# Predicting Wine Variety With Multi-Class Text Classification

**Jing Ren**

## Abstract

This project aims to predict wine variety with multi-class text classification based on tasters' reviews. The main techniques implemented include Logistic Regression, Support Vector Machine (SVM), Fasttext, and Convolutional Neural Network (CNN), which will be discussed in this paper along with their experiments and results. All source code could be found in GitHub via https://github.com/MSIA/jrb1992_msia_text_analytics_2020/tree/project.

## 1 Introduction

Nowadays, learning to taste wine is no different than learning to appreciate art or music. It is a multisensory activity that explores aromas and flavors, which requires good practices. In order to judge wine quality, you need exposure to different kinds of wines. While such events are not likely to happen during COVID, wine tasters' reviews could be reflective upon varieties of wines and give beginners a general sense of how to identify wines through blind tasting, and that is the purpose of this project – using NLP models to predict the wine variety with words in the reviews.

## 2 Related Work

Besides regression analysis, the multi-class classification is one of the most common machine learning tasks in NLP, with the rising need for handling the surge in clinical and legal data analytics. In order to improve efficiency and reduce the cost of document review, many researchers have investigated techniques best suited for multi-class classification.

SVM is proved to be able to enable automation in various fields due to the advancements of computational capabilities and machine learning techniques (Boutaba, Salahuddin, & Limam, 2018). Duan and Keerthi (2005) propose that SVM are normally implemented through combining multiple two-class SVMs, and two widely used techniques include the one-versus-all and one-versus-one methods.

Fasttext introduced by Facebook is considered an emerging technique for approximate text searching (Wu & Manber, 1992). While errors can result from misspelling and approximate patterns, Fasttext allows errors when searching for a pattern in text files with flexibility and efficiency.

In recent years, more scholars start to focus on Convolutional Neural Networks for sentence classification, which is trained on pre-trained word vectors for sentence-level classification task (Kim, 2014). Kim proposes that a comparatively simple CNN along with hyperparameter tuning and static vectors could yield great performance on multiple benchmarks.

Based on these previous research, this project's goal is to conduct experiments on various popular methods and evaluate their performance.

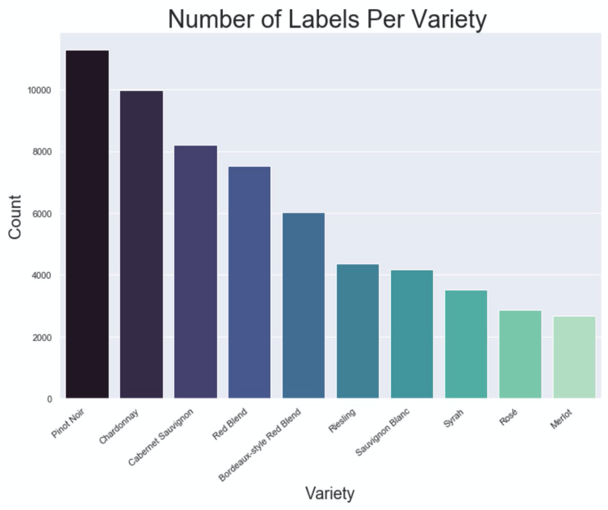
## 3 Dataset

This dataset contains 129971 reviews, providing 13 features such as wine's origin country, points, price, and taster's reviews/description, etc. The goal is to predict wine variety (e.g. Bordeaux-style Red Blend) based on the reviews.

In order to perform data cleaning, duplicated values are removed, and missing values are handled by data imputation. Specific regions regarding wine growing area are dropped, since province already provides enough geographical information, while having much fewer missing values. Tasters' names and twitter handle are also removed by assuming the objectivity of scoring and reviews. In addition, we will not further collect data through Twitter API. Rows with null values in remaining columns are eliminated. Prices with null values are replaced with the mean.

Since there are hundreds of labels for varieties, top 10 varieties with the most occurrences are chosen as the final labels. The summary statistics for the cleaned dataset indicates that there are 60612 reviews left and the average word length of reviews is around 47.72.

