Sentiment Analysis for Amazon Reviews

Wanliang Tan
wanliang@stanford.edu

Xinyu Wang
xwang7@stanford.edu

Xinyu Xu
xinyu17@stanford.edu

Abstract

Sentiment analysis of product reviews, an application problem, has recently become very popular in text mining and computational linguistics research. Here, we want to study the correlation between the Amazon product reviews and the rating of the products given by the customers. We use both traditional machine learning algorithms including Naive Bayes analysis, Support Vector Machines, K-nearest neighbor method and deep neural networks such as Recurrent Neural Network(RNN), Recurrent Neural Network(RNN). By comparing these results, we could get a better understanding of the these algorithms. They could also act as a supplement to other fraud scoring detection methods.

1. Introduction

Recent years have seen an increasing amount of research efforts expanded in understanding sentiment in textual resources. As we can see the statistics from web of knowledge in Figure one, the papers published on sentiment analysis have been increasing for the past years. One of the subtopics of this research is called sentiment analysis or opinion mining, which is, given a bunch of text, we can computationally study peoples opinions, appraisals, attitudes, and emotions toward entities, individuals, issues, events, topics and their attributes. Applications of this technique are diverse.

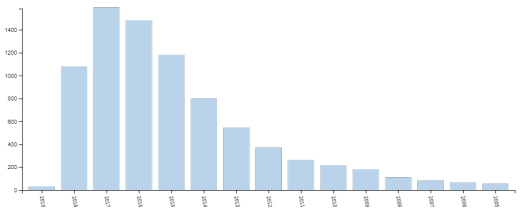

Figure 1. Published Papers on Sentiment Analysis

For example, businesses always want to find public or consumer opinions and emotions about their products and services. Potential customers also want to know the opinions and emotions of existing users before they use a service or purchase a product. Last but not least, researchers[2] uses these information to do an in-depth analysis of market trends and consumer opinions, which could potentially lead to a better prediction of the stock market.

However, saying this, to find and monitor opinion sites on the Web and distill the information contained in them remains a formidable task because of the proliferation of diverse sites. Each site typically contains a huge volume of opinionated text that is not always easily deciphered in long forum postings and blogs. The average human reader will have difficulty identifying relevant sites and accurately summarizing the information and opinions contained in them[5]. Besides, to instruct a computer to recognize sarcasm is indeed a complex and challenging task given that at the moment, computer still cannot think like human beings.

The objective of this paper is to classify the positive and negative reviews of the customers over different products and build a supervised learning model to polarize large amounts of reviews. Our dataset consists of customers' reviews and ratings, which we got from Consumer Reviews of Amazon products. We extracted the features of our dataset and built several supervised model based on that. These models not only include traditional algorithms such as naive bayes, linear supporting vector machines, K-nearest neighbor, but also deep learning metrics such as Recurrent Neural Networks and convolutional neural networks. We compared the accuracy of these models and got a better understanding of the polarized attitudes towards the products.

2. Related Work

So far, there are a lot of research papers related to product reviews, sentiment analysis or opinion mining. For example, Xu Yun[8] el al from Stanford University applied existing supervised learning algorithms such as perceptron algorithm, naive bayes and supporting vector machine to predict a review's rating on Yelp's rating dataset. They used hold out cross validation using 70% data as the training data and 30% data as the testing data. The author used different classifiers to determine the precision and recall values. In paper[3], Maria Soledad Elli and Yi-Fan extracted sentiment from the reviews and analyze the result to build up

a business model. They claimed that this tool gave them pretty high accuracy. They mainly used Multinomial Naive Bayesian(MNB) and support vector machine as the main classifiers. Callen Rain[6] proposed extending the current work in the field of natural language processing. Naive Bayesian and decision list classifiers were used to classify a given review as positive or negative.

Deep-learning neural networks is also popular in the area of sentiment analysis. Ronan Collobert[1] et al used a convolutional network for the semantic role labeling task with the goal avoiding excessive task-specific feature engineering. On the other hand, in paper[7], the authors proposed proposed using recursive neural networks to achieve a better understanding compositionality in tasks such as sentiment detection.

In this paper, we want to apply both traditional algorithms including Naive Bayesian, K-nearest neighbor, Supporting Vector Machine and deep-learning tricks. By comparing the accuracy of these models, we would like to get a better understanding how these algorithms work in tasks such as sentiment analysis.

