RESEARCH ARTICLE OPEN ACCESS

Exploration and Assessment of Multiple Crowd Sourced Feedbacks

Gaddam Akhil Reddy[1], Kothapally Nithesh Reddy[2],
Dr. Vijayalakshmi Kakulapati[3]

1(Department of Information Technology, Sreenidhi Institute of Science and Technology, Hyderabad, Telangana, India
Email: akhilreddyg21pa@gmail.com)
2(Department of Information Technology, Sreenidhi Institute of Science and Technology, Hyderabad, Telangana, India
Email: nitheshreddy.k939@gmail.com)
3(Department of Information Technology, Sreenidhi Institute of Science and Technology, Hyderabad, Telangana, India
Email: vijayalakshmik@sreenidhi.edu.in)

Abstract:

The majority of moviegoers use websites like IMDb, Amazon, and Yelp to research and select their next viewing or purchase. Currently, moviegoers make their choices based on IMDb or Amazon ratings and reviews of films they've seen. As outlined in this article, a better technique is proposed: the subjective content of movie critics' movie scores and critiques may be evaluated and projected onto a picture, resulting in an emotional map. The second step is to seek films with emotion maps that have certain emotion map patterns that appeal to you. Concerning decision-making processes, sentiment analysis has long been a point of concern for marketing, business, and management departments.

When we conduct sentiment analysis, we're looking for emotions and points of view in a piece of writing. It identifies and validates a person's feelings about a certain piece of material for the reader. Product reviews, blogs, status updates, and tweets, among other things, are all examples of sentiment data seen on social media sites. This massively generated data may be used for sentiment analysis to express the opinions of the general public on products. This article presents a highly accurate sentiment analysis algorithm for Amazon, IMDB, and Yelp product, movie, and restaurant reviews. These reviews are classified into two categories: positive and negative, and we employ a variety of classifiers to determine which category is more favorable. The best classifier is chosen based on how accurate it is.

Keywords — **Opinion Mining, SVM, Logistic Regression, Lexicon, Pipeline, TF-IDF Vectorizer, Ensemble Learning.**

I. INTRODUCTION

Natural Language Processing (NLP), which deals with the processing and analysis of enormous amounts of natural language data, includes text classification as a significant issue. The unstructured nature of the text makes it difficult to glean valuable information from it, despite its abundance. As a result of consumer contributions, text data may immediately influence company decisions by supplying quick input. Consequently, businesses devote significant resources to gathering, collecting, storing, and analyzing this type of data. Categorizing text based on its content is known as text classification. This approach can be aided by machine learning, which uses pre-labeled samples to classify previously undiscovered observations. Since machine learning algorithms find the correlation between entering text fragments and a label, this is a realistic goal to attain. This article's goal is to show how machine learning decisions may affect the outcome of a classification problem using many data sources. This paper will explain how.

Analysis of how people feel about specific topics and events often called sentiment analysis or

opinion mining (OM), is a technique. Subjective textual content such as views, emotions, and subjectivity are treated using computational means. When it comes to buying a purchase, customers these days pay close attention to what other people have to say about it online and in reviews. This is known as the customer's viewpoint or behavior. In the same way, a movie review follows the same pattern. A text's sentiment perspective is automatically classified using sentiment analysis (positive or negative). It's handy for classifying online products as "recommended" or "not recommended," depending on whether they are satisfied or not satisfied.

The positivity or negativity of a review or words can be determined using one of two primary OM techniques. The first strategy is based on supervised, unsupervised, and semi-supervised learning, which is prevalent in sentiment analysis (SA). The dataset is labeled for supervised learning to provide a coherent and comprehensible result. For unsupervised learning, there is no need for labeled data, in contrast to supervised learning. Unlabeled data processing necessitates the use of clustering algorithms. The second approach involves using a dictionary or existing lexicon of words, idioms, or phrases that are either negative or positive. These methods were created in response to actual market demands. This study's goal is to see how different supervised learning algorithms perform on different kinds of labeled data. Both binary and multi-class classifications are popular methods for classifying emotions. Any feedback document or dataset is categorized as either positive or negative using binary statistical classification (SC). However, each document in multi-class SC can be classified into more than two classes, with the degree of emotion ranging from strongly positive to neutral to strongly negative.

