Spring 6-1-2016

Multi Faceted Text Classification using Supervised Machine Learning Models

Abhiteja Gajjala
San Jose State University

Multi Faceted Text Classification using Supervised Machine Learning Models

A Writing Project Report

Presented to

The Faculty of the Department of Computer Science

San Jose State University

In Partial Fulfillment

Of the Requirements for the Degree

Master of Science

By

Abhiteja Gajjala

MAY2015

The Designated Project Committee Approves the Project Titled

Multi-Faceted Text Classification using Supervised Machine Learning Models

By

Abhiteja Gajjala

APPROVED FOR THE DEPARTMENTS OF COMPUTER SCIENCE

SAN JOSE STATE UNIVERSITY

MAY 2015

Dr. Melody Moh (Department of Computer Science)

Dr. Teng Moh (Department of Computer Science)

Mr. Vinay Pai.B.H (Software Engineer @ Cisco)

ABSTRACT

In recent year's document management tasks (known as information retrieval) increased a lot due to availability of digital documents everywhere. The need of automatic methods for extracting document information became a prominent method for organizing information and knowledge discovery. Text Classification is one such solution, where in the natural language text is assigned to one or more predefined categories based on the content. In my research classification of text is mainly focused on sentiment label classification. The idea proposed for sentiment analysis is multi-class classification of online movie reviews. Many research papers discussed the classification of sentiment either positive or negative, but in this approach the user reviews are classified based on their sentiment to multi classes like positive, negative, neutral, very positive and very negative. This classification task would help the business to classify the user reviews same as star ratings, which are manually given by users. This paper also proposes a better classification approach with multi-tier prediction model. The goal of this research is to provide a better understanding classification for sentiment analysis by applying different preprocessing techniques and selecting suitable features like bag of words, stemming and removing stop words, POS Tagging etc. These features are adjusted to fit with some of the machine learning text classification algorithms such as Naïve Bayes, SVM, sand SGD on frameworks like WEKA, SVMLight & Scikit Learn.

ACKNOWLEDGEMENTS

I am very thankful to my advisor Dr. Melody Moh for her guidance and walked me through the research with encouragement. I would like to thank my committee member Dr. Teng Moh for his help throughout this project. His suggestions are the keys to the successful completion of this project. I would also like to thank Mr. Vinay Pai for his suggestions and comments on this research. Finally, I would like to convey my sincere thanks for the learning opportunities provided by my committee.

TABLE OF CONTENTS

List of Acronyms

SVM	Support Vector Machine
SGD	Stochastic Gradient Descent
POS	Parts Of Speech
PCA	Principal Component Analysis
NLP	Natural Language Processing
NLTK	Natural Language Toolkit
WEKA	Waikato Environment for Knowledge Analysis

CHAPTER 1

INTRODUCTION

1.1 Introduction

Large datasets found in text data mining projects are mined to transform the text data into numerical values and link with knowledge database. The knowledge information is analyzed with traditional data mining algorithms and provide summary of words, determine the similarities and relate them to various interests in mining project. Text mining is very common and help to represent majority of hidden patterns and information available to particular research.

1.2 Text Classification

Text classification is playing crucial role in machine learning, where classification of large databases is in high demand. The classification of documents that are subjective depend user's opinion and style of expressing it. Many sources like rotten tomatoes, amazon, eBay and imdb are depending upon user reviews. Supervised learning takes a classified input data and labels from various sources, train the data using a model built using machine learning tools and predict the labels for new incoming text data. Classification for labels can have known values as 'true', 'false', numbered as binary representation ('0','1').

1.3 Sentiment Analysis

Sentiment analysis is the field of study that analyzes sentiments and emotions for various products, movies and more. These analysis is to find the mood of people/users and uses natural language processing and data mining techniques to classify the sentiment. There are many levels of sentiment analysis like document, sentence and aspect level. When we go deep into the sentiment we can look at the subjectivity

and emotions of each phrase. Every review, news or text can present both the facts, personal feelings and their views. Knowing the sentiment from the text is not finding facts, but finding the opinions and feelings.

1.4 Problem Description

The need of automatic methods for extracting document information became a prominent method for organizing information and knowledge discovery. For example, if a user need to watch any movie, the reviews given by previous users is very important to judge the rating. Reading the entire review and understanding the importance takes lot of time. A model is needed to read the text and classify them according to the sentiment [1]. Many research papers discussed the classification of sentiment either positive or negative, but in this approach the user reviews are classified based on their sentiment to multi classes [2] like positive, negative, neutral, very positive and very negative. The dataset is extracted from rotten tomatoes database.

Users provide star ratings and reviews to business and classification of movie reviews is based on user sentiment as positive or negative. But these star ratings to the corresponding reviews may not be accurate for business to classify the sentiment. A better understanding multi-class classification system is proposed to help business classify the sentiment using various pre-processing techniques, feature selection methods and supervised machine learning algorithms. This idea would help the business to classify the user reviews same as star ratings, which are manually given by users.

In Sentiment classification tasks, Words are considered as features by applying some feature extraction tasks and tokenizing them. Some features which are not important and omitted with feature selection

tasks like stop words, stemmers, pos tagging, lowercase conversion, n-grams, grid search etc. These tasks can improve the significance of important features and reduce noisy data.

This work is explained in next chapters: chapter 2 gives a review of related work; chapter 3 describes the classifiers used for multiclass classification; chapter 4 shows the pictorial diagram of classification system and proposed prediction model and implementation; chapter 5 lists the evaluation and results of the experiments done for text classification using proposed model. The final chapter describes conclusion and future work.

CHAPTER 2

BACKGROUND

On my research in machine learning and prediction algorithms, the focus is pointed to the text classification and sentiment analysis which helps individual/business to analyze and track the sentiment of the users.

The research work in [2] to implement a complete system for sentiment data mining on large datasets using Naïve Bayes classifier on top of Hadoop framework. They aimed to evaluate the scalability of Naïve Bayes classifier, so they implemented NBC instead of mahout library on Hadoop. This results are surprising and accuracy improved till 82% with large dataset. In future, this work can be used for large datasets like robotic applications and imagery and text data.

In this paper [3], research was done on spam categorization using various machine learning algorithm models and evaluated the results with famous data mining tools. As stated in the paper, the best tool is Weka for spam categorization and best classifier model using Weka is random forest. The also aimed to evaluate performance for the models in future work.

This research provided a new concept with word clouds on text analytics [4], which are used as visualization of text. The data used here is extracted from famous news articles. The prototype word cloud explorer is developed as an application where the text is transformed into word clouds based on the information and interactive features provided in the text. As discussed in paper, the application is also tested with individuals for better understanding.