3. Dataset and features

3.1. Data Preprocessing

Our dataset comes from Consumer Reviews of Amazon Products[1]. This dataset has 34660 data points in total. Each example includes the type, name of the product as well as the text review and the rating of the product. To better utilize the data, first we extract the rating and review column since these two are the essential part of this project. Then, we found that there are some data points which has no ratings when we went through the data. After eliminating those examples, we have 34627 data points in total.

Besides, to have a brief overview of the dataset, we have plot the distribution of the ratings. In Figure 2, it shows that we have 5 classes - rating 1 to 5 as well as the distribution among them. Also, these five classes are actually imbalanced as class 1 and class 2 have small amount of data while class 5 has more than 20000 reviews. Here is one sample from our dataset:
Review text: 'This product so far has not disappointed. My children love to use it and I like the ability to monitor control what content they see with ease.'
Rate: '5'

In the subsection '3.3 Features', we will illustrate how we convert a review text into an input vector, and we simply take the rate of a review as its label.

[1]https://www.kaggle.com/datafiniti/consumer-reviews-of-amazon-products

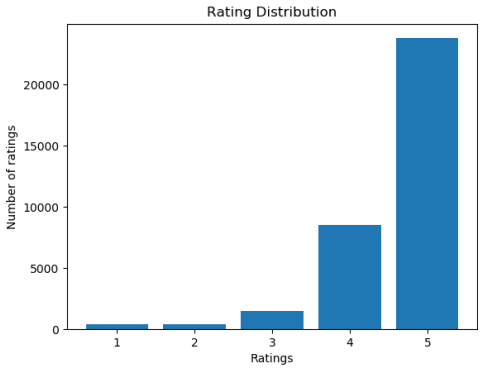

Figure 2. Rating Distribution of Amazon Reviews

3.2. Data Resampling

Due to the imbalance of our dataset, we have tried data resampling in some of our experiments. Data resampling is a popular way of dealing with imbalanced data. In this project, we tried to oversample the data of class 1,2 and 3 by repeatedly sampling those reviews because these three classes have far less samples than the other two. Therefore, the original reviews of label 1,2 and 3 has been repeated 15 times in our training set. However, since there are many repeated samples in the training set, it is easy for the model to overfit.

3.3. Features

We have tried two types of features in the project. The first type is a traditional method. Basically, we build a dictionary based on the common words and index each word. We set the threshold for the word dictionary to be 6 occurrence and ended up collecting 4223 words from our entire dataset. Then we transform each review into a vector, where each value represents how many times the word shows up. For this, we actually tried changing the threshold and the length of the dictionary. What we found is that the increase of the dictionary's length did not have too much effect on the accuracy.

Another type of feature we used is the 50-d glove[2] dictionary which was pretrained on Wikipedia. For this part, we basically want to take advantage of the the meanings of each word. In this case, we represent each review by the mean vector of 50-d glove vectors of all individual words making up the review. As we will see in our result, because of the way we represent each review, the features got weakened and the distance between different reviews actually is not that accurate.

[2]https://nlp.stanford.edu/projects/glove/

4. Methods

4.1. Naive Bayes

Naive Bayes is one of the most common generative learning algorithms for classification problems. This algorithm assumes that $x_i's$ are conditionally independent given y, which is called Naive Bayes assumption.

$$p(x_1, ..., x_k|y) = \prod_{i=1}^{k} p(x_i|y)$$

We also incorporated Laplace Smoothing in our model to make it work better. The prediction of an example is given by the formula below:

$$\hat{y}^{(i)} = \arg\max_j \prod_{i=1}^{k} p(x_i|y=j)\phi(j)$$

With the first way of representing review texts, it takes an array of non-negative integers, and models $p(x_i|y)$ with multinomial distribution. With the second way of representing review texts using glove dictionary, the inputs are no longer non-negative integers, so we chose to model $p(x_i|y)$ with Gaussian distribution.