The top 10 varieties distribution is shown below:



The dataset can be downloaded here: https://www.kaggle.com/zynicide/wine-reviews.

## 4 Method

### 4.1 Feature Vectorization

To prepare data for model building, the first step is to apply NLTK word tokenization, convert letters to lower cases, remove non-alphabetic chars and stop words that do not add much meaning to a sentence. The next step is to apply TfidfVectorizer to transform text into feature vectors that can be used as input to estimator. Note that the sublinear tf scaling is chosen to replace tf with 1 + log(tf), addressing the problem that 20 occurrences of a word are probably not 20 times more important than 1 occurrence. Setting max_df = 0.90 ignores terms that appear in more than 90% of the documents. Setting min_df = 50 ignores terms that appear in less than 50 documents. By building TfidfVectorizer for both 1-gram and 1-gram+2-gram, the impact of varying bag-of-words representation will be tested after running sets of experiments.

### 4.2 Logistic Regression

Logistic regression is normally chosen as the benchmark performance to compare with other models. In addition, logistic regression models require relatively less training time and computational power. By specifying option ‘ovr’ for parameter multi_class, a binary problem is fit for each label so as to build multi-class classification.

While testing out different sets of bag-of-words representation and the inverse of regularization strength (C), four experiments are conducted to optimize model performance.

| | F1-score | Accuracy |
|---|---|---|
| 1gram, C=1 | 0.7428 | 0.7700 |
| 1gram, C=0.5 | 0.7351 | 0.7649 |
| 1gram+2gram, C=1 | 0.7457 | 0.7731 |
| 1gram+2gram, C=0.5 | 0.7339 | 0.7652 |

Table 1: Logistic Regression Experiments

As shown above, 1-gram+2-gram model with C = 1 has the best performance out of four models. Overall, 1-gram+2-gram models have slightly better performance in terms of predictive power, comparing to 1-gram models. As we decrease the inverse of regularization strength (C), smaller values specify stronger regularization. The models with C = 0.5 are expected to have comparatively better performance, but the results are similar in this case.

### 4.3 Support Vector Machine (SVM)

Support Vector Machine (SVM) proves to be robust in many cases when it comes to text classification. It finds the optimal hyperplane that maximizes the width of the margin separating classes. Linear SVC algorithm is chosen because text is often linearly separable, and the linear kernel works often well. There are less parameters to optimize and it is comparably faster to train.

Six sets of experiments are conducted with different bag-of-words representation, the inverse of regularization strength (C), as well as the loss function.

| | F1-score | Accuracy |
|---|---|---|
| 1gram, C=1, loss=hinge | 0.7414 | 0.7680 |
| 1gram, C=1, loss=squared_hinge | 0.7411 | 0.7674 |
| 1gram, C=0.5, loss= squared_hinge | 0.7455 | 0.7712 |
| 1gram+2gram, C=1, loss=hinge | 0.7416 | 0.7688 |
| 1gram+2gram, C=1, loss=squared_hinge | 0.7403 | 0.7661 |
| 1gram+2gram, C=0.5, loss=squared_hinge | 0.7457 | 0.7715 |

Table 2: SVM Experiments

As shown above, 1gram+2gram model with C = 0.5 and loss = squared_hinge yields the best result. The models with C = 0.5 have similar performance comparing to those with C = 1, though higher C value could end up with a smaller margin. By specifying loss function to hinge or squared hinge, it is intended to inspect different loss functions' impact on adjusting weights for iterations, which is not quite obvious in this case.

