Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Hybrid Classifier for Sparse Data: Integrating Naïve Bayes, Decision Trees, and Random Forests for Enhanced Text, Image, and Anomaly Detection

MD. FAIYAZ ABDULLAH SAYEEDI[1], ANAS MOHAMMAD ISHFAQUL MUKTADIR OSMANI[2], JANNATUL FERDOUS DEEPTI[3], WASIMUL KARIM[4], AND SAYEEM BEEN ZAMAN.[5]

[1] United Internation University, United City R/A, Dhaka, 1212, Bangladesh (e-mail: msayeedi212049@bscse.uiu.ac.bd)
[2] United Internation University, United City R/A, Dhaka, 1212, Bangladesh (e-mail: aosmani203004@bscse.uiu.ac.bd)
[3] United Internation University, United City R/A, Dhaka, 1212, Bangladesh (e-mail: jdeepti212008@bscse.uiu.ac.bd)
[4] United Internation University, United City R/A, Dhaka, 1212, Bangladesh (e-mail: wasimulkarim19@gmail.com)
[5] United Internation University, United City R/A, Dhaka, 1212, Bangladesh (e-mail: sayeemzzzaman@gmail.com)

Corresponding author: Md. Faiyaz Abdullah Sayeedi (e-mail: msayeedi212049@bscse.uiu.ac.bd).

ABSTRACT Sparse data, where most features are either irrelevant or missing, presents a significant challenge in machine learning. Text Classification, Image Classification, Medical Diagnosis, and Fraud/Spam Detection are the major domains where we need to deal with sparse data. To address this, we introduce a novel methodology that combines the simplicity of Naïve Bayes with the robustness of Decision Tree and Random Forest classifiers. Our approach begins by leveraging Naïve Bayes, a probabilistic algorithm, to handle sparse data efficiently. The Naïve Bayes component captures conditional probabilities, allowing it to model the basic structure of the sparse dataset. We then integrate these probabilities as features into the original dataset, enriching the feature space. Next, we employ Decision Tree and Random Forest classifiers on the enriched dataset. Decision Tree excels in capturing intricate data patterns, while Random Forest, an ensemble learning technique, ensures model stability and accuracy. Furthermore, we introduce an Ensemble Classifier, combining the strengths of Naïve Bayes and Random Forest predictions through a weighted average. This hybrid model compensates for individual model weaknesses, resulting in a more accurate and reliable classification. Our study rigorously evaluated four novel feature-engineered algorithms alongside existing classifiers, employing comprehensive metrics like classification accuracy, precision, recall, and f1-score, along with 10-fold cross-validation on 9 benchmark datasets from Kaggle, UCI (University of California, Irvine) machine learning repository, and sklearn datasets websites. The outcomes demonstrated the exceptional performance of our proposed methods in addressing real-life sparse data challenges. These techniques not only showcased impressive classification results but also demonstrated their capability to automatically discern crucial training data and identify effective attributes within noisy and complex datasets with substantial attribute dimensions. [a]

[a] Code implementation available at: https://github.com/faiyazabdullah/ML-Research

INDEX TERMS Classification Algorithms, Data Mining, Decision Trees, Ensemble Classifiers, Feature Engineering, Naïve Bayes Classifier, Random Forest, Weighted Average, Sparse Data

I. INTRODUCTION

In recent years, the field of machine learning has witnessed an increasing need to tackle complex challenges posed by real-world datasets, especially when dealing with sparse data. Sparse data, characterized by a prevalence of irrelevant or missing features, poses a significant challenge in the realm of machine learning by making traditional classification methods less effective.

In response to this challenge, this paper presents a novel approach that leverages Naïve Bayes, a probabilistic algorithm known for its simplicity and efficiency [1], [2]. Naïve Bayes excels in handling sparse data by capturing conditional probabilities, allowing it to model the basic structure of such datasets. The conditional probabilities derived from Naïve Bayes are then integrated as additional features into the original dataset, enriching the feature space. To further enhance the classification process [3], we employ both Decision Tree and Random Forest classifiers on the enriched dataset. Decision Tree [4] is adept at capturing intricate data patterns and relationships, making it a valuable tool for handling complex datasets. Meanwhile, Random Forest [5], an ensemble learning technique, adds another layer of robustness to the model by ensuring stability and accuracy through aggregating multiple decision trees. In addition to these individual components, we introduce an Ensemble Classifier that combines the strengths of Naïve Bayes and Random Forest predictions through a weighted average [6]; [7]. This hybrid model compensates for the weaknesses of individual classifiers, resulting in a more accurate and reliable classification system [8]. It has the ability to adapt to the challenges posed by sparse data and deliver superior performance in real-world scenarios. In other words, We combine the simplicity of Naïve Bayes with the robustness of Decision Tree and Random Forest classifiers, providing a solution to the intricacies of handling sparse data. In this domain of data preprocessing, a variety of techniques are at our disposal. Among these techniques, we are considering: (a) Sparse Data Handling: Utilize Naïve Bayes for effective handling of sparse data. (b) Capture Conditional Probabilities: Use Naïve Bayes to model the dataset's structure by capturing conditional probabilities. (c) Feature Space Enrichment: Enhance the dataset by integrating these conditional probabilities. (d) Classifier Integration: Employ Decision Tree and Random Forest classifiers on the enriched dataset, combining their strengths for superior performance. (e) Hybrid Ensemble: Introduce an Ensemble Classifier combining Naïve Bayes and Random Forest predictions for improved accuracy and adaptability in real-world scenarios.

In our comprehensive evaluation, we thoroughly examined: (a) Four innovative feature-engineered algorithms: These algorithms were developed with a focus on addressing distinct challenges present in real-world datasets, such as handling sparse data, noise, and high dimensionality. (b) Established classifiers: We included widely recognized classifiers to serve as a basis for comparison and benchmarking. This approach allowed us to gauge the relative performance and effectiveness of our novel algorithms in practical scenarios. We meticulously assessed the performance using a comprehensive set of well-established confusion metrics [9], including (a) Classification Accuracy (b) Precision (c) Recall, and (d) F1-Score. The evaluation process offers a comprehensive view of our model's performance. These metrics extend beyond simple accuracy percentages to uncover the model's strengths and weaknesses. They are particularly valuable in scenarios involving class imbalances or varying costs of errors. By considering all these metrics, we aim to gain a nuanced understanding of our approach's effectiveness in real-world applications.

