Meets Specifications

Dear student,  
Thanks for updating your answers based on the previous reviewer's feedback and congratulations on completing the first mandatory project of MLND! 🎉

You seem to have solid understanding of the topic and I wish you good luck with the upcoming projects! 🙂

**Data Exploration**

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

Great start!  
You've correctly leveraged the power of NumPy to get the basic statistics of your data set. Remember you should *always explore* your data before trying to train a model on it. You'll see different variations of data exploration throughout this nanodegree.

It's always important to be aware of tools you use. For example, the Pandas' Series.std() will by default give you different result than numpy.std().

>>> pd.Series([7,20,22,22]).std()

7.2284161474004804

>>> np.std([7,20,22,22])

6.2599920127744575

The code was taken from [the StackOverflow thread that also explains why this is the case](https://stackoverflow.com/questions/25695986/pandas-why-pandas-series-std-is-different-from-numpy-std).

**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. Remember visualisations are powerful way of finding the latent feature correlations and presenting your intuitions to your boss or a client.

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)

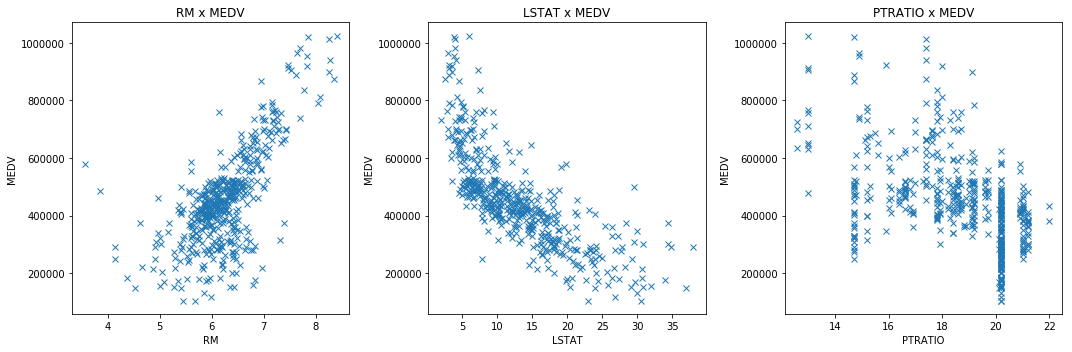
plt.plot(data[col], prices, 'x')

plt.title('%s x MEDV' % col)

plt.xlabel(col)

plt.ylabel('MEDV')

You will get a plot similar to this one

[](https://udacity-reviews-uploads.s3.us-west-2.amazonaws.com/_attachments/173742/1521931147/price_correlations.png)

**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.**

You are correct.  
The R^2 score of 0 means the model cannot successfully predict the target variable y from features X; while R^2 score of 1 means *the perfect* capture of the data variation—thus we can say having a score so close to 1 means we have a pretty decent fit.

On the other hand, we should always remember that evaluating any model on a single metric can be deceiving. I really recommend reading more about [R^2 caveats](https://en.wikipedia.org/wiki/Coefficient_of_determination#Caveats)—you will learn that the R^2 score won't tell you *everything* you need to know about your model's performance.

**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.**

That is right!

Although having *“a lot of”* training data can sometimes lead to overfitting model, the main sources of overfitting and underfitting are usually the model complexity (as in setting of hyper parameters).

So remember that splitting the dataset definitely **doesn't** automatically guarantee the model won't overfit.

Our goal is to create a model that generalizes well on any new (unseen) data and setting aside a testing set *allows us to measure* the model's performance on such unseen data.

**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 observations!  
However I would argue that adding more points probably wouldn't be beneficial since the model's training/testing curves after ~300 data points are becoming flat with no tangible improvement. It's also important to note that collecting additional data in real life scenario might be really time consuming and expensive and doesn't ensure improvement in your model—thus it might be useful to plot something like these learning curves to determine whether additional data collection is required.

On the other hand, note that some algorithms (mostly those in deep learning) can make use of more and more data to improve their performance.

