Sentiment Analysis using Term Frequency Inverse Document Frequency

Hieu Tran
Intermediate Progress for Final Project
CS310 - Machine Learning - Spring 2019

Introduction

Nowadays, the Internet is not only a valuable source of information but also a place for people to express emotions, feeling and thoughts about daily life problem. With the rapid growth of text data, text mining has increasing practical impact in real life.

Unlike the field of traditional data mining, instead of focusing on user activities such as purchase history, accessing time, etc., the field of Sentiment Analysis involves extracting, understanding, classifying and presenting the emotions and opinions expressed by users. Therefore, the Sentiment Classification problem can be seen as a combination of data mining and natural language processing. Up to now, solving the Sentiment Classification problem has many different approaches and different levels of processing. Our approach is based on traditional term weighting functions - Term Frequency-Inverse Document Frequency (TF-IDF) weighting scheme. This method utilizes a combination of the overall frequency count of terms and proportional presence count distribution.

The subject of this study is the sentiment analysis of reviewers toward food restaurants and their food based on reviews and ratings on Yelp, which is a website specializing in ratings in the U.S. The dataset is collected through web a crawling process from 100 food restaurants in San Francisco. The paper includes an introduction to Text Data Mining and Sentiment Classification in general, discusses different applications of Text Mining, define the Term Frequency-Inverse Document Frequency (TF-IDF) technique, provides context on the history and usage of TF-IDF, describes the method for web crawling and pre-processing data, and analyzes different machine learning algorithms used for this project. Finally, this study evaluates the effect of using TF-IDF scheme while checking the performance of the trained model on different machine learning algorithms.

Overview

Sentiment Classification Problem

Motivation and Problem Description With the rapid development of the internet, people can sit for hours every day reading newspapers, listening to music, sharing emotions, writing notes and commenting. Hidden in those comments are users feelings such as happiness, sadness, love, hate. From an e-commerce standpoint, Accurate classification of user sentiment will help advertisers produce better marketing content to attract customers. For example, knowing a person is tired, advertisers can suggest some energy drinks, entertainment games, or simply a soothing song to serve that customer. Comprehending these emotions, although natural to human, proves to be a major challenge for computers.

In order to address these issues, this paper would explore the topic of "User Sentiment Classification"

Prior Works Due to some characteristics of the text data on the Internet such as the limited number of characters or the wide variation of possible reaction to the same content, the sentiment classification on the Internet is a challenging problem. Previous studies were mainly focused on vocabulary - Lexicon Based Method (Taboada et al. 2010) and Machine Learning Based Methods (Pang, Lee, and Vaithyanathan. 2002). The lexicon-based approach involves calculating orientation for a document from the semantic orientation of words or phrases in the document. The machine-based approach involves building classifiers from labeled instances of texts or sentences. For Lexicon-based methods, the results depend heavily on the degree of words that express emotions. For machine-based learning methods such as SVM, Naive Bayes, the results depend a lot on how to extract features.

Based on the characteristics of my data set, in which all the reviews have been labeled as either negative or positive, the machine-based approach is an appropriate method. Besides that, the TF-IDF feature weighting scheme would be utilized as it shows the highest classification accuracy among other information retrieval schemes (Paltoglou and Thelwall. 2010).

Data Processing Model For Sentiment Analysis There are many different approaches for solving this problem, however, the study group led by Pravesh Kumar Singh has developed general model in the International Journal on Soft Computing-IJSC 2014 as seen in Figure 1(Singh et al. 2014).

