MAULANA AZAD NATIONAL INSTITUTE OF TECHNOLOGY

(A Deemed University)

Bhopal-462003

DEPARTEMENT OF COMPUTER SCIENCE AND ENGINEERING

MINOR PROJECT REPORT

ON

"EVALUATION OF VARIOUS MACHINE LEARNING MODEL FOR SENTIMENTAL ANALYSIS"

SUBMITTED FOR EVAUATION OF MINI TEST

Presented by:

Ayush Mishra 161112238

Yashwant Shrivastava 161112246

Shreyash Gajbhiye 161112247

Guided by:

Prof. Nilay Khare

&

Dr. Mansi Gyanchandani

MAULANA AZAD NATIONAL INSTITUTE OF TECHNOLOGY

Bhopal-462003

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DECLARATION

We hereby declare that the work which is being presented in the project titled **"Evaluation of various model of Machine Learning in Sentimental Analysis"** in partial fulfillment of the requirements of the degree of technology in computer science and engineering is an authentic record of our own work carried out under the guidance of **Dr. Mansi Gyanchandani.** The work has been carried out in **Maulana Azad National Institute of Technology, Bhopal** and has not been submitted for any other degree of diploma. This is to certify that the above declaration is correct to the best of our knowledge.

Ayush Mishra

161112238

Yashwant Shrivastava

161112246

Shreyash Gajbhiye

161112247

ACKNOWLEDGMENT

We have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. We would like to extend my sincere thanks to all of them. We are highly indebted to Dr. Manish Pandey sir for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project. We would like to express my gratitude towards ours parents & each other for their kind cooperation and encouragement which help us in completion of this project.

Our thanks and appreciations also go to our colleague in developing the project and people who have willingly helped us out with their abilities.

NAME :	**SCHOLAR NO.** :	**SIGNATURE :**
Ayush Mishra	161112238	
Yashwant Shrivastava	161112246	
Shreyash Gajbhiye	161112247	

CONTENTS

Topics **Page no.**

ABSTRACT

Sentiment analysis is a type of natural language processing for tracking the mood of the public about a particular product or topic. Sentiment analysis, which is also called opinion mining, involves in building a system to collect and examine opinions about the product made in blog posts, comments, reviews or tweets. Sentiment analysis can be useful in several ways.

In fact, it has spread from computer science to management sciences and social sciences due to its importance to business and society as a whole. In recent years, industrial activities surrounding sentiment analysis have also thrived. Numerous startups have emerged. Many large corporations have built their own in-house capabilities. Sentiment analysis systems have found their applications in almost every business and social domain.

The goal of this report is to give an introduction to this fascinating problem and to present a framework which will perform sentiment analysis on Microblogging site like twitter and others reviews by evaluating various model like naïve bayes, Linear SVC, SGD classifier, MNB classifier, Bernoulli classifier and Logistic Regression Classifier and NuSVC Classifier.

INTRODUCTION

Microblogging websites have evolved to become a source of varied kind of information. This is due to nature of microblogs on which people post real time messages about their opinions on a variety of topics, discuss current issues, complain, and express positive sentiment for products they use in daily life. In fact, companies manufacturing such products have started to poll these microblogs to get a sense of general sentiment for their product. Many times these companies study user reactions and reply to users on microblogs. One challenge is to build technology to detect and summarize an overall sentiment. In this Report, we look at one such popular microblog called Twitter and build models for classifying "tweets" into positive and negative classes. We build models for two classification tasks: a binary task of classifying sentiment into positive and negative classes and a 2-way task of classifying sentiment into positive and negative classes.

In Sentiment Analysis, numbers of sentences or sentences of documents. All these documents or sentences may convey opinion or maybe not. Formally, there is document set D= {d1, d2, …, dN}, sentence set S= {S1, S2, …, Sn} and all these documents and sentences belong to some specific entity e where e is a product, service, topic, issue, person, organization, or event.

