Wine Sentiment Classification

Roger Tanuatmadja
PID A53044163

ABSTRACT

Product review sites are ubiquitous nowadays. Often times, when a reviewer writes a review for a product, he/she would specify both textual description and a review score that would serve to summarize his/her overall sentiment towards the product. In this paper we will describe a simple model that will predict the likely sentiment bucket a wine product will fall under based on the review text. In other words the predictive task will be a multi class classification task based on sentiment analysis.

1. INTRODUCTION

The data set that will be used to build the model originates from the CellarTracker website and can be found at `http://snap.stanford.edu/data/cellartracker.txt.gz`. The data set consists of 2025995 reviews, where each review is a semistructured document consisting of wine name, unique identifier, variant, year, review score, review time, unique user id, user name, and review text. Of the whole data set, around a quarter (456340) have no associated review score. The model that is described in this paper then can ultimately be used to fill in the range of review scores those reviews will likely fall under.

The selection of model, range of review scores and features are based on both explanatory analysis of the data as well as existing literature. In this section, we will show some basic trends related to the data before delving to existing literature in Section 3. In Section 4, we will identify the features that are relevant to the prediction task as well as discuss the preprocessing steps to extract those features. In Section 5, the model and other alternatives will be discussed in greater detail and finally in Section 6, we will describe the results of our experiments and our conclusions.

Number of unique users	44268
Number of distinct wines	485179
Max Review Score	100.0
Min Review Score	50.0
Average Review Score	88.82
Number of Variants	830
Range of Wine Year	1720 - 2012

Table 1: Basic statistics on the data set.

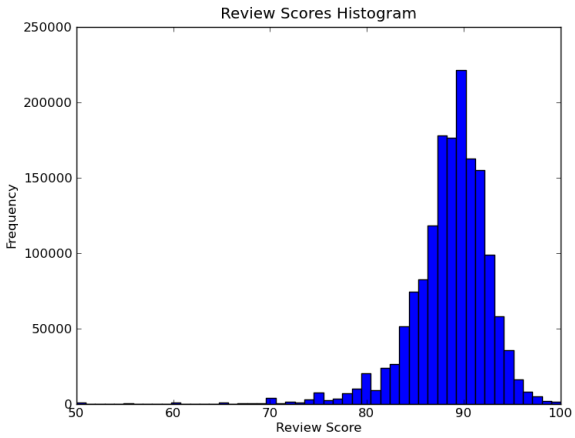

Figure 1:

Based on the average review score in Table 1, we can see that the review scores tend to skew to the higher side, in fact this can be seen from the distribution in Figure 1.

Determining the strength of opinions would be a simple task if a number of opinion words always appear for a certain range of review scores for example, if the word ”perfect” (and all words that share the same stem) always appears in reviews whose score is between 90 and 100. This is however not true in practice. An example is the word ”undrinkable” which does appear in strongly positive reviews for example, a review whose score is 97 has the following review

text: "Fabulous and hard to describe. A massive fruit attack on the palate, not as **undrinkable** as the Mordoree for instance which means already good integration of the tannins and acidity." In other words, for strongly positive reviews, often times, the word "undrinkable" appears in the context of comparison with another wine. The intuition behind sentiment analysis though does assume that certain words are used relatively more often to express a certain sentiment. In Table 2, we show the frequency of some of the more heavily used terms in our corpus, grouped into their respective range of review scores.

	Review Score 80 - 89	Review Score 80 - 89	Review Score 90 - 100
fruit	22.91%	35.88%	40.01%
wine	29.33%	28.60%	34.58%
finish	14.98%	24.38%	29.98%
very	20.25%	25.12%	31.04%
dark	4.50%	11.11%	16.34%
palate	11.91%	19.27%	22.24%
great	2.85%	8.00%	16.45%
not	41.79%	38.19%	33.52%
but	29.87%	36.56%	32.98%
good	11.42%	23.73%	19.20%

Table 2: The percentage numbers are calculated by the number of occurrences of the word in a particular review score range divided by the total number of reviews in the range.

