Natural Language Processing

Sentiment Analysis   
of Rotten Tomatoes Movie Reviews

Olga Perera

Syracuse University

[Introduction](#iq0g4obyk)

[Project Background and Description](#jdbro0a3nys3)

Data Description

Part 1: [Data Processing](#ke7qc7g7ymm4)

[Data Cleaning & Preparation](#rz88riuq57lk)

[Word](#mgmx7dt7lkht) Could & Most Frequent Words

[Bi-grams](#io1soyf4zg7y) Frequency Distribution

Part 2: Sentiment Analysis

[Definition](#ytmh87bflfy4) & Steps

Naïve Bayes Classifier Metrics

[Naïve](#cg3nxr8t384a) Bayes for movie reviews data

[Statistical Analysis Overview](#ypgg5dreke6j)

[Part](#nfv6rcjgwj0) 3: Metric Improvement

[Negation](#ht0ivqsm2tsi) words feature

[Recommendations](#r9a34t8msnls)

[Bibliography](#knl624oi3o50)

Project Background and Description

This study examines Kaggle dataset of movie reviews taken from the original Pang and Lee movie review corpus which is based on reviews from the Rotten Tomatoes web site and focuses on sentiment analysis of the reviews. The project’s data was manually annotated with following sentiment labels: “negative”, “somewhat negative”, “neutral”, “somewhat positive”, “positive” by Socher’s group who used Amazon's Mechanical Turk to create fine-grained labels for all parsed phrases in the corpus. The dataset can be found on Kaggle.com website:

https://www.kaggle.com/c/sentiment-analysis-on-movie-reviews

Sentiment analysis is a valuable tool for many organizations. It is defined “as the task of finding the opinions of authors about specific entities. The decision-making process of people is affected by the opinions formed by thought leaders and ordinary people. When a person wants to buy a product online he or she will typically start by searching for reviews and opinions written by other people on the various offerings. Sentiment analysis is one of the hottest research areas in computer science.” (Ronen Feldman, 2013) It helps organizations to monitor social media activities in real time and review posts and articles to extract specific values, which can lead to gaining a competitive advantage and superior returns. On the other hand, individuals receive beneficial information regarding certain products or services and in this case, movie feedback.

Data Description

According to Kaggle.com the dataset is comprised of tab-separated files with phrases. The train/test split has been preserved for the purposes of benchmarking, but the sentences have been shuffled from their original order. Each sentence has been parsed into many phrases by the Stanford parser. Each phrase has a PhraseId. Each sentence has a SentenceId. Phrases that are repeated (such as short/common words) are only included once in the data.

* train.tsv contains the phrases and their associated sentiment labels. SentenceId tracks which phrases belong to a single sentence.
* test.tsv contains just phrases. Sentiment label will be assigned to each phrase.

The sentiment labels are:

0 - negative  
1 - somewhat negative  
2 - neutral  
3 - somewhat positive  
4 - positive

Part 1: Data Processing

**Data Cleaning & Preparation**

According to Techopedia, data cleansing is the process of altering data in a given storage resource to make sure that it is accurate and correct. (Data cleansing, n.d., [para.](https://www.techopedia.com/definition/1174/data-cleansing)1). For this project, it is important to extract text data and later, determine if more transformation is beneficial to achieve better results. Movie reviews dataset is split into training and test sets.

Train and Test datasets consist of four columns: PhraseID, SentenceID, Phrase, Sentiment (Train dataset only). The Sentiment column represents manually labeled sentiment. Each sentence is split into phrases with PhraseID and SentenceID identifying phrase and sentence respectively.

| **PhraseId** | **SentenceId** | | | **Phrase** |
| --- | --- | --- | --- | --- |
| 156061 | | 8545 | An intermittently pleasing but mostly routine ... | |
| 156062 | | 8545 | An intermittently pleasing but mostly routine ... | |
| 156063 | | 8545 | An | |
| 156064 | | 8545 | intermittently pleasing but mostly routine effort | |
| 156065 | | 8545 | intermittently pleasing but mostly routine | |

During data preparation process, phrase text was extracted and converted to string format for future processing.

Next both datasets were tokenized, and frequency distribution was created to understand if further data transformation is needed for classification task.

