Anthony LaRosa

11/18/20

Professor Werner

DSC550 – Case Study Final

In Search of: Top Rated Ramen

**Background Information**

College students, individuals on a budget, and asian noodle lovers have all encountered ramen at one point in their lives. Many American’s think that ramen is just ramen, that there isn’t different types of brands or have knowledge that the ramen may even come from European and African countries.

Now, I can’t say I’m a ramen fan or even eat it that much, but I do certainly enjoy it occasionally after a night out at the bar.

My first experience with non-traditional U.S. ramen is a very mainstream company from South Korea, Nong Shim. It was quite good and spicy.

I have sourced a dataset of over 2,500 ramen reviews of noodles and flavors produced all over the world.

Now I’m on the quest to find the source of top rated ramen in DSC550.

**Hypothesis**

Top rated ramen, rated 4.0 or higher out of 5.0, can be predicted using country of origin, brand, style, and variety.

**Variables**

Review # - integer variable which is the ID number of the review as it occurred

Brand – String variable of the manufacturer

Variety – String variable of the type of Ramen

Style – String variable of the type of packaging it is delivered in. IE: Cup, pack, bowl, tray etc

Country – String variable of the country of original

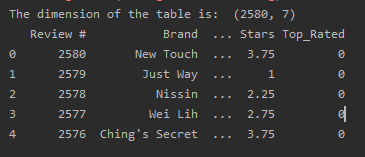
Stars – float variable 0 to 5.0 rating

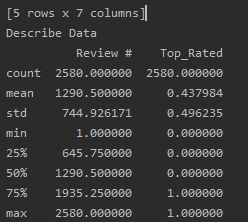
Top\_Rated - Binary variably 0 or 1 which describes whether the ramen is rated 4.0 and up or not

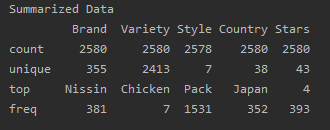
**Step-by-step instructions for Graph Analysis**

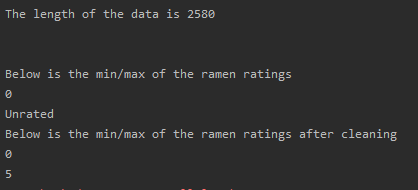
1. Create a new feature and transformed the rating variable into a binary top rated variable via a conditional statement
2. Load the data into a Pandas dataframe
3. Review the dimension of the table
4. Check out the first 5 entries in the table
5. Review the summarized variable data from the table
6. Display the length of the data
7. Display min,max, of ramen rating
8. Remove any row in which the Stars variable is equal to ‘Unrated’
9. Create bar charts of country and style to look for outliers
10. Clean data for discrepancy between USA and United States being separate labels but the same thing
11. Convert the stars variable to a float so that it can be sorted properly