Three levels of SA classification are used: sentence (SL), aspect (AL), and document (DL). The AL is a classification system for sentiments regarding a product's different attributes or components. It's difficult to determine whether each remark in the SL is neutral, positive, or suggests a certain perspective. The main goal of the DL is to figure out if the overall tone of a document is positive or unfavorable. Individuals' acceptance and rejection of new information cannot be determined with sufficient precision using the SL and DL investigations. This study focuses on document-level sentiment analysis. What's left of the paper is organized as follows: Section 2 presents a survey of the SA research literature. Using Section 3, we can see how the different datasets differ. Section 4 explains the Sentiment Analysis method. Section 5 examines the results of the proposed technique on several datasets. Conclusions and research constraints are discussed in Section 6.

II. LITERATURE SURVEY

According to Tripathy et al., several machine learning approaches were used to classify movie reviews, including Naive Bayes (NB), support vector machines (SVM), maximum entropy (ME), and stochastic gradient descent (SGD). These algorithms are tested on the IMDB dataset using various n-gram technique combinations, such as bigram and trigram, unigram and bigram, bigram trigram, and unigram. Support vector machines had the highest accuracy when using trigram, bigram, and unigram as feature extractors, according to their research. Cornell film reviews, Amazon product evaluations, and Stanford film reviews were utilized by Deng et al. to extract features using SVM and the Importance of a Term in a Document (ITD).

In addition to these two datasets, their approach surpasses Best Matching (BM25), while the difference is insignificant on the small Cornell movie review dataset. By integrating NB with Hadoop, Liu et al. created a basic framework for Sentiment Analysis on the Cornell film review dataset. According to the results, the NB classifier can handle large datasets with ease. The resulting precision is below the desired level of 82%. There are still lexicon-based sentiment analysis alternatives to machine learning-based sentiment analysis. Sentiment analysis in combination with a vocabulary or corpus has been utilized in several research projects.

Recent research has focused on sentiment analysis at the textual level, which is considered a combination of words. This study used Wilson and colleagues' approach to determine whether or not anything is neutral or polemic, then disambiguate the polarity of polemic expressions (Wilson, Wiebe, & Hoffmann, 2005). To better understand the context of a PoS element, they tagged phrases rather than words in the document. This supplied context knowledge for rule-based categorization. There is no longer any need for human classification thanks to the Classifier Levels in Information Extraction (Agarwal et al.), which improved WordNet's capacity to score an overwhelming majority of terms in the input automatically (Agarwal, Biadsy, & Mckeown, 2009). Sentences in a document are mined for constituent N-grams based on previous evaluations, and context is provided.

A wide range of classification methods is included in machine learning, including Support Vector Machines (SVM), Artificial Neural Networks (ANN), Logistic Regression (LR), and Decision Trees. Product reviews can be classified using these methods. Feature vectors may be created by using the presence of each letter, the number of appearances of every character, and textual sentences that include negation as features, as proven by this work (Mejova Y. et al., 2009). He also showed how to use unigram and bigram techniques to construct effective feature vectors for the analysis of the sentiment.

Using dependent features with the Naive Bayes classifier, another article (Domingo P., 1997) found that it performed well. An innovative Bayesian classifier was developed in this study (Niu Z. et al., 2012). The model incorporated a wide range of useful approaches for selecting features, calculating weights, and classifying objects. According to the study (Barbosa L., & Feng J., 2010), automated sentiment analysis was used to classify tweets in two steps. To begin, tweets were divided into two categories: subjective and objective. In the second phase, subjective tweets were classified as positive or negative.

According to the study (Celikyilmaz A. et al., 2010), words may be categorized depending on how they sound. Loud tweets can be normalized with this method. While many words have a similar sound, they don't always have the same meaning. As a result, methods for settling this kind of disagreement were developed. With the aforementioned method, tokens that have the same sound are grouped and given the same number of common tokens. Wang, Yu, and Ren (2011) suggested the development of an analysis model for Twitter sentiments.

