Definition

Project Overview

Determining the cost of a bottle of wine is very challenging for the average consumer because there are so many traits that contribute to a wine's value. Furthermore, for any given trait there is a plurality of options (i.e. there are dozens of grape varieties). Thus, when presented with a multivariate optimization problem such as this it's no wonder humans find it difficult to evaluate the value of a bottle of wine. Computers, on the other hand, excel at this type of problem and, in theory, should be much more accurate.

In this project I've created a model that accepts several traits of a bottle of wine from a user and predicts its price. The model uses an XGBoost regression algorithm trained on a large Wine Enthusiast dataset.

Problem Statement

The goal of this project is to train a model that can accept several traits of a bottle of wine as features and predict its price. The final model would be theoretically useful in scenarios where someone may want to assess whether, given the traits of the particular bottle, the price seems fair.

Metrics

R^2 – This metric is a measure of how well future prices will be predicted by a model. The best possible value is 1, corresponding to perfect prediction fidelity to actual labels. R^2 is one of the most frequently used evaluation metrics for regression models. I'll be using it here because the baseline model, which I discuss later, uses this as an evaluation metric. **[1]**

Calculation: https://en.wikipedia.org/wiki/Coefficient_of_determination

RMSE – Root mean squared error is the standard deviation of the residuals (aka prediction errors). RMSE values close to 0 indicate predictions that are clustered closely around the line of best fit for the underlying labels. RMSE is also the default evaluation metric used by XGBoost, therefore implementing it here as a means of evaluation allows me to compare results between test and train/validation sets to determine bias and variance. [2]

Calculation: https://en.wikipedia.org/wiki/Root-mean-square_deviation

Analysis

Data Exploration

The dataset I've selected to train my model is the "Wine Reviews" data set from Kaggle [3]. It's comprised of a CSV file of 130,000 wine reviews scraped from WineEnthusiast.com on November 22nd, 2017. The file contains the following columns:

- **Points**: The number of points WineEnthusiast rated the wine on a scale of 1-100 (though they say they only post reviews for wines that score >=80)
- **Title**: The title of the wine review, which often contains the vintage
- **Description**: Tasting notes provided by the taster
- **Taster name**: WineEnthusiast taster that provided the description
- **Taster twitter handle**: Self explanatory
- **Price**: The cost for a bottle of the wine
- **Designation**: The vineyard within the winery where the grapes that made the wine are from
- **Variety**: The type of grapes used to make the wine (i.e. Pinot Noir)
- **Region_1**: The wine growing area in a province or state (i.e. Napa)
- **Region_2**: Sometimes there are more specific regions specified within a wine growing area (i.e. Rutherford inside the Napa Valley), but this value can sometimes be blank
- **Province**: The province or state that the wine is from
- **Country**: The country that the wine is from
- **Winery**: Wine-maker that produced the wine

I've also extracted vintage, the year that the grapes were harvested, from the title column and added it as an additional feature.

In the notebook entitled "Data Visualization & Exploration" I've provided code illustrating several relevant statistics about the data.

Fig 1. *The following table shows that several of the columns in the data set have significant missing values.*

	Missing Values	% of Total Values
region_2	79460	61.1
designation	37465	28.8
taster_twitter_handle	31213	24.0
taster_name	26244	20.2
region_1	21247	16.3
price	8996	6.9
year	6345	4.9
country	63	0.0
province	63	0.0
variety	1	0.0

Due to the high incidence of missing values the region_2, designation, taster_twitter_handle and taster_name columns will have to be omitted from this model.

Fig 2. *The following tables shows, of the categorical features in the data set, that several of the columns are highly cardinal.*

	Unique Vals	% of Total Vals
description	119955	92.3
title	118840	91.4
designation	37979	41.1
winery	16757	12.9
region_1	1229	1.1
variety	707	0.5
province	425	0.3
region_2	17	0.0
country	43	0.0
taster_name	19	0.0
taster_twitter_handle	15	0.0

