# Project 2 Evaluation Rubric

Instructions:

1. Rename this file with your city and name last name first i.e. Project 1 Evaluation Tool - LA - Cook\_Joshua.docx
2. For each of the selected questions, compare your code with the given solution.
3. It's OK if the path you took to get there was different, but the result should be the same.
4. Evaluate yourself based on:
   1. your code
   2. conceptual understanding
   3. what you could've improved
5. **As this project is MUCH more open-ended, your results may differ significantly from ours. Evaluate yourself on the intent of the question and response, and your ability to execute.**

Keep in mind:

* Self-evaluation is important, and learning how to identify issues yourself is critical.
* The goal here is for both you AND an instructor to evaluate your work. You should be thinking critically about your answers and compare your self-evaluation with instructor feedback.
* Your comments on your work are in an integral part of how we will be providing feedback. The more detailed that you are with your discussion of your work, the more feedback we will be able to give you to help you grow.
* **Our answers should not be thought of as gospel**. According to the Zen of Python, “There should be one-- and preferably only one --obvious way to do it. Although that way may not be obvious at first unless you're Dutch.” Keep in mind that none of your instructors are Dutch and while we do our best to adhere to [PEP 8](https://www.python.org/dev/peps/pep-0008/), we may do things that are less than Pythonic.
* You should use the [Skitch](https://evernote.com/products/skitch) tool or some other similar tool to take screenshots of your work that you paste into this document as we have done with our answers.

Submit this doc on Sunday by 11:59 PM PST.