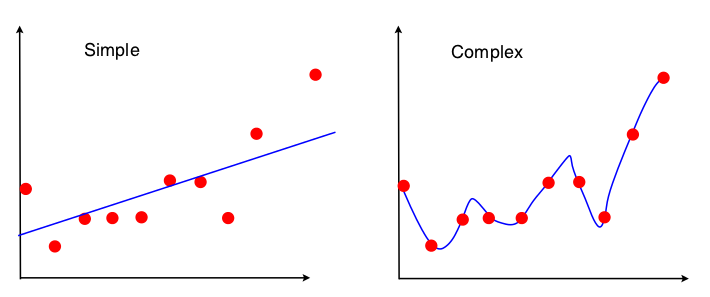
If you're interested in further reading on this topic, I can recommend these 2 articles:

* [How much data is enough?](http://fastml.com/how-much-data-is-enough/)
* [How Much Training Data is Required for Machine Learning?](https://machinelearningmastery.com/much-training-data-required-machine-learning/)

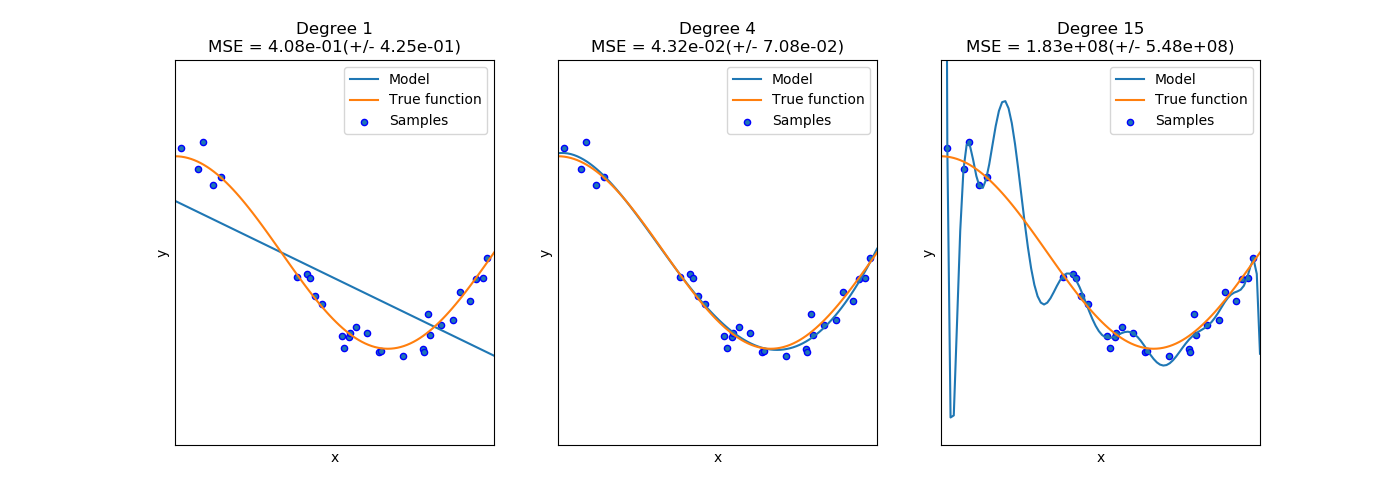
**Student correctly identifies whether the model at a max depth of 1 and a max depth of 10 suffer from either high bias or high variance, with justification using the complexity curves graph.**

You have a pretty good understanding of bias/variance tradeoff—I like how you used the visual cues in the complexity curves graph to justify your answer.

It's also nice to visualise what the high bias/variance models look like when plotted over the actual data set. I think the following picture sums this up pretty well and can be even used to teach the basics of high bias/variance to complete novices.

[](https://i.stack.imgur.com/8RlJk.png)  
[(image source)](https://stats.stackexchange.com/q/19102)

If you're interested, you can check out the [sklearn's visualisation of high bias and variance](http://scikit-learn.org/stable/auto_examples/model_selection/plot_underfitting_overfitting.html" \t "_blank).

[](http://scikit-learn.org/stable/_images/sphx_glr_plot_underfitting_overfitting_001.png)

Also, you can read more about bias-variance tradeoff [on Scott Fortmann-Roe's blog](http://scott.fortmann-roe.com/docs/BiasVariance.html).

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

Although the model with max\_depth of 3 isn't *the best* choice here (as you found out later in the project), it is still *a very good choice* that shows you understand the complexity curves correctly.

**Evaluating Model Performance**

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

Looks like you understand grid search correctly.

Since grid search is an *exhaustive search* (meaning it has to train and evaluate a whole model for each hyperparameter combination), it's computationally expensive and memory inefficient.

In those cases, you might want to use an alternative way to tune hyper parameters. I’d recommend looking at *randomized search*. You can see the [scikit-learn implementation](http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html" \t "_blank) or read [more about randomized search in this article](https://medium.com/rants-on-machine-learning/smarter-parameter-sweeps-or-why-grid-search-is-plain-stupid-c17d97a0e881).

