Learning the Language of Wine

CS 229 Term Project - Final Report

Category: Natural Language
Team Members: Aaron Effron (aeffron), Alyssa Ferris (acferris), David Tagliamonti (dtag)

1 Introduction & Motivation

Wine has been an integral element of human society for millennia. Accounts of wine date as far back as Egyptian and Roman times and wine often has symbolic significance for certain religions. In modern times, wine is often the beverage of choice for social gatherings and celebrations.

Unfortunately, despite its historical and contemporary significance, wine can often seem daunting and intimidating to many due to the oft use of strange and esoteric descriptors by wine professionals, or *sommeliers* (from French: "wine steward"). Thus, we are motivated to ask the question: can machine learning help demystify wine and make it more accessible?

2 Task Definition

Our broad task is to "learn the language of wine." More concretely however, our aim is to build models that take as input the description of a wine by a wine expert and output:

1. Grape type: red or white (binary classification)
2. Grape variety (multi-class classification)
3. Similar wines (recommendation)

3 Data and Features

3.1 Dataset

Our data source is a Kaggle Featured Dataset, which has not been used for a competition [1]. It contains a set of just over 150,000 wine reviews scraped from the Wine Enthusiast magazine website. An example corresponds to a single (approx. 15-35 word) review of a unique wine, along with characterizing information such as region of origin, grape variety, price, and rating. A sample review is shown below:

This is the Hearst family's run at a luxury-level Chardonnay, and it's a promising start, with creamy aromas of honeysuckle, light butter and citrus rinds. There's a touch of vanilla bean on the palate, which also shows apple ice cream, flashes of cinnamon and a firmly wound texture. It's delicate, clean, light and quaffable.

We see that the word 'Chardonnay' appears in the text. As one of our tasks is to predict grape variety, before running our models, we censored the reviews to remove any variety names from the descriptions, including misspellings and abbreviations.

Additionally, because our original dataset has over 600 unique grape varieties (classes in task 2), many of which only appear a few dozen times in the dataset, we decided to filter our dataset to only include the grape varieties with at least 1,000 reviews. This reduces the complexity of the modeling task by reducing the number of classes and ensures that there is enough training data for each class. We then further removed reviews of wines that are blends, as these correspond to multiple grape varieties, and we removed Rosés, because these are not strictly red or white wines.

Filtering our dataset in this way left us with just under 100,000 reviews and 24 unique grape varieties. With this filtered dataset, we manually mapped grape varieties to grape color (e.g. Cabernet Sauvignon is 'red', whereas Sauvignon Blanc is 'white') and noted a 2/3 to 1/3 split of red vs. white wines.

Subsequent references to our dataset refer to this censored, filtered dataset, augmented with grape color.

3.2 Word and Character n-gram Features

After filtering and censoring our dataset, we used two main feature types for classification:

- **Word Features**: Feature for every word in the reviews, after censoring removal of stop words.
- **Character n-gram Features**: Feature for every n-gram (sequence of n characters) in the reviews, after censoring removal of stop words.

3.3 word2vec Features

We condensed the space of all words in all reviews to a 400-dimensional vector, using context within a window of 5 for each word. This is done via word embeddings with a skip-gram model [2].

For each example, we maximize

$J_{NEG} = \log Q_\theta(D=1|w_t,h) + k \underset{\widetilde{w}\sim P_{noise}}{\mathbb{E}} [\log Q_\theta(D=0|\widetilde{w},h)]$

where $Q_\theta(D=1|w,h)$ is the model's binary logistic regression probability of seeing the word w in the context h in the dataset D, calculated in terms of our learned embedding vectors θ. In practice, this expectation is estimated through drawing k contrastive words from the noise distribution [2]. We use a library implementation of word2vec.

4 Supervised Learning

4.1 Methods

We used the following five models with word/character n-gram features and word2vec embeddings of the wine reviews:

1. Decision Tree
2. Random Forest
3. Naive Bayes
4. Logistic Regression
5. SVM

Decision Tree and Random Forest: Decision Tree is a supervised learning algorithm that learns rules to split an initially agglomerated dataset. The algorithm tries to minimize impurity at the nodes after each split according to the equation $G(Q,\theta) = \frac{n_{left}}{N_m} H(Q_{left}(\theta)) + \frac{n_{right}}{N_m} H(Q_{right}(\theta))$ and minimizing the θ parameter. We used the Gini measure of impurity $H(X_m) = \sum_k p_{mk}(1-p_{mk})$ where p_{mk} is the proportion of class k points in node m. Random forest is a variant where multiple decision trees are constructed and then combined using bootstrap aggregation to form a single consensus tree [3].