Below are results of Train dataset tokenization:   
Number of tokens: 1124695  
Frequency distribution for top 50 tokens:

[('the', 51220), (',', 42006), ('a', 36124), ('of', 32308), ('and', 31763), ('to', 22448), ('.', 17939), ("'s", 16971), ('in', 13745), ('is', 13447), ('that', 12327), ('it', 11677), ('as', 8623), ('with', 7750), ('for', 7417), ('its', 7051), ('film', 6689), ('an', 6479), ('movie', 5905), ('this', 5677), ('but', 5126), ('be', 4977), ('you', 4827), ('on', 4730), ("n't", 3970), ('by', 3918), ('more', 3888), ('his', 3827), ('about', 3682), ('one', 3609), ('``', 3582), ('not', 3562), ('at', 3550), ('or', 3472), ('from', 3462), ('than', 3458), ('--', 3356), ('all', 3201), ('have', 3134), ('like', 3071), ('are', 3059), ('has', 2938), ("'", 2924), ('so', 2644), ('story', 2520), ('-rrb-', 2438), ('out', 2372), ('who', 2359), ('most', 2227), ('into', 2163)]

Below are results of Test dataset tokenization:   
Number of tokens: 442986

Frequency distribution for top 50 tokens:

[('the', 19942), (',', 16118), ('a', 13918), ('and', 12471), ('of', 12351), ('to', 9051), ('.', 7165), ("'s", 6570), ('in', 5537), ('is', 5322), ('it', 4857), ('that', 4531), ('as', 3353), ('with', 2972), ('its', 2942), ('for', 2847), ('this', 2529), ('an', 2448), ('film', 2436), ('movie', 2230), ('but', 2001), ('be', 1930), ('on', 1817), ('you', 1812), ('by', 1654), ('more', 1642), ('his', 1608), ("n't", 1586), ('than', 1510), ('not', 1438), ('at', 1384), ('like', 1383), ('about', 1352), ('from', 1320), ('--', 1310), ('are', 1307), ('one', 1260), ('or', 1255), ('have', 1201), ('has', 1180), ('all', 1172), ('who', 1025), ('so', 1022), ("'", 979), ('story', 946), ('too', 924), ('-rrb-', 910), ('``', 910), ('most', 901), ('much', 901)]

Additional text processing is required as majority of most common words can be categorized as stopwords. Stopwords list and a function to remove non-alphabetical characters was created for text processing. Since negation is an important factor effecting the results of sentiment, a customized list of stopwords was created:

Below is the list of stopwords used for text processing:

stopwords = ['to', 'in', 'on',

'that', "that'll", 'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about','i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'sha','wo','y',"'s","'d","'ll","'t","'m","'re","'ve","-lrb-", "-rrb-"]

Total token count after stopwords removal in train test: 671065, test set: 266371.

**Word cloud & Most Frequent Words**

Below is a word cloud representing frequencies of most common words in the train set.

A screenshot of a cell phone

Description generated with high confidence  
Frequency distribution for train set reviews:

[('film', 6689), ('movie', 5905), ("n't", 3970), ('more', 3888), ('one', 3609), ('not', 3562), ('from', 3462), ('than', 3458), ('all', 3201), ('like', 3071), ('so', 2644), ('story', 2520), ('out', 2372), ('most', 2227), ('into', 2163), ('too', 2143), ('up', 2104), ('good', 2043), ('characters', 1882), ('much', 1862), ('no', 1801), ('can', 1769), ('time', 1747), ('comedy', 1721), ('just', 1714), ('some', 1701), ('even', 1597), ('little', 1575), ('will', 1567), ('funny', 1522), ('way', 1511), ('life', 1484), ('any', 1456), ('very', 1451), ('make', 1396), ('only', 1393), ('movies', 1345), ('love', 1296), ('new', 1278), ('there', 1257), ('enough', 1248), ('work', 1243), ('us', 1218), ('bad', 1211), ('own', 1207), ('other', 1154), ('something', 1152), ('would', 1118), ('never', 1114), ('director', 1099)]

**Bi-Grams Frequency Distribution**

Bi-gram is defined as “a sequence of two adjacent elements from a string of tokens, which are typically letters, syllables, or words. A bigram is an *n*-gram for *n*=2. The frequency distribution of every bigram in a string is commonly used for simple statistical analysis of text in many applications, including in computational linguistics, cryptography, speech recognition, and so on.” (Wikipedia, n/d).

Below is a bi-gram frequency distribution for the train dataset with bi-gram scores to determine the most common pairs of words in the review text.

