Jurnal Teknik Informatika Vol. 17 No. 1, April 2024 (88-97)
ISSN: p-ISSN 1979-9160 (Print)| e-ISSN 2549-7901 (Online)
DOI: https://doi.org/10.15408/jti.v17i1.38651

JURNAL TEKNIK INFORMATIKA

Homepage : http://journal.uinjkt.ac.id/index.php/ti

A Comparative Analysis of Random Forest, XGboost, and LightGBM Algorithms for Emotion Classification in Reddit Comments

Nenny Anggraini[1], Syopiansyah Jaya Putra[2], Luh Kesuma Wardhani [3*], Farid Dhiya Ul Arif [4], Nashrul Hakiem[5], Imam Marzuki Shofi[6]

[1]Doctoral Program of Islamic Studies, Graduate School, Syarif Hidayatullah State Islamic University Jakarta
[2]Information System Department, Faculty of Science and Technology, Syarif Hidayatullah State Islamic University Jakarta
[3,4,5,6]Informatics Engineering Department, Faculty of Science and Technology, Syarif Hidayatullah State Islamic University Jakarta
[1,2,3,4,5,6] Jl. Ir. H. Juanda No. 95 South Tangerang, Banten, 15412, Indonesia

ABSTRACT

Article:
Accepted: March 23, 2024
Revised: January 14, 2024
Issued: April 30, 2024

***Correspondence Address:**
luhkusuma@uinjkt.ac.id

This research aims to compare the performance of three classification algorithms, namely Random Forest, XGBoost, and LightGBM, in classifying emotions in Reddit comments. Emotion classification in Reddit comments is a complex classification problem due to its numerous variations and ambiguities. This research utilizes the GoEmotions Fine-Grained dataset, filtered down to 7,325 Reddit comments with 5 different basic emotion labels. In this study, data preprocessing steps, feature extraction using CountVectorizer and TF-IDF, and hyperparameter tuning using GridSearchCV for each algorithm are conducted. Subsequently, model evaluation is performed using Cross-Validation and confusion matrix. The results of the study indicate that Random Forest outperforms the XGBoost and LightGBM algorithm with an accuracy of 75.38% compared to XGBoost with 69.05% accuracy and LightGBM with 66.63% accuracy.

Keywords: *emotion classification; xgboost, random forest; lightGBM; reddit comments;*

1. INTRODUCTION

1.1. Research Background

In this rapidly evolving technological era, data has become the key to the success of businesses and organizations. Data provides valuable insights, such as customer behavior, market trends, and business performance [1]. Therefore, data collection, analysis, and management are crucial. Social media, as one of the data sources, is particularly significant due to its easy accessibility and widespread usage [2]. The website backlinko.com mentions Reddit as the ninth largest social media platform in the US with a value of 10 billion dollars in 2023. Reddit, with its diverse sub-forums, creates a varied dataset that can be utilized for various purposes, including business, organizations, and government. Machine learning has become a primary technique for processing and analyzing data. It enables computers to learn independently from data without explicit instructions. Emotion classification, one of the methods in machine learning, is utilized in applications such as chatbots, speech recognition, and sentiment analysis. Algorithms like Random Forest, XGBoost, and LightGBM are commonly used for emotion classification.

1.2. Preliminary Information

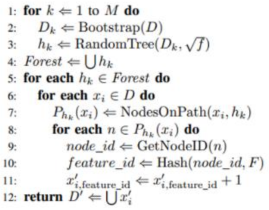
1.2.1. GoEmotions

GoEmotions, created in 2020 by Demszky et al., is a dataset developed by researchers from the University of California, Berkeley, and Stanford University [3]. This dataset aims to facilitate research in natural language processing (NLP) and sentiment analysis, as well as to advance the development of better systems in understanding and processing human emotions in everyday language.

1.2.2. Random Forest

Random Forest is a popular classification algorithm in machine learning [4]. It is an improvement over Decision Trees, and this algorithm is frequently used in classification and regression problems. Its superiority is reflected in the fact that it has been used in more than 1000 journals since 2018. In previous research, "Sentiment Analysis of Movie Reviews Using the Random Forest Algorithm" [5], this algorithm achieved an accuracy of 75.44%, demonstrating its suitability for avoiding overfitting and dependency on specific datasets.

The operation of Random Forest involves selecting random samples from the dataset, constructing decision trees for each selected sample, and performing voting for the prediction results from each decision tree. The final prediction result is obtained from the most frequently predicted outcome through averaging. In classification, this means taking the mode, while in regression, the prediction result is taken from the mean. Generally, increasing the number of trees in the Random Forest correlates with improved accuracy, allowing the algorithm to comprehensively learn from the data as shown in Figure 1.