A rapid response is taken to improve the chance of an effect if a user recommendation is found in a tweet in this study. Pak A. and Paroubek P. (2010) presented a technique for sentiment analysis using automated Twitter tweets in their study. For sentiment analysis, a Naive Bayes classifier was created, which considered the emotional impact of the text. To find out what people think about movies, the news, and other issues, researchers turned to Twitter. To compile the study's findings, the authors used data from many publicly available databases, including those found on sites like IMDB (Internet Movie Database).

III. PROPOSED METHODOLOGY

A. Data collection:

To begin, we extract the Yelp reviews or dataset for sentiment analysis from the "yelp labelled.txt" file. The data in this case consists of text that we refer to as a food review. This dataset contains 1,000 phrases labelled with a sentiment score of 0 or 1, where a score of 0 indicates a negative sentiment and a score of 1 indicates a positive sentiment. We proceed with another dataset for the analysis of Amazon reviews from the "amazon cells labelled.txt" file. Even here, we categorise each review as either a '0' or a '1'. Additionally, this dataset contains 1,000 phrases. Finally, the dataset for the imdb analysis is extracted from the file "imdb labelled.txt." There are a total of 748 phrases in this section, each of which contains the sentiments '0' or '1'.

```
data_yelp = pd.read_csv('yelp_labelled.txt', sep
='\t', header = None)

data_amazon =
pd.read_csv('amazon_cells_labelled.txt', sep ='\t',
header = None)

data_imdb = pd.read_csv('imdb_labelled.txt', sep
='\t', header = None)
```

Architecture:

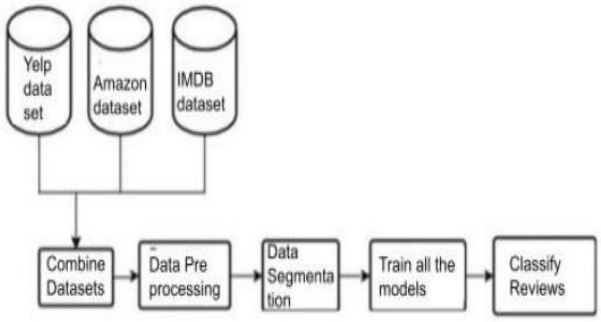

Fig 1 Architecture

B. Integration of all datasets:

Combining the three Data Frames

```
data = data_yelp.append ([data amazon, data
imdb], ignore index = True)
```

Calculating the dataset's total size

After combining the datasets, the total size was determined to be 2748, and the dataset consists of two attributes, one for the review and another for the sentiment.

```
data.shape

(2748,2)
```

C. Text Preprocessing:

Data cleaning enhances insights by removing all capitalization and punctuation from sentence words. In addition, it reduces the computational strain on models by eliminating words that aren't necessary. Text cleaning depends on the circumstances. Lemmatization is a text-cleansing method that combines word forms into a single phrase for further analysis. Because there may be a link between word inflection and sentiment in the reviews, we won't use it here. Stop words convey very little. These words don't help a model figure out what an input means, therefore they may be removed with no ill effects.

Text cleaning process:

1. Separate sentences using whitespace to tokenize them.
2. Lowercase all of the words that were previously in caps.
3. Tokens such as "%" and "()*+,./:;=>?@ [] |) should be removed.
4. Remove tokens that aren't alphanumeric or alphabetic.
5. Discard tokens that have stop words in them.
6. Reassemble the remaining tokens into sentences.

