[[1]](#footnote-1)

Movie Review Sentiment Analysis

Xiyue Zhang (20601564)

*Abstract*—this project focuses on the sentiment analysis about text based movie review and classify the review into 5 sentiment classes. Firstly it introduces the database and some methods for sentiment mining. Then it transfers text based reviews into numeric vectors as classification input. Finally it creates different classifiers and chooses the best classifier with the highest accuracy to predict sentiment type. According to the experimental result, it will give some improvement for the further work.

Index Terms – Sentiment Analysis, Movie Review

# **INTRODUCTION**

The Internet now makes it possible to find out the opinions and experiences of those in the vast pool of people. And more and more people are making their opinions available to strangers via the Internet [1]. Especially when people are shopping online, the customer reviews are helpful because customers can identify product quality based on review and company can detect the customer preference to improve the product quality. So it is important to mine useful information based on sentiment analysis for customers and company.

There are different research areas about text sentiment analysis. The first area is classifying the polarity of a given text. To increase sensitive to detect positive and negative segments, it can exclude neutral text and assess the rest in terms of positive and negative sentiments. Or it can assess the text in terms of positive, negative and neutral sentiments. As most of reviews are neutral in real dataset, it is important to separate the neutral text from the whole dataset.

The second area is subjectivity/objectivity identification. This task is commonly defined as classifying a given text into one of two classes: objective or subjective. The subjectivity of words and phrases may depend on their context and an objective document may contain subjective sentences (e.g., a news article quoting people's opinions). However, removing objective sentences from a document before classifying its polarity helped improve performance.

The third area is feature/aspect-based and determines the opinion expressed on different features or aspects. Different features can generate different sentiment responses, for example the phone may have good camera but bad battery. This problem involves several sub-problems like identifying relevant entities, extracting their features and determining whether an opinion expressed on each feature/aspect is positive, negative or neutral.

This project focuses on the polarity of movie review. Besides, some papers below can help to implement sentiment analysis process. Paper [2] covers approaches that promise to directly enable opinion-oriented information-seeking systems. Link [3] determines sentiment polarity (positive or negative) or subjective rating (e.g., "two and a half stars") based on the movie review. Link [3] and [4] offer movie review dataset, some implemented code and extended methods for sentiment analysis.

# **DataSet Background**

The dataset is a corpus of movie reviews used for sentiment analysis, originally collected by Pang and Lee [5]. Socher et al. create fine-grained labels for all parsed phrases in the corpus based on Treebank [6]. The whole data is divided into train.tsv and test.tsv. The train.tsv contains the SentenceId, phraseId, phrase and their sentiment labels. The test.tsv includes everything in train.tsv except predicted labels. As shown in Figure 2.1 and Table 2.1, each sentence is parsed into phrases by the Stanford parser [7], and the label value is one of 5 classes which are negative (0), somewhat negative (1), neutral (2), somewhat positive (3), positive (4). There are many neutral opinions and few extreme opinions. And the dataset is balanced in terms of the opinions polarity.

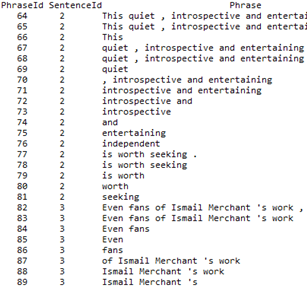


Figure 2.1 Data Structure in train.tsv

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| sentiment | (0) | (1) | (2) | (3) | (4) |
| phrases | 7072 | 27273 | 79582 | 32927 | 9206 |

Table 2.1 Data Sentiment Percent in train.tsv

As most of sentiment prediction systems work just by looking at words in isolation, giving positive points for positive words and negative points for negative words. That way, the order of words is ignored and important information is lost. For example, the sentence ‘this movie was actually neither funny nor witty’, funny and witty are positive but the following sentence is still negative So Stanford Sentiment Group proposes that Sentiment Treebank and Recursive Neural Network which build up a representation of whole sentences based on the sentence structure [7].

