SENTIMENT CLASSIFICATION ON AMAZON MOVIE REVIEWS

CIS 563 – Intro to Data Science

Harper He

Email: xhe128@syr.edu | Student ID: 331490565

Table of Contents

1. Introduction

One of the main applications of NLP techniques is to categorize documents, for example by genre, topics or the sentiment expressed in the text. For our project, we focused on sentiment detection problem of movie reviews.

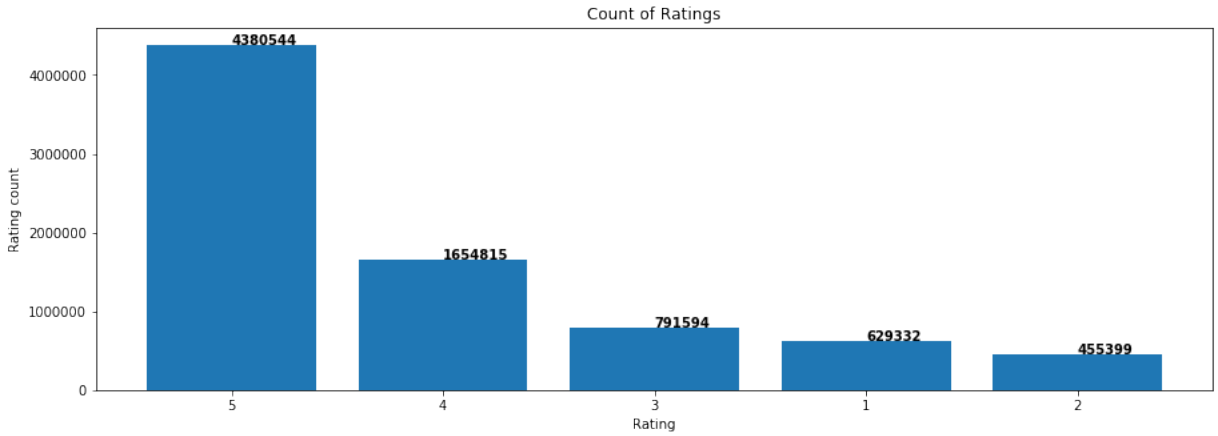
This domain is appealing for several reasons. First, it is experimentally convenient. The correct labels can be extracted from the explicit ratings which reflect the overall sentiment of reviewers. We do not need to label the data manually for classification or evaluation purposes. Second, sentiment information about reviews are useful. While explicit ratings provide the audience's opinion quantitatively, reviews provide us a deeper insight on the tastes and opinions of audiences and what the strong and weak points of movies, which can be an asset for movie industry and recommender systems.

The objective of this project is to apply different machine learning methods in the task of sentiment analysis of movie reviews. There are many movie topic websites where users can rate and comment on movies based on their opinions, such as Rotten Tomatoes, IMDB and Amazon. Our sentiment analysis project aims at using raw movie review data from Amazon to classify reviews on a scale of two classes: negative and positive.

The raw data set consists of 7,911,684 movie reviews of 253,059 movies. Below is an example of the raw data.

```
product/productId: B00006HAXW
review/userId: A1RSDE90N6RSZF
review/profileName: Joseph M. Kotow
review/helpfulness: 9/9
review/score: 5.0
review/time: 1042502400
review/summary: Pittsburgh - Home of the OLDIES
review/text: I have all of the doo wop DVD's and this one is as good or better than the 1st ones. Remember once these performers are gone, we'll never get to see them again. Rhino did an excellent job and if you like or love doo wop and Rock n Roll you'll LOVE this DVD !!
```

As we can see from the distribution of reviewers' ratings, the original dataset is very imbalanced.



The sentiment of reviews is binary, meaning the rating below 3 results in a sentiment class "negative", and rating above 3 has a sentiment class "positive". Reviews with a rating of 3 were regarded "neutral" and were removed from the research area for this project. To avoid domination of the documents by a

small number of unhelpful, meaningless reviewers, we also considered the helpfulness (fraction of users who found the review helpful) of reviews, so we kept the reviews that more than 80 users found helpful. After imposing these limitations, the final dataset was a corpus of 34,552 reviews, 30,216 positives and 4,336 negatives.

