A Negative Statement?
Using an Automatically Annotated, Imbalanced Dataset to Train a Naive Bayes Classifier

Ana Natalia Donaldson
University of New Mexico
adonalds@unm.edu

Report for Distributed Research Experiences for Undergraduates (DREU)
Kansas State University
Manhattan, Kansas
August 16, 2014

1. Abstract

By analyzing social media data, researchers can potentially evaluate how users feel about products or recent events. Researchers may accomplish this by investigating whether authors express positive or negative sentiment. They can train classifiers to determine the polarities of social media posts.

When researchers investigate polarity, they must have access to data that someone has marked as positive or negative. They may have to automatically annotate data posts if they plan to extract large datasets from social media sites. One way of annotating social media posts automatically is to use emoticons, text representations of facial expressions, as sentiment labels. While emoticons can often be accurate sentiment labels, they can also be misleading.

I use a dataset of Twitter posts ("tweets") published between 2006 and 2012 as the basis for my training and test sets. To develop the training set, I extract tweets with emoticons from the dataset and use the emoticons as sentiment labels. The test set consists primarily of tweets about movies. To annotate the test tweets, I use emoticons as sentiment labels and compare words within the tweets to positive and negative word lists.

When I extract tweets that contain emoticons, I retrieve more positive tweets than negative tweets. While the classifier achieves over 80% accuracy on the emoticon-based datasets, it also has difficulty recognizing negative tweets. Classifier performance improves if I remove tweets that contain URLs from the test sets. This trend indicates that the test tweets may convey objective information if they carry URLs.

2. Introduction

Social media sites like Twitter are potentially a gold mine for researchers. For example, researchers can gauge how people feel about products or recent events.

Researchers can assess changes in collective mood or mine opinions through sentiment analysis. One sentiment analysis technique involves identifying the polarity of a particular post. When making a "positive" post, an author expresses a positive emotion or opinion. When making a negative post, an author expresses a negative post or opinion.

To train classifiers that can identify polarity, researchers have to acquire training documents that someone has already marked positive or negative. When researchers work with smaller datasets, they can add labels by hand. Researchers have to annotate posts automatically if they want to take advantage of the vast amounts of data that social media sites offer. One common technique is to compare words within posts to sentiment lexicons, lists of positive and negative words.

Researchers may also look at other textual clues to determine polarity. For example, authors sometimes clarify meaning or highlight their emotional states by adding text representations of facial expressions. Called "emoticons", these text facial expressions sometimes have close associations with positive or negative emotions. Researchers can identify emoticons that represent emotions and treat them as indicators of positive or negative sentiment.

Using emoticons as sentiment labels, researchers can quickly generate new datasets. However, emoticons can mislead a classifier even when posts are short. In [1] and [2], researchers describe how the emoticons that a Twitter author uses may correspond poorly with the true meaning of her post. Read [1] mentions that Twitter authors may use sarcasm in their posts. Twitter posts may also contain both positive and negative emoticons. Cholick [2] observes that Twitter authors may include emoticons in objective tweets.

In this paper, I explore using emoticons to automatically annotate Twitter posts (“tweets”). I train a Naive Bayes classifier on automatically annotated tweets and have it categorize six different tweet test sets. While the training set contains tweets about many topics, the test sets primarily contain tweets about movies.

3. Related Work

Multiple researchers have used emoticons as sentiment labels and trained Naive Bayes classifiers on automatically annotated data:

- Read [1] has investigated how accurately his classifiers categorize data from different domains. He trained both a Naive Bayes classifier and a SVM classifier on UseNet newsgroup articles that contained emoticons. When the classifiers categorized test data from different domains, they achieved only slightly better accuracy than a randomized sorting would. The classifiers performed better when classifying emoticon-labeled data.
- Pak and Paroubek [3] have assessed whether their classifier performs better if they break down tweets into unigrams, bigrams, or trigrams. Their classifier performed best when they used bigrams.
- Zhao et al. [4] use graphic emoticons as sentiment labels when they analyze posts from the Twitter-like Chinese site Weibo. They link emoticons to emotions rather than to positive or negative sentiment. Their system MoodLens can analyze a continuous stream of posts. Zhao et al. have used MoodLens to examine how Weibo authors' moods change over time and how Weibo authors react to highly publicized events.