## 4.4 Fasttext

Fasttext provides more efficient text classification. Building upon Word2Vec, Fasttext is commonly used since it is simple and fast. Parameters tuned are wordNgrams and learning rate. While wordNgrams specifies the max length of word n-gram, learning rate is one the most important hyper-parameters of a model. When learning rate is set too high, it might cause the model to converge very quickly towards a sub-optimal solution; when learning rate is set too low, it could take much longer time for training process and get stuck at certain point.

| | Precision | Recall | F1-score |
|---|---|---|---|
| lr=1.0, wordNgrams=1 | 0.9256 | 0.9256 | 0.9256 |
| lr=1.0, wordNgrams=2 | 0.9370 | 0.9370 | 0.9370 |
| lr=0.5, wordNgrams=2 | 0.9384 | 0.9384 | 0.9384 |
| lr=0.05, wordNgrams=2 | 0.9396 | 0.9396 | 0.9396 |

Table 3: Fasttext Experiments

As shown above, 2-gram model with lr = 0.05 has the best performance out of the four models. As learning rate decreases, the model performance gets better in terms of F1-score.

Note that Fasttext does not have accuracy metric in built-in functions, and therefore only precision, recall, and F1-score are calculated in this case.

## 4.5 Convolutional Neural Network (CNN)

By tuning activation function and batch sizes, the model with dense_activation = relu and batch_size = 32 has the most predictive power. In terms of changing activation, relu has comparably better performance comparing to sigmoid function. As for batch size, smaller batch sizes normally lead to a smaller number of iterations for training and higher accuracy, which proves to work better on this dataset. The results for four experiments are shown below.

| | F1-score | Accuracy |
|---|---|---|
| dense_activation=relu, batch_size=128 | 0.7691 | 0.7687 |
| dense_activation=sigmoid, batch_size=128 | 0.7333 | 0.7302 |
| dense_activation=relu, batch_size=32 | 0.7727 | 0.7797 |
| dense_activation=sigmoid, batch_size=32 | 0.7733 | 0.7723 |

Table 4: CNN Experiments

## 5 Results

For each method listed, the best performance model with its corresponding parameter setting is recorded in table below, with accuracy and F1-score. Accuracy is the measure of all correctly classified cases, while F1-score is the harmonic mean of precision and recall, which gives better measure of incorrectly labeled cases. Besides, F1-score works better when there are imbalanced classes.

| | Parameter Setting | F1-score | Accuracy |
|---|---|---|---|
| Logistic Regression | 1gram+2gram, C=1 | 0.7457 | 0.7731 |
| SVM | 1gram+2gram, C=0.5, loss= squared_hinge | 0.7457 | 0.7715 |
| Fasttext | lr=0.05, wordNgrams=2 | 0.9396 | 0.9396 |
| CNN | dense_activation =relu, batch_size=32 | 0.7727 | 0.7797 |

Table 5: Final Results

## 6 Discussion

As shown in previous section, Fasttext model with lr = 0.05 and wordNgrams = 2 yields the best performance out of all methods. CNN model performs slightly worse, while logistic regression and SVM followed next. This result is considered reasonable based on past research works, considering the outstanding performance of Fasttext methodology. However, CNN is expected to perform better than logistic regression and SVM, which is not quite significant in this case. In order to further tune parameters, changing the number of hidden layers and units and using Grid search to exhaustively search all possible parameter combinations could be implemented given sufficient computing power.

## References

Boutaba, R., Salahuddin, M.A., Limam, N. et al. A comprehensive survey on machine learning for networking: evolution, applications and research opportunities. *J Internet Serv Appl* 9, 16 (2018). https://doi.org/10.1186/s13174-018-0087-2

Duan KB., Keerthi S.S. (2005) Which Is the Best Multiclass SVM Method? An Empirical Study. In: Oza N.C., Polikar R., Kittler J., Roli F. (eds) Multiple Classifier Systems. MCS 2005. Lecture Notes in Computer Science, vol 3541. Springer, Berlin, Heidelberg. https://doi.org/10.1007/11494683_28

Sun Wu and Udi Manber. 1992. Fast text searching: allowing errors. Commun. ACM 35, 10 (Oct. 1992), 83–91. DOI:https://doi.org/10.1145/135239.135244

Kim, Y. (2014) Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882.