2. Prior Work

There are many comprehensive reviews related to applying feature selection and machine learning classification techniques to sentiment analysis.

Extraction of proper features in reasonable numbers is always an issue in sentiment analysis. Most of the researches focus on some simple features, including bad of words, bigrams or the combination of these features. Some other studies have adopted different feature selection methods. For example, Juan Ramos[1] provide evidence that Term Frequency Inverse Document Frequency (TF-IDF) efficiently categorizes relevant words that can enhance query retrieval.

Though many feature selection methods have been studied individually, there are not many studies that have compared and analyzed the performance of different types of features selection methods, when used with different machine learning techniques according to Yiming Yang and Jan O. Pedersen[2] . This contributed to my interest in comparing the performance of different feature extraction with the same classifiers in this project.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan[3] employed supervised machine learning techniques such as Naive Bayes, maximum entropy classification and support vector machines to perform sentiment classification. Though these methods did not perform as well on sentiment classification as on traditional topic-based classification, they were the first to apply machine learning to the problem of sentiment categorization. The results of their experiments showed that using unigrams as attributes did about 5% better than using bigrams, which justified our decision to use the bag-of-words representation as our text representation and Naive Bayes as one of the machine learning techniques.

Decision Tree is not as commonly used in sentiment analysis as SVM and Naïve Bayes. Michelle Annett and Grzegorz Kondrak[4] compared a series of method by applying them to a publicly available set of movie reviews. Alternating Decision Tree (ADTree) algorithms were one of the most accurate in their preliminary experiments.

Sumandeep Kaur, Geeta Sikka, Lalit Kumar Awasthi[5] combined N-gram algorithm and kNN classifier to classify input data into positive, negative and neutral classes. The results of their experiment showed that it performed well as compared to the existing model which is based on SVM classifier.

3. Methods

3.1 Data Preparation

For our dataset, we performed a series of preprocessing steps to remove noises in order to build models. Data preparation consisted of following steps:

- removal of punctuation like “!”,”?” and numbers because they do not provide important information.
- lemmatization, converting a word to its base form by using Wordnet lemmatizer in order to reduce the number of features.

- removal of stop words from the reviews using NLTK stop words list.
- tokenization by splitting strings into words and form a bag of words.

After these preprocessing steps, we got a cleaned dataset with only review text, individual words of each review and the label of each review.

	words	token	label
3	trailer catch attention long show begin unfami...	[trailer, catch, attention, long, show, begin,...	pos
4	since watcheed first episode justify ive fanat...	[since, watcheed, first, episode, justify, ive...	pos
5	love justify much order season daughter u im w...	[love, justify, much, order, season, daughter,...	pos
6	love first episode gladly order new season bas...	[love, first, episode, gladly, order, new, sea...	pos
7	tim come burst big screen jennifer garner love...	[tim, come, burst, big, screen, jennifer, garn...	pos

3.2 Feature Extraction

To implement machine learning algorithms, we converted the words to some numeric representation. The feature extraction is an important task in sentiment analysis for two reasons. First, it decreases the size of the informative vocabulary to improve the efficiency of classifiers. Second, it can increase the classification accuracy because it ignores useless vocabulary. In this project, we have tried two feature extraction methods, bag-of-words and TF-IDF.

Bag-of-words

The bag-of-words model is a typical and straightforward way to numerically represent texts. It learns all the words across all of the reviews, then it counts the number of times each word appears. As the total number of words in our dataset was huge, we limited the size of the feature vectors. Specifically, we extracted the 3,000 most frequent words as features to train the classifier. Since words only occur a few times in the dataset would contribute little to the classifier, the loss of information was acceptable.

