MOVIE REVIEWS

SENTIMENT ANALYSIS BASED ON ASPECT-LEVEL

Abhishek Kumar
Information Technology
SRM Institute of Science and Technology
Chennai, India
abhishekkumar0598@outlook.com

Kashish Kharyal
Computer Science
SRM Institute of Science and Technology
Chennai, India
kharyalkashish.3@outlook.com

***Abstract*— The aspect-level sentiment analysis refers to the detection of emotional/sentiment polarity from random text on a well-configured element or level of visibility. The research using element trend analysis using opinion pieces. User opinions on various parts of a film, like performance, orientation, performance art, shooting, and so on, are common for user ratings. we studied a new linguistics regulation system that recognizes elements in opinion pieces, uncovers viewpoint in that feature, & uses linguistics techniques the estimate an emotion word of that perspective. Sentiment analysis has become a popular tool for determining how people feel regarding a service, product, a public issue or the location. When a text has to be sentimentally assessed, a machine learning technique is often applied. Each machine learning model's classification accuracy may be further refined by hyperparameter tweaking for obtaining greater accurateness than when model's default hyperparameter values are used.**

***Keywords*—Aspect-level sentiment analysis · Linguistics Techniques · Opinion profiling**

I. INTRODUCTION

Sentiment Analysis is a sort of artificial intelligence function and categorizes subjective data onto favourable, negative, and impartial (no sentiment polarity) categories using software techniques. There are three ways to look at sentiment levels: Statement layer (give orientation to each word), document level (allocate polarization to each article), header stage (attach bias it to entire page), & facet stage (allocate voltage to each facet) (assign polarity to aspect). In many different situations, Sentiment analysis is helpful. Many businesses use sentiment analysis to figure out the thought process of their customers. For effective organizational decision-making, opinions on products/services are needed. In addition, sentiment analysis may be used for a variety of objectives. As an example, Movie evaluations assist viewers in deciding which film to see. Earlier According to research, the majority that quantity of research effort has been elevated of the document and Recently at the level aspect, scientists have begun doing studies. Since at relation to the issue, research & studies using trend analysis.

Sentiment Analysis [1] is a field of Natural Language Processing research that focuses on assessing text-based perspectives. Product reviews, customer feedback, political discourse, and other applications for sentiment analysis already exist. The process of dissecting a review sentence into smaller fragments to gain a more precise and nuanced understanding of the review sentence is known as aspect-based sentiment analysis (ABSA). Sentiment analysis is used by large corporations to understand consumer experience and product reputation, as well as to monitor brand and public opinion. Consider the following restaurant review: The food was delicious, but the service was appalling. Because a single line contains multiple emotions and topics in this case, categorising the entire review as positive or negative would be inaccurate. ABSA enters the scene by extracting the statement's aspect terms and assigning each one an emotion polarity. Meal and Service are two aspect names in this case, with Positive ('The food was delectable') and Negative ('The service was subpar') sentiment polarities. Aspect-Based Sentiment Analysis is divided into two parts: aspect term extraction and aspect polarity classification.

Streaming services like Amazon Movies, Hulu as well as Netflix, have made it easier for individuals to express themselves through various media. Text-based, video-based, and rating-based reviews all demonstrate a spectrum of emotions, ranging from happiness to sadness. In order to better understand customer preferences and enhance the customer experience, businesses use opinion mining to extract feelings from text. There may be misspellings, acronyms, emoticons, and slang that make these instructions difficult to comprehend. To obtain accurate analyses of how users feel, it is necessary to determine the most efficient method for mining opinions. The organization employs experts in sentiment analysis to determine the significance of the collected data. Sentiment analysis is a computerized examination of how individuals feel about a product. It is one of the most researched subjects.

Text sentiment study is extraction of the subjective data out of the text. It's gaining popularity because it's highly effective as well as it can rapidly analyse thousands of opinions using a variety of techniques. Among the many uses are:

- Recommendation Methodology
- Improving Customer Service and Product Quality
- Administrative Final Thoughts
- Preferences of Customers
- An examination of marketing strategy

The IMDB dataset for movie review analysis was obtained from Kaggle for this study. Different sentiment analysis models will be developed based on these data for a variety of film genre characteristics. This study aims to develop classification models that can determine whether a viewer is positive or negative. This study will employ Aspect-Based Sentiment Analysis rather than traditional sentiment analysis, which only identifies emotions without defining their components (ABSA). This will aid in the examination of popular film jargon, slang, emoticons, and typos. Slang, acronyms, emoticons, and typographical errors are all examples of Aspect Extraction (AE). To complete this task, each input word must be classified as B, I, or O, with B denoting the start, I denoting the inside, and O denoting the outside. Because facets can contain two or more terms, the position of each term must be indicated. Positive and negative emotions are classified using Aspect Classification (ASC). The work comprises acquiring review-data out of the IMDB website, evaluating the dataset for the absent data or the outliers, ensuring that it is not skewed, and then developing models on the cleansed dataset. In addition to the AE and ASC aspect component factors and genre driving variables, Support Vector Machine, Logical Regression, and Nave Bayes will be employed.

The subject of this paper's study is opinion examination at the viewpoint level. We suggested a strategy that uses linguistics approaches to identify characteristics from movie reviews and compute sentiment polarity. The polarity of one specific component is summed up across all reviews, and a sentiment profile for the film is created.

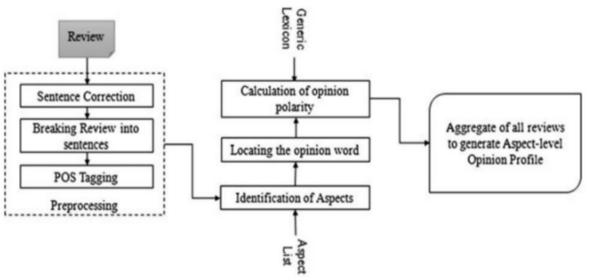
II. PROPOSED METHODOLOGIES

1. Introduction

User reviews of movies serve as the processing input. Using a crawler, our system can collect user reviews for a certain film. In freeform language, each review reflects the user's thoughts on various parts of a certain film. As a result, the assignment is like (i) identify element terms; (ii) find out what others think on every topic; (iii) calculate the sentiment orientation for one feature Furthermore, any such technique is designed to create any intrinsic part perspective synopsis of a movie based on the ratings. This chart illustrates this circuit diagram using facet trend analysis. To begin, the review content is pre-processed by replacing numerous punctuation marks with single punctuation marks, such as "????" replacing "?." Then, using delimiters— question mark (?), apostrophe (!), and final stop —each review is divided into sentences (.). After that, Stanford POS Tagger is used to parse the sentences. a. Recognize Aspect terms Because we've been using aspect-based analytics, identifying a component is indeed a requirement and we'll go through this section. Another or many critics have explored all conceivable aspects of the film in many articles. We examined every one of the accessible variables is specified by major film prizes, movie review websites, or cinema periodicals. Because different terms might be used to represent certain elements in different evaluations. Acting, for example, may be represented through act, portrayal, and so on. provided a list of aspect vectors, which we employed. Using an aspect vector list, we have used the component detection method to discover the facet in every statement.

A. Locate Opinions about Each Aspect

Following the identification of the features, a next stage is to learn what others think about them. One or more aspects may be included in the sentence. Figure depicts a hypothetical sentence construction, in which $SLn$and SRn denote 1 or even more phrases on comment upon that L.H.S. & R.H.S. of appearance An, together. They discovered the facet initially but looked to the political spectrum about an attitude phrase. According to the POS tagger, an attitude term is just an adjective. We'll go over the complete procedure for allocating whichever judgment language to an interface called in the following section.