The words in Table 2 are basically the next tier of most common words found in the corpus other than words such as "and", "the", "of", "with", "this", "to","in", etc that are very common across buckets. Unlike the really common words, these words seem to have a more predictive value for example: we expect the word "great" to be more frequently associated with the upper level review scores as opposed to "good".

2. PREDICTIVE TASK

The CellarTracker website caters to wine collectors. We believe it is reasonable to assume that users of the website are highly knowledgeable about wine and they are very attuned to their own taste, which may explain the relatively constrained distribution of review scores i.e. these people are highly unlikely to purchase "bad" wine and the wines they purchase tend to be either good or very good.

Based on the frequency and distribution of review scores as well as our assumption above, our predictive task would be to use the review text to predict whether a certain review belongs to one of the Sentiment Category Buckets in Table 3.

To evaluate the accuracy of our model we will rely mainly on simple precision, which is calculated as the number of reviews that the algorithm classifies correctly divided by the total number of reviews in the test set. From our data set, the reviews classified as Very Positive is the most common among all the reviews so we can simply get a precision of 0.4870 by blindly labeling each review as Very Positive. In our classification task we will then use 0.4870 as our baseline for precision.

Sentiment Category Bucket	Review Score	Frequency
Very Positive	90 - 100	48.70%
Positive	80 - 89	48.44%
Average and Below	Below 80	2.86%

Table 3: The Frequency column serves to complement Figure 1 i.e. it is the percentages of the data set that belongs to a particular Sentiment Category Bucket

3. RELATED WORK

The goal of sentiment analysis is to classify users' reviews based on their sentiment. This section describes a number of works that are relevant to our goal.

Most early works on sentiment classification were focused on movie reviews. In particular the task was a binary classification task whereby a review is classified as having a positive or negative orientation. Turney (2002) describes an algorithm which computes sentiment orientation of phrases in the review where a phrase is considered positive if it is strongly associated with the word "excellent" and negative if it is strongly associated with the word "poor". The overall sentiment orientation of the review is then determined by the average sentiment orientation of the phrases.

Another often quoted work by Pang et al (2002), tried to determine whether it is appropriate to apply machine learning techniques towards sentiment analysis. In particular Pang et al evaluated 3 algorithms i.e. Naive Bayes, Maximum Entropy, and Support Vector Machine (SVM) with the main finding being the difficulty of classifying movie reviews due to various reasons as well the SVM approach performing better than the other two approaches.

Pang and Lee (2005) then followed up on the work above by proposing to classify reviews into finer-grained rating scales using a multi-class sentiment classifier such as the SVM One vs All Classifier.

And finally, an interesting work done by Fang and Chen (2011) tried to incorporate domain specific lexicon as a possible way to further improve sentiment classification. In the paper, the authors described their own algorithm for generating domain specific lexicon pertaining to camera reviews and using the result as another input in SVM classification. The insight behind the work is that the sentiment of specific words or phrases is context dependent for example

the word "long" has a positive sentiment orientation when it comes to battery life but not when it comes to "Shutter Lag". Fang and Chen found that incorporating lexicon information "can significantly improve the accuracy for fine-grained sentiment analysis tasks".

After reading many of the review texts in the data set, and continuing on our assumption that the reviewers are highly knowledgeable about the subject we notice the reviews are often expressed in a very specialized lexicon for example, a review whose score is 96 has the following accompanying text: "Olive, horse sweat, dirty saddle, and smoke..... lovely, loaded with tapenade, leather, dry and powerful, very black olive, meaty. A terrific bottle..." Someone who is not knowledgeable about wine like us would totally miss the very positive sentiment of the review had the author elected to omit words like "terrific", "lovely", and "powerful". At a first glance then, the work by Fang and Chen seems to be something that we can incorporate into our model. Unfortunately however, their algorithm for building domain specific lexicon is non trivial to implement involving the use of linguistic patterns to perform a query on a search engine among other things.