We conducted this evaluation using 10-fold cross-validation, a robust technique ensuring result reliability. Our assessment included benchmark datasets from Kaggle, the University of California, Irvine (UCI) machine learning repository, and sklearn datasets websites. This approach not only scrutinized the performance of our novel algorithms but also provided a valuable comparison to existing methods, resulting in a comprehensive and insightful assessment of our approach.

Our research results underscore the exceptional performance of our proposed methods in addressing the challenges posed by sparse data. These techniques not only yield impressive classification outcomes but also possess the inherent ability to automatically identify vital training data and effective attributes within noisy and complex datasets characterized by substantial attribute dimensions. This work equips machine learning with valuable tools for enhancing classification in real-world applications dealing with sparse data.

In this paper, we follow a structured approach to delve into the world of Naïve Bayes Classifier, Decision Tree, Random Forest, and Ensemble Classifier in sparse data scenarios. Section II presents a comprehensive review of related academic research, offering insights into the landscape. Sections III to IV focus on the core of our work, where we explore the basic classification techniques (Section III) and introduce innovative feature-engineered algorithms (Section IV) for each of these classifiers. Section V reports on our experiments, unveiling results and engaging in insightful discussions. Finally, in Section VI, we conclude our findings and outline promising avenues for future research, closing the narrative on our exploration.

II. RELATED WORKS

In this section, we conduct an in-depth analysis of recent academic research pertaining to the utilization of Naïve Bayes Classifier, Decision Tree, Random Forest, and Ensemble Classifier in diverse real-world sparse data scenarios.

A. NAÏVE BAYES CLASSIFIER

Naive Bayes has been reported to have good predictive performance on large and sparse datasets [3]. A. Y. Ng investigated the convergence behavior of naive bayes with regard to the number of characteristics for binary data sets generally in 2001, comparing discriminative and generative classifiers [10]. In the same year, a study was aimed to understand the data characteristics influencing the performance of the Naive Bayes classifier, employing Monte Carlo simulations to systematically investigate classification accuracy for randomly generated problems and revealing that Naive Bayes performs best with completely independent or functionally dependent features [11]. In 2009, the Multiclass Odds Ratio (MOR) and the Class Discriminating Measure (CDM) were introduced by Chen as two feature assessment metrics for the

Naïve Bayesian classifier applied on multiclass text datasets [12]. A 2011 study proposed a very straightforward, and effective hybrid approach based on C4.5 and naive Bayes [8]. It performed similarly to Naive Bayes Tree in terms of classification accuracy, but was noticeably more efficient than Naive Bayes Tree, according to the experimental results on a large number of UCI data sets. In 2014, Ruta proposed a fast feature selection method that was optimised with the Naive Bayes classifier and appeared to be particularly suitable for handling very large data sets of sparse features [13]. In 2016, Priyanga Chandrasekar showed that combining Naïve Bayes classification with the proper data pre-processing can improve the prediction accuracy and also proved that the preprocessing phase has a larger impact in the performance of the Naïve Bayes classifier especially with the reduced number of false positives [2]. In the same year, Li Showed the convergence behavior of Naive Bayes on large sparse user behavior datasets. Based on the observations, Li offered theoretical insights and practical recommendations while exploring the effects of various mechanisms of data sparsity and missingness using the generated datasets [14]. In 2017, Mykhailo Granik proposed a naive Bayes approach for fake news detection problems achieving 74% classification accuracy [7]. In 2020, a study presented a selective naïve Bayes algorithm that can perform attribute selection by model selection [1]. According to a study in 2022, Youngmi Kwon showed the results of a comparison of the accuracy and predictive error rates where their proposed model, the Hadoop MapReduce Naïve Bayes method improved the accuracy of spam and ham email identification 1.11 times and the prediction error rate 14.13 times compared to the non-Hadoop Python Naïve Bayes method [6].

B. DECISION TREE

Decision trees are one of the leading forms of intelligible models. Despite several attempts over the last several decades to enhance the optimality of decision tree algorithms, the CART and C4.5 decision tree algorithms (and other avaricious tree-growing variants) have remained predominant methods in practice. In [15], They have described two methods for speeding up machine learning algorithms like decision tree learning algorithms when the data is sparse. Hong Kuan Sok et al. [16] presented a sparse decision tree learning algorithm ADTree, that supports boosting and performs well on spectral datasets. In, [17] their authors illustrated the first practical algorithm to optimize decision trees for binary variables. They used the Binary Optimal Classification Trees (BinOCT) method, to assess the accuracy and compare it with the Optimal Sparse Decision Trees (OSDT). Sathiyanarayanan et al. [18] used the DT algorithm under the supervised learning mechanism to reveal breast cancer with a maximum accuracy of 99%. Sarker et al. [19] presented a Behavioral Decision Tree named "BehavDT", a context-aware predictive model based on user-diverse behavioral activities with smartphones. With an accuracy exceeding 90%, this model stands out as the most robust among other traditional machine learning models. DT outperforms KNN, LR, SVM, and NB on the UCI ML datasets for "Email Spam Detection" acquiring the highest accuracy of 99.93% [20]. In the same year, Xiao compared some Decision Tree methods for handling missing data with all methods exhibiting improved performance, with information on missing data enhancing the performance of certain imputation methods in sparse datasets with a missing data mechanism at random [21]. In [22], Their study illustrates how enhancing data splitting strategies and employing different feature selection methods for DSM can significantly improve a decision tree's (CART) performance. Predicting human mobility patterns, using sparse datasets demonstrated superior results, emphasizing the merit of a hybrid approach utilizing an ensemble of decision tree-based classifiers [23].

C. RANDOM FOREST

Random forest has been a prominent machine learning strategy for building prediction models since Breiman introduced it in 2001 [24]. In 2007, Xu proposed a random forest approach that showed great potentials in dealing with the data sparseness problem in language modeling [25]. In 2016, Vrushali presented a weighted hybrid decision tree model for the random forest classifier, and the approach demonstrated a significant improvement in random forest accuracy [26]. Based on applications in expert and intelligent systems, a comparison of random forest variable selection techniques was presented in 2019 by JL Speiser with 311 datasets [27]. In the same Year, Demidova showed how a hybrid model with the random forest classifier can enhance the classification results [28]. In 2020, Li introduced an intrusion detection model based on random forest feature selection, that reduced the detection time and effectively improved the prediction accuracy on sparse matrix and dense matrix [29]. In the same year, Feng proposed an improved random forest model, addressing limitations of traditional random forests by incorporating feature selection methods for enhanced data preprocessing, obtaining optimal feature subsets, and introducing sparse matrix projection with improved accuracy [30]. AH Sakib's 2022 model for estimating the price of a mobile phone combining decision tree, random forest, and hybrid ensemble learning was highly precise [31].