This work of text classification is performed on support vector machines. They provide a classification system to categorize the sport

relevant text using SVM Light tool. In their work [5], the training data is pre-processed, a model is generated using the SVM learn and classify the data using svm classify. As stated in the paper, the system can be used to perform multi-classification and also svm ranking on the result of the classification.

Many students are spending time on searching research papers for relevant articles and domains. This search requires so much of effort in searching and extracting papers from big databases. The research [6] provided a better search tool to extract all the research articles and get best results. They used k-means clustering and trained algorithm using searching patterns of input text. The papers are added to database based on the categories and input text search patterns are compared to knowledge database and resultant papers are extracted. As a future work, they considered to extent the search engine into portable devices and implement a portable application for various devices.

A research work is done by Microsoft research professionals on click predictions [7] to explore the user psychology on sponsored search like advertisements. As described in the paper, the experiments are done on real world click predictions by users on advertisement searches. Historical clicks, desire pattern, desire level features and relevance features are used to compare the data. The experiment showed effective results for various combinations of ad categories.

Mining Social media data has improved a lot to predict the sentiment of the individuals. In this work [8], the ‘Facebook’ status updates are analyzed to find the sentiment of the people in a specific period. For this research they choose random Facebook profile status for a period of 3 months from Tunisia. Their main idea is to explore the status updates in the special time chosen and further track the

changes and topics they discussed using machine learning tools and compare their accuracy.

Another social media analysis research work to track opinions on drugs and cosmetic products decision making and predict adverse effects [9]. This work implemented a product safety framework for binding and handling the views and experiences of users of popular brands of drug and cosmetic products reported on social media platforms using text mining and sentiment analysis. For this research, they also developed a custom lexicon and training data for modeling the Naïve Bayes classifier for sentiment prediction. The work is also extended in future to use clustering tweets and user sentiments by location and more models.

In my research for advanced classification techniques, I followed hierarchical classification approach. In a research [10] on multi class classification, they proposed an approach to find facial images using SVM and PCA algorithms. The paper [11] is discussing the improvements of SVM classification using hierarchical approach and also improve the computation capability and accuracy. The authors from paper [12] solved multiclass classification problems using fuzzy hierarchical structure and SVM classifier to achieve more precise accuracy and classification.

Now-a-days making a decision in stock market is not really easy and predicting the market changes is a difficult task. A lot of analysis is required to perform these functionality. In this research [8], they proposed a framework which takes data from various sources like company web source, stock exchange data stores and unauthentic data using natural language processing tools from historical data. This data is categorized, analyzed and processed using NLP tools to make a

decision. In future, this work can be extended to perform technical analysis using this framework.

All the above research on text mining and sentiment analysis motivated me to perform more research on the domain. In the next section, classifiers used for sentiment analysis are discussed.

CHAPTER 3

CLASSIFIERS FOR SENTIMENT ANALYSIS

For multi-class classification, I have worked on the following text classification models which are trending in current machine learning community.

3.1 Naïve Bayes Classifier

Naïve Bayes classifier is a simple probabilistic classifier based on applying Bayes theorem. This strongly assumes independence between the features [19]. It uses the probability (i.e. How much percent of reviews are labeled as negative in the train set?) and the likelihood (i.e. what can be the probability of getting a word 'boring' in negative reviews?) to determine the following probability (i.e. if a tweet contains words 'sad', 'bad' and 'noise', what is the probability of classifying this tweet as negative?) in order to classify the inputs.

Abstractly, the probability model for a classifier is a conditional model

$$P(cj|d) = P(cj|x_1, x_2, x_3, x_4, \ldots x_n)P(cj) \qquad (1)$$

Where x1,x2,x3,x4,…xn – Features/words of document, cj is set of classes used in classification, p(cj|d) is conditional probability and P(cj) is prior probability of class c in equation (1). One of the problem is, if the number of features is large or when a feature can take on a large number of values, the calculation of probability is difficult and we need to manipulate the model by changing the parameters and filtering the features to make it more efficient.

In this paper, we are trying to evaluate the probability of movie reviews which is positive, negative or neutral, given its labels. This can

be restated as the probability of the reviews occurring if they are predetermined as positive, negative or neutral. The point of view that renders this process a “naive” Bayesian one is that we make a large assumption about how we can calculate the probability of the document occurring; it is equal to the product of the probabilities of each word within its occurrence. This means there is no match between each word and it is called Independence assumption. The probability of word for either positive or negative sentiment can be estimated by learning through a set of examples containing both sentiments and counting how often it occurs in each category. This method of training preclassified data examples is called supervised learning.

3.2 Support Vector Machine Classifier

Support Vector Machine (SVM) classifier is used for classifying linear as well as nonlinear data using a margin. This is a margin based classifier which selects the maximum margin. The idea of SVM was found by Vladimir Vapnik and his coworkers. This classifier technique is to separate the classes with surface that maximize the margin [20].The data points which are near to the margin are called as “support vectors”. The decision boundary is called as”hyperplane”.

In SVM model the data is trained using the known labelled classes and a model is built. Support vectors are important elements of training data generated during svm learning module. These model and vectors are used to classify the test data.

SVM based classification for linear data is mathematically represented as optimization problem [3]. The idea of non-linear SVM is implemented using kernels [3]. For the classification of non-linear data, SVM uses non-linear mapping to transform original training data

to higher dimension. In higher dimension, it searches for the linear separating hyperplane.

3.3 Stochastic Gradient Descent Classifier

Stochastic gradient descent (SGD) is a gradient descent optimization method for minimizing an objective function that is written as a combination of differentiable functions [18]. This efficient approach to fit linear models is used to solve error function known as “cost function” that measures the given line showing how well the data fits on the line.

SGD is one of the important classifier in machine learning community. These algorithms gained importance for large scale data. SGD has been applied for sparse machine learning problems evaluated in text classification and natural language processing. For a given sparse data, the classifier can be applied for very large training examples.

SGD Classifier implements a simple stochastic gradient descent learning routine which supports different loss functions and penalties for classification. SGD Classifier supports multi-class classification by combining multiple binary classifiers in a “one vs all” scheme. For every n classes, a binary classifier is used for learning between nth class and n-1 classes. In testing phase, we calculate the prediction score for each classifier and pick the class with highest score.

Driven by these classifiers, especially after looking several text mining, sentiment analysis approaches and based on architectural framework design of various multi class predictive models, I proposed a novel framework design of sentiment classification predictive model for multi class classification. A detailed description of the proposed framework is provided in next chapter.

CHAPTER 4

DESIGN AND IMPLEMENTATION

This chapter describes the architectural design of multi class classification system which has been motivated by the text classification research work and classifiers outlined in previous chapters. Initially, the design framework of the classification system is pictured and all the phases are explained in detail.

