Wine Price Prediction
ORIE 4741 Project

Yuanzheng Cao (yc2575), Zhoutong Li(zl683), Tao Ge(tg367)

December 2019

Abstract

In this paper we made efforts to build a model to predict the price of wines. With the dataset from Wine Enthusiast reviews, we explored both linear models and non-linear models, and achieved satisfying results. We hope that the prediction model would help both consumers and wine manufacturers as a reference when purchasing or pricing the wine.

1 Introduction

If you have ever been shopping for a bottle of wine, you would probably be overwhelmed by different types of wines and their price. When evaluating a bottle of wine, we often consider the producer, the variety, the winery and the time that it produced. However, we do not have any formulas to calculate the price of the wine and do not know whether the real value matches the price labeled or not. What factors affect the price of wine and based on those features and models established, how do we predict wine price?

Our goals and models aim to predict the price of the wine given its producer, variety, winery, ratings and reviews, etc. Furthermore, the model could not only help consumers know the approximate price given by their preference, but also make producers aware that what kind of wine are more valuable, and gain more profits.

2 The Dataset

The dataset we used is the Wine Enthusiast reviews dataset found in Kaggle. This dataset contains more than 100k lines of wine reviews and information from a professional wine information sharing and ecommerce website, Wine Enthusiast, originated from Wine Enthusiast Magazine. Following are descriptions of some important features in the dataset. Detailed statistics will be introduced in the upcoming section.

`price`: prices of the wine, in US dollars.

`points`: rating of the wine, on a scale of 0-100.

`country`, `province`, `region_1`, `region_2`: origin of the wine in different geographical granularity.

`variety`: type of grape used for the wine.

`winery`: producer of the wine.

`description`: a brief description of the wine, provided by individual tasters.

`taster_name`, `taster_twitter_handler`: taster identifier.

3 Exploratory Data Analysis

We first examined the type, missing values and unique values of each feature, and details can be found in the table. Then we plotted the distribution of real valued features: points and price. Point is roughly normally distributed and ranges from 80 to 100. As for price, the vast majority are below 150 dollars, with only several ones (1.2%) more expensive than 150 dollars.

column	unique values	missing count	missing%
Unnamed: 0	0	0.0%	int
country	43	63	0.0%
description	-	0	0.0%
designation	35,777	37,465	28.8%
points	-	0	0.0%
price	-	8,996	6.9%
province	423	63	0.0%
region_1	1205	21,247	16.3%
region_2	18	79,460	61.1%
taster_name	-	26,244	20.2%
taster_twitter_handle	-	31,213	24.0%
title	-	0	0.0%
variety	698	1	0.0%
winery	15855	0	0.0%
year	89	0	0.0%

Figure 1: The Dataset

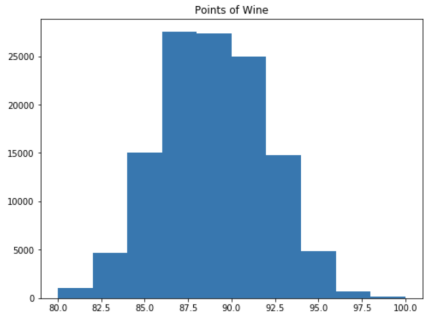

Figure 2: Points distribution

We performed simple linear regression to examine the relation between point and price. As in the table, they are positively related, but point alone is definitely insufficient to predict the price. Then we run correlations between the categorical features. As we can judge from the table below, the regional related features are highly correlated with each other and we decided to select one of them by running by linear model between features and price with the lowest mean squared error -`region_1`. We also utilized Catboost feature importance ranking to further reduced our features if necessary. According to the result (figure), we eventually used features-`points`, `variety`, `region_1` because they are significantly important and designation since it is hard to get `taster_name` given by a new wine and winery has too many different and somehow distributed sparsely.

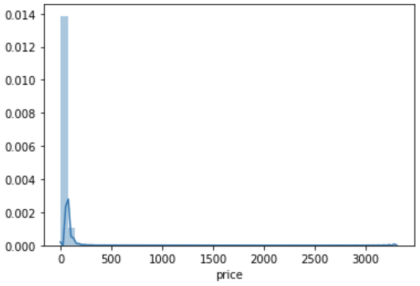

Figure 3: Price distribution

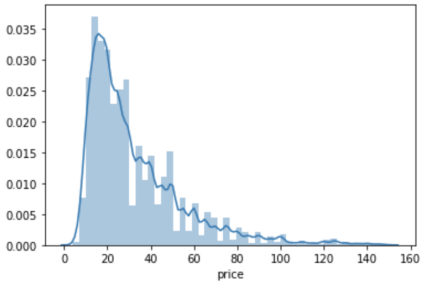

Figure 4: Price$150 distribution

	price	country	region_1	region_2	province	variety	designation	winery
country	0.08	-	-	-	-	-	-	-
region_1	0.12	0.99	-	-	-	-	-	-
region_2	0.19	nan	1.00	-	-	-	-	-
province	0.08	1.00	0.99	1.00	-	-	-	-
variety	0.05	0.50	0.40	0.28	-	-	-	-
designation	0.37	0.64	0.58	0.60	-	-	-	-
winery	0.24	0.93	0.60	0.82	0.72	0.40	0.57	-

Figure 5: Correlations between some features

4 Data Pre-processing

4.1 Categorical Data

Since our goal is to predict wine price, we removed all rows missing the `price` column. We also removed one rows missing the `points`(rating) and the `variety` column (only one line).

