Opinion Mining of Film Reviews Using Natural Language Processing

Mahesh Koppala, Rahul Guda, Suryateja
University of Texas at Arlington (UTA), Texas, United States

Abstract

Sentiment analysis is a field of study which employs techniques like natural language processing to analyse any text and give its emotion. With the vast advancements in technology, internet has found its way into every corner of the world thus generating ginormous amounts of data every day. For instance, there are millions of user reviews on movies which we can analyse and extract useful insights from them with the help of sentiment analysis. When it comes to finding the polarity of a sentence nothing can beat humans, however, when dealing with huge amounts of data, machines always outperform humans. Hence, it is better to train computers in such a way that their accuracy will come very close to that of humans. This paper uses different machine learning approaches like Multinomial Naïve Bayes, SVM, and KNN for sentiment analysis of movie reviews using natural language processing and aims to increase the accuracy with the help of various techniques.

Introduction

During these digital times, there is a lot of content for us to watch online, and we are always confused about what movie to watch. However, we can overcome this issue by looking at the vast number of reviews and ratings left by other people online. For a human, just by looking at the review, we can detect the sentiment of it. But, for a computer, it's not possible unless we train them.

With the internet being available to many people, the amount of digital foot-print everybody is leaving behind is huge. Instead of piling this data, one has to put this data to good use such that it is beneficial to all parties. Sentiment analysis does exactly this, and due to this very reason, there is a never-before rise in academic interest in this particular field of study in the past few years. Despite so many advancements in computing power and the approaches we employ to extract opinions, there is always room for improvement, and nothing is perfect.

Sentiment analysis is not just limited to finding the polarity of a movie review, but it can do much more. Just throw data it and it will spit out useful insights. For instance, a company might want to know the user feedback for a product. One can conclude whether the product was a hit or a miss by analysing the user reviews and mining the polarity of them. Coming back to the movie reviews, we used different classification methods like Multinomial Naive Bayes, SVM, and KNN on the given large movie dataset and concluded which method best suited the cause. We evaluated the performance of these methods using techniques like confusion matrix, classification report, and plain accuracy.

Dataset Description

The dataset we used for our project is ‘Large Movie Review Dataset v1.0’ which we have obtained from the Stanford University website. The dataset has a total of 50,000 reviews which are divided into training and testing sets. Each set consists of 25,000 reviews and are in-turn divided into positive and negative reviews. Each review is stored in a different text file, and any review with a score less than or equal to 4 is considered as negative and greater than or equal to 7 is positive.

The given data is in the form of text files. Hence, for convenience purposes, we extracted all the reviews from 50,000 text files and created two CSV files, each containing training, and testing data respectively. These datasets now have two attributes, namely, review and score. Any review with a rating less than or equal to 4 is assigned a score of '-1' and '1' is assigned to any review with a rating of greater than or equal to 7. Other than that, the data is already cleaned and is ready for our project.

Project Description

1. Description:

Natural Language Processing (or NLP) applies Machine Learning models to text and language to classify the data. The main goal of our project is to perform opinion mining on the given large movie data set to know if the review is positive or negative. The first step involves data cleaning using techniques like tokenization, stop words removal,

and stemming. The next step is feature extraction where we convert the words into numerical representation using models like Bag of Words and Tfidf.

Finally, we choose the optimal classifier to fit the model and to predict the output. Confusion matrix is used as an evaluation metric to calculate the accuracy. Different classification algorithms like multinomial naïve bayes, SVM and KNN are used which has given different results.

Therefore, based on the accuracy that we get using feature extraction models, appropriate classifier must be chosen. Accuracy is measured using confusion matrix and classification report.

Our project has 4 phases:
1. Data Pre-processing
2. Feature Extraction
3. Classifiers
4. Evaluation

a. Data Preparation

Tokenization:
Tokenization is the process of dividing a text into words, symbols and phrases which are known as tokens. Processing of the raw data happens at the token level. Tokenization is the first step while modelling the text data.

Removing unwanted data:
Most of the real word data is not clean, data pre-processing is a main step to convert the raw data into a format so that machine learning model can be applied. In text files, unnecessary data can be words, symbols, punctuations, and html tags. Stop words should also be removed which does not add any meaning to the sentence. Removing stop words can result in efficient processing of the text. Necessary libraries from the NLTK can be used for this purpose.HTML tags and other unnecessary symbols and punctuations should also be removed.

Stemming:
Stemming is the process of reducing the word to its base or root form. For instance, filtered and filtration can be reduced to its stem form filter which will help in analyzing the text faster and reduces the space occupied. However, based on the data, stemming technique must be used because in some cases, it might decrease efficiency.

b. Feature Extraction

Bag of words model:
The bag of words model is one of the most used feature extraction method. In this model, text is represented as a bag of words and the occurrence of each word is used as a feature for training the classifier. This model only identifies whether a word is present in the document or not but not where in the document. Bow model involves three steps which are collecting the data, designing the vocabulary, creating the document vectors.