**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 explanation of k-fold cross-validation!

Just we only perform k-fold CV on the training data and still leave some testing set for final evaluation, that’s a *really* important difference.

I like that you mentioned that the *specific* benefit of k-fold CV for grid search lies in minimizing the risk of overfitting hyperparameters onto the training data set. Many people forget about overfitting hyperparameters.

If you ever find yourself in a position where you need to revise k-fold CV or teach this technique to someone else, I really recommend [watching this elegant video on model selection](https://www.youtube.com/watch?v=hihuMBCuSlU).

**Student correctly implements the fit\_model function in code.**

Your code implementation of grid search is correct.

Please note that you should use random\_state parameter as much as you can. This will ensure the training results will be consistent among many runs of your program.

regressor = DecisionTreeRegressor(random\_state=9) # where 9 can be any number

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

Correct!

If you run this multiple times, you might see getting a different best max\_depth. To avoid this, you should set random\_state for all functions that support this argument to ensure consistent reproducible results (here the functions are train\_test\_split, ShuffleSplit and 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.**

Excellent discussion!

I would also recommend plotting the clients over the data set to *visualise* how their homes compare to the market in the individual features. You can do this easily by running the following snippet in the Question 10 section of your notebook.

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])

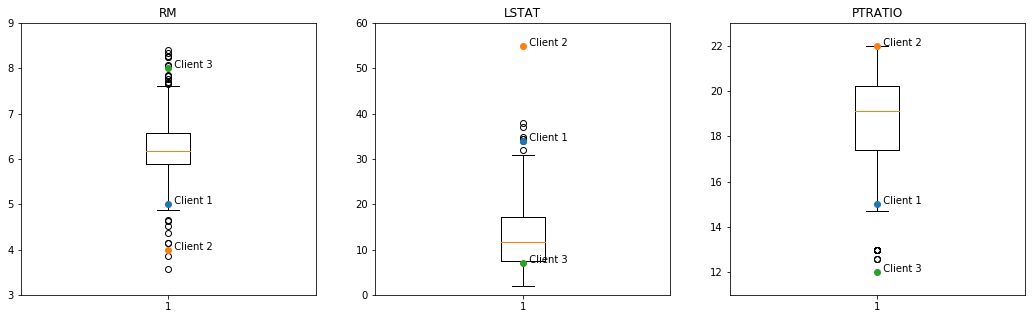
plt.title(col)

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]))

The plot will look something like this. Notice how these positions relate to your intuition about the feature correlations in Question 1 and compare them to your model's predictions.

[](https://udacity-reviews-uploads.s3.us-west-2.amazonaws.com/_attachments/173742/1521938578/clients.png)

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

You provided some great points on why this model shouldn't be used in real-world setting—and I couldn't agree with you more. Great job!

### Requires Changes

#### 5 SPECIFICATIONS REQUIRE CHANGES

This is a really good first attempt. Very well done. Please try to provide a bit more elaborate and try to write well-explained answers as we reviewers would like to know whether you have understood the concept thoroughly. It will also make for a really good project when you showcase this in your portfolio.

### Data Exploration

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

Good job implementing all the statistics in Numpy

PRO TIP:  
Try to format the numbers correctly using Python Formatting. Consider using python format function to restrict decimal values to less than 2 places as this is a monetary value.  
Use print("{:.2f}".format(minimum\_price));

Check this link for more info: [Python String Format Cookbook](https://mkaz.tech/code/python-string-format-cookbook/)

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

You are right about the number of rooms, but I disagree with you on the other two.

Although LSTAT and PTRATIO are not features of the house. They are the features of the neighbourhood/schools in the neighbourhood. Usually in wealthy neighbourhoods the price of houses tend to go up. Check out Beverly hills in California (Many Actors/Actresses and other celebrities live here) or maybe a coastal area in any city. Most of the houses in these areas will be much higher than the houses.

Also, PTRATIO is quite important as many parents do select the houses based on if the location has a good school. These do matter.

In any case, Try plotting the data points of each of the features, you will be able to see the trend. Try the code given below.

import matplotlib.pyplot as plt

import seaborn as sns

plt.figure(figsize=(20, 5))

for i, col in enumerate(features.columns):

plt.subplot(1, 3, i+1)

sns.regplot(data[col],prices, marker='+',scatter\_kws={"color": "black"}, line\_kws={"color": "red"})

plt.title(col + ' Pearson correlation ' + str(np.round(np.corrcoef(data[col], prices)[1][0], 3)))

plt.xlabel(col)

plt.ylabel('prices')

plt.show()

### 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.**

Very well done.