In our modelling, we will then stick to an SVM approach which appears to be the most widely accepted state of the art approach towards sentiment classification. However as part of Section 5 we will also present the result of training the data utilizing Naive Bayes to provide another comparison against the SVM approach.

4. FEATURE SELECTION

In existing sentiment analysis literature, it is very common to represent a piece of text as a feature vector whereby the entries correspond to individual terms. In the case of a corpus made up of unigrams and utilizing a binary feature extractor for example, the feature vector would be a vector of values of either 0 or 1 depending on whether words in the corpus can be found in the text.

We have presented as part of Section I the results of our explanatory data analysis where we have found certain words appearing in greater frequency depending on how positive they are towards a certain wine product. And indeed term frequency feature vector normalized through the use of Term Frequency Inverse Document Frequency (TF-IDF) is another common way of representing a text based on existing literature. Term Frequency computes the amount of time a certain term is found in the corpus for reviews in a certain review score range. This however tends to overestimate the importance of extremely common words such as "and", "this", etc. The IDF term is then applied to moderate the Term Frequency by penalizing the former if a certain term is found in many documents. In other words, terms that are frequent in a single or a small group of documents should have a higher TF-IDF value than very common terms.

In their paper, Pang et all found that using presence i.e. the first approach above outperforms the usage of frequency when it comes to sentiment classification of movie reviews, and we have found this to be true in our chosen data set as well. As part of this paper we will utilize both approaches and discuss the results.

4.1 Preprocessing Task

Our choice of feature representation naturally leads to a very high dimensional features space. In fact, our initial experiment to utilize all the words in the review texts of our training examples as the corpus resulted in a training time that is prohibitively long. In fact we did not manage to train the model using Naive Bayes even after running the process for more than 10 hours. We then decided to build a corpus consisting of "n" most common words, with n chosen to be 5000. This n value is chosen based on experiments and utilizing it allows us to filter out most words whose frequency in the training set is 1 for reviews between 80 to 89 and 90 to 100. In fact, the lowest frequency of words for the former is 6 and there is only one word whose frequency is 1 in the later.

The corpus is built by removing words whose length is 1 as well as stripping the following characters '.', ',', ';', '/>', '(', '.<br', "" from words. The later step is required since characters such as periods and commas are often attached to a preceding word for example: "end.". All words are also converted into lower case to ensure that "Excellent" and "excellent" are not represented as 2 different words.

Negation tagging. In natural languages, it is very common to express a negative sentiment by using the word "not" to invert a word that is commonly associated with a positive sentiment for example: "not good". In our experiments we have come up with two different sets of the most common 5000 words, one employing negation tagging and one without. In existing literature, negation tagging is often employed by creating new "terms" by concatenating the token "not_" to words following the word "not" up to a period or a comma. For example, given the following sentence "The wine is not enjoyable by itself.", standard negation tagging would create the following terms: "not_enjoyable", "not_by", and "not_itself.". We have chosen to deviate from existing literature and employ negation tagging only on adjective words. We believe this is a reasonable modification since it more closely resembles natural language usage. And finally, for the the training labels, we simply convert review scores into the appropriate Sentiment Category Bucket according to Table 3.

5. EXPERIMENTS

All Naive Bayes experiments is based on dividing the existing data set into a training (70% of the data) and test set (remaining 30%), while for the SVM related experiments we divided the data into training (50% of the data), validation

(20% of the data) and test set (remaining 30% of the data). Additionally, we have kept the test set the same for both approaches.

5.1 Naive Bayes

The first model that we tried actually used a subset of the features described in the Features section above. From the dictionary of 5000 unigrams that we computed, we used the Python NLTK package to extract all the adjective words. This resulted in a total number of 508 adjectives that serve as our initial features for training a Naive Bayes classifier. The intuition behind this initial selection of feature words is mainly driven by the Turney's work. The accuracy for this particular model is 0.4800, which is negligibly worse than our baseline.