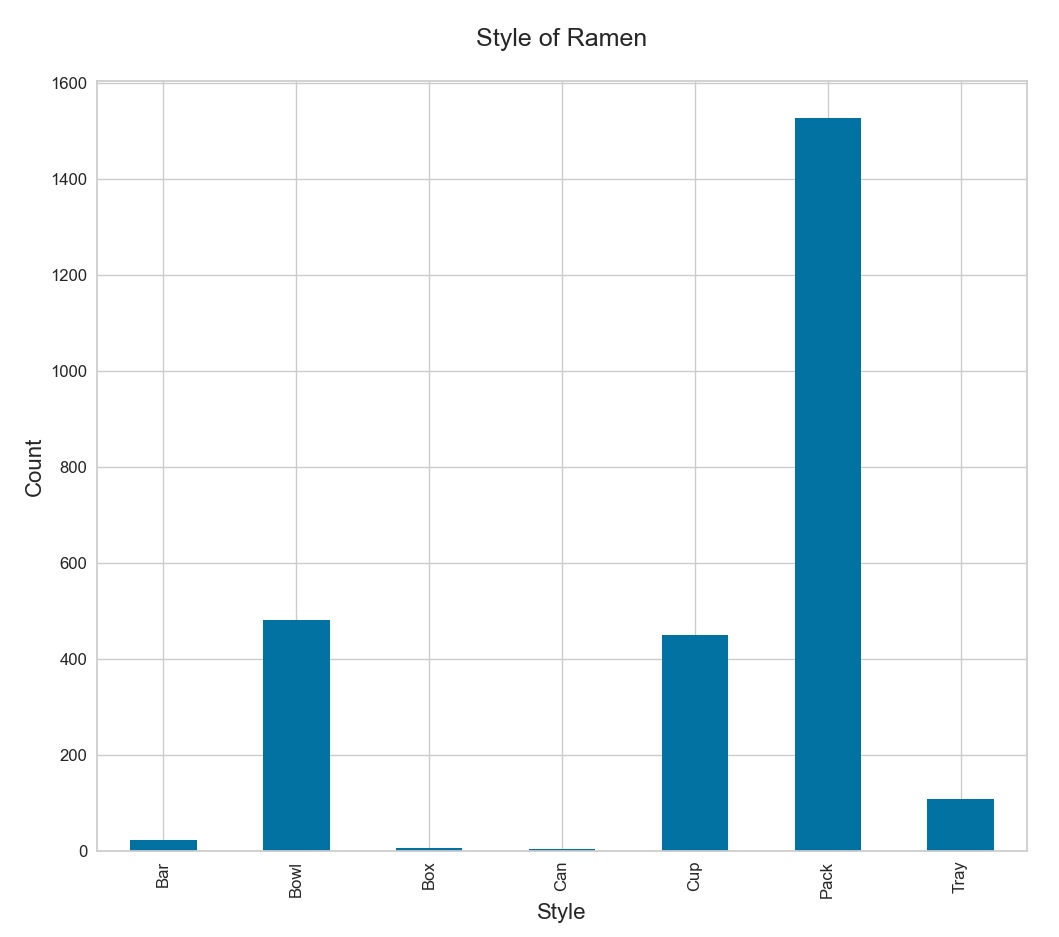
**Graphical Analysis**

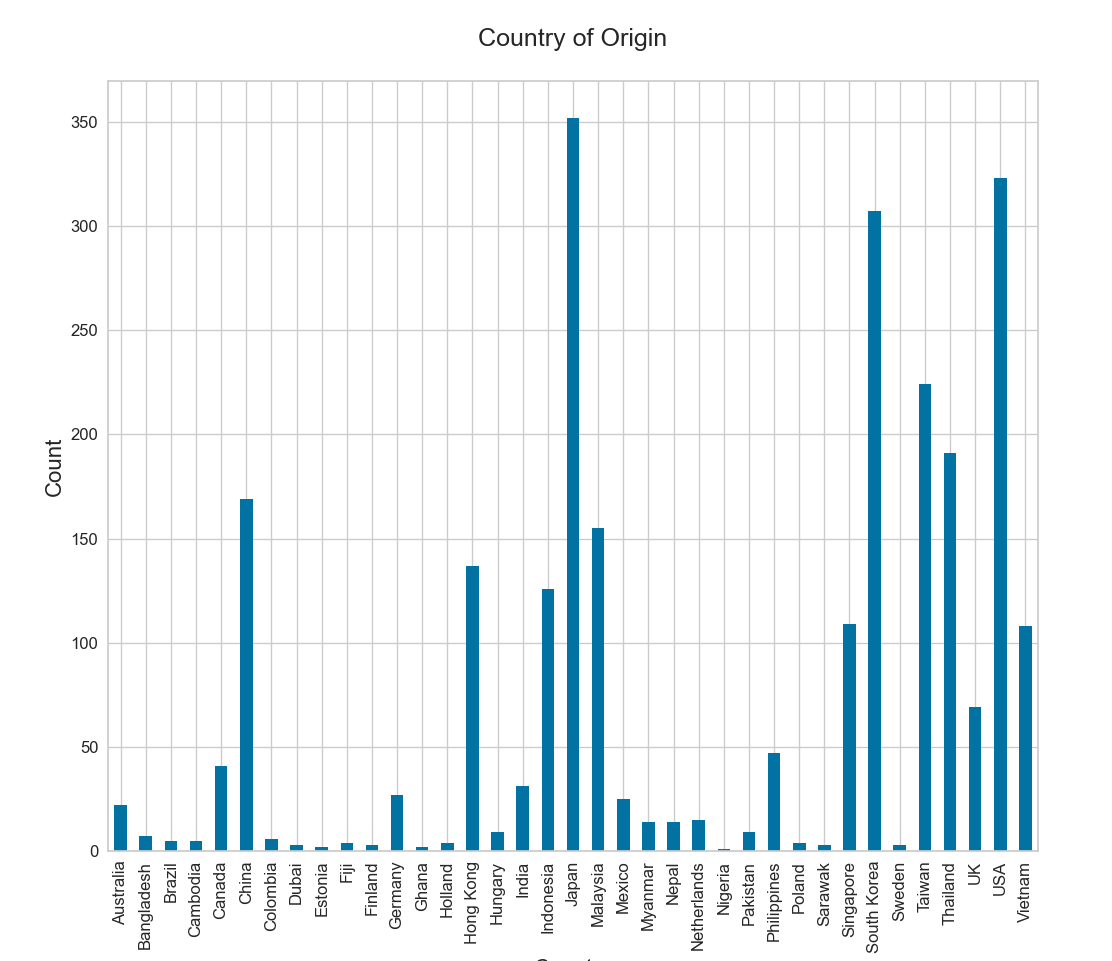












Part 2

**Step-by-step instructions for Feature Reduction and Dimensionality**

First, I want to explore the summarized data from the feature set, and remove features that would not assist in building a strong model

Print the summarized data for the variables

Review the “unique” row in the summarized data to see if there are extreme cases that would not assist in building a strong model

1. Remove the variety feature, it has 2413 unique entries
2. Remove the brand, it has 355 unique entries
3. Verify the dataframe with the features dropped
4. Generate the Average Rating by Country
5. Generate the Average Rating by Style
6. Find and remove row index 1425 which has outlier ‘bar’ and ‘5.0’
7. Verify the Stars mean by Style to confirm that Bar was dropped
