The Wine Price is Right

A Machine Learning Wine Price Analysis

Yasmine Aitouny, Daniel Carmona, Terisha Kolencherry, Pooja Nagrecha

Table of Contents

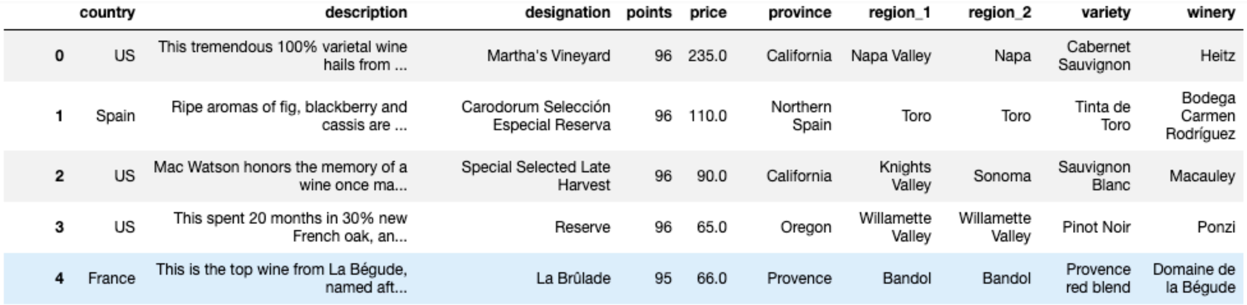
I. Introduction

This project seeks to utilize different wine review features to predict wine prices. Our team took a dataset with 150k bottles of wine, scraped from Wine Enthusiast's website and posted to Kaggle, and built a machine learning model to predict the price of a wine given its country of origin, province of the grapes, designation, winery, variety, and whether it's red or white. The group then hosted a live application that takes in user input and delivers an estimate of wine price using the finalized model. One use for this tool would be to estimate the price of a gift bottle of wine.

II. Data

Data Overview

The dataset came from Kaggle, a free website that offers a variety of products, including free datasets. The data source is Wine Enthusiast magazine, which reviews a plethora of different wines and assigns points to indicate the quality of the wine and also notes different features. This information was scraped from the website and published as a CSV file, which our team downloaded and read into a DataFrame (Figure 2.1).

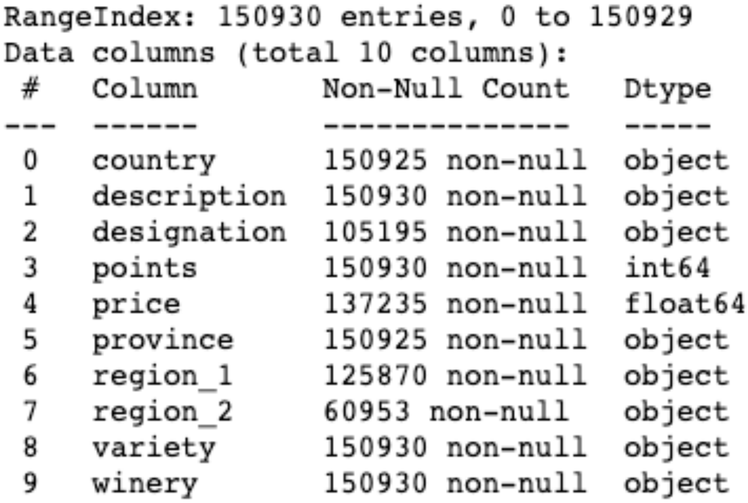


	country	description	designation	points	price	province	region_1	region_2	variety	winery
0	US	This tremendous 100% varietal wine hails from ...	Martha's Vineyard	96	235.0	California	Napa Valley	Napa	Cabernet Sauvignon	Heitz
1	Spain	Ripe aromas of fig, blackberry and cassis are ...	Carodorum Selección Especial Reserva	96	110.0	Northern Spain	Toro	Toro	Tinta de Toro	Bodega Carmen Rodríguez
2	US	Mac Watson honors the memory of a wine once ma...	Special Selected Late Harvest	96	90.0	California	Knights Valley	Sonoma	Sauvignon Blanc	Macauley
3	US	This spent 20 months in 30% new French oak, an...	Reserve	96	65.0	Oregon	Willamette Valley	Willamette Valley	Pinot Noir	Ponzi
4	France	This is the top wine from La Bégude, named aft...	La Brûlade	95	66.0	Provence	Bandol	Bandol	Provence red blend	Domaine de la Bégude

Figure 2.1 - Raw dataset read into Pandas DataFrame

Data Cleaning

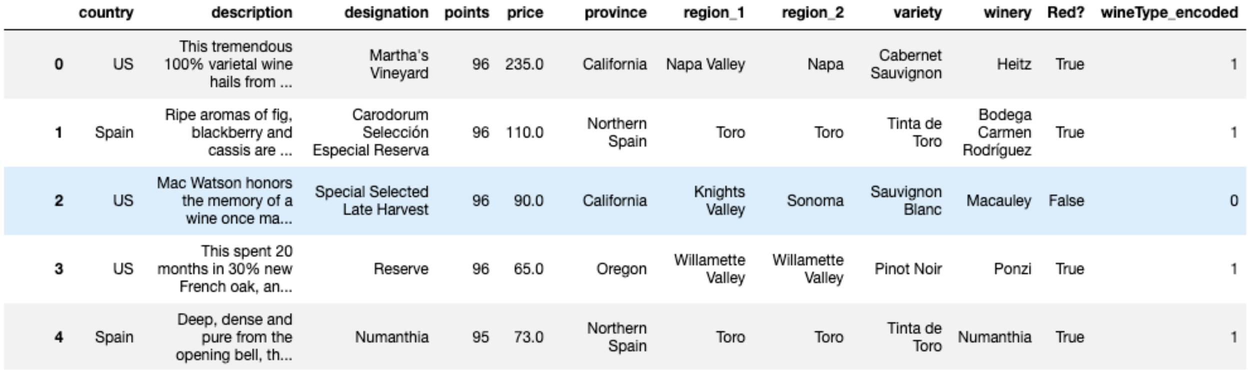
The dataset had several null values, see Figure 2.2, which we approached in a couple of different ways. For the designation column, we simply changed the NaN values to "*Unknown Wine*", since it seemed that it was common to not have a designation. For the region columns if region_1 was blank but region_2 was filled, we utilized the latter's cell value to fill in for region_1 and vice versa.



```
RangeIndex: 150930 entries, 0 to 150929
Data columns (total 10 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   country      150925 non-null  object
 1   description  150930 non-null  object
 2   designation  105195 non-null  object
 3   points       150930 non-null  int64
 4   price        137235 non-null  float64
 5   province     150925 non-null  object
 6   region_1     125870 non-null  object
 7   region_2     60953 non-null   object
 8   variety      150930 non-null  object
 9   winery       150930 non-null  object
```

Figure 2.2 - Description of the number of non-null values in DataFrame

While the data had useful features, we wanted to see if there was a difference between red and white wines, which wasn't immediately apparent from the raw dataset. To add this feature, we manually looked up whether a variety was red or white for all varieties with 30 or more wines in the dataset. We then joined the "Red vs White" CSV to our cleaned data and dropped any wines that didn't have a "red vs white" code (Figure 2.3), which brought our data set down to 130k rows.



	country	description	designation	points	price	province	region_1	region_2	variety	winery	Red?	wineType_encoded
0	US	This tremendous 100% varietal wine hails from ...	Martha's Vineyard	96	235.0	California	Napa Valley	Napa	Cabernet Sauvignon	Heitz	True	1
1	Spain	Ripe aromas of fig, blackberry and cassis are ...	Carodorum Selección Especial Reserva	96	110.0	Northern Spain	Toro	Toro	Tinta de Toro	Bodega Carmen Rodríguez	True	1
2	US	Mac Watson honors the memory of a wine once ma...	Special Selected Late Harvest	96	90.0	California	Knights Valley	Sonoma	Sauvignon Blanc	Macauley	False	0
3	US	This spent 20 months in 30% new French oak, an...	Reserve	96	65.0	Oregon	Willamette Valley	Willamette Valley	Pinot Noir	Ponzi	True	1
4	Spain	Deep, dense and pure from the opening bell, th...	Numanthia	95	73.0	Northern Spain	Toro	Toro	Tinta de Toro	Numanthia	True	1

Figure 2.3 - Cleaned data set with red vs white wine type included and encoded

Exploratory Data Analysis

Data visualization was an important element to incorporate to explore our data further. Our EDA for this project built on earlier exploratory data analysis from Project One. The graphs created on Tableau sought to answer questions such as "Which state has the highest-priced wine?" and "Which wine was the best quality?". We filtered on state, province, wine variety, and wine quality. Color coding on a wine-themed gradient was emphasized to organize areas of our dataset that were meaningful. We can conclude through our visualizations that the top tasted wine varieties were Chardonnay and Pinot Noir. The best quality wine variety according to the points column in the dataset is Nebbiolo. Washington was the state with the highest quality of wine in the US (Figure 2.4) while Nevada carried the spot for most expensive wine in the US.