D. ENSEMBLE CLASSIFIER

A study in 2012 showed that ensemble classifications present elegant and powerful solutions to diverse machine learning problems, originally designed to enhance classification accuracy by mitigating variations in classifier outputs [32]. In 2014, Yin proposed a mathematical framework of a classifier ensemble with a sparsity and diversity learning strategy [33]. Research has shown that ensemble classifiers often perform better than conventional classifiers [34]. In 2020 Mienye et al. showed improved performance using ensemble learning to predict heart disease. The proposed ensemble achieved classification accuracies of 93% and 91% on the Cleveland and Framingham test sets, respectively.

Compared to other machine learning methods and recent scholarly works, the proposed approach showed improved performance [35]. The ensemble model demonstrated exceptional performance, achieving an overall accuracy of 99.88% in diagnosing COVID-19 from routine blood tests. It outperformed other classifiers, producing better outcomes [36]. In a research on ensemble learning and feature selection the authors compared seven individual classifiers to determine the most suitable base classifiers for ensemble learning, the findings indicate that Logistic Regression, Decision Trees, and Gradient Boosting stand out as the top predictive models. The ensemble model, incorporating these, achieved a final accuracy of 98.8% [37]. In 2021, Kiziloz conducted an empirical study comparing six different classifier ensemble methods, which combined predictions from five classifiers, including Naive Bayes and Decision Tree, in the context of feature selection, demonstrating that while using classifier ensembles improves accuracy, they come at the cost of slower execution compared to single classifiers [38].

III. SUPERVISED CLASSIFICATION

Classification stands as one of the fundamental techniques in data mining, empowering intelligent decision-making processes [39]. In this section, we delve into essential methods for data classification utilizing Naïve Bayes Classifier, Decision Tree, Random Forest, and Ensemble Classifier. This process unfolds in two pivotal steps:

1) **Learning Phase (Training):** In this initial phase, the classification model, or classifier, is constructed by analyzing a set of instances where the correct class labels are known. This process is analogous to a teacher learning from solved problems before guiding students through new, similar problems. The learning phase enables the classifier to discern patterns and associations within the data, establishing the foundation for accurate predictions in the future.
2) **Classification Phase:** Following the learning phase, the classification model is deployed to predict class labels for new data instances. This phase involves applying the acquired knowledge from the learning phase to make informed decisions about the class labels of previously unseen data.

Table 1 provides a summary of the key symbols and terminologies extensively used in this paper.

TABLE 1: Commnly used symbols and terms

Symbol	Description
SL	Supervised Learning
USL	Unsupervised Learning
$\mathbb{X}$	Set of training instances/training dataset
N	Size of $\mathbb{X}$
Xi	Feature/attribute in the training data
$(x^{(i)}, y^{(i)})$	i-th instance pair in $\mathbb{X}$ (SL)
$x^{(i)}$	Input of i-th training instance in $\mathbb{X}$ (USL)
$x_j^{(i)}$	Value of feature j in i-th training instance
D	Dimension of an instance $x^{(i)}$
K	Dimension of a label $y^{(i)}$
$X \in \mathbb{R}^{(N \times D)}$	Design matrix, where X_i : denotes $x^{(i)}$
F	Hypothesis space of functions to be learned
$C[f]$	Cost function of $f \in F$
$C[\theta]$	Cost function of θ parameterizing $f \in F$
(x', y')	Testing pair
$\hat{y}$	Label predicted by f, i.e., $\hat{y} = f(x')$ (SL)
$P(x, y)$	Data generating distribution/Probability
$\|CL\|$	Total number of classes in the dataset
$CL^{(i)}$	Class label
NB	Naive Bayes Classifier
DT	Decision Tree
RF	Random Forest
P	Class probabilities using NB
P^{+}	Probabilities for the positive class
X'	Enriched Dataset

A. NAÏVE BAYES CLASSIFICATION

A Naïve Bayes classifier is a probabilistic method used for predicting class membership probabilities [40]. Its simplicity and efficiency make it advantageous; it is easy to use and requires only a single scan of the training data for probability generation. The classifier adeptly handles missing attribute values by omitting corresponding probabilities, simplifying likelihood calculations. Moreover, it assumes class conditional independence: the effect of an attribute on a class is independent of other attributes.

Given a training dataset $\mathbb{X}$, where each data record $(x^{(i)}, y^{(i)})$, containing attributes Xi and classes $CL^{(i)}$, the Naïve Bayes classifier predicts the class label $CL^{(i)}$ for a test instance $x^{(i)}$ based on maximizing the posterior probability $P(CL^{(i)}|x^{(i)})$ using the Maximum Posterior Hypothesis.

$$P(CL^{(i)}|x^{(i)}) = \frac{P(x^{(i)}|CL^{(i)})P(CL^{(i)})}{P(x^{(i)})} \tag{1}$$

In the Bayes theorem shown in the equation 1, is maximized, where class prior probabilities $P(CL^{(i)})$ are calculated as $P(CL^{(i)}) = \frac{|CL^{(i)}, \mathbb{X}|}{N}$.

To reduce computation, the Naïve Bayes assumption of class conditional independence is applied, allowing efficient estimation of $P(x_k|CL^{(i)})$ for categorical and continuous attributes. For categorical attributes, $P(x_k|CL^{(i)})$ is the count of instances in $CL^{(i)}$ with value x_k divided by $|CL^{(i)}, \mathbb{X}|$. For continuous attributes, assuming a Gaussian distribution with mean $\mu_{CL^{(i)}}$ and standard deviation $\sigma_{CL^{(i)}}$, $P(x^{(i)}|y)$ is calculated as:

$$P(x^{(i)}|y) = \frac{1}{\sqrt{2\pi\sigma^2_{CL^{(i)}}}} \exp\left(-\frac{(x^{(i)} - \mu_{CL^{(i)}})^2}{2\sigma^2_{CL^{(i)}}}\right)$$

The class label of instance X is predicted as $CL^{(i)}$ if $P(X|CL^{(i)})P(CL^{(i)})$ is maximized among all classes $CL^{(i)} \in \mathbb{X}$. Table 3 illustrates the prior probabilities. Tables 4 and 5 illustrate the conditional probabilities. All the probabilities are generated using the playing tennis dataset in Table 2.

TABLE 2: Example dataset for playing tennis.

Outlook	Temperature	Humidity	Wind	Play
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rain	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rain	Mild	High	Strong	No

TABLE 3: Prior probabilities for each class.