Researchers may also examine multiple indicators of sentiment to determine polarity:

- Hu et al. report in [5] that their system ESSA incorporates emoticons and sentiment lexicons as well as textual similarity between tweets and word co-occurrence. ESSA also employs matrix tri-factorization to determine polarity. Hu et al. found that ESSA achieved slightly better accuracy scores (around 70%) than several other approaches when it categorized two tweet datasets.
- Mukherjee et al. [6] have devised the system TwiSent, which includes a spam filter and uses sentiment lexicons, text features, and dependency relationships between words to determine polarity. TwiSent categorized automatically annotated data more accurately than manually annotated data. Mukherjee et al. believe that TwiSent has learned to recognize when Twitter authors express positive or negative sentiment explicitly. It lacks the ability to analyze less direct expressions of sentiment such as sarcasm.

4. Corpus Construction

Twitter authors write casually. As Pak and Paroubek note in [3], Twitter authors can only write tweets that are 140 characters or less. Tweets often contain misspellings and slang.

I have modified a tweet dataset assembled for another project. Cholick describes how he has collected the tweets in [2]. He retrieved tweets that contained emoticons using a Twitter API. Cholick also retrieved tweets that mentioned movies by using movie titles as keywords. While seeking out tweets about movies, Cholick took advantage of multiple Twitter APIs[1] as well as the social media analytics site Topsy[2]. Tweets

1 https://dev.twitter.com/start
2 http://topsy.com/

that contained emoticons appeared on Twitter in 2011. Tweets about movies appeared on Twitter between 2006 and 2012.

I refer to the tweets that contain emoticons as “emoticon tweets” and tweets about movies as “movie tweets.”

4.1 Training Set Construction

I have identified positive and negative emoticons that I can use as sentiment labels. These emoticons come from several Websites that offer lists of emoticons and descriptions.[3][4][5][6][7] The selected emoticons seem to clearly convey positive or negative emotions. In addition, I have also found emoticons in the original dataset that closely resemble emoticons on the Web. My search has yielded 54 positive emoticons and 67 negative emoticons.

To construct the training set, I extract tweets from Cholick's dataset that contain at least one positive or negative emoticon. As I extract tweets, I throw out any that contain both positive and negative emoticons. I consider these tweets ambiguous.

4.2 Test Set Construction

I use three different methods to determine the polarities of movie tweets. To make three of the datasets, I also remove any tweets that contain URLs. The movie tweets may be objective if they point to reviews or articles on other Websites. Over 80% of Cholick's movie tweets contain URLs.

To see how well the classifier categorizes data similar to the training data, I have created emoticon-based test sets. While I construct these test sets the same way that I construct the training set, I only use movie tweets that contain emoticons.

I also generate test sets by checking if words in the tweets appear in sentiment lexicons. When I generate the lexicon-based test sets, I refer to the positive and negative word lists[8] that Liu, Hu, and Cheng [7] have produced.

3 https://support.skype.com/en/faq/FA12330/what-is-the-full-list-of-emoticons
4 http://cool-smileys.com/text-emoticons
5 http://www.computeruser.com/emoticons?name_directory_startswith=#
6 https://messenger.yahoo.com/features/emoticons
7 http://en.wikipedia.org/wiki/List_of_emoticons
8 http://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html

Hu et al. [5] mention determining polarity based on whether a tweet contains more positive or negative words. When I use this method, I drop any tweets that have the same number of positive and negative words. I refer to this method as “majority voting.”

The remaining datasets consist of tweets that contain only positive and objective words or negative and objective words. As a result, these tweets contain no explicit sentiment conflicts.

By comparing test results for the lexicon-based test sets, I may learn whether majority voting affects classifier performance by introducing ambiguity into the data.

Mukherjee et al. [6] and Pak and Paroubek [3] consider that negation words like “not” and “neither” alter word polarity. I also try to account for this. When the negation words “not,” “neither,” or “nor” appear in a tweet, I assign the opposite polarity to the first subsequent word that appears in a word list. For example, the sentence “Pie does not make me happy.” has a negative polarity although the key word, “happy”, is positive.