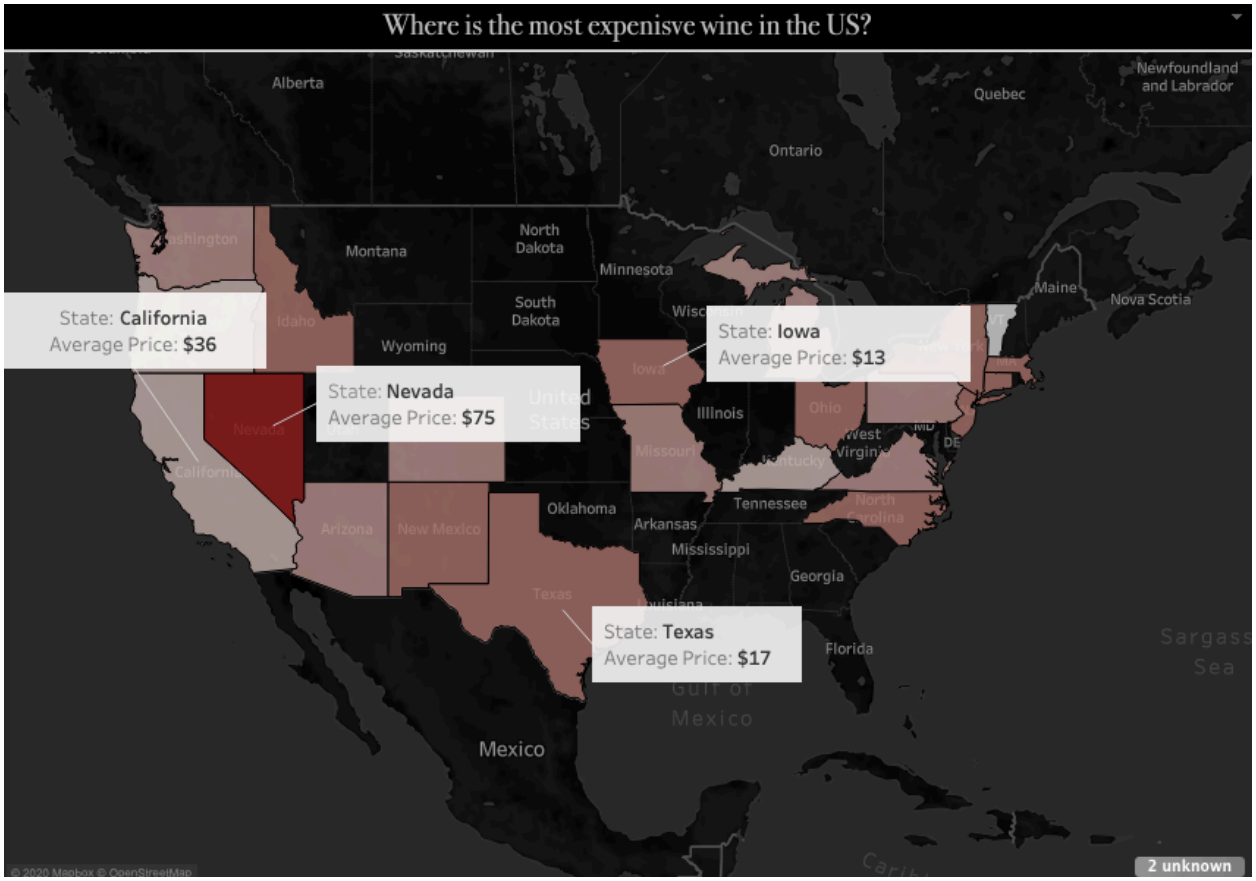
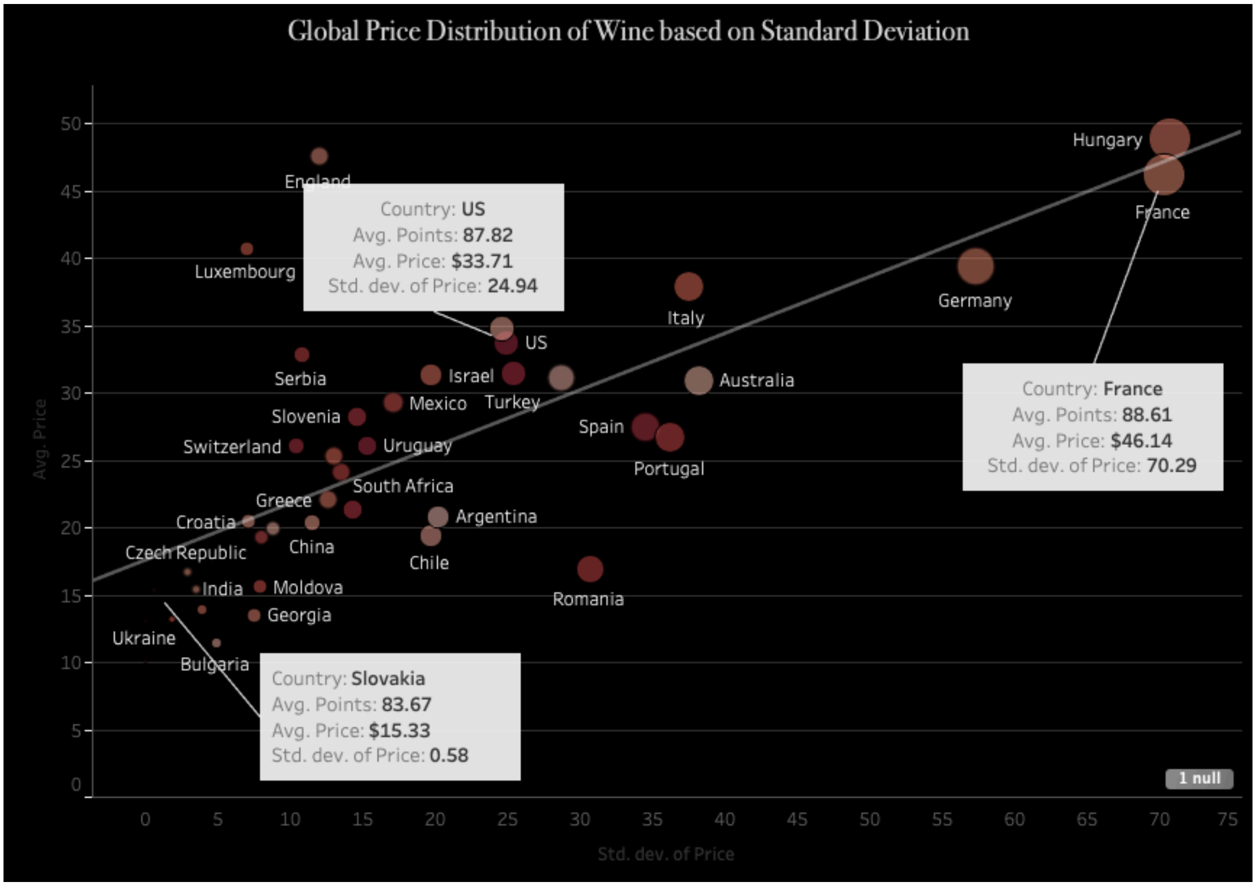


Figure 2.4 - Tableau visual of average wine price per state

Maps were created for the user to visualize where the best quality and most expensive wine is located in the US. The Global Distribution of Price graph (Figure 2.5) was created for the user to analyze how far each country is from the standard deviation of price.



Country: **US**
Avg. Points: **87.82**
Avg. Price: **$33.71**
Std. dev. of Price: **24.94**

Country: **France**
Avg. Points: **88.61**
Avg. Price: **$46.14**
Std. dev. of Price: **70.29**

Country: **Slovakia**
Avg. Points: **83.67**
Avg. Price: **$15.33**
Std. dev. of Price: **0.58**

Figure 2.5 - Tableau visual of the global distribution of wine prices

III. Sentiment Analysis

Data Cleaning

Sentiment analysis is the machine learning technique we utilized on the description column to detect the amount of polarity within the dataset. Since the data used only contains reviews points from 80 to 100 (Figure 3.1), we decided to look at polarity as “good” and “excellent” wine as opposed to “good” vs “bad” wine.

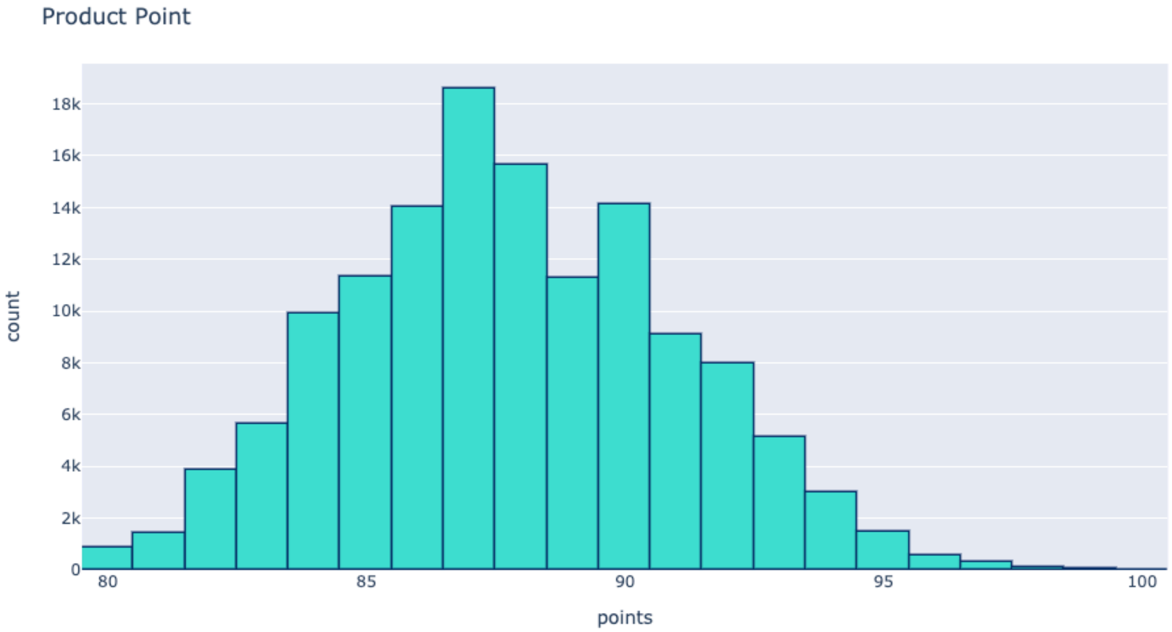


Figure 3.1 – Distribution of points per wine

We calculated some statistics regarding the annotations, such as the number of "excellent" and "good" wines as well as the total number of annotations. We calculated the sentiment of each review using a binary classification model, which takes a sentence as an input and returns 1 or 0, corresponding to "excellent" or "good", respectively. We added a column to the data frame called "Top_Notch_Wine", containing 1 or 0 depending on the wine's points. All reviews with points greater than or equal to 90 were classified as 1 and reviews with points less than 90 are classified as 0.

Sentiment Calculation

After classifying the reviews into two categories, we created DataFrames for each one and built respective wordclouds (Figures 3.2 and 3.3).