As an aside, as described in the Related Works section, the Turney work did not actually use Naive Bayes for sentiment classification. We had actually tried implementing the Turney algorithm, however the later requires a sentiment dictionary that classifies words as having a positive and negative orientation, and our attempt of utilizing a similar dictionary by Pang, which can be found on the following: `http://www.cs.uic.edu/~liub/FBS/opinion-lexicon-English.rar`, resulted in less than 10% of our adjective words being classified.

Our next attempt was then to utilize the full 5000 unigrams as the feature space of the Naive Bayes model. This however resulted in an accuracy of 0.2397, which is considerably worse than the baseline accuracy.

And finally we concluded our Naive Bayes experiments by utilizing a feature set that employs Negation tagging.

	Precision on Training Set	Precision on Test Set
Adjective only Feature Vector	0.4814	0.4798
5000 most common words Feature Vector	0.2391	0.2397
5000 most common words Feature Vector with Negation Tagging	0.2400	0.2394

Table 4: Given the poor performance of the Naive Bayes model, overfitting is not an issue, but Training Set accuracy is provided for completeness.

5.2 Support Vector Machine

Similar to our first attempt with Naive Bayes, we also began our SVM related experiments by utilizing only the adjectives words extracted from our dictionary of 5000 unigrams. This feature set is then fed into our multiclass SVM Linear Classifier in two ways, with the first being a binary feature vector and the second being a feature vector computed using the TF-IDF score. The best precision we managed to obtain by employing the binary feature vector was 0.63312, while the best precision obtained based on the TF-IDF based feature vector was 0.61419.

We then utilized a slightly different feature vector that utilizes the result of negation tagging as our adjectives list. For this feature vector, the best precision obtained through the binary feature vector was 0.63396, while the best precision we obtained through the TF-IDF based feature vector was 0.61513.

And finally, we move onto our last sets of experiments. We first utilized the 5000 most frequent terms that do not employ negation tagging as our feature set. The best precision that we obtained through the binary feature vector was 0.73093, while the TF-IDF based feature vector gave us the best precision of 0.70256. When using the 5000 most frequent terms that employ negation tagging, the best precision that we obtained through the binary feature vector was 0.73144, while the TF-IDF based feature vector gave us the best precision of 0.70292.

In order to obtain the precision values above, we had to experiment with the value of the C parameter of the Linear SVC classifier that we use and the best C value is typically either 1.0 or 10.0. The results of our experiments are presented in both table 5 and table 6.

As a final note, the SVM experiments were all implemented utilizing the OneVsAllClassifier and LinearSVC that comes with Python's sklearn library.

6. DISCUSSIONS AND CONCLUSIONS

In this paper we explore the use of machine learning techniques to predict sentiment orientation towards wine products utilizing just words in the review text as features. One thing that is interesting is that although most studies were performed on a different domain, the relative performance that we see seems to be applicable at least to the wine domain as well.

In terms of relative performance, based on the wine data set that we have, the best performance is obtained through the use of multi class SVM utilizing a binary term feature vector using the 5000 most common words and negative tagging (precision score of 0.73144) with the worst performance obtained through Naive Bayes. The disadvantages of the later is well known including its assumption of feature independence. In addition to that, performance when compared to using an adjectives only feature set suggests that using 5000 terms (with or without negative tagging) served only to add a lot of noise to the model i.e. the relative frequency of both common and uncommon words in different range of review scores served to distort the prediction process.

