Journal of Computing & Biomedical Informatics
ISSN: 2710 - 1606

Volume 05 Issue 01
2023

Research Article
https://doi.org/10.56979/501/2023

TF-IDF Feature Extraction based Sarcasm Detection on Social Media

Hafsa Ahmad[1*], Wasif Akbar[1], Naeem Aslam[1], Azka Ahmed[1] and Mohsin Khurshid[2]

[1]Department of Computer Science, NFC Institute of Engineering and Technology, Multan, 60000, Pakistan
[2]Department of Computer Science, The Islamia University of Bahawalpur, Pakistan
[*]Corresponding Author: Hafsa Ahmad. Email: hafsa.ahmed717@gmail.com

Received: December 11, 2022 **Accepted:** May 12, 2023 **Published:** June 05, 2023

Abstract: The popularity of social media has significantly changed how people live their everyday. Twitter is being the most widely used platform for information sharing and emotional expression. However, the excess of false and controversial information has raised concerns as social media usage increases. One problem in particular is the prevalence of sarcastic text, which can be challenging to identify manually. In proposed research, an innovative approach is developed that makes use of cutting-edge machine learning methodologies for sarcasm detection on social media. The proposed approach includes four machine learning algorithms for classification and the TF-IDF (Term Frequency- Inverse Document Frequency) to extract features from the text. Different evaluation metrics like recall, precision, F1, and accuracy scores were used to evaluate the highest values of the various algorithms.

Keywords: Sarcasm detection, Social media, Twitter, irony, Machine learning.

1. Introduction

The People have been known to utilize social media to propagate false information, spread rumors, and make meaningless interpretations of events [3]. People start this problem by posting material that can mislead people or lead to misunderstandings regarding specific topics. Sarcasm is the harsh use of language acquisition, frequently done in good fun, to make fun of someone or something. Disagreement may be used in sarcasm, but it is not always ironic [1].

In generally, people utilize sarcasm in their daily interactions to make jokes and attempt to be funny. Sarcasm is also used to ridicule and make remarks about certain people and their ideas [2]. Sarcasm is a pervasive form of communication on social media platforms and can be challenging to detect and interpret accurately. Given the volume of social data being generated, identifying and understanding sarcastic remarks and comments is becoming increasingly important for businesses, researchers, and individuals alike [4].

Detecting sarcasm on social media can be a challenging task because the lack of non-verbal cues, such as facial expressions and tone of voice, can make it difficult to distinguish between sincere comments and sarcastic ones. However, researchers have developed methods for detecting sarcasm in social media text. For one, sarcasm is often context-dependent and reliant on a person's tone of voice [6]and facial expressions [7], which are difficult to capture in written text. Additionally, sarcasm can take on many different forms, from ironic statements to satirical humor, which can make it challenging to develop a universal approach for detecting it.

Recent studies have shown that machine learning algorithms can be effective in detecting sarcasm in social media data [11, 12, 13, 14]. These algorithms use natural language processing techniques [8] to analyze the text of social media posts and comments and identify patterns and cues that suggest sarcasm.

The main contribution is to develop and evaluate different machine learning models for sarcasm detection specifically, the project employs Stochastic Gradient Descent, K-nearest neighbor, random forest, and Naive Bayes classifiers to identify sarcastic comments from a large corpus of text data. Additionally, this resaerch provides insights into the importance of feature engineering and parameter tuning in improving the performance of these models. The main objective of this research article is to provide the maximum accuracy rate by using machine learning models.

2. Materials and Methods

This research aims to classify the tweets whether it is sarcastic ironic, regular, or figurative. Feature engineering and Feature selection techniques are used to extract the features from each tweet for the classification of sarcasm. Various machine learning algorithms like Naive Bays, Random Forest, KNN, and SGD have been applied for the experiment. The overall methodology of the proposed framework is described below.