Class	Prior Probability
No	5/14
Yes	9/14

TABLE 4: Conditional probabilities for each attribute value (Play = Yes).

Attribute	Value	Conditional Probability
Outlook	Sunny	P(Outlook=Sunny|Yes) = 2/9
	Overcast	P(Outlook=Overcast|Yes) = 4/9
	Rain	P(Outlook=Rain|Yes) = 3/9
Temperature	Hot	P(Temperature=Hot|Yes) = 2/9
	Mild	P(Temperature=Mild|Yes) = 4/9
	Cool	P(Temperature=Cool|Yes) = 3/9
Humidity	High	P(Humidity=High|Yes) = 3/9
	Normal	P(Humidity=Normal|Yes) = 6/9
Wind	Weak	P(Wind=Weak|Yes) = 6/9
	Strong	P(Wind=Strong|Yes) = 3/9

TABLE 5: Conditional probabilities for each attribute value (Play = No).

Attribute	Value	Conditional Probability
Outlook	Sunny	P(Outlook=Sunny|No) = 3/5
	Overcast	P(Outlook=Overcast|No) = 0/5
	Rain	P(Outlook=Rain|No) = 2/5
Temperature	Hot	P(Temperature=Hot|No) = 2/5
	Mild	P(Temperature=Mild|No) = 2/5
	Cool	P(Temperature=Cool|No) = 1/5
Humidity	High	P(Humidity=High|No) = 4/5
	Normal	P(Humidity=Normal|No) = 1/5
Wind	Weak	P(Wind=Weak|No) = 2/5
	Strong	P(Wind=Strong|No) = 3/5

B. DECISION TREE INDUCTION

A decision tree classifier operates as a top-down greedy approach, offering an effective method to classify data instances [4]. Each Decision Tree (DT) functions as a set of rules, recursively partitioning the training datasets until each subset exclusively belongs to one class. ID3 (Iterative Dichotomiser) stands out as a widely utilized DT algorithm, employing information theory for attribute selection [41].

The root node selection relies on the highest information gain of the attribute. Given a training dataset $\mathbb{X}$, the expected information to classify an instance $(x^{(i)}, y^{(i)}) \in \mathbb{X}$ correctly, is calculated as in Equation 2, where $P(y^{(i)})$ represents the probability of $(x^{(i)}, y^{(i)}) \in \mathbb{X}$ belonging to class $CL^{(i)}$, estimated by $\frac{|CL^{(i)},\mathbb{X}|}{|\mathbb{X}|}$.

$$\text{Info}(\mathbb{X}) = -\sum_{i=1}^{N} P(y^{(i)}) \cdot \log_2 P(y^{(i)}) \qquad (2)$$

In Equation 2, Info($\mathbb{X}$) represents the average information required to identify $CL^{(i)}$ of an instance $(x^{(i)}, y^{(i)}) \in \mathbb{X}$. The objective of DT is to partition $\mathbb{X}$ iteratively into subsets $\{\mathbb{X}_1, \mathbb{X}_2, \ldots, \mathbb{X}_N\}$, where all instances in each $\mathbb{X}$ belong to the same class $CL^{(i)}$. Info($\mathbb{X}_{Xi}$) denotes the expected information needed to correctly classify an instance $(x^{(i)}, y^{(i)})$ from $\mathbb{X}$ based on the partitioning by attributes X. Equation 3 shows how Info($\mathbb{X}_{Xi}$) is calculated, where $\frac{|\mathbb{X}_j|}{N}$ acts as the weight for the jth partition.

$$\text{Info}(\mathbb{X}_{Xi}) = -\sum_{j=1}^{K} \frac{|\mathbb{X}_{Xi}^{j}|}{N} \cdot \text{Info}(\mathbb{X}_{Xi}^{j}) \qquad (3)$$

Information gain represents the difference between the original and new information requirements, as defined in Equation 4.

$$\text{Gain}(\mathbb{X}_{Xi}) = \text{Info}(\mathbb{X}) - \text{Info}(\mathbb{X}_{Xi}) \qquad (4)$$

The Gain Ratio, an extension of the information gain approach, is utilized in DT, such as C4.5 [41]. A C4.5 classifier is the successor of the ID3 classifier and applies a form of normalization to information gain, employing a "split information" value defined analogously to Info($\mathbb{X}$), as shown in Equation 5.

$$\text{SplitInfo}(\mathbb{X}_{Xi}) = -\sum_{j=1}^{K} \frac{|\mathbb{X}_{Xi}^{j}|}{N} \cdot \log_2 \frac{|\mathbb{X}_{Xi}^{j}|}{N} \qquad (5)$$

The attribute with the maximum Gain Ratio is chosen as the splitting attribute, as defined in Equation 6.

$$\text{GainRatio}(\mathbb{X}_{Xi}) = \frac{\text{Gain}(\mathbb{X}_{Xi})}{\text{SplitInfo}(\mathbb{X}_{Xi})} \qquad (6)$$

The Gini Index is used in Classification and Regression Trees (CART) algorithm, which generates the binary classification tree for decision-making. It measures the impurity of $\mathbb{X}$. Equation 7 defines the Gini index for a dataset $\mathbb{X}$, where P_j represents the frequency of class $CL^{(i)} \in \mathbb{X}$.

$$\text{Gini}(\mathbb{X}_{Xi}) = 1 - \sum_{j=1}^{K} (P(y^{(i)}|x_{Xi}^{j}))^2 \qquad (7)$$

The quality of the split of $\mathbb{X}$ into subsets $\mathbb{X}_1$ and $\mathbb{X}_2$ is determined by Equation 8.

$$\text{GiniSplit}(\mathbb{X}_{Xi}) = \frac{N_1}{N} \cdot (\text{Gini}(\mathbb{X}_1)) + \frac{N_2}{N} \cdot (\text{Gini}(\mathbb{X}_2)) \quad (8)$$

In this way, the split with the best Gini index is chosen. Now, for each Xi, each of the binary splits is considered. The Xi that maximizes the reduction in impurity is selected as the splitting feature, shown in Equation 9

$$\Delta\text{Gini}(\mathbb{X}_{Xi}) = \text{Gini}(\mathbb{X}) - \text{GiniSplit}(\mathbb{X}_{Xi}) \tag{9}$$

To illustrate the operation of DT, consider a small dataset in Table 6, described by four attributes: Outlook, Temperature, Humidity, and Wind, representing the weather condition of a specific day. Each attribute has unique values. The 'Play' column represents the class category of each instance, indicating whether a specific weather condition is suitable for playing tennis. Figure 1 shows the decision tree model constructed using the playing tennis dataset in Table 6 [39].