4.3 Cleaning Process

I initially remove or replace noisy features such as

- **HTML symbol entities**: All printable HTML symbol entities transform into their ASCII equivalents. For example, the code "&lt;" becomes "<". If an HTML symbol entity has an unprintable equivalent, I remove the tweet.
- **URLs**: None of the tweets contain URLs. When constructing some of the test sets, I remove the tweets completely if they contain URLs.
- **Twitter-specific features:** I remove Twitter-specific features. These include usernames ("@username") and retweet abbreviations ("RT"). I also remove hashtags.

After I remove noisy features, I also remove duplicate tweets.

Prior to performing any experiments, I remove emoticons from the data. I target emoticons that I am using as sentiment labels as well emoticons that have no meaning in the experiments. This process also involves locating lengthened or exaggerated emoticons. For example, a Twitter author may expand the common emoticon ":)" to ":))))))".

I use two methods to remove non-English tweets. First, I assume that only the first 128 Unicode characters can appear in English tweets. Second, I use a spellchecker to screen for non-English tweets.

I treat the spellchecking software PyEnchant[9] as a non-English tweet filter. If PyEnchant recognizes 70% or more of the words in a tweet, the tweet may become part of a dataset. By setting the threshold at 70%, I hope to retain more tweets that contain proper nouns, slang, and misspellings.

I also preserve contractions in the datasets.

5. Experimental Setup

I use the data-mining software Weka[10] to classify tweets. Designed by Hall et al. [8], Weka provides a multinomial Naive Bayes classifier that I use in all of my experiments.

The next few sections provide some background on Naive Bayes and describe measures of classifier performance. Section 5.1 explains how Weka represents each tweet and introduces Naive Bayes. Section 5.2 introduces measures of classifier performance.

Later sections discuss the experimental setup. Section 5.3 explores how I develop a feature vocabulary. Section 5.4 describes the training and test sets.

5.1 Naive Bayes

To prepare the data for the multinomial Naive Bayes classifier, Weka selects *n* words (features) from a dataset to use as a vocabulary. It then converts every tweet into a feature vector of *n* elements. Every element *w* in a feature vector *W* represents the number of times that feature *w* appears in the corresponding tweet.

Pak and Paroubek give a review of Naive Bayes, the basis for the Naive Bayes classifier, in [3]. The probability that a tweet *t* is class *c* given that the tweet contains particular features is

$$P(c|t)=\frac{P(c)\times\prod_{w\in W}P(w|c)}{\prod_{w\in W}P(w)}$$

5.2 Measures of Classifier Performance

Three useful measures provide information about how well a classifier performs: accuracy, the Precision-Recall Curve (PRC) curve, and the Receiver Operator Character (ROC) curve. Accuracy is the simplest measure and tells how many tweets that a classifier correctly places into both classes. Manning et al. [9] note that the bases for PRC curves, precision and recall, reveal more about classifier performance than accuracy if the dataset contains significantly more instances of one class.

Manning et al. [9] and Davis and Goadrich [10] both review ROC and PRC curves. Equations for PRC and ROC curves incorporate how accurately a classifier categorizes the instances of a particular class *c*. Before Weka can build the curves, it has to know the number of instances that are

- ***True positives (tp):*** Instances that the classifier has accurately assigned to *c*
- ***True negatives (tn):*** Instances that the classifier has accurately assigned to the class opposite *c*
- ***False positives (fp)***: Instances that the classifier has incorrectly assigned to *c*
- ***False negatives (fn)***: Instances from *c* that the classifier has incorrectly assigned to the opposite class

Weka builds PRC curves by plotting precision vs. recall. Precision examines what percentage of tweets assigned to *c* are actually in *c*. The mathematical definition is

$$Precision=\frac{tp}{tp+fp}$$

9 http://pythonhosted.org/pyenchant/
10 http://www.cs.waikato.ac.nz/ml/weka/

Dataset	**Accuracy (%)**	**ROC Area**	**PRC Area**	
			Positive	**Negative**
Non-English Filter Omitted	83.739	0.800	0.945	0.496
Non-English Filter Included	86.022	0.810	0.953	0.482

Table 1. Averaged Cross Validation Results for Possible Training Sets

Recall (R) examines what percentage of tweets in c receive an accurate label from the classifier. More concisely,

$$Recall = \frac{tp}{tp + fn}$$

Weka builds ROC curves by plotting the true positive rate (recall) vs. the false positive rate (FPR).