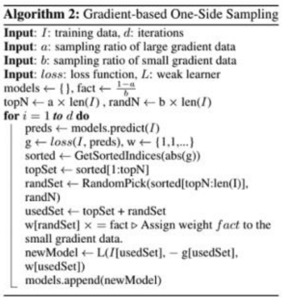


```
for $k \Leftarrow 1$ to $M$ do
    $D_k \Leftarrow \text{Bootstrap}(D)$
    $h_k \Leftarrow \text{RandomTree}(D_k, \sqrt{f})$
$Forest \Leftarrow \bigcup h_k$
for each $h_k \in Forest$ do
    for each $x_i \in D$ do
        $P_{h_k}(x_i) \Leftarrow \text{NodesOnPath}(x_i, h_k)$
        for each $n \in P_{h_k}(x_i)$ do
            $node\_id \Leftarrow \text{GetNodeID}(n)$
            $feature\_id \Leftarrow \text{Hash}(node\_id, F)$
            $x'_{i,\text{feature\_id}} \Leftarrow x'_{i,\text{feature\_id}} + 1$
return $D' \Leftarrow \bigcup x'_i$
```

Figure 1. *Random forest pseudocode*

1.2.3. LightGBM

LightGBM is a gradient-based tree learning library. This algorithm utilizes histogram-based methods, reducing memory usage, and applies a leaf-wise leaf growth strategy with depth constraints to accelerate model building [6]. LightGBM also introduces effective techniques such as leaf-based tree selection and dividing datasets into small parts, enhancing the efficiency of the learning process. We can see the pseudocode in Figure 2.



```
Algorithm 2: Gradient-based One-Side Sampling
Input: $I$: training data, $d$: iterations
Input: $a$: sampling ratio of large gradient data
Input: $b$: sampling ratio of small gradient data
Input: $loss$: loss function, $L$: weak learner
models ← {}, fact ← $\frac{1-a}{b}$
topN ← a × len($I$) , randN ← b × len($I$)
for $i = 1$ to $d$ do
    preds ← models.predict($I$)
    g ← $loss$($I$, preds), w ← {1,1,...}
    sorted ← GetSortedIndices(abs(g))
    topSet ← sorted[1:topN]
    randSet ← RandomPick(sorted[topN:len(I)],
    randN)
    usedSet ← topSet + randSet
    w[randSet] × = fact ▷ Assign weight $fact$ to the
    small gradient data.
    newModel ← L($I$[usedSet], − g[usedSet],
    w[usedSet])
    models.append(newModel)
```

Figure 2. *LightGBM pseudocode*

LightGBM is a machine learning algorithm developed by Microsoft. LightGBM optimizes gradient boosting with a method of fast and efficient data splitting [7], allowing it to run faster and use less memory compared to XGBoost. In previous research, "Comparison of Accuracy between Adaboost and LightGBM Algorithms for Diabetes Disease Classification"

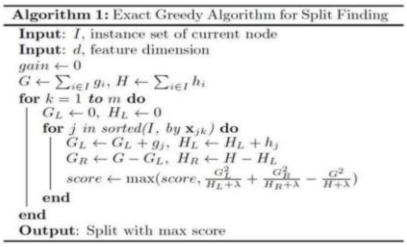
[6], LightGBM achieved an accuracy of 90% in processing diabetes disease data.

In its operation, LightGBM gradually builds a series of Decision Tree or linear regression models, where each model corrects the errors of the previous model. Through gradient boosting, this algorithm learns from the previous models and adds features to improve the results. By utilizing techniques such as tree selection and histogram, LightGBM speeds up the learning process and reduces memory requirements at each stage [8].

1.2.4. XGBoost

XGBoost (eXtreme Gradient Boosting) is a machine learning method that handles classification and regression problems [8]. Developed by Tianqi Chen in 2014, XGBoost addresses the weaknesses of previous gradient boosting algorithms, resulting in a more efficient and accurate algorithm [9]. In the Kaggle survey of 2022, approximately 50% of the 24,000 respondents had used XGBoost/CatBoost/LightGBM. The advantages of XGBoost include faster computational time and the ability to accurately handle data with many features compared to Random Forest [10]. In previous research, "Success Prediction using Random Forest, CatBoost, XGBoost and AdaBoost for Kickstarter Campaigns" [11], XGBoost achieved an accuracy of 74.29% in processing Kickstarter campaign data.

This algorithm applies the Gradient Boosting Decision Tree technique, combining several small models, usually Decision Trees, sequentially to form a stronger model and achieve higher accuracy. XGBoost utilizes Decision Trees with the same depth, following either a level-wise or depth-wise approach. To enhance efficiency and effectiveness, XGBoost introduces various techniques such as regularization and loss functions [12]. Figure 3 shows the pseudocode of the XGBoost algorithm.



```
Algorithm 1: Exact Greedy Algorithm for Split Finding
Input: $I$, instance set of current node
Input: $d$, feature dimension
$gain \leftarrow 0$
$G \leftarrow \sum_{i \in I} g_i,\ H \leftarrow \sum_{i \in I} h_i$
for $k = 1$ to $m$ do
    $G_L \leftarrow 0,\ H_L \leftarrow 0$
    for $j$ in $sorted(I,$ by $\mathbf{x}_{jk})$ do
        $G_L \leftarrow G_L + g_j,\ H_L \leftarrow H_L + h_j$
        $G_R \leftarrow G - G_L,\ H_R \leftarrow H - H_L$
        $score \leftarrow \max(score, \frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{G^2}{H+\lambda})$
    end
end
Output: Split with max score
```