TABLE 6: Example dataset for playing tennis.

Day	Outlook	Temperature	Humidity	Wind	Play
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

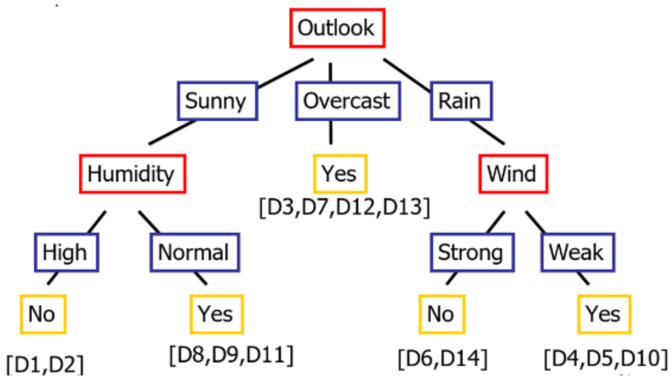

FIGURE 1: Decision tree model for the playing tennis dataset.

C. RANDOM FOREST CLASSIFICATION

Random Forest is an ensemble learning method based on decision trees, aiming to improve the accuracy and robustness of classification tasks [42]. It operates by creating a multitude of decision trees during training and outputting the class that is the mode of the classes (classification) or the mean prediction (regression) of the individual trees [43]. Each tree is constructed using a subset of the dataset and a subset of the features, injecting variability into the model. The Random Forest classifier combines these individual predictions to produce a more accurate and stable prediction. Two common methods used in Random Forest classification are voting and averaging [5].

1) Voting

In the voting method, each decision tree in the Random Forest independently predicts the class label for a given input instance. The final prediction is determined by a majority vote among the individual tree predictions. Specifically, let $DT_1, DT_2, ..., DT_n$ represent the individual decision trees in the Random Forest. Each tree provides a class prediction $CL^{(i)}$ for a given input $x^{(i)}$. The class label CL with the highest number of votes among all individual trees is considered the final prediction for $x^{(i)}$. Mathematically shown in Equation 10,

$$CL = \text{argmax}_{CL^{(i)}} \sum_{j=1}^{K} [CL^{(i)} = CL] \tag{10}$$

where $[CL^{(i)} = CL]$ is the indicator function that evaluates to 1 if $CL^{(i)} = CL$ and 0 otherwise. CL is the predicted class label for the input instance $x^{(i)}$.

2) Averaging

In the averaging method, each decision tree in the Random Forest independently predicts a numerical value (in regression tasks) for a given input instance. The final prediction is the average of these individual tree predictions. For a set of decision trees $DT_1, DT_2, ..., DT_n$, each providing a numerical prediction $y^{(i)}$ for the input $x^{(i)}$, the final prediction y is calculated as the mean of all individual predictions shown in Equation 11,

$$y = \frac{1}{N} \sum_{i=1}^{N} y_i \tag{11}$$

where y represents the final numerical prediction for the input instance $x^{(i)}$.

Random Forest's ability to handle high-dimensional data, capture complex relationships, and mitigate overfitting makes it a popular choice for various classification tasks.

D. ENSEMBLE LEARNING WITH WEIGHTED AVERAGE

Ensemble Learning combines the predictions of multiple classifiers, leveraging their diverse strengths to enhance overall accuracy. In this research, the Ensemble Learning model is created by employing a weighted average of predictions from Naïve Bayes (NB) and Random Forest (RF) classifiers. Each classifier contributes to the final prediction based on its reliability and performance.

Let $P_{NB}(CL^{(i)}|x^{(i)})$ and $P_{RF}(CL^{(i)}|x^{(i)})$ represent the class probabilities predicted by the Naïve Bayes and Random Forest classifiers, respectively, for a given input instance $x^{(i)}$. The final prediction $P_{\text{ensemble}}(CL^{(i)}|x^{(i)})$ is calculated using a weighted average shown in Equation 12:

where α is the weight assigned to the Naïve Bayes classifier, ensuring a balanced contribution from both classifiers. Adjusting the value of α allows fine-tuning of the influence of each classifier on the final prediction.

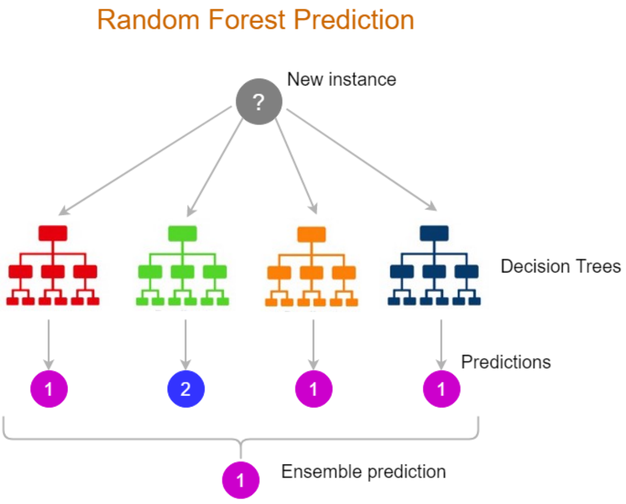

FIGURE 2: Random Forest classifier combining individual decision trees

IV. THE PROPOSED FEATURE ENGINEERED LEARNING AND CLASSIFICATION ALGORITHMS

In this section, we introduce four distinct feature-engineered algorithms designed to enhance the classification accuracy in sparse data learning tasks, focusing on Naïve Bayes Classifier, Decision Tree, Random Forest, and Ensemble Classifier. The proposed algorithms are described in the following Sections IV-A, IV-B, IV-C, and IV-D.

A. THE PROPOSED FEATURE ENGINEERED NAÏVE BAYES CLASSIFIER

The proposed feature engineered Naive Bayes classifier incorporates additional features derived from probabilistic predictions. These features enhance the model's ability to capture intricate patterns in the data, especially in sparse datasets. The algorithm leverages both traditional Naive Bayes techniques and feature engineering to achieve improved classification accuracy.

Algorithm 1 Proposed Feature Engineered Naive Bayes Classifier

```
Require: Original sparse features: $X$
Ensure: Predicted labels: $\hat{y}$
    Train NB on $X$
    $P$ = Obtain class probabilities using NB for training and test data
    $P^+$ = Extract probabilities for the positive class
    $X'$ = Combine $P^+$ with $X$ as a new feature
    Train NB on $X'$
    $\hat{y}$ = Predict using NB for $X'$
    return $\hat{y}$ =0
```

B. THE PROPOSED FEATURE ENGINEERED DECISION TREE

The proposed feature engineered Decision Tree algorithm enhances the traditional Decision Tree model by incorporating enriched features. These features are engineered using probabilistic predictions, providing a more comprehensive representation of the input data. This approach aims to capture complex relationships in sparse datasets, leading to improved classification performance.