The false positive rate examines what percentage of instances from the opposite class that the classifier assigns to c. More concisely,

$$FPR = \frac{fp}{fp + tn}$$

While Weka calculates the ROC area for both classes, I choose to report the ROC area as a single value. The ROC area scores for each class always match.

Davis and Goadrich explain in [10] how to begin building ROC and PRC curves. Classifiers like multinomial Naive Bayes determine the probability that an instance belongs to c. The classifier assigns an instance to c if that probability is above a certain threshold. To build a PRC or ROC curve, Weka repeatedly changes the threshold and alters the tp, tn, fp, and fn values.

5.3 Feature Vocabulary

I have configured Weka to remove features from the vocabulary for two reasons. First, Weka removes any word that appears on its internal stop word list. Second, Weka removes a feature if it appears less than five times in the training set.

miss	can't
sad	don't
follow	bad
hate	hurts
love	great
good	hey
sick	cry
feel	sucks
happy	didn't
ugh	birthday

Table 2. The 20 Most Relevant Features

The majority of emoticon tweets contain positive emoticons. If I leave in non-English tweets, the dataset contains 341,482 positive tweets and 70,428 negative tweets. When I remove non-English tweets, the dataset contains 136,669 positive tweets and 24,783 negative tweets. Positive tweets make up around 80% of either dataset.

I have examined how well the classifier performs when it trains on filtered and unfiltered datasets. For each dataset, I use 10-fold validation to examine classifier performance. As Table 1 shows, using PyEnchant as a filter improves accuracy.

I have also used mutual information (MI) to select relevant features from the filtered training data. In [11], Hall defines the mutual information between a class c and a feature w as

$$MI = H(c) - H(c|w)$$

where H is the entropy function. Table 2 shows the 20 highest ranking features.

When the classifier categorizes the test tweets, it only uses as a vocabulary those features that Weka has isolated through feature selection.

Annotation Method	Positive Tweets	Negative Tweets
Majority Voting		
All Tweets	19118	14710
Tweets without URLs	4153	2631
No Sentiment Conflicts		
All Tweets	17190	13173
Tweets without URLs	3480	2150
Emoticons		
All Tweets	883	99
Tweets without URLs	513	44

Table 3. Test Set Proportions

Annotation Method	Accuracy (%)	ROC Area	PRC Area	
			Positive	Negative
Majority Voting				
All Tweets	60.627	0.724	0.758	0.661
Tweets without URLs	67.895	0.779	0.836	0.687
No Sentiment Conflicts				
All Tweets	60.587	0.731	0.764	0.668
Tweets without URLs	68.188	0.790	0.848	0.696
Emoticons				
All Tweets	87.882	0.679	0.948	0.222
Tweets without URLs	89.767	0.743	0.971	0.244

Table 4. Classification Results for Test Sets

5.4 Test Set Characteristics

In Section 6, I examine how well the classifier sorts six different test sets. Table 3 shows the number of positive and negative tweets in each test set. I have only found a small number of movie tweets that contain emoticons. Like the training set, the emoticon-based tests sets are imbalanced. The other four lexicon-based test sets have a more balanced ratio of positive to negative tweets.

6. Results

Table 4 summarizes how well the classifier performs when it categorizes each test set. The classifier achieves better accuracy scores and lower negative PRC area scores when it categorizes imbalanced datasets. This pattern holds whether the classifier encounters the tiny emoticon tests sets or segments of the training data. In contrast, accuracy scores fall and negative PRC area scores rise when the classifier categorizes the more balanced lexicon-based test sets.

When the classifier categorizes the lexicon-based datasets, classifier performance varies. The classifier achieves slightly higher accuracy, ROC area, and PRC area scores if the tweets have no sentiment conflicts.

When the test sets include tweets that contain URLs, the classifier achieves worse performance. In particular, the classifier only attains a slightly better accuracy score than a randomized sorting would when a lexicon-based test set includes all eligible tweets.

7. Discussion

Table 4 shows that noisy movie tweets may sometimes contain URLs. When I remove movie tweets that contain URLs, the classifier achieves better accuracy, ROC area, and PRC area scores.

As Table 4 also shows, the classifier tends to achieve slightly better accuracy, ROC area, and PRC area scores when the tweets have no sentiment conflicts. Twitter authors may sometimes be expressing more sophisticated ideas in tweets that contain both positive and negative words. Like Mukherjee et al.'s classifier, my classifier may have trouble recognizing sentiments that Twitter authors state implicitly. The classifier may also achieve slightly better performance because the "no sentiment conflict" tests sets are smaller.