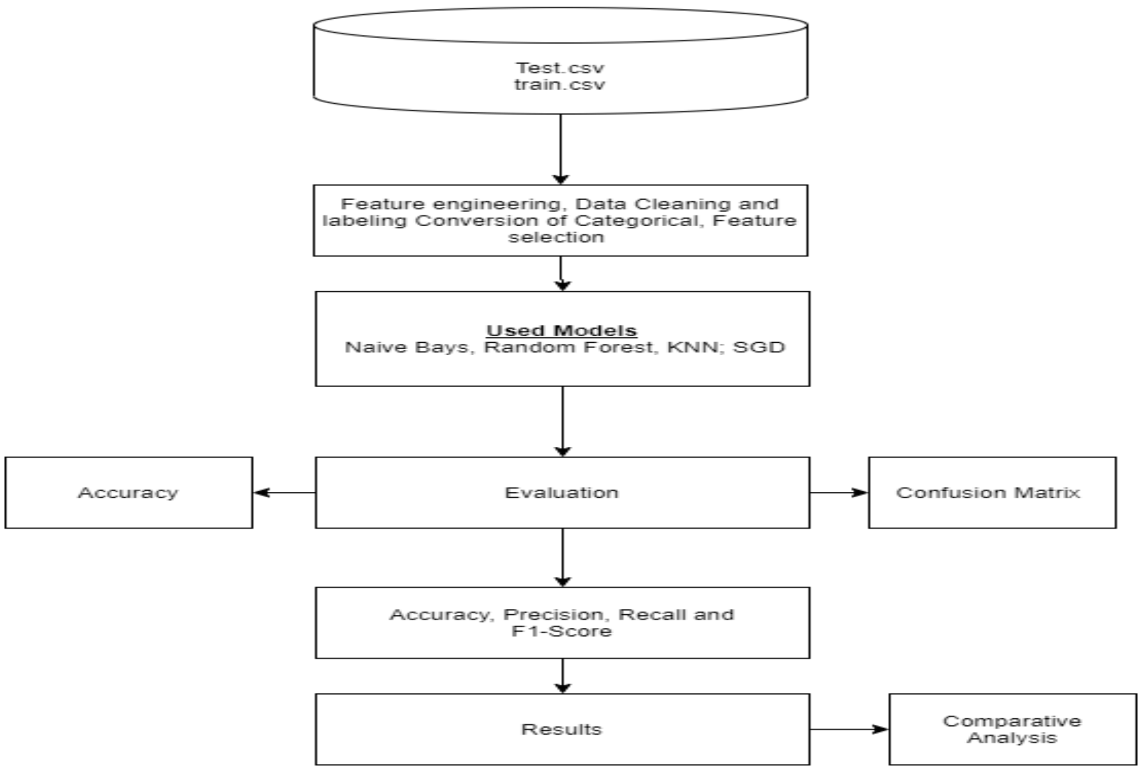

Test.csv
train.csv
Feature engineering, Data Cleaning and labeling Conversion of Categorical, Feature selection
Used Models
Naive Bays, Random Forest, KNN; SGD
Accuracy
Evaluation
Confusion Matrix
Accuracy, Precision, Recall and F1-Score
Results
Comparative Analysis

Figure 1. Proposed Framework

A collection of tweets that were labeled as ironic, figurative, sarcastic, or regular were collected for this study. This dataset, which was built on the sarcastic-Tweets2 dataset, was created by merging human annotation with keyword-based filtering.

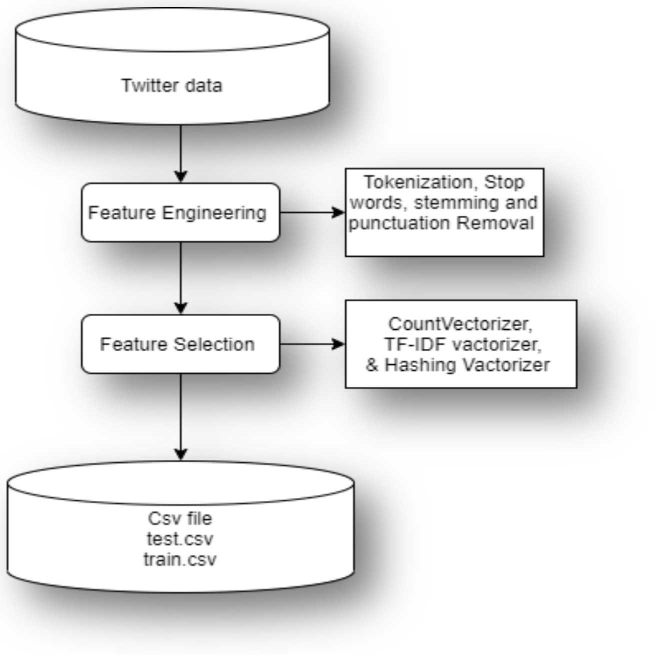

Twitter data
Feature Engineering
Tokenization, Stop words, stemming and punctuation Removal
Feature Selection
CountVectorizer, TF-IDF vactorizer, & Hashing Vactorizer
Csv file
test.csv
train.csv

Figure 2. Data Analysis Methodology

After the dataset was generated, four different machine learning algorithms—Nave Bayes (NB), Random Forest (RF), Stochastic Gradient Descent (SGD), and KNN—were trained on it. These algorithms were chosen because they are commonly employed for text classification tasks and have successfully been applied to datasets with a similar structure.