Any six of the aforementioned procedures are performed by the function we created for text preprocessing entitled "text data cleaning," allowing us to remove all unnecessary data and analyze all reviews gathered across three datasets.

```
import spacy
nlp = spacy.load('en_core_web_sm')
def text_data_cleaning(sentence):
    doc=nlp(sentence)
    tokens = []
    for token in doc:
        if token.lemma_ != "-PRON-":
            temp = token.lemma_.lower().strip()
        else:
            temp = token.lower_
        tokens.append(temp)
    cleaned_tokens = []
    for token in tokens:
        if token not in stopwords and token not in
            punct:
            cleaned_tokens.append(token)
```

return cleaned_tokens

Using TF-IDF Vector as a Tool:

Determine the relative relevance of words using the TF-IDF vectors (Term Frequency Inverse Document Frequency). There is a formula that takes into account the number of times a term appears in a text as well as how many other documents include the same phrase. It's safe to assume that any phrase with a very high TF-IDF value is essential to the document's overall meaning. The first part of the TF-IDF formula is:

TF = (Number of Specific Terms) / (Number of All Terms)

IDF = log ([# Total Number of Documents]/[# Total Number of Documents containing SpecificTerm])

The formula then multiplies these components together: TF-IDF equals TF IDF.

It's possible to construct TF-IDF vectors for letters, words, or n-grams. As the number of words in each token increases, so does the number of N-grams. The lower and upper boundaries of the n-grams are specified using a TfidfVectorizer() function argument called the n-gram range.

import sklearn.feature extraction.text from sklearn.feature extraction.text TfidfVectorizer

We're going to pass our function "def_text data_cleaning" to the TF-IDF vector for tokenization in this case.

tfidf = TfidfVectorizer(tokenizer = text_data_cleaning)

Utilizing a Pipeline:

Tabular data is often processed using machine learning techniques. The preprocessing of this data before running our machine learning algorithm may be beneficial. Data processing stages can be chained together using a pipeline. Most machine learning tasks call for transforming the raw dataset several times before they can be completed. Because of the pipeline, we can run many transformations and evaluations at the same time, which drastically decreases the computing burden.

from sklearn.pipeline import Pipeline

D. Data Segmentation

The textual data has been cleaned up. Now, we divide the dataset into a training set for the classifiers to learn from and a testing set for prediction and performance calculation. We divide the data in 80% for training purposes and 20% for testing purposes.

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 0)

x_train.shape, x_test.shape, y_train.shape, y_test.shape

E. Developing Multiple Classifiers and Training them

Machine Learning Classifiers can make predictions as well as classifications. They are nothing more than classification algorithms for input data created by machine learning algorithms. When it comes to classification, the process entails making predictions about the category to which a set of data points will belong. To figure out if a review is good or bad, several supervised machine learning algorithms are tallied up. To train the classifiers, we will build them one at a time and then train them all separately. Many machine learning models will be trained on the dataset once it has been processed and examined to compare their efficiency and choose the best one.

1. Support Vector Machine

To understand how to find the best hyperplane between two classes, we use the Support Vector Machine (SVM). Each piece of information is stored in n-dimensional space, where n is the total number of features in the dataset. The hyperplane separation of the input data classes leads to classification. The hyperplane with the greatest distance between the nearest data points of both

classes is used (support vectors). The margin is the distance between the hyperplane and the two classes' closest points. There is less chance of misclassification if the margin is larger. The SVM classifies classes as exactly as possible before attempting to maximize the margin. This behavior may lead to incorrect categorization or the need for the use of various kernels in circumstances when class separation is not linear. In the case of non-linear class separations, a kernel function raises the input space of the issue from the current low level.

```
from sklearn.svm import LinearSVC
svm = Pipeline([("tfidf",TfidfVectorizer()) ,
("classifier",LinearSVC())])

Fitting the the dataset into the model

svm.fit(x_train, y_train)
```

2. Random Forest

Ensemble learning techniques such as Random Forest (RF) models use a large number of decision trees to create predictions for classification based on the class model of each tree's findings. Overfitting of decision trees may be compensated for by using Random Forest, which lowers the overall variance. Branches indicate event outcomes, leaves represent class labels, and each internal node represents a predictor event. The principles used to classify the outcome are defined by the journey they take from root to leaf. With the bagging ensemble algorithm, each of the ensemble's decision trees gets a random sample of data from the training set and the features.

```
from sklearn.ensemble import
RandomForestClassifier
rf = Pipeline([("tfidf",TfidfVectorizer()) ,
("classifier",RandomForestClassifier())])

Fitting the the dataset into the model

rf.fit(x_train, y_train)
```