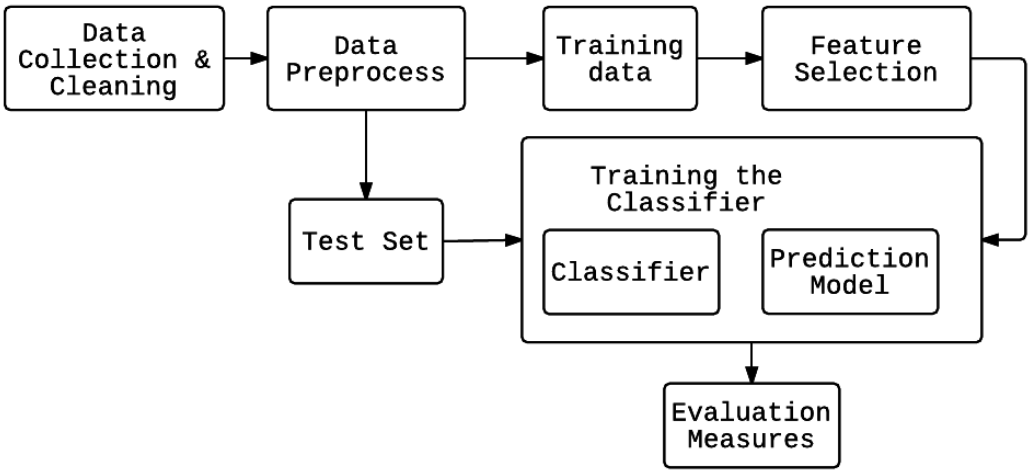


Fig 1: Multi-class Classification System Architecture

The classification system consists of seven stages: Data Collection & Cleaning, Data pre-processing, Training data, Feature Selection, Training the classifier (prediction model, classifier), Test set, Evaluation measures. Each stage helps in performing tasks related to data preparation and sentiment analysis. Following is brief description of entire process:

4.1 Data Collection & Cleaning

In Text classification, the user provides a set of data from any valid resource and use a part of it for training the classifier model and then predict the result of the remaining data. The data instances are branched into multiple classes. The method of training instances from a set of correctly classified instances is called “Supervised Learning”. In detail, for training the model each instance is provided with labeled class which are used for predicting the correct class for new set of unclassified instances. In this project we considered the correctly classified movie reviews data from rotten tomatoes database.

Very Negative - 0
Negative - 1
Neutral - 2
Positive - 3
Very Positive - 4

These movie reviews consists of predefined classes based on the sentiment level and are scored from 0-4.

Below is the sample dataset with sentiment labels.

PhraseId	SentenceId	Phrase	Sentiment
1	1	A series of escapades demonstrating the adage that what is good for the goose is also good for the gander some of which occasionally amuses but none of which amounts to much of a story .	1
2	1	A series of escapades demonstrating the adage that what is good for the goose	2
3	1	A series	2
4	1	A	2
5	1	series	2
6	1	of escapades demonstrating the adage that what is good for the goose	2
7	1	of	2
8	1	escapades demonstrating the adage that what is good for the goose	2
9	1	escapades	2
10	1	demonstrating the adage that what is good for the goose	2
11	1	demonstrating the adage	2
12	1	demonstrating	2
13	1	the adage	2
14	1	the	2
15	1	adage	2
16	1	that what is good for the goose	2
17	1	that	2
18	1	what is good for the goose	2
19	1	what	2
20	1	is good for the goose	2
21	1	is	2
22	1	good for the goose	3
23	1	good	3
24	1	for the goose	2
25	1	for	2
26	1	the goose	2
27	1	goose	2
28	1	is also good for the gander some of which occasionally amuses but none of which amounts t	2
29	1	is also good for the gander some of which occasionally amuses but none of which amounts t	2
30	1	is also	2

Fig 2: Input Dataset with Sentiment Labels

Generally the movie reviews are just sentences or paragraphs. But this set of labeled data consists of phrases for each review and each phrase is provided with class label. This makes the data processing a bit difficult with repetition of words and unwanted text. The data from public websites can contain special characters and null spaces. The data must be cleaned to avoid all those junk data to be processed in further stages.

4.2 Data Pre-processing

From the data prepared in the previous stage we have to organize and partition the datasets which can be used for training the model and testing the data. In machine learning while using the supervised learning, it is necessary to use a training set with labeled data to train the model. This training dataset should signify the entire dataset to increase the accuracy of the prediction model. In most of the machine learning preprocessing methods the train set and test set are separated using 80-20 rule, where 80% of data is used for training and 20% is used as test set.

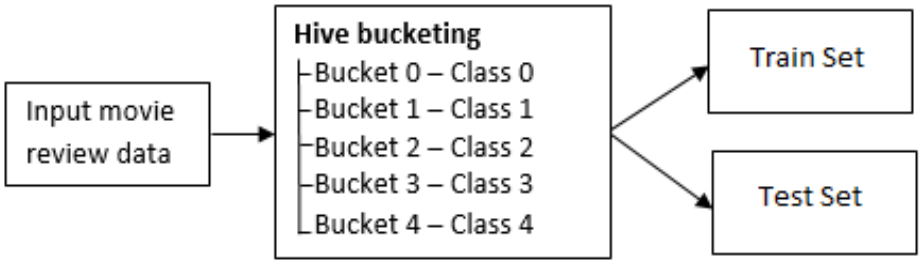


Fig 3: Hive Bucketing

The reviews data consists of unique id for each phrase and unique id for each review sentence. Each instance in the data has both unique phrase identifier, review identifier, review and sentiment label. The

whole data is classified with 5 labels which represent the sentiment. Initially we have split data based on the unique id, but the ratio of classified labels is uneven, over fitting and can't be sufficient for training the model which may result in unreliable accuracy.

To mitigate this, I have considered using Hive bucket split method to extract random samples from each class. The data is loaded to Hadoop database and hive queries are used to split the data in equal proportions.

First, a table is created with all the data fields and cluster them using the sentiment. Below is the query to load data to bucketed table.

```
  CREATE TABLE review_split
  (
  PhraseId INT, SentenceId INT, Phrase
STRING, Sentiment INT
  )
  CLUSTERED BY (Sentiment) into 5 buckets
  ROW FORMAT DELIMITED FIELDS
  TERMINATED BY '\t‘
   STORED AS TEXTFILE;
```

Each bucket is selected using the tablesample clause and a built-in function is used to select random sample of train and test set from each bucket.

Here is the query to split the trainset and test set as 80-20 ratio.

```
  from (
  select *, (rand() * 100 <= 20) as is_test_set
  from review_split
  tablesample (bucket 1 out of 5 on Sentiment)
  ) t
  insert overwrite directory '/user/test/test_set'
  select * where is_test_set = true
  insert overwrite directory
'/user/train/train_set'
  select * where is_test_set = false;
```

This query generates test and train sample for bucket 1. Similarly queries are executed for all the remaining buckets. The generated data is concatenated and a final trainset and test set are created.

