Twitter Opinion Mining Using Sentiment Analysis

Queen Mary University of London

Lanre Labulo
April 22, 2019

Contents

1 Abstract

This project I will investigated the natural language processing field of sentiment analysis and how it is used to extract opinions from written text. Furthermore, I will examine the machine learning algorithm called Bayes' Naive Classifier and how it's applied in sentiment analysis. Finally, I will apply the Bayes' Naive Classifier to a dataset of tweets predicting the polarity of the tweets; this will be done using R studio.

2 Introduction

Over the past years, sentiment analysis has become a trending topic in the fields of machine learning and natural language processing. The term sentiment analysis implies the automated process of understanding a particular opinion regarding a specific subject from spoken or written language (Liu, 2015). To comprehend given views as far as a certain text is concerned, various tools are employed, including natural language processing, analysis of the text, biometrics, and computational linguistics. These tools are instrumental in the systematic identification, extraction, quantifying, and studying of the affective states and subjective data. Other terms that are usually used in the place of sentiment analysis are emotion AI or opinion mining. In particular, this type of analysis is applied mainly to materials, which represent the voice of customers such as survey responses, reviews, and social media. Such an analysis helps in determining the positivity, negativity or neutralism associated with a given piece of writing (Liu, 2015).

Mostly, Governments and businesses deal with large amounts of data. However, sorting large amounts of data like reviews manually may consume a significant amount of time. Therefore, nearly all governments and businesses employ sentiment analysis to aid in processing various opinions directed to them. To this end, government administrators usually apply sentiment analysis regarding social media to identify and strive to meet the needs of the citizens because such an analysis enables citizens and consumers to influence the delivery of public services, the development of new services, or the identification of better use of the already existing utilities (Perner, 2015). Sentiment analysis made in social networks assists businesses and public administrators to clearly understand the current situation and, thus, having a base for predictions regarding the future. Nevertheless, capturing of the opinions from the media is more reliable and less expensive compared to the research surveys or focus groups.

In businesses, sentiment analysis has shown a significant breakthrough in the revitalisation of a complete business brand (van Zaanen Menno & Dewi, 2017). Successful businesses exploit unstructured data using sentiment analysis to gain actionable insights. Additionally, sentiment data analysis in relation to new and established products helps managers to estimate the retention rate of customers (Alessia, 2015). As for this type of analysis, businesses always have an opportunity to adjust to the current situation in the market and satisfy clients in a better manner.

Generally, a high percentage of the data in the world is neither structured nor organised in any given pre-defined way. Most of the unorganised information comes from text data like in social media, emails, articles, and documents. Therefore, understanding, analysing and sorting through of the text data is usually time-consuming, difficult, and expensive. Hence, sentimental analysis comes in to bring some sense into the much-unorganised text through the automation of business processes, saving the time of processing data manually as well as attaining actionable acumens (Alessia, 2015). In social media monitoring, sentiment analysis is beneficial as it aids in tracking trends over a given period. In turn, keeping an eye on the trends helps in monitoring competition, which is instrumental in designing new strategies. Scalability forms one of the benefits brought

about by sentiment analysis. Markedly, there is much information in the world for manual data processing. Therefore, sentiment analysis confers a benefit to the field of science by allowing the processing of data based on a scale, which is cost-effective and efficient.

Although sentiment analysis is associated with various benefits, it has some limitations, which affects its application. Firstly, a sentiment like all opinions is subjective from one individual to another and may appear outright irrational. In addition, a sentiment may vary based on a person's mood; thus, it is essential to have the data examined from a given time standpoint. Moreover, when doing sentiment analysis, it is difficult for machines to detect sarcasm and ironic language.

Some of the machine learning algorithms used in sentiment analysis:

- Bayes' Naive Classifier
- Natural Networks
- Maximum Entropy
- Support Vector Machines

3 Overview of related topics

3.1 Machine Learning

Machine Learning is a field of Artificial Intelligence that derives information from data, using self-learning algorithms in order to make predictions (Raschka, 2017). A research area that takes aspects from computer science, statistics, and mathematics, Machine Learning is the technology behind AI assistants, self-driving cars and facial recognition. The most common algorithmic techniques used in Machine Leaning are *supervised learning* and *unsupervised learning*. They differ in the types of problems they solve and will be discussed in more detail below.