Figure 3. *XGBoost pseudocode*

1.2.5. TF-IDF

The TF-IDF (Term Frequency-Inverse Document Frequency) method is an approach used to calculate the weight of words commonly used in information retrieval. The popularity of TF-IDF is attributed to its efficiency, ease of implementation, and the accuracy of the results it provides [13]. This method provides information about the significance of a word in a document within a corpus [14].

TF-IDF is based on two main metrics: Term Frequency (TF) and Inverse Document Frequency (IDF). TF measures the frequency of a word's occurrence in a specific document, providing an indication of the relevance of that word to the document. On the other hand, IDF measures how common the word is across the entire document collection, capturing its importance across the corpus as a whole.

1.3. Related Works

There are several previous studies that have relevance to the research conducted by the author, the research conducted by Daoud [15] entitled "Comparison between XGBoost, LightGBM and CatBoost Using a Home Credit Dataset" using a home credit dataset to compare the three algorithms with the parameters of feature ranking, auc, and modelling time. In this study, the XGBoost, LightGBM and Decision Tree algorithms produced 92.5% accuracy. Random Forest algorithm produces 92.2% accuracy. Another research conducted by Ahsana et al. [16] entitled "Comparison of Accuracy of Adaboost Algorithm and LightGBM Algorithm for Diabetes Disease Classification" using a diabetes disease dataset to compare the two algorithms with accuracy, precision, recall, f1-score and auc parameters. This study also compares the performance of machine learning calculations with Adaboost SAME, Adaboost SAME.R, LightGBM GBDT, LightGBM, DART, LightGBM GOSS. Then, the research conducted by Zhang et al [17] entitled "The Comparison of LightGBM and XGBoost Coupling Factor Analysis and Pre Diagnosis of Acute Liver Failure" by comparing accuracy and efficiency and including dimensionality reduction on LightGBM, external dimensionality reduction on XGBoost. This study compared the performance of the two algorithms with the results of 75.8% accuracy on LightGBM and 67.7% accuracy on XGBoost.

There is also a research on the emotion classfication conducted by Anand et al. in 2022 [18] titled "EmoSens: Emotion Recognition based on Sensor Data Analysis using LightGBM" utilized EEG and ECG datasets with multi-emotions, specifically 9 classes of emotions. This study compared several classification algorithms including XGBoost, LightGBM, Decision Tree, and Random Forest. The XGBoost, LightGBM, and Decision Tree algorithms achieved an accuracy of 92.5%, while the Random Forest algorithm produced an accuracy of 92.2%. These findings suggest that LightGBM can be a sustainable method for emotion analysis based on sensor data, providing a foundation for future research to enhance emotion recognition systems with high accuracy and efficiency.

Building on the theme of leveraging machine learning for classification tasks, research conducted by Syukron et al at 2020 [19] titled "Comparison of Smote Random Forest and Smote XGBoost Methods for Classification of Hepatitis C Disease Levels on Imbalance Class Data" aimed to compare the classification algorithms Random Forest, XGBoost, Random Forest with SMOTE, and XGBoost with SMOTE. This study compared accuracy and recall, finding that the highest accuracy was achieved by the Random Forest algorithm balanced with SMOTE at 80.97%, while the highest recall was achieved by XGBoost balanced with SMOTE at 76.82%. These results underscore the importance of using SMOTE for class imbalance correction, suggesting a sustainable approach to improving the classification of imbalanced medical data, which can be crucial for better disease diagnosis and treatment.

Continuing the exploration of machine learning algorithms effectiveness in various domains, the research conducted by Jhaveri et al at 2019 [20] titled "Success Prediction using Random Forest, CatBoost, XGBoost and AdaBoost for Kickstarter Campaigns" aimed to compare the performance of these algorithms using three parameters: accuracy, precision, and recall. The study found that the highest accuracy was achieved by CatBoost at 83.3%, the highest precision by Random Forest combined with AdaBoost at 82%, and the highest recall by CatBoost at 75%. These findings indicate that CatBoost provides a sustainable and robust method for predicting the success of Kickstarter campaigns, offering a valuable tool for future applications in crowdfunding platforms to enhance campaign success rates through more accurate predictions.

From the literature review, random forest, XGBoost and lightGBM algorithms have been widely used for classification. Many previous studies have compared the performance of these three algorithms and other algorithms on various case studies, but there has been no research comparing the performance of these three algorithms for emotion classification in Reddit comments. So this research aims to compare the performance of random forest, XGBoost and lightGBM algorithms for emotion classification in Reddit comments.