Treebank is the corpus with fully labeled parse trees to analyze the compositional effects of sentiment in language. The traditional Treebank only used for binary classification of positive vs. negative. However most reviews are neutral and others are between neutral and extreme. There are rarely extreme opinions in review. Besides the stronger sentiment leads to longer phrases while neutral sentiment appears with shorter phrases. So it is more reasonable to predict the sentiment based on 5-class (– –, –, 0, +, ++)

Recursive Neural Networks represent a phrase through word vectors and a parse tree and then compute vectors for higher nodes in the tree using the same tensor-based composition function. In Figure 2.2 and Figure 2.3, the leaf node, corresponding to a word, is represented as a vector. The parent vectors are computed from bottom to top using compositionality functions g. In this way both parent and leaf vectors can be treated as phrases in Figure 2.1.

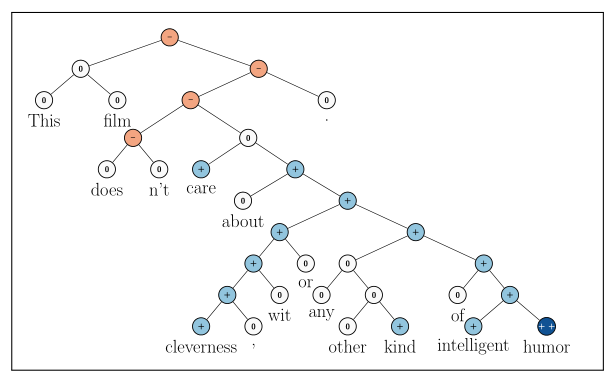


Figure 2.2 Treebank Structure

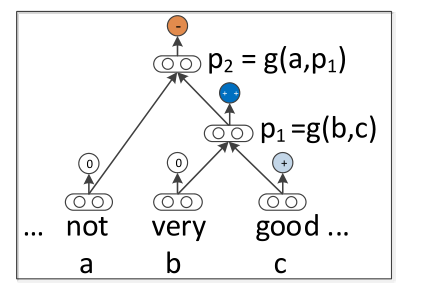


Figure 2.3 Compositionality function for parent vectors

# **Implementation**

## Data preprocessing

As the input for classification, the review text should be converted to structured data. Generally the text preprocessing includes noise reduction, feature extraction, feature vectors representation. Here nltk python library is used to implement this part.

### Noise Reduction

Before transfer the text into vector as classification input, it is necessary to preprocess the data to remove the noise or extract the phrase with strong sentiment. For example, we can delete useless information like web link or time because they have nothing to do with review sentiment. And we need to translate the emoticons into sentiment words because some people prefer to post emoticons rather than long sentence to show their opinion. Besides, we can transform capital letters into lowercase or fix spelling mistake, in this way, similar expressions combined to show same opinion and classified into the same sentiment type.

### Feature Extraction

After reducing noise, it needs to extract feature of the phrases to represent the sensitive meaning. As shown in Figure1.3, it collects 4 phrases from training set as an example, and apply ‘bag-of-n-grams’ ,‘WordNet’ and ‘Harvard General Inquirer’ to the feature extraction respectively. For each phrase, the feature can be either single word or sub-phrase to represent the whole phrase sentiment. However there are many similar words but different expression structure to represent the same sentiment. In Figure 3.1 second two phrases have the same meaning but different phrases structure. So it is not reasonable to present the phrase just by single word.



Figure 3.1 the phrases before Feature Extraction

#### Bag-of-n-grams

The “bag-of-n-grams” is the method to extract feature based on sub-phrases. It can also be used to turn the collection of text documents into numerical feature vectors, which will be discussed in the next section. As the sub-phrase contain more information than single word. It can extract word and word [sequence](https://en.wikipedia.org/wiki/Sequence) together as features in case that the word relationship may be lost. Specifically, n-gram is a contiguous sequence of n items from the given phrases. The model with larger n can store more information but also take time to transfer into feature vector. So it is better to control the n value to express the full meaning while reduce the time complexity.

Here is an example that how the n-grams feature is presented in each phrase. In Table 3.1, 1-grams use single word as features, which can’t express the full sentiment. In Table 3.2, 2-grams use combination of neighbor word as features which can express enough hidden information and it is not necessary to use larger n-grams. However there is still some duplicated features express the same sentiment, for example (entertaining) and (and entertaining) actually have the same meaning. These two features should be combined into one so that the dimension is reduced.

|  |  |
| --- | --- |
| 70 | (,) (introspective) (and) (entertaining) |
| 71 | (introspective) (and) (entertaining) |
| 72 | (introspective) (and) |
| 73 | (introspective) |