4.3 Feature Selection

Feature selection is one of the data processing step which plays key role in reducing the feature space and improving the classifier accuracy. Feature selection is nothing but a process of choosing relevant attributes which help in model construction. These methods are needed to eliminate unwanted data, trim the long texts which provide noise to the data and irrelevant data that doesn't help in predicting accuracy of the model. In this project, the movie review data has gone through the cleaning and pre-processing stages where the structure of data is validated, initial dataset for training and testing the model is generated. All the features applied must be in an order and they should be applied one after another to carefully understand and experiment on them. This section mainly focus on selecting features and contribution of various methods for increasing prediction performance.

4.3.1 Tokenization

Tokenization is a method to break the sentences into meaningful words and phrases. This process is achieved by using delimiters like comma, stop and space. Inbuilt nltk libraries have tokenize function capability to divide into words, convert all the text data to lowercase.

4.3.2 Stop word removal

When working with text classification methods, removal of stop words is a common approach to reduce noise in the data. These stop words can be in any language, not only English. If we eliminate the

stop words, we can reduce the number of features in the corpus and concentrate more on the actual features instead.

Stop words are generally single set of words. For example, determiners like ‘a, an, the’, prepositions like ‘above, under, near’. These are many repeated words which can be eliminated from the corpus. In this project we are mainly considering the domain as movies and public reviews. Some of the domain specific stop words need to be handled. For example, movie, popcorn, noon show etc.

Stopwords are removed using inbuilt stop word list from nltk library.

4.3.3 Stemmers

Stemming performs basic functionality as trimming the text to original form of word. For example, if we have a word ‘celebrating’, stemming will cut off the word to ‘celebrate’ and use same feature for both the occurrences of the word. This can directly added to positive sentiment. In some cases like “wait today” will mean as doctor checkup, while “long waiting” may be regarding queue. In these phrases, stemming will confuse the classifier.

To overcome this issue we need to use additional features like N-grams which will help in distinguishing same words which have same meaning.

4.3.4 Using N grams features

N-grams is the selection feature probabilistic model where n items are considered as each word from given instance. There are multiple N-grams like unigram, bigram, trigrams etc. simplest N-gram is unigram, where only one word is considered for each feature processing. Similarly a Bigram has 2 words for each feature. Many

experiments in previous research have proven that N-grams will improve the quality of feature sets.

4.3.5 Parts of Speech Tagging (POS)

Part-of-speech tagging is one of the most important text analysis tasks used to classify words into their part-of-speech and label them according to tag category. Tag category consists set of tags that are used for tagging parts of speech. Conjunctions, prepositions and pronouns doesn't help in deciding the sentiment of the review. The idea is to separate sentiment carrying words from content words, without completely ignoring entire word categories.

1. General syntax:

```
sent_tags=nltk.word_tokenize("file_name");
nltk.pos_tag(sent_tags)
```

2. Original text:

Tartakovsky 's team has some freakish powers of visual charm , but the five writers slip into the modern rut of narrative banality

3. POS_tagging:

```
Tartakovsky 's| NN team|NN has|TN some|DT freakish|JJ powers|NN of|IN visual|JJ charm|NN , but|DT the|DT five|NN writers|NN slip|VBG into|IN the|IN modern|RB rut|NN of|IN narrative|RB banality|JJ
```

4. Steps Involved:

Each input review text is tokenized using word Tokenizer

POS tags are added to each word using the package (maxent_treebank_pos_tagger) from nltk.

Filtered the words based on rule based parts of speech tagging.

All the parts of speech other than Nouns, adjectives, adverbs, verbs are not considered.

4.4 Prediction Model

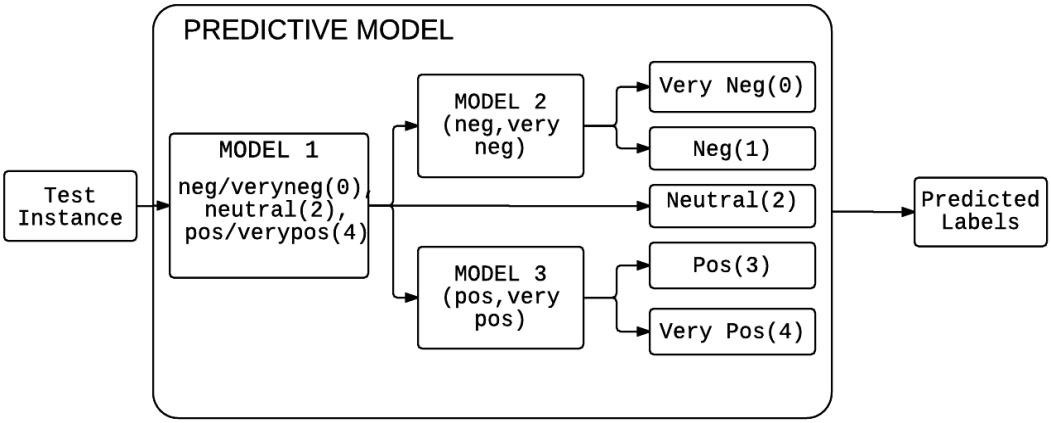


Fig 4: Multi-tier Prediction Model

This part of the architecture has an important role in sentiment classification. The prediction model is used to consider the labeled data and train the classifier. With the trained model the test set is used to predict each review instance and set label to the unclassified test data. For the base model, all the labeled data is considered as a single tier and whole data is trained. In the above proposed predictive model framework, multi-class classification is performed using multi-tier model architecture.

It consists of three models, each model inherits the same configuration as the base model.

Model-1: Used to train the classifier using whole reviews data but the data is labelled as 3 classes.

Negative {1} and Very Negative {0} sentiment data as class ‘0’

Positive {3} and Very Positive {4} sentiment data as class ‘4’

During the prediction, when the test instances are invoked each instance gets classified into any of the three classes {0, 2, 4}. Our aim is to predict all the 5 sentiments, so the instances which are classified as negative are given to model-2 to further classify the review. Similarly, the instances which are classified as positive are provided to model-3. The neutrally classified review is directly appended to the final predicted labelled output.

Model-2: Used to train the classifier using trainset consisting of Negative and Very negative labels. The test instance provided to this model are only negative reviews classified by model-1. So, the instance can be classified either {0} or {1}.

Model-3: This model trains the classifier using dataset consisting of positive and very positive labels. When the test instance is provided to predict the label, this model can classify the incoming instance as positive {3} or very positive {4}.