Project 2 Evluation Rubric

**3b. Fill missing data.**

**4a. Benchmarking**

**4c. Cross-validated models**

**4D. Model Selection**

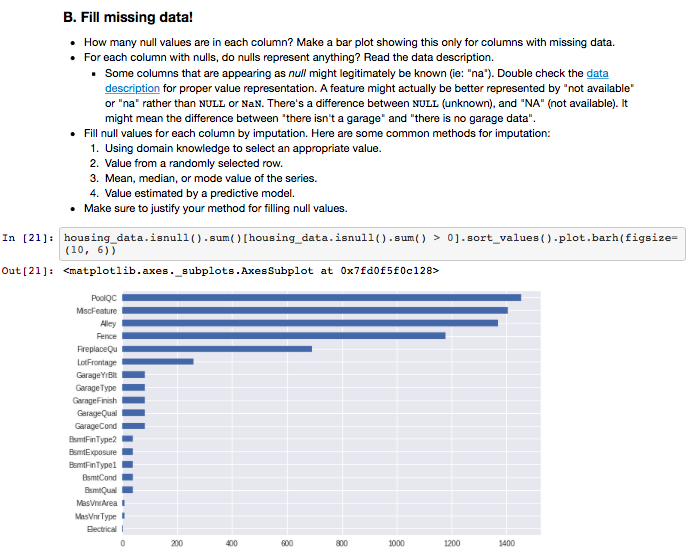
**High-Level Evaluation**

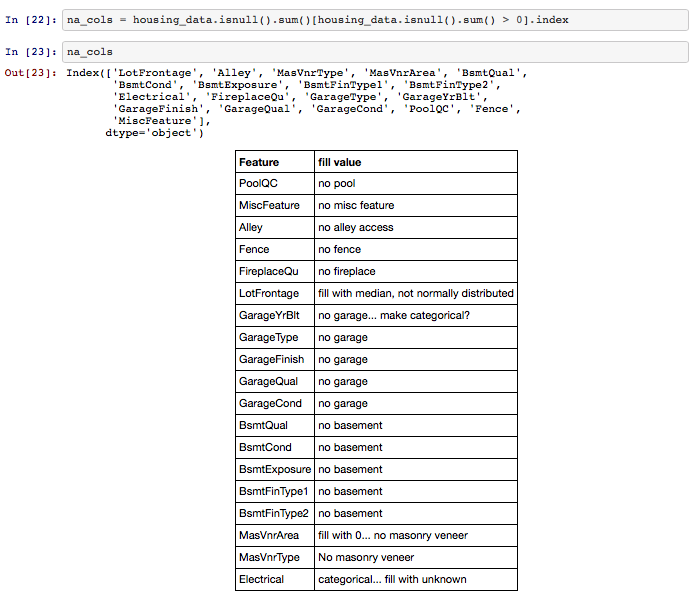
# 

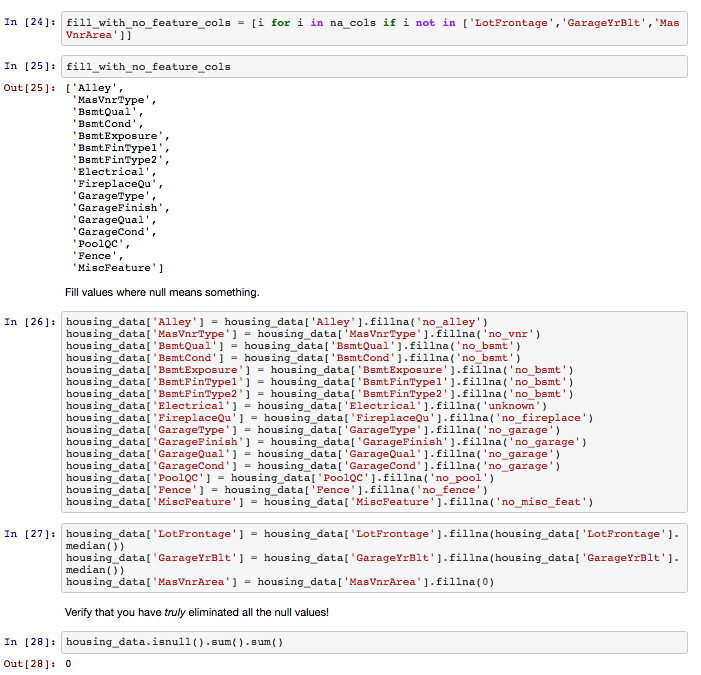
# 

# **3b. Fill missing data**

***Our Answer***

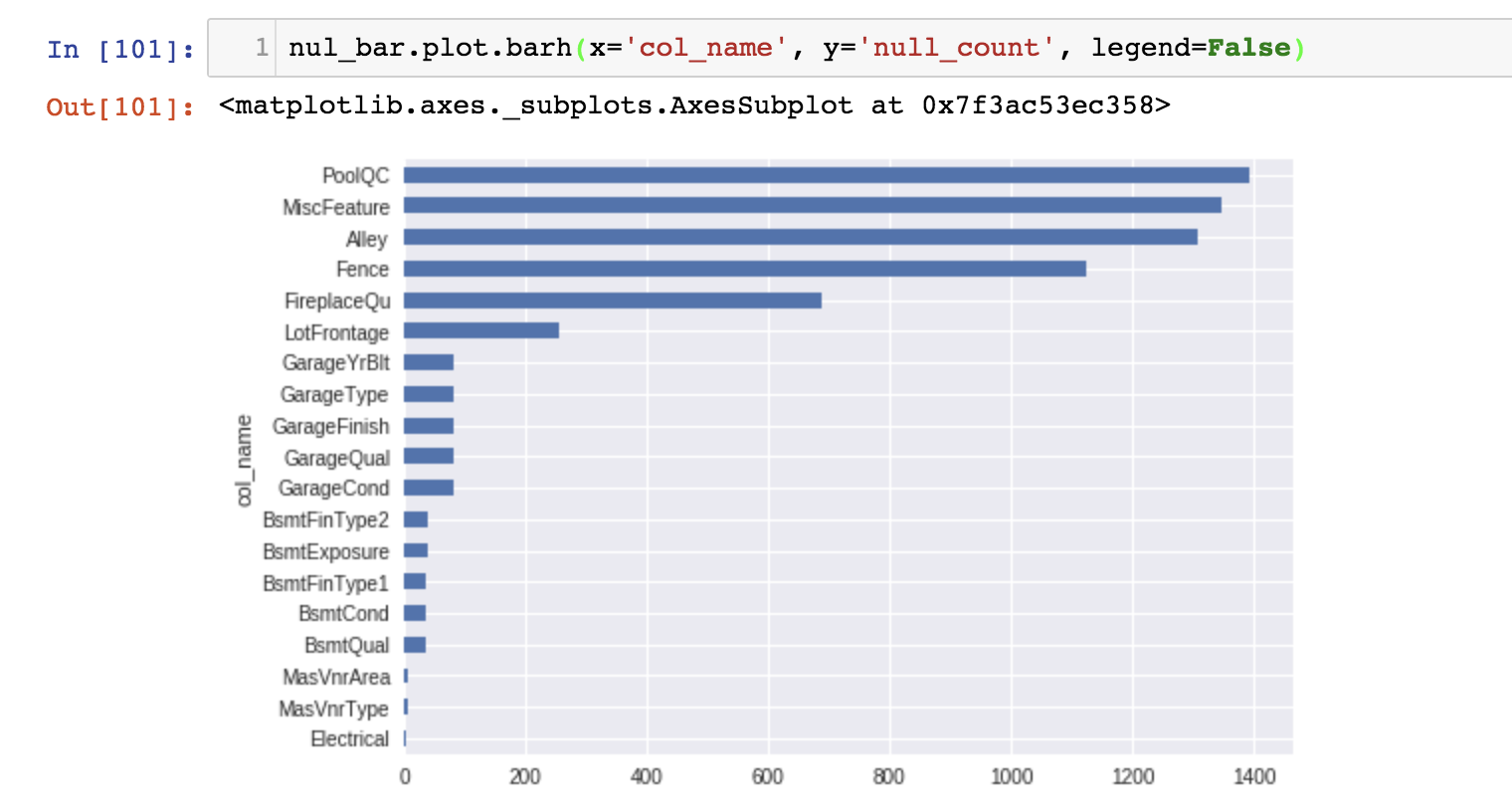






***Your*** ***Response***

******

**

**

******

***Your Rating for your response: (3/2/1)***

( 3 - I killed it, 2 - I did pretty good, 1 - I struggled)

**RATING: 2**

***Why you selected this rating***

I generated the same null-count bar graph as the solution but used a for loop instead of a list comprehension (going to try and use more list/dict comprehensions on future projects, I just like the traditional for loop structure for readability.)