8. Convert the categorical Style variable into binary to allow for quantitative analysis
9. Remove the review ID column

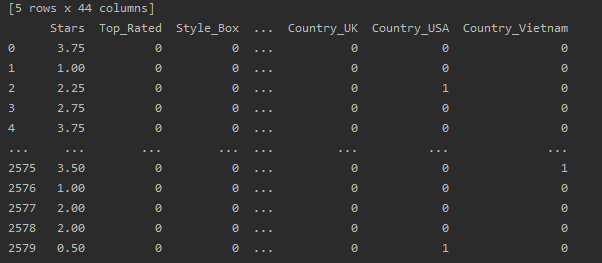
I included a PDF of the output.

Next week I have to see if I can tweak the indexing to prevent NaNs.

Part 3

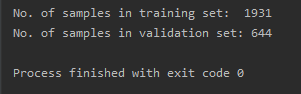
**Step-by-step instructions for Model Evaluation and Selection**

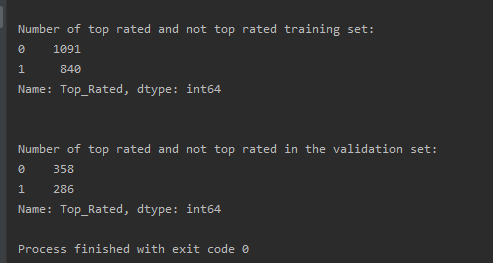
First from last week I wanted to take care of two items, removing the NaNs from my concatenation, and removing the review ID column. I removed the NaNs from the Professor’s recommendation that the dummies package is intelligent enough to not touch already numeric features.



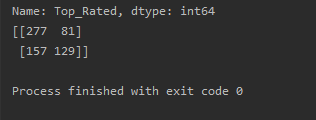
My outcome variable is either True or False, so I will be testing a logistic regression this week.