We followed four steps of classification.

1.) First step: We have extracted the top-N feature from Data set.

2.) Second Step: Creating the frequency distribution table after removing the stop word and punctuation.

3.) Third Step: Create Word Feature using 2000 most frequently occurring words.

4.) Fourth Step: Making a bag of word model for unigram, bigram, trigram and unigram + bigram.

In this particular area just challenges are proposed but still researchers are trying to find out efficient solution to get analyzed these kinds of implicit opinions in the objective sentences, so in order to find out best classifier we have use a number of models to evaluate which model predict the best results.

METHODOLOGY AND WORK DESCRIPTION

Twitter is a social networking and microblogging service that allows users to post real time messages, called tweets. Tweets are short messages, restricted to 140 characters in length. Due to the nature of this microblogging service (quick and short messages), people use acronyms, make spelling mistakes, use emoticons and other characters that express special meanings.
We acquire 11,875 manually annotated Twitter data (tweets) from a commercial source. They have made part of their data publicly available. For information on how to obtain the data, see Acknowledgments section at the end of the paper. They collected the data by archiving the real-time stream. No language, location or any other kind of restriction was made during the streaming process. In fact, their collection consists of tweets in foreign languages. They use Google translate to convert it into English before the annotation process. Each tweet is labeled by a human annotator as positive and negative.

Collecting the datasets

We're going to start by trying to use the movie reviews database that is part of the NLTK corpus. From there we'll try to use words as "features" which are a part of either a positive or negative movie review. The NLTK corpus movie_reviews data set has the reviews, and they are labeled already as positive or negative.

Data cleaning and preprocessing

So far, the data we have collected from various resources is in the raw form hence we cannot use it directly for our algorithm.

After Collecting the data next task is to pre-process it.

We will do data cleaning by removing stop words and punctuations.

Remove Stop Words

Features Extraction

Given the huge size of our datasets, we need to extract those features only which will be important to us.

We use the bag-of-words feature. Here, we clean the word list (i.e. remove stop words and punctuation). Then, we create a dictionary of cleaned words.

Classification Methodology

We used multiple popular classifiers to train and predict, namely Support vector Machines, SGD regression and logistic regression, Naïve Bayes, MNB classifier, Bernoulli classifier and Logistic Regression Classifier and K-nearest Neighbor Algorithm.. The scikit-learn library in python made it easy for us to implement these classifiers. We carried out a stratified shuffle split, reserving 20% of the data for testing purpose. The stratified shuffle split helps preserve the class distribution of our original data in the train-test split.

TOOLS AND TECHNOLOGY USED

TOOLS

1. Google Colaboratory- Colaboratory is a research tool for machine learning education and research. It's a Jupyter notebook environment that requires no setup to use

2. Python libraries - Used for coding the entire project. As it has various libraries pertaining to machine learning, such as pandas, scikit-learn, seaborn etc.

3. Tweepy - Tweepy is a Python library for accessing the Twitter API. It is great for simple automation and creating twitter bots. Tweepy has many features.

IMPLEMENTATION AND CODING

Data Preprocessing

```
[123]  def clean_words(words, stopwords_english):
           words_clean = []
           for word in words:
               word = word.lower()
               if word not in stopwords_english and word not in string.punctuation:
                   words_clean.append(word)
           return words_clean

       # feature extractor function for unigram
       def bag_of_words(words):
           for i in range ( len(words) ) :
                tup = (words[i],)
                words[i]= tup
           words_dictionary = dict([word, True] for word in words)
           return words_dictionary

       # feature extractor function for ngrams (bigram)
       def bag_of_ngrams(words, n=2):
           words_ng = []
           for item in iter(ngrams(words, n)):
               words_ng.append(item)
           words_dictionary = dict([word, True] for word in words_ng)
           return words_dictionary

       # feature extractor function for ngrams (bigram)
       def bag_of_3grams(words, n=3):
           words_ng = []
           for item in iter(ngrams(words, n)):
               words_ng.append(item)
           words_dictionary = dict([word, True] for word in words_ng)
           return words_dictionary

       '''
```