2. METHODS

This research flow includes several stages as shown in Figure 4:

a. Initial research involves literature review, analysis of research needs, and dataset collection.
b. Data preparation is done through several preprocessing stages, such as case folding, data cleaning, data selection, data augmentation, tokenization, stopword removal, and stemming.
c. After dataset cleaning, weights are assigned to each word using the TF-IDF method.
d. If the dataset has been formatted in binary form by TF-IDF, the data is divided into two parts: training data (80%) and test data (20%). In the initial classification stage, Random Forest, XGBoost, and LightGBM algorithms are used.
e. The best hyperparameters are searched using Grid Search and 10-fold cross-validation for each algorithm.
f. The hyperparameter results from Grid Search are used for each algorithm.
g. Next, the average performance of the Random Forest, XGBoost, and LightGBM algorithms is calculated using accuracy, precision, recall, and f1-score parameters.

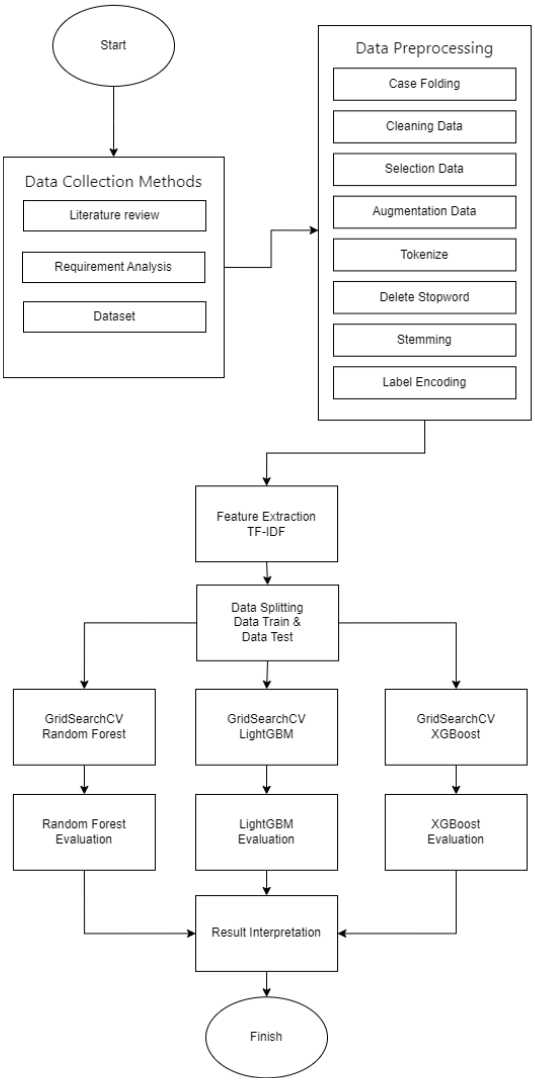


Figure 4. *Research flow*

2.1. Data Collection

2.1.1. Literature Review

In the literature review, most of the data that the authors used came from journals. the author uses comes from journals. There are several journals that used as a reference in this research and most of them have websites. have websites. Researchers also utilise journals that discuss sentiment analysis classification as a data source for classification purposes. In addition, some journals also provide code and programme resources as well as dictionaries that can be utilised in this research.

2.1.2. Requirement Analysis

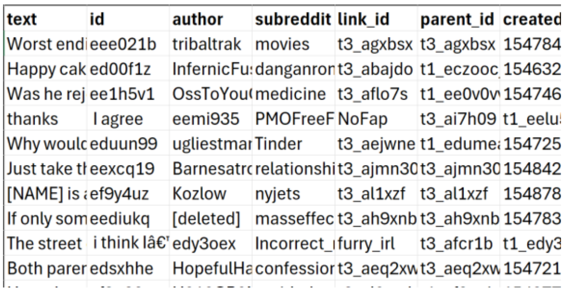
The implementation of this research requires devices used for research. The devices used in this research use 2 types, namely hardware and software. The software specifications used are describe in Table 1.

Table 1. *Software specifications*

Operating System	POP!_OS 18.04 LTS x86_64
Kernel	5.3.0-7648-generic
Tools	jupyter notebook
Programming Language	python v3.8
Library	sklearn, seaborn, pandas, matplotlib, XGBoost, LightGBM, Random Forest, re, PyStemmer, sastrawi

2.1.3. Dataset

In this initial stage, the dataset needed is a collection of English-language data on the Reddit site that will be modelled using Random Forest, XGBoost and LightGBM. This dataset is called GoEmotions by its creator and the example is shown in Figure 5.



