curve here, we can clearly see that it has converged to its optimal score, so more data is not necessary.

Therefore lastly for this section make sure you also describe the initial trends on the training and testing curves, before they converge(increasing or decreasing?). As this is key in terms of how models initially scale to more data.

**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.**

Nice justification here! As the low training score is what truly depicts high bias. You clearly understand the bias/variance tradeoff.

- As a max_depth of 1 suffers from high bias, visually this is due to the low training score(also note that it has low variance since the scores are close together). As this model is not complex enough to learn the structure of the data
- And a max_depth of 10 suffers from high variance, since we see a large gap between the training and validation scores, as we are basically just memorizing our training data and will not generalize well to new unseen data

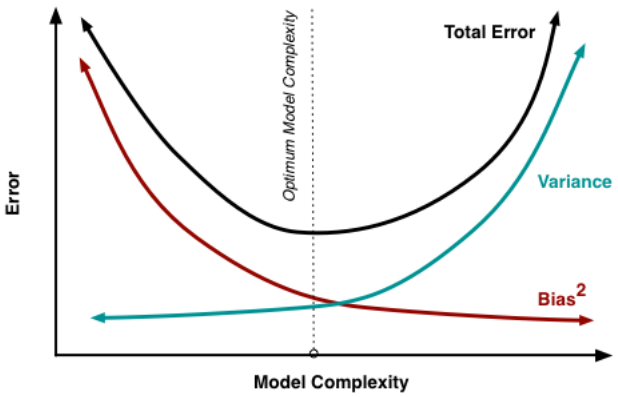



✓

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

I would choose the same! As we are definitely 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).

Check out this visual, it refers to error, but same can be applied to accuracy(just flipped)




## Evaluating Model Performance

✓

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

Excellent! And very nice example here with a decision tree and max depth in this project. Can also note that since this "*exhaustively evaluate the model performance with every combination of hyperparameter*", one limitation of GridSearch is that it can be very computationally expensive when dealing with a large number of different hyperparameters and much bigger datasets. Therefore there are two other techniques that we could explore to validate our hyperparameters

- RandomizedSearchCV which can sample a given number of candidates from a parameter space with a specified distribution. Which performs surprisingly well!
- Or a train / validation / test split, and we can validate our model on the validation set. Often used with much bigger datasets

✓

**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 description of the k-fold cross-validation technique, probably the most use CV method in practice.

---

> "In grid search, there is a danger to overfit to the validation set since we use it many times to evaluate performance of different points on the grid and choose a point that delivered good performance. Hence, with more grid points, we are more likely to find a point that is good only by chance. With k-fold cross-validation, the

> overfitting problem is mitigated since our effective validation set size is larger."

Good. This is an extremely important concept in machine learning, as this allows for multiple testing datasets and is not just reliant on the particular subset of partitioned data. For example, if we use single validation set and perform grid search then it is the chance that we just select the best parameters for that specific validation set. But using k-fold we perform grid search on various validation set so we select best parameter for generalize case. Thus cross-validation better estimates the volatility by giving you the average error rate and will better represent generalization error.

If you would like a full run example, run this code based on the iris data set in your python shell or something and examine the print statements, as this is a great example

```
import numpy as np
from sklearn import cross_validation
from sklearn import datasets
from sklearn import svm

iris = datasets.load_iris()

# Split the iris data into train/test data sets with 30% reserved for testing
X_train, X_test, y_train, y_test = cross_validation.train_test_split(iris.data, iris.target, test_size=0.3, random_state
=0)

# Build an SVC model for predicting iris classifications using training data
clf = svm.SVC(kernel='linear', C=1, probability=True).fit(X_train, y_train)

# Now measure its performance with the test data with single subset
print clf.score(X_test, y_test)

# We give cross_val_score a model, the entire data set and its "real" values, and the number of folds:
scores = cross_validation.cross_val_score(clf, iris.data, iris.target, cv=5)

# Print the accuracy for each fold:
print scores

# And the mean accuracy of all 5 folds:
print scores.mean()
```

✓

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

Nice implementation! Could also set a `random_state` in your DecisionTreeRegressor for reproducible results.

```
regressor = DecisionTreeRegressor(random_state = "any number")
```

🔄

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

Make sure you also address the question of

HOW DOES THIS RESULT COMPARE TO YOUR GUESS IN QUESTION 6?

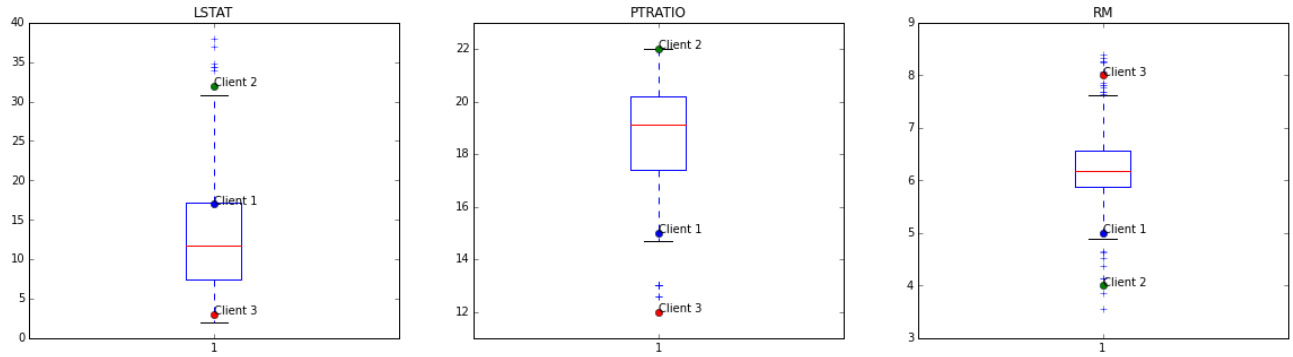
✓

**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.**

Excellent justification for these predictions by comparing them to the features and descriptive stats. Maybe also plot some box plots and see how the Client's features compare to the inter quartile range, median, whiskers

```
import matplotlib.pyplot as plt
plt.figure(figsize=(20, 5))
y_ax = [[3,9],[0,40],[11,23]]
for i, col in enumerate(features.columns):
    plt.subplot(1, 3, i)
    plt.boxplot(data[col])
    plt.title(col)
    for j in range(3):
        plt.plot(1, client_data[j][i], marker="o", )
```

```
        plt.annotate('Client '+str(j+1), xy=(1,client_data[j][i]))
        plt.ylim(y_ax[i])
```




A more advanced and great idea would be to compare these to the descriptive stats of the features. We can compute the five number summary of the descriptive stats of the features with

```
features.describe()
```

For example -- Client 3 RM value is in the top 25 percentile in the data, while being near the minimum value for LSTAT, and since an increase in RM would lead to an increase in MEDV. An increase in LSTAT would lead to a decrease in MEDV. The predicted price near the maximum value in the dataset makes sense

✓

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

Would agree, as this dataset is quite old and probably doesn't capture enough about housing features to be considered robust!

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

Student FAQ