3. Dataset Description

The dataset is available on Kaggle and contains 62904 tweets related to sarcasm. The dataset is divided into test and train.csv files. The tweet column displays the textual tweet that a Twitter user made, and the class column displays the target label that is divided into the categories of regular, sarcasm, and irony in Table 3-1.

Table 1. Dataset Description

Features	No of instances	Type	Description
Tweets	62904	Object	The text of tweets
Class	62904	Object	The respective class to which tweets belong.

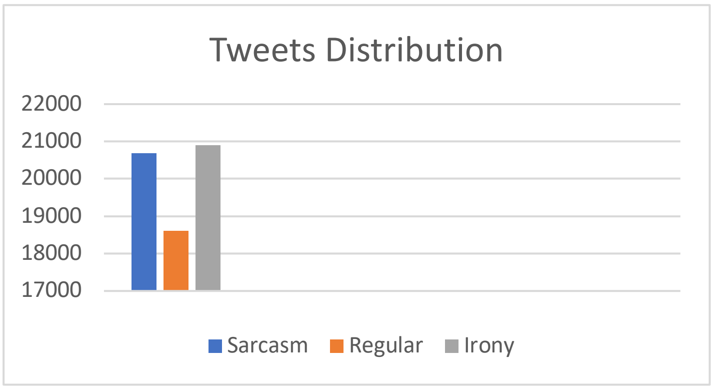


Figure 3. Tweet Distribution

3.1 Text Categorization

The automated classification of massive amounts of textual data is made possible by the interdisciplinary area of text categorization, which combines NLP and machine learning techniques. Text categorization is the process of automatically classifying documents into predetermined groups. However, text data needs to be pre-processed to extract useful features for machine learning algorithms because it contains raw data.

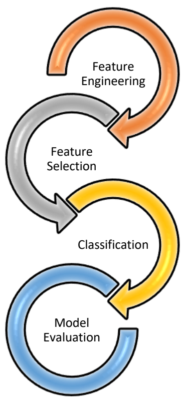


Figure 4. Text categorization process

3.2 Feature Engineering

The process of choosing and modifying relevant features from raw data to enhance the effectiveness of a machine-learning model is known as feature engineering. It involves building novel factors or features that can give the model access to more relevant data and support more precise forecasts. A well-designed collection of features can enhance the model's accuracy and interpretability, reduce over-fitting, and increase its efficiency and scalability. Statistical analysis, domain expertise, decrease in dimensionality, and feature selection are methods for feature engineering.

3.3 Tokenization

Tokenization, a term used in natural language processing (NLP), is the division of a text into tokens, which are usually words or sub-words. Many NLP tasks, such as language modeling, text categorization, and machine translation, depend on tokenization. Depending on the task and the language, the tokenization procedure may be simple or complicated. Because words in English are divided by spaces, for instance, tokenization is typically simple. Tokenization is more difficult in languages where words are not divided by spaces, such as Chinese and Japanese.

Tweet Text: On holidays and weekends, I enjoy playing sports.

Tokenized Text: ‘holidays’ ‘and’ ‘weekends’ ‘,’ ‘I’ ‘enjoy’ ‘playing’ ‘sports’

3.4 Eliminating Stop Word

Stop words are frequently used words that, in natural language processing (NLP), are usually filtered out of the text before further processing. Because stop words are frequently noisy keywords that don't contribute much meaning to the text, removing them is a common preprocessing step in many NLP tasks.

By eliminating stop terms, you can shrink your Vocabulary boosts the effectiveness and precision of tasks that come later, like text classification and sentiment analysis.

Tokenization:'On' 'holidays' 'and' 'weekends' ',' 'I' 'enjoy' 'playing' 'sports'

Text after Removal of stop word: 'holidays' 'weekends' ',' 'I' 'enjoy' 'playing' 'sports'

3.5 Elimination of Punctuation

Punctuation in natural language processing (NLP) refers to symbols and special characters that are used in text to structure and communicate meaning, including commas, periods, question marks, and exclamation points. Although punctuation can help us comprehend how a sentence is put together, it is frequently removed during text preprocessing to make the text simpler and lower noise in subsequent tasks.