Naive Bayes: For each example, we simply classify as $\hat{y} = \underset{y}{\text{argmax}} P(y) \prod_{i=1}^{n} P(x_i|y)$, where we use MAP estimation to estimate $P(y)$ and $P(x_i|y)$ [4].

Logistic Regression: We use the cross-entropy loss with L2 regularization [9], and used weighting to emphasize examples from underrepresented classes. Our weighted and regularized logistic (softmax) objective is

$J(\theta) = -\sum_{i=1}^{m} w^{(i)} \sum_{k=1}^{K} {y_k}^{(i)} \log \hat{y}_k^{(i)} + \lambda \sum_{r=1}^{R} \|\theta_r\|_2^2$

SVM We use a multi-class SVM with a "one-vs-rest" approach, choosing the class with the greatest margin.

$\min_{w,b,\zeta} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \zeta_i$

Subject to $y_i(w^T \phi(x_i) + b) \geq 1 - \zeta_i, \zeta_i \geq 0, i = 1, ..., n$ [5]

4.2 Results & Discussion

We summarize results from running our models on the 100,000 examples from our dataset, using a 60/20/20 train/dev/test split.

When using single word features ($n = 1$) for logistic regression to classify between red and white wines (i.e. Task 1), we found the top 5 significant words for predicting red were:

tannins cherry berry blackberry strawberry

And, the top 5 words for predicting white instead of red were:

yellow tropical pear pineapple apple

This is very reassuring because we know red wines are typically described in terms of red fruits and berries whereas white wines are described in terms of citruisy fruits.

In multi-class classification using logistic regression, we found that the top 5 significant words in predicting Sauvignon Blanc were:

gooseberry grass herbaceous herbaceousness fig

In contrast, the top 5 significant words in classifying against Sauvignon Blanc were:

tannins cherry berry offdry blackberry

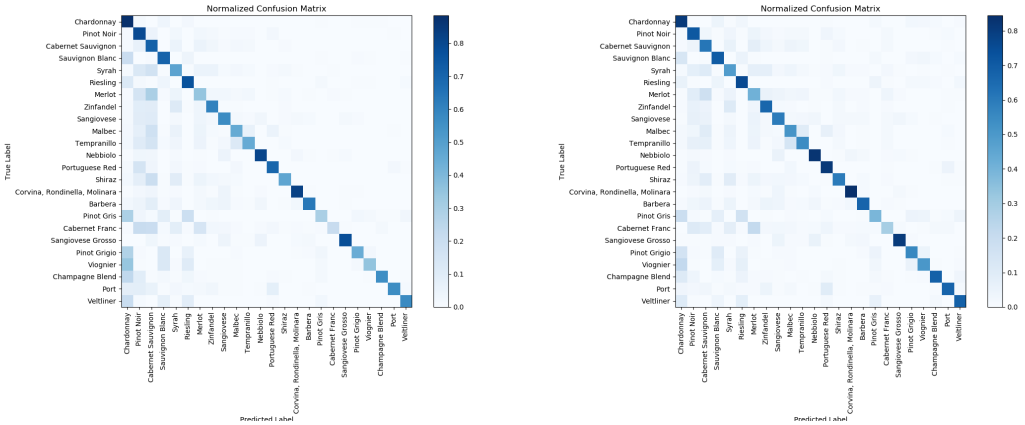
Note that 4 of the 5 words above correspond to the top 5 words when predicting red vs. white wine using logistic regression. Because Sauvignon Blanc is a white wine, it seems intuitive that words typically associated with red wines tend to suggest a review is not of a Sauvingon Blanc wine (i.e. they have negative values in the θ vector for the Sauvignon Blanc class).