Table 3.1 the 1-grams as features in each phrase

|  |  |
| --- | --- |
| 70 | (,) (introspective) (and) (entertaining) (,introspective) (introspective, and) (and entertaining) |
| 71 | (introspective) (and) (entertaining) (introspective, and) (and, entertaining) |
| 72 | (introspective) (and) (introspective, and) |
| 73 | (introspective) |

Table 3.2 the 2-grams as features in each phrase

#### WordNet

Apart from the “bag-of-n-grams”, The WordNet is another method to extract feature based on synonyms. The WordNet is a lexical database for the English language. It groups English words into sets of synonyms called synsets, provides short definitions and usage examples, and records a number of relations among these synonym sets or their members As shown in Figure3.2, the word in the left group has the similar meaning of wet and the other group has the similar meaning of dry. It can replace the all of synonyms with their unified expression of that group. As shown in Table 3.3, it removes useless feature like (“the”, “a”, “is”) and combine the sentence with same meaning like phrase (70, 71). However, according to the training set, the phrase 72 and 73 actually have different sentiment label. We can’t combine the phrase 72 and 73 together. What is worse, even though separating the phrase 72 and 73, it is still unreasonable to label the same structured phrased with different sentiment class. So we should try to make a tradeoff between few feature and correct expression.

|  |  |
| --- | --- |
| 70 | Synset('introspective.a.01')Synset('entertain.v.01') Synset('entertain.v.02')Synset('harbor.v.01') Synset('entertaining.s.01') |
| 71 | Synset('introspective.a.01')Synset('entertain.v.01') Synset('entertain.v.02')Synset('harbor.v.01') Synset('entertaining.s.01') |
| 72 | Synset('introspective.a.01') |
| 73 | Synset('introspective.a.01') |

Table 3.3 the synsets as features in each phrases

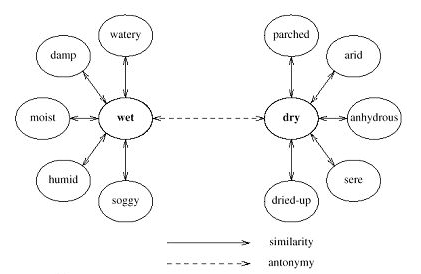


Figure 3.2 the WordNet synsets example

#### Harvard General Inquirer

The Harvard General Inquirer similar to the WordNet is a lexicon attaching syntactic, semantic, and pragmatic information to part-of-speech tagged words [10]. As shown in Figure 3.3, the spreadsheet works as a dictionary to illustrate sentiment category of each word. Each of the spreadsheet's 11,788 rows begins with an entry word or word sense in strict alphabetical order. The first row is corresponded to the Inquirer tag names and the following rows are corresponded to the words. The second column indicates whether the entry word appeared in the Harvard ("H4"), Lasswell ("Lvd") or both ("H4Lvd") dictionaries. The following 182 columns each show the assignments for a specified category.

Based on the original spreadsheet, the columns of useless categories should be removed and the rows of some new word should be added into the new spreadsheet. As the project focuses on the positive and negative opinion of reviewer, It only selects four columns which are ‘Positiv’, ‘Negativ’, ‘IAV’, ’Strong’ to represent the main sentiment of phrases, ‘Positiv’ and ‘Negativ’ are two large valence categories in spreadsheet and illustrate the words of positive and negative outlook. ’Strong’ imply the strength of positive and negative opinion. ‘IAV’ gives an interpretative explanation of an action, such as encourage, mislead. After the spreadsheet has been modified, it should be "saved as" a tab-delimited text file with an initial label row. This saved file can then be the dictionary input file to the Inquirer tagging procedure.

This project selects the columns of ‘Positiv’, ‘Negativ’, ‘IAV’, ’Strong’ based on common sense. There may be other columns related to the sentiment meaning and have more influence on correct sentiment expression. Also some word of review is not included in the spreadsheet. For example the word of ‘introspective’ and ‘entertaining’ in phrase 70 is not included in the spreadsheet. The spreadsheet only includes the similar word of ‘entertainment’. So it is not as good as the WordNet to deal with synonyms. However, the spreadsheet can be added some widely used words according to the common expression of the movie review. It works like the bag-of-n-grams to control n-gram but more intuitive to manage the way to extract features based on sentiment meaning.