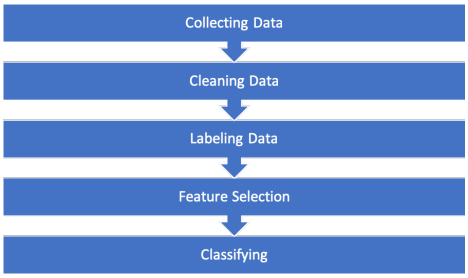

Figure 1: Data processing model for sentiment classification

- Collecting data: In this step, text data is collected from a website. To perform this step, we can either use the database of the website or use Crawler or Agent to perform crawling data automatically. The collected data has HTML or txt format.
- Cleaning Data: In terms of structure, the data obtained in the first step is in HTML or txt format, so the cleaning step filters only the necessary information such as user comments, ratings, etc. In addition, the collected user comments may not satisfy grammatical or semantic requirements. This step also removes inappropriate or modified comments to ensure the data collected is appropriate for labeling or feature selection in the following steps.
- Labeling Data: With the Lexicon Based approach, this step uses the data from that lexicon to match or label the words from the corresponding text in the dictionary. With the Machine Learning Based approach, the data is labeled either based on the features of the text such as labeling sentence, words or emoticons.
- Feature Selection: In machine learning, the more features the higher the accuracy; however, the excessive amount of features makes the training process and the classification process longer and take up more memory. Therefore, it is necessary to choose from a set of features a smaller subset that still ensures the accuracy of the classification process. Machine learning algorithms use techniques such as n-grams or POS tagging to train and test with the standard data set from the previous steps.
- Classifying: Input data after preprocessing steps go through the classifier using machine learning algorithms. With the Lexicon-based approach, this step applies grammar or semantic rules to calculate the final weight, then decides the semantic meaning of the comment.

Machine Learning Based Approaches for Solving Sentiment Analysis

The Machine Learning objective is to build an algorithm to identify new input data from previously trained data. There are two main stages in data classification:

- Training stage: Our model learns to associate a particular input (i.e. a text) to the corresponding output (tag) based on the training data set.
- Prediction stage: Classifying new text based on the model that has been built from the training phase.

In both the training and prediction stages, data must go through two main phases: Data pre-processing and Feature selection. Specifically, these steps are as follows:

- Data Pre-processing: In this step, data is removed or modified to suit the objectives of the classifier. With natural language processing, the raw data contains redundant information or spelling and grammatical errors. Thus, raw data need to be cleaned to eliminate excess information and fix existing errors to ensure that the data meets the language requirements. In the next phase, the data are tokenized into individual linguistic units such as sentence splitting and word splitting. The information on these units greatly affect the semantics of the sentence or text and is used in Feature Selection to build the classification model.
- Feature Selection: In this step, linguistic components are selected appropriately so that a text can be vectorized. Specifically, a document is represented as follows:

$$Doc_i = \{v_1, v_2, \ldots, v_n\}$$

where $v_1, v_2, \ldots, v_n$ are feature vectors.

After the Feature Selection phase, we can apply different classified algorithms such as Naive Bayes or Support Vector Machine (SVM).

Classification using Naive Bayes Naive Bayes is a popular classification technique in supervised machine learning for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. Naive Bayes is widely used in solving text categorization problems, building automatic spam filter, or in sentiment analysis because of its simplicity as well as accuracy.

The basic idea of the Naive Bayes approach is to use conditional probabilities between features and labels to predict the probability of a document's classification. The important point of this method is that it assumes all features in the text are independent of each other. That assumption makes the calculation of Naive Bayes more efficient and faster than other methods because it does not use the combination of features to predict the label. The predicted results are affected by the size of the data set and the quality of the characteristic space. Despite their naive design and apparently oversimplified assumptions, Naive Bayes classifiers have worked quite well in many complex real-world situations

Naive Bayes algorithm based on Bayes theorem is stated as follows:

$$P(Y|X) = \frac{P(X \cap Y)}{P(X)} = \frac{P(X|Y)P(Y)}{P(X)}$$

Applied to the classification problem:
Let D be a training set of tuples and their class labels, where each tuple X is represented by n attributes, $X = (x_1, x_2, \ldots, x_n)$ and there are m classes, $C_1, C_2, \ldots, C_m$. According to Bayes theorem:

$$P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)}$$

By the assumption of independent feature:

$$P(X|C_i) = \prod_{k=1}^{n} P(x_k|C_i)$$

where:
$P(C_i|X)$ is the posterior probability that the sample X belong to class C_i
$P(C_i)$ is the prior probability of class C_i
$P(x_k|C_i)$ is the posterior probability that the k^{th} attribute is x_k given X belongs to class C_i
Then, the step for Naive Bayes Classification:

- Training Naive Bayes classifier: calculating $P(C_i)$ and $P(x_k|C_i)$
- Predicting: Given sample $X^{new} = (x_1, x_2, \ldots, x_n)$, X^{new} is classified to the class that maximize the posterior probability the sample X belong to:

$$\max_{C_i \in C} \left(P(C_i) \prod_{k=1}^{n} P(x_k|C_i) \right)$$

Classification using Support Vector Machine (SVM) The Support Vector Machine (SVM) is a classification method proposed by Cortes Vapnik [17]. The main idea of the algorithm is using nonlinear mapping to transform the original training data into a higher dimension and then search for the hyperplane that linearly separates the data points of one class from another.
Consider an example of the classification problem of Figure 5 where we have to find a straight line (decision boundary) so that the left side is all red points and the right side is all the blue points. The problem that uses this line to divide is called linear classification.

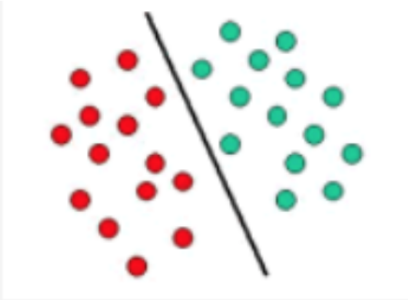

Figure 2: SVM example

We have the equation for the hyperplane as follow:

$$y(x) = w^T \Phi(x) + b \quad (1)$$

where:
$w \in R^m$ is the weight vector
$b \in R$ is a scalar
$\Phi(x)$ is the feature vector. Φ is the mapping function that transform the input vectors into feature vectors.

The input data set consists of N samples input vector $\{x_1, x_2, \ldots, x_N\}$, with corresponding label $\{t_1, t_2, \ldots, t_N\}$ in which $t_n \in \{-1, 1\}$.

Assume the data samples can be completely classified into two classes, there exists w and b in equation 1 such that $y(x_n) > 0$ for samples that have label $t_n = +1$ and $y(x_n) < 0$ for samples that have label $t_n = -1$. Hence, we have $t_n y(x_n) > 0$ for all training samples.

SVM attempts to solve this problem using a concept called margin. The selected margin is the distance from the separating line to the nearest points as can be seen in Figure 3.

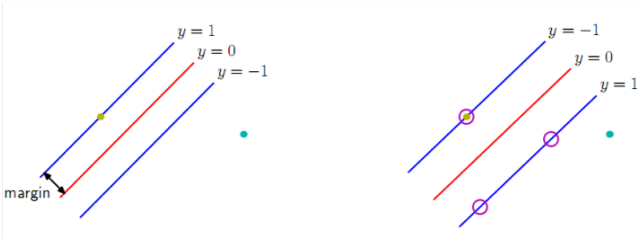

Figure 3: SVM margin

The goal of the SVM is to maximize the margin (maximize the linear separation of data points).

Model and Steps in Sentiment Classification

General Model

The input of the model is a collection of review on restaurants from the Yelp website. The Crawler collects review data (in the form of text) and the rating of each review (on scale from 1 to 5) and stores them in the database. The raw text data is tokenized and stripped off stop words (words that have no meaning in expressing emotions), special characters, white characters, and corrected misspellings. Raw text data is now called processed data. Within the scope of the paper, based on the rating star, review text data is categorized into one of two classes: ”Positive” and ”Negative”.

The next job is to build the feature vector for the processed data set. The purpose of this is to represent text data in the form of the feature vector. Vector is a standard representation for the next step using classification techniques such as Naive Bayes and SVM. This project builds the feature vector using the TF-IDF technique.

After having the feature vector, we perform classification with Naive Bayes and SVM techniques using the sklearn library.

Finally, we conduct an evaluation and comparison and make some conclusion from the achieved results.

Collecting Data

The project performs data collection on ”www.Yelp.com”, which is a website specializing in rating of U.S businesses. Specifically, we collect data from 100 restaurants in the San Francisco area. For each restaurant, around 200 reviews and ratings are collected.