After reading through the description document, I made a list of columns that use none/null as a categorical descriptor so that I could compare it against all columns that have nulls and get the list of columns that require cleaning.

Differences from the solution code:

1. I used the mean to fill nulls for ‘MasVnrArea’. I now realize that I should have put ‘none’ for that since the ‘MasVnrType’ column had a none value, and the rows missing the ‘MasVnrArea’ were ‘none’ in the ‘MasVnrType’ column.
2. *Something that I think I did better than the solution:* I replaced the one missing value in the ‘Electrical’ column with the value ‘SBrk’. I looked at the values for all of the houses built in the same year as the the home missing the ‘Electrical’ feature and 100% of them had the same value.
3. I used .info() to look at the counts for each column to make sure that I had successfully filled my nulls. I like how the solution code used .isnull().sum(), will remember that trick for the future.
4. I used a different format than the solution code to fill null values. The solution used .fillna(), which I hadn’t seen but seems very useful! I just reassigned the column values using .mask().

***Our Rating for your response: (3/2/1)***

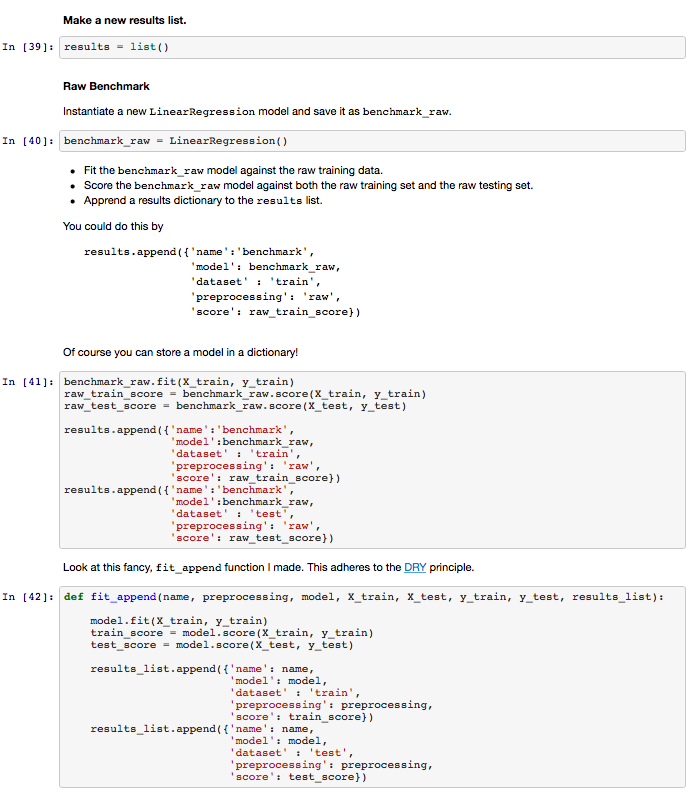
( 3 - You killed it, 2 - You did pretty good, 1 - There are some things you can improve)