text	id	author	subreddit	link_id	parent_id	created
Worst endi	eee021b	tribaltrak	movies	t3_agxbsx	t3_agxbsx	154784
Happy cak	ed00f1z	InfernicFu:	danganron	t3_abajdo	t1_eczooc	154632
Was he rej	ee1h5v1	OssToYou(	medicine	t3_aflo7s	t1_ee0v0w	154746
thanks	I agree	eemi935	PMOFreeF	NoFap	t3_ai7h09	t1_eelu
Why woulc	eduun99	ugliestmar	Tinder	t3_aejwne	t1_edume:	154725
Just take tl	eexcq19	Barnesatrc	relationshi	t3_ajmn30	t3_ajmn30	154842
[NAME] is :	ef9y4uz	Kozlow	nyjets	t3_al1xzf	t3_al1xzf	154878
If only som	eediukq	[deleted]	masseffec	t3_ah9xnb	t3_ah9xnb	154783
The street	i think lâ€'	edy3oex	Incorrect_	furry_irl	t3_afcr1b	t1_edy3
Both parer	edsxhhe	HopefulHa	confessior	t3_aeq2xw	t3_aeq2xw	154721

Figure 5. *Dataset GoEmotions*

2.2. Data Preprocessing

In emotion classification research that uses raw data, this preprocessing stage is necessary. The stages in preprocessing are case folding, data cleaning, data selection, data augmentation, tokenization, stemming and stopword removal. The results of this preprocessing data will be used as learning data in making models with Random Forest, XGBoost, and LightGBM algorithms. The purpose of this preprocessing data is to equalize words, removing noise, cleaning the dataset, and balancing the class.

2.2.1 Case Folding

In the case folding stage, the conversion from capital letters to lowercase letters is carried out as shown in Table 2.

Table 2. *Case Folding example*

Sentence	Result
I love 4 and 5 just as much as the early seasons. You can't recapture lightning in a bottle; I'm just glad they kept at it.	i love 4 and 5 just as much as the early seasons. you can't recapture lightning in a bottle; i'm just glad they kept at it.

2.2.2 Data Cleaning

At the data cleaning stage, researchers perform several stages, namely:

a. Deletion of words in square brackets such as "[NAME]".
b. Deletion on 1 line if the sentence on that line has a sentence length of less than 4, such as "I love you".
c. sentence length less than 4 such as "I love you".
d. Deleting words whose length is less than 3 such as the word "me".
e. Deleting URLs such as "www.google.com".
f. Symbols and numbers such as "123!@#".
g. Removing duplicate data.

The example of data cleaning is shown in table 3.

Table 3. *Data Cleaning Example*

Sentence	Result
i love 4 and 5 just as much as the early seasons. you can't recapture lightning in a bottle; i'm just glad they kept at it.	love and just much the early seasons you can't recapture lightning bottle just glad they kept

2.2.3 Data Selection

At the data selection stage, researchers deleted columns and only took reddit comment columns and selected emotions, namely anger, fear, joy, love, and sadness. At this stage, the dataset which was originally 58 thousand data was reduced to 7 thousand data because researchers only chose 5 basic emotions.

2.2.4 Data Augmentation

In the data augmentation stage, researchers add synthetic data with the back translation method, which means translating into a language and then reversing it back into the original language. The purpose of this augmentation can improve the performance of the algorithm and balance the data in the class. Table 4 show the example of data augmentation.

Table 4. *Data augmentation example*

Sentence	Result
I love 4 and 5 just as much as the early seasons. You can't recapture lightning in a bottle; I'm just glad they kept at it.	I like 4 and 5 the same as the initial season. You cannot catch lightning in the bottle; I'm glad they keep doing it

Table 5. *Adding data augmentation result*

Emotion	Sadness	Love	Joy	Fear	Anger
Original Data	696	1457	1697	1697	1778
Synthetic Data	1305	544	304	304	223
Total Data	2001	2001	2001	2001	2001

At this stage, the data distribution has been balanced in each class as seen in Table 5 and is ready for further preprocessing.

2.2.5 Tokenize

Table 6. *Tokenize data example*

Sentence	Result
love seasons recapture lightning bottle glad	['love', 'seasons', 'recapture', 'lightning', 'bottle', 'glad']

In the tokenisation stage, researchers divided sentences into words with the aim of preparing for further processing. Tokenize example is shown in Table 6.

2.2.6 Stopword Deleting

Table 7. *Stopword deleting example*

Sentence	Result
I love and just as much as the early seasons you can't recapture lightning in a bottle I'm just glad they kept at it	love seasons recapture lightning bottle glad

In the stage of removing stopwords, researchers removed words that have less important or no important meaning at all such as me, you, and, at like the example in Table 7.

2.2.7 Stemming

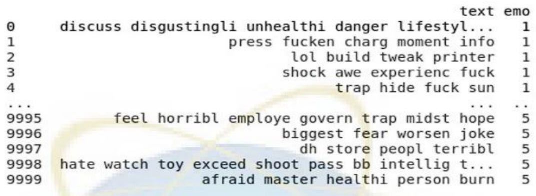
In the stemming stage, the researcher removes the affixes on the word to get the base word as shown in Table 8, for example "running" and "runner" will be converted to "run".