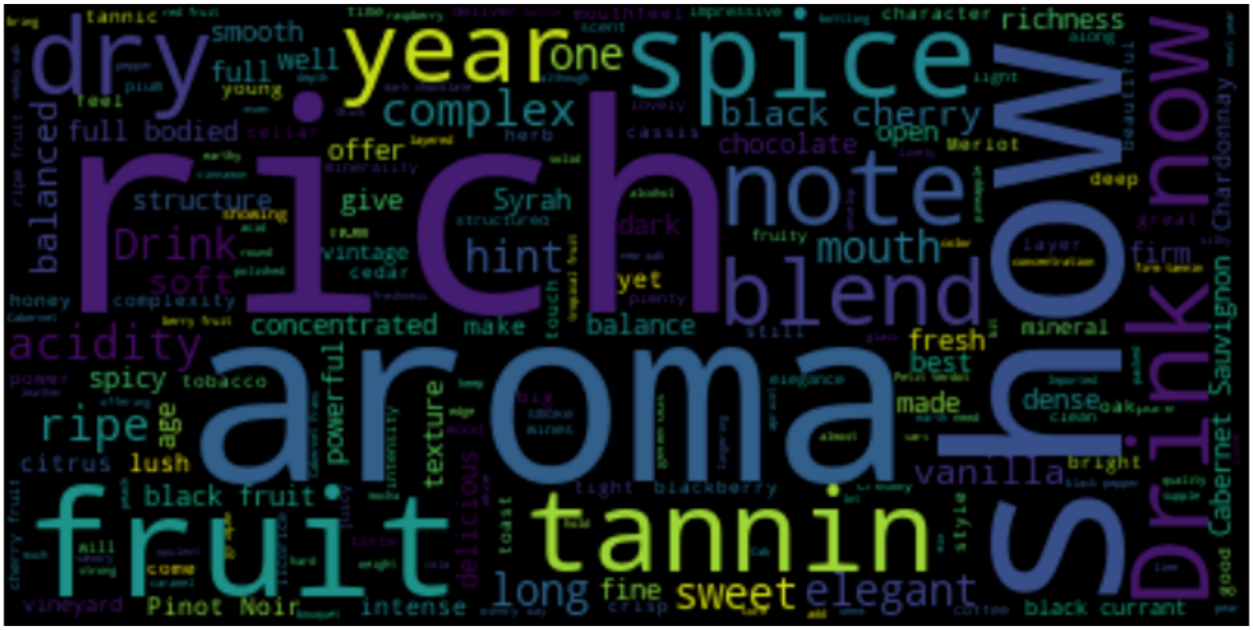


Figure 3.2 - Word cloud for “Excellent” sentiment category

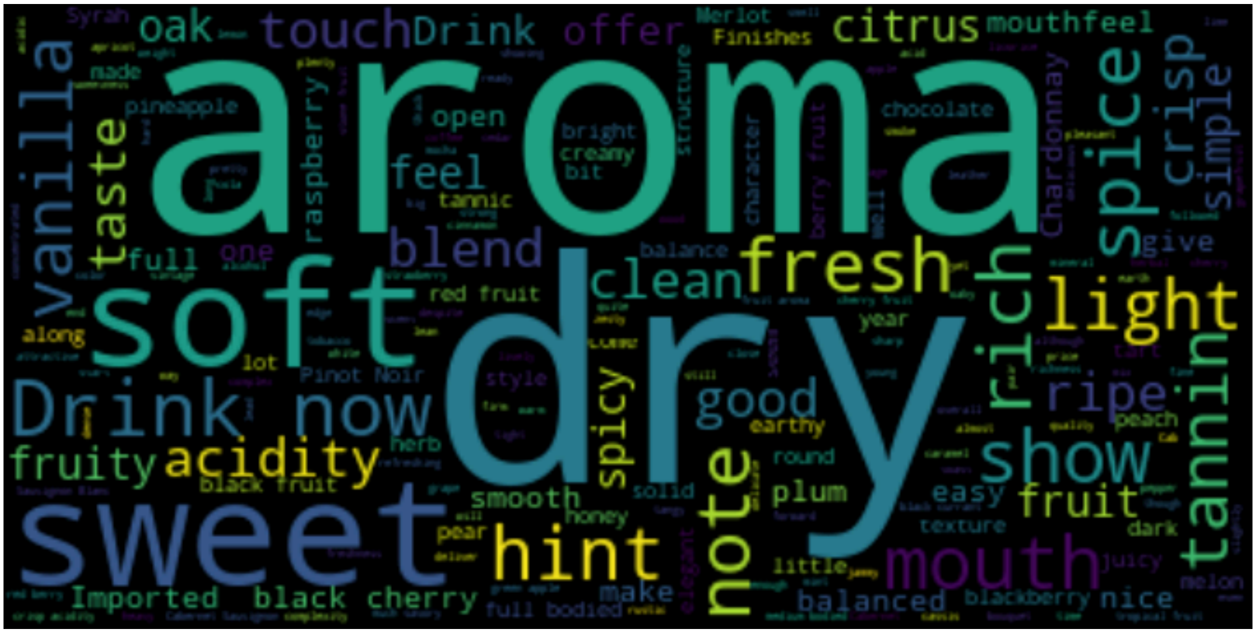


Figure 3.3 - Word cloud for “Good” sentiment category

As seen in the above visualizations, we removed the common words that appear in both visualizations such as “wine”, “flavors”. The “excellent” sentiment word cloud had the word “rich” in high frequency, which does not appear clearly in the “good” sentiment word cloud.

Looking at the distribution of reviews with sentiment across the dataset, we can see that “good” sentiment is double the “excellent” sentiment (Figure 3.4).

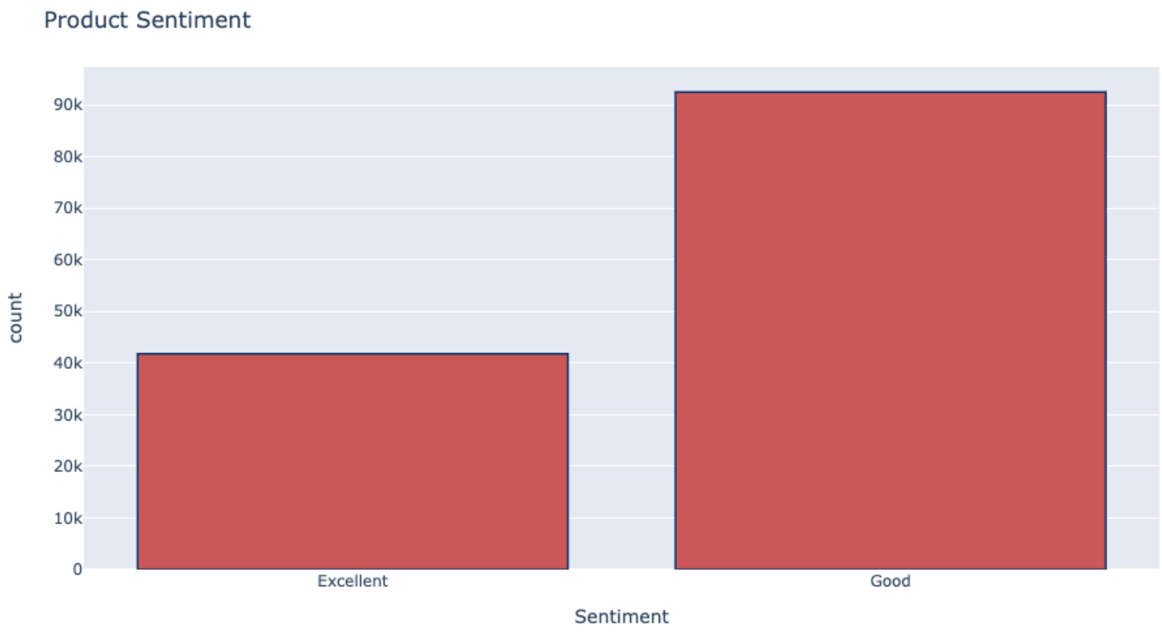
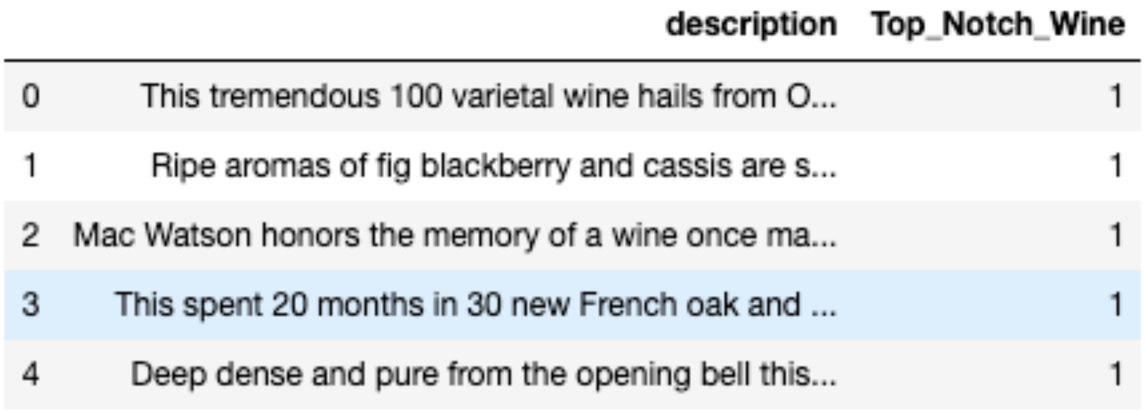


Figure 3.4 – Distribution of wine reviews by sentiment

Building the Sentiment Analysis Model

This model took the description of wine as input, and came up with a prediction on whether the description belongs to the “good” wine category or the “excellent” wine category. Logistic regression is used to solve binary classification problems; since this is a classification task, we trained a simple logistic regression model to solve the problem. First, we started by removing all the punctuation and changed all letters to lowercase. Then, we split the data frame with the reviewed text data and the target variable (sentiment column) as shown in Figure 3.5.