(('more', 'than'), 0.0004916888578681332)

(('ca', "n't"), 0.0004347845415868302)

(('so', 'much'), 0.0002960802706511543)

(('rather', 'than'), 0.00023384117471847923)

(('romantic', 'comedy'), 0.00022228248547383957)

(('each', 'other'), 0.00019471945727508346)

(('too', 'much'), 0.0001920520674493974)

(('so', 'many'), 0.00017782598837907165)

(('too', 'many'), 0.0001769368584371763)

(('new', 'york'), 0.00017160207878580416)

(('may', 'not'), 0.0001680455590182227)

(('feel', 'like'), 0.00015559773983168771)

(('subject', 'matter'), 0.000153819479947897)

(('running', 'time'), 0.0001511520901222109)

(('not', 'only'), 0.00014848470029652484)

(('special', 'effects'), 0.00014670644041273411)

(('much', 'more'), 0.00014581731047083875)

(('love', 'story'), 0.00014226079070325733)

(('soap', 'opera'), 0.0001404825308194666)

(('better', 'than'), 0.00013781514099378054)

(('good', 'time'), 0.00012003254215587337)

(('even', 'more'), 0.00011558689244639658)

(('far', 'more'), 0.00011380863256260586)

(('big', 'screen'), 0.00010936298285312907)

(('little', 'more'), 0.00010491733314365228)

(('more', 'like'), 0.00010491733314365228)

(('can', 'not'), 0.00010224994331796621)

(('too', 'long'), 9.513690378280334e-05)

(('away', 'from'), 9.335864389901262e-05)

(('plays', 'like'), 9.246951395711726e-05)

(('de', 'niro'), 9.069125407332655e-05)

(('other', 'than'), 8.891299418953582e-05)

(('sit', 'through'), 8.713473430574511e-05)

(('very', 'funny'), 8.624560436384975e-05)

(('far', 'from'), 8.446734448005904e-05)

(('very', 'good'), 8.268908459626833e-05)

(('will', 'probably'), 8.268908459626833e-05)

(('all', 'too'), 8.179995465437296e-05)

(('could', "n't"), 8.179995465437296e-05)

(('might', 'not'), 8.179995465437296e-05)

(('first', 'film'), 8.002169477058225e-05)

(("n't", 'care'), 7.824343488679153e-05)

(('action', 'sequences'), 7.735430494489618e-05)

(('feels', 'like'), 7.646517500300081e-05)

(("n't", 'make'), 7.646517500300081e-05)

(('suffers', 'from'), 7.646517500300081e-05)

(('two', 'hours'), 7.646517500300081e-05)

(('no', 'one'), 7.557604506110545e-05)

(('less', 'than'), 7.46869151192101e-05)

(("n't", 'know'), 7.46869151192101e-05)

Bi-grams reflect the most common moods and feelings of the reviewers. Bi-grams such as “far more”, “little more”, “too many”, “will probably” represent different levels of individual reaction to the movie. This helps us to analyze various levels of sentiment.

Part 2: Sentiment Analysis

**Definition & Steps**

Sentiment analysis is dedicated to extracting subjective emotions, moods, personality traits, attitudes referred as to affective aspects of semantics. One common use of sentiment analysis is to determine if a text expresses negative or positive feelings. Movie reviews dataset is a great application for doing sentiment analysis.

Sentiment analysis steps:

1. Tokenization
2. Feature extraction
3. Classifiers ­ Naïve Bayes

Movie reviews dataset have been prepared and split into train and test sets. During the last step algorithm is trained using the reviews and classifications in train.csv, and then predictions are made on the reviews contained in test.csv file. Error calculations are performed using the actual classifications in test.csv file.

Naive Bayes classifier works by figuring out the probability of different attributes of the data being associated with a certain class. This is based on Bayes theorem. It states that the probability of A given that B is true equals the probability of B given that A is true times the probability of A being true, divided by the probability of B being true.

Naïve Bayes classifier is a probabilistic machine learning classifier. It assumes that the presence of a specific feature in a class is unrelated to the presence of any other feature.

***Pros:***

* When assumption of independence holds, a Naive Bayes classifier performs better compare to other models like logistic regression and you need less training data.
* It performs well in case of categorical input variables compared to numerical variables.

***Cons:***

* If categorical variable has a category (in test data set), which was not observed in training data set, then model will assign a 0 (zero) probability and will be unable to make a prediction. This is often known as “Zero Frequency”.
* Limitation of Naive Bayes is the assumption of independent predictors. In real life, it is almost impossible that we get a set of predictors which are completely independent.

**Naïve Bayes Classifier Metrics**

Following metrics are considered for evaluating the effectiveness of a classifier:

1. Precision: measures the exactness of the classifier. Higher precision means less false positives. True Positives/True Positives + False Positives
2. Recall: measures the completeness of the classifier Higher recall means less false negatives. Improving recall may decrease precision. True Positives/True Positives + False Negatives.
3. F- measure: weighted harmonic mean of precision and recall.
4. Macro-average: used to evaluate performance across different classes. It takes the average of precision and recall of different sets.
5. Micro-average: used to evaluate performance of a specific class. It sums up true positives, false positives, and false negatives of different sets. It does not take label imbalance into consideration.

