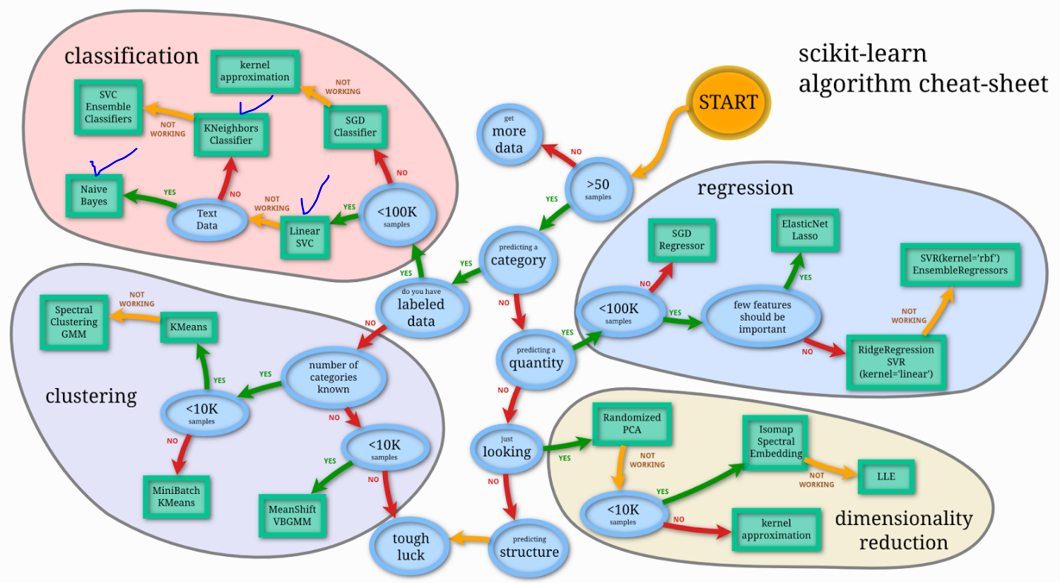
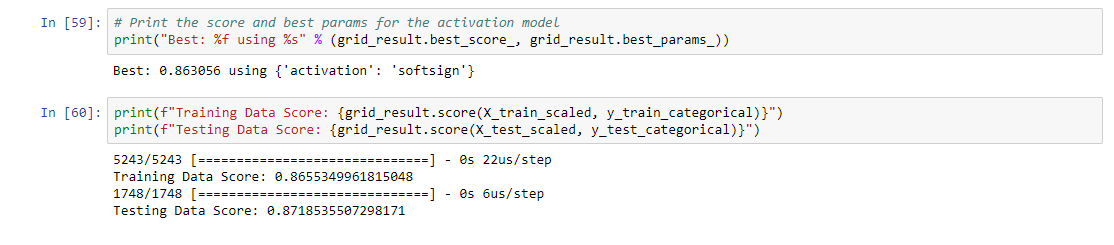
**Multiclassification Exoplanet Challenge Summary**

For this machine learning project, I built a automated feature selection model using the XGBoost module. This is a gradient booster that you can use to extract the most important features of a data set based off of their caculated weight in relation to the target after running through the model. This uses a series of weak decision trees (like random forest) to make strong assumptions and essentially picks the most resilient features that won the micro competitions the most in the underlying algorythym. This about this like a final for bracket… Whoever winds up in the finals has a higher weight since they won the most game. This feature selection increased the probability of quick and easy training for my prediction models.

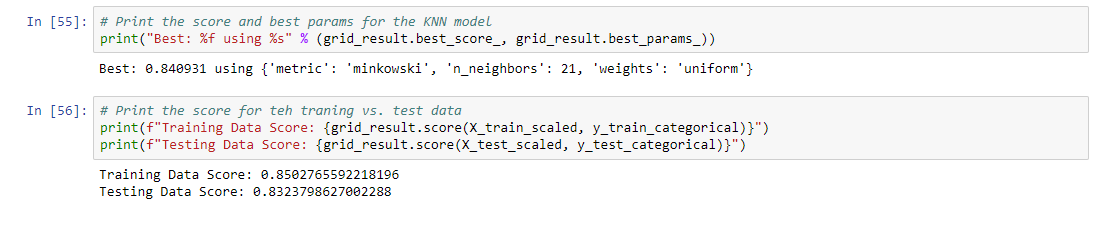
Furthermore, I also tested for multicollinearity between my features to and got rid of one feature in the highest correlated feature in feature pair. I chose to go with four models. Per the SciKitLear cheat sheet below, I used the K Nearest Neighbors, Naïve Bayes, and Linear SVC models in accordance to the characteristics of the data set. I also create a custom Keras Classifier Neural Network to perform estimations. The below are my results and the guide that I referenced to choose my models (marked by the blue checks):



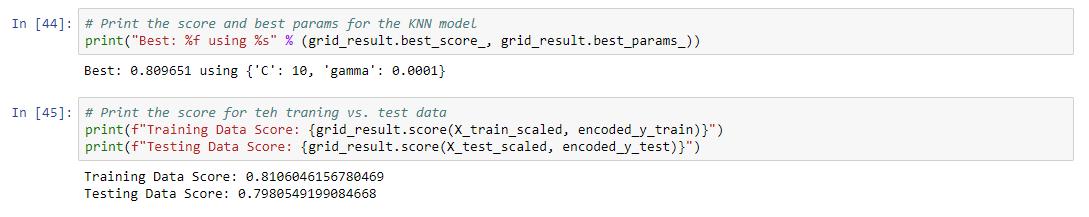
Keras Deep Neural Network:



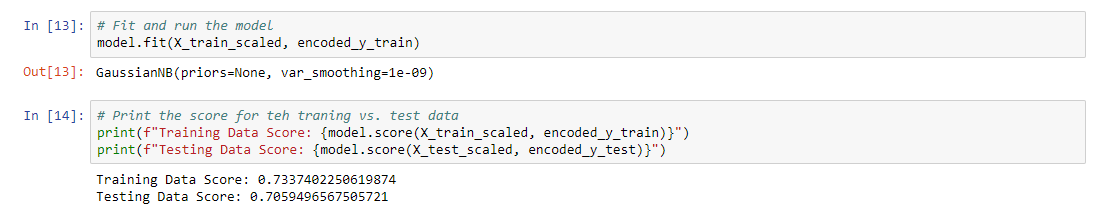
K Nearest Neighbors:



Linear SVC:



Naïve Bayes:



\*\*\*Note for the Naieve Bayes model, after some research, apparently, you can tune hyperparameters for this model since it is so simple. So I just left it as is…

As you can see from the above, the Keras NN got the heist score on the training data and beat the 85% threshold from the project guidelines by scoring circa 87% on the testing data. A close second was the K Nearest Neighbors model with a 83% score for the testing data. The Linear SVC and Nieve Bayes models performed the worst on the testing data with a score of circa 79% and 70% accuracy respectively. So, clearly, the winner in this lineup is the Keras Deep NN I created and thus this is what will be submitted as my top model.