Both `taster_name` and `taster_twitter_handler`, which encodes wine reviewer information, are removed. Assuming that the description are objective, we do not need to take into consideration who wrote the review. Besides, it is impractical to ask the same group of viewers to write description for the wine we want to predict the future. All the categorical features are then processed with one-hot encoding. Title contains repetitive information that can be represented by other columns, so we extracted year from it and then removed title. We treat the years as categorical variable since years does not have obvious numeric relationship with price but we get a good score if we treat it as classification features after running logistic regression. For description with long texts,we used NLP tools to process them, which is introduced in the next subsection.

For the missing values in `region_1`, if its province was not missing, we randomly chose one `region_1` under that `province`. For province is missing, we treat province as `region_1` and it at least give some information for that whole areas.

We tried one-hot encoding for the categorical features. Because most of our features are categorical features and the unique values added are quite large, we used hash function which transfer each categorical value into d features vectors using different number and same value will have the same vectors.

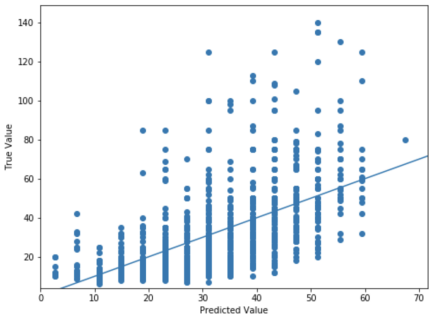

Figure 6: first figure

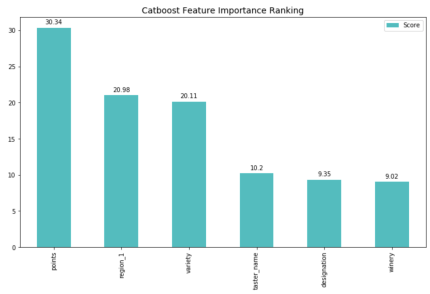

Figure 7: second figure

4.2 Text Pre-processing

`Description` is a paragraph of text describing the wine. At first, we used simple uni-gram (bag-of-words) model to represent the description. Uni-gram suffers from keeping semantics like "not good". As an improvement, we then applied bi-gram (two consecutive words) and used TF-IDF (see reference for an easy-to-understand introduction) instead of simple word count as feature representation. The whole text pre-processing includes removing punctuation and stopwords(most common words like "a", "the"), creating bi-gram tokens, calculating TF-IDF and forming feature matrix. However, this resulted in 736,059 columns, which was more than five times the number of data entries and would almost definitely result in overfitting. So we gave up bigram TF-IDF and turned to Google Universal Sentence Encoder.

Google Universal Sentence Encoder has a pre-trained neural network and can encode arbitrary length of text into a feature vector of length 512. We took advantage of it and transformed description into a feature matrix with the width of 512.

5 Models and Results

We tried linear models and trees, and they will be introduced in detail. To detect and avoid overfitting, we divided the dataset into training set and test set with the ratio of 4:1 (20% data as test set), and calculated both training and test MSE, MAE, R2 and explained variance. For the models with hyperparameters, we used 5-fold cross validation to first determine the hyperparameter(s).

For linear models, we tried the most simple least squares as our baseline.

- Huber Regression

Featuring the robust huber loss function that decomposes the error into two parts, a small part of normal distribution and a large part that is robust. Huber regression is particularly useful when large outliers appear. The result of huber regression is no better than least squares. This is probably because we have already removed wines with very high prices. Though they were not necessarily outliers, when it comes to abnormally highly priced merchandise, or in this case wine, the reason of its price may not be the objective attributes but other considerations like brand positioning, which are not reflected in the data.

- Lasso and Ridge Regression

Lasso and ridge regression uses L1(absolute value) and L2 regularizers respectively. There is a hyperparameter, λ, we can play with to get better performance. To determine λ, we used 5-fold cross validation on potential values of 1e-15,1e-10,1e-8,1e-4,1e-3,1e-2,1,5,10 and 20. The final result showed that when =0.001 and 20 respectively, lasso and ridge regression performs the best.

- Trees and Random Forest

Generally the performance of linear models did not vary much, so we sought help from non-linear models.

The dataset was full of categorical features, so we thought that decision trees might work well. Because we have many categorical features, to avoid overfitting, we pruned the tree

by restricting max tree depth. Also, to explore the power of ensemble methods, we tried random forest as well. From the table of result we can see that trees outperformed linear models. This makes sense as usual ensemble methods predicts better than a single model.

Gradient boosted regression (GBR) builds an additive model in a forward stage-wise fashion; it allows for the optimization of arbitrary differentiable loss functions. In each stage a regression tree is fit on the negative gradient of the given loss function.