Table 8. *Stemming Example*

Sentence	Result
['love', 'seasons', 'recapture', 'lightning', 'bottle', 'glad']	['love', 'season', 'recapture', 'lightn', 'bottle', 'glad']

2.2.8 Label Encoding

At the label encoding stage, researchers made changes to the emo column dataset which was previously in the form of emotion category string values into integer values for emotion categories 1,2,3,4 and 5 as shown in Figure 5.

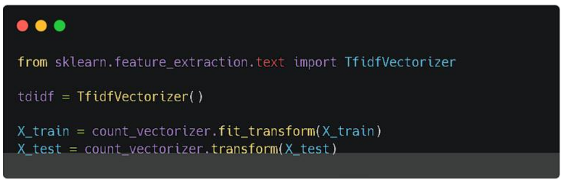


```
                                                   text  emo
0     discuss disgustingli unhealthi danger lifestyl...    1
1                       press fucken charg moment info    1
2                            lol build tweak printer    1
3                         shock awe experienc fuck    1
4                              trap hide fuck sun    1
...                                                ...   ..
9995          feel horribl employe govern trap midst hope    5
9996                         biggest fear worsen joke    5
9997                            dh store peopl terribl    5
9998  hate watch toy exceed shoot pass bb intellig t...    5
9999                    afraid master healthi person burn    5
```

Figure 5. *Label encoding example*

2.3. TF-IDF Feature Extraction

After the data is clean and free from noise, researchers use value weighting on the dataset using the TF-IDF method as a tool for converting text data into numerical data. This method counts the number of words that appear in the preprocessed data.



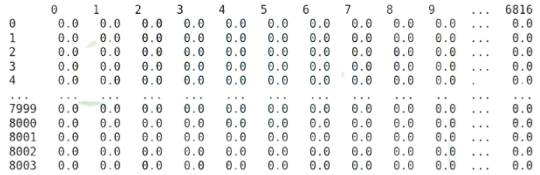
```
from sklearn.feature_extraction.text import TfidfVectorizer

tdidf = TfidfVectorizer()

X_train = count_vectorizer.fit_transform(X_train)
X_test = count_vectorizer.transform(X_test)
```

Figure 6. *Feature Code Extraction Using TF-IDF*

Figure 6 show the conversion of a dataset containing words into a dataset containing weighted values calculated by TF-IDF.



```
        0    1    2    3    4    5    6    7    8    9   ...  6816
0     0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  ...   0.0
1     0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  ...   0.0
2     0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  ...   0.0
3     0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  ...   0.0
4     0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  ...   0.0
...   ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...   ...
7999  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  ...   0.0
8000  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  ...   0.0
8001  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  ...   0.0
8002  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  ...   0.0
8003  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  ...   0.0
```

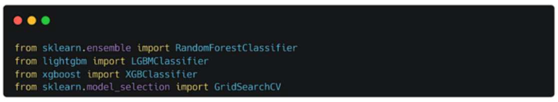
Figure 7. *Result of Word Value Weighting with TF-IDF*

In the picture above is the result of weighting the value of each word contained in the dataset. The results of weighting using the TF-IDF method are called features. There are 6,816 features used in this research.

2.4. GridSearchCV

Grid Search is a method used in hyperparameter selection to improve model performance. Hyperparameters are parameters that are set at the beginning of the model because they cannot be changed when the model is trained [21]. Definition of CV in Grid Search CV is Cross Validation which aims to validate the model crosswise.

Before creating the model, researchers conducted GridSearchCV to find the best hyperparameters in Random Forest, XGBoost and LightGBM algorithms. The parameters for GridSearchCV were taken from previous research that performed hyperparameter optimisation on each algorithm. GridSearchCV is combined with cross validation to get the best hyperparameter for each algorithm.



```
from sklearn.ensemble import RandomForestClassifier
from lightgbm import LGBMClassifier
from xgboost import XGBClassifier
from sklearn.model_selection import GridSearchCV
```

Figure 8. *Import library random forest, XGBoost dan LightGBM*

The code above is the initiation of the library used, namely the Random Forest, XGBoost, LightGBM and GridSearchCV algorithms to find hyperparameters.

2.5. Evaluation

The model evaluation stage is the last stage of classification. Researchers created 2 functions to calculate the evaluation with the name cross_val_predict and create a confusion matrix plot with the name plot_confusion_matrix. Making this function so that there is no repetition when used in other algorithms.

3. RESULTS AND DISCUSSION

3.1 GridSearchCV

Based on the previous processes conducted, the researchers performed a search for the best algorithm hyperparameters using GridSearchCV on each algorithm, utilizing 80% of the data or 8004 instances and 20% of the training data or 2001 instances. This GridSearchCV employs 10-fold cross-validation to obtain precise hyperparameters. Below are the results of the best hyperparameter search using GridSearchCV.

The best hyperparameter search results using GridSearchCV for the Random Forest algorithm shown in the table 9.