Text after Removal of stop word:

'On' 'holidays' 'and' 'weekends' ',' 'I' 'enjoy' 'playing' 'sports'

Text after Removal of Punctuation:

'holidays' 'and' 'weekends' 'I' 'enjoy' 'playing' 'sports'

3.6 Stemming

By removing any affixes, such as suffixes or suffixes, stemming in natural language processing (NLP) reduces words to their base or stem form. Although the resulting stem might not be a real term, it captures the essence of the original word. To increase the effectiveness and accuracy of NLP tasks, stemming aims to condense the vocabulary and group together related terms.

Text after Removal of Punctuation:

'On' 'holidays' 'and' 'weekends' 'I' 'enjoy' 'playing' 'sports'

Text after Stemming:

Holidays and weekends I enjoy play sports

Table 2. Feature engineering of the tweet

Tweet Text:
On holidays and weekends, I enjoy playing sports.
Tokenized Text:
'holidays' 'weekends' ',' 'I' 'enjoy' 'playing' 'sports'
Text after Removal of stop word:
'holidays' 'and' 'weekends' 'I' 'enjoy' 'playing' 'sports'
Text after Removal of Punctuation:
'holidays' 'and' 'weekends' 'I' 'enjoy' 'playing' 'sports'

Text after Stemming:

holidays and weekends I enjoy play sports

3.7 Feature Selection

To increase the efficacy and precision of a predictive model, feature selection is a method used in machine learning and natural language processing (NLP). It involves choosing a subset of the features that are most significant or variables from a larger set of features. The purpose of feature selection is to simplify the model, avoid over-fitting, and enhance the findings' readability.

The Term Frequency-Inverse Document Frequency (TF-IDF) refers to a method used frequently in natural language processing to assess the significance of a term in a document. Each term receives a score based on its usage frequency in the text and its inverse usage frequency in the corpus. The frequency of the term in the corpus, which serves to weed out common terms that do not help differentiate between documents, offsets the TF-IDF score for a term that rises proportionally to how many times it appears in the document. Machine learning algorithms for tasks like classification, clustering, and information retrieval can use the resulting number as a feature.

3.8 Evaluation Metrics

Testing and evaluating each applied model involves the use of a scientific evaluation metric parameter. Four advanced metrics are employed to assess the performance of used models, and in this research study there are several performance evaluation metrics including accuracy and precision. The following are the basic notations of evaluations metrics parameters.

Four advance evalution metrics are used:

Precision: This evaluates the percentage of cases the model predicted as positive that turned out to be true positives **(TP).** (i.e., TP + FP).

Accuracy: This measures the proportion of occurrences in the dataset that were successfully divided out of all the instances.

Recall: This determines the proportion of cases between all the true positives that belong to the positive class. (i.e., TP + FN)

F1 Score: This is the harmonic mean of recall and precision. It is a combination measure that strikes a compromise between precision and memory and is helpful in situations where both precision and recall are crucial.

4. Experiment & Results

Python is an intelligent development tool that is considered the language of choice for computer models of learning and education. Python also includes specific tools that are quite helpful when we are working with machine learning systems. There are numerous Python frameworks and modules available for various purposes, and some of the most popular ones are NumPy, Pandas, Scikit-learn, etc. All these tools assist professionals in learning Python.

4.1 System Specifications

The specifications of the system that is used for the experiment are given as:

Table 3. System Specification.

Processor	**Core i5**
Model	Intel
RAM	4GB
OS	Windows 10
Tool	Jupyter Notebook

Language	Python

4.2 Feature Analysis

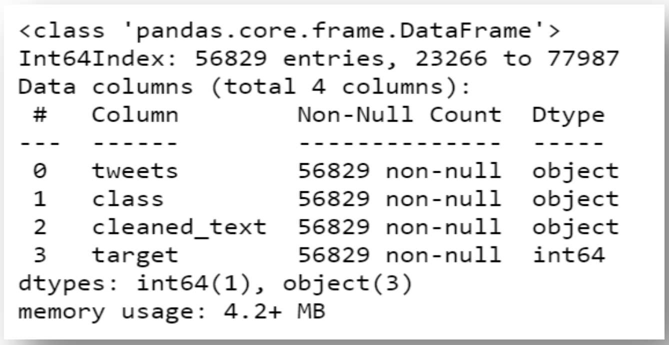
The biggest problem addressed in our dataset is that the data is not labeled. It is necessary to label data before applying the classification. It helps us to improve the accuracy of ML models. In this study, we used classification models such as Naïve Bayes, Random Forest, SGD (Stochastic Gradient Descent), and KNN (K-nearest neighbor). We also predict the precision, recall, F1 score, and accuracy of each ML Model. By leveraging different test options offered by Python as well as percentage split, we were able to achieve model results.