All the intermediate models perform a distributed task compared to base model. This approach helps in improving the classifier accuracy as the classification task complexity reduces. The probability of finding the labels increases as each model is trained with less dataset.

CHAPTER 5

EXPERIMENTS AND RESULTS

5.1 The Experiment Data

The dataset is movie reviews obtained from data processing stage where whole data is split into training and test set. From the split we had 120k training data and 30k test data. Entire data is cleaned and verified in preprocessing stage.

5.2 Naive Bayes Classifier using Weka

To demonstrate the working of proposed framework, a machine learning open source tool “WEKA” is used to perform the text classification task [15]. This tool performs several machine learning tasks and have in built algorithms to train and predict the accuracy of the test data.

5.2.1 Execution steps:

WEKA tool uses a unique data format to process and analyze the data. It uses .arff format for the text files where the header and data attributes are mentioned separately with prefix (‘@’) and the data is comma delimited. The input file is formatted and changed to .arff extension.

The attributes 'PhraseIdent' and 'ReviewIdent' doesn't play any role during the text classification. So we have removed them from the dataset for easy processing.

Here is the distribution of labels as shown in the Weka Explorer:

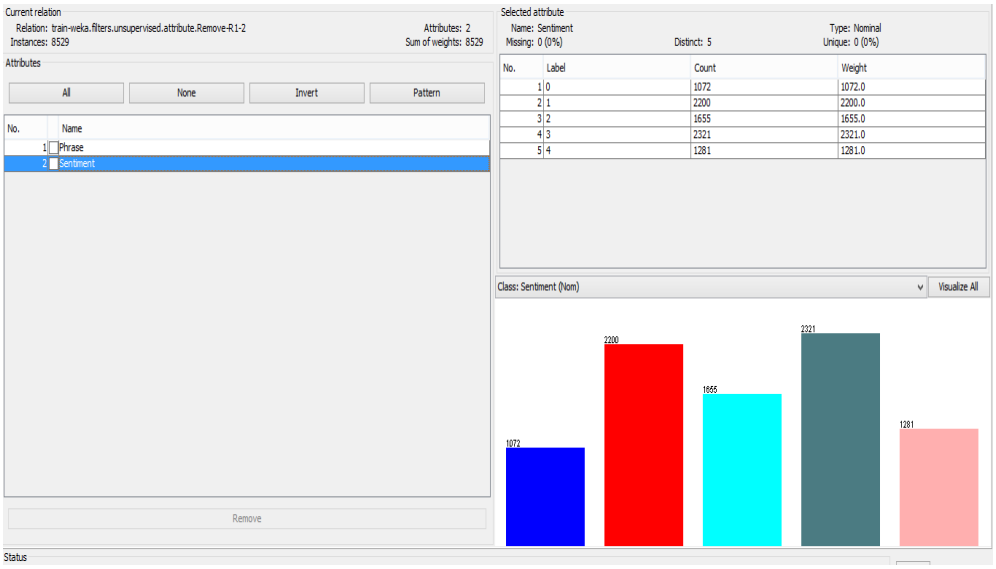


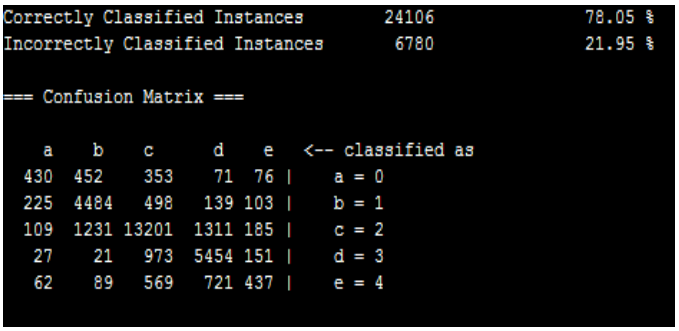
Fig 5: Data Distribution in WEKA Explorer

Over the entire data, feature selection tasks are applied to reduce the noise and increase the performance of text classification. In this tool we have the option of tokenizing the data, convert all the data to lower case and checked the usage of stemmers. An inbuilt stopwords list is used and we also added an external stop word list which is created after data mining the initial raw data.

Weka has many machine learning algorithms to perform text classification tasks. Out of them Naïve Bayes multinomial text classifier is used to test this movie reviews data. This tool provides Naive Bayes Multinomial Text algorithm which works directly on the string attribute. So, this algorithm get rids of 'StringToWord' vector conversion.

5.2.2 Evaluation & Results:

Confusion Matrix:



```
Correctly Classified Instances        24106               78.05 %
Incorrectly Classified Instances       6780               21.95 %

=== Confusion Matrix ===

    a    b    c     d    e   <-- classified as
  430  452  353    71   76 |    a = 0
  225 4484  498   139  103 |    b = 1
  109 1231 13201 1311  185 |    c = 2
   27   21  973  5454  151 |    d = 3
   62   89  569   721  437 |    e = 4
```

Fig 6: Results of Naïve Bayes with WEKA

The Naïve Bayes Multinomial Text model is built and various feature selection tasks are performed to improve the accuracy as shown below,

Model	With stopwords	Null Stemmer	N-grams	Accuracy
NB1	NO	NO	YES	73.29%
NB2	NO	YES	NO	74.63%
NB3	YES	NO	YES	75.02%
NB4	**YES**	**YES**	**YES**	**75.58%**

Table 1: Various Experimental results of Naïve Bayes Classifier

In above table, features like stopwords, stemmers and inbuilt stemmer in weka are applied for different experiments and the model gained the accuracy.

5.3 SVM classifier using svmlight

To demonstrate the usefulness of the proposed prediction model, we used SVMLight multiclass classification [5] for evaluating the SVM model [20] with movie reviews data.

5.3.1 SVMLight Multiclass:

SVMLight multiclass consists of learning module (svm_multiclass_learn) and a classification module (svm_multiclass_classify). The input data is applied with learning module to train the data and classification module is applied on learned model.

5.3.2 Input format:

SVMLight input data contains training data. The first lines may contain comments and are ignored if they start with '#'. Each of the following lines represents one training example and is of the following format:

```
<line>.=.<target><feature>:<value><feature>:<value>...
<feature>:<value>
```

A space character separates the target value and each of the feature/value pairs. Feature/value pairs must be ordered by increasing feature number. Features having zero can be skipped.

5.3.3 Execution Steps:

SVMLight input data should be represented in vectors. Original train data is converted into vector data format. This vector data is transformed into SVMLight input format using string manipulations scripts.