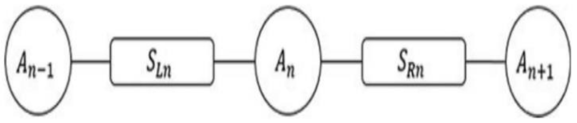


Review
Sentence Correction
Breaking Review into sentences
POS Tagging
Preprocessing
Generic Lexicon
Calculation of opinion polarity
Locating the opinion word
Identification of Aspects
Aspect List
Aggregate of all reviews to generate Aspect-level Opinion Profile

Fig. 1 Block diagram for aspect-based sentiment analysis

B. Sentiment Polarity Computation

The sentiment polarity method was used, as shown here. Following the detection of the feature, its engine travels right & left in the pursuit of the word having a different opinion. If the word is determined to be an adverb, the Mainly covers AAC process is employed to compute a sentiment rating of all facets. SentiWordNet Adjective Adverb Combination (SWN AAC) is the acronym for SentiWordNet adjective adverb combination. To compute the emotion score, we also



employed Generic Lexicon as a lexical dictionary.

Fig. 2 Sentence structure

Pseudocode for Aspect Detection Algorithm

```
1 Locate the aspect $(A_n)$
2 FOR each located aspect
    1. Initialize $S_{LA_n} = 0, S_{RA_n} = 0,$
    2. TRAVERSE LEFT in search of polarity feature
    3. IF polarity feature adj THEN
       i. $S_{LA_n} = SWN_A AC$( adj, Sentence )
    4. TRAVERSE RIGHT in search of polarity feature
    5. REPEAT step 3, then we will have $S_{RA_n}$
    6. $S_{A_n} = S_{LA_n} + S_{RA_n}$
    7. $IF\ (S_{LA_n} == 0 \&\& S_{RA_n} == 0) THEN$
       i. POL_TO_DET_LIST (An)
3. FOR each $A_n \in$ POL_TO_DET_LIST
    1. Locate $A_n$ in LIST
    2. TRAVERSE LEFT tO $A_{n-1}$
       i. $IF\ (S_{RA_{n-1}} != 0)$ THEN
            • $S_{LA_n} = S_{RA_{n-1}}$
       ii. ELSE IF $(S_{LA_{n-1}}$ != 0) THEN
• $S_{LA_n} = S_{LA_{n-1}}$
    3. $IF\ (S_{LA_n} = 0 \&\& S_{RA_n} = 0)$ THEN
       i.     TRAVERSE RIGHT to $A_{n+1}$
• IF $(S_{LA_{n+1}} != 0) THEN$
            ○ $S_{RA_n} = S_{LA_{n+1}}$
            • $ELSE$ IF $(S_{RA_{n+1}} != 0)$
              THEN
            ○ $S_{RA_n} = S_{RA_{n+1}}$
```

2. Machine Learning Models:

The goal of this research is to categorize the reviews as negative or positive. As a result, we investigated several classification models using the aforementioned feature representations for this classification job. We utilized a variety of models, from basic Logistic Regression to the cutting-edge SVM Classifier. Other classification models that we used were SGD Classifier as well as the Random Forest Classifier. Apart from that, we trained the aforementioned feature representations using the Nave Bayes' Classifier, which is commonly used in conjunction with Bag of Words and Text Vectorization in text mining. We utilized sklearn modules for all of the above models, tweaking their parameters but without modifying their implementations, therefore we won't delve into their theory in this report.

We have used the Mean Absolute Error rather than Mean Squared Error to assess performance (MSE). Because MAE will inform us precisely how much misclassification we're making with each model, this is a good thing. We employed cross-validation methodologies to

fit parameters on set selection during these training operations, as previously indicated.

3. Dataset preparation:

Two datasets were used in this study investigation. These data sets were gathered from a movie database on the internet (IMDb). There are sentences in each dataset. Train and Test are the names given to these two datasets. The Train–Test datasets have been wriggled from www.imdb.com, manually pre-processed and annotated as well as analysed by three major authors, and eventually analysed for accuracy by two additional impartial authors. For every author delegated directionality to a different aspect of the data source. This same tagging efficiency was calculated using the cross-contract performance measures Cross Webservice Continuity (IIC) and Cohen's Kappa Statistics. The results of the analysis point to a high level of precision.

Each document's data is in CSV format. To make the data legible by these technologies, some alteration is required. Figure 3 depicts the transformation steps. To do the change, we used Python and the packages NumPy and pandas to iteratively load the TXT file into the above-mentioned folders. A pandas is created from all of the documents read from the TXT files. Data Frame is a useful tool for creating CSV files. In the Data Frame, we randomly selected 1,000 instances from the dataset's 25,000 documents and separated them into three sets: development (20%), cross-validation (70%), and final test (100%). (10%). Within this operation, non-ASCII symbols are indeed removed from the data. These datasets are exported from Python in CSV format. These datasets are then put into WEKA in order to generate an ARFF format file. In addition, the output CSV from the pandas and numpy modules is treated as nominal values rather than string values, as it was in WEKA. To overcome this issue, we utilized the NominaltoString filter in WEKA Explorer's pre-process tab to convert nominal data into string data in an ARFF file. String input, on either side, could be loaded in WEKA straight in different classifiers. Lets transform input values into vectors (each document is represented by a vector), we used StringToWordVector.

4. Feature Extraction and Machine Learning Models

In certain cases, we also use text vectorization. The purpose of this research was to explore whether using sentiment classification as a special case of topic-based categorization (with positive and negative sentiment as the two "themes") would suffice, or if more specialized sentiment-categorization techniques would be necessary. Logistic regression, Naive Bayes classification, and support vector machines were the three techniques we employed. These three algorithms have quite different approaches to text categorization, yet each has been demonstrated to be successful in previous experiments. For document data, usual bag-of-features architecture has been used to build these machine learning techniques. Let {f1,..., fm} represent a preset collection of m features in a text, such as the word "still" or the bigram "really stinks." There are ni instances of fi in document (d:= (n1(d), n2(d)) is a document vector (d).

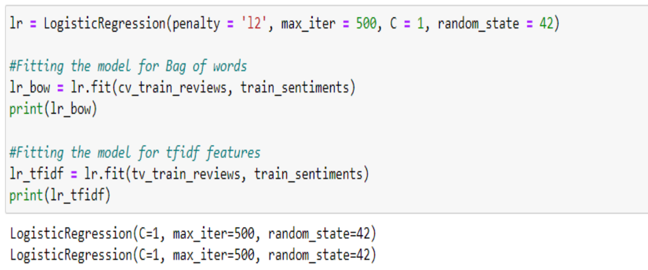
In addition, we apply text vectorization in certain situations. Frequency of Term Inverse Document Frequency (TFI-DF) analysis is a simple and reliable approach for determining a text's context. In a longer text, term frequency and inverse document frequency are used to locate related material and essential terms and phrases. TF-IDF analysis is simple to implement with Python. Computers are unable to comprehend the meaning of writing, but they can comprehend numbers. The link between the words may be understood by converting them to numbers. The TF-IDF is the result of TF and IDF. The TF-IDF is typically one of the finest metrics for determining the importance of a phrase in a text. It denotes the significance of a word in a certain document.