**Naïve Bayes Classifier for movie reviews dataset**

* 1. Define the features for each document, using just the words, Bag of Words (BOW) or unigram features: classification of occurrences of each word as a feature for training classifier.

First, split dataset into K equal folds (K=5). Use 1000 random phrases, each fold size = 200. Results are demonstrated below:

Read 156060 phrases, using 1000 random phrases

(['that', 'substitute', 'for', 'acting'], 2)

(['infuses', 'the', 'film'], 3)

(['loved', 'ones'], 3)

(['the', 'aaa', 'of', 'action', ',', 'xxx', 'is', 'a', 'blast', 'of', 'adrenalin', ',', 'rated', 'eee', 'for', 'excitement', '.'], 4)

(['unnamed', ',', 'easily', 'substitutable', 'forces'], 2)

(['unguarded', 'moments'], 2)

(['school', 'special'], 2)

(['there', 'are', 'many', 'things', 'that', 'solid', 'acting', 'can', 'do', 'for', 'a', 'movie', ',', 'but'], 2)

(['and', 'second', ',', 'what', "'s", 'with', 'all', 'the', 'shooting', '?'], 1)

(['in', 'a', 'joyous', 'communal', 'festival', 'of', 'rhythm'], 4)

2596

Each fold size: 200

Fold 0

Precision Recall F1

0 0.143 0.091 0.111

1 0.300 0.310 0.305

2 0.746 0.654 0.697

3 0.154 0.261 0.194

4 0.100 0.143 0.118

Fold 1

Precision Recall F1

0 0.111 0.250 0.154

1 0.167 0.375 0.231

2 0.766 0.514 0.615

3 0.192 0.323 0.241

4 0.111 0.111 0.111

Fold 2

Precision Recall F1

0 0.000 0.000 0.000

1 0.171 0.273 0.211

2 0.842 0.574 0.683

3 0.279 0.429 0.338

4 0.000 0.000 0.000

Fold 3

Precision Recall F1

0 0.000 0.000 0.000

1 0.150 0.188 0.167

2 0.911 0.626 0.742

3 0.128 0.300 0.179

4 0.000 0.000 0.000

Fold 4

Precision Recall F1

0 0.000 0.000 0.000

1 0.178 0.421 0.250

2 0.888 0.537 0.669

3 0.070 0.176 0.100

4 0.000 0.000 0.000

Average Precision Recall F1 Per Label

0 0.051 0.068 0.053

1 0.193 0.313 0.233

2 0.830 0.581 0.681

3 0.165 0.298 0.210

4 0.042 0.051 0.046

Macro Average Precision Recall F1 Over All Labels

0.256 0.262 0.245

Label Counts {0: 43, 1: 166, 2: 519, 3: 224, 4: 48}

Micro Average Precision Recall F1 Over All Labels

0.504 0.426 0.444

Results interpretation:

* The highest count predicted by the algorithm is 519 for neutral reviews, 43- negative reviews, 166- somewhat negative, 224- somewhat positive, 48- positive.
* Macro-average metrics are much lower than micro-average metrics.

Part 3: Metric Improvement & Recommendations

**Negation words feature**

1. Add negation feature to classification algorithm and run it with the same count of random phrases. Two sets of negation words were created in the attempt to improve classification results.

Negation words, set #1:  
negationwords = ['no', 'not', 'never', 'none', 'nowhere', 'nothing', 'noone', 'rather', 'hardly', 'scarcely', 'rarely', 'seldom', 'neither', 'nor']

Negation words, set #2:  
negationwords = ['no', 'not', 'never', 'none', 'nowhere', 'nothing', 'noone', 'rather', 'hardly', 'scarcely', 'rarely', 'seldom', 'neither', 'nor', 'not', 'n\'t', "don't","never", "nothing", "nowhere", "noone", "none", "not", "hasn't", "hadn't", "can't", "couldn't", "shouldn't", "won't", "wouldn't","don't","doesn't","didn't","isn't","aren't","ain't", 'bad', 'terrible', 'useless', 'hate']

Partial Output for negation set #1:

Macro Average Precision Recall F1 Over All Labels

0.246 0.275 0.235

Label Counts {0: 48, 1: 170, 2: 516, 3: 219, 4: 47}

Micro Average Precision Recall F1 Over All Labels

0.500 0.421 0.432

Partial Output for negation set #2:

Macro Average Precision Recall F1 Over All Labels

0.260 0.271 0.248

Label Counts {0: 38, 1: 166, 2: 512, 3: 226, 4: 58}

Micro Average Precision Recall F1 Over All Labels

0.523 0.436 0.454

Results interpretation:

* Negation set #2 produced higher precision, recall, F values, meaning that count of false positives and false negatives slightly decreased. All three metrics have increased between 1.5 to 2.5%.
* There is slight improvement in metrics comparing to the first (BOW) feature with 1.9% increase in Precision and 1% increase in Recall.