Algorithm 2 Proposed Feature Engineered Decision Tree

```
Require: Original sparse features: $X$
Ensure: Predicted labels: $\hat{y}$
    Train NB on $X$
    $P$ = Obtain class probabilities using NB for training and test data
    $P^+$ = Extract probabilities for the positive class
    $X'$ = Combine $P^+$ with $X$ as a new feature
    Train DT on the $X'$
    $\hat{y}$ = Predict using DT for $X'$
    return $\hat{y}$ =0
```

C. THE PROPOSED FEATURE ENGINEERED RANDOM FOREST

The proposed feature engineered Random Forest algorithm integrates enriched features obtained through probabilistic predictions. By combining the strengths of Random Forest with feature engineering, the model gains the ability to discern intricate patterns in sparse datasets. This hybrid approach enhances the predictive accuracy and robustness of the Random Forest classifier.

Algorithm 3 Proposed Feature Engineered Random Forest

```
Require: Original sparse features: $X$
Ensure: Predicted labels: $\hat{y}$
    Train NB on $X$
    $P$ = Obtain class probabilities using NB for training and test data
    $P^+$ = Extract probabilities for the positive class
    $X'$ = Combine $P^+$ with $X$ as a new feature
    Train RF on $X'$
    $\hat{y}$ = Predict using RF for $X'$
    return $\hat{y}$ =0
```

D. THE PROPOSED FEATURE ENGINEERED ENSEMBLE CLASSIFIER

The proposed feature engineered Ensemble Classifier combines the strengths of Naive Bayes and Random Forest models. By integrating probabilistic predictions from Naive Bayes with Random Forest's robustness, the ensemble approach achieves superior classification performance. Weighted averaging of predictions ensures that the model adapts dynamically to varying confidence levels, resulting in an effective and interpretable ensemble classifier.

$$P_{\text{ensemble}}(CL^{(i)}|x^{(i)}) = \alpha \times P_{NB}(CL^{(i)}|x^{(i)}) + (1-\alpha) \times P_{RF}(CL^{(i)}|x^{(i)}) \tag{12}$$

Algorithm 4 Proposed Feature Engineered Ensemble Classifier

```
Require: Original sparse features: $X$
Ensure: Predicted labels: $\hat{y}$
1: Train NB on $X$
2: $P$ = Obtain class probabilities using NB for training and test data
3: $P^+$ = Extract probabilities for the positive class
4: $X'$ = Combine $P^+$ with $X$ as a new feature
5: $\hat{y}$ = Combine the NB and RF predictions using weighted average
6: return $\hat{y}$ =0
```

V. EXPERIMENTS

In this section, we discussed our chosen datasets, experimental setup, and result analysis.

A. DATASET PREPARATION

In this section, we present an overview of our dataset and detail the preprocessing steps employed to enhance its quality and suitability for analysis.

1) Description of the dataset

The datasets utilized in this study cover a wide array of applications, showcasing the versatility of the proposed techniques. For evaluating the proposed algorithms, we have selected 9 datasets. All of these datasets were collected from the Kaggle, UCI (University of California, Irvine) machine learning repository, and sklearn datasets websites.

The SMS Spam dataset comprises a collection of text messages, each labeled as spam or ham (non-spam). Ling Spam dataset consists of email messages categorized into spam and ham. The Sentiment140 dataset encompasses tweets, each labeled with positive or negative sentiments. In the Breast Cancer dataset, features extracted from breast mass images aid in classifying malignancy. The Load Digits dataset comprises images of handwritten digits, serving as a benchmark for digit recognition tasks. The Fashion MNIST dataset contains 28 by 28 pixel images of different fashion items and has 60,000 examples. It was created to replace the popular MNIST dataset. The BBC news dataset is made up of 1490 news articles along with their respective categories. Similarly, the AG news dataset has 120,000 articles categorized into 4 classes. The hate speech and offensive language dataset consists of tweets labelled as either hate-speech or offensive or neither of them. A summary of the datasets is depicted in Table 7.

2) Data pre-processing

1) **Data Cleaning:** Data cleaning is a vital step to ensure the integrity of the datasets. Irrelevant characters and special symbols were removed from the text data, ensuring that only relevant information was retained. To prepare the dataset for training the model, datasets that contained nominal attributes were cleaned by removing stop-words (i.e. common English words that carry negligible weight in describing the label).

TABLE 7: Dataset Descriptions

Datasets	No. of Att.	Att. Types	Instances	Classes
SMS Spam	10	Categorical	5,572	2
Email Spam	113	Categorical	4,000	2
Sentiment140	1	Categorical	1,600,000	2
Breast Cancer	30	Real	569	2
Load Digit	64	Real	1,797	10
Fashion MNIST	784	Real	60,000	10
BBC News	2,225	Categorical	1490	5
AG News	1,296	Catagorical	120,000	4
Hate speech	3072	Real	10,000	3

2) **Tokenization and Feature Extraction:** Tokenization is the process of breaking down text into smaller units like words or subwords, which was applied to textual datasets (SMS Spam, Ling Spam, Sentiment140, BBC News, AG News, and Hate Speech and Offensive Language) to convert the nominal training data into numerical data in the form of matrices. Tokenized words were then transformed into numerical features using techniques such as Count Vectorization or TF-IDF (Term Frequency-Inverse Document Frequency). This would allow various forms of mathematical operations to be conducted that are required during the training of the model. In the case of image datasets (Load Digits), images were transformed into numerical arrays, enabling their use in machine learning algorithms.

3) **Label Encoding:** Label encoding was performed to convert categorical labels into numerical format. This conversion is essential for machine learning algorithms, as most models require numerical input. Categorical labels in the datasets were encoded into distinct numerical values, ensuring compatibility with various classification algorithms.

B. EXPERIMENTAL SETUP

In this section, we discuss our hardware and software environment and the evaluation matrices we used for the validation of our experiment.