**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.**

Brilliant Answer.

### 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.**

we should never use the testing data for training, instead of that we can split our data into one more section called as cross validation set which we can use to train our data and select the best model graph.

I am afraid that you have answered something entirely different for this section. Here we expect you to pick a graph and explain the graph with respect to training and testing point and the other questions underneath it.

Please do rewrite it.

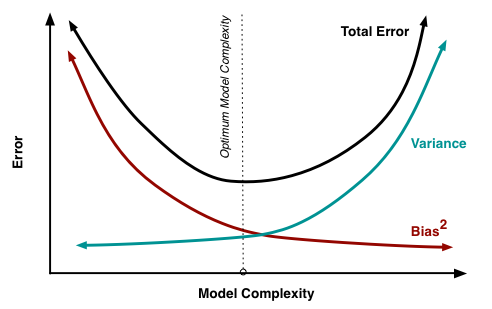
**Student correctly identifies whether the model at a max depth of 1 and a max depth of 10 suffer from either high bias or high variance, with justification using the complexity curves graph.**

When the model is trained with a maximum depth of 1 then it will work fine but the score genrated for both testing and validating curve at this point is less than 0.5 so the prediction for the cost will be poor.

I am afraid a depth of 1 won't work fine. Here since the testing and training curves are close by as well as of low accuracy, we can see that it is underfitting the data. The model is not complex enough to learn from the data.

Also, please explain whether it is bias or variance for both depths of 1 and 10.

Extra Reading :  
Just some additional reading for how to balance between Bias and Variance in your spare time. [Bias Variance Balancing](https://towardsdatascience.com/balancing-bias-and-variance-to-control-errors-in-machine-learning-16ced95724db)

Also, read this {Scott Fortman - Bias Variance}[[http://scott.fortmann-roe.com/docs/BiasVariance.html](http://scott.fortmann-roe.com/docs/BiasVariance.html" \t "_blank)]  
[](https://udacity-reviews-uploads.s3.us-west-2.amazonaws.com/_attachments/47950/1526048897/biasvariance.png)

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

3 would be a better choice than 2 I believe.

### Evaluating Model Performance

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

Good answer. Very well done. Although there are some grammatical errors / extra words in the answer. Please do correct them.

**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.**

This is a good answer. But I need a bit more explanation from your part in the K Fold section. i.e how many times do you run the training? What do you do at the end of training? What model do you select?

Also although you are right about the fact that k-fold will give us more data to train on as we can use the whole data and not just training set an keep a separate test set, That is not the main advantage when it comes to how K-Fold helps GridSearchCV

Think about how GridSearchCV works, you have a cross-validation set and then you train the training set multiple times with different parameters and then you try to validate the model trained with the selected hyperparameters on the same test set all the times. What do you think will happen in this scenario? For every hyper-parameters combination that we select we are using the same validation set. This might cause overfitting right? Now think about how K-Fold will solve this problem. I hope you get the hint.

**Student correctly implements the fit\_model function in code.**

Good implementation here.

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

Yup. 2 is a bit too small, better go with 5 as that is what the computer says.

**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.**

That is a right. In the last sentence, i think you have missed a few words, I did not understand it properly. WHatdo you mean by " "neighborhood with lowest poverty level" and "Students-teachers ratio of nearby school" is put". I think you have missed something, Please do complete the answer.

Yes, the prices seems reasonable with the values for the respective features as no of rooms is putting heavy weightage for predicting SP of house and "neighborhood with lowest poverty level" and "Students-teachers ratio of nearby school" is put

just make sure you also give some brief ideas in terms of how the predictions compare to the descriptive stats of the housing prices that you computed in the Data Exploration section.  
How do they compare to the mean / median / min / max of the MEDV variable.

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

That is right. This model is not the best. But it is a good starting point. There are many features to be added as per today's standards. Also when we did the sensitivity test in the code snippet above the last question. We saw there is a lot of variation. All that makes for a subpar model to be used in real life. Real estate agents will not be happy.

it is totally irrelevant if we try to predict the cost of house now with the data that was collected from 1978, as due to inflation the cost of the house varries alot with each feature. Assets and property always face inflation with time and it can be cause by the increase of demand also.

Do note that the data has been modified to factor in inflation. If you read the getting started section, you can see that.

The feature 'MEDV' has been multiplicatively scaled to account for 35 years of market inflation.

Just correct that part anyway.