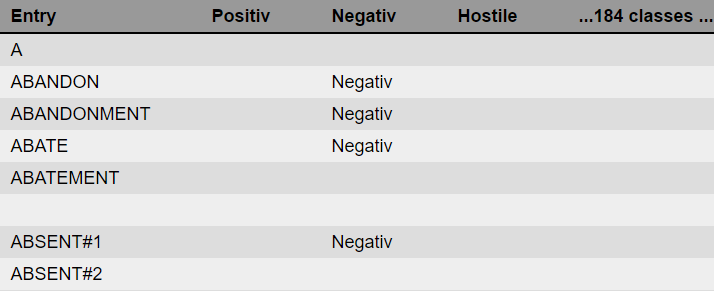


Figure3.3 Harvard General Inquirer spreadsheet

### Feature Vector

The feature vector is the algebraic model to represent text documents. Each phrase or sample is represented as a vector and each [dimension](https://en.wikipedia.org/wiki/Dimension_(vector_space)) of the vectors corresponds to a separate term. Typically terms are single words or longer phrases. This paper chooses word as the term for bag-of-n-grams, synsets as the term for WordNet, and selected column as the terms for Harvard General Inquirer, the dimensionality of the vector is the number of term occurring in the [phrases](https://en.wikipedia.org/wiki/Text_corpus).

Many papers have illustrated the good scheme called tf-idf weighting which combines the inverse document frequency to form term weight. However the dataset in this project is stored in one document, it only needs to calculate term frequency, in other words, the numbers of every synset occurrences among all phrases. Given the feature vector, this paper will explore the relationship between feature vectors and sentiment class.

Besides, another method called Word2vec is widely used to represent text documents. Word2vec is a group of related models that are used to produce word embeddings. These models are shallow, two-layer neural networks that are trained to reconstruct linguistic contexts of words. Word2vec takes as its input a large corpus of text and produces a vector space, typically of several hundred dimensions, with each unique word in the corpus being assigned a corresponding vector in the space. Word vectors are positioned in the vector space such that words that share common contexts in the corpus are located in close proximity to one another in the space. However Word2vec learns a vector representation in unsurpervised way with a simplified neural network type of structure. It is not fit for our dataset with sentiment label even though it models surrounding words by skip-gram model or continuous bag of words which is similar to bag-of-n-grams

### Dimensionality Reduction

Even though extracting the feature based on the methods above, there is still large amount of features for WordNet and bag-of-n-grams. As the feature vectors with lower dimension can focus on the useful feature and increase the classification speed, it needs to reduce the dimension of feature vectors. However, after transforming the unstructured document into feature vector, it is hard to select the important feature manually from feature vectors because there are large amount of terms, and it is impossible to tell which terms is important to the classification accuracy. For example some synset frequency has no relationship with the classification result because there are some synset with few occurrences time but strong sentiment bias. It is the same to the n-gram of bag-of-n-grams method. According to the large information set, this paper plans to reduce the dimension by selecting the classification score as feature. That is to use a fast SGDClassifier to produce a vector of decision functions separating target classes in one-versus-rest fashion and the score of each function can be regard as a feature.

The SGDClassifier implements regularized linear models with stochastic gradient descent learning: the gradient of the loss is estimated each sample at a time and the model is updated along the way with a decreasing strength schedule. As the SGDClassifier only select part of training data to train classifier, it takes less time especially when the size of sample and feature is too large in the document sentiment analysis. However the SGDClassifier is binary classification method, we need to extend the binary classification into multi-class classification.

There are two fashions which are one-vs.-rest and one-vs.-one to extend the binary classification problem into 5-class classification problem. For one-vs.-rest fashion, it needs to train 5 classifiers. Each SGDClassifier is trained for each class that the samples in the one class as positive and the samples in other class as negatives, SGDClassifier as base classifiers to produce a real-valued confidence score for its decision. For one-vs.-one fashion, it needs to train 10 binary classifiers. Each SGDClassifier is trained to distinguish the pair classes that the samples in the one class as positive and the samples in another class as negatives, The SGDClassifier as base classifiers to produce a real-valued confidence score for its decision. However the one-vs.-rest fashion often leads to imbalanced datasets the sample of one class always much fewer than that of rest class. The one-vs.-rest fashion trains a separate classifier for each different pair of labels. This is much less sensitive to the problems of imbalanced datasets but is much more computationally expensive. So to avoid the influence from imbalanced data, this paper chooses one-vs.-one fashion to do multi-class classification.