BAG OF WORD MODEL:

```
# let's define a new function that extracts all features
# i.e. that extracts all unigram and bigrams and trigrams features
def bag_of_all_words(words, mode):
    words_clean = clean_words(words, stopwords_english)
    words_clean_for_bigrams = clean_words(words, stopwords_english_for_bigrams)

    unigram_features = bag_of_words(words_clean)
    bigram_features = bag_of_ngrams(words_clean_for_bigrams)
    trigram_features = bag_of_3grams(words_clean_for_bigrams)
    all_features = unigram_features.copy()

    if mode == "unigram":
         return all_features

    if mode == "bigram":
         all_features.update(bigram_features)
         return all_features

    if mode == "trigram" :
         all_features.update(trigram_features)
         return all_features

print ("Unigram features are " , bag_of_all_words(words, mode = "unigram"))
print ("Bigrams features are " , bag_of_all_words(words, mode = "bigram"))
print ("Trigrams features are " , bag_of_all_words(words, mode = "trigram"))
```

CLASSIFIER MODELS:

```
classifier = NaiveBayesClassifier.train(train_set)
print("Original Naive Bayes Algo accuracy percent:", (nltk.classify.accuracy(classifier, test_set))*100)
classifier.show_most_informative_features(15)


MNB_classifier = SklearnClassifier(MultinomialNB())
MNB_classifier.train(train_set)
print("MNB_classifier accuracy percent:", (nltk.classify.accuracy(MNB_classifier, test_set))*100)

BernoulliNB_classifier = SklearnClassifier(BernoulliNB())
BernoulliNB_classifier.train(train_set)
print("BernoulliNB_classifier accuracy percent:", (nltk.classify.accuracy(BernoulliNB_classifier, test_set))*100)

LogisticRegression_classifier = SklearnClassifier(LogisticRegression())
LogisticRegression_classifier.train(train_set)
print("LogisticRegression_classifier accuracy percent:", (nltk.classify.accuracy(LogisticRegression_classifier, test_set))*100)

SGDClassifier_classifier = SklearnClassifier(SGDClassifier())
SGDClassifier_classifier.train(train_set)
print("SGDClassifier_classifier accuracy percent:", (nltk.classify.accuracy(SGDClassifier_classifier, test_set))*100)

##SVC_classifier = SklearnClassifier(SVC())
##SVC_classifier.train(train_set)
##print("SVC_classifier accuracy percent:", (nltk.classify.accuracy(SVC_classifier, test_set))*100)

LinearSVC_classifier = SklearnClassifier(LinearSVC())
LinearSVC_classifier.train(train_set)
print("LinearSVC_classifier accuracy percent:", (nltk.classify.accuracy(LinearSVC_classifier, test_set))*100)

NuSVC_classifier = SklearnClassifier(NuSVC())
NuSVC_classifier.train(train_set)
print("NuSVC_classifier accuracy percent:", (nltk.classify.accuracy(NuSVC_classifier, test_set))*100)
```

RESULTS:

```
Original Naive Bayes Algo accuracy percent: 75.5
Most Informative Features
          ('ludicrous',) = True              neg : pos    =     12.6 : 1.0
       ('effortlessly',) = True              pos : neg    =     12.3 : 1.0
        ('uninvolving',) = True              neg : pos    =     12.3 : 1.0
         ('one', 'worst') = True             neg : pos    =     12.2 : 1.0
       ('well', 'worth') = True              pos : neg    =     11.7 : 1.0
             ('warns',) = True               pos : neg    =     11.7 : 1.0
            ('avoids',) = True               pos : neg    =     11.7 : 1.0
            ('seagal',) = True               neg : pos    =     11.7 : 1.0
       ('outstanding',) = True               pos : neg    =     11.2 : 1.0
        ('astounding',) = True               pos : neg    =     11.0 : 1.0
    ('steven', 'seagal') = True              neg : pos    =     11.0 : 1.0
           ('alicia',) = True                neg : pos    =     11.0 : 1.0
         ('palpable',) = True                pos : neg    =     10.3 : 1.0
        ('insulting',) = True                neg : pos    =      9.8 : 1.0
         ('seamless',) = True                pos : neg    =      9.7 : 1.0
MNB_classifier accuracy percent: 85.0
BernoulliNB_classifier accuracy percent: 67.25
LogisticRegression_classifier accuracy percent: 84.5
SGDClassifier_classifier accuracy percent: 84.25
LinearSVC_classifier accuracy percent: 85.5
NuSVC_classifier accuracy percent: 85.0
```