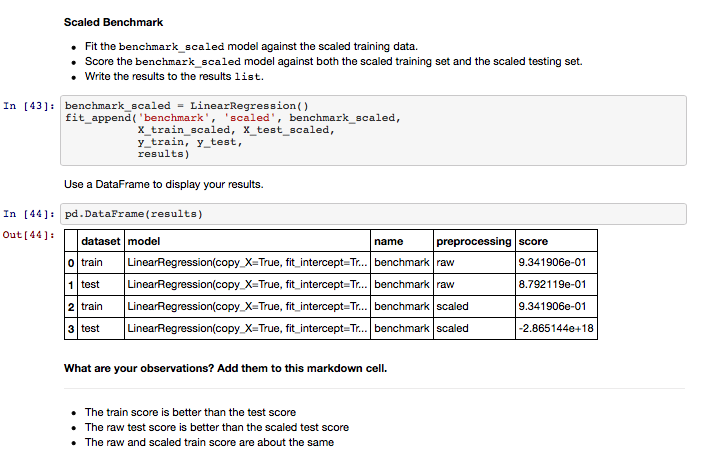
***Why we selected this rating***

## 

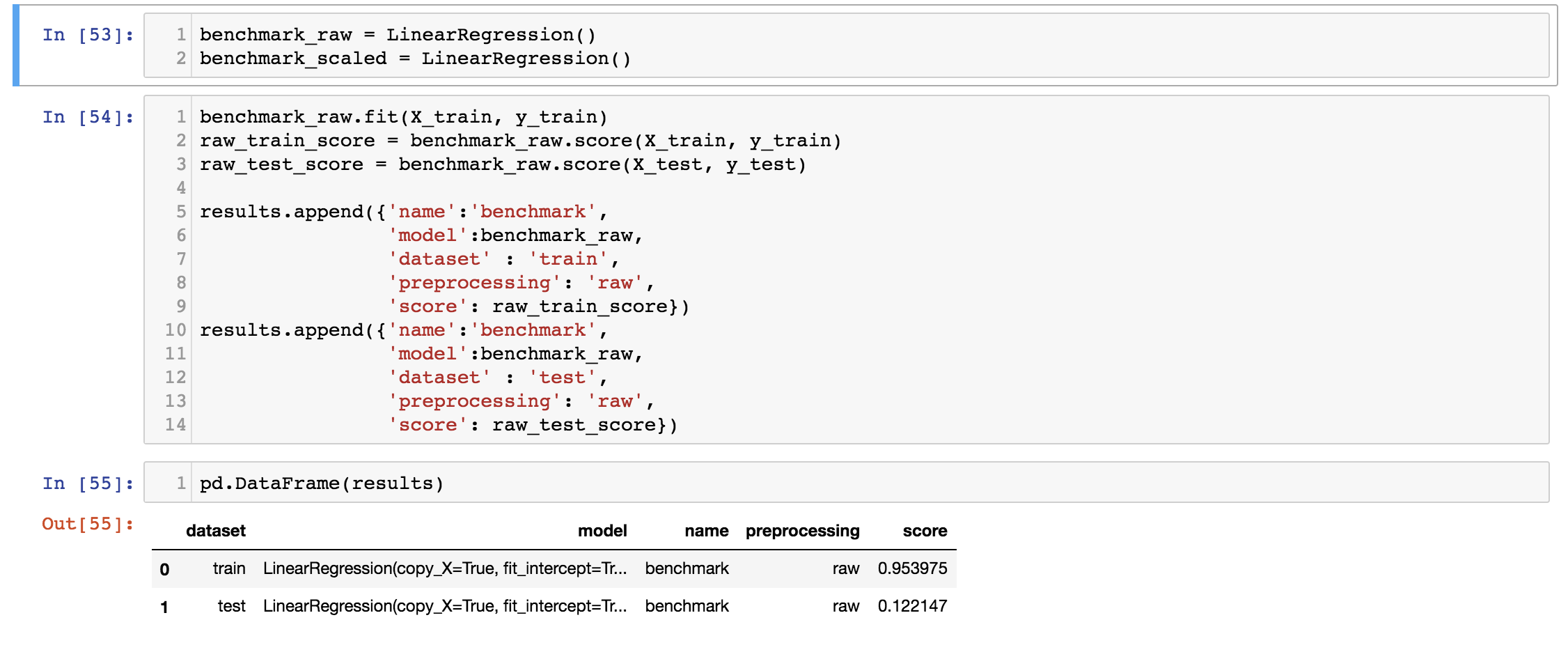
# **4a. Benchmarking**

***Our Answer***





***Your*** ***Response***

**

**

***Your Rating for your response: (3/2/1)***

( 3 - I killed it, 2 - I did pretty good, 1 - I struggled)

**RATING: 2**

***Why you selected this rating***

I used identical code to the solution (except for the fancy fit\_append function), but I have a terrible score for the raw test data. I’m not 100% able to deduce why that might be since I can’t see the rest of the solution code, but I have a couple of ideas and it might be all of them combined to make a big difference.

1. I would like to see how the solution code removed outliers. Maybe more were removed and so the solution was more accurate.
2. Since we changed nulls a bit differently, maybe that would cause a difference when we used get\_dummies.
3. My train\_test\_split has a test\_size=0.3 and a random\_state=42. Maybe the solution code did a .5 split and a different seed.

***Our Rating for your response: (3/2/1)***

( 3 - You killed it, 2 - You did pretty good, 1 - There are some things you can improve)