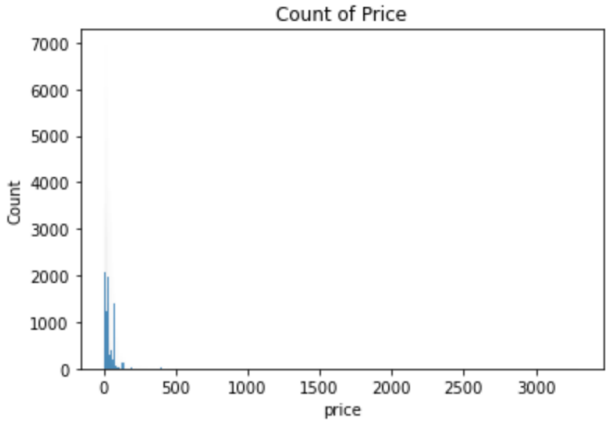
Because several of the columns are highly-cardinal, categorical features dimensionality reduction techniques will be employed to constrain the number of features used to train my model.

Exploratory Visualizations

Price

The following histogram shows how price values are distributed. This visualization is useful because it readily shows where most prices are clustered and that there are outlying values that must be addressed.

Fig 3. *Histogram illustrating a count of price incidence.*



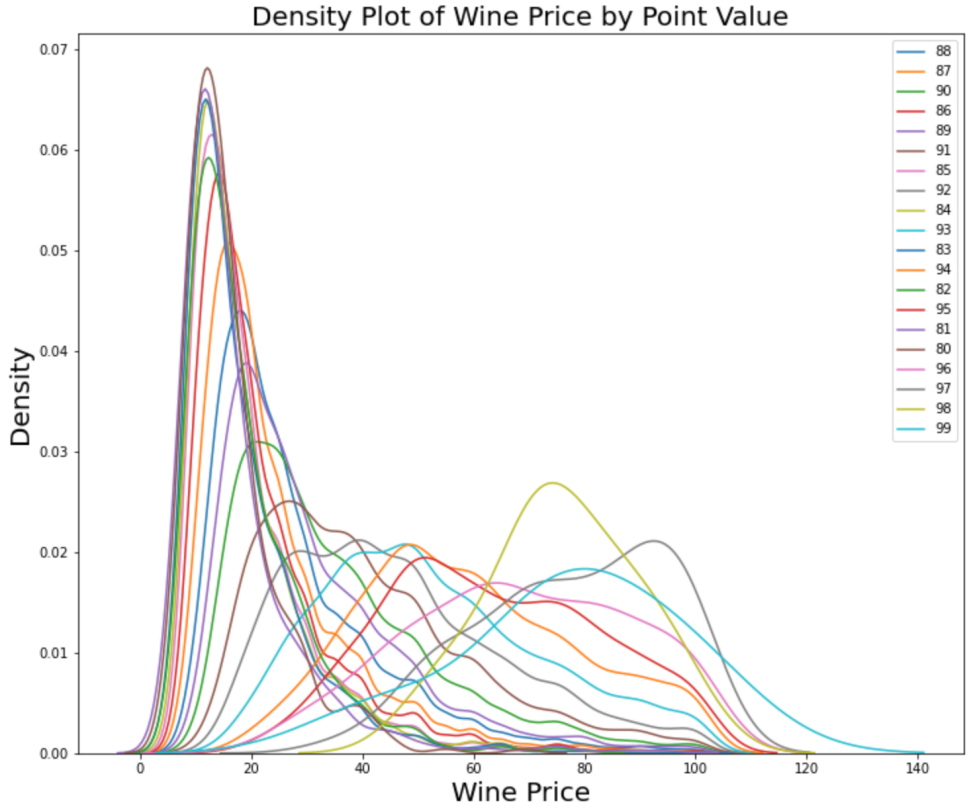
As expected most prices are clustered between 0-$100 but also vary upward widely. This makes intuitive sense because many bottles of wine are priced for the average consumer, but some bottles are rare/deemed high quality and thus are much more expensive than the average.

I intend this model to be used by the average consumer, rather than wine collectors, so bottles that are significant outliers from the average price of a bottle of wine will be removed. In practice I have done this by removing bottles priced at 3 times the interquartile range above and below the 25th and 75th quartiles. This is based on a widely accepted method for removing outliers.

Point Value

The following density plot shows how prices are distributed with respect to Wine Enthusiast point value. This type of plot is useful for visually representing correlations between feature values and price. With this information it can be assessed how impactful a given feature will be for price prediction.