As shown in Table 3.4 and Table 3.5, there are ten scores as features for each sample. Table 3.4 shows the score calculated based on feature vectors transformed by ‘bag-of-n-grams’. Table 3.5 shows the score calculated based on feature vectors transformed by ‘WordNet’, and the score of first two phrases is the same and that of second two phrases are the same. It is corresponded to the feature vectors of ‘WordNet’ Table 3.6 shows the feature vector for Harvard General Inquirer and all the term frequency values of 4 phrases are the zero because the word of 4 phrases is not included in Harvard General Inquirer spreadsheet.

As mentioned above, ‘bag-of-n-grams’, ‘WordNet’ and ‘Harvard General Inquirer’ all have their advantage and disadvantage, we should try to make a tradeoff between few feature and correct expression. The fewer features decrease the classification cost but reduce the classification accuracy. This project has implemented four methods to generate classification input and compare their classification performance. The first method is to combine the score of ‘bag-of-n-grams’, the score of ‘WordNet’ and the feature vector of ‘Harvard General Inquirer’ into one vectors as the classification input. The other three methods are to use the score of ‘bag-of-n-grams’, the score of ‘WordNet’ and the feature vector of ‘Harvard General Inquirer’ respectively as the classification input. In next section, we will analysis the classification accuracy of each method to choose the best one to predict the sentiment type of phrases.

|  |  |
| --- | --- |
| 70 | [ 1.70055321 1.95280204 2.53904613 2.1254442 1.42879649 2.41556834 1.22819801 1.03940117 0.00802125 -1.04908995] |
| 71 | [ 1.52686571 1.81463883 2.29028249 1.88269335 1.35406841 2.28311757 0.90103354 1.00391148 -0.01445375 -1.04908995] |
| 72 | [ 1.81634488 1.63042121 1.59374428 0.66893909 1.14856618 0.72682103 -0.84384361 -0.71733832 -1.45285317 -1.71051561] |
| 73 | [ 1.93213655 1.74555722 1.34498064 0.18343738 1.03647406 0.49503218 -1.17100807 -0.85929707 -1.52027814 -1.56878154] |

Table 3.4 the score as feature for ‘bag-of-n-grams’

|  |  |
| --- | --- |
| 70 | [ 1.93284138 2.64643643 3.5332739 3.5735376 1.47329581 2.30963446 1.35196986 1.16496008 -0.63650792 -1.47843589] |
| 71 | [ 1.93284138 2.64643643 3.5332739 3.5735376 1.47329581 2.30963446 1.35196986 1.16496008 -0.63650792 -1.47843589] |
| 72 | [ 2.28021638 2.57735482 1.89143384 0.41777651 1.02492731 0.78645061 -0.99270881 -0.8756969 -1.98500737 -1.8563934 ] |
| 73 | [ 2.28021638 2.57735482 1.89143384 0.41777651 1.02492731 0.78645061 -0.99270881 -0.8756969 -1.98500737 -1.8563934 ] |

Table3.5 the score as feature for ‘WordNet’

|  |  |
| --- | --- |
| 70 | [0 0 0 0] |
| 71 | [0 0 0 0] |
| 72 | [0 0 0 0] |
| 73 | [0 0 0 0] |

Table3.6 the feature vector for Harvard General Inquirer

## Classification

There are many classifiers available for multi-class classification. Some classifier themselves can do multi- classification such as KNN, Decision Tree and Random Forest. Other classifier need to be extended from binary classifier into to multiclass classifier such as Naive Bayes SVM and SGD classifier. The advantage and disadvantage of each classifier will be analyzed as below.

The Naive Bayes classifier is based on the hypothesis of class independency. It assumes each feature is conditionality independent given class. As it’s hard to tell that whether the dimension of feature vector has the independent relationship with sentiment class in this project, the Naive Bayes classifier is not good for analyze our movie review

The SVM classifier is based on the idea to maximize the margin. It maximizes the minimum distance from the separating hyperplane to the nearest example. The basic SVM supports only binary classification. The additional parameters and constraints can be added to the optimization problem to handle the separation of the different classes. However it takes time to choose the right parameters like kernel function and soft margin to train the best model. Besides, this project uses python to implement SVM classification process and find it too slow to get the final classification result. So The SVM classifier is not recommended in this project.