4.2. K-nearest Neighbor

K-nearest Neighbor(KNN) is a nonparametric classification method. It has been widely used recently. When making a prediction, this method fist look for the $K = n$ nearest neighbours of the input. Then, it will assign the majority of that n neighbours' class. The distance between each neighbour is euclidean distance, which is able to measure the similarity between each data point.[4]

$$\hat{f}(x) = \frac{1}{K} \sum_{x \in N_K(x)} y_i$$

The equation above shows the mathematical representation of KNN algorithm. The general idea of KNN is that if the inputs are similar to each other, then the output would be the same. In this project, we have tuned the number of nearest neighbours K among 4,5 and 6.

4.3. Linear Support Vector Machine

Linear SVM is a mthod that creates a classifier(a vector) that separates the labeled datasets. Geometrically given two types of points, circles and x's, in a space, it tries to maximize the minimum distance from one of the points to the other. In other words, it maximizes the margin. The optimization problem that SVM tried to solve is below:

$$\arg\max_{\gamma,w,b} \frac{1}{2}||w||^2$$
$$s.t. y^i(w^T x + b) \geq 1, i = 1, 2, ...m$$

It tried to find the w to satisfy the maximum margin problem and satisfy the separability constraint.

4.4. Long Short Term Memory

Long Short Term Memory(LSTM) is unit of Recurrent Neural Network(RNN). A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. The cell remembers values over arbitrary time intervals and the three gates regulate the flow of information into and out of the cell. LSTM networks are well-suited to classifying, processing and making predictions based on time series data, since there can be lags of unknown duration between important events in a time series. The configuration is shown as the Figure 2 below.

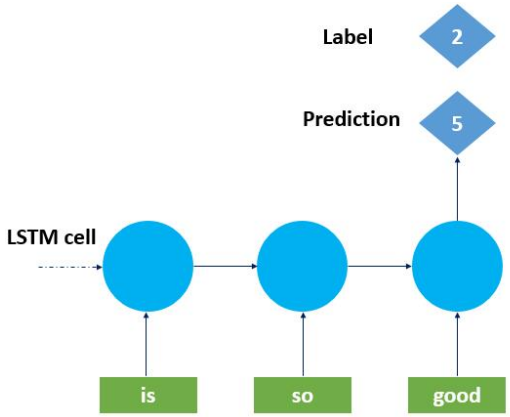

Figure 3. LSTM configuration

For this method, we have also tried to input the original text with glove embedding. We found that there are 33629 reviews' length less than 100 words, which is about 97.1 percent of the whole dataset. Therefore, the max text length has been set to 100. Then, all the review text data has been padding to 100 word length and the words which are after 100 have been removed. Because when implementing this method, the input shape of the data should be the same. After that, each word is represented by glove word vector as the input of the neural network.

In this project, we used the LSTM with 128 hidden units and then used a dense net with softmax as the activation function to predict these 5 classes. The data has been trained for 20 epochs in experiments using LSTM. Adam optimizer has been used to optimize the parameters , the learning rate is 0.01 and the batch size is 32. To prevent overfitting, a dropout rate of 0.2 was set in the LSTM layer.

5. Results and Discussion

5.1. Results

The entire dataset of 34,627 reviews was divided into a training set of size 21000 (60%), a validation set of size 6814 (20%) and a test set of size 6813 (20%).

With 4223-d input features representing review text, we implemented Multinomial Naive Bayes, SVM with Linear Kernel, SVM with RBF Kernel, KNN-4, 5, 6 and LSTM. KNN-5 outperforms the other 2 KNN models and SVM with Linear Kernel slightly outperforms SVM with RBF Kernel. The SVM with Linear Kernel seems to have overfitting problem indicated by the significant gap between training accuracy and test accuracy. LSTM performs best in term of test accuracy among all of them.

With 50-d input features from glove dictionary, we run Gaussian Naive Bayes, SVM with Linear Kernel and KNN-4, 5 6 and LSTM. KNN-5 outperforms the other 2 KNN models again. Besides, we tried data resampling on LSTM model but unfortunately it did not improve the test accuracy due to overfitting problem. It turns out that LSTM generates best predictions among all models again.

Detailed results of training and test accuracy of all models are listed in Table 1.