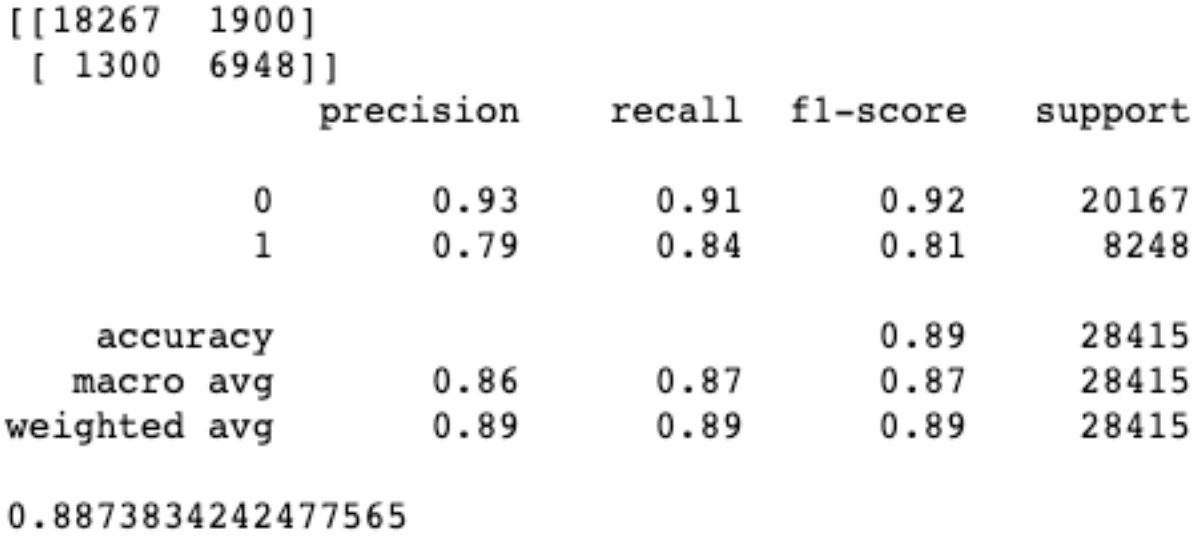
	description	Top_Notch_Wine
0	This tremendous 100 varietal wine hails from O...	1
1	Ripe aromas of fig blackberry and cassis are s...	1
2	Mac Watson honors the memory of a wine once ma...	1
3	This spent 20 months in 30 new French oak and ...	1
4	Deep dense and pure from the opening bell this...	1

Figure 3.5 - Sentiment analysis DataFrame

Bag-Of-Words

We split the DataFrame into train and test sets. 80% of train data was used to train the model and the remaining 20% test data was the data on which the model predicted the classification and the data used to check accuracy. We used the bag-of-words model as a representation that turns arbitrary text into vectors which keeps a count of the total occurrences of most frequently used words, and used CountVectorizer()

for this process. After successfully building a logistic regression model, we trained the data and made predictions using the model (Figure 3.6).



```
[[18267  1900]
 [ 1300  6948]]
              precision    recall  f1-score   support

           0       0.93      0.91      0.92     20167
           1       0.79      0.84      0.81      8248

    accuracy                           0.89     28415
   macro avg       0.86      0.87      0.87     28415
weighted avg       0.89      0.89      0.89     28415

0.8873834242477565
```

Figure 3.6 - Classification report for logistic regression model based on bag-of-words.

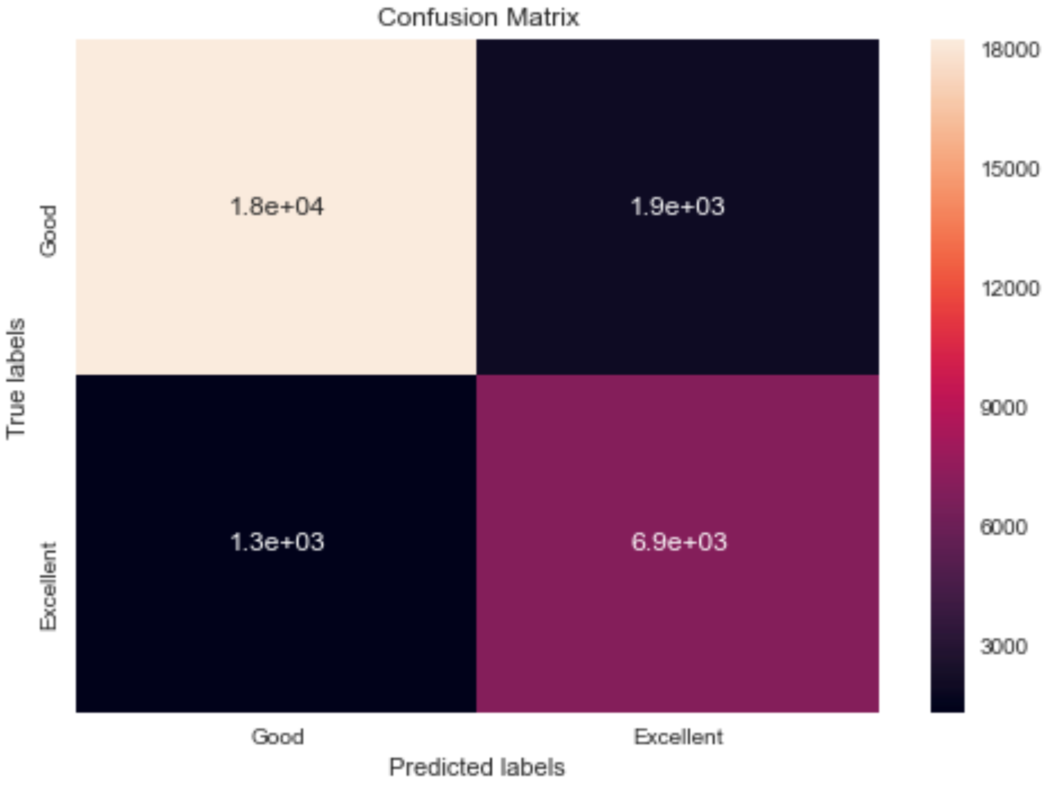


Figure 3.7 - Confusion matrix for the logistic regression model based on bag-of-words

The classification report gives a little perspective of the model performance. In the above result, 0 and 1 represent the “good” and “excellent” wine respectively. It shows the overall accuracy of the model on the test data, 89%, which represents a pretty good result. Accuracy for the “good” sentiment is better compared to the “excellent” sentiment.

IV. Machine Learning

Data Preparation

Initially, our group had run a linear regression model for red wines and white wines between points and prices to see what the correlation was. Looking at the residual plots (Figure 4.1) it’s clear that the basic model was bad at predicting more expensive wines, so our group decided to bin the prices and run a classification model.

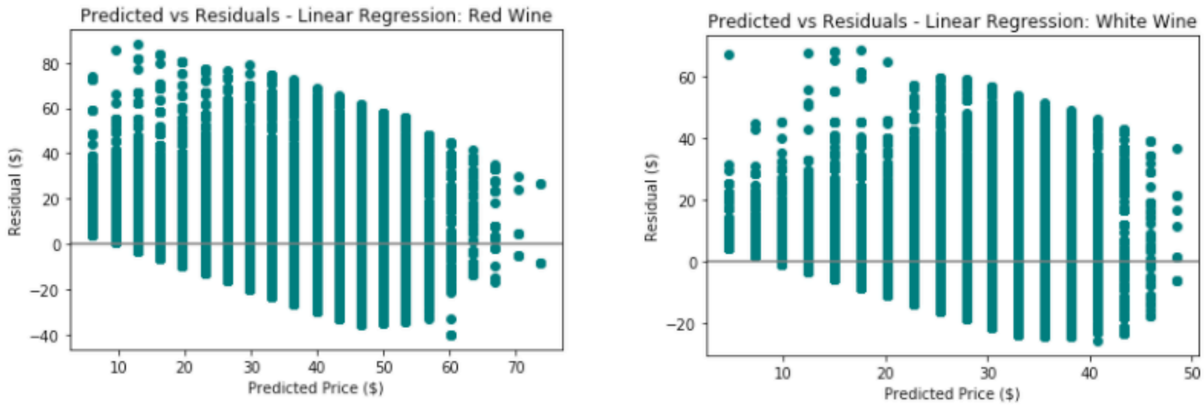
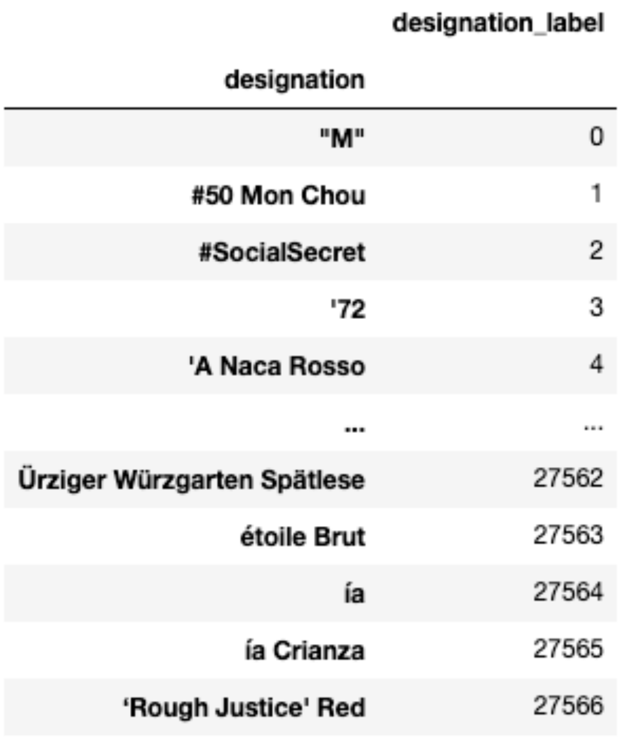


Figure 4.1 - Residual plots for red and white wine linear regressions

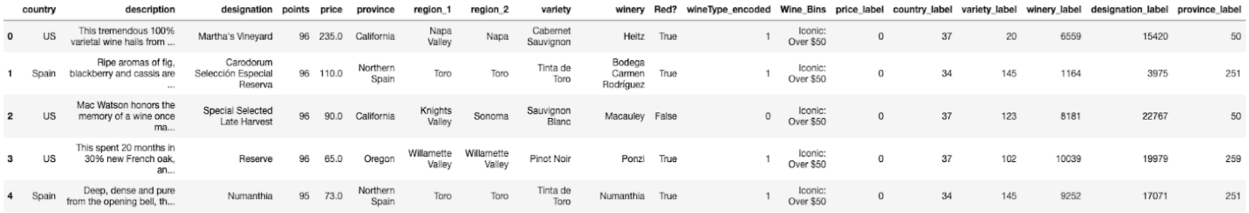
As discussed below, the number of bins and cutoffs took some tweaking, but ultimately our group decided on six bins.

To incorporate categorical features into our model, we had to transform them. Our group decided to use label encoding because of the sheer number of options within each feature (Figure 4.2). For each round we picked features to encode, ultimately picking five additional features: country, variety, winery, designation, and province (Figure 4.3). In each round, we then examined the correlations between our prediction feature, price, and the encoded features (Figure 4.4) and selected features with r > 0.5 and r ≠ 1.