In order to perform the web data collection, this project uses the selenium library and Chrome driver. After crawling, review text data and star rating data are stored into two separate files with corresponding order as in Figure 4:

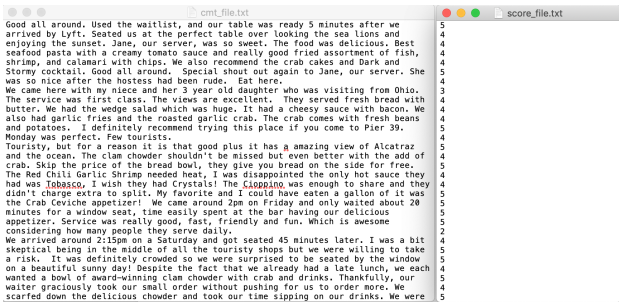

Figure 4: Crawling Data is stored in database

Preprocessing Data

In Natural Language Processing, text pre-processing is extremely important as it corrects the text content and removes excess parts. This contributes greatly to increasing the accuracy of the classifier.

Data preprocessing consists of 2 main parts:

- labeling: This part removes contents that are objective or unrelated to user moods or are misleading. This work is often done manually. Thanks to the collected rating score, we don't need to label the review data by hand.
  We have collected: 6414 positive reviews and 1470 negative reviews
- Standardizing data: In this part, the project proceeds to correct spelling errors, making standard abbreviations. Standardized operations are performed as in Figure 5. Here, abbreviations or unorthodox languages have been standardized. Also, the entire sentence is converted to lowercase characters and punctuation marks are removed for later processing.

Example		Steps
Review before pre-processing	I luv the Food here so much, such great experience!!!	- Convert text to lowercase - Remove numbers, whitespace - Fix spelling: luv → love - Remove punctuation
Review after pre-processing	I *love* the *food* here so much such great experience	

Figure 5: Example of preprocessing data

Feature Selection and Vectorization

In order to apply the Naive Bayes or SVM classifier, the input data need to be converted into proper case format. Specifically, from the pre-processed data, for each review, we must choose right features and convert them into vectors. During this project, we use the Vector Space Model, specifically the term frequency-inverse document frequency model for feature selection.

Vector Space Model Vector space model is an algebraic model for representing text documents as vectors of identifiers, such as, for example, index terms:

$$d_j = (w_{1,j}, w_{2,j}, \ldots, w_{t,j})$$

Each dimension corresponds to a separate term (the definition of "term" depends on the application, during this project, terms are single words). If a term occurs in the document, its value in the vector is non-zero . Several different ways of computing these values, also known as (term) weights, have been developed. One of the best-known schemes is TF-IDF weighting.

Term Frequency-Inverse Document Frequency In information retrieval, TF-IDF is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus.

- TF: Term Frequency measures how frequently a term occurs in a document. Since every document is different in length, it is possible that a term would appear much more times in long documents than shorter ones. Thus, the term frequency is often divided by the document length (also know as the total number of terms in the document) as a way of normalization:

$$TF(t) = \frac{\text{Number of times term t appears in a document}}{\text{Total number of terms in the document}}$$

- IDF: Inverse Document Frequency, which measures how important a term is. While computing TF, all terms are considered equally important. However, it is known that certain terms, such as "is", "of", and "that", may appear a lot of times but have little importance. Thus we need to weigh down the frequent terms while scaling up the rare ones, by computing the following:

$$IDF(t) = log\left(\frac{\text{Total number of documents}}{\text{Number of documents with term t in it}}\right)$$

- TF-IDF is calculated by:

$$TF - IDF(t) = TF(t) * IDF(t)$$

This project performs the feature selection and vectorization with TF-IDF weighting by using the `TfidfVectorizer` module from the sklearn library.

Classification using Naive Bayes and SVM

After the feature selection and vectorization step, we use the built-in Naive Bayes or SVM classifier in the sklearn library for training and predicting data. We split 80% of data samples for training phase and 20% for testing.