3. Logistic Regression

To establish the connection between a categorical dependent variable and independent factors, logistic regression (LR) estimates probability by employing the log-sigmoid function (a function with an S-shaped curve). It is close to the Logit function, which returns the logarithm of the likelihood that an event will take place, that the logistic function is used to approximate that. The Logit function's value increases exponentially as probability estimates near one (1). The Logit function's value decreases as probability estimations go towards zero (0). Optimizing the Log-Likelihood function is done using Least Squares (LR). You may use the Likelihood function in statistical models with unknown parameter values to see how well the data fits the model's predictions. Maximal likelihood function values are those for which the likelihood of finding data is highest. The Log-Likelihood function is widely employed in the optimization process because of its simple form while trying to identify the peak.

```
from sklearn.linear_model import
LogisticRegression
lm = Pipeline([("tfidf",TfidfVectorizer()) ,
("classifier",LogisticRegression(max_iter=800))])

Fitting the the dataset into the model

lm.fit(x_train,y_train)
```

4. Decision Tree

A decision tree produces a set of rules for classifying data based on a set of qualities and their associated classes. Because of its simplicity, the Decision Tree can handle numerical as well as categorical data with minimum data preprocessing. Because of their complexity and difficulty in generalization, decision trees can produce unstable results because little changes in the data might lead to the creation of a completely new tree.

```
from sklearn.tree import DecisionTreeClassifier
dst= Pipeline([("tfidf",TfidfVectorizer()) ,
("classifier",DecisionTreeClassifier(max_depth =
1000, random_state = 0,criterion = 'entropy'))])
```

Fitting the the dataset into the model

```
dst.fit(x_train,y_train)
```

5. Naive Bayes

The Bayes' Theorem is the foundation of the Naive Bayes (NB) classification algorithm. It makes use of supervised machine learning. An NB classifier assumes that class characteristics are unconnected and independent. The posterior probability of a class given a predictor P (c | x) is calculated using three terms: the prior probability of the class P (c); the prior probability of the P (x) is a predictor, and P (x | c) is the probability of a predictor given a class (likelihood). The posterior probability is determined for each class, and the prediction is made based on the class with the highest result.

```
from sklearn.naive_bayes import MultinomialNB
munb = Pipeline([("tfidf",TfidfVectorizer()) ,
("classifier",MultinomialNB())])
```

Fitting the the dataset into the model

```
munb.fit(x_train,y_train)
```

6. K Nearest Neighbors

The Bayes' Theorem is the foundation of the Naive Bayes (NB) classification algorithm. It makes use of supervised machine learning. An NB classifier assumes that class characteristics are unconnected and independent. The posterior probability of a class given a predictor P (c | x) is calculated using three terms: the prior probability of the class P (c); the prior probability of the P (x) is a predictor, and P (x | c) is the probability of a predictor given a class (likelihood). The posterior probability is determined for each class, and the prediction is made based on the class with the highest result.

```
from sklearn.neighbors import
KNeighborsClassifier
knn = Pipeline([("tfidf",TfidfVectorizer()) ,
("classifier",KNeighborsClassifier(n_neighbors=
50))])
```

Fitting the the dataset into the model

```
knn.fit(x_train,y_train)
```

F. Classification of the Review

After training the models, it's time to use these machine learning algorithms to predict the target variable based on the input variable. We'll begin with svm and continue in this manner until we reach knn.

```
1. For Support Vector Machine:
y_pred_svm = svm.predict(x_test)

2. For Random Forest:
y_pred_rf = rf.predict(x_test)

3. For Logistic Regression:
y_pred_lm = lm.predict(x_test)

4. For Decision Tree:
y_pred_dst = dst.predict(x_test)

5. For Naive Bayes:
y_pred_munb = munb.predict(x_test)

6. For K Nearest Neighbors:
y_pred_knn = knn.predict(x_test)
```

Below is the script which compares accuracy scores of all the machine learning models

```
alg = ["SVM", "RF", "LR", "DST", "MNB",
"KNN" ]
for i in range(6):
   print(f" {alg[i]} -> {round(l1[i]*100,2)} %")
```