4.3 Dataset Anatomy

Our dataset has 2 csv files i.e. train.csv and test.csv and the total entries are 62904.

Train dataset:

The following figure explains the complete anatomy of our dataset. Each feature has its datatype such as the target has an int64 data type. Tweets, class, and cleaned text have an object data type. The memory usage of the dataset is 4.2+ MB.

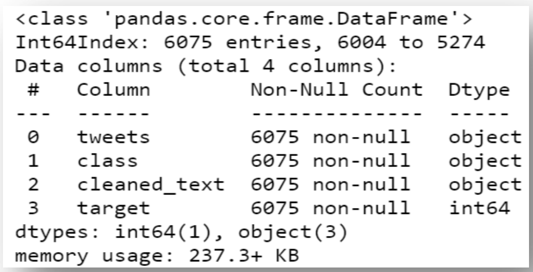


```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 56829 entries, 23266 to 77987
Data columns (total 4 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   tweets        56829 non-null  object
 1   class         56829 non-null  object
 2   cleaned_text  56829 non-null  object
 3   target        56829 non-null  int64
dtypes: int64(1), object(3)
memory usage: 4.2+ MB
```

Figure 5. Train Dataset Anatomy

Test dataset:

The following figure explains the complete anatomy of our test dataset. Each feature has its data type such as the target has int64 data type. Tweets, class, and cleaned text have an object data type. The memory usage of the dataset is 4.2+ MB.

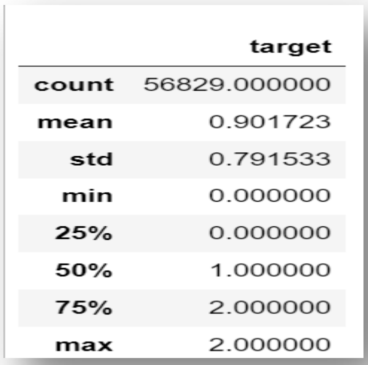


```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 6075 entries, 6004 to 5274
Data columns (total 4 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   tweets        6075 non-null   object
 1   class         6075 non-null   object
 2   cleaned_text  6075 non-null   object
 3   target        6075 non-null   int64
dtypes: int64(1), object(3)
memory usage: 237.3+ KB
```

Figure 6. Test Data Anatomy

Train Data Description

The following figure describes the all-important aspects of the training dataset. It includes the count of all features, minimum value, maximum value, mean and standard deviation of all features of the dataset.

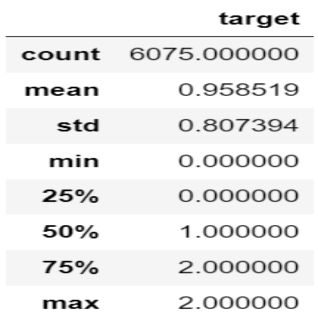


	target
count	56829.000000
mean	0.901723
std	0.791533
min	0.000000
25%	0.000000
50%	1.000000
75%	2.000000
max	2.000000

Figure 7. Train Data Description

Test Data Description:

The following figure describes the all-important aspects of the test dataset. It includes the count of all features, minimum value, maximum value, mean and standard deviation of all features of the test dataset.

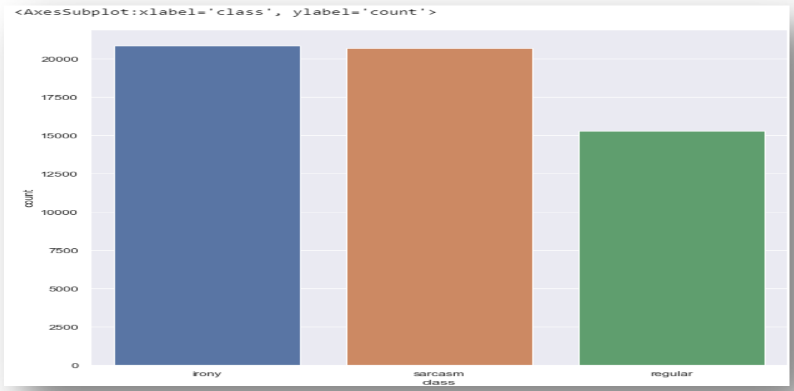


	target
count	6075.000000
mean	0.958519
std	0.807394
min	0.000000
25%	0.000000
50%	1.000000
75%	2.000000
max	2.000000

Figure 8. Test Data Description

Data Visualization

Figure 4.5 shows the graphical representation of the dataset by using the "matplotlib" library. The figure size is [10, 10]. The features of a dataset are plotted in the following figure.



```
<AxesSubplot:xlabel='class', ylabel='count'>
```

Figure 9. Data Visualization

Performance Model Evalution

To construct our model we use four classifiers: Naive Bayes Classifier, Random forest, Stochastic Gradient Descent (SGD) Classifier, and K-nearest neighbor.