Tfidf:
Tfidf stands for term frequency and inverse document frequency. Term frequency calculates how many times a word is appearing in the current document. Inverse document frequency calculates how rare the word is spread across documents. It evaluates how important a word is to document. Term frequency depends on the document length as in large documents, the frequency of a word appearing may be high.

$$\text{Tfidf}(t, d) = \text{tf}(t, d) * \log(N/(df + 1))$$

```
#this is a function to create Bag of words model
def bowVectorizer(processed_train_reviews, processed_test_reviews):
    vectorizer_bow = CountVectorizer(analyzer = "word", max_features = 5000)
    train_featVec_bow = vectorizer_bow.fit_transform(processed_train_reviews)
    train_featVec_bow = train_featVec_bow.toarray()
    #type(train_featVec_bow)
    #print(train_feature_vectors[2500])
    test_featVec_bow = vectorizer_bow.transform(processed_test_reviews)
    test_featVec_bow = test_featVec_bow.toarray()
    return train_featVec_bow, test_featVec_bow

#this is a function for Tfidf
def tfidfVectorizer(processed_train_reviews, processed_test_reviews):
    vectorizer_tfidf = TfidfVectorizer(min_df = 5,max_df = 0.8,sublinear_tf = True,use_idf = True)
    train_featVec_tfidf = vectorizer_tfidf.fit_transform(processed_train_reviews)
    test_featVec_tfidf = vectorizer_tfidf.transform(processed_test_reviews)
    return train_featVec_tfidf, test_featVec_tfidf
```

Code Snippet for Feature Extraction

c. Classifiers

Multinomial Naïve-Bayes:
Multinomial Naïve Bayes classifier algorithm is generally used for classification with discrete features such as word count because it normally requires integer feature counts. Probability of each word per class is determined, then we will combine the probability distribution with fraction of documents belonging to each class. If the word appears again and again, the probability of it appearing goes up. Laplace smoothing must be performed if a word has zero count. To take stop words into account, we must add inverse document frequency on each word.

```
def naiveBayes(train_featVec, test_featVec):
    model = MultinomialNB()
    model.fit(train_featVec, y_train)
    y_pred = model.predict(test_featVec)
    return y_pred
```

Naïve Bayes Implementation

Support Vector Machines:
Support Vector Machines (SVM) finds the best hyperplane which acts as the boundary between the vectors and in this case, it divides the positive review and negative review areas. So, to apply SVM effectively, texts should be transformed into vectors. SVM finds where to draw the best line to categorize into positive review subspace and negative subspace. Support vectors represents a set of coordinates in each space. We have got the highest accuracy of 87% by using SVM.

```
def svm(train_featVec, test_featVec):
    model = LinearSVC()
    model.fit(train_featVec, y_train)
    y_pred = model.predict(test_featVec)
    return y_pred
```

SVM Implementation

K Nearest Neighbor:
K Nearest Neighbor algorithm classifies the data by finding the k nearest matches in training data. We will identify the k nearest neighbors which has the highest score among the documents. In KNN, all the computation is deferred till prediction. We convert each document text into a list of sets of each word. Similarity score provides the score between two texts/documents.KNN algorithm has got the least accuracy.

```
def knn(train_featVec, test_featVec):
    model = KNeighborsClassifier(n_neighbors = 3)
    model.fit(train_featVec_tfidf, y_train)
    y_pred = model.predict(test_featVec_tfidf)
    return y_pred
```

KNN Implementation

d. Evaluation

Three different methods have been used for evaluating the classification algorithms. First one is the simple accuracy method which determines how often the algorithm correctly predicts the result. Second method is the confusion matrix. True positives, False positives, False negatives, False positives are used in the confusion matrix. Third one is the classification report which consists of precision, recall and F1 score.

Accuracy:
This is a simple method that is based on how many times the classifier classified correctly. It needs the predicted values and the testing values as the input data. This method gave the scores of 87% for Support Vector Machine (SVM), 84% for Multinomial Naïve Bayes, and 67% for K Nearest Neighbors (KNN).

Confusion Matrix:
Confusion Matrix is the most used evaluation metric. It clearly says how many times the classification algorithm has predicted the result correctly and it compares with the actual result in both the sides such as positive and negative. Basically, confusion matrix predicts the actual target values with the values predicted by the machine learning model.
Columns represent the actual values of the target variable and rows represent the predicted values of the target variable.
The confusion matrix method fetched the following results.

```
Confusion matrxi of Multinomial Naive Bayes is  [[10828  1672]
 [ 2346 10154]]
Accuracy of confusion matrix for Multinomial Naive Bayes is  0.83928


Confusion matrxi of SVM is  [[11041  1459]
 [ 1705 10795]]
Accuracy of confusion matrix for SVM is  0.87344


Confusion matrxi of KNN is  [[8606 3894]
 [4242 8258]]
Accuracy of confusion matrix for KNN is  0.67456
```