Results and Discussion

In order to evaluate the performance of classifiers, we use the F-score measurement. The value of F-Score depends on the value of Precision and Recall. In particular, Precision measures the accuracy of the classifier, determined by the number of correctly classified reviews over the total number of reviews classified to that class. Recall is determined

by the number of correctly classified reviews over the total number of reviews that are actually in that class. F-score is determined by Precision and Recall (the higher the values of these measurements, the better the classification of the classifier). Specifically:

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F - score = 2\frac{Precision * Recall}{Precision + Recall}$$

where:

- TP (True Positive): The number of reviews that is correctly classified into class x.
- FP (False Positive): The number of reviews that is incorrectly classified into class x.
- FN (False Negative): The number of reviews belonging to class x but are incorrectly classified into other class.

The Table 1 and Table 2 shows the result for trained models using Naive Bayes and SVM on testing 20% of data set.

label	precision	recall	f1-score	support
0	1.00	0.06	0.12	296
1	0.82	1.00	0.90	1281

Table 1: Results for classifying using Naive Bayes before over-sampling

label	precision	recall	f1-score	support
0	0.81	0.59	0.68	311
1	0.91	0.97	0.93	1266

Table 2: Results for classifying using SVM before over-sampling

We can see that for both models, the f1-score for result on label 1 is really good while the f1-score for label 0 is bad, extremely low for Naive Bayes model in Table **??**. This is because our data set is skew, in which the number of samples in positive class outnumbers the numbers of sample in the negative class.

We try address this issue by using an over-sampling method, in which the number of data samples for label 0 is increased by add one copies of each negative review into our initial data set. After applying this method, the results are shown in 3 and Table 4:

label	precision	recall	f1-score	support
0	0.98	0.45	0.61	581
1	0.80	1.00	0.89	1290

Table 3: Result for classifying using Naive Bayes after over-sampling

label	precision	recall	f1-score	support
0	0.87	0.93	0.90	565
1	0.97	0.94	0.95	1306

Table 4: Result for classifying using Naive Bayes after over-sampling

In this case, the f1-scores for both labels in two models have improved significantly, especially for label 0. The results also indicates that the model trained by SVM show better preformance than the model trained by Naive Bayes.

Conclusion and Future Work

To solve the problem of sentiment classification, the paper has collected user review data toward food restaurants, applied the vector space model, specifically the TF-IDF model and different classifiers to build the best model for addressing the problem. In particular, the project has achieved the following results:

- Survey different approaches to solve the problem of sentiment classification.
- Develop applications and apply feature selection and classification models.
- Analyze, evaluate and compare the results of the application of different classifiers. Specifically, SVM classifier is better than Naive Bayes classifier in solving this problem.

Besides these achievements, this project also has some drawbacks during the research process:

- The objective of labeling. We label the class based on the rating scores associated with the review comments. However, the rating score may not accurately represent the attitude of the reviewer and each reviewer has a different standard in rating.
- The skewness of the data, where the number of samples in positive class outnumbers the numbers of sample in the negative class.
- The limit number of label class. The paper only focuses on classifying 2 labels: ”positive” and ”negative”.
- Issue of detecting sarcasm or hyperbole is still a major problem for Natural Language Processing techniques.

For future improvement, we suggest the following:

- Collecting more text data sample.
- Expand the classification to 6 basic label classes: fun, sad, angry, surprised, hate and fearful.

References

B. Pang, L. Lee, and S. Vaithyanathan. 2002 Thumbs up?: sentiment classification using machine learning techniques, in Proceedings of the ACL-02 conference on Empirical methods in natural language processing - EMNLP 02, Not Known, 2002, vol. 10, pp. 7986.

G. Paltoglou and M. Thelwall, A Study of Information Retrieval Weighting Schemes for Sentiment Analysis, p. 10.

J. Han, M. Kamber and J. Pei. 2012 Data mining: concepts and techniques. Amsterdam: Elsevier/Morgan Kaufmann.

M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede, Lexicon-Based Methods for Sentiment Analysis, Computational Linguistics, vol. 37, no. 2, pp. 267307, Jun. 2011.

P. K. Singh and M. Shahid Husain, Methodological Study Of Opinion Mining And Sentiment Analysis Techniques, International Journal on Soft Computing, vol. 5, no. 1, pp. 1121, Feb. 2014.

S. Marsland .2015 Machine learning: an algorithmic perspective. Boca Raton: CRC Press.