1. To split the dataset into features and target variables, first create a variable for the feature columns
2. Set X equal to the feature columns
3. Set Y equal to the target variable
4. Using the train\_test\_split() function, split the data into test and train

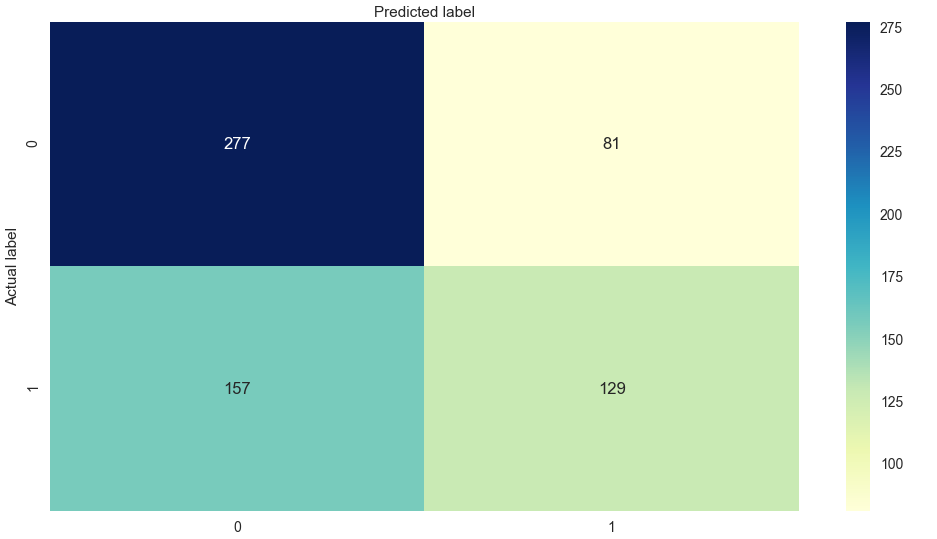




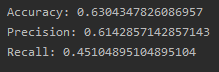
1. Instantiate the model using default parameters
2. Fit the model with data
3. Evaluate the model using a confusion matrix