1) Hardware and Software Environment

We have used Google Colaboratory's free version for conducting most of our experiments. It is a cloud-based Python editor that allows multiple users to access the same codebase where each user can have varying privileges (viewer or editor). For some of the experiments that demanded more computational resources, the code was implemented in a Linux-based laptop having a RAM of 16 GB, CPU Ryzen

7 5800U, and a dedicated NVIDIA GeForce RTX 3050 Ti mobile GPU. Both in Google Colab and in the local setup, the Interactive Python Notebook format was used for better annotations of the code compared to more commonly used inline comments and to run it in a step-wise manner using multiple cells.

2) Experiment outine

Firstly, the traditional Naive Bayes, Decision Tree, and Random forest algorithms are trained with the non-feature-engineered datasets. In addition to these existing algorithms, we also trained an Ensemble Learning Classifier. Then, their corresponding accuracy matrices (discussed in the evaluation matrices section below) were calculated. This set of experiments later served as the baseline accuracy that was targetted by our proposed algorithms to prove as the better alternatives.

Secondly, the selected datasets were feature-engineered by incorporating the Baysian probabilities extracted from the previously trained Naive Bayes Classifier. These enriched datasets were used for training a new set of models and will be referred to as the "processed" version of the algorithm. So, in total 8 models were trained(4 with non-enhanced dataset and 4 with enhanced dataset). Following the training of these models, their corresponding accuracy matrices were also calculated.

Finally, for better visualization of the model's performance graphs showing the accuracy across the different datasets were plotted for each classifier. This aids us in conducting a rigorous analysis of the consistency of the proposed algorithms across various classification objectives.

3) Evaluation matrices

To assess the effectiveness of the developed hybrid techniques, we employed rigorous evaluation metrics including classification accuracy, precision, recall, F1-Score, and conducted a thorough 10-fold cross-validation.

1) **Accuracy:** This fundamental metric measures the overall correctness of predictions. It quantifies the percentage of correctly classified instances over the total instances evaluated. In other words, it answers the question: "How many of the predictions were correct?" High accuracy is generally desired, but it may not be sufficient on its own, particularly when dealing with imbalanced datasets, where one class significantly outnumbers the others. The classification accuracy is measured by Equation 13.

$$\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{13}$$

2) **Precision:** Precision is crucial, especially when we want to ensure the accuracy of positive predictions. It quantifies the ratio of true positives (correctly predicted positive instances) to the sum of true positives and false positives (incorrectly predicted positive instances). In practical terms, precision tells us how many of the positive predictions made by the model were correct. High precision is vital in situations where false positives are costly or have significant consequences. Precision is measured by Equation 14.

$$\text{precision} = \frac{TP}{TP + FP} \tag{14}$$

3) **Recall:** Also known as Sensitivity, Recall indicates the model's ability to identify true positives. It's calculated as the ratio of true positives to the sum of true positives and false negatives (positive instances that were incorrectly classified as negative). High recall is valuable when the cost of missing positive instances (false negatives) is high, and we want to minimize such errors. Recall is measured by Equation 15.

$$\text{Recall} = \frac{TP}{TP + FN} \tag{15}$$

4) **F1-Score:** The F1-Score is a measure that strikes a balance between precision and recall. It considers both false positives and false negatives and calculates a harmonic mean between precision and recall. The F1-Score provides a comprehensive evaluation of the model's ability to balance between making accurate positive predictions and not missing any positive instances. It is particularly useful when there's a need to find an optimal trade-off between precision and recall. The F1-Score is measured by Equation 16.

$$F1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{16}$$

Here, TP, TN, FP, and FN respectively denote true positives, true negatives, false positives, and false negatives.

In cross-validation, the dataset is meticulously divided into k distinct subsets or 'folds'. Each fold is consecutively used as the test set while the remaining partitions collectively form the training data. The process is repeated k times, ensuring a comprehensive evaluation of the model's performance across different subsets of data. Specifically for our experiment, 10-fold cross-validation divides the data into ten sets, training the classifier on nine sets and testing it on the tenth. This iterative process culminates in a mean accuracy rate, providing a robust estimate of the model's predictive power and generalizability across the entire dataset. This approach not only validates the model's efficacy but also safeguards against overfitting, ensuring its reliability in real-world applications.

C. RESULTS AND DISCUSSION

In this section, we present the results of our experiments on different datasets using various classifiers. We introduce novel feature engineering techniques to enhance the classification performance of standard classifiers.

TABLE 8: Classification Accuracies of Classifiers using Training Datasets

Training Datasets	**Non-enhanced NB (%)**	**Non-enhanced DT (%)**	**Non-enhanced RF (%)**	**Non-enhanced Ensemble(%)**	**Processed NB (%)**	**Processed DT (%)**	**Processed RF(%)**	**Processed Ensemble(%)**
SMS Spam	99.4	100	100	99.8	99.5	100	100	100
Ling Spam	89.8	100	100	99.9	92.4	100	100	100
Sentiment140	97.1	80.9	100	100	86.8	91.8	100	100
Breast Cancer	88.5	100	100	88.6	88.5	100	100	88.6
Load Digits	90.5	100	100	100	90.5	100	100	90.6
Fashion MNIST	66.6	99.8	100	95.5	66.8	100	100	96.5
BBC News	99.4	100	100	99.8	99.6	100	100	100
AG News	94.2	100	100	99.5	94.3	100	100	100
Hate Speech and Offensive Language	98.7	100	100	99.3	100	100	100	100

1) Classification Accuracies of Classifiers using Training Datasets

Table 8 provides an overview of the classification accuracies of classifiers using training datasets. The existing classifiers include Naive Bayes (NB), Decision Tree (DT), Random Forest (RF), and our novel Ensemble Learning combining NB and RF. Those non-enhanced methods are denoted as "non-enhanced NB", "non-enhanced DT", and "non-enhanced RF", and "non-enhanced ensemble" consistently.

Our proposed feature engineering techniques involve enriching the dataset with additional information derived from the classifiers. Those enhanced algorithms are denoted as "Processed NB", "Processed DT", "Processed RF", and "Processed Ensemble" where Naive Bayes probabilities are integrated as features. This additional information enhances the classifiers' ability to capture intricate patterns in the data and outperform their non-enhanced counterparts.

2) Performance Metrics and Analysis

Tables 9, 10, 11, 12 and Figures 3, 4, 6, 7 illustrate the performance metrics for the Naive Bayes (NB), Decision Tree (DT), and Random Forest (RF) classifiers, and the Ensemble Learning Classifier, respectively. Comparing the accuracy, precision, recall, and F1-score values of the existing classifiers with our proposed methods, it is evident that the proposed techniques consistently yield higher accuracy and better overall performance.