TF-IDF

TF-IDF stands for term frequency-inverse document frequency. Different from the bag-of-words which concentrates more on higher frequency words in the documents, TF-IDF decreases the weight of words that occur very often in the document and increases the weight of terms that occur rarely. One shortcoming of bag-of-words is that it ignores words which might occur less frequently but have significance for the sentiment of the review to some extent. While TF-IDF overcomes this. Similar to the bag-of-words model, we extracted the 3,000 most important words as features to train the classifier. IDF (inverse document frequency) is used to measure term's importance. It is calculated as below so that it gives more importance to the rarely occurring terms in the document.

$$IDF(t) = log_e \frac{\text{Total number of documents}}{\text{Total number of documents with term } t \text{ in it}}$$

After getting the IDF, we can calculate the final weight for a term t in a document d as:

$$TF - IDF(t,d) = TF(t,d) \times IDF(t)$$

3.3 Models

Before developing classifiers to learn the sentiments of movie reviews, we randomly divided the data corpus into two datasets – training dataset (80% of the original dataset) and testing dataset (20% of the original dataset). The training process of classifiers was performed on the reviews from the training

dataset. We explored multiple classification models on above extracted features and performed the 3-fold cross-validation to evaluate the quality of these classifier. The classifier with best cross-validation performance was used to predict the sentiment of reviews in the testing dataset.

Once we had the feature vectors for each review and divided the data corpus into training and testing dataset, we developed the following classifiers to learn the sentiments of reviews: Decision Tree, Naïve Bayes classifier and k-Nearest Neighbors.

Decision tree
Decision tree performs classification by using "yes" or "no" conditions at each level. In this project, it simply checked for the presence of a single feature. The label of each review was assigned according to the decision node of the tree.

Naive Bayes
Naive Bayes assumes that all features are independent and applies Bayes' rule on the review to calculate the probability for its label in the training dataset. The likelihood estimate of each label was from the weight of all features, and the review was assigned the label with the highest likelihood estimate.

k-Nearest Neighbors (kNN)
k-Nearest Neighbors first calculates the distance of each reviews' numeric representations, then classifies each review by taking the majority class of its k (decided by the user) nearest neighbors.

4. Results and findings

When evaluating classifiers, some commonly used evaluation metrics include accuracy, precision, recall, F-score as well as ROC&AUC.

In this model training process of this project, we used accuracy as metric to select the best model. After applying the best model to the testing dataset, we evaluated the performance of classifier using accuracy, precision, recall, f1-score and ROC&AUC.

		PREDICTIVE VALUES	
		POSITIVE (1)	NEGATIVE (0)
ACTUAL VALUES	POSITIVE (1)	TP	FN
	NEGATIVE (0)	FP	TN

Accuracy, precision, recall and f1-score can be calculated using the value of true positive (TP), true negative (TN), false positive (FP) and false negative (FN) which can be seen in the confusion matrix.

- Accuracy = (TP+TN)/ (TP+TN+FP+FN)
- Precision = TP/(TP+FP)
- Recall = TP/(TP+FN)
- F1-score = 2*Precision*Recall/ (Precision + Recall)

Accuracy is the relation between the reviews that were correctly classified and all the reviews in the test set. However, it might give us flawed results because the number of positive reviews was much larger than the number of negative ones. The accuracy would be very high even if the classifier did not learn anything and assigned all the testing dataset to the "positive" class. So, we took precision, recall and F1-score into consideration.

ROC curve shows the performance of a classification model by plotting two parameters: True Positive Rate and False Positive Rate. The former is a synonym for recall while the latter can be calculated as FPR=FP/(FP+TN). An ideal ROC curve has a True Positive Rate 1 and a False Positive Rate as 0. AUC

stands for Area under the curve. AUC gives the rate of successful classification by the classification model. The AUC makes it easy to compare the ROC curve of one model to another.

The average accuracies computed by three-fold cross-validation using different machine-learning techniques over two set of features are listed above.

Features/Models	Decision tree	Naïve Bayes classifier	k-Nearest Neighbors
Bag of words	97%	85.78%	95.37%
TF-IDF	97.17%	91.45%	95.34%

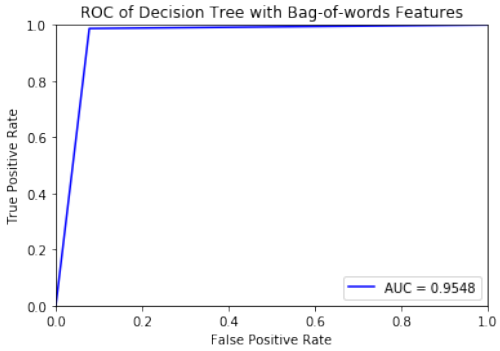
Generally, the results from Decision tree, Naïve Bayes and k-Nearest Neighbors were pretty close. Considering the fact that in the training dataset, the majority class "positive" takes up 87.39%, we could tell the Naïve Bayes classifier with bag-of-words features did not work well because its accuracy is lower than the percentage of the majority class. For Decision tree and k-Nearest Neighbors, the difference of feature extraction of reviews did not have an appreciable effect on their performance. TF-IDF feature extraction method outperformed bag-of-words features in the Naïve Bayes classifier.