Figure 1 lists the obtained classification accuracy across various models and features. Word Features(1) refers to using single word features, Char Features(5) refers to using character 5-grams as features, and word2Vec Features refers to mapping each sentence to a 400-dimensional vector, built from the skip-gram model with context window 5. The *baseline* accuracy refers to the strategy of predicting the majority class (in the training set) for all examples. Our best model for Task 1 achieves a 99% test accuracy and our best model for Task 2 achieves a 76% accuracy. Given the strong performance for Task 1, we focus our remaining discussion on Task 2.

Task 1: Red vs. White	Word Features (1)			Char Features (5)			word2Vec Features		
	Train	**Dev**	**Test**	**Train**	**Dev**	**Test**	**Train**	**Dev**	**Test**
Naïve Bayes	0.750	0.688	0.685		N/A			N/A	
SVM	0.988	0.980	0.982	0.999	0.982	**0.987**	0.986	0.976	0.977
Decision Trees	0.994	0.972	0.970	0.997	0.971	0.972	0.994	0.958	0.959
Random Forest	0.998	0.983	0.983	0.999	0.984	0.983	0.998	0.981	**0.981**
Logistic Regression	0.997	0.984	**0.984**	1.000	0.986	0.986	0.976	0.974	0.976
Baseline	0.663	0.663	0.665						
Human Benchmark			0.945						

Task 2: Grape Variety	Word Features (1)			Char Features (5)			word2Vec Features		
	Train	**Dev**	**Test**	**Train**	**Dev**	**Test**	**Train**	**Dev**	**Test**
Naïve Bayes	0.670	0.456	0.448		N/A			N/A	
SVM	0.760	0.659	0.660	0.993	0.760	**0.764**	0.553	0.537	0.548
Decision Trees	0.847	0.578	0.582	0.864	0.590	0.593	0.831	0.518	0.515
Random Forest	0.990	0.721	**0.723**	0.996	0.709	0.711	0.988	0.697	**0.699**
Logistic Regression	0.883	0.688	0.697	1.000	0.762	0.761	0.512	0.490	0.499
Baseline	0.151	0.146	0.151						
Human Benchmark			0.260						

Figure 1: Model performance, as measured by classification accuracy



(a) Without Example Weighting (b) With Example Weighting

Figure 2: Confusion Matrix for Logistic Regression

We see that the train accuracy is consistently high, suggesting our features are capable of capturing the structure of the data. Furthermore, it is also promising that Naive Bayes is outperformed by other models, as this means there are more subtle elements to the structure of the data being discovered. Lastly, it is sensible that the training accuracy for Char Features (5) is (generally) even higher, as there are many more features when extracting character n-grams

vs. single word features.

This being said, for every model, the Dev accuracy is considerably lower than the Train accuracy, suggesting there is overfitting present. As we trained with increasing fractions of the dataset, the accuracy continued to improve suggesting that more data would be the most straightforward way to increase the prediction accuracy. However, with word2vec features, a less complex model which ideally should decrease variance, the dev accuracy does not improve.

Common ways to regularize decision tree and random forest models are to limit the maximum number of tree leaves and increase the number of trees used for random forest. We found that the only parameter that had a significant effect was changing the number of trees used in the random forest, and this improvement plateaued after using more than 20 trees.

The confusion matrix in Figure 2 illustrates our motivation for using weighted logistic regression. In particular, without weighting, the model is biased towards predicting the more common classes, i.e. Chardonnay, Pinot Noir, etc. (class labels appear in decreasing order of frequency). By weighting each example by the inverse frequency of the class to which it belongs, we achieve better results. This can be visualized in panel (b) which has a generally darker diagonal. The result is better classification accuracy.

One explanation for why it is very difficult to achieve higher accuracy in this type of classification task is because some grapes are extremely similar. For instance, the Shiraz grape is a near clone of Syrah, differentiated only by the region in which the grape is grown. There tends to be only a small difference in taste profiles. In Figure 2 we see that many errors where the true label is Shiraz have a prediction of Syrah. We note that even in the Court of Master Sommeliers' Master-level examination, the highest distinction for sommeliers, candidates must only achieve a 75% score in the tasting portion of the exam, where they must predict a wine's grape variety, among other qualities [8].