3.2 Supervised Learning

The goal of supervised learning is to model labeled training data so that predictions can be made on unseen data. (Raschka, 2017). In the example below, there is input data X and output data Y. The aim would be to find the model to accurately predicts the new input data 4, a reasonable model in this example is $Y = x^3$.

Input (X)	1	2	3	4
Output (Y)	1	8	27	?

The two types of problems solved in Supervised Learning are *classification* and *regression* (Raschka, 2017):

- **Classification**: A classification problem is when the output variable is a category, such as trying to predict if particular pieces of text are positive or negative.
- **Regression**: A regression problem is when the output variable is continuous values, such as trying to predict someones' height or weight based on particular features .

In supervised learning, the aim is to train the algorithm data and for it to make accurate predictions on new data that has the same characteristics as the training data. If the algorithm is able to make accurate predictions on new data, it is said that it is able to generalise from the training data to test data. So the aim is to build an algorithm that is able to generalise as accurately as possible.(Raschka, 2017)

Some of the popular supervised learning algorithms :

- K-Nearest Neighbors (Classification)
- Linear Regression (Regression)
- Bayesi Naive Classifier (Classification)

3.3 Unsupervised Learning

Unlike supervised learning, in unsupervised learning problems, the data isn't labeled and the structure is unknown. The aim of unsupervised learning is to find patterns, rules, and groups in a data set. (Andreas C. Müller, 2016).

The two types of problems solved in unsupervised learning are *clustering* and association:

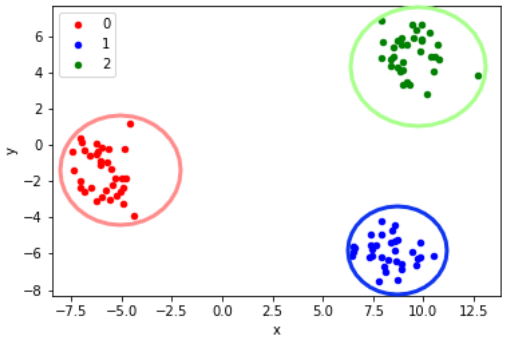
- **Clustering**: is where subgroups are found in the data, such as Netflix grouping customers based on the movies they watch.
- **Association**: is where rules are found which describe the data set. For example, people who watch the movie "Spiderman" also watch the movie "Superman".

A major challenge in unsupervised learning is accessing if the model has learned something useful. As unsupervised learning algorithms are usually applied to unlabelled the data, the right output is unknown. Therefore, it is hard to know if the algorithm did 'well'. Since there is no way to tell the algorithm what we are looking for, the assessment of its performance is usually done manually. As a consequence data analyst use unsupervised learning in an exploratory setting, as a way to gain a better understanding of the data, rather than an automatic process in which the algorithm searches for groupings/patterns haphazardly. (Andreas C. Müller, 2016)

Algorithms used in *unsupervised learning*:

- K- means (Clustering)
- Apriori algorithm (Association)

Clustering Example:



3.4 Natural Language Processing

Natural language processing, a field in artificial intelligence that helps computers understand human language. Natural Language Processing combines computer science, statistics and computational

linguistics, to bridge the gap between human communication and computer understanding. (SAS, 2019)

Applications of Natural Language Processing:

- Word translation: The process of translating from one language to another (Spanish to English), Google Translate being a popular application of this.
- Document summarisation: The process of summarising books, news articles, webpages and blogs to a shorter and comprehensible version.
- Speech to text conversion: The process of converting speech in to text, the driving force behind the AI assistants Siri and Cortana.

4 Methodology

4.1 Naive Bayes Algorithim

Based on Bayes' Theorem, with an assumption of independence between features/predictors; the Naive Bayes classifier is an algorithm that assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. For example, a fruit is considered a strawberry if it's red, heart-shaped and 2-inches in length. Realistically, these features most likely depend on one another. With the independence assumption, all the features are said to independently contribute to the fruit being a strawberry and therefore the algorithm is referred to as naive. (Ray, 2017)