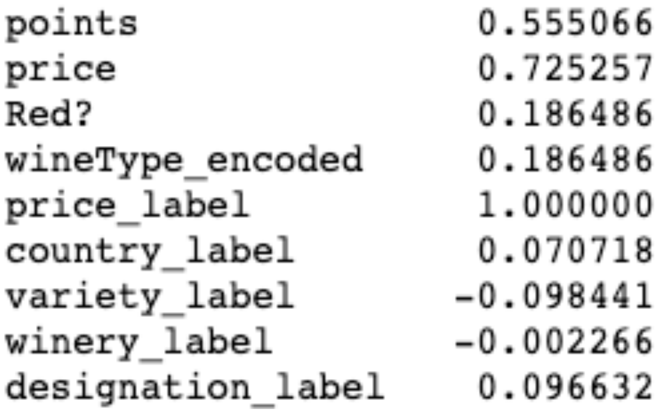
designation	designation_label
"M"	0
#50 Mon Chou	1
#SocialSecret	2
'72	3
'A Naca Rosso	4
...	...
Ürziger Würzgarten Spätlese	27562
étoile Brut	27563
ía	27564
ía Crianza	27565
‘Rough Justice' Red	27566

Figure 4.2 - 27,567 unique values for the designation features



	country	description	designation	points	price	province	region_1	region_2	variety	winery	Red?	wineType_encoded	Wine_Bins	price_label	country_label	variety_label	winery_label	designation_label	province_label
0	US	This tremendous 100% varietal wine hails from ...	Martha's Vineyard	96	235.0	California	Napa Valley	Napa	Cabernet Sauvignon	Heitz	True	1	Iconic: Over $50	0	37	20	6559	15420	50
1	Spain	Ripe aromas of fig, blackberry and cassis are ...	Carodorum Selección Especial Reserva	96	110.0	Northern Spain	Toro	Toro	Tinta de Toro	Bodega Carmen Rodríguez	True	1	Iconic: Over $50	0	34	145	1164	3975	251
2	US	Mac Watson honors the memory of a wine once ma...	Special Selected Late Harvest	96	90.0	California	Knights Valley	Sonoma	Sauvignon Blanc	Macauley	False	0	Iconic: Over $50	0	37	123	8181	22767	50
3	US	This spent 20 months in 30% new French oak, an...	Reserve	96	65.0	Oregon	Willamette Valley	Willamette Valley	Pinot Noir	Ponzi	True	1	Iconic: Over $50	0	37	102	10039	19979	259
4	Spain	Deep, dense and pure from the opening bell, th...	Numanthia	95	73.0	Northern Spain	Toro	Toro	Tinta de Toro	Numanthia	True	1	Iconic: Over $50	0	34	145	9252	17071	251

Figure 4.3 - DataFrame with some encoded features appended



```
points               0.555066
price                0.725257
Red?                 0.186486
wineType_encoded     0.186486
price_label          1.000000
country_label        0.070718
variety_label       -0.098441
winery_label        -0.002266
designation_label    0.096632
```

Figure 4.4 - Correlation coefficients for selected features vs price

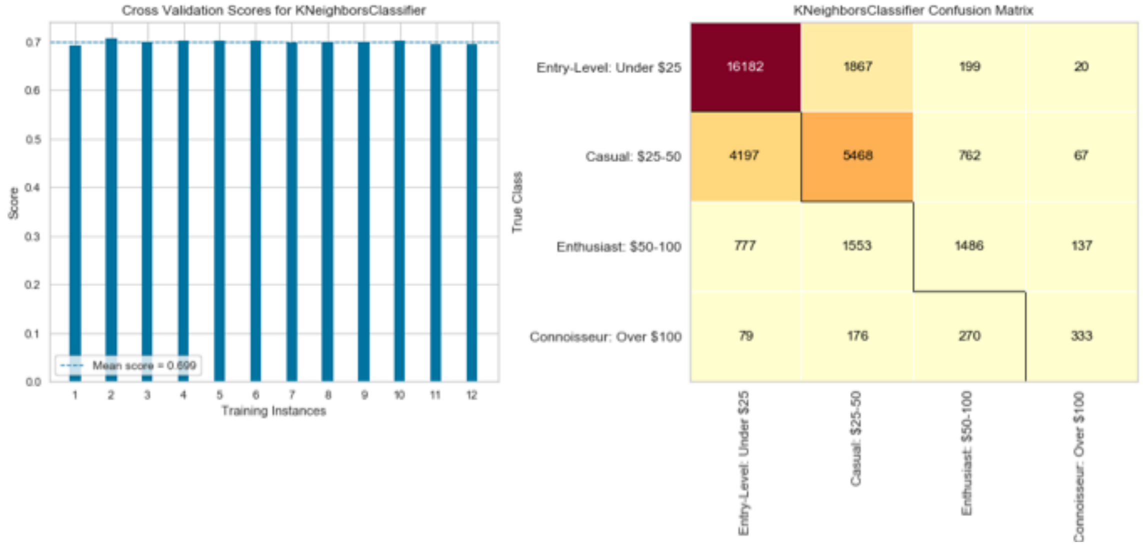
After selecting the appropriate X and y features, we then split the data into test and train sets along a 75% - 25% split. We then scaled the X_train and X_test data sets, which allowed us to run our models.

Model Selection Criteria

When looking at the data that we worked with, it was almost immediately apparent that the data wasn't uniformly distributed and skewed right, which impacted what metrics we utilized to assess different models.

Cross-Validation Scores

To ensure that our models were consistent in their prediction accuracy, we wanted to cross-validate with a sufficient number of folds. However, we were aware that we'd likely have multiple different types of models and several rounds of model creation and a limited deployment deadline, so we didn't want an excessive amount of folds. 12 seemed like a reasonable compromise, so we cross-validated each model 12 times for every round. Overall, the models were pretty consistent in their accuracy scores across the different folds, which was encouraging. An example of the cross-validation graph for one of the models is shown below in Figure 4.5 - note that the dotted line represents the average accuracy score.



	Entry-Level: Under $25	Casual: $25-50	Enthusiast: $50-100	Connoisseur: Over $100
Entry-Level: Under $25	16182	1867	199	20
Casual: $25-50	4197	5468	762	67
Enthusiast: $50-100	777	1553	1486	137
Connoisseur: Over $100	79	176	270	333

Figure 4.5 - Cross validation chart for KNN model in Round One

Confusion Matrices

We used color-indexed confusion matrices to easily see the accuracy and precision of our models. Darker colors represented a higher percentage of values and we aimed for a model with dark red to dark orange across the leading diagonal and as many pale yellow boxes as possible elsewhere. Two examples of confusion matrices are displayed below - note that in Figure 4.6 there are several darker boxes above the leading diagonal, indicating, for example, that the model often had trouble distinguishing between Premium, Super Premium, and Ultra Premium wines. However, in Figure 4.7, the leading diagonal contains mostly dark red and red boxes, with the surrounding boxes mostly pale yellow or dark yellow, suggesting that the number of false predictions was much lower.

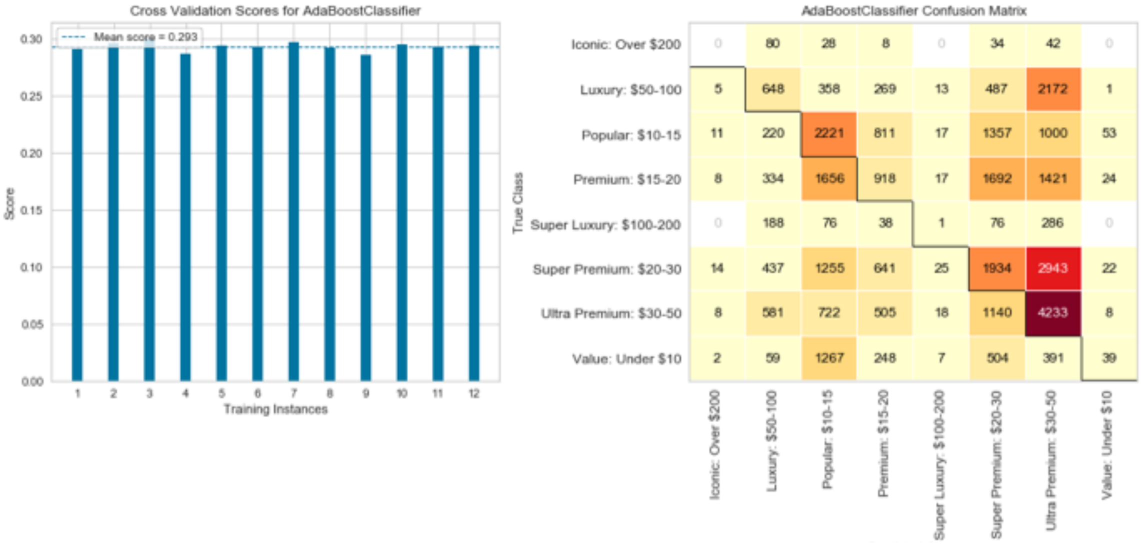


Figure 4.6 - Ada Boost Classifier Matrix, Round Three

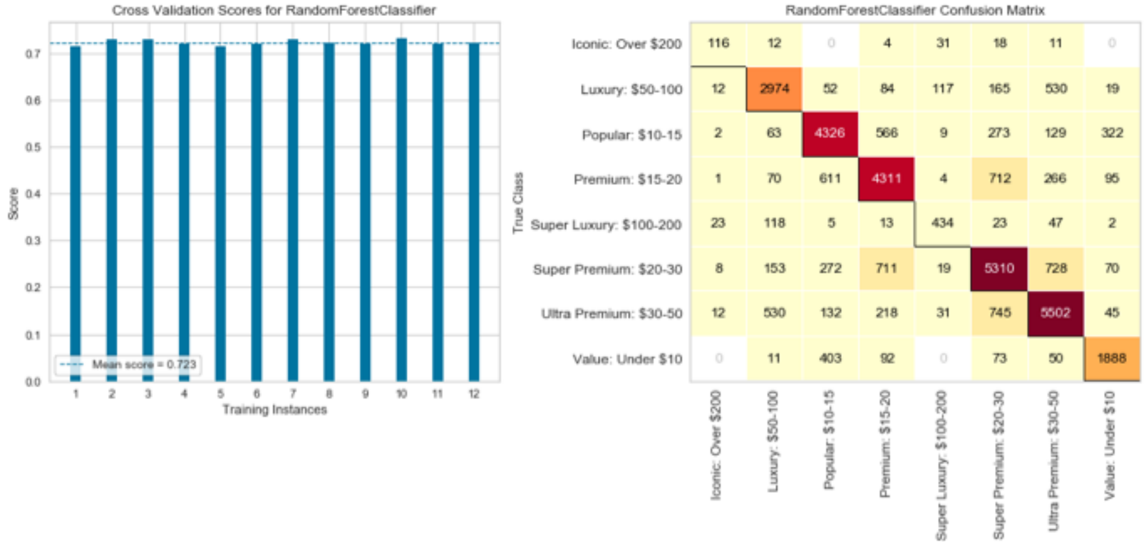


Figure 4.7 - Random Forest Classifier Matrix, Round Three

Classification Reports

The classification reports, like the confusion matrices, are color coded to convey the relative percentages for each prediction class. While normally we would look at accuracy to compare models, since the data is skewed right, we chose to look at f1, which is a weighted average of precision and recall, and is better suited for unevenly distributed data (Huilgol). When evaluating models on the classification reports, we not only focused on the scores but the consistency of scores across prediction classes. Figure 4.8 shows an example of a model that predicted one class with high precision but was inconsistent across the different models.