Here are the steps followed to transform vectored data to SVMLight input format:

1. Change the vector format for SVMLight input

2. SVMLight multiclass classifier accepts classes starting from 1, so we changed the class range from [0...4] to [1…5].

3. All the <feature>:<value> sets need to be arranged in ascending order inorder to recognize the featured vectors.

A sample of SVMLight input format is shown below:

```
1 2222:9.820667266845703 10017:6.06250524520874 12041:6.2578935623168945 14890:6.844694137573242 15166:9.778107643127441
1 1266:6.474584579467773 1276:8.246631622314453 4529:7.887257099151611 4901:9.660325050354004 5362:8.193987846374512
5383:8.219963073730469 5501:4.826986312866211 6178:8.264813423156738 8450:8.021687507629395 8806:4.2370076179504395
9428:6.087147235870361 11457:3.290613889694214 12387:5.0614705085754395
1 1266:6.474584579467773 1276:8.246631622314453 4901:9.660325050354004 5362:8.193987846374512 5383:8.219963073730469
5501:4.826986312866211 6178:8.264813423156738 8450:8.021687507629395 9428:6.087147235870361 11457:3.290613889694214
12387:5.0614705085754395
1 1266:6.474584579467773 1276:8.246631622314453 4901:9.660325050354004 5362:8.193987846374512 5383:8.219963073730469
5501:4.826986312866211 6178:8.264813423156738 8450:8.021687507629395 9428:6.087147235870361 11457:3.290613889694214
12387:5.0614705085754395
1 1266:6.474584579467773 1276:8.246631622314453 4901:9.660325050354004 5362:8.193987846374512 5383:8.219963073730469
5501:4.826986312866211 6178:8.264813423156738 8450:8.021687507629395 9428:6.087147235870361 11457:3.290613889694214
12387:5.0614705085754395
1 5362:8.193987846374512 5383:8.219963073730469
1 1266:6.474584579467773 1276:8.246631622314453 4901:9.660325050354004 5501:4.826986312866211 6178:8.264813423156738
8450:8.021687507629395 9428:6.087147235870361 11457:3.290613889694214
1 1266:6.474584579467773 1276:8.246631622314453 4901:9.660325050354004 5501:4.826986312866211 6178:8.264813423156738
8450:8.021687507629395 9428:6.087147235870361 11457:3.290613889694214
1 658:8.283332824707031 3138:9.660325050354004 3711:9.588865280151367 3925:6.675765514373779 4517:6.89237642288208
5766:9.372642517089844 6594:9.0849609375 7253:4.172612190246582 7739:8.513510704040527 8764:5.275062561035156 9246:5.981682777404785
9321:11.01025104522705 9962:8.638673782348633 11854:7.061758518218994 12588:9.004918098449707 15026:9.14949893951416
1 658:8.283332824707031 3138:9.660325050354004 3711:9.588865280151367 3925:6.675765514373779 6594:9.0849609375 7739:8.513510704040527
9246:5.981682777404785 9962:8.638673782348633 11854:7.061758518218994 12588:9.004918098449707 15026:9.14949893951416
1 3711:9.588865280151367 7739:8.513510704040527 9246:5.981682777404785 12588:9.004918098449707 15026:9.14949893951416
1 3711:9.588865280151367 7739:8.513510704040527 9246:5.981682777404785 12588:9.004918098449707 15026:9.14949893951416
1 4517:6.89237642288208 5766:9.372642517089844 7253:4.172612190246582 8764:5.275062561035156 9321:11.01025104522705
1 1049:7.952215194702148 13953:7.357739448547363
```

Fig 7: SVMLight sparse vector format

5.3.4 Evaluation & Results

Use the train data as input to learning module and outputs the learned rule as a model.

```
$./svm_multiclass_learn -c 200 reviews/train.dat
reviews/model
```

The model is written to a file and used for predictions.

Classify the learnt model using the test data and generate predictions.

```
$./svm_multiclass_classify reviews/test.dat reviews/model
reviews/predictions
```

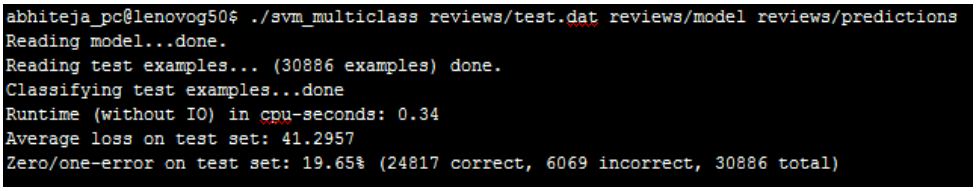


Fig 8: Execution result of SVM classifier with SVMLight

Our objective is to minimize the error rate from the predictions. We tuned our model using parameters and selected features of data.

Regular parameter C is a trade-off between training error and the flatness of the solution. We tested the model with variation of C values. A Grid Search algorithm is used to find the optimized C parameter for training data. A range of values has been provided to run the learning module and generate a model for each iteration and the resultant predictions also vary as C increases. Even though the value we provided is not the optimized solution, we can find a best suitable classifier approach using grid search.

Parameter - C	No of Iterations	Error rate on test set	Accuracy (%)	Execution Time (sec)
0.01	3	31.63	68.37	71
1	5	31.24	68.76	72
50	15	30.99	69.01	75
200	24	29.66	70.34	84
400	38	29.3	70.7	97
800	54	28.89	71.11	110
1000	**67**	**28.47**	**71.53**	**127**

Table 2.1: Experiment results for SVM Parameters

As we increase C, the final training error is less. For large values of C, the hyperplane margin is small and the features get classified correctly in the training data. On the other hand, a small value of parameter c will change the margins and the features can be misclassified over the hyperplane. If there is a tiny parameter value,

then we should get most of the training data misclassified even though the algorithm is satisfies linear separable data.

ε is used to level the accuracy of the approximated function. This parameter value can affect the number of support vectors used to construct the regression function. With the increase in ε, less support vectors are supported.

Parameter - ε	Parameter - C	No of Support Vectors	Error rate on test set	Accuracy (%)
10	200	3	28.8	71.2
5	200	4	28.46	71.54
1	200	5	26.39	73.61
0.1	200	11	26.22	73.78
0.01	**200**	**23**	**26.1**	**73.9**

Table 2.2: Experiment results for SVM Parameters

If the value is larger than the range of the target values we cannot expect a good result. If epsilon is zero, we can expect over fitting.

To improve the accuracy, stop words are omitted from the original data set. The input sparse vector data file is modified to omit stop words.

With stop words removal there is a change in input vector data. This helps in accuracy increase of 2% compared to the SVM classifier original data.

Feature	Accuracy (%)
SVM Original Model	70.25
Including Grid Search	73.90
With Stopwords	74.27

Table 2.3: Experiment results for SVM Parameters

5.4 SGD classifier using Scikit Learn

Scikit Learn is an important python module which is integrated with various supervised and unsupervised learning models [18]. This is a library built upon SciPy (scientific Python). This is a BSD licensed and have many distributions in Linux. This python module is used for many educational institutions and commercial purposes. Libraries included in this module are NumPy, SciPy, Matplotlib, IPython, Sympy, Pandas.