1. Increase count of random phrases to 10000 with each fold = 2000, K=5.

Output:

Read 156060 phrases, using 10000 random phrases

(['captures', ','], 2)

(['the', 'problems', 'of', 'the', 'people', 'in', 'love', 'in', 'the', 'time', 'of', 'money'], 2)

(['this', 'film', "'s", 'cast'], 2)

(['throwback', 'war', 'movie'], 2)

(['impulsive', 'niches'], 2)

(['parking', 'lots'], 2)

(['you', 'swinging', 'from', 'the', 'trees', 'hooting', 'it', "'s", 'praises'], 3)

([',', 'accompanying', 'the', 'stunt-hungry', 'dimwits', 'in', 'a', 'random', 'series', 'of', 'collected', 'gags', ',', 'pranks', ',', 'pratfalls', ',', 'dares', ',', 'injuries', ',', 'etc', '.', '.'], 1)

(['are', 'the', 'lively', 'intelligence', 'of', 'the', 'artists', 'and', 'their', 'perceptiveness', 'about', 'their', 'own', 'situations', '.'], 3)

(['viewers', 'are', 'asked', 'so', 'often', 'to', 'suspend', 'belief', 'that', 'were', 'it', 'not', 'for', 'holm', "'s", 'performance'], 1)

10339

Each fold size: 2000

Fold 0

Precision Recall F1

0 0.302 0.138 0.190

1 0.280 0.402 0.330

2 0.746 0.646 0.692

3 0.279 0.459 0.347

4 0.220 0.195 0.207

Fold 1

Precision Recall F1

0 0.235 0.090 0.130

1 0.272 0.427 0.332

2 0.750 0.674 0.710

3 0.256 0.409 0.315

4 0.314 0.252 0.279

Fold 2

Precision Recall F1

0 0.391 0.149 0.216

1 0.233 0.394 0.292

2 0.776 0.637 0.699

3 0.232 0.418 0.298

4 0.372 0.302 0.333

Fold 3

Precision Recall F1

0 0.359 0.178 0.238

1 0.275 0.405 0.328

2 0.748 0.676 0.710

3 0.267 0.423 0.328

4 0.339 0.257 0.292

Fold 4

Precision Recall F1

0 0.378 0.160 0.225

1 0.246 0.376 0.297

2 0.736 0.635 0.682

3 0.274 0.458 0.342

4 0.320 0.252 0.282

Average Precision Recall F1 Per Label

0 0.333 0.143 0.200

1 0.261 0.401 0.316

2 0.751 0.653 0.699

3 0.262 0.433 0.326

4 0.313 0.252 0.279

Macro Average Precision Recall F1 Over All Labels

0.384 0.376 0.364

Label Counts {0: 443, 1: 1771, 2: 5039, 3: 2164, 4: 583}

Micro Average Precision Recall F1 Over All Labels

0.514 0.515 0.504

Results interpretation:

* Higher macro average metrics, meaning that there are more balanced scores between folds.
* Improved micro average metrics: 7% increase (improvement) in Recall or decrease in false negative count.

**Recommendations**

To improve results of the classification model, following steps are recommended:

1. In general people use positive words preceded by “not” or “no” to express negative moods and feelings. When using BOW model, we assume that each word is independent, and the algorithm might not see phrases such as “not good” as a negative comment. Recommendation is to train algorithm on bi-grams.
2. Majority of words in the movies review dataset were classified as neutral, meaning that the actual sentiment is in state of “void’. Sentiment might improve if we remove neutral (meaningless) words from our data.
3. Run the model with increased number of random phrases, which provides better metric values across all sets of data.

**References**

Bigram. (2019, April 01). Retrieved from https://en.wikipedia.org/wiki/Bigram

Joshinav. (2018, March 12). 5 reasons why you must care about customer sentiment analysis. Retrieved from https://www.allerin.com/blog/5-reasons-why-you-must-care-about-customer-sentiment-analysis

What is Data Cleansing? - Definition from Techopedia. (n.d.). Retrieved from https://www.techopedia.com/definition/1174/data-cleansing

Word Cloud Definition. (n.d.). Retrieved from http://www.peachbelt.com/information/thePBCis/wordcloud