From a machine learning point of view, the performance of the Naïve Bayes classifier was below the others. The interesting observation was that Decision Tree outperform other techniques on this sentiment classification problem.

Based on their performance in training process, we chose Decision tree to predict the label of reviews in testing dataset using both bag-of-words and TF-IDF features. The confusion matrix and ROC of each prediction are listed below.

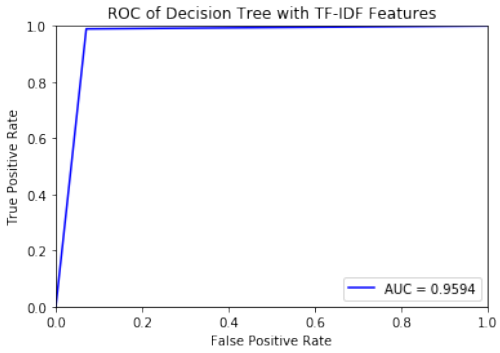
Confusion Matrix of Decision tree with bag-of-words features

		Predicted Class	
		Positive	Negative
Actual Class	Positive	5,986	78
	Negative	66	785



Confusion Matrix of Decision tree with TF-IDF features

		Predicted Class	
		Positive	Negative
Actual Class	Positive	5,995	65
	Negative	60	791



The evaluation metrics of prediction are listed below. Different from the training process, features extraction affected the performance of Decision tree in the testing dataset. The TF-IDF features gave us a higher accuracy, f1-score and AUC.

Model	Features	accuracy	precision	recall	f1-score	AUC
Decision Tree	bag-of-words	97.92%	98.91%	98.71%	0.9881	0.9548
	TF-IDF	98.19%	99.01%	98.92%	0.9897	0.9594

By developing three machine learning methods in sentiment analysis project, we deepened our understanding of the advantages and disadvantages of these classifiers. Decision Tree is relatively fast in learning the data and it works well in this specific task. Naïve Bayes is a little slower than Decision Tree and gives low performance in this project compared to other models. kNN requires more time and memory to build the model but performs well.

It took us 1.3 hours to run the script of the whole project, from extracting information and features to model building and evaluation. Among all these tasks, the most time-consuming were: building kNN models, feature extractions and text cleaning.

Our model gave us an extremely high accuracy which is very unlucky in real world sentiment analysis. This is due to the nature of our imbalanced data where one class take up nearly 90% of the reviews. To mimic the practical sentiment analysis tasks, we can manipulate the data records to make it balanced so that the evaluation would be more reasonable. We can over-sample rare class or generate artificial rare class instances or under-sample the majority class to make the data balanced. But we may change the distribution of the real data, which should be taken into consideration in the evaluation phrase.

The predictive performance of Decision Tree is comparable with Naïve Bayes and kNN in our project. But it is less popular in sentiment analysis according to prior work. We would like to further investigate the potential of Decision Tree in solving the sentiment classification problem.

References

[1] Ramos, J.E. (2003). Using TF-IDF to Determine Word Relevance in Document Queries.

[2] Yang, Y., & Pedersen, J.O. (1997). A Comparative Study on Feature Selection in Text Categorization. *ICML*.

[3] Pang, B., Lee, L., & Vaithyanathan, S. (2002). Thumbs up? Sentiment Classification using Machine Learning Techniques. *ArXiv, cs.CL/0205070*.

[4] Annett, M., & Kondrak, G. (2008). A Comparison of Sentiment Analysis Techniques: Polarizing Movie Blogs. *Canadian Conference on AI*.

[5] Kaur, S., Sikka, G., & Awasthi, L.K. (2018). Sentiment Analysis Approach Based on N-gram and KNN Classifier. *2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC)*, 1-4.