In this paper we demonstrate the importance of proposed prediction model using the SGD classifier with Scikit Learn. NLTK Libraries are also used to tokenize the data. Initial setup requires Python, a virtual environment to install Scikit Learn along with NumPy, SciPy and Nltk.

5.4.1 Execution Steps:

The train and test data are read and stored in separate variables. In classification, the data should be represented in features. Here reviews are in words and we use words as the features.

```
from sklearn.feature_extraction.text import CountVectorizer
```

Using CountVectorizer as a feature extraction method, the dataset is translated into feature vectors in the form of a sparse matrix. The parameters used for CountVectorizer are:

`Tokenizer, min_df, ngram_range : tuple (min_n, max_n), stop_words, lowercase.`

```
def senti_countvector(tokenizer, min_df, ngram, stopwords,lowercase):
    return vectorizer(CountVectorizer(tokenizer=lambda x: x.split(),
                                      min_df=1,
                                      ngram_range=(1, 2),
                                      stop_words=yes),
                                     lowercase=yes)
```

Scikit Learn has various linear classifier models integrated in the library. Out of which SGD classifier is used in this experiment.

```
from sklearn.linear_model import SGDClassifier
from sklearn.pipeline import Pipeline, FeatureUnion
from sklearn.grid_search import GridSearchCV
```

Once all the features are extracted, a configuration need to be set for the pipeline along with features selected.

```
def senti_pipeline(classifier=SGDClassifier, classifier_args=None, lowercase=True,
                   min_df=1, ngram=1, stopwords=True, duplicates=False):
```

Use gridsearchCV to fit and predict the model using below parameters

```
senti_fit = GridSearchCV(senti_pipeline, parameters, cv=2, verbose=True, n_jobs=-2)
```

With the initial setup of grid search, train the model using the train set,

```
senti_fit.fit(train_data, train_label)
```

Test the unclassified data using predict function. Below command predicts the labels.

```
predictions = senti_fit.predict(test_data)
```

The result file will be generated consists of test instances with unique phrase identifier and sentiment label.

5.4.2 Evaluation Results:

The predicted results from Scikit Learn are compared with the original dataset with labels to match and find the final accuracy. Load the initial data with labels and predicted labels into separate tables in Hive. Below are the queries used to create and load data.

```
create external table test_out (PhraseIdent String, ReviewIdent String, Review String, Sentiment int)
                        row format delimited fields
                        terminated by '\t' location '/user/data/orig_test';


create external table predicted_out (PhraseIdent String, Sentiment int)
                            row format delimited fields
                            terminated by ',' location '/user/data/predicted';
```

Evaluate the results by comparing labels in both datasets.

```
select count(*) from (select A.PhraseIdent from test_out A join predicted_out B
        on (A.PhraseIdent = B.PhraseIdent)
        where A.Sentiment = B.Sentiment)t;
```

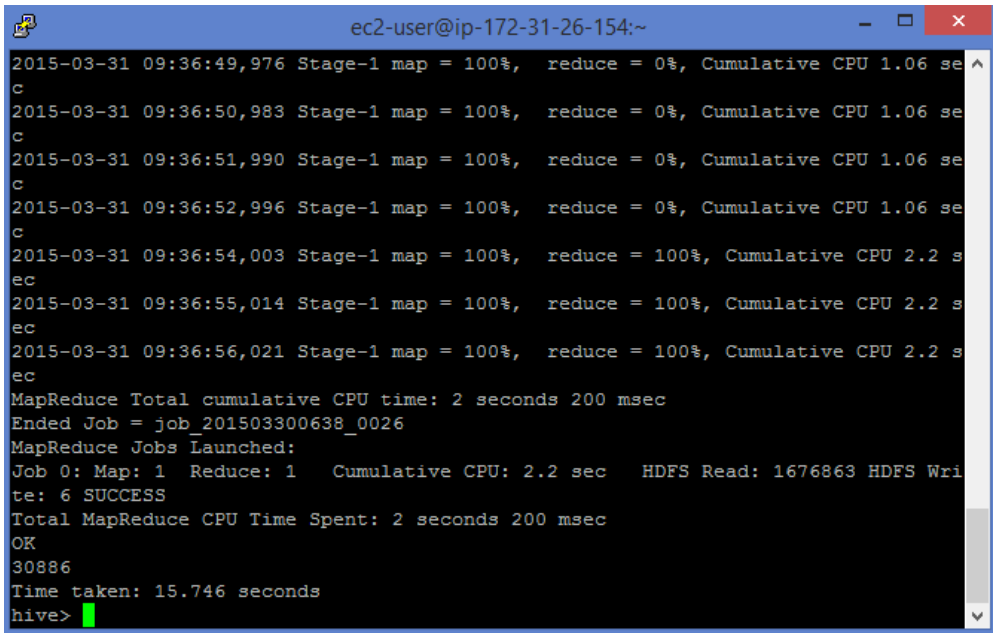


Fig 9.1: Execution result of SGD Classifier

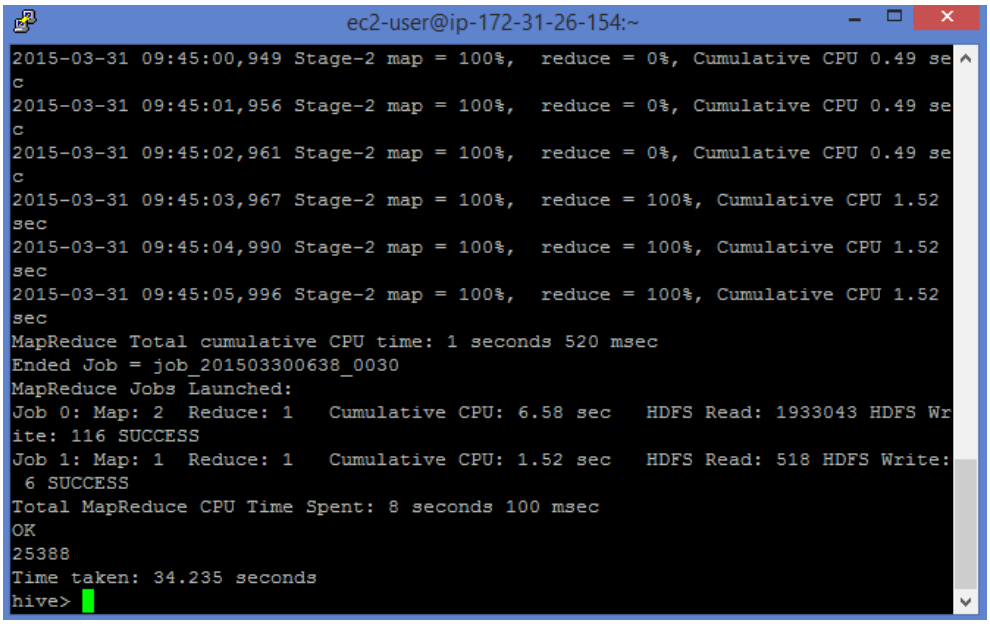


Fig 9.1: Execution result of SGD Classifier

Accuracy: predicted_instances/actual_instances

25388/30886 = 82.19%

Feature	Accuracy (%)
Feature 1 = Stopwords + Stemmers + N grams	79.42%
Feature 2 = Feature 1+ Grid Search	81.73%
Feature 2 + POS Tagging	82.19%