The problem with such approaches is that they are incapable of comprehending synonyms, semantics, and other emotional components of language. Large and huge, for example, are synonyms, but such systems cannot distinguish between them. On two sentence datasets, we got the aspect-level sentiment analysis result. The findings were assessed using conventional accuracy and recall measures. The accuracy and recall were calculated using the macro-averaging approach.

III. EXPERIMENTS AND EVALUATION

1. Logistic Regression

When thresholds are applied to the probability predicted for each class, logistic regression may be utilised to solve classification issues. The probability of the classes are calculated using either the Sigmoid or Softmax functions. We shall utilise the Sigmoid function in our situation because we do not have a multiclass classification problem. We can utilise Scikit Learn to ensure that our implementation of logistic regression was correct by comparing the performance measurements. This is accomplished as follows:



```
lr = LogisticRegression(penalty = 'l2', max_iter = 500, C = 1, random_state = 42)

#Fitting the model for Bag of words
lr_bow = lr.fit(cv_train_reviews, train_sentiments)
print(lr_bow)

#Fitting the model for tfidf features
lr_tfidf = lr.fit(tv_train_reviews, train_sentiments)
print(lr_tfidf)
```

```
LogisticRegression(C=1, max_iter=500, random_state=42)
LogisticRegression(C=1, max_iter=500, random_state=42)
```

Fig.3. Implementation of Logistic Regression

The accuracy and recall values produced by the aspect detection method are presented in the table.

Table 1 Classification Report

The algorithm has a recall of approximately 76% and an accuracy of 75%. That chart shows the recall and precision achieved either by emotion computation technique. Z inside the list denotes the average number of all elements whereby the system identified valence. A 11 sentiments profile is generated for numerous components of a movie based here on aggregate the aspect-based sentiment orienting findings received for every evaluation.

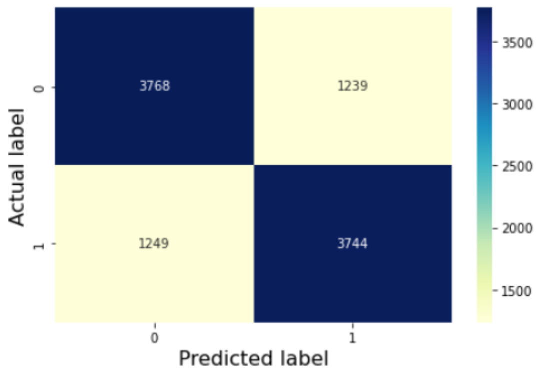


Fig. 4 Confusion matrix for Bags of Words Model features

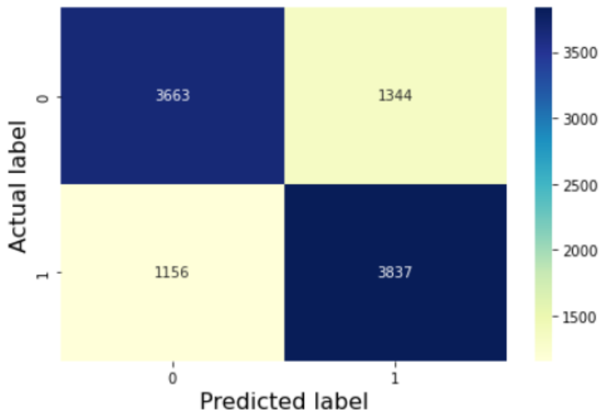
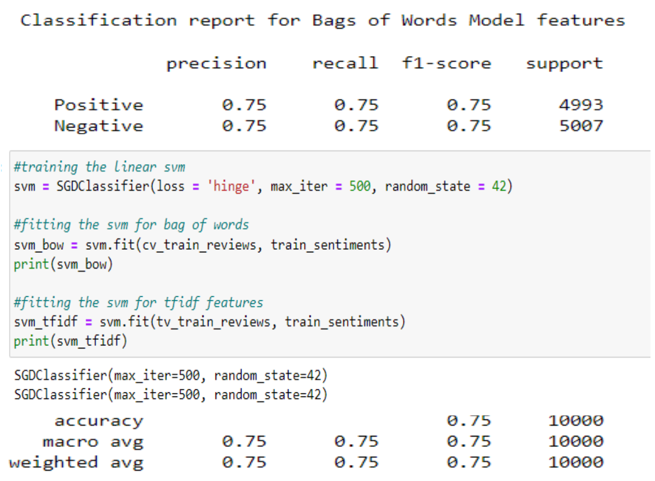


Fig. 5 Confusion matrix for TF_IDF features

2. Support Vector Machines

SVMs typically outperform Naive Bayes in the context of standard text classification (Joachims, 1998). They are margin-based classifiers as opposed to probabilistic classifiers such as Naive Bayes as well as MaxEnt. In a two-category case, the main goal of training strategy is for finding a hyperplane, shown in the form of the vector w, that not only separates document vectors from one class from those from the other, but also gives the most space between the two classes.

$$\vec{w} := \sum_j \alpha_j c_j \vec{d}_j, \alpha_j \geq 0$$



```
Classification report for Bags of Words Model features

              precision    recall  f1-score   support

    Positive       0.75      0.75      0.75      4993
    Negative       0.75      0.75      0.75      5007
```

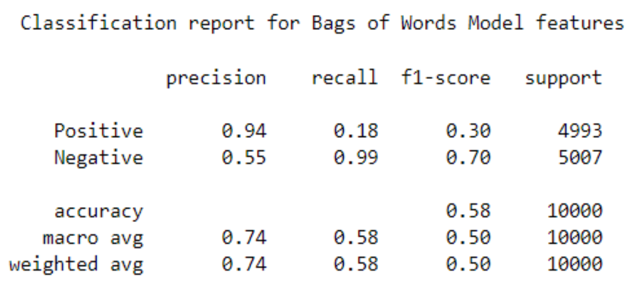
```
#training the linear svm
svm = SGDClassifier(loss = 'hinge', max_iter = 500, random_state = 42)

#fitting the svm for bag of words
svm_bow = svm.fit(cv_train_reviews, train_sentiments)
print(svm_bow)

#fitting the svm for tfidf features
svm_tfidf = svm.fit(tv_train_reviews, train_sentiments)
print(svm_tfidf)
```

```
SGDClassifier(max_iter=500, random_state=42)
SGDClassifier(max_iter=500, random_state=42)
    accuracy                           0.75     10000
   macro avg       0.75      0.75      0.75     10000
weighted avg       0.75      0.75      0.75     10000
```

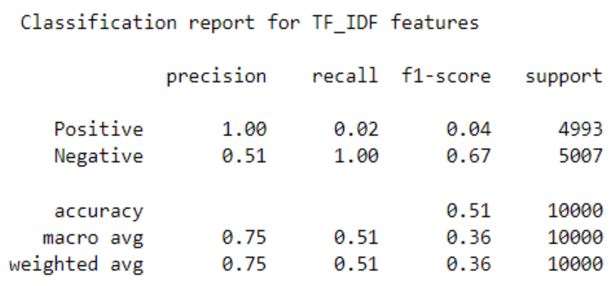
The α_j 's have been attained by solving a dual-optimization problem. Support vectors are those $\vec{d}_j$ where α_j is bigger than zero, as these are have been the only document vectors that contribute to $\vec{w}$. The α_j's are found by solving a dual optimization problem. Support vectors are those $\vec{d}_j$ where $\vec{a}_j$ is bigger than zero, since these are the only document vectors that contribute to $\vec{w}$. The classification of test cases merely entails deciding which side of the $\vec{w}$ hyperplane they fall on. Using Scikit-learn we will train our model on Linear SVM as shown below:

Fig. 6 Implementation of SVM

The accuracy and recall values produced by the aspect detection method are presented in the table



```
Classification report for Bags of Words Model features

              precision    recall  f1-score   support

    Positive       0.94      0.18      0.30      4993
    Negative       0.55      0.99      0.70      5007

    accuracy                           0.58     10000
   macro avg       0.74      0.58      0.50     10000
weighted avg       0.74      0.58      0.50     10000
```



```
Classification report for TF_IDF features

              precision    recall  f1-score   support

    Positive       1.00      0.02      0.04      4993
    Negative       0.51      1.00      0.67      5007

    accuracy                           0.51     10000
   macro avg       0.75      0.51      0.36     10000
weighted avg       0.75      0.51      0.36     10000
```

TABLE 2 Classification Report

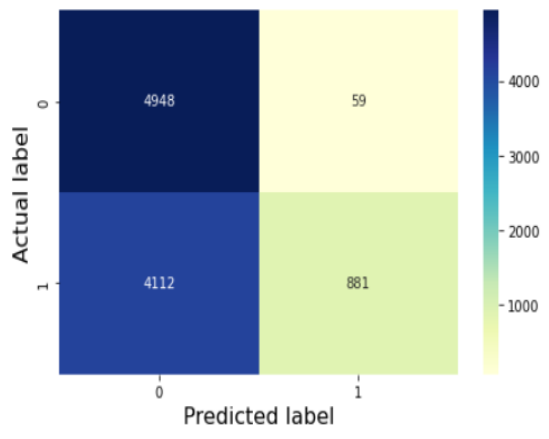


Fig. 7 Confusion matrix for Bags of Words Model features

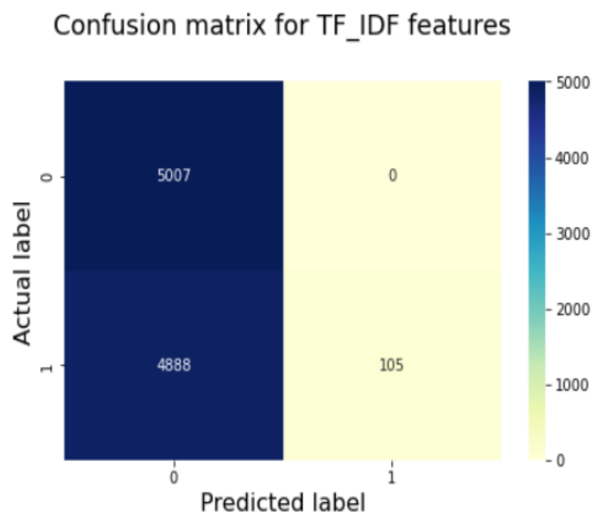


Fig. 8 Confusion matrix for TF_IDF features.

3. Naïve Bayes:

Assigning the class $c^* = arg\ max_c P(c \mid d)$ to a given document d is one method of text categorization. The Naive Bayes (NB) classifier is derived by first recognising that, according to Bayes' rule,

$$P(c \mid d) = \frac{P(c)P(d \mid c)}{P(d)},$$

$P(d)$ has no bearing on the selection of c^*. Naive Bayes decomposes the term $P(d \mid c)$, by assuming that the f_is are conditionally independent given d's class:

$$P_{NB}(c \mid d) := \frac{P(c)\left(\prod_{i=1}^{m} P(f_i \mid c)^{n_i(d)}\right)}{P(d)}$$

Our training technique uses add-one smoothing to estimate the relative frequency of $P(c)$ as well as $P(f_i \mid c)$.

Despite its simplicity and the fact that its conditional independence assumption does not hold in the real world, Naive Bayes-based text categorization performs admirably (Lewis, 1998); in fact, Domingos and Pazzani (1997) demonstrate that Naive Bayes has been optimum for the specific issue classes with reliable characteristics. Using a Naive Bayes model, we were able to determine sentence polarity with a recall of approximately 76% and an accuracy of 75%.

TABLE 3: Classification Report

```
 Classification report for Bags of Words Model features

              precision    recall  f1-score   support

    Positive       0.75      0.76      0.75      4993
    Negative       0.75      0.75      0.75      5007

    accuracy                           0.75     10000
   macro avg       0.75      0.75      0.75     10000
weighted avg       0.75      0.75      0.75     10000

 Classification report for TF_IDF features

              precision    recall  f1-score   support

    Positive       0.75      0.76      0.75      4993
    Negative       0.75      0.74      0.75      5007

    accuracy                           0.75     10000
   macro avg       0.75      0.75      0.75     10000
weighted avg       0.75      0.75      0.75     10000
```

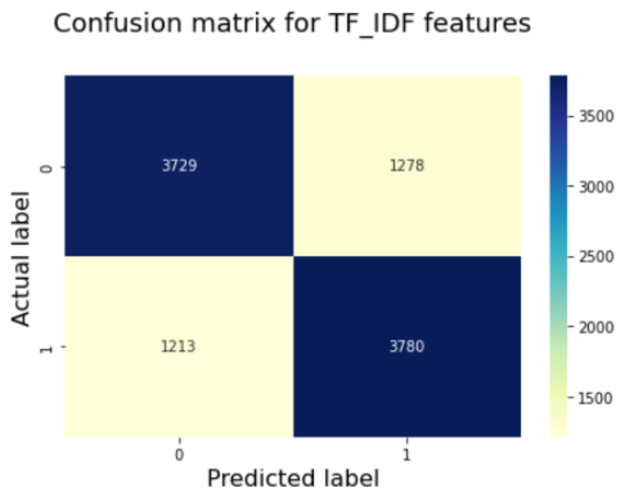


Fig. 9 Confusion matrix for Bags of Words Model Features

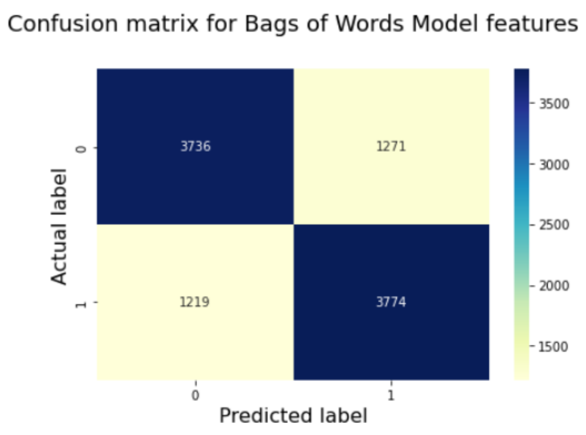


Fig. 10 Confusion matrix for TF_IDF Features

well
littl
master
product
techniqu
realism
particularli
halliwel
piec
veri
onli
sens
hal
michael
sometim
comfort
fashion
got
sheen
voic
actor
give
discomfort
unassum
entir
pat
truli
edit
film
dream
oldtimebbc

Fig. 11 Word cloud for Positive Review words

film
keitel
look
bad
oneonli
watch
comment
forward
worst
encourag
mistak
live
way
wa
edit
extrem
almost
everi
truli
pace
seen
act
bore
song
soundtrack
play
countri
rare
time
cheap
give
happi
lame
storylin