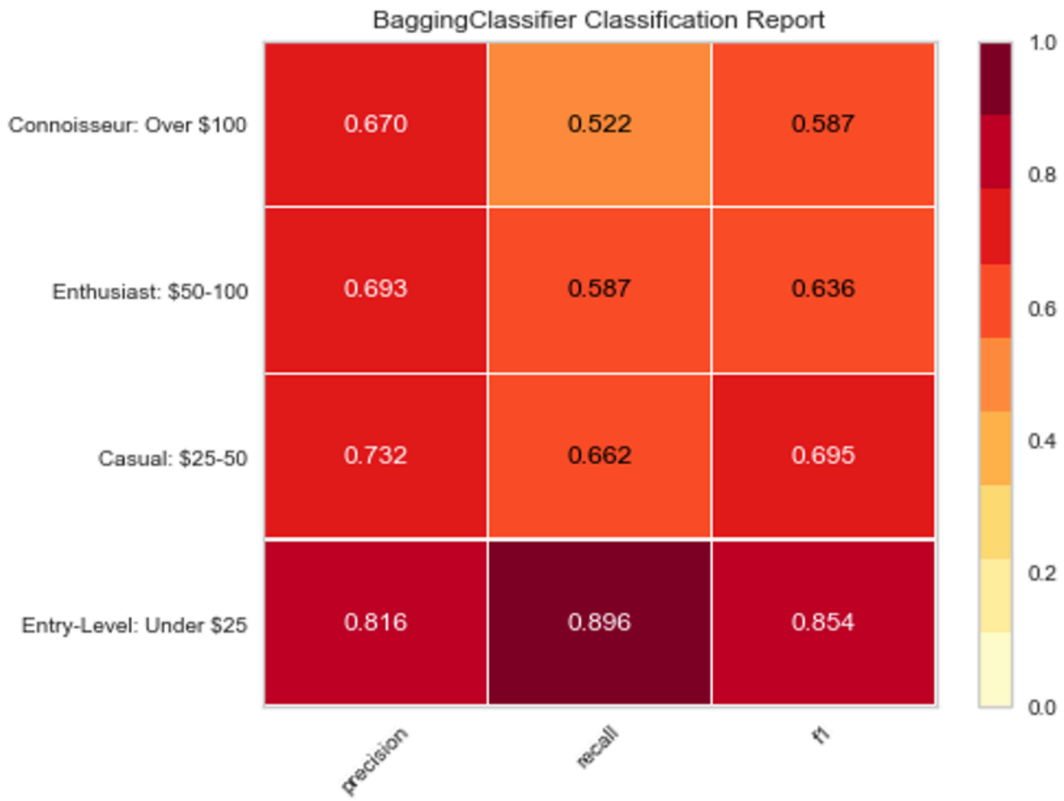


Figure 4.8- Classification Report for Bagging Classifier Model in Round Two

ROC Curves and AUC Scores

The ROC curves are another way to visualize the accuracy of our models, showcasing the proportion of true positive to true negatives. Since our data was skewed, we chose to focus on the micro-average AUC score, which assigns weights to each class's score. Figure 4.9 shows an example of a ROC curve with AUC scores printed in the lower right.

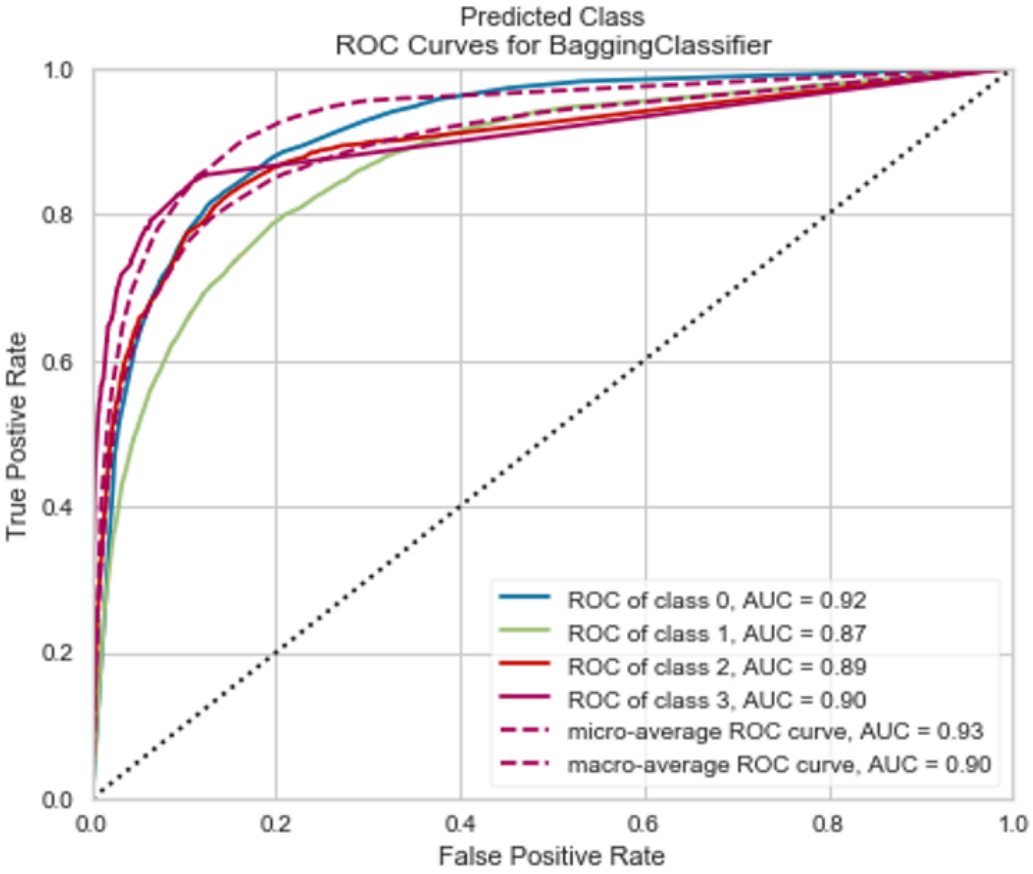


Figure 4.9 - ROC Curve for Bagging Classifier for Round Two

Feature Importance

With the random forest classifier models, we were able to run a feature importance function to tweak what features we wanted to include. Figure 4.10 shows our feature importance visual from our second round.

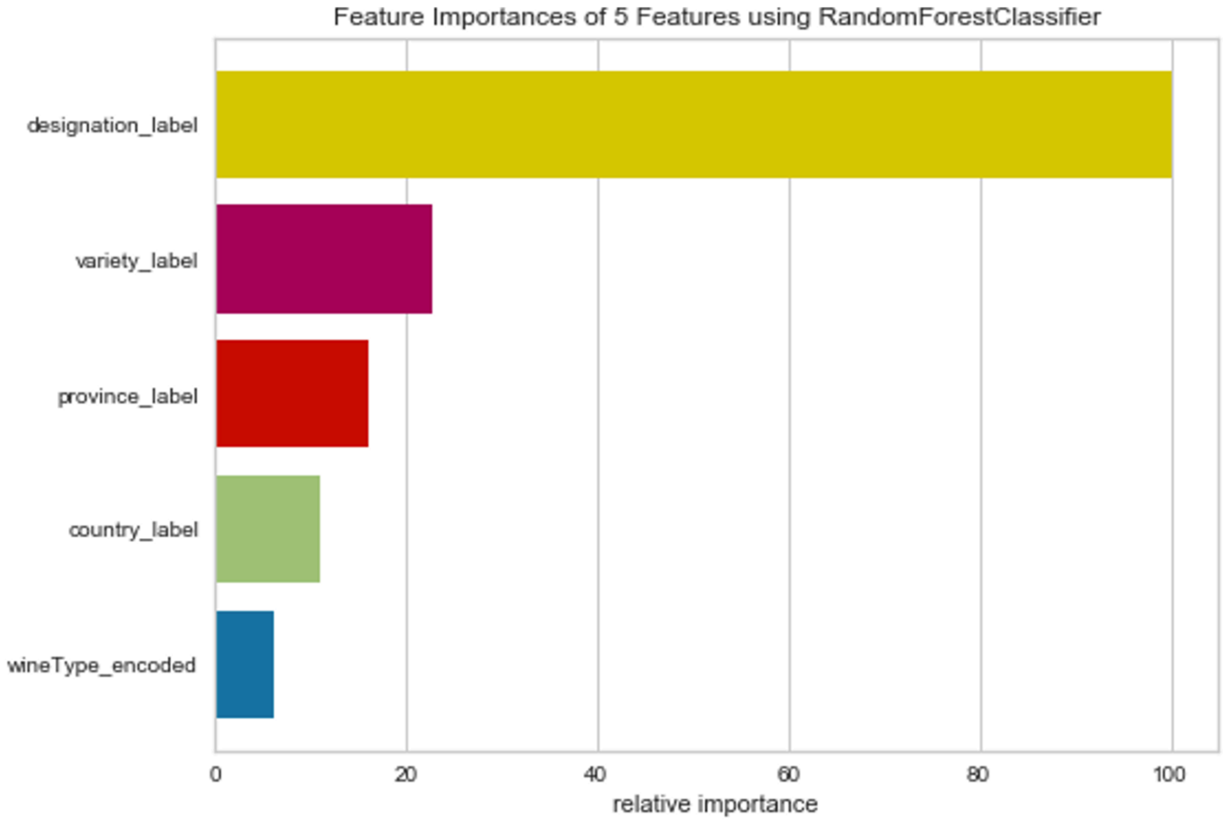


Figure 4.10 - Feature Importance for Round Two

Pearson Coefficients (r)

Finally, we looked at r values between features to determine if there was collinearity between two variables. If there was a high correlation, then the impact of either variable could have been overstated in our model. Figure 4.11 shows an example of a Pearson correlation visual where there was a slight correlation between two features - province and country. We took out the province feature and re-ran the model, and didn't find a significant difference in metrics so we decided to keep both features in.