Models	Training Acc.	Test Acc.
Multinomial NB	75.1%	70.6%
Linear SVM	83.4%	69.6%
RBF SVM	69.7%	69.2%
KNN-4	61.7%	61.7%
KNN-5	65.5%	65.4%
KNN-6	64.9%	64.6%
LSTM	73.5%	71.5%
Gaussian NB w/ Glove	52.2%	52.4%
Linear SVM w/ Glove	68.7%	68.6%
KNN-4 w/ Glove	58.1%	57.6%
KNN-5 w/ Glove	62.6%	62.2%
KNN-6 w/ Glove	61.3%	61.6%
LSTM w/ Glove	70.1%	70.2%
LSTM w/ Glove(Resample)	85.6%	65.6%

Table 1. Performance of different models

In general, all models perform better with traditional input features than with glove input features. Specifically, LSTM generates the most accurate predictions over all other models. The Figure 4 shows the ranking of different models according to their test accuracy.

5.2. Discussion

We observed that KNN required much higher computation complexity than Naive Bayes and SVM during train time. As in KNN algorithm, it needs to calculate the distance of all the evaluation data points and all the training data points, which is more time consuming.

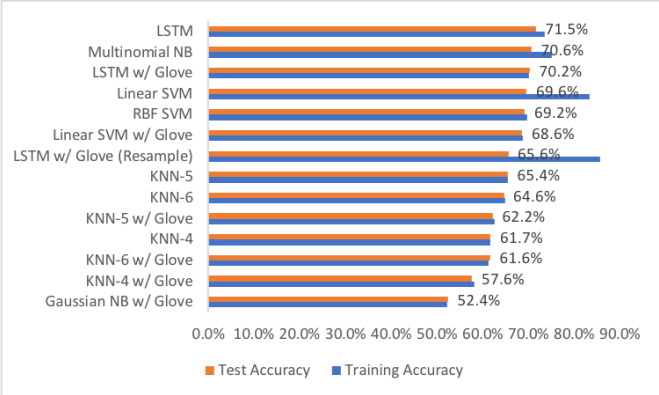

Figure 4. Ranking of different models by test accuracy

In addition, the increase of the dictionary's length did not have too much effect on the accuracy. One explanation is that when we decrease the threshold of the dictionary, the length of dictionary will increase. But the problem is that we only have less than 40,000 reviews. If we think about it, the number of data points is not that significantly larger than the dimension of feature space. So the curse of dimensionality could the issue here.

The result using glove mean is worse than the method of normal word count. The possible reason is that if we use the average, the individual word feature will be weakened, then the distance between different reviews will be inaccurate.

When it comes to LSTM, the result is a little bit better than other conventional machine method due to the bigger amount of the parameters. We can also see from table 1, after resampling, the training accuracy of LSTM with Glove has reached 85.6 %. However, the test accuracy is only 65.6 %, which means this model has overfitted on the resampled data, since there are many repeated examples.

6. Conclusion and Future Work

In summary, we have tried two types of features. For this two type of features, we tried all the algorithms we mentioned in the model part including Naive Bayes, SVM, KNN, LSTM. From the results, we can see that our accuracy on the test set is the best when we use LSTM on the first type of feature. One of the main reason our accuracy is not high enough is because of the data imbalance. We tried resampling and different weighting techniques that we got from the feedbacks of the audience during the poster session. But that didn't help too much. Another possible solution we haven't tried is to find more data points from other resources. We think that might help us solve the problem of data imbalance.

7. Contribution

Wanliang Tan: Responsible for Support Vector Machine algorithm and converting review texts to input features.

Xinyu Wang: Responsible for Naive Bayes algorithm.

Xinyu Xu: Responsible for the KNN and LSTM algorithm, data preprocessing and resampling.

References

[1] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537, 2011.

[2] K. Dave, S. Lawrence, and D. M. Pennock. Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. In *Proceedings of the 12th international conference on World Wide Web*, pages 519–528. ACM, 2003.

[3] M. S. Elli and Y.-F. Wang. Amazon reviews, business analytics with sentiment analysis.

[4] S. Hota and S. Pathak. Knn classifier based approach for multi-class sentiment analysis of twitter data. In *International Journal of Engineering Technology*, pages 1372–1375. SPC, 2018.

[5] B. Liu and L. Zhang. *A Survey of Opinion Mining and Sentiment Analysis*, pages 415–463. Springer US, Boston, MA, 2012.

[6] C. Rain. Sentiment analysis in amazon reviews using probabilistic machine learning. *Swarthmore College*, 2013.

[7] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

[8] Y. Xu, X. Wu, and Q. Wang. Sentiment analysis of yelps ratings based on text reviews, 2015.