Accuracy:

Table 4.8 is containing the accuracies of all four ML models. Each model has one type of accuracy for testing data. It contains the accuracy of NB at 89.63%, RF at 99.95% SGD at 99.93%, and KNN at 80.02%.

Table 4. Accuracy

Models	Accuracy
NB	89.63
RF	99.95
SGD	99.93
KNN	80.02

Figure 4.11 depict the accuracy of different models. The models SGD, RF, and the NB Classifier have the Highest Accuracy on testing data, and the lowest Accuracy is given on the model KNN Classifier.

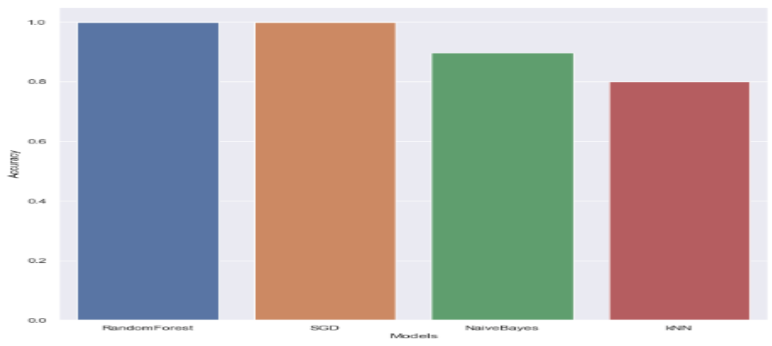


Figure 10. Comparison of model

In the above Graph accuracy ML models are compared. The maximum accuracy is calculated by the Random forest classifier as 99.95% and the minimum accuracy is calculated by the KNN classifier as 80.02%.

Precision:

The Precision scores of all selected four ML Models are given in Table 4.9 with each class. RF and SGD classifiers perform well with every class with an accuracy of 100 Percent accuracy. KNN gives Maximum precision in the Regular class with a score of 93 % and gives minimum precision in the sarcasm class with a score of 72%. The NB Classifier gives Maximum accuracy at class Regular with a precision of 98% and the other two classes' irony and sarcasm gave the same accuracy with a precision of 87%.

Table 5. Precision

Models	Irony	Sarcasm	Regular
NB	87	87	98
RF	100	100	100
SGD	100	100	100
KNN	81	72	93

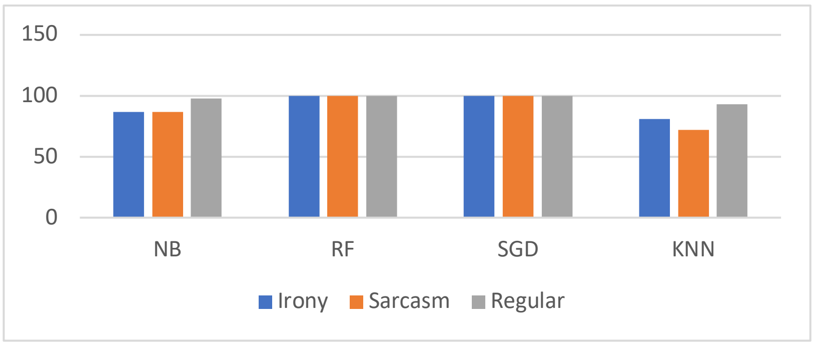


Figure 11. Precision

Recall of all four ML Models are given in Table 4.10 with each class (Irony, Sarcasm, and Regular). Recall of RF and SGD Classifiers has maximum at all classed with the performance of 100 percent. NB Classifier gives maximum Recall on the Irony class with a score of 96 % and gives minimum Recall on the class Regular with a poor performance of 78 %. KNN Classifier gives maximum Recall on the Sarcasm class with a score of 92 % and gives minimum Recall on the class Regular with poor performance of 70 %.