Confusion Matrix Results

Classification report:
A classification report involves precision, recall, and f1 score. Precision means when the algorithm has predicted yes, how often it is correct. Recall is the true positive rate and whereas F-score is the weighted average of the precision and recall. All these three values can be used to determine the accuracy of the classification algorithm.

```
Report of Multinomial Naive Bayes is
              precision    recall  f1-score   support

          -1       0.82      0.87      0.84     12500
           1       0.86      0.81      0.83     12500

    accuracy                           0.84     25000
   macro avg       0.84      0.84      0.84     25000
weighted avg       0.84      0.84      0.84     25000


Report of SVM is
              precision    recall  f1-score   support

          -1       0.87      0.88      0.87     12500
           1       0.88      0.86      0.87     12500

    accuracy                           0.87     25000
   macro avg       0.87      0.87      0.87     25000
weighted avg       0.87      0.87      0.87     25000


Report of KNN is
              precision    recall  f1-score   support

          -1       0.67      0.69      0.68     12500
           1       0.68      0.66      0.67     12500

    accuracy                           0.67     25000
   macro avg       0.67      0.67      0.67     25000
weighted avg       0.67      0.67      0.67     25000
```

Classification Report Result

2. Main References Description

In [1] we learned about how sentimental analysis is used for text classification and end to end way of applying the machine learning models to get the actual output.

In [2] and [3] we came to know about the different classification algorithms and how to implement them.

In [4] and [5], we came to know about the data pre-processing techniques and feature extraction models.

3. Difference in Approach between our Project and Projects of Reference:

In the references , we found that they only used one feature extraction and only confusion matrix has been used as the evaluation metric. But here in our project we are using two feature extraction models namely the Bag Of Words and Tfidf. In the references mostly only Naïve Bayes Classification algorithm was used. Here in our project, we used classification algorithms namely Naïve Bayes, KNN and SVM. Out of these three classification algorithms, we noticed that SVM has the highest accuracy. Therefore we chose SVM combined with Tfidf as the best algorithm.

4. Difference in Accuracy between our Project and Projects of Reference:

When compared to the references , the results in our project have better accuracy. Inorder to find the accuracy we utilized three different evaluation methods. A single evaluation metric cannot give the right accuracy. Therefore we tried to use different evaluation metrics to chose the right classification model and the model that has highest accuracy is the best classification model.

Classifier	Accuracy
Multinomial Naïve Bayes	84%
SVM	87%
KNN	67%

Evaluation Results

5. List of Contributions:

1. Utilized data pre-processing techniques like stop words removal and tokenization.
2. Implemented the TF-IDF.
3. Implemented the Bag-of-Words model.
4. Utilized three different types of classifiers algorithms
5. Used the methods like confusion matrix, simple accuracy, and classification report inorder to evaluate the three classifiers.

Analysis

1. What did I do well:

In the concept of Sentimental analysis and opinion mining, choosing the best classifier algorithm is vital for a given dataset. Here in our project we chose three classifier algorithms and we evaluated them to find out the best and optimal classifier for the dataset. We used the references and learnt some techniques and we found out that SVM is the best among the three classifiers as it has the best accuracy among the three.

2. What could I have done Better:

We learnt a lot from different sources and references we found. And we also used most of the techniques and we found out the algorithm with the best accuracy amond the three classifiers. But we believe that there still so much to learn. There are many other data pre-processing methods and feature extraction techniques too. We need to go through them to increase our accuracy.

3. What is Left for Future:

Accuracy of the algorithms is the most vital thing. Inorder to classify the test data, the algorithms need huge amount of training data. Inorder to reduce that amount of training data, we implement complex algorithms. These algorithms will also reduce the time taken to learn and the resources. In the Natural Language Processing tasks, deep learning is proven to be one of the best. Therefore we say that doing more research in the deep learning field and also in neural networks etc as they may extract the sentiment from the texts in a better way.

Conclusion

We found that SVM classifier has the best accuracy among the three algorithms. It performed well with an accuracy of 87%. But we think the accuracy can still be improved further by using other data pre-processing techniques and by using different feature extraction methods. Also researches should be done in fields like deep learning and neural networks inorder to increase the accuracy in the opnion mining area.

References

[1]. Shabina Dhuria and Harmunish Taneja, “A Survey on Sentiment Analysis and Opinion Mining”, International Conference on Computing Sciences, pp. 124-128, Nov. 2013, Elsevier.

[2]. P. Rudy and Th. Mike, “Sentiment analysis: A combined approach” , International Journal of Informatics, 3, pp. 143–157, 2009.

[3]. Jagtap V. S. and Pawar Karishma, “Analysis of different approaches to Sentence-Level Sentiment Classification”, International Journal of Scientific Engineering and Technology, 2(3), pp. 164-170, April 2013.

[4]. Özgür Genç , “The basics of NLP and real time sentiment analysis with open source tools”, Towardsdatascience.

[5]. Mananmongia, “NLP Analysis of Restaurant Reviews”, GeeksforGeeks.