Table 9. *Grid search random forest range*

Hyperparameter	Parameter Random Forest
‘max_features’	'sqrt', 'log2'
‘n_estimator’	10, 100, 500, 1000, 1500

From the results of GridSearchCV in Random Forest with several parameters mentioned above, GridSearchCV produces n_estimator of 1500 and max_features with log2. Table 10 shows the best hyperparameter search results using GridSearchCV for the LightGBM algorithm.

Table 10. *Grid search lightgbm range*

Hyperparameter	LightBM Parameter
'max_depth'	2, 4, 6, 8, 10
'learning_rate'	0.01, 0.025, 0.05, 0.1
'n_estimator'	30, 50, 100, 150, 200, 250, 500, 750, 1000

From the results of GridSearchCV in LightGBM with several parameters mentioned above, GridSearchCV produces n_estimator of 500, max_depth of 2, and learning_rate of 0.1.

Table 11 shows the best hyperparameter search results using GridSearchCV for the XGBoost algorithm.

Table 11. *XGBoost grid search range*

Hyperparameter	Parameter XGBoost
'max_depth'	1,2,3
'learning_rate'	0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9
'n_estimator'	100, 150, 200, 250

From the results of GridSearchCV in XGBoost with several parameters mentioned above, GridSearchCV produces 100 to 250 n_estimator.

3.2 Evaluation of Training Results

In the evaluation stage of modelling results, researchers use the hyperparameters that have been obtained. The evaluation using 10 cross validations by calculating the accuracy, precision, recall and average F1-score of the Random Forest, XGBoost, and LightGBM models. Researchers also compared the confusion matrix results of the three algorithms.

Table 12 shows the results of the model classification performance in the Random Forest model using the n_estimator hyperparameter of 1500 and max_features with log2.

Table 12. *Results from gridsearch random forest*

Accuracy	85.71%
Precision	85.67%
Recall	85.76%
F1-Score	85.67%

In the LightGBM model, the hyperparameter n_estimator is used as much as 500, max_depth as much as 2 and learning_rate as much as 0.1 which results in model classification performance as shown in Table 13.

Table 13. *Results from gridsearch lightgbm*

Accuracy	70.88%
Precision	71.11%
Recall	70.93%
F1-Score	70.92%

In the XGBoost model, the hyperparameter n_estimator is used as much as 100, max_depth as much as 2 and learning_rate as much as 0.8 which results in model classification performance as shown in table 14.

Table 14. *Results from GridSearch XGBoost*

Accuracy	74.07%
Precision	74.17%
Recall	74.15%
F1-Score	74.06%

Figure 9 show the comparison results between the model outcomes of Random Forest, LightGBM, and XGBoost algorithms:

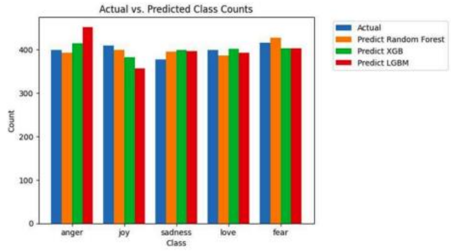


Figure 9. *Comparison result*

CONCLUSION

The research comparing the performance of three algorithms using the Reddit comments emotion dataset (10,000 instances) led to several conclusions. For the initial dataset comprising 7,325 instances, the Random Forest algorithm, with parameters n_estimators set to 1500 and max_features to log2, achieved an accuracy, precision, and recall of 85%. LightGBM, configured with a max_depth of 2, a learning_rate of 0.1, and n_estimators set to 500, provided an accuracy of 70%, precision of 71%, and recall of 70%. XGBoost, with a max_depth of 2, learning_rate of 0.8, and n_estimators set to 100, attained an accuracy, precision, and recall of 74%. Among these, the Random Forest classification algorithm demonstrated the highest accuracy compared to LightGBM and XGBoost. According to Saputri et al. (2021), an accuracy of 70% is considered high; however, improving accuracy can further enhance the model's ability to classify accurately.

While this research provides significant results, there are still shortcomings and opportunities for further development. Future research could focus on increasing the use of data augmentation to improve model accuracy. Additionally, comparisons with other classification algorithms or alternative data augmentation and feature extraction methods could yield better performance. Updating the dataset to increase the quantity of data would also enhance model quality. Furthermore, adding resources by renting cloud servers could facilitate the creation of more complex models after expanding the data.

REFERENCES

[1] Basuki, A. T., & Yuliadi I. (2014). Electronic Data Processing (SPSS 15 dan EVIEWS 7) (1st ed.). Yogyakarta: Danisa Media.

[2] Shu, K., Mahudeswaran, D., Wang, S., Lee, D., & Liu, H. (2020). FakeNewsNet: A Data Repository with News Content, Social Context, and Spatiotemporal Information for Studying Fake News on Social Media. Big Data, 8(3), 171–188. https://doi.org/10.1089/big.2020.0062

[3] Demszky, D., Movshovitz-Attias, D., Ko, J., Cowen, A., Nemade, G., & Ravi, S. (2020). GoEmotions: A Dataset of Fine-Grained Emotions. http://arxiv.org/abs/2005.00547

[4] al Amrani, Y., Lazaar, M., & el Kadirp, K. E. (2018). Random forest and support vector machine based hybrid approach to sentiment analysis. Procedia Computer Science, 127, 511–520. https://doi.org/10.1016/j.procs.2018.01.150

[5] Jihad, M. A. A., Adiwijaya, & Astuti W. (2021). Analisis Sentimen Terhadap Ulasan Film Menggunakan Algoritma Random Forest. E-Proceeding of Engineering, 8(5), 10153–10165.