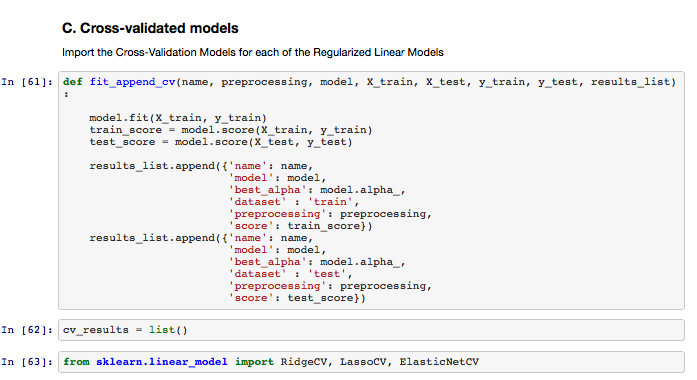
***Why we selected this rating***

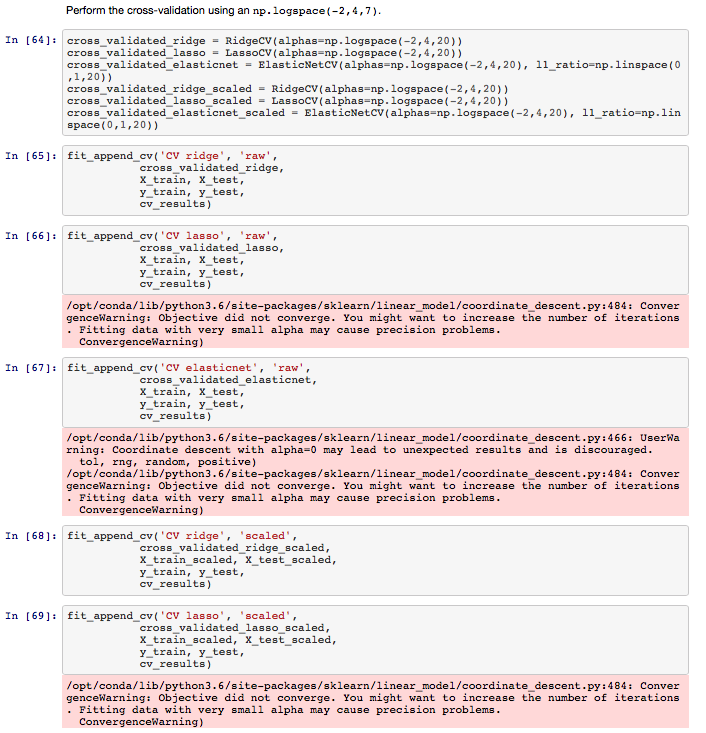
## 

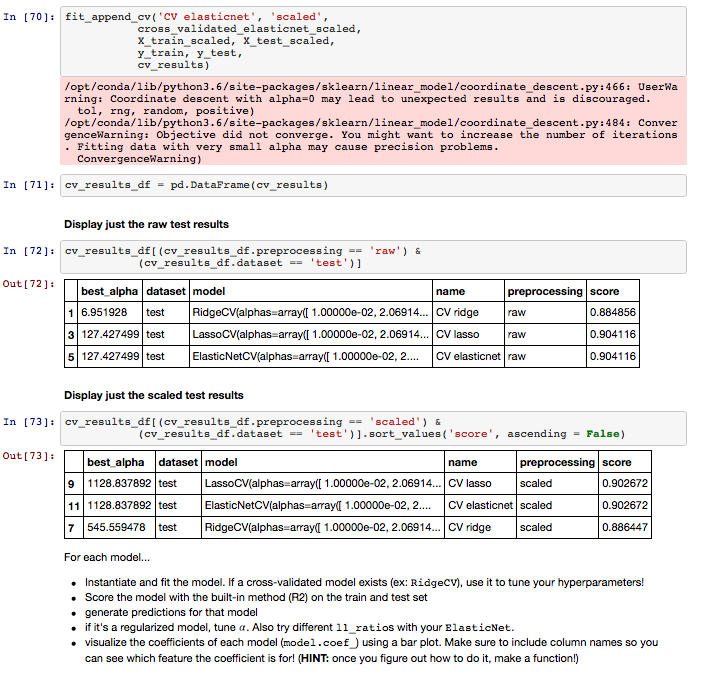
## 

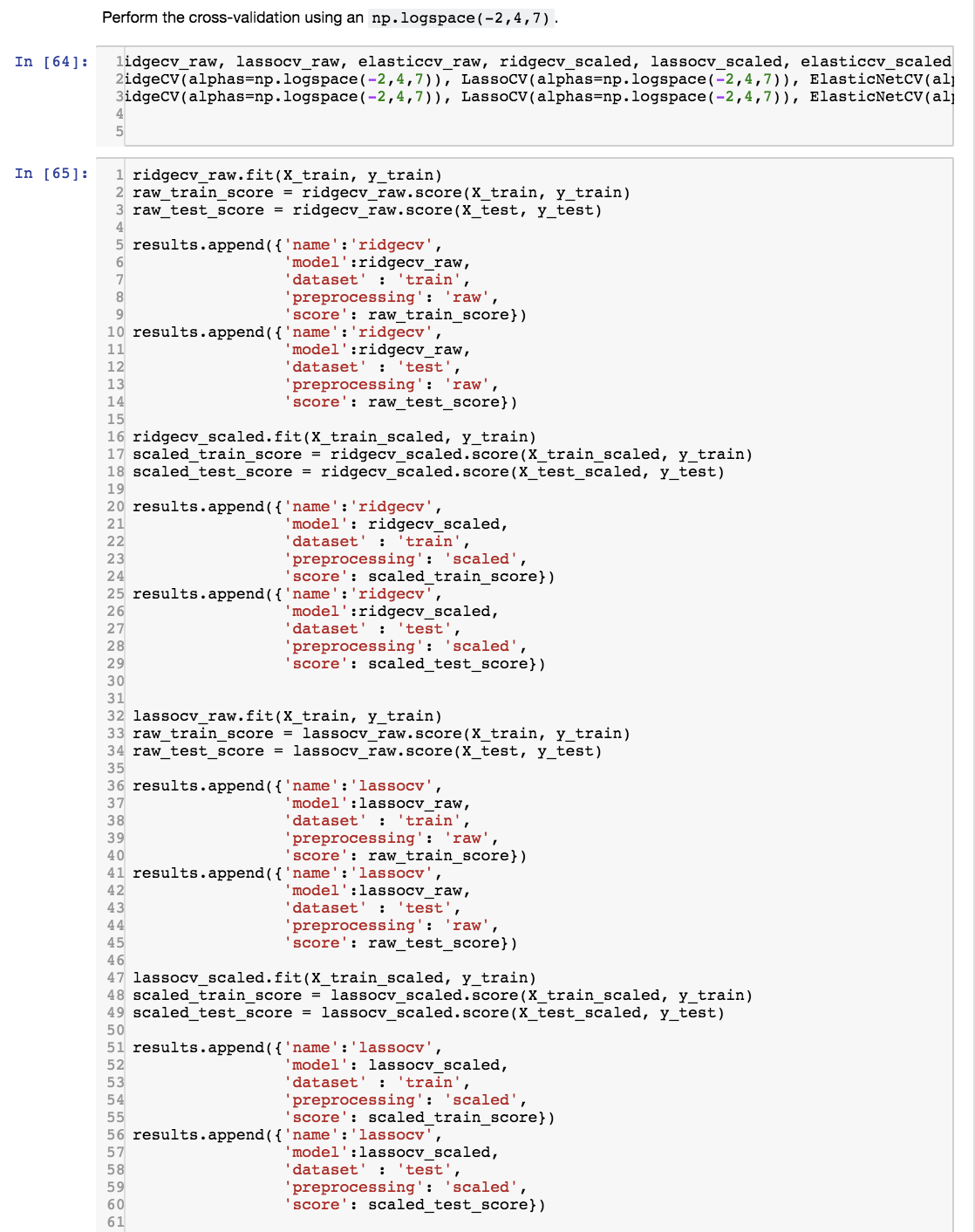
# **4c. Cross-validated models**

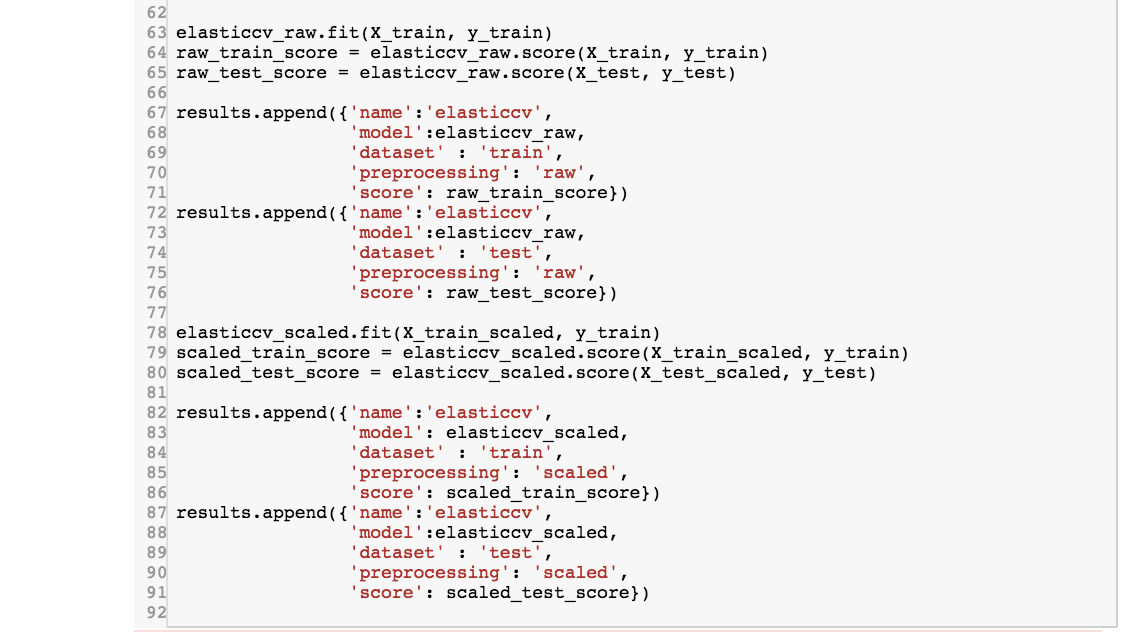
***Our Answer***

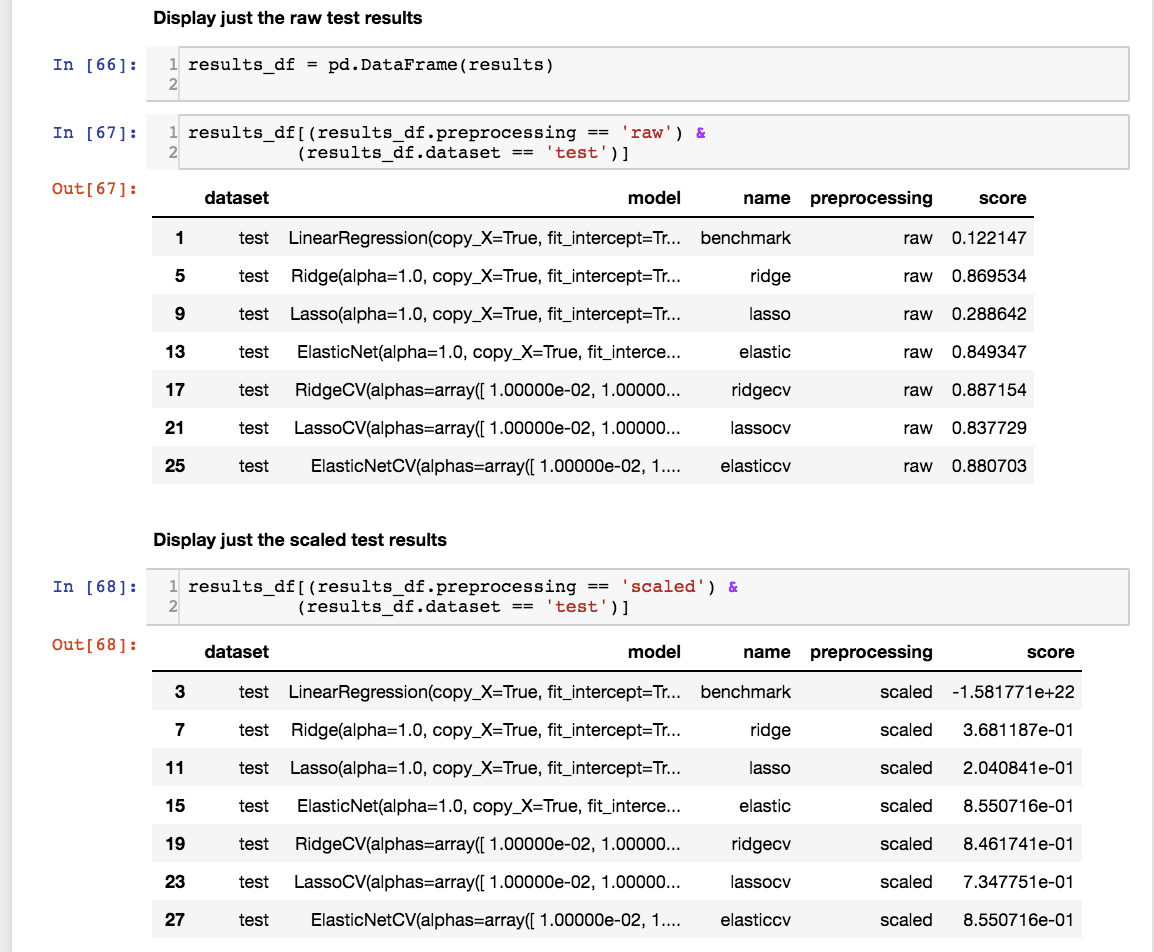






***Your*** ***Response***

**

**

***Your Rating for your response: (3/2/1)***

( 3 - I killed it, 2 - I did pretty good, 1 - I struggled)

***Why you selected this rating***

**RATING: 2**

My response is very similar to the question above. My code looks identical except I didn’t use the fit\_append function and my test scores are lower. I think I did well on this particular question, but I think the issue is coming from a difference in how we set up the train\_test\_split, filled nulls, and removed outliers earlier on in the project.

Also, the solution code didn’t follow the directions exactly. The instructions provided said to run the cross validated models using np.logspace(-2,4,7) and the solution code used np.logspace(-2,4,20), and also set an l1\_ratio. I went and reran the code, and it does change the output with those updates.

Would love to hear in the instructor feedback section about your theories as to why my output is different. It might be helpful if a full solution code is sent out when the students do the self evaluations so that we can troubleshoot our differences more effectively.

***Our Rating for your response: (3/2/1)***

( 3 - You killed it, 2 - You did pretty good, 1 - There are some things you can improve)