Bayes' Theorem, named after the Mathematician Reverend Thomas Bayes, is 'the conditional probability of each of a set of possible causes for a given observed outcome can be computed from knowledge of the probability of each cause and the conditional probability of the outcome of each cause' (Dictionaries, 2019). In other words, the probability of the event A given B is equal to the probability of event B given A, multiplied by the probability of A upon the probability of B. The theorem provides a method to calculate the probability of the occurrence of a particular event, based on prior conditions/events. For example, as height is related to gender, someone's sex can be more accurately predicted knowing the height of the person. The formula of Bayes Theorem is as follows (Kendall, Stuart, & Ord, 1987):

BAYES THEOREM:

$$P(A \mid B) = \frac{P(B \mid A) \cdot P(A)}{P(B)}$$

Conditions:

Here A and B are events and P(B) $\neq$ 0:

- $P(A \mid B)$ is a conditional probability: the likelihood of event A occurring given that B is true.
- $P(B \mid A)$ is also a conditional probability: the likelihood of event B occurring given that A is true.
- $P(A)$ and $P(B)$ are the probabilities of observing A and B independently of each other; this is known as the marginal probability.

In a Bayes Naive classifier problem, there is an object with features $f_1, ..., f_n$ as well as possible classes it can take $C_1, ..., C_m$. The aim is to to predict the class of this object. First, each $P(C_i \mid f_1, ..., f_n)$ is calculated - the probability that an object belongs to each class C_i given the features $f_1, ..., f_n$. Then, a maximum is found, $\hat{p}$ - where $\hat{p}$ is our correct prediction - from the set $\{P(C_1 \mid f_1, ..., f_n), ..., P(C_m \mid f_1, ..., f_n)\}$, with $P(C_i \mid f_1, ..., f_n)$ equating to (Triton, 2018):

$$P(C_i \mid f_1, ..., f_n) = \frac{P(f_1, ...f_n \mid Ci)\, P(C_i)}{P(f_1, ..., f_n)}$$

$\hat{p}$ equating to:

$$\hat{p} = \mathbf{max} \frac{P(f_1, ...f_n \mid C_i)\, P(C_i)}{P(f_1, ..., f_n)}$$

Since there is an independence assumption between features, $P(f_1, ..., f_n \mid C_i) = P(f_1 \mid C_i) \cdot P(f_1 \mid C_i) \cdot ..... \cdot P(f_n \mid C_i)$ (Triton, 2018). Changing the previous formula to:

$$P(C_i \mid f_1, ..., f_n) = \frac{\prod_{j=1}^{n} P(f_j \mid C_i) \cdot P(C_i)}{P(f_1, ..., f_n)}$$

Since the denominator of the $P(C_i \mid f_1, ..., f_n)$'s never change, the denominator can be ignored (Triton, 2018). Simplifying $\hat{p}$ to:

$$\hat{p} = \mathbf{max} \prod_{j=1}^{n} P(f_j \mid C_i) \cdot P(C_i)$$

In the analysis that is done in this project, the features are individual words (w_j) in tweets (t_k). There is an assumption that the order of words doesn't matter, so "amazing features" will been seen as the same as "feature amazing". The classes will be positive (P_{os}) or negative (N_{eg}). Therefore, the maximum to be found between:

$$P(P_{os} \mid t_k) = \prod_{j=1}^{n} P(w_j \mid P_{os}) \cdot P(P_{os})$$

and

$$P(N_{eg} \mid t_k) = \prod_{j=1}^{n} P(w_j \mid N_{eg}) \cdot P(N_{eg})$$

As this is a supervised learning problem, the probabilities are obtained from an already labeled dataset for tweets (t_k). The probabilities are calculated by the following:

$$P(P_{os}) = \frac{\text{Tweets classed as positive}}{\text{Total number of tweets}}$$

$$P(N_{eg}) = \frac{\text{Tweets classed as negative}}{\text{Total number of tweets}}$$

$P(w_j \mid C_i)$ is calculated using the following:

$$P(w_j \mid C_i) = \frac{\text{Number of times } w_j \text{ appears in } C_i}{\text{Total number of words in class } C_i}$$