To give a better sense of where our model performance stands, we also compared it to a human benchmark. In particular, one of the authors and an amateur wine enthusiast [7] tried to predict grape color and variety from a random subset of 100 examples drawn from our test set. The results were averaged and are included in Figure 1. The performance on Task 1 was strong. And, although the performance on Task 2 exceeded our baseline, it was not as strong as any of our models.

5 Wine Recommendations

In this section, we discuss an approach to making wine recommendations through unsupervised learning. Specifically, given as input the description of a wine, we can use our word2vec feature model to recommend similar wines. To do so, we followed the following sequence of steps

1. Train the word2vec model (as described in 3.3)
2. Map each description to a vector by summing its component word vectors
3. Compute the vector representation of the input description and calculate cosine similarity (a measure of closeness of two vectors [6]) with all other description vectors. Cosine similarity between vectors A and B is calculated as the cosine of the angle between them:
   $Sim(A, B) = cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|}$
4. Choose the wine whose description vector has the highest similarity score as the recommendation (i.e. the model output)

As an example of the model in action, the following wines were paired:

- Input: *A lush, sexy wine, this is the next step up the J. Lohr hierarchy from the Seven Oaks line. Smoky, herbal notes on the nose accent the ripe cassis fruit, ending in an avalanche of soft, velvety tannins.*
- Output *Aromas of lush black cherries, coffee grinds and darkly toasted oak meld effortlessly in winemaker Roman Roth's powerful Long Island Merlot. After 21 months in French oak, the wine is smooth as silk yet intensely focused with a rich fruit palate accented by hints of forest floor and bramble, velvety soft tannins and a balanced astringency. Lovely all around.*

These appear to be a good match, as both mention soft, velvety tannins as well as a rich fruit flavor and smooth character. This approach can be generalized to find the top K wines given an input description.

6 Understanding Grape Varieties through Unsupervised Learning

"Which wines do you like?", asked the sommelier.

Perhaps one of the most daunting aspects of wine is the sheer number of distinct grape varieties. Understanding relationships between grape varieties is key for a sommelier to make appropriate recommendations. This understanding comes with many years of training. We seek to illuminate these similarities and differences with machine learning, so that the average wine consumer can get a better sense of where their taste profile lies.

To do so, we made use of our word2vec features to embed each wine review in $\mathbb{R}^n$ space by summing over the word vectors in that description, where we used $n = 400$. From here, we use PCA to reduce the dimensionality of each example to k dimensions, where we used $k = 2$. We then group and average the individual example vectors by grape variety, giving a $\mathbb{R}^k$ vector for each grape variety. We can then visualize the relative distances of this reduced dimension representation of grape varieties in the plane. This is shown in Figure 3. The intuition behind this approach is to use word2vec features to capture the semantics of words used in descriptions, then to use PCA to reduce the dimension of our data and allow for easy visualization. The averaging across grape varieties allows us to observe the prominent characteristics of a particular variety by averaging-out any idiosyncrasies in the description of a single wine.

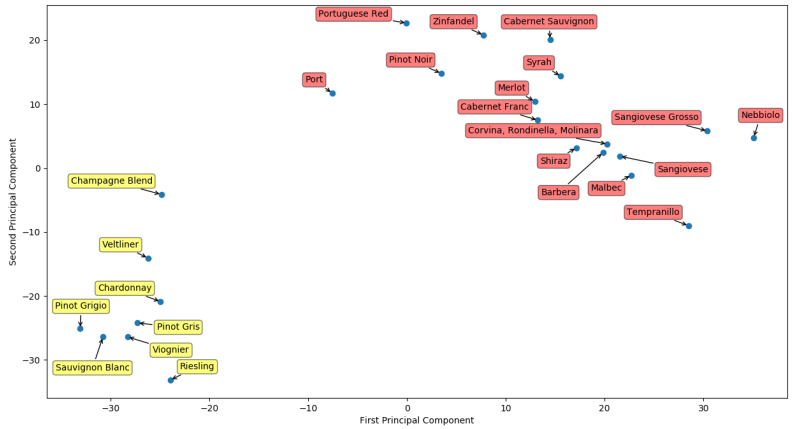


Figure 3: Visualizing Grape Variety Taste Profiles

Intuitively, the relative distance of points in Figure 3 measures the degree of textual similarity between reviews of different grape varieties. The clear separation of reds and whites implies that our model has learned to differentiate between descriptions of reds and whites, which explains our high accuracies in the task of predicting red vs. white.