RESULT ANALYSIS

Out of the 2000 items in the data set, various machine learning models were trained on 1600 items of the data set and tested on 400 items of the data set.

The following results were obtained :-

MODELS	**UNIGRAM**	**UNIGRAM +BIGRAM**	**UNIGRAM+BIGRAM+TRIGRAM**
Naïve bayes	**66.75 %**	**78.5 %**	**75.75 %**
Linear SVC	**87.5 %**	**87.75 %**	**85.0 %**
SGD classifier	**82.5 %**	**85.25 %**	**80.25 %**
MNB classifier	**82.75 %**	**83.0 %**	**82.0 %**
Bernoulli classifier	**76.75 %**	**67.75 %**	**61.75 %**
Logistic Regression Classifier	**88.75 %**	**87.75 %**	**85.5 %**
NuSVC Classifier	**86.5 %**	**89.5 %**	**84.25 %**

As we can see here we got the best accuracy from NuSVC Model and thus we can conclude that this machine learning model would be best suited for handoff as its accuracy rounds up to 89.5 %. We have also found that tweets from twitter is found to be approximately 80% predictable average. Thus we can conclude that sentimental analysis for movie review is achieved by easy means and therefore prediction for tweets is successfully achieved.

For each tweet which we take from tweepy , it is predicted over all the seven models .if more than 3 out of 7 says it is a positive tweet then it is considered as a positive tweet else it will consider as a negative tweet.

CONCLUSION AND FUTURE SCOPE

We presented results for sentiment analysis on Twitter. We use previously proposed state-of-the-art unigram model as our baseline and report an overall gain of over 4% for two classification tasks: a binary, positive versus negative and a 2-way positive versus negative. We presented a comprehensive set of experiments for both these tasks on manually annotated data that is a random sample of stream of tweets. We investigated seven kinds of models:
Naïve Bayes, Linear SVC, SGD classifier, MNB classifier, Bernoulli classifier, Logistic Regression, Classifier and NuSVC classifier.
For our feature-based approach, we do feature analysis which reveals that the most important features are those that combine the prior polarity of words and their parts-of-speech tags. We tentatively conclude that sentiment analysis for Twitter data is
not that different from sentiment analysis for other genres. In future work, we will explore even richer linguistic analysis, for example, parsing, semantic analysis and topic modeling.

REFERENCES

[1] Apoorv Agarwal, Fadi Biadsy, and Kathleen Mckeown. 2009. Contextual phrase-level polarity analysis using lexical affect scoring and syntactic n-grams. Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2017), pages 24–32, March.
[2] Luciano Barbosa and Junlan Feng. 2055 Robust sentiment detection on twitter from biased and noisy data. Proceedings of the 23rd International Conference on Computational Linguistics: Posters
[3] G.Vinodhini and RM.Chandrasekaran, “Sentiment Analysis and Opinion Mining: A Survey”, Volume 2, Issue 6, June 2012 ISSN: 2277 128X International Journal of Advanced Research in Computer Science and Software Engineering
[4] Sentiment analysis of Twitter data: Case study on digital India
Prerna Mishra ; Ranjana Rajnish ; Pankaj Kumar