If a word w_1 doesn't appear in the dataset or under a particular classification then $P(w_j \mid C_i) = 0$, which leads to $P(t_k \mid C_i) = 0$, causing all information in the tweet to be lost. To midget this problem, changes are made to the calculation of $P(w_i \mid c_i)$. 1 is added to the number of times w_i appears in the class c_i and the vocabulary (V) count of the data set is added to the total number of words in c_i. This technique is called Lapace smoothing. Changing $P(w_j \mid C_i)$ to the following:

$$P(w_j \mid C_i) = \frac{\text{Number of times } w_j \text{ appears in } C_i + 1}{\text{Total number of words in class } C_i + \text{V}}$$

4.2 Worked Naive Bayes Example

In the following example, the Bayes Naive Classifier will be applied to the dataset below, in an attempt to determine if the tweet "Amazing features" is has positive or negative sentiment. (The 'tweets' which are manually classified as positive (=1) and negative (=0))

Classification	Comment
1	"Amazing"
1	"Nice features"
0	"Bad"
0	"Hate features"
0	"Very Very bad"
?	"Amazing features"

Key Information:

Total number of tweets: 5

Number of positive tweets: 2
Number of negative tweets: 3
Number of words in negative tweets: 6
Number of words in positive tweets: 3
Word count in "Amazing features" (n): 2
Vocabulary Count (V): 6

The Calculation:

$$P(P_{os}) = \frac{\text{Tweets classed as positive}}{\text{Total number of tweets}} = \frac{2}{5}$$

$$P(N_{eg}) = \frac{\text{Tweets classed as positive}}{\text{Total number of tweets}} = \frac{3}{5}$$

So a maximum will be found between:

$$\prod_{i=1}^{n} P(w_i \mid N_{eg}) \cdot P(N_{eg}) \text{ and } \prod_{i=1}^{n} P(w_i \mid P_{os}) \cdot P(P_{os})$$

with

$$\prod_{i=1}^{n} P(w_i \mid N_{eg}) \cdot P(N_{eg}) = P(''amazing'' \mid N_{eg}) \cdot P(\text{"}feature\text{"} \mid N_{eg}) \cdot P(N_{eg})$$

$$\prod_{i=1}^{n} P(w_i \mid P_{os}) \cdot P(P_{os}) = P(''amazing'' \mid P_{os}) \cdot P(\text{"}feature\text{"} \mid P_{os}) \cdot P(P_{os})$$

Doing the calculations:

$P(\text{"}amazing\text{"} \mid P_{os}) = \frac{\text{The number of times "Amazing" appears in positive tweets + 1}}{\text{number of words in positive tweets + V}} = \frac{1+1}{3+6} = \frac{2}{9}$

$$P(\text{"}features\text{"} \mid P_{os}) = \frac{\text{The number of times "features" appears in positive tweets + 1}}{\text{number of words in positive tweets + V}} = \frac{1+1}{3+6} = \frac{2}{9}$$

$$P(\text{"}amazing\text{"} \mid N_{eg}) = \frac{\text{The number of times "Amazing" appears in negative tweets + 1}}{\text{number of words in negative tweets + V}} = \frac{1}{6+6} = \frac{1}{12}$$

$$P(\text{"}features\text{"} \mid N_{eg}) = \frac{\text{The number of times "features" appears in negative tweets + 1}}{\text{number of words in negative tweets + V}} = \frac{1+1}{6+6} = \frac{2}{12}$$

$$\prod_{i=1}^{n} P(w_i \mid P_{os}) \cdot P(P_{os}) = \frac{2}{9} \times \frac{2}{9} \times \frac{2}{5} = \frac{8}{405}$$

$$\prod_{i=1}^{n} P(w_i \mid N_{eg}) \cdot P(N_{eg}) = \frac{1}{12} \times \frac{2}{12} \times \frac{3}{5} = \frac{1}{120}$$

$\frac{8}{405} > \frac{1}{120}$

Which implies:

$$P(P_{os} \mid \text{"}Amazing features\text{"}) > P(N_{eg} \mid \text{"}Amazing features\text{"})$$

So the tweet is classified as positive.

5 Results

5.1 Confusion Matrix

The results of the code are represented in the form of a confusion matrix. A confusion matrix presents the actual and predicted classifications of a classification algorithm. For the projects results, a confusion matrix of a two class problem is used. The rows of the matrix are the Predicted class while the columns are the actual class (Chawla, Bowyer, Hall, & Kegelmeyer, 2011).