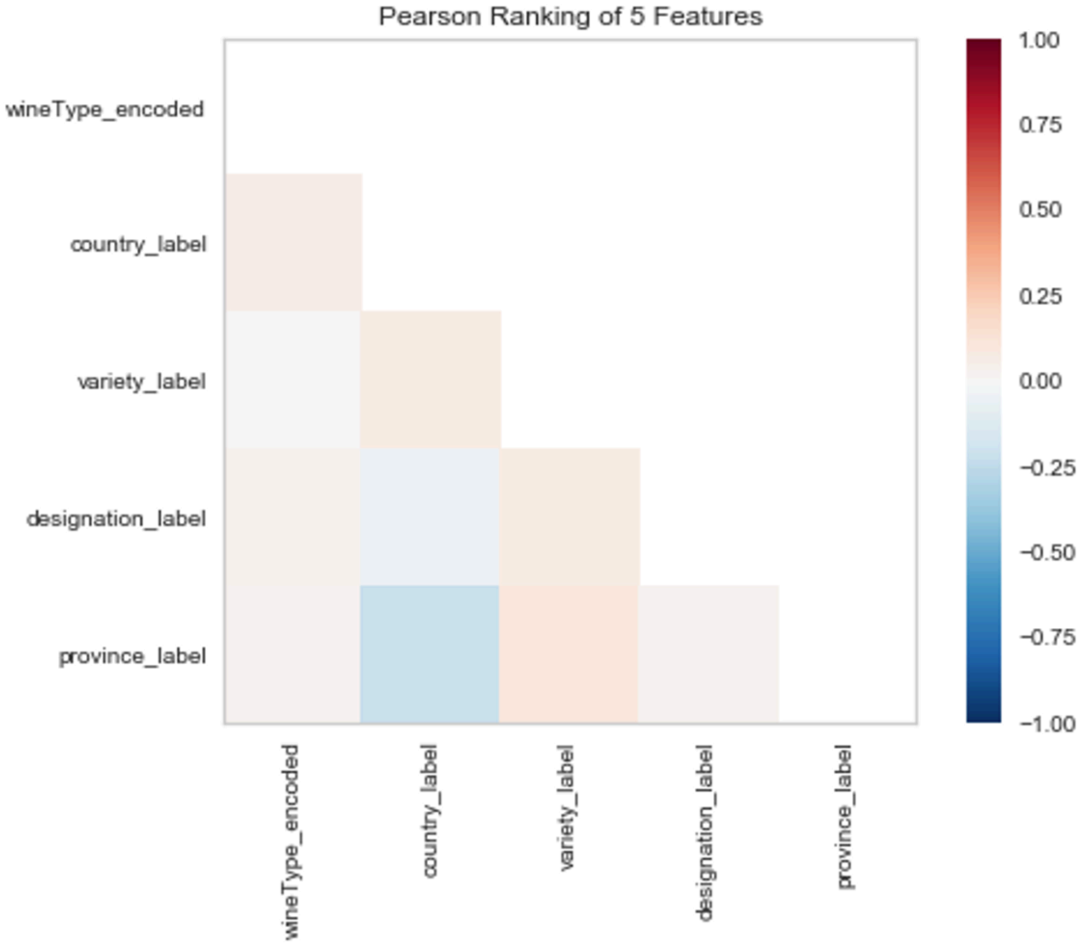


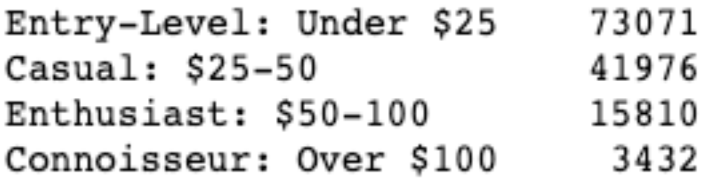
Figure 4.11 - Correlation coefficients between features for Round Four

Model Creation

While there were several tweaks between each round of model trials, all models had three characteristics in common: a random seed of 42, a train-test split of 75% - 25%, and 12 cross-validation folds.

Round One

In the first round of the model trial, we selected the following features: points, wine type, country, variety, and designation. We had broken the prices down into four different bins (Figure 4.12).

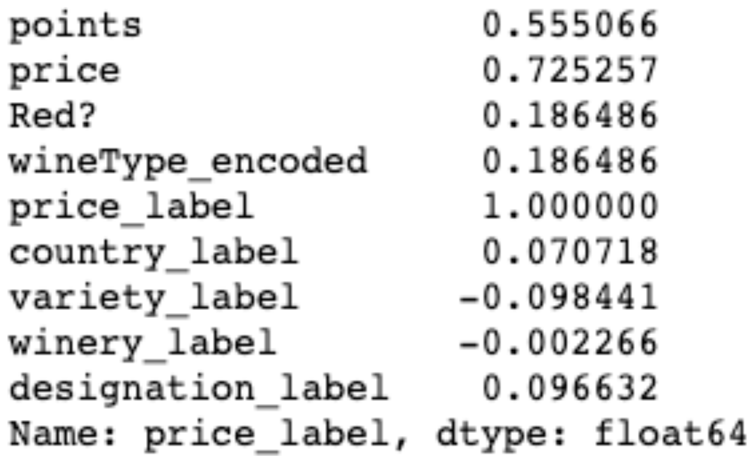


```
Entry-Level: Under $25    73071
Casual: $25-50            41976
Enthusiast: $50-100       15810
Connoisseur: Over $100     3432
```

Figure 4.12 - Frequency of each price bin in the dataset

Round Two

In the second round of the model trial, we selected the following features: wine type, country, variety, designation, and province. While the points feature had a strong relationship to price (Figure 4.13), we realized that a given user wouldn't have that information conveniently available to them and decided to remove the feature from our model. We maintained the same number of bins from the previous round.



```
points                0.555066
price                 0.725257
Red?                  0.186486
wineType_encoded      0.186486
price_label           1.000000
country_label         0.070718
variety_label        -0.098441
winery_label         -0.002266
designation_label     0.096632
Name: price_label, dtype: float64
```

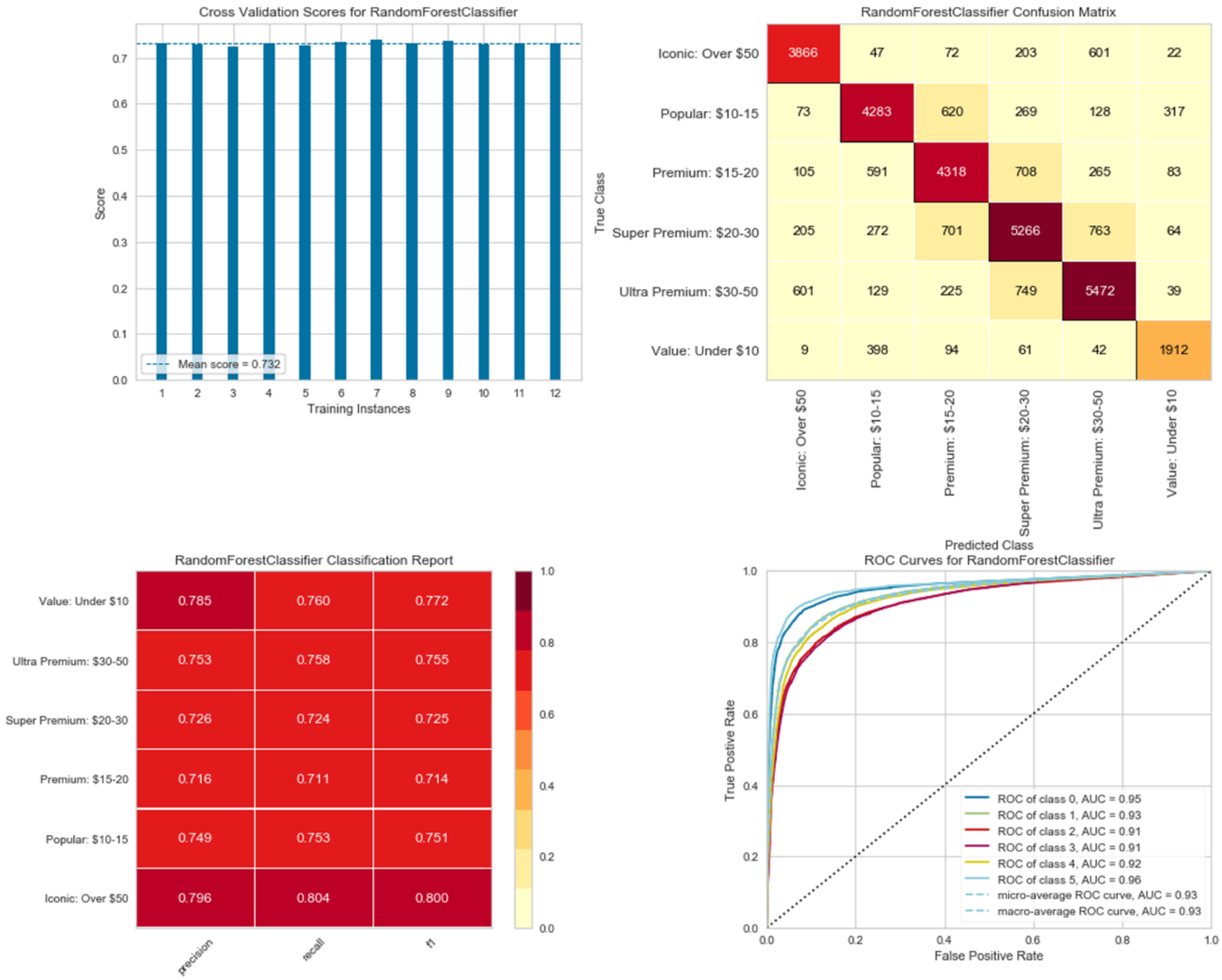
Figure 4.13 - Correlation coefficients between various columns and price_label

Round Three

In the third round of the model trial, we kept the same feature selection but changed the binning, which made the precision of our model more consistent, however the model wasn’t as accurate since there were now too many bins and not enough samples in each class.

Round Four

In the fourth round of the model trial, we kept the same feature selection and decreased the number of bins to six, which gave us an acceptable balance of accuracy and precision. We were between two regression models - Random Forest and Ada Boost (Figure 4.14 and 4.15).