IV. RESULT

We began this research with a total of three datasets. Later, we merged them all into one. The initial dataset had 1,000 phrases extracted from Yelp reviews. Amazon's subsequent dataset also

included a total of 1000 reviews. Finally, the last dataset included 748 movie reviews from IMDb. After merging them all, there were a total of 2748 reviews, of which we used 80% for training and 20% for testing. Numerous error analyses have been conducted, including the omission of stop words, extraneous letters, and punctuation. Following that, all Machine Learning classifiers were trained and predictions were made. Both Logistic Regression and Support Vector Machine have dominated the battle for the highest accuracy, while the Decision Tree has come last. Future work on the model may focus on innovative techniques such as deep learning and the inclusion of contextual information into the feature space. The following sections provide a comparative analysis and summary of the paper.

```
alg = ["SVM", "RF", "LR", "DT", "NB", "KNN" ]

for i in range(6):
    print(f" {alg[i]} -> {round(l1[i]*100,2)} %")
```

```
SVM -> 82.36 %
RF -> 80.91 %
LR -> 82.55 %
DT -> 69.45 %
NB -> 81.82 %
KNN -> 74.91 %
```

Fig 2 Comparison of Multiple Classifiers

For Logistic Regression:

Confusion Matrix:

```
print(confusion_matrix(y_test, y_pred_lm))
```

```
[[231  48]
 [ 48 223]]
```

Fig 3 Confusion Matrix for Logistic Regression

Classification Report:

```
print(classification_report(y_test, y_pred_lm))
```

	precision	recall	f1-score	support
0	0.83	0.83	0.83	279
1	0.82	0.82	0.82	271
accuracy			0.83	550
macro avg	0.83	0.83	0.83	550
weighted avg	0.83	0.83	0.83	550

Fig 4 Classification Report for Logistic Regression

For SVM:

Confusion Matrix:

```
print(confusion_matrix(y_test, y_pred_svm))
```

```
[[228  51]
 [ 46 225]]
```

Fig 5 Confusion Matrix for SVM

Classification Report:

```
print(classification_report(y_test, y_pred_svm))
```

	precision	recall	f1-score	support
0	0.83	0.82	0.82	279
1	0.82	0.83	0.82	271
accuracy			0.82	550
macro avg	0.82	0.82	0.82	550
weighted avg	0.82	0.82	0.82	550

Fig 6 Classification Report for SVM

V. CONCLUSION

We can deduce that IMDB, Amazon, and Yelp reviews can be evaluated using machine learning-based techniques. Given the prevalence of slang words, hashtags, and other slang-based expressions, sentiment analysis can be difficult. To classify the reviews, a pre-processed data set is fed into various machine learning-based classifiers. Naive Bayes, SVM, Logistic Regression, Random Forest, K Nearest Neighbours, and the Decision Tree are six different supervised machine learning algorithms used in this study. The number of reviews on IMDB and Amazon differ. Several algorithms' results

showed substantial differences in performance. The overall accuracy was in the 65 to 85% range.

There was a significant improvement in accuracy using Logistic Regression and Support Vector Machines, according to our findings. The highest accuracy of Logistic Regression in estimating the testing dataset's sentiment is 82.55 percent. With a score of 82.36 percent, Support Vector Machine comes in second. The Decision Tree had the lowest accuracy, scoring just 69.46%. Both Logistic Regression and Support Vector Machine demonstrated substantial accuracy when compared to other techniques. Sentiment analysis has several problems that might be exploited to further this study in the future. It is planned that future research would focus on the detection and application of negative sentiment analysis to a wider variety of domains such as Yelp and Facebook and cross-domain. To identify sentiment in mixed source datasets, it's a good idea to use a Naive Bayes model using Count Vectorization because it's simple to construct and fits well in real-world data. Using the Logistic Regression classifier in the future may allow us to stop the spread of false information about some sensitive issues, or perhaps prevent acts of terrorism from occurring.

ACKNOWLEDGMENT

We would like to express our deepest gratitude to our professor, Dr. Vijayalakshmi Kakulapati, for guiding us through the process of writing this research paper.