Fig 4. *Density plot showing the distribution of price with respect the Wine Enthusiast Point Value. Note: outlying prices have been removed.*



As we'd initiatively expect, there appears to be a correlation between point value and price. To the right we see the olive, grey and light blue lines corresponding to 97, 98 and 99, respectively, skewing more expense. Alternatively, values in the 80s are clustered at or below the $20 range.

Algorithms & Techniques

The price predictor was trained using the Extreme Gradient Descent Boosting (XGBoost) algorithm. XGBoost has seen wide adoption due to having been the driving strategy behind several notable Kaggle competitions and, as a result, is considered by many to be one of the most performant and efficient algorithms for solving regression problems. XGBoost is a decision-tree based algorithm that employs an ensemble approach, where multiple decision trees are generated simultaneously, and gradient boosting techniques, where loss from earlier rounds of training is used to inform and improve later rounds. [4]

The following parameters can be tuned to optimize the classifier:

- **Max_depth:** Maximum depth of a tree (prevents overfitting).
- **Eta**: Shrinks the feature weights (make the boosting process more conservative).
- **Min_child_weight:** Minimum sum of instance weight (hessian) needed in a child (prevents overfitting).
- **Subsample:** Subsample ratio of the training instance (prevents overfitting)
- **Gamma:** Minimum loss reduction required to make a further partition on a leaf node (regularization parameter)

Note that these are a common subset of a plurality of other parameters that can be modified. Full list here.

Benchmark

In the Medium article, "Vintage, AVA and Quality: A Study of Napa Valley Wines" the author tests several types of regression models - Linear, SVM, Decision Trees, Ensemble - to predict the price of wine, ultimately landing on a Random Forest Regressor with an R^2 value of 0.51 [5]. In her model she uses AVA (American Vinicultural Area), Grape, Score and Vintage (year) as features. I've used this result as a benchmark, attempting to improve performance by adding different/additional features and testing a different algorithms, comparing my results to the reported R^2.

Methodology

Data Preprocessing

The preprocessing in the "Data Cleaning & Preparation" notebook consists of the following steps:

1. Load data into memory
2. Extract vintage (year) column from title
3. General Data Cleaning: these are data cleaning steps that can be performed across the dataset without risk of data leakage

    - Remove index column
    - Remove duplicate rows
    - Remove columns with significant null values
    - Remove categorical columns with insurmountable cardinality
    - Remove rows where price (the label) is null
4. Split data into train, validation and test sets
5. Sensitive Data Cleaning: these are data cleaning steps that need to be performed after the train/validation/test split
    - Remove outlying price values
    - Impute missing vintage values with mode
    - Replace countries of low-incidence with 'other'
    - Replace provinces of low-incidence with 'other'
    - Replace regions of low-incidence with 'other' (based on incidence on a country basis)
    - Replace varieties of low-incidence with 'other'
6. One hot encode categorical values

Implementation

Implementation of my price predictor occurred in the "Regression" notebook and consists of the following steps:

- Loading preprocessed training and validation data sets from S3
- Retrieving the location of an AWS XGBoost image and use it to instantiate an estimator object
- Set training hyperparameters
- Create a hyperparameter training object using the estimator object and hyperparameter ranges
- Train multiple models with the hyperparameter tuner
- Attach the XGBoost estimator object to the best model from the previous step (evaluated based on RMSE)
- Use the best-model estimator object to instantiate a Sagemaker transformer
- Using the transformer and test set, make predictions
- Plot actual price vs. predicted price to get a visual sense of prediction accuracy
- Calculate R^2 and RMSE for the test set
- If evaluation metrics indicate large amounts of prediction error either a) return to bullet three, or b) return to preprocessing/engineering of features

Refinement

As mentioned in the benchmark section, the comparison model, trained using AVA, grape, score and vintage, achieved an R^2 of .51. To get an initial point of comparison I selected country, WineEnthusiast point value, variety (grape) and vintage as features. The resulting

model trained using XGBoost resulted in test set R^2 and RMSE values of .45 and 14.9, respectively. Since the RMSE value of the validation set, as output by the XGBoost training process, was 17.05 this indicated to me that we were in a high bias situation – as opposed to high variance – and thus more/better features were needed to improve model performance.

In subsequent trainings I tested swapping out and in different features individually and in combination, evaluating how each affected R^2 and RMSE. I also used SageMaker's hyperparameter tuner, which trains several models in parallel with varied parameter values allowing for efficient tuning.