Fig. 12 Word Cloud for Negative Review Words

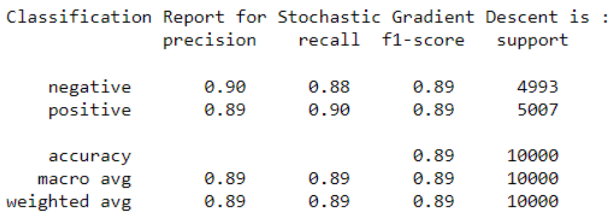
Data leakage is a serious concern when utilising machine learning to build predictive models. When information from outside the training dataset is utilised to generate the model, this is known as data leaking. Predictive modelling purpose is to create a model that can make accurate predictions based on fresh data not observed during training. This is a complicated situation. It's troublesome since we can't test the model against something we don't have. As a result, we must estimate the model's performance on unknown data by training and assessing it on a portion of the data. This approach underpins cross validation and other advanced processes that aim to limit the variation in this estimate. Forecasting models may be unduly optimistic, if not entirely wrong, as a result of data breaches. Data leakage happens when outside data ior creating model. This additional input may permit the model to learn or know something it wouldn't have known otherwise, undermining the mode's expected performance. Data preparation (for example, dealing with missing information, scaling/encoding, and feature extraction) is the first step in the machine learning process. While learning this method, we do the data preparation one step at a time. This may take some time since we need to prepare both training and testing data. Pipelines allow us to streamline this process by combining the preliminary processes as well as making the model modification along with monitoring more convenient. Scikit-Pipeline Learn's class gives a framework to steer a series of the data changes followed by an estimation (Mayo, 2017). There are many benefits when implementing a Pipeline:

- Convenience and grouping: We use the fit function to fit an entire series of estimators, predicting the data only once.
- Joint parameter selection: A grid-search can be used to select the appropriate values for the parameters of all estimators in the pipeline.
- Pipelines aid in preventing testing data from entering a cross-validated model that has already been trained. Ensure that the predictors and transformers were trained on the same samples to accomplish this.

So, for reducing the data leakage and get better accuracy we use a Pipeline to minimise the error rate or data leakage which help to boost our accuracy as we tried eight different classifiers. Logistic Regression, Decision Tree Classifier, Random Forest classifier, Stochastic Gradient Descent, Gradient Boosting Classifier, XGB Classifier, Multinomial Naïve Bayes Classifier, Bernoulli Naïve Bayes Classifier. Pipelines contain our pre-processing procedures and models, simplifying the machine learning workflow. If necessary, we may perform several pre-processing steps before fitting a model into the pipeline.

After using Pipeline to reduce the data leakage our accuracy of Logistic regression model and Stochastic Gradient Descent model is increased to approximately 89% which is shown below:

Table 4 Classification Report



```
Classification Report for Stochastic Gradient Descent is :
              precision    recall  f1-score   support

    negative       0.90      0.88      0.89      4993
    positive       0.89      0.90      0.89      5007

    accuracy                           0.89     10000
   macro avg       0.89      0.89      0.89     10000
weighted avg       0.89      0.89      0.89     10000
```

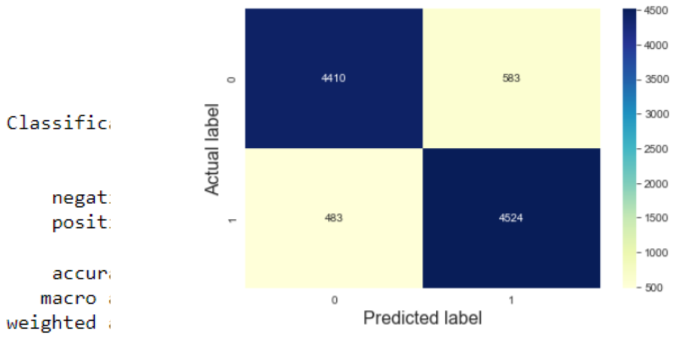
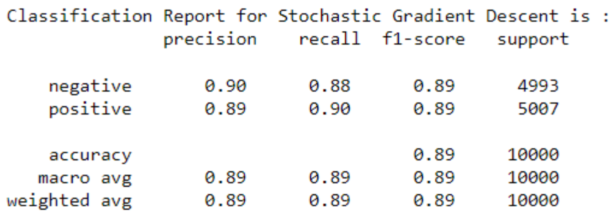


Fig. 13 Confusion matrix for Logistic Regression using TF-IDF Vectorizer

Table 5 Classification Report



```
Classification Report for Stochastic Gradient Descent is :
              precision    recall  f1-score   support

    negative       0.90      0.88      0.89      4993
    positive       0.89      0.90      0.89      5007

    accuracy                           0.89     10000
   macro avg       0.89      0.89      0.89     10000
weighted avg       0.89      0.89      0.89     10000
```

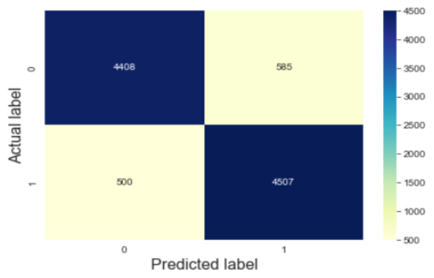


Fig 14 Confusion matrix for Stochastic Gradient Descent using TF-IDF Vectorizer

Accuracy and Confusion matrix of Decision Tree Classifier, Random Forest classifier, Gradient Boosting Classifier, XGB Classifier, Multinomial Naïve Bayes

Classifier and Bernoulli Naïve Bayes Classifier are displayed below:

Table 6 Classification Report

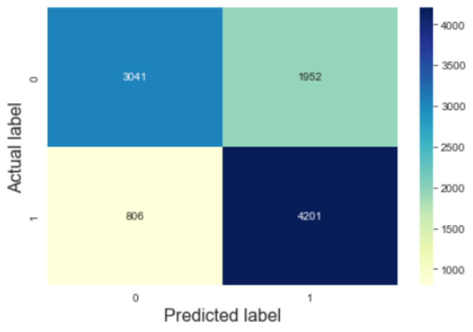
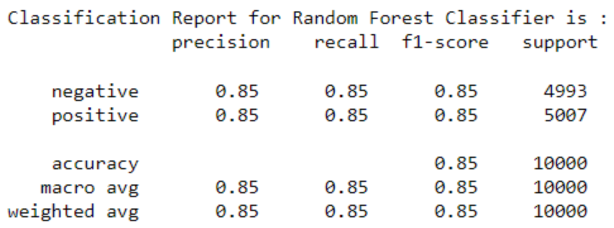


Fig 15 Confusion matrix for Decision Tree Classifier using TF-IDF Vectorizer

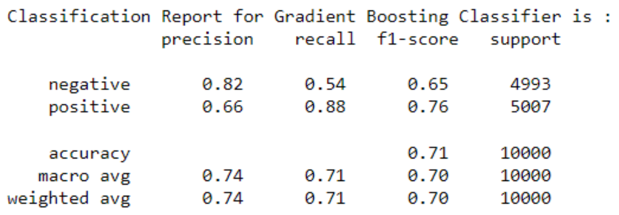
Table 7 Classification Report



```
Classification Report for Random Forest Classifier is :
              precision    recall  f1-score   support

    negative       0.85      0.85      0.85      4993
    positive       0.85      0.85      0.85      5007

    accuracy                           0.85     10000
   macro avg       0.85      0.85      0.85     10000
weighted avg       0.85      0.85      0.85     10000
```