	Iconic: Over $50	Popular: $10-15	Premium: $15-20	Super Premium: $20-30	Ultra Premium: $30-50	Value: Under $10
Iconic: Over $50	3866	47	72	203	601	22
Popular: $10-15	73	4283	620	269	128	317
Premium: $15-20	105	591	4318	708	265	83
Super Premium: $20-30	205	272	701	5266	763	64
Ultra Premium: $30-50	601	129	225	749	5472	39
Value: Under $10	9	398	94	61	42	1912

	precision	recall	f1
Value: Under $10	0.785	0.760	0.772
Ultra Premium: $30-50	0.753	0.758	0.755
Super Premium: $20-30	0.726	0.724	0.725
Premium: $15-20	0.716	0.711	0.714
Popular: $10-15	0.749	0.753	0.751
Iconic: Over $50	0.796	0.804	0.800

Figure 4.14 - Random Forest Classification Metrics- Round Four

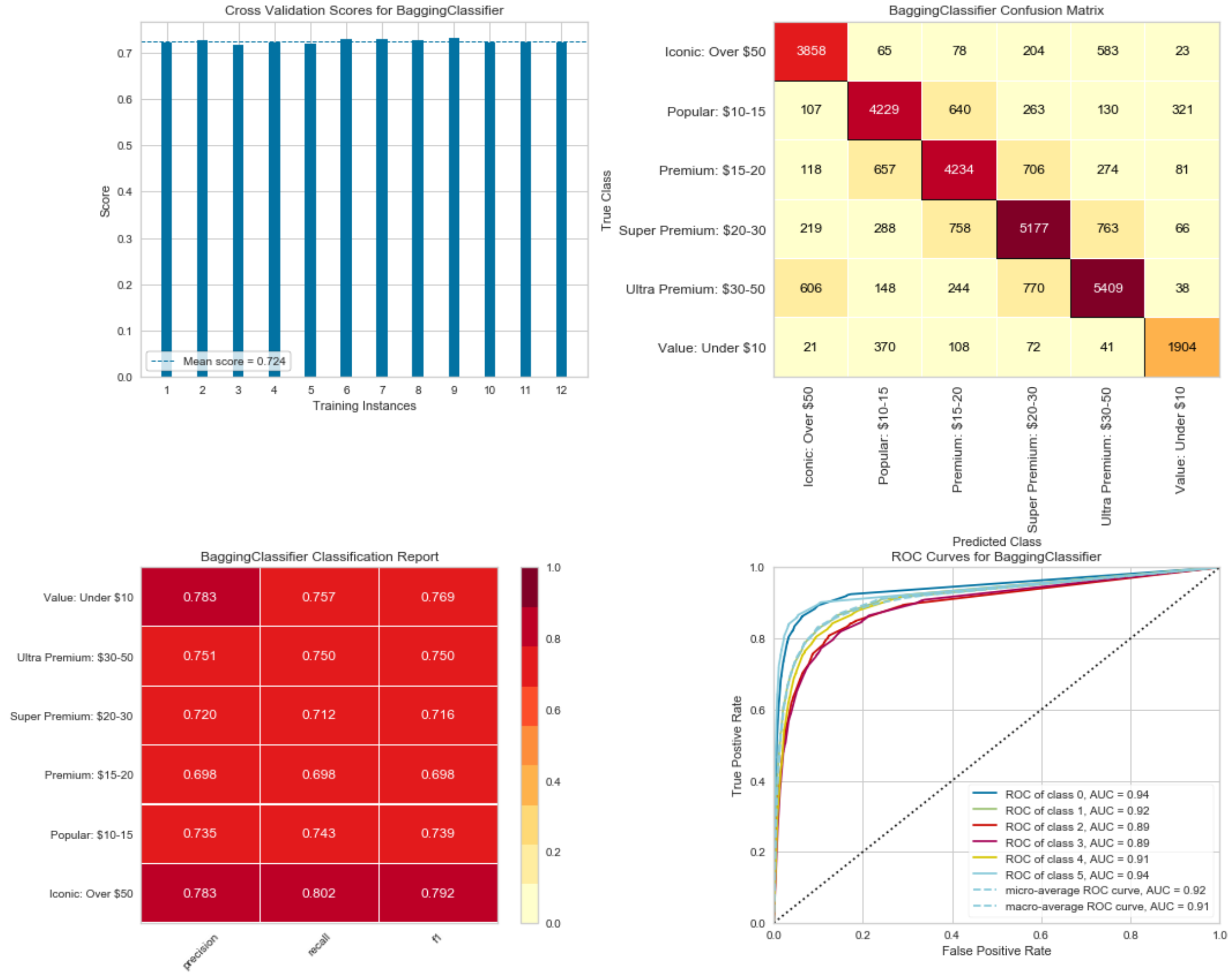
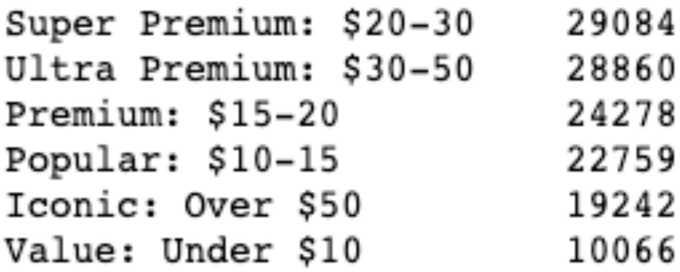


Figure 4.15 - Bagging Classifier Metrics - Round Four

Final Model Selection and Prediction Ability

We settled on the Random Forest Regressor because it was more consistent across our target classes, while still maintaining moderate accuracy and precision scores. When we went to test predictions on our model, it seemed that we had resolved the issue of our model being unable to predict high priced wines. However, we noticed that the opposite trend was occurring - our model seemed to have issues predicting extremely cheap wines, this difficulty is likely due to the lower number of wines in the category (Figure 4.16).



```
Super Premium: $20-30     29084
Ultra Premium: $30-50     28860
Premium: $15-20           24278
Popular: $10-15           22759
Iconic: Over $50          19242
Value: Under $10          10066
```

Figure 4.16 - Spread of wines across different outcome classes

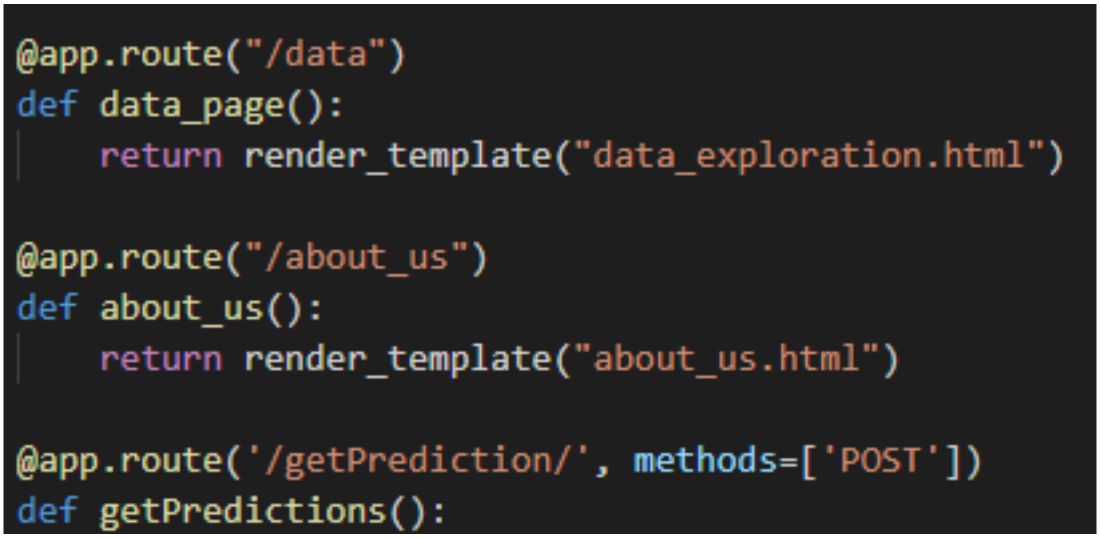
V. Application Creation and Connection

Creation

To begin our final application, we had to first build a skeleton. In order to run a machine learning model on the back-end, we had to integrate the Flask library in Python. The Flask application allows for a front-end user to interact with the back-end to process requests. - in our case, machine learning predictions and file downloads. The app structure was created with a Python file containing the Flask application and templates for the web pages with Javascript and CSS files as needed. After all the files were in place, the final step was to host the completed app publicly onto Heroku.

Connection

For our web page to function correctly, every interactive feature on the HTML needed a route within the Flask app, such as the examples in Figure 5.1. As such, every webpage template was effectively funneled through the Flask app. Connecting the application to Heroku was straightforward and involved connecting the Heroku deployment to our GitHub repository.



```
@app.route("/data")
def data_page():
    return render_template("data_exploration.html")

@app.route("/about_us")
def about_us():
    return render_template("about_us.html")

@app.route('/getPrediction/', methods=['POST'])
def getPredictions():
```

Figure 5.1 – Example app routes for Flask app

VI. Challenges and Future Considerations

Challenges

Most of the challenges our team faced surrounded the machine learning portion of our presentation. For the machine learning portion, it took several iterations to find an acceptable mix of features and bins. While a regression model would have been more accurate, it would have been harder to predict prices since the spread of the data is so uneven.

It was time consuming to run the different rounds of machine learning model trials and keeping track of which output corresponded with which model and round required a certain level of organization. We used Excel to manage this process, which was helpful.

Future Considerations

Given the time constraints for the project we were unable to include the natural language processing models in our final model, however this additional feature could have improved the model. The current model has some difficulty sorting wines on either end of the price scale. By sorting wines into two distinct categories (“good” vs “excellent”) first and then running the current model, the group might be able to better the application’s predictive ability.

Additionally, another iteration could be set up so that a user inputs the name of a wine and then code runs in the background to go to the Wine Enthusiast website, scrape the pertinent data from the wine review, consult the lookup dictionaries we created for each feature, and then feed the corresponding numbers to our model to return a prediction.

Works Cited

Huilgol, Purva. “Accuracy vs. F1-Score.” *Medium*, Analytics Vidhya, 24 Aug. 2019, medium.com/analytics-vidhya/accuracy-vs-f1-score-6258237beca2.

Inspiration

Argabrite, Chelsea. “Chelsea Argabrite - General Assembly - San Francisco ...” *Wine Tableau Dashboard*, Tableau Public, 2020, www.linkedin.com/in/chelsea-argabrite.

S, Terence. “Predicting Wine Quality with Several Classification Techniques.” *Medium*, Towards Data Science, 8 May 2020, towardsdatascience.com/predicting-wine-quality-with-several-classification-techniques-179038ea6434.