[6] Ahsana, R., Rohmat Saedudin, R., & Widartha, V. P. (2021). Perbandingan Akurasi Algoritma Adaboost Dan Algoritma Lightgbm Untuk Klasifikasi Penyakit Diabetes. E-Proceeding of Engineering, 9738–9748.

[7] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. (n.d.). LightGBM: A Highly Efficient Gradient Boosting Decision Tree. https://github.com/Microsoft/LightGBM

[8] Luo, S., & Chen, T. (2020). Two derivative algorithms of gradient boosting decision tree for silicon content in blast furnace system prediction. IEEE Access, 8, 196112–196122. https://doi.org/10.1109/ACCESS.2020.3034566

[9] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. (n.d.). LightGBM: A Highly Efficient Gradient Boosting Decision Tree. https://github.com/Microsoft/LightGBM.

[10] Supriya, B. N., & Akki, C. B. (2021). Sentiment Prediction Using Enhanced Xgboost And Tailored Random Forest. International Journal of Computing and Digital Systems, 10(1), 191–199.https://doi.org/10.12785/ijcds/100119

[11] Jhaveri S, Khedkar I, Kantharia Y, & Jaswal S. (2019). Success Prediction using Random Forest, CatBoost, XGBoost and AdaBoost for Kickstarter Campaigns. Proceedings of the Third International Conference on Computing Methodologies and Communication (ICCMC 2019), 1170–1173. https://doi.org/10.1109/ICCMC.2019.8819828f

[12] Muslim, I., & Karo, K. (2020). Implementasi Metode XGBoost dan Feature Importance untuk Klasifikasi pada Kebakaran Hutan dan Lahan. In Journal of Software Engineering, Information and Communication Technology (Vol. 1, Issue 1).

[13] Maarif A. (2015). Penerapan Algoritma TF-IDF untuk Pencarian Karya Ilmiah. Universitas Dian Nuswantoro.

[14] Ikegami, A., Dewa, I., Bayu, M., & Darmawan, A. (2022). Analisis Sentimen dan Pemodelan Topik Ulasan Aplikasi Noice Menggunakan XGBoost dan LDA. In JNATIA (Vol. 1, Issue 1).

[15] Daoud, E. Al. (2019). Comparison between XGBoost, LightGBM and CatBoost Using a Home Credit Dataset. World Academy of Science, Engineering and Technology International Journal of Computer and Information Engineering, 13(1).

[16] Syukron, M., Santoso, R., & Widiharih, T. (2020). Perbandingan Metode Smote Random Forest Dan Smote Xgboost Untuk Klasifikasi Tingkat Penyakit Hepatitis C Pada Imbalance Class Data. JURNAL GAUSSIAN, 9(3), 227–236. Retrieved from https://ejournal3.undip.ac.id/index.php/gaussian/

[17] Zhang, D., & Gong, Y. (2020). The Comparison of LightGBM and XGBoost Coupling Factor Analysis and Prediagnosis of Acute Liver Failure. IEEE Access. https://doi.org/10.1109/ACCESS.2020.3042848

[18] S, G., Anand, A., Vijayvargiya, A., M, P., Moorthy, V., Kumar, S., & S, H. B. S. (2022). EmoSens: Emotion Recognition based on Sensor data analysis using LightGBM. 2022 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT). https://doi.org/10.1109/CONECCT55679.2022.9865753

[19] Syukron, M., Santoso, R., & Widiharih, T. (2020). Perbandingan Metode Smote Random Forest Dan Smote Xgboost Untuk Klasifikasi Tingkat Penyakit Hepatitis C Pada Imbalance Class Data. JURNAL GAUSSIAN, 9(3), 227– 236. Retrieved from https://ejournal3.undip.ac.id/index.php/gaussian/https://doi.org/10.1109/CONECCT55679.2022.9865753

[20] Jhaveri S, Khedkar I, Kantharia Y, & Jaswal S. (2019). Success Prediction using Random Forest, CatBoost, XGBoost and AdaBoost for Kickstarter Campaigns. In Proceedings of the Third International Conference on Computing Methodologies and Communication (ICCMC 2019) (pp. 1170–1173). https://doi.org/10.1109/ICCMC.2019.8819828f

[21] Gowriswari, S., & Brindha, S. (2022, March). Hyperparameters optimization using gridsearch cross validation method for machine learning models in predicting diabetes mellitus risk. In 2022 International conference on communication, computing and Internet of Things (IC3IoT) (pp. 1-4). IEEE.