Table 3: Execution results of SGD Classifier using various Features

To further improve the results, some more experiments are performed on individual models (Model-2 & Model-3) by adding custom dictionaries. We collected most frequently occurred words from labels (negative, very negative) to accurately distinguish negative from very negative. Similarly, the custom dictionary is built for Model

3. Some of the words which differentiates the above labels are [lacks, bit, despite, little, but, instead, thin, light, barely, sometimes and so on]. From the results, we analyzed the wrongly classified instances and revised the dictionary.

Feature	Accuracy (%)
Without custom Dictionaries	82.19%
With custom Dictionaries	83.60%
With Revised Dictionaries from results	83.93%
Dictionaries build using user defined functions	**87.23%**

Table 9. Execution results of SGD classifier with custom dictionaries

From the above experiments in the research, SGD Classifier has provided good results with Scikit learn. Using the multi-tier prediction model, the accuracy has raised and showed better results along with features selected in the model.

In this project, the document is classified to 5 classes. From the above experiments, this multi-tier classification system can be used for classifying document into various classes and can also be used for other domains like social media, news and neural networks. The corpus used in the project has been improved by adding various feature selection methods to the classified data. More emotional lexicons and features can be added to improve better results and accuracy depending on the application.

CHAPTER 6

CONCLUSION AND FUTURE WORK

This research proposes a novel approach for multi-class classification problem. In my research, I proposed a solution for text classification to predict the sentiment of movie reviews on a scale of 5 levels using different machine learning algorithms. This paper also demonstrated a multi-tier prediction model for multi-class classification system. The classification of data is improved based on feature selection methods. These experiments and results have outperformed and gave a better classifier for supervised text classification. I believe that my research will help in future to increase accuracy levels in multi-class text classification. In Future, the research work is planned to test on document level datasets. This prediction model can be used by updating the corpus using various emotional lexicons and features. Another direction of expanding work is implementing the model for other large data sets containing social media/ news data.

REFERENCES

[1] V. Singh, Piryani, Uddin, and Waila, 'Sentiment analysis of Movie reviews and Blog posts', 2013 3rd IEEE International Advance Computing Conference (IACC), Jan. 2013

[2] C. Troussas, M. Virvou, K. J. Espinosa, K. Llaguno, and J. Caro, 'Sentiment analysis of Facebook statuses using Naive Bayes classifier for language learning', IISA 2013, 2013

[3] R. Mishra and R. S. Thakur, 'An Efficient Approach for Supervised Learning Algorithms Using Different Data Mining Tools for Spam Categorization',2014 Fourth International Conference on Communication Systems and Network Technologies, Jan. 2014.

[4] F. Heimerl, S. Lohmann, S. Lange, and T. Ertl, 'Word Cloud Explorer: Text Analytics Based on Word Clouds', 2014 47th Hawaii International Conference on System Sciences, Jan. 2014.

[5] S. Aurangabadkar and M. A. Potey, 'Support Vector Machine based classification system for classification of sport articles',2014 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT), Jan. 2014.

[6] E. Calvillo, Padilla, Munoz, Ponce, and J. Fernandez, 'Searching research papers using clustering and text mining',CONIELECOMP 2013, 23rd International Conference on Electronics, Communications and Computing, Jan. 2013.

[7] T. Wang, J. Bian, S. Liu, Y. Zhang, and T.-Y. Liu, 'Psychological advertising', Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '13, Jan. 2013.

[8] J. Akaichi, 'Social Networks' Facebook' Statutes Updates Mining for Sentiment Classification', 2013 International Conference on Social Computing, Jan. 2013.

[9] H. Isah, P. Trundle, and D. Neagu, 'Social media analysis for product safety using text mining and sentiment analysis',2014 14th UK Workshop on Computational Intelligence (UKCI), Jan. 2014.

[10] D. Drume and A. S. Jalal, 'A multi-level classification approach for facial emotion recognition', 2012 IEEE International Conference on Computational Intelligence and Computing Research, 2012.

[11] B. Demir and S. Ertrk, 'Improving SVM classification accuracy using a hierarchical approach for hyperspectral images', 2009 16th IEEE International Conference on Image Processing (ICIP), 2009.

[12] T. Guernine and K. Zeroual, 'SVM Fuzzy Hierarchical Classification Method for Multi-class Problems', 2009 International Conference on Advanced Information Networking and Applications Workshops, 2009

[13] B. Liu, E. Blasch, Y. Chen, D. Shen, and G. Chen, 'Scalable sentiment classification for Big Data analysis using Naïve Bayes Classifier',2013 IEEE International Conference on Big Data, Jan. 2013.

[14] S. S. Abdullah, M. S. Rahaman, and M. S. Rahman, 'Analysis of stock market using text mining and natural language processing',2013 International Conference on Informatics, Electronics and Vision (ICIEV), Jan. 2013.

[15] L. Dan, L. Lihua, and Z. Zhaoxin, 'Research of Text Categorization on WEKA', 2013 Third International Conference on Intelligent System Design and Engineering Applications, 2013.

[16] H.-X. Shi and X.-J. Li, 'A sentiment analysis model for hotel reviews based on supervised learning', 2011 International Conference on Machine Learning and Cybernetics, Jan. 2011.

[17] P. C. Njolstad, L. Hoysaeter, W. Wei, and J. A. Gulla, 'Evaluating Feature Sets and Classifiers for Sentiment Analysis of Financial News', 2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT), Jan. 2014

[18] R. Garreta, G. Moncecchi, and G. R. M. Guillermo, Learning Scikit-Learn: Machine Learning in Python. United States: Packt Publishing, 2013.

[19] C. D. Manning, P. Raghavan, and H. Schutze, An Introduction to Information Retrieval, 1st ed. New York: Cambridge University Press, 2008.

[20] N. Christianini, J. Shawe-Taylor, and N. Cristianini, An Introduction to Support Vector Machines and Other Kernel-based Learning Methods, 1st ed. United Kingdom: Cambridge University Press, 2010.