Table 6. Recall

Models	Irony	Sarcasm	Regular
NB	96	94	78
RF	100	100	100
SGD	100	100	100
KNN	77	92	70

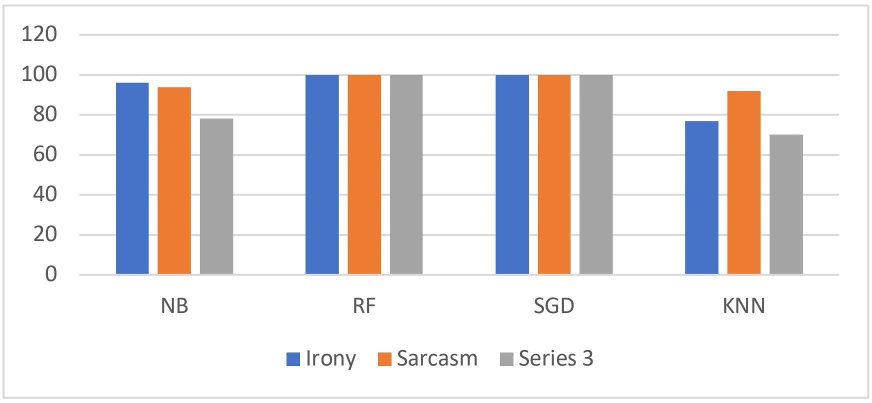


Figure 12. Recall

F-1 Score of selected ML models are given in Table 4.11 with each class. F-1 Score of the NB Classifier Shows that the highest F1 Score in the class Irony with an accuracy of 91 and the lowest F-1 Score in Sarcasm and Regular emotional behavior of the users with an accuracy of 90 and 87 percent respectively. RF Classifier gives a maximum F-1 Score on all classes with a performance of 100%. The SGD Classifier gives maximum performance in all classes with a Score of 100% percent. KNN Classifier gives maximum F-1 Score at Sarcasm with 81 percent and poor performance.

Table 7. F-1 Score

Models	Irony	Sarcasm	Regular
NB	91	90	87
RF	100	100	100
SGD	100	100	100
KNN	79	81	80

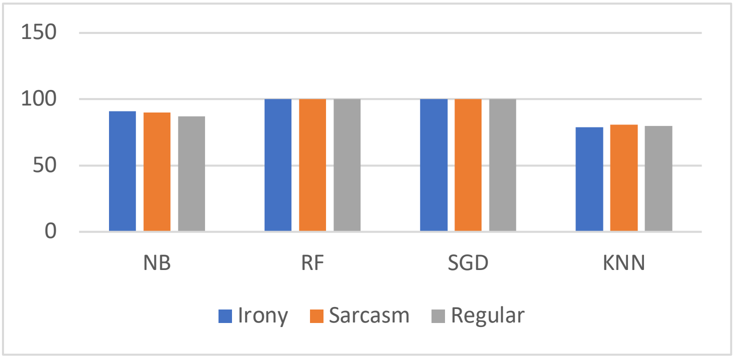


Figure 13. F1-Score

5. Conclusions

Communicating in a way that seems to be the complete reverse of what is intended to mock or ridicule someone or something is known as sarcasm. Due to the rising popularity of social media and the spread of dubious and misleading information, it is mandatory to detect sarcasm. Based on the study's findings, it can be said that the suggested method for sarcasm detection on social media is successful and has a high accuracy score of 99.01%. The features of the dataset were expanded to increase the accuracy of ML models by evaluating training, and testing the dataset. It uses the TF-IDF for feature selection, along with other suggested methods for feature engineering and four machine learning algorithms (KNN, Random forest, SGD, and Naïve Bayes). The ML models worked effectively with optimized feature selection and dataset cleaning. In the proposed methodology unique ML models are chosen to attain more accurate results. This shows the potential of machine learning techniques for automatically detecting sarcasm, which can be difficult to discern manually on social media sites.