Fig 16 Confusion matrix Random Forest Classifier using TF-IDF Vectorizer

Table 8 Classification Report



```
Classification Report for Gradient Boosting Classifier is :
              precision    recall  f1-score   support

    negative       0.82      0.54      0.65      4993
    positive       0.66      0.88      0.76      5007

    accuracy                           0.71     10000
   macro avg       0.74      0.71      0.70     10000
weighted avg       0.74      0.71      0.70     10000
```

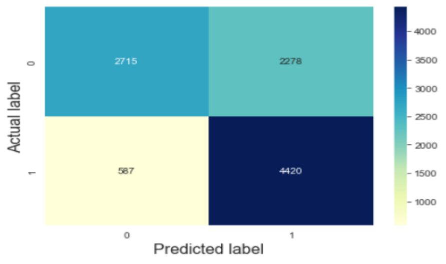
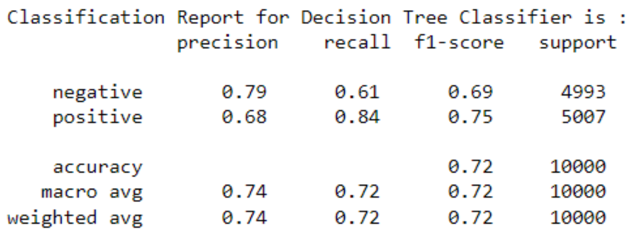
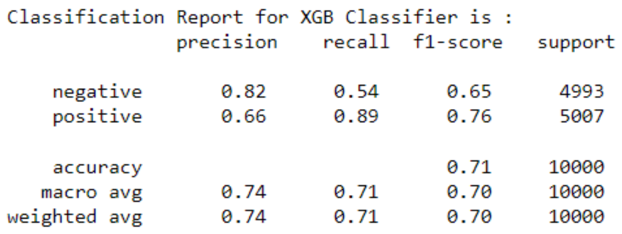


Fig 17 Confusion matrix for Gradient Boosting Classifier using TF-IDF Vectorizer

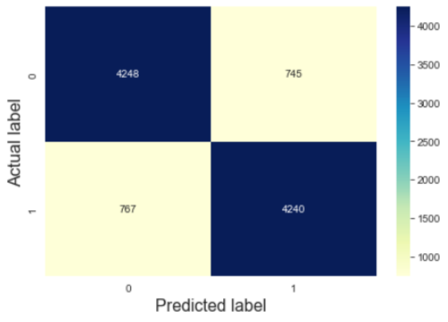
Table 9 Classification Report



```
Classification Report for Decision Tree Classifier is :
              precision    recall  f1-score   support

    negative       0.79      0.61      0.69      4993
    positive       0.68      0.84      0.75      5007

    accuracy                           0.72     10000
   macro avg       0.74      0.72      0.72     10000
weighted avg       0.74      0.72      0.72     10000
```



```
Classification Report for XGB Classifier is :
              precision    recall  f1-score   support

    negative       0.82      0.54      0.65      4993
    positive       0.66      0.89      0.76      5007

    accuracy                           0.71     10000
   macro avg       0.74      0.71      0.70     10000
weighted avg       0.74      0.71      0.70     10000
```



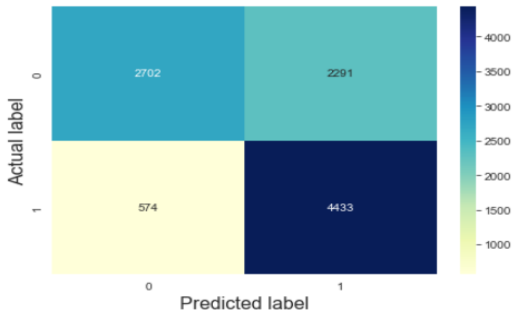


Fig 18 Confusion matrix for XGB Classifier using TF-IDF Vectorizer

Table 10 Classification Report

```
Classification Report for Multinomial Naive Bayes Classifier is :
              precision    recall  f1-score   support

    negative       0.85      0.88      0.86      4993
    positive       0.88      0.85      0.86      5007

    accuracy                           0.86     10000
   macro avg       0.86      0.86      0.86     10000
weighted avg       0.86      0.86      0.86     10000
```

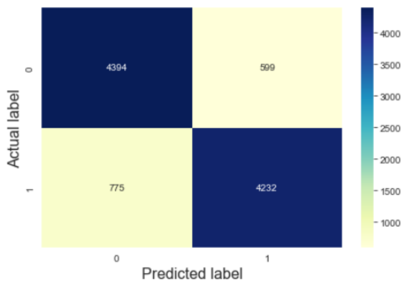


Fig. 19 Confusion matrix for Multinomial Naïve Bayes Classifier using TF-IDF Vectorizer

Table 11 Classification Report for Bernoulli Naïve Bayes Classifier is: precision recall f1-score support

```
Classification Report for Bernoulli Naive Bayes Classifier is :
              precision    recall  f1-score   support

    negative       0.84      0.88      0.86      4993
    positive       0.87      0.83      0.85      5007

    accuracy                           0.85     10000
   macro avg       0.86      0.85      0.85     10000
weighted avg       0.86      0.85      0.85     10000
```

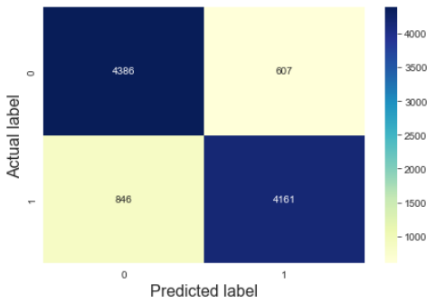


Fig. 20 Confusion matrix for Bernoulli Naïve Bayes Classifier using TF-IDF Vectorizer

From which we observed that Logistic model and Stochastic Gradient Descent giving the accuracy above 89% with precision (negative) 90% and (positive) 89%, precision (negative) 90% (positive) 88% respectively, recall (negative) 88% and (positive) 90%, recall (negative) 88% and (positive) 90% respectively and for f1-score 89% for both negative and positive for logistic model and stochastic gradient descent.

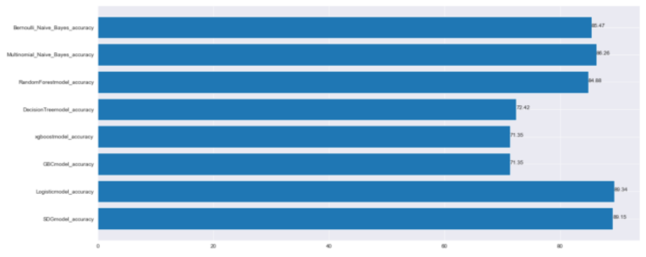


Fig. 21 Accuracy Comparison of Different Models

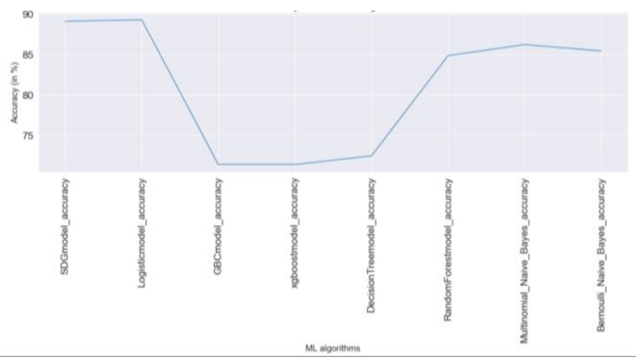


Fig. 22 Accuracy Curve of ML algorithms

IV. RESULTS AND DISCUSSIONS

As previously stated, we run three categorization models initially. On different feature depictions of information that is textual as presented in the reviews, support vector machines, Logistic Regression as well as Naïve-Bayes classification were used. The Logistic Regression performs better than the other two, whereas the Support Vector Machine Classifier failed to meet the requirements for all of the set of features, as well as we were unable to find an acceptable solution. But even though Logistic regression gave us the highest accuracy it was not good enough. So, to solve this issue, we resorted to solving the problem of data leakage and made pipelines, applied k-cross validations to avoid as much data leakage as possible to improve the accuracy and finally ran eight classifier models on the training data. The classifiers were namely Logistic Regression, Random Forest classifier, Stochastic, XGB Classifier, Gradient Descent, Decision Tree Classifier, Bernoulli Naïve Bayes as well as Gradient Boosting Classifier and Multinomial Naïve Bayes Classifier. The Logistic Regression model again appeared to be the best performer among all for all the features, with an approximate accuracy of 89.34%.