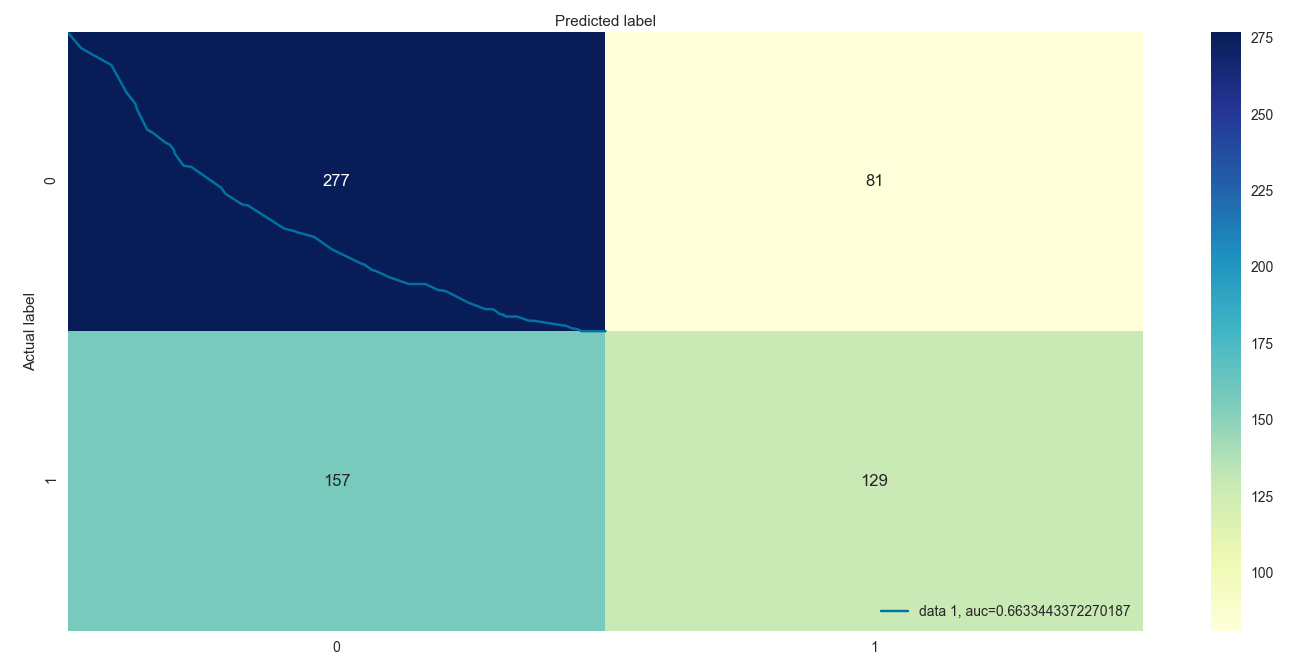
1. Heatmap the confusion matrix



1. Print the detailed results of the confusion matrix

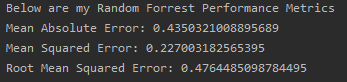


1. Create an ROC Curve

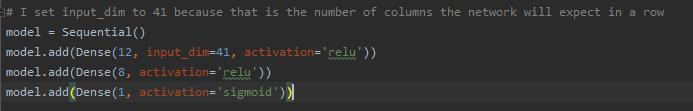


################ Start Part 4 ################

1. Start a Random Forest Regressor by testing the model with various amounts of estimators
2. Train the Random forest model on the training data
3. Use the forest predict method on the test data set
4. Calculate the errors
5. Print the absolute mean error
6. Determine the Performance metrics



1. Start a Keras neural network by first defining the model



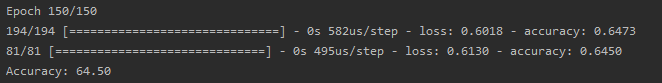
1. Compile the Keras model



1. Fit the Keras model to the dataset



1. Evaluate the keras model



**Conclusion**

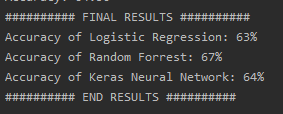
When I began this case study, I was searching datasets related to the food industry. The one that caught my eye was related to ramen. In the past, and I would assume like most individuals, ramen is just ‘ramen’, cheap late night sodium and calories when you are tight on money or a college student. When I saw over 2500 samples in the dataset from all over the world, I realized there is something more to ramen than I was aware of. Additionally, the dataset had a feature that defined a rating to the ramen and each ramen sample was unique in some way.

The idea came to me; why not define top rated ramen? Let’s say ramen >4.0 are considered top ramen, and see if we can predict what ramen will be top from the dataset.

I started out the project with the usual path of the data wrangling, preparation, and normalization.

With guidance from my professor and mentor for this project, we decided down the path of exploring different models from the most basic to advanced, to review, tune, and test the predictive outcome. I have to say that I much enjoyed this approached and it gave me great perspective on building and running models. The three models that I chose to utilize were, logistic regression, random forest, and a Keras Neural Network.

Below the final results of the project can be viewed:

****

They were all within the same ballpark with the parameters I had set, from 63%-67% accuracy. When I was building and working on this project, I had referenced a paper published on SMU to research their testing on logistic regression versus Random Forest models. They took each of them, set base criteria to measure their performance against, and then ran case studies to view the results.

The case study they found that random forest had a higher true positive rate, but also a higher false positive rate, than logistic regression. I remember running into this same thing at one point in our semester and commenting it back on someone’s discussion post. It seems to me, as a student and far from being an expert, that random forests may be good candidates in medicine because they tend to lean towards having false positives than false negatives. They did a second test and found that when the noise increases with random forests, the rate of false positives increased even further.

When it comes to the neural network, I mentioned in some of my posts that this is my first experience with a neural network, and I found that there seems to be a lot of knobs to spin for dialing it in. When I did research on Tensorflow’s website I found that there is actually a Keras library tuner called the Keras Tuner which helps you pick the optimal set of hyperparameters for the machine learning application.

That is something that I would like to look into in the future if I am dealing with Keras neural networks.

Finally, I am happy with the random forest in this case. I thought it performed rather well, and since it leans towards false positives, I rather false positives in my food reviews than false negatives. I find that often times the poor experience of one person could be a favorable experience to a different individual.

Citations:

Kirasich, Kaitlin; Smith, Trace; and Sadler, Bivin (2018) "Random Forest vs Logistic Regression: Binary Classification for Heterogeneous Datasets," SMU Data Science Review: Vol. 1 : No. 3 , Article 9.

Introduction to the Keras Tuner &nbsp;: &nbsp; TensorFlow Core. (n.d.). Retrieved November 21, 2020, from https://www.tensorflow.org/tutorials/keras/keras\_tuner