References

1. G. W. Index, "Social Global Web Index's Flagship Report on the Latest Trends in Social Media," ed: GlobalWebIndex London, 2018. t. t. desk. "Quarterly results."
2. W. McKinney, Python for Data Analysis. October 2012.
3. B. S. R. Namasani Sagarika, Vanka Varshitha, Kodavati Geetanjali, N V Ganapathi Raju*, Latha, and Kunaparaju, "Sarcasm Discernment on Social Media Platform " 2021.
4. E. K. Steven Bird, Edward Loper, Natural Language Processing with Python. O'Reilly Media, Inc.
5. M. Bouazizi and T. O. Ohtsuki, "A pattern-based approach for sarcasm detection on Twitter," IEEE Access, vol. 4, pp. 5477-5488, 2016.
6. C. Florea, L. Florea, and C. Vertan, "Computer Vision for Cognition: An Eye Focused Perspective," in Computer Vision for Assistive Healthcare: Elsevier, 2018, pp. 51-74.
7. A. Ashwitha, G. Shruthi, H. Shruthi, M. Upadhyaya, A. P. Ray, and T. Manjunath, "Sarcasm detection in natural language processing," Materials Today: Proceedings, vol. 37, pp. 3324-3331, 2021.
8. M. U. Khamdamovna, "Aesthetic and Psychological Features of Irony," International Journal on Integrated Education, vol. 4, no. 10, pp. 184-187, 2021.
9. P. Kałowski, K. Szymaniak, and O. Maciantowicz, "Exploring the links between trait anger, self-reported sarcasm use, and narcissism," Advances in Cognitive Psychology, vol. 17, no. 4, pp. 261-273, 2021.
10. A. Reyes, P. Rosso, and T. Veale, "A multidimensional approach for detecting irony in Twitter," Language resources and evaluation, vol. 47, pp. 239-268, 2013.
11. A. Reyes, P. Rosso, and D. Buscaldi, "From humor recognition to irony detection: The figurative language of social media," Data & Knowledge Engineering, vol. 74, pp. 1-12, 2012.
12. L. Xu and V. Xu, "Project Report: Sarcasm Detection," published in, 2019.
13. S. M. Sarsam, H. Al-Samarraie, A. I. Alzahrani, and B. Wright, "Sarcasm detection using machine learning algorithms in Twitter: A systematic review," International Journal of Market Research, vol. 62, no. 5, pp. 578-598, 2020.
14. S. Saha, J. Yadav, and P. Ranjan, "Proposed approach for sarcasm detection in Twitter," Indian Journal of Science and Technology, vol. 10, no. 25, pp. 1-8, 2017.
15. R. Sivanaiah, S. Angel Deborah, R. Sakaya Milton, T. Mirnalinee, and R. Venkatakrishnan, "TechSSN at SemEval-2022 Task 6: Intended Sarcasm Detection using Transformer Models."
16. K. Maity, P. Jha, S. Saha, and P. Bhattacharyya, "A Multitask Framework for Sentiment, Emotion, and Sarcasm Aware Cyberbullying Detection from Multi-modal Code-Mixed Memes," presented at the Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2022.
17. P. Singh, A. Maladry, and E. Lefever, "Combining Language Models and Linguistic Information to Label Entities in Memes," CONSTRAINT 2022, p. 35, 2022.
18. J. Godara, R. Aron, and M. Shabaz, "Sentiment analysis and sarcasm detection from social network to train health-care professionals," World Journal of Engineering, 2021. "<ICICCS48265.2020.9120917.pdf>."
19. E. Sevenois, "SENTIMENT ANALYIS OF INTERNET MEMES ON SOCIAL MEDIA PLATFORMS," Ghent University, 2021. "<2020.nlpbt-1.3.pdf>."
20. X. S. Xinyu Wang1, Tan Yang2, Hongbo Wang, "Building a Bridge: A Method for Image-Text Sarcasm Detection Without Pretraining on Image-Text Data," 2020.
21. A.-M. Bucur, A. Cosma, and I.-B. Iordache, "BLUE at Memotion 2.0 2022: You have my Image, my Text, and my Transformer," arXiv preprint arXiv:2202.07543, 2022."<2020.aacl-main.31.pdf>."
22. Y. Zhang et al., "CFN: a complex-valued fuzzy network for sarcasm detection in conversations," IEEE Transactions on Fuzzy Systems, vol. 29, no. 12, pp. 3696-3710, 2021.
23. A. Kumar, S. Dikshit, and V. H. C. Albuquerque, "Explainable artificial intelligence for sarcasm detection in dialogues," Wireless Communications and Mobile Computing, vol. 2021, 2021.
24. S. K. Bharti, K. S. Babu, and R. Raman, "Context-based sarcasm detection in Hindi tweets," in 2017 Ninth International Conference on Advances in Pattern Recognition (ICAPR), 2017: IEEE, pp. 1-6.
25. K. Rathan and R. Suchithra, "Sarcasm detection using combinational Logic and Naïve Bayes Algorithm," Imperial Journal of Interdisciplinary Research (IJIR), vol. 3, no. 5, pp. 546-551, 2017.
26. K. Sreelakshmi and P. Rafeeque, "An effective approach for detection of sarcasm in tweets," in 2018 International CET Conference on Control, Communication, and Computing (IC4), 2018: IEEE, pp. 377-382.