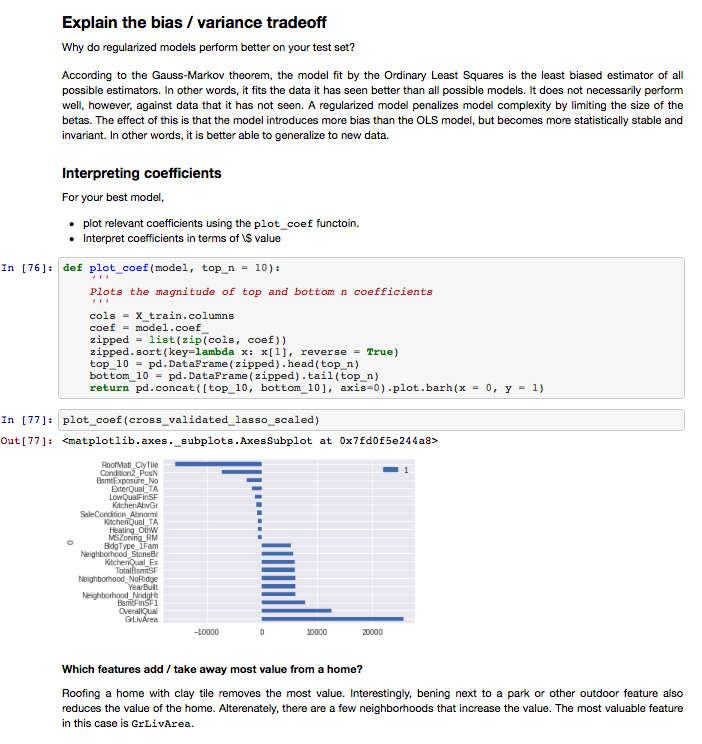
***Why we selected this rating***

## 

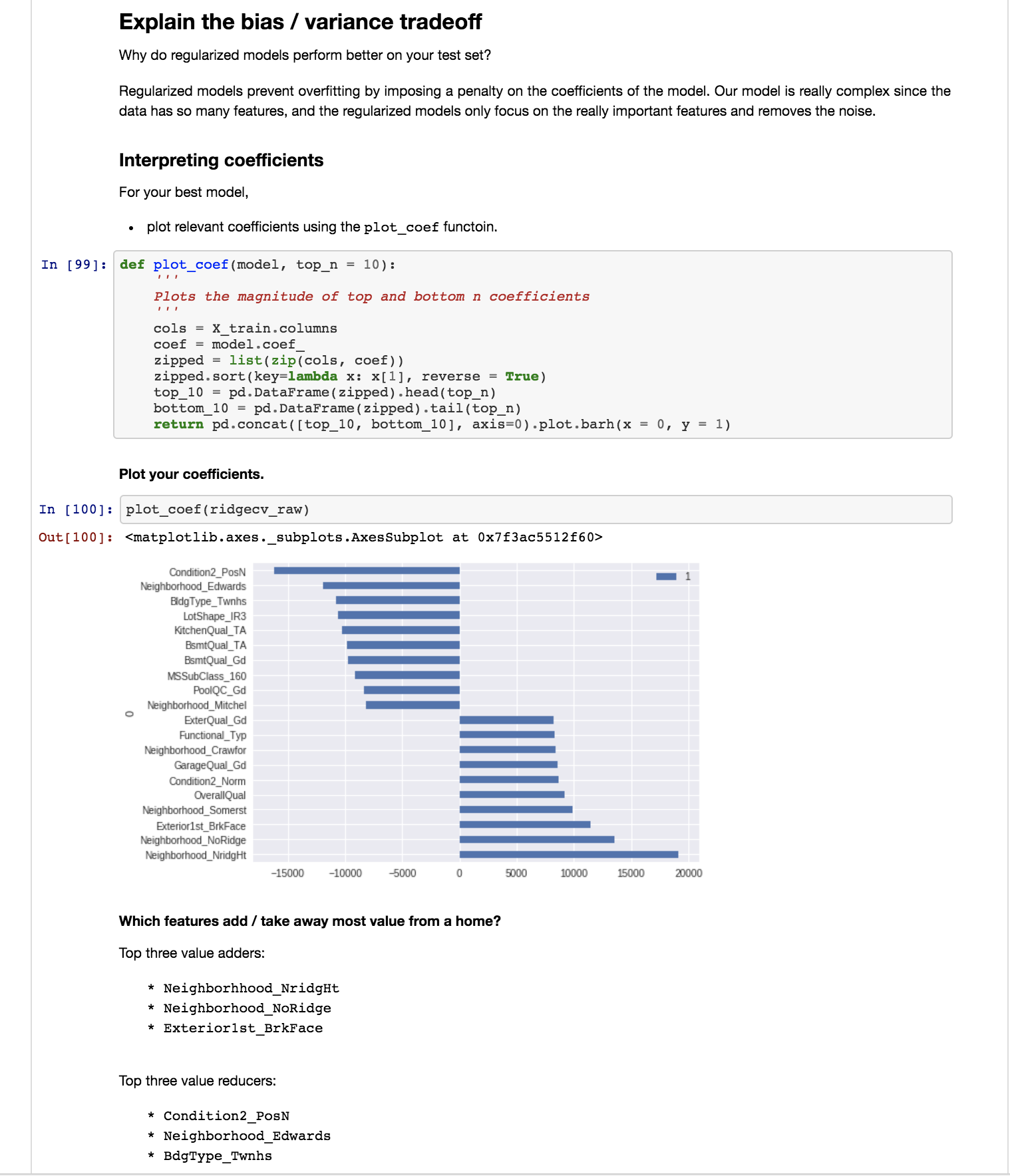
## 

# **Model Selection: Bias/Variance Tradeoff and Plotting Coefficients**

***Our Answer***



***Your*** ***Response***

**

***Your Rating for your response: (3/2/1)***

( 3 - I killed it, 2 - I did pretty good, 1 - I struggled)  
**RATING 2**

***Why you selected this rating***

I was not as verbose as I could have been in the document when I answered the bias/variance trade off question, though my answer is not wrong. Let me go into more detail now: Overfitting is the result of variance causing the model to fit to random error / noise instead of generalizing enough to predict the outcome well. We had hundreds of parameters after using get\_dummies, and multicollinearity is most likely present in the data. For example, in my results I saw that ‘Condition2\_PosN’ was a price reducer. This condition is that the home is “near positive off-site feature--park, greenbelt, etc.” which means they are probably not in a densely populated urban area that has higher prices due to housing scarcity. Another example, is that maybe the ‘Neighborhood\_NridgHt’ has a lot of homes made of brick, aka ‘Exterior1st\_BrkFace’, which is why they both showed as important price predictors.

My output, like my answer to the previous questions, is different. I am unable to deduce why without the rest of the solution’s code.

***Our Rating for your response: (3/2/1)***

( 3 - You killed it, 2 - You did pretty good, 1 - There are some things you can improve)

***Why we selected this rating***

# **High-Level Evaluation**

***One thing I did very well on this project was …***

* *Last project, my progress was slow because I was struggling with basic python syntax. This project went much smoother for me in terms of how many errors I was getting, and I believe that is a direct result of the time I’m putting in outside of class studying my python.*

***One thing that I’m not clear on and would like to know more about is …***

* *Interpreting my model scores. I feel like I missed the mark a bit with my models in this project and I’m not sure why.*

***If I was starting this project over from the beginning knowing what I know now, one thing I would do differently is …***

* I would have played around with the train\_test\_split more to see if I could get better outputs.