A potential extension of this research is mining Twitter for opinions about movies. However, the emoticons present in a tweet may reveal an author's general mood rather than an opinion about a movie. For example, a Twitter author can be upset about missing theatrical showings of particular movies:

> *i haven't seen toy story 3, or any of the other movies like letters to juliet, karate kid, eclipse! i keep missing the last full shows :(*[11]

Although the ":(" marks this tweet as negative, the Tweet author expresses positive sentiment about the movies that she mentions.

I extract more positive tweets from Cholick's dataset for one of two reasons. First, my set of emoticons may be biased. Second, Twitter authors may use positive emoticons much more frequently. When Park et al. [12] examined Twitter data from 2006-2009, they found that ":)" was the most popular emoticon. They also found that Twitter authors typically used emoticons in positive tweets rather than negative ones.

8. Conclusion

Researchers can use emoticons to automatically annotate large social media datasets. Previous research has shown that when classifiers train on emoticon-labeled data, they tend to achieve better results when they categorize test sets from the same domain.

To create my training set, I search for tweets that contain emoticons and use those emoticons to label the tweets. Whenever I use this method, the resulting dataset contains more positive tweets than negative tweets. While the classifier achieves high accuracy scores on the emoticon-based test sets, it has trouble recognizing negative tweets.

When the test sets lack tweets with URLs, classifier performance improves. This trend suggests that tweets with URLs are more likely to convey objective information.

11 The source for this tweet is Topsy.

9. Acknowledgments

The program Distributed Research Experiences for Undergraduates (DREU) provided me with the opportunity conduct research under Dr. Doina Caragea.

10. References

[1] J. Read, "Using emoticons to reduce dependency in machine learning techniques for sentiment classification," in *Proceedings of the ACL Student Research Workshop*, June 2005, pp. 43-48.

[2] M. Cholick, "Vision document for movie tweets: A micro-blogging bootstrapped recommender, Version 2.0," 2012. [Online]. Available:

https://sites.google.com/site/cholickmseproj/documentation

[3] A. Pak and P. Paroubek, "Twitter as a corpus for sentiment analysis and opinion mining," in *Proceedings of the Annual Conference on Language Resources and Evaluation*, 2010.

[4] J. Zhao, L. Dong, J. Wu, and K. Xu, "MoodLens: An emoticon-based sentiment analysis system for Chinese tweets," in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2012, pp. 1528-1531.

[5] X. Hu, J. Tang, H. Gao, and H. Liu, "Unsupervised sentiment analysis with emotional signals," in *Proceedings of the 22nd International Conference on World Wide Web*, 2013, pp. 607-618.

[6] S. Mukherjee, A. Malu, B. A.R., and P. Bhattacharyya, "TwiSent: A multistage system for analyzing sentiment in Twitter," in *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, 2012, pp. 2531-2534.

[7] B. Liu, M. Hu and J. Cheng, "Opinion observer: Analyzing and comparing opinions on the Web," in *Proceedings of the 14th International World Wide Web Conference*, May 2005.

[8] M. Hall et al., “The WEKA data mining software: An update," *SIGKDD Explorations*, vol. 11, no. 1, 2009.

[9] C. D. Manning, P.Raghavan, and H. Schütz, "Evaluation in information retrieval," in *Introduction to Information Retrieval*, Cambridge, England: Cambridge University Press, 2008. [Online]. Available:

http://nlp.stanford.edu/IR-book/html/htmledition/evaluation-in-information-retrieval-1.html

[10] J. Davis and M. Goadrich, "The relationship between Precision-Recall and ROC curves," in *Proceedings of the 23rd International Conferenceon Machine Learning*, June 2006, pp. 233-240.

[11] M. Hall, "Class InfoGainAttributeEval," [Online]. Available:

http://weka.sourceforge.net/doc.dev/weka/attributeSelection/InfoGainAttributeEval.html

[12] J. Park, V. Barash, C. Fink, and M. Cha, "Emoticon style: Interpreting differences in emoticons across cultures," in *Proceedings of the 7th AAAI Conference on Weblogs and Social Media*, 2013, pp. 466-475.