We employed Grid Search, a hyper parameter tuning strategy, to boost the accuracy of our Logistic regression model even further. Because Grid Search cycles across each set of hyperparameters, it requires more iterations than other search methods. It assesses the performance of every combination of all provided hyperparameters as well as their values before calculating the best value for hyperparameters. Due to enormous number of hyperparameters included, processing is time-taking as well as quite expensive.

As a result, utilising Grid Search and cross-validation to find the best hyperparameters takes a long time. In our Machine Learning challenge, we enhanced the accuracy from 89.34 percent to 89.81 percent by utilising GridSearchCV.

The table below represents the accuracy using GridSearchCV tuning different parameters using:

Results Table:

Accuracy	**CV**	**Best hyper Parameter**

89.72%	5	'classifier_penalty':['l1', 'l2'], 'classifier_C':np.logspace(-4, 4, 20), 'classifier_solver':['liblinear']
89.72%	7	'classifier_penalty' : ['l1', 'l2'], 'classifier_C':np.logspace(-4, 4, 20), 'classifier_solver':['liblinear']
89.72%	11	'classifier_penalty':['l2'], 'classifier_C':np.logspace(-4, 4, 20), 'classifier_solver':['liblinear']
89.72%	5	'classifier_penalty' : ['l2'], 'classifier_C':np.logspace(-4, 4, 20), 'classifier_solver': ['liblinear','sag','saga','lbfgs']
89.8%	5	'classifier_penalty':['l2'], 'classifier_C':np.logspace(-10, 10, 20), 'classifier_solver':['liblinear']
89.8%	5	'classifier_penalty':['l1','l2'], 'classifier_C':np.logspace(-10, 10, 20), 'classifier_solver':['liblinear']
89.81%	5	'classifier_penalty':['l2'], 'classifier_C':np.logspace(-20,20, 40), 'classifier_solver':['liblinear']}
89.81%	5	'classifier_penalty':['l2'], 'classifier_C':np.logspace(0, 20, 40), 'classifier_solver':['liblinear']

Proposed System

We believe that there should be some room for improvement in model performance at this time. After looking at some of the characteristics, we've discovered that the present Logistic model fails to grasp the context of text features. To enhance this, we may use bi-grams, tri-grams, and stretchy patterns instead of unigram bag-of-words features, or even include word vectors from the Continuous Bag-of-Words (CBOW) or skip-gram models for word representations. Using models can remove characteristics in chunks of memory, including deep neural networks, and systems that recover sequential data, including machine learning algorithms, is another way to grasp the context. These methods are worth a go if you want to increase your text classification ability. we can also use two different datasets to handle the sentimental analysis and try different classification model to increase the accuracy and also build a deployment model with recommendation movies.

V. CONCLUSION

The Sentiment analysis has become a popular tool for determining how people feel regarding a service, product, a public issue or the location. When a text has to be sentimentally assessed, a machine learning technique is often applied. Each machine learning model's classification accuracy may be further refined by hyperparameter tweaking for obtaining greater accurateness than when model's default hyperparameter values are used.

Our findings reveal that:

- Logistic Regression provides the greatest accuracy both before as well as after hyperparameter adjustment, with the GridSearchCV scoring the highest at 89.81 percent.

Thus, we can show that comparatively we have improved our accuracy from the first approach using pipelines and hyper-parameter tuning which is visible in our results.

REFERENCES

[1] Liu, B.: Sentiment analysis and opinion mining. vol. 5 (05 2012)

[2] [2] Hu, M., & Liu, B. (2004). Mining and summarizing customer reviews. Proceedings of the 2004 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '04. doi:10.1145/1014052.1014073.

[3] [3] Determining the Sentiment of Opinions, Kim & Hovy (2004): DOI:10.3115/1220355.1220555

[4] [4] Recognizing Contextual Polarity in Phrase-Level Sentiment Analysis, (Wilson, Wiebe, & Hoffmann, 2005), DOI:10.3115/1220575.1220619

[5] [5] . Hatzivassiloglou, V; McKeown, K, 1997. "Predicting the Semantic Orientation of Adjectives." 35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics, July, pp. 174-181

[6] [6] Pang, B., Lee, L. 2004. "A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts." Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04), pp. 271-278, July.

[7] [7] Gruhl, D., R. Guha, Ravi Kumar, Jasmine Novak, and Andrew Tomkins. 2005. "The predictive power of online chatter." KDD '05: Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, pp. 78-87, August. doi: 10.1145/1081870.1081883.

[8] [8] Cambria, E; Schuller, B; Xia, Y; Havasi, C (2013). "New avenues in opinion mining and sentiment analysis". IEEE Intelligent Systems. 28 (2): 15–21. CiteSeerX 10.1.1.688.1384.

[9] [9] Ortony, Andrew; Clore, G; Collins, A (1988). The Cognitive Structure of Emotions (PDF). Cambridge Univ. Press.

[10] [10] Sankar, H., Subramaniyaswamy, V. 2017. "Investigating sentiment analysis using machine learning approach." International Conference on Intelligent Sustainable Systems (ICISS), IEEE, pp. 87-92, December 7-8.

[11] [11] Lee, Sung-Won ; Jiang, Guangbo ; Kong, Hai-Yan ; Liu, Chang(2020), " A difference of multimedia consumer's rating and review through sentiment analysis", Multimedia tools and applications

[12] [12] Shekar, B.H.; Dagnew, G. Grid Search-Based Hyperparameter Tuning and Classification of Microarray Cancer Data. In Proceedings of the 2019 Second International Conference on Advanced Computational and Communication Paradigms (ICACCP), Gangtok, India, 25–28 February 2019; pp. 1–8

[13] [13] Bergstra, J.; Bengio, Y. Random Search for Hyper-Parameter Optimization. J. Mach. Learn. Res. 2012, 13, 281–305.

[14] [14] Liashchynskyi, P.; Liashchynskyi, P. Grid Search, Random Search, Genetic Algorithm: A Big Comparison for NAS. arXiv 2019, arXiv:abs/1912.06059.

[15] [15] Villalobos-Arias, L.; Quesada-López, C.; Guevara-Coto, J.; Martínez, A.; Jenkins, M. Evaluating Hyper-Parameter Tuning Using Random Search in Support Vector Machines for Software Effort Estimation. In Proceedings of the PROMISE'20: 16th International Conference on Predictive Models and Data Analytics in Software Engineering, Virtual Event, Association for Computing Machinery, New York, NY, USA, 8–9 November 2020; pp. 31–40.