Binary Features	Linear SVC C Value	Validation Set Precision	Train Set Precision
5000 most frequent words with Negative tagging	0.5	0.73116	0.74901
5000 most frequent words with Negative tagging	1.0	0.73116	0.74902
5000 most frequent words with Negative tagging	10.0	0.72888	0.74612
5000 most frequent words with Negative tagging	100.0	0.68811	0.70527
5000 most frequent words with No Negative tagging	0.5	0.73111	0.74884
5000 most frequent words with No Negative tagging	1.0	0.73107	0.74885
5000 most frequent words with No Negative tagging	10.0	0.72724	0.74541
5000 most frequent words with No Negative tagging	100.0	0.68498	0.70331
Adjectives only with Negative tagging	0.5	0.63447	0.64230
Adjectives only with Negative tagging	1.0	0.63446	0.64231
Adjectives only with Negative tagging	10.0	0.63446	0.64227
Adjectives only with Negative tagging	100.0	0.62644	0.63202
Adjectives only with No Negative tagging	0.5	0.63318	0.64082
Adjectives only with No Negative tagging	1.0	0.63318	0.64082
Adjectives only with No Negative tagging	10.0	0.63319	0.64085
Adjectives only with No Negative tagging	100.0	0.60948	0.61514

Table 5: The table shows shows both Training and Validation Set accuracy for different features and C value utilizing the Binary Vector

TF IDF Features	Linear SVC C Value	Validation Set Precision	Train Set Precision
5000 most frequent words with Negative tagging	0.5	0.70310	0.72308
5000 most frequent words with Negative tagging	1.0	0.70296	0.72305
5000 most frequent words with Negative tagging	10.0	0.70282	0.72303
5000 most frequent words with Negative tagging	100.0	0.67945	0.70345
5000 most frequent words with No Negative tagging	0.5	0.70285	0.72280
5000 most frequent words with No Negative tagging	1.0	0.70269	0.72279
5000 most frequent words with No Negative tagging	10.0	0.70250	0.72275
5000 most frequent words with No Negative tagging	100.0	0.68559	0.70906
Adjectives only with Negative tagging	0.5	0.61554	0.62472
Adjectives only with Negative tagging	1.0	0.61553	0.62469
Adjectives only with Negative tagging	10.0	0.61551	0.62470
Adjectives only with Negative tagging	100.0	0.60223	0.61172
Adjectives only with No Negative tagging	0.5	0.61425	0.62312
Adjectives only with No Negative tagging	1.0	0.61426	0.62313
Adjectives only with No Negative tagging	10.0	0.61424	0.62313
Adjectives only with No Negative tagging	100.0	0.59092	0.61514

Table 6: The table shows shows both Training and Validation Set accuracy for different features and C value utilizing the TF IDF Vector

The results of the SVM approach shows a clear improvement on our baseline precision of 0.4870. We believe that it partly validates our intuition of certain terms being more closely associated with a certain Sentiment Bucket. However what the results also show is that it is not just the presence of individual terms taken in isolation that determine the outperformance rather it is the presence of a combination of terms, which explains the poor performance of the Naive Bayes i.e. the probabilities of terms in the later model will need to be adjusted upwards/downwards based on the presence/absence of other terms.

And finally, as argued in the Related Work section, there is work that can be done that may improve the precision even more. In fact another possible approach may simply be to try extracting the "features" of the wine and use them as our feature vector, for example given the following review text: "Dark color. The nose shows marzipan, honeysuckle and clove, almost to the point of cloying. The palate is stunningly spicy and lush, quite explosive. The finish is hot, sharp and short.", we maybe able to extract the "feature" values for the wine's color, nose, palate, and finish. These approaches though would need to be tackled in a future work.

7. REFERENCES

1. Pang B, Lee L, Vaithyanathan S. Thumbs up? Sentiment Classification using Machine Learning Techniques. In *Proceedings of EMNLP.* 2002; 79-86

2. Turney P. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In

Proceedings of the ACL 2002;417-424
3. Pang B, Lee L. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the ACL* 2005; 115-124
4. Fang J, Chen B. Incorporating Lexicon Knowledge into SVM Learning to Improve Sentiment Classification. In *Proceedings of the Workshop on Sentiment Analysis where AI meets Psychology (SAAIP)* 2011; 94-100