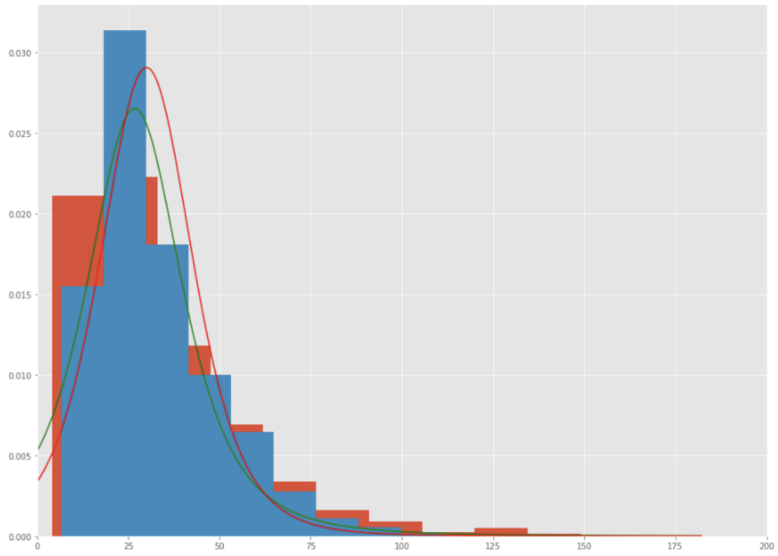

Figure 8: Prediction vs true values

Model	Training MSE	Test MSE	R-squared (test)	Explained Variance score (test)	Training MAE	Test MAE
Least Squares	285.6423	278.2928	0.4284	0.4284	12.0193	11.9451
Huber Regression	378.8702	366.5700	0.2471	0.2702	12.8395	12.7322
Lasso Regression (α=0.001)	-	-	0.4224	-	-	-
Ridge Regression (α=20)	-	-	0.4224	-	-	-
Decision Tree (max_depth=9)	240.8223	229.7826	0.4779	0.4779	10.6419	10.9321
Random Forest (max_depth=9)	211.7693	229.7826	0.5281	0.5281	10.0530	10.4071
GBR Regression (max_depth=9, loss=ls)	191.9242	200.2878	0.5886	0.5886	9.4059	9.6091
GBR Regression (max_depth=5, loss = huber)	219.7524	219.3468	0.5495	0.5539	0.6368	9.7301
GBR Regression (max_depth=5, loss = quantile)	543.5930	552.7905	-0.1354	0.4486	19.1595	19.3692

Figure 9: Model Results

We used t-distribution to simulate both the real data price and predicted price for all models we applied, and we find that all the graphs are similar and look like the one on the left. From the graph, the predicted price distribution has fatter tails and lower peak, which means it has less probability in 15−35 and much larger probability less than $15

and over $75.The reason is that tails consist of less than 10% of data and we did not view it as outliers. Therefore we could explore further about the parts where wines' prices are larger than $75 or less than $15 and redo the preprocessing of the data. It could potentially decrease the error significantly.

6 Discussion

Decision trees are very interpretable to humans. Considering the application of the scenario where consumers might probably want to know why the wine should be priced as the model says, and where retailers may want to look into the reason why they should price wine according to the model, decision tree is a very suitable solution for the problem. Though Random Forest has better performance, its interoperability is not as good as Decision Trees.

- Is it WMD?

Since this model is all about pricing wines and no human-related factors are used, this is very unlikely to become a Weapon of Math Destruction. The biggest impact would be that the price is set too low or too high. For manufacturers, they will use the result as a reference to help them price the wine, but they will definitely manually check the price to make sure it is at least higher than cost and not too high to be sold. For consumers, the biggest problem caused might be spending money on a bottle of wine or missing an adequately priced bottle of wine. Those are far from destruction.

- Fairness

In terms of fairness, problems exists like that the model may have "stereotype" of wines in some regions, which affects people's opinion on them. To solve this, we can make sure that we have enough samples for each region, especially for each price range, to make sure the model could predict accordingly. However this is not too big an issue. Companies can always use marketing to change consumers' opinions, and merchandises are more than the features this model included.

- Application

I think the model is very promising in real world applications. Wines are becoming more common in ordinary people's lives, and consumers are becoming more and more rational as information is more approachable. If we were running a company, we would like to adopt the model as a reference when price wines, and we will definitely use the tool to see whether a wine is worth its price tag before we buy it.

References

1 Hash Encoding https://towardsdatascience.com/understanding-feature-engineering-part-2-categorical-data-f54324193e63

2 TF-IDF https://monkeylearn.com/blog/what-is-tf-idf/

3 Google Universal Sentence Encoder https://colab.research.google.com/github/tensorflow/hub/blob/master/examples/colab/semantic_similarity_with_tf_hub_universal_encoder_lite.ipynb#scrollTo=pSkjuGYoCBfU

4 Google Universal Sentence Encoder https://tfhub.dev/google/universal-sentence-encoder/2