[16] [16] Andonie, R.; Florea, A.C. Weighted Random Search for CNN Hyperparameter Optimization. Int. J. Comput. Commun. Control 2020, 15. [CrossRef]

[17] [17] Probst, P.; Boulesteix, A.; Bischl, B. Tunability: Importance of Hyperparameters of Machine Learning Algorithms. J. Mach. Learn. Res. 2019, 20, 53:1–53:32.

[18] [18] Syarif, I.; Prugel-Bennett, A.; Wills, G. SVM parameter optimization using grid search and genetic algorithm to improve classification performance. Telecommun. Comput. Electron. Control 2016, 14, 1502.

[19] [19] Wicaksono, A.S.; Supianto, A.F. Hyper Parameter Optimization using Genetic Algorithm on Machine Learning Methods for Online News Popularity Prediction. Int. J. Adv. Comput. Sci. Appl. 2018, 9, 263–267

[20] [20] Martínez-Cámara, E.; Barroso, N.R.; Moya, A.R.; Fernández, J.A.; Romero, E.; Herrera, F. Deep Learning Hyper-parameter Tuning for Sentiment Analysis in Twitter based on Evolutionary Algorithms. In Proceedings of the 2019 Federated Conference on Computer Science and Information Systems (FedCSIS), Leipzig, Germany, 1–4 September 2019;

[21] [21] Alayba, A.M.; Palade, V.; England, M.; Iqbal, R. Improving Sentiment Analysis in Arabic Using Word Representation. In Proceedings of the 2018 IEEE 2nd International Workshop on Arabic and Derived Script Analysis and Recognition (ASAR), London, UK, 12–14 March 2018

[22] [22] Al-Twairesh, N.; Al-Negheimish, H. Surface and Deep Features Ensemble for Sentiment Analysis of Arabic Tweets. IEEE Access 2019, 7, 84122–84131.

[23] [23] Duwairi, R.; Qarqaz, I. Arabic Sentiment Analysis Using Supervised Classification. In Proceedings of the 2014 International Conference on Future Internet of Things and Cloud, Barcelona, Spain, 27–29 August 2014

[24] [24] Duwairi, R.; El-Orfali, M. A study of the effects of preprocessing strategies on sentiment analysis for Arabic text. J. Inf. Sci. 2014, 40, 501–513.

[25] [25] Štrimaitis, R.; Stefanoviˇc, P.; Ramanauskaite, S.; Slotkien ˙ e, A. Financial Context News Sentiment Analysis for the Lithuanian ˙ Language. Appl. Sci. 2021, 11, 4443.

[26] [26] Siji George, C.G.; Sumathi B. Genetic Algorithm Based Hybrid Model Of Convolutional Neural Network And Random Forest Classifier For Sentiment Classification. Turk. J. Comput. Math. Educ. 2021, 12, 3216–3223

[27] [27] Pouransari, H.; Ghili, S. Deep learning for sentiment analysis of movie reviews. CS224N Proj. 2014, 1–8

[28] [28] Sayed, A.A.; Elgeldawi, E.; Zaki, A.M.; Galal, A.R. Sentiment Analysis for Arabic Reviews using Machine Learning Classification Algorithms. In Proceedings of the 2020 International Conference on Innovative Trends in Communication and Computer Engineering (ITCE), Aswan, Egypt, 8–9 February 2020; pp. 56–63

[29] [29] Boudad, N.; Faizi, R.; Oulad Haj Thami, R.; Chiheb, R. Sentiment analysis in Arabic: A review of the literature. Ain Shams Eng. J. 2018, 9, 2479–2490.

[30] [30] Nguyen, Heidi; Veluchamy, Aravind; Diop, Mamadou; and Iqbal, Rashed (2018) "Comparative Study of Sentiment Analysis with Product Reviews Using Machine Learning and Lexicon-Based Approaches," SMU Data Science Review: Vol. 1 : No. 4 , Article 7

[31] [31] Yao, Yao; Angelov, Ivelin; Rasmus-Vorrath,Jack; Lee, Mooyoung; and Engels, Daniel W. (2018) "Yelp's Review Filtering Algorithm," SMU Data Science Review: Vol. 1 : No. 3 , Article 3.

[32] [32] Ramya, V.Uma; "Sentiment Analysis of Movie Review using Machine Learning Techniques" International Journal of Engineering & Technology, 7 (2.7) (2018) 676-681.

[33] [33] Somya Dwivedi, Harsh Patel and Shweta Sharma; "Movie Reviews Classification Using Sentiment Analysis", Indian Journal of Science and Technology, Vol 12(41), November 2019

[34] [34] Swathi H., S. S. Aravinth, V. Nivethitha, T. Saranya, R. Nivethanandhini; "Sentiment Analysis of Movie Review using data Analytics Techniques", MAR 2019, IRE Journals, Volume 2 Issue 9.

[35] [35] Baid, Palak & Gupta, Apoorva & Chaplot, Neelam. (2017). "Sentiment Analysis of Movie Reviews Using Machine Learning Techniques". International Journal of ComputerApplications. 179. 45-49. 10.5120/ijca2017916005.

[36] [36] Atif Khan, Muhammad Adnan Gul, M. Irfan Uddin, Syed Atif Ali Shah, Shafiq Ahmad, Muhammad Dzulqarnain Al Firdausi, Mazen Zaindin, (2020) "Summarizing Online Movie Reviews: A Machine Learning Approach to Big Data Analytics", Scientific Programming, vol. 2020, Article ID 5812715

[37] [37] Kang, Hanhoon ; Yoo, Seong Joon ; Han, Dongil (2012) "Senti-lexicon and improved Naïve Bayes algorithms for sentiment analysis of restaurant reviews", Elsevier Ltd Expert systems with applications, Vol.39 (5), p.6000-6010

[38] [38] Khaleghi, Ryan; Cannon, Kevin; and Srinivas, Raghuram (2018) "A Comparative Evaluation of Recommender Systems for Hotel Reviews," SMU Data Science Review: Vol. 1 : No. 4 , Article 1.

[39] [39] Bird, Steven & Klein, Ewan & Loper, Edward. (2009). Natural Language Processing with Python

[40] [40] Lorenzo, P.R.; Nalepa, J.; Kawulok, M.; Ramos, L.; Ranilla, J. Particle swarm optimization for hyper-parameter selection in deep neural networks. In Proceedings of the Genetic and Evolutionary Computation Conference, Berlin, Germany, 15–19 July 2017.

[41] [41]] Snyder, Benjamin; Barzilay, Regina (2007). "Multiple Aspect Ranking using the Good Grief Algorithm". Proceedings of the Joint Human Language Technology/North American Chapter of the ACL Conference

[42] [42] Pang, Bo; Lee, Lillian; Vaithyanathan, Shivakumar (2002). "Thumbs up? Sentiment Classification using Machine Learning Techniques". Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP).

[43] [43] pandas: a Foundational Python Library for Data Analysis and Statistics, Wes Mckinney

[44] [44] Aspect Based Sentiment Analysis of Movie Reviews: Finding the Polarity Directing Aspects, Viraj Parkhe, Bhaskar Biswas.

[45] [45] Aspect-based Sentiment Analysis of Movie Reviews, Samuel Onalaja, Eric Romero, Bo Yun, Faizan Javed

[46] [46] Hyperparameter Tuning for Machine Learning Algorithms Used for Arabic Sentiment Analysis, Enas Elgeldawi, Awny Sayed, Ahmed R. Galal and Alaa M. Zaki.