The KNN classifier is considered among the oldest non-parametric classification algorithms. To classify an unknown example, the distance from that example to every other training example is measured. The k smallest distances are identified, and the most represented class by the k nearest neighbors is considered the output class label. It is important how to select the proper k value as the best parameter for this project. The larger k decreases the variance but increases the bias. For example, if increasing k too much, then the classifier no longer follow the true boundary line and observe high bias. So this paper selects k=5 as appropriate value to make accurate classification.

The SGD classifier has been mentioned above and it only select part of training data to train classifier, so it takes less time when the size of sample and feature is too large in the document sentiment analysis. For the feature vectors which only contain the score of ‘bag-of-n-grams or the score of ‘WordNet’, it only needs to continue the SGDclassifier process to get the final classification result. But for the feature vectors which include column of Harvard General Inquirer, we have to restart training process again. However compared with other classification models it is still faster to get the classification result.

The Decision Trees are a powerful classification technique. The tree tries to infer a split of the training data based on the values of the available features to produce a good generalization. The algorithm can naturally handle binary or multiclass classification problems. The leaf nodes can refer to either of the K classes concerned.

The Random Forest is the ensemble structure of Decision Tree and reduces the classification error based on Bagging and Random subspace method. Firstly, the Bootstrapping method creates datasets of the same size as original data size and random resample these datasets with-replacement. This will result in {T1, T2, ... TS} datasets. Each of these is called a bootstrap dataset. Due to "with-replacement" every dataset Ti can have duplicate data records and Ti can be missing several data records from original datasets. Secondly, the Random subspace method randomly selects sub-features out of all possible features to create S decision trees. Each Ti bootstrap dataset create a tree Ki. Then the Random Forest can get the classification result based on a weighted sum of all decision trees result. However the proper parameters of training step have great influence on classification accuracy and overfitting problem. These parameters may be the depth of trees or the numbers of the trees. Overall the Random Forest has the highest classification accuracy compared with other classification model.

## Evaluation

The accuracy of a sentiment analysis system is, in principle, how well it agrees with human judgments. This is usually measured by precision and call. According to research about human raters typically agree 79% of the time [11].Thus, a 70% accurate program is doing nearly as well as humans, even though such accuracy may not sound impressive. If a program were "right" 100% of the time, humans would still disagree with it about 20% of the time, since they disagree that much about any answer [12].

With the proceed dataset, this paper implements the classification and evaluation process by scikit-learn library. As shown in the Table 3.7, it trains four classifiers and calculates the classification accuracy for the feature vector combination of ‘bag-of-n-grams’, ‘WordNet’ and ‘Harvard General Inquirer’. The Random Forest has the highest accuracy and the model with 100 trees enjoy the accuracy of 0.65920. The other two classifiers have lower accuracy of 0.63, but it is near to the highest accuracy. Compared with the human work accuracy of 0.79 in the paper [11], these models all work well on machine learning classification.

As mentioned above, the proper parameter has great influence on the accuracy and overfitting problem. The accuracy is shown in Table 3.7, and the overfitting problem should be detected by compared the accuracy of testing set and the cross validation score of training set, in other word, to compare the values of Table 3.7 and Table 3.8. If there are much differences between the accuracy and validation score, there will be overfitting problem. Actually the overfitting problem is that the classifier is over fit the training set but ignores the testing set. The 10-fold cross validation is to split the whole training set into training part and validation part, in other word, treat validation part as the testing part of whole training set to check how the classifier fit the training set. If the 10 fold validation score is much higher than accuracy, then it means the classifier it bias to the training set.

Here the Random Forest with 100 trees have higher accuracy than that with 20 trees. Also the validation score of 0.61160 is similar to the accuracy of 0.65920 for Random Forest, so there is no overfitting problem. So the Random Forest with 100 trees is our best classifier to predict the sentiment type.

|  |  |
| --- | --- |
| Classifier | Accuracy |
| Random Forest (trees=100) | 0.65920 |
| Random Forest (trees=20) | 0.65656 |
| KNN | 0.63255 |
| SGD | 0.63619 |

Table 3.7 overall classification accuracy for combination of ‘bag-of-n-grams’ ‘WordNet’ and ‘Harvard General Inquirer’

|  |  |
| --- | --- |
| Classifier | 10-fold validation score |
| Random Forest (trees=100) | 0.61160 |
| Random Forest (trees=20) | 0.60926 |
| KNN | 0.58111 |
| SGD | 0.58142 |

Table3.8 10-fold cross validation for combination of ‘bag-of-n-grams’ ‘WordNet’ and ‘Harvard General Inquirer’