REFERENCES

1. *Boya Yu, Jiaxu Zhou, Yi Zhang, Yunong Cao, "Identifying Restaurant Features via Sentiment Analysis on Yelp Reviews", arxiv,20 Sep 2017*

2. *Filieri, R.; Raguseo, E.; Vitari, C. When are extreme ratings more helpful? Empirical evidence on the moderating effects of review characteristics and product type. Comput. Hum. Behav. 2018, 88, 134–142.*

3. *Huang, A.H.; Chen, K.; Yen, D.C.; Tran, T.P. A study of factors that contribute to online review helpfulness.Comput. Hum. Behav. 2015, 48, 17–27.*

4. *Pantelidis, I.S. Electronic meal experience: A content analysis of online restaurant comments. Cornell Hosp. Q.2010, 51, 483–491.*

5. *P. Sasikala, L.M. I Sheela,"Sentiment Analysis of Online Food Reviews using Customer Ratings", International Journal of Pure and Applied Mathematics, Volume 119 No. 15, 3509-3514,2018.*

6. *Yan, X.; Wang, J.; Chau, M. Customer revisit intention to restaurants: Evidence from online reviews. Inf. Syst Front. 2015, 17, 645–657*

7. *.Tri Doan, Jugal Kalita, ''Sentiment Analysis of Restaurant Reviews on Yelp with Incremental Learning", 2016 15th IEEE International Conference on Machine Learning and Applications.*

8. *Gao, B.; Hu, N.; Bose, I. Follow the herd or be myself? An analysis of consistency in behavior of reviewers and the helpfulness of their reviews. Decis. Support Syst. 2017, 95, 1–11*
.

9. *Chua, A.Y.; Banerjee, S. Helpfulness of user-generated reviews as a function of review sentiment, product type and information quality. Comput. Hum. Behav. 2016, 54, 547–554.*

10. *Ma, Y.; Xiang, Z.; Du, Q.; Fan, W. Effects of user-provided photos on hotel review helpfulness: An analytical approach with deep learning. Int. J. Hosp. Manag. 2018, 71, 120–131.*

11. *Nam, S.; Ha, C.; Lee, H. Redesigning In-Flight Service with Service Blueprint Based on Text Analysis. Sustainability 2018, 10, 4492.*

12. *L. Garcia-Moya, H. Anaya-Sanchez, and R. Berlanga-Llavori, ''Retrieving product features and opinions from customer reviews,'' IEEE Intell. Syst.,vol. 28, no. 3, pp. 19–27, May 2013.*

13. *R. Jing, Y. Yu, and Z. Lin, ''How service-related factors affect the survival of B2T providers: A sentiment analysis approach,'' J.Organizational Comput. Electron. Commerce, vol. 25, no. 3, pp. 316–336, Jul. 2015.*

14. *Ł. Augustyniak, P. Szymański, T. Kajdanowicz, and W. Tuligłowicz, ''Comprehensive study on lexicon-based ensemble classification sentiment analysis,'' Entropy, vol. 18, no. 1, p. 4, Dec. 2015.*

15. *D.K.Kirange and 2Dr. Ratnadeep R. Deshmukh, "Aspect and Emotion Classification of Restaurant and Laptop Reviews Using Svm", International Journal of Current Research Volume. 8, Ise-03, pp. 28352-28356, March, 2016.*

16. *Wang, X.; Tang, L.R.; Kim, E. More than words: Do emotional content and linguistic style matching matter on restaurant review helpfulness? Int. J. Hosp. Manag. 2019, 77, 438–447*

17. *Hu, Y.H.; Chen, K.; Lee, P.J. The effect of user-controllable filters on the prediction of online hotel reviews.Inf. Manag. 2017, 54, 728–744.*

18. *A.-M. Popescu and O. Etzioni, ''Extracting product features and opinions from reviews,'' in Proc. Conf. Human Lang. Technol. Empirical Methods Natural Lang. Process. (HLT), Stroudsburg, PA, USA: Association for Computational Linguistics, 2005, pp. 339–346, doi: 10.3115/1220575.1220618.*