There are other interesting features to extract from this plot [7]. We can interpret the vertical axis as measuring the fruit aromas a wine evokes, where higher values suggest aromas of black and red fruit, neutral values suggest non-fruit aromas, and negative values suggest citrusy fruits. Indeed, high on the plot we find Cabernet Sauvignon which is often said to evoke aromas of blackberry and cassis. Near the middle we find Tempranillo, which is often cited as having flavors of vanilla and tobacco. Near the bottom, we have Riesling, which evokes aromas of lime and green apples.

Moreover, the horizontal axis captures sweetness/bitterness, with centrally-localed grapes being sweeter and those on the extremes being more bitter. Indeed, Nebbiolo, which is to the far right, is known for being a particularly bitter grape. Similarly, Pinot Grigio, known as a relatively bitter white wine, is located to the far left. Furthermore, the reds appear to be located on a downward sloping diagonal. In the framework of our axis interpretations, this means that a sweeter wine (further left) evokes more fruit (higher up) and that more bitter wines (further right) evoke less fruit aromas (further down) - a perfectly natural conclusion.

7 Conclusion

We have used techniques from natural language processing to shed light on language usage in describing wines. In particular, we used a dataset of wine reviews to build models that predict grape color and variety. Moreover, we used unsupervised learning to build wine recommendation systems. Figure 3 presents some our most exciting and practical results, where we have used unsupervised learning to map the relationship between different grape varieties. This plot can help a wine novice quickly locate their taste profile given a small sample of wines she enjoys and allows for exploration of new varieties.

Acknowledgements

We would like to thank the CS 229 teaching staff for their helpful advice and guidelines, both in lecture, and during project office hours, in helping shape our project objectives and contributing to its success. Additionally, we are ever grateful to the key insights and domain-specific knowledge offered by Patrick Solmundson, Actuarial Mathematics student at the University of Manitoba, Canada, and self-proclaimed *wine geek*. His insights have helped shed light on our results, particularly in endowing our unsupervised characterization of grape varieties with a meaningful interpretation, and illuminating the inherent challenges in predicting grape varieties.

References

[1] Kaggle Featured Dataset. (2017). Wine Reviews [Data file]. Retrieved from: https://www.kaggle.com/zynicide/wine-reviews

[2] TensorFlow Developers. "Vector Representations of Words." 2 Nov. 2017. Retrieved from: https://www.tensorflow.org/tutorials/word2vec

[3] Scikit-learn Developers. "1.10. Decision Trees." scikit-Learn 0.19.1 Documentation, 2017. Retrieved from: scikit-learn.org/stable/modules/tree.html

[4] Scikit-learn Developers. "1.9. Naive Bayes." scikit-Learn 0.19.1 Documentation, 2017. Retrieved from: scikit-learn.org/stable/modules/naive_bayes.html

[5] Scikit-learn Developers. "1.4. Support Vector Machines." scikit-Learn 0.19.1 Documentation, 2017. Retrieved from: scikit-learn.org/stable/modules/svm.html.

[6] Singhal, Amit. "Modern information retrieval: A brief overview." IEEE Data Eng. Bull. 24.4 (2001): 35-43.

[7] Solmundson, Patrick. Personal Interview. 13 Dec. 2017.

[8] Court of Master Sommeliers. "Master Sommelier Diploma Examination". Retrieved from: https://www.mastersommeliers.org/courses/master-sommelier-diploma-examination

[9] CS 229 Problem Set 4, Question 1.

Team Member Contributions

The following table highlights the contributions of each team member:

Team Member	**Contributions**
aeffron	Exploratory Data Analysis Computing TDIF scores and key words to determine suitability of data for our tasks Naive Bayes model results SVM model results word2vec implementation Drafting and review of final paper
acferris	Exploratory Data Analysis Word feature extractor Decision trees model results Random forest model results k-means implementation Drafting and review of final paper
dtag	Exploratory Data Analysis Character feature extractor Common code base to produce sparse design matrices from sparse feature vectors Logistic regression model results Implementation of cosine similarity recommendations Implementation of unsupervised classification of grape varieties Drafting and review of final paper