To compare the performance of different feature vectors it also implement 10-fold cross validation for ‘bag-of-n-grams’, ‘WordNet’and ‘Harvard General Inquirer’. As the Kaggle testing dataset in unseen, and it allows to submit for limit time to check classification accuracy. Here it uses 10-fold cross validation score instead of classification accuracy to check classification performance. As shown in Table3.8, Table3.9, Table3.10 and Table3.11, the feature vector combination in Table3.8 is the best document presentation model. The ‘bag-of-n-grams’ is better than the ‘WordNet’ and the ‘WordNet’ is better than the ‘Harvard General Inquirer’.

Actually this comparison only focuses on our dataset. The ‘WordNet’ has lower score because it combines the sample with similar expression structure into one sample but ignore they have different label, in other words, the ‘WordNet’ is not fit for the phrases sentiment meaning separated by Treebank. The ‘Harvard General Inquirer’ has lower score because spreadsheet fails to include the word in our movie review. The ‘bag-of-n-grams’ has the problem to include useless features, but it does not influent much about classification accuracy of our dataset, maybe our dataset does not include much useless word like “the”, “a”. The following work will improve the sentiment expression for both ‘WordNet’ and ‘Harvard General Inquirer’.

|  |  |
| --- | --- |
| Classifier | 10-fold validation score |
| Random Forest (trees=100) | 0.60574 |
| Random Forest (trees=20) | 0.60342 |
| KNN | 0.57753 |
| SGD | 0.59464 |

Table3.9 10-fold cross validation for ‘bag-of-n-grams’

|  |  |
| --- | --- |
| Classifier | 10-fold validation score |
| Random Forest (trees=100) | 0.58627 |
| Random Forest (trees=20) | 0.58259 |
| KNN | 0.54842 |
| SGD | 0.56402 |

Table3.10 10-fold cross validation for ‘WordNet’

|  |  |
| --- | --- |
| Classifier | 10-fold validation score |
| Random Forest (trees=100) | 0.52108 |
| Random Forest (trees=20) | 0.52119 |
| KNN | 0.46514 |
| SGD | 0.48895 |

Table 3.11 10-fold cross validation for ‘Harvard General Inquirer’

# conclusion

This paper has analysed document sentiment area and implement the classification process for sentiment type by python. Specifically it analyses how to separate the whole document into phases with sentiment label, how to present the document by feature vectors and how to classify the phrases into sentiment class. The feature vectors of ‘bag-of-n-grams’ ‘WordNet’ and ‘Harvard General Inquirer’ are all available to present the document sentiment information but the combination of these three method is the best presentation model. Then it trains Random Forest, KNN and SGD classifier based on these feature vectors and find Random Forest is the best classifier to predict sentiment class. Here are some following works need to be done, for example exploring a better model to present document information, choosing better parameters of Random Forest or other classifier to improve the classification accuracy.

References

1. Pang, Bo, and Lillian Lee. "Opinion mining and sentiment analysis." Foundations and Trends® in Information Retrieval 2.1–2 (2008): 1-135.
2. Liu, Bing. "Sentiment analysis and opinion mining." Synthesis lectures on human language technologies 5.1 (2012): 1-167.
3. <http://www.cs.cornell.edu/people/pabo/movie-review-data/>
4. <https://www.kaggle.com/c/sentiment-analysis-on-movie-reviews>
5. Pang, Bo, and Lillian Lee. "Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales." Proceedings of the 43rd annual meeting on association for computational linguistics. Association for Computational Linguistics, 2005.
6. Socher, Richard, et al. "Recursive deep models for semantic compositionality over a sentiment treebank." Proceedings of the conference on empirical methods in natural language processing (EMNLP). Vol. 1631. 2013.
7. <https://nlp.stanford.edu/sentiment/>
8. <http://www.wjh.harvard.edu/~inquirer/>
9. <http://www.wjh.harvard.edu/~inquirer/homecat.htm>
10. <http://sentiment.christopherpotts.net/lexicons.html#inquirer>
11. Ogneva, Maria. "How companies can use sentiment analysis to improve their business." Mashable (2010).
12. Roebuck, Kevin. Encryption: High-impact Strategies-What You Need to Know Definitions, Adoptions, Impact, Benefits, Maturity, Vendors. Tebbo, 2011.

1. [↑](#footnote-ref-1)