The final model, which was established using the best result of feature tests and hyperparameter tuning, resulted in test set R^2 and RMSE values of .46 and 14.98, respectively.

Results

Model Evaluation and Validation

During training a validation hold-out set was used for evaluation. Afterwards, a Sagemaker transformer object was used to evaluate model performance on a further test hold-out dataset.

The final list of features and hyperparameters were chosen because they performed best out of the combinations tested.

To get a sense of where price predictions were falling I've plotted price vs. predicted price. Also, sampling a subset of 25 examples from the test set, I used the model to predict their prices and looked at the features of these examples to see if there were any obvious trends in errors.

Fig 5. *Scatter plot of actual bottle price compared to predicted price.*

If the model was perfectly predicting price values we'd expect to see points fall along a 45 degree line in this scatter plot. Because points are widely distributed we can see that the model is not predicting prices with high-enough accuracy.

Fig 6. *Selected wines, their features, price and predicted prices.*

Features	*Price*	*Predicted Price*
Points: 91 *Year: 2003* *Country: Austria* *Province: Other* *Region: Kremstal* *Variety: Riesling*	**$84**	**$38**
Points: 84 *Year: 2008* *Country: France* *Province: Other* *Region: Burgundy* *Variety: Chardonnay*	**$18**	**$19.66**
Points: 90 *Year: 2013* *Country: US* *Province: Washington* *Region: Yakima Valley* *Variety: Other*	**$15**	**$31.58**

Some price predictions show glimpses of accuracy, such as the French Chardonnay above, but an equal number, such as the first and third examples above, are too inaccurate to be useful.

Justification

As is clearly evidence by comparison of the their R^2 values, the final model in this project fails to exceed the performance of the benchmark. Looking at individual examples (as I have done at the end of the 'regression' notebook) there are many examples where the predicted price is enticingly close to the ground truth price value. Unfortunately in this case there are also many values that deviate significantly (as is clear from the scatter plot above). Because of this, the current model does not adequately solve the problem posed in this project.

Conclusion

Reflection

The stages for this project can be summarized at a high level as:

1. Identify problem and a suitable dataset
2. Identify benchmark
3. Load and preprocess data
4. Train a regression model iteratively
5. The best model was identified, validated and tested

During implementation of this project I found step 3, preprocessing, to be the most challenging. Naively, when I selected the problem and dataset, I assumed that because of the many factors used to describe a bottle of wine, some combination of the several I had to choose from would result in a strong predictor of bottle price. What I failed to realize, though, was that most of these factors are categorical and highly cardinal. And because I was unfamiliar with the strategies for preprocessing/engineering these type of features, doing so required significant research.

Conversely, the part of the project that I most enjoyed and felt strongest in was using AWS's Sagemaker package. By the end of the project I had a good sense of how to use the estimator, hyperparameter tuner and transformer objects such that training and deploying models became one of the easiest parts of the workflow.

Improvement

The current model illustrates low R^2 and high RMSE values which are characteristic of models with high bias. In these situations much more time must be dedicated to data preprocessing/feature engineering and iterative training using new and different sets of features.

Given more time those additional steps I would likely take are:

- Delve deeper into preprocessing highly cardinal, categorical features to determine the most effective techniques and evaluate multiple solutions.

- Dig into and attempt to group the examples that show significant error to determine if they are of a particular class or are similar in some way that can be addressed in prepossessing.
- Add additional features to the dataset such as location data that takes into account weather factors such as sunlight, temperature and rainfall, all of which, based on my research, affect wine quality and thus wine price. [6]
- Find additional data with a wider range of vintages (spanning back further than 2000 and more recently than 2017) to expand the underlying dataset and further inform training.

Citations

1. https://www.kaggle.com/residentmario/model-fit-metrics
2. https://www.statisticshowto.com/probability-and-statistics/regression-analysis/rmse-root-mean-square-error/
3. https://www.kaggle.com/zynicide/wine-reviews
4. https://towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d
5. https://medium.com/the-wine-nerd/vintage-ava-and-quality-a-study-of-napa-valley-wines-565e6f164c08
6. https://www.winespectator.com/articles/whats-behind-the-bottle-price-9931