		Prediction outcome	
		Positive	**Negative**
actual value	**Positive**	A	B
	Negative	C	D

The entries of the confusion matrix have the following meaning: D is number of negative examples classified as negative (True Negatives) , C is the number of positive examples classified as positive -True Positives, B is the number of positive examples classified as negative -False Negative, A is the number of negative examples classified as positive -False Positive(Chawla et al., 2011).

The predictive accuracy, often seen as the key measure of performance is given by:

$$\text{Predictive Accuracy} = \frac{A+D}{A+B+C+D}$$

In the present of an unbalanced dataset, predictive accuracy might not be a good measure of the performance of the algorithm. For example, suppose we have a sample of 10 cases, 9 positive and 1 negative. If the algorithm classifies all cases as negative, the accuracy will be 90%, even though all the negative classifications are wrong. Two other useful statistical measures of performance of two class classification problems are sensitivity and specificity. Sensitivity, also referred to as true positive rate or recall, is the percentage of positives that have been correctly classed as positives. Specificity, also referred to as true negative rate, is the percentage of negatives correctly classified as negative (of Regina, 2019). The formulas for sensitivity and specificity are as follows:

$$\text{Sensitivity} = \frac{A}{A+B}$$

$$\text{Specificity} = \frac{C}{C+D}$$

5.2 The Dataset

The data is a publicly available data set taken from Kaggle.com. The data is comprised of comments from Twitter users in the one column and the sentiment of comments in the next column (0 representing negative and 1 representing positive). The data consists of 99,989 comments, with 43,532 labeled as 0 (Negative) and 56,457 as 1(Positive). Some examples of the tweets:

"no movie times for sunday! Rats will have to plan tomorrow. I guess this means I have to work on those two presentations I am doing!", classified as negative.

""we were standing out in the pouring rain.. We were sitting on top of the world.. – summer sucks –"", classified as negative.

"@tjwriter Hubby has Briso on DVD. I haven't seen it all yet but I enjoy it. I saw Army of Darkness and Evil Dead because of Hubs too", classified as positive.

"@yhf My twenties: Look, a bed! WONDER WHAT WE CAN DO. My thirties: Look, a be–ZZZZZZZZZZzz http://tr.im/n88N", classified as positive.

5.3 Data Preperation

Applying the Bayes Naive Classifier on the dataset, the aim is to predict the polarity of the tweets (classifying them as positive or negative).

Before the algorithm is trained, cleaning steps must be applied, for the information presented in each tweet to only consists of words. Punctuation, emails addresses, Twitter usernames, numbers, whitespaces are all removed. It is also useful if for example, "World", "worlD" and "World" are seen as instances of the word "world", so all tweets are converted to lowercase. Then, filler words such as "and", "but" and "therefore" are removed, as they reveal no sentiment, thus they are unlikely to be of benefit to the algorithm.

After the data cleaning, each tweet is split into individual components using a process called tokenisation. A token is an element of string text; in this case, the elements are words. The sentence "That was a great movie" would result in ["That", "was", "a", "great", "movie"].The process is represented as a matrix, where the columns are all the unique words that occurred in the dataset and the rows are each tweet. Each entry is the number of times each word occurred in each tweet. This allows for R to see the tweets as a collection of individual words.

The dataset is split into two parts: the training set, consisting of 80% of the tweets, is used the train the algorithm and the testing set, consisting of 20% of the tweets, which the algorithm is tested on.

To improve the run time of the code, only words with a frequency count of more than 40 times are used to train the algorithm. Setting a frequency count of 40 reduces the number of features (unique words) used in the algorithm from 114,194 to 2154.

After all the data preparation, the data is represented in the form of a matrix, where the columns are all the unique words that occurred in the dataset, the rows are each individual tweet, the entries are the number of occurrences in the data:

	Features									
Tweets	dont	get	good	just	know	like	lol	love	now	thanks
11347	0	0	0	0	0	0	0	0	0	0
14512	0	0	0	0	0	0	0	0	0	0
1901	0	0	0	0	0	1	0	0	1	0
4632	0	0	0	0	0	0	0	0	0	0
4670	0	0	0	0	0	0	0	0	0	0
7099	0	1	0	0	0	0	0	1	2	0
73136	0	0	0	0	0	0	0	0	0	0
8835	0	0	0	2	0	0	0	0	0	1
9496	0	0	0	0	0	0	0	0	0	0
95201	1	1	0	0	0	0	0	0	0	0

5.4 Results

		Prediction Outcome	
		Positive	**Negative**
Actual Value	**Positive**	9584	2257
	Negative	2930	5227

$$\text{Predictive Accuracy} = \frac{9584 + 5227}{19998} = 74.06\%$$

$$\text{Sensitivity} = \frac{9584}{9584 + 2257} = 80.09\%$$

$$\text{Specificity} = \frac{5227}{2930 + 5227} = 64.08.\%$$

5.5 Improving the Performance

The Bayes' Naive Classifier is dependent on the number of features (In this case, unique words) it has to work with, so an increase in the number of features could improve the performance. In the first attempt of running the algorithm, only words that appeared in the data set more 40 times were used to train the algorithm. A reduction in the frequency count from 40 to 30 increases the number of features (from 1677 to 2154).

The results when the frequency count is reduced to 30:

		Prediction outcome	
		Positive	**Negative**
actual value	**Positive**	9556	2285
	Negative	2870	5287

$$\text{Predictive Accuracy} = \frac{9556 + 5287}{19998} = 74.22\%$$

$$\text{Sensitivity} = \frac{9556}{9556 + 2285} = 80.07\%$$

$$\text{Specificity} = \frac{5287}{2870 + 5297} = 64.74\%$$

Reducing the minimum frequency count from 40 to 30 only slightly improve the performance. The predictive accuracy increased from 74.06% to 74.22%, the sensitivity sightly decreased from 80.09% to 80.07% and the specificity increased from 64.08% to 64.74%.

5.6 Conclusion

In this project I researched sentiment analysis and how its can be applied using the Bayes' Naive Classifier and I was able to successfully apply the algorithm to a dataset of tweets. I was also able to slightly improve the accuracy of the algorithm by reducing the frequency count from 30 to 40, this increased the run time of the code from 45mins to about 2 hours. I would have liked to find

out what frequency count produced the best predictive results but my laptop crashed when the frequency count was below 25. With a more powerful device, all the unique words could have been used to create train algorithm. Using all the unique words, laplace smoothing could have been applied which might have improved the accuracy.

References

Alessia, D. (2015). *International journal of computer applications* (No. 26-33). Foundation of Computer Science.

Andreas C. Müller, S. G. (2016). *Introduction to machine learning with python* (No. 1-30). O'Reilly Media.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2011). Smote: Synthetic minority over-sampling technique. Journal Of Artificial Intelligence Research, Volume 16, pages 321-357, 2002. doi: 10.1613/jair.953

Dictionaries, O. L. (2019). *Bayes' theorem.* `https://en.oxforddictionaries.com/definition/bayes'\_theorem`. ([Online; accessed 03-April-2019])

Kendall, M. G., Stuart, A., & Ord, J. K. (1987). *Kendall's advanced theory of statistics* (No. 289). Oxford University Press, Inc.

Liu, B. (2015). *Sentiment analysis : mining opinions, sentiments, and emotions* (No. 1-15). Cambridge University Press.

of Regina, U. (2019). *Confusion matrix.* `http://www2.cs.uregina.ca/~dbd/cs831/notes/confusion_matrix/confusion_matrix.html`. ([Online; accessed 03-April-2019])

Perner, S. (2015). *Sentiment analysis for government: an optimized approach. international workshop on machine learning and data mining in pattern recognition* (No. 98-112). Springer International Publishing.

Raschka, S. (2017). *Python machine learning - second edition* (No. 1-15). Packt Publishing.

Ray, S. (2017). *6 easy steps to learn naive bayes algorithm.* `https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/`. ([Online; accessed 03-April-2019])

SAS. (2019). *What is natural language processing.* `https://www.sas.com/en\_gb/insights/analytics/what-is-natural-language-processing-nlp.html`. ([Online; accessed 03-April-2019])

Triton. (2018). *Naive bayes' classifier: How to build a sentiment analysis program.* `https://triton.ml/blog/sentiment\-analysis`. ([Online; accessed 18-August-2018])

van Zaanen Menno, & Dewi, L. M. (2017). *Meaning of sentiment analysis for companies.* The 3rd PIABC Parahyangan International Accounting and Business Conference.