3) Analysis of Results

Our proposed feature engineering techniques, involving the integration of classifier probabilities, provide the models with richer information, enabling them to make more informed decisions. This additional data allows classifiers to capture

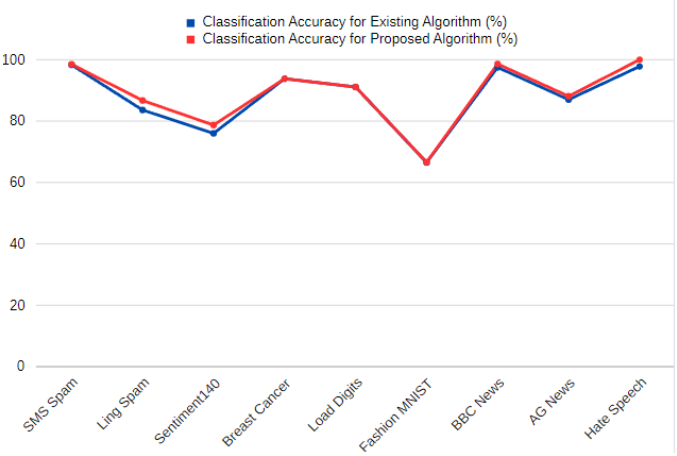

FIGURE 3: Performance Line Graph for NB Classifier

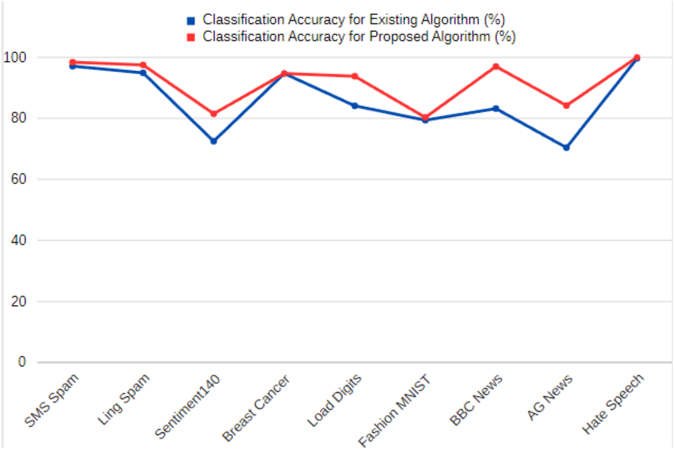

FIGURE 4: Performance Line Graph for DT Classifier

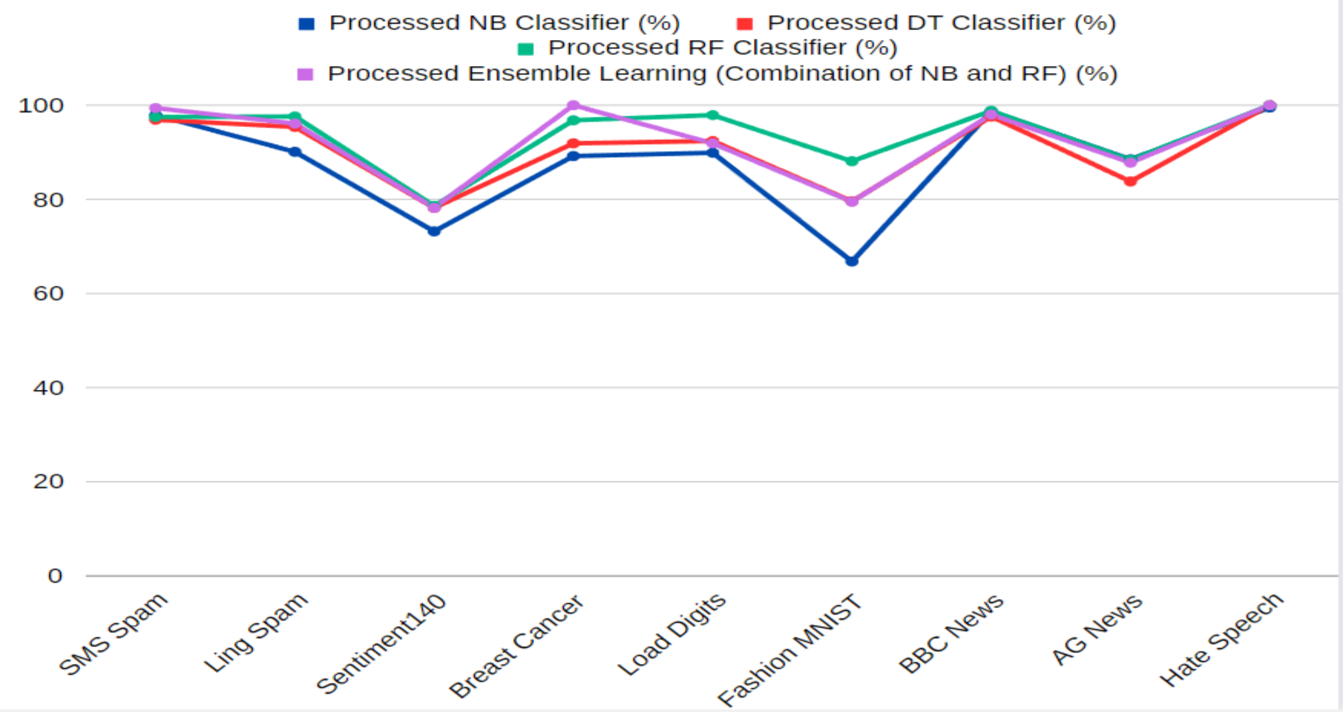

FIGURE 5: Performance Line Graph for Classifiers with 10-fold Cross Validation

TABLE 9: Performance Metrics for NB Classifier

Datasets	Accuracy (Non-enhanced) (%)	Accuracy (Proposed) (%)	Pre-cision	Recall	F1-Score
SMS Spam	98.3	98.5	0.98	0.96	0.97
Ling Spam	83.6	86.7	0.89	0.87	0.84
Sentiment 140	76	78.7	0.83	0.79	0.7
Breast Cancer	93.8	93.8	0.96	0.92	0.93
Load Digits	91.1	91.1	0.92	0.91	0.91
Fashion MNIST	66.5	66.6	0.67	0.67	0.64
BBC News	97.5	98.6	0.98	0.98	0.98
AG News	87	88.1	0.87	0.89	0.88
Hate Speech	97.8	100	1.00	1.00	1.00

TABLE 10: Performance Metrics for DT Classifier

Datasets	Accuracy (Non-enhanced) (%)	Accuracy (Proposed) (%)	Pre-cision	Recall	F1-Score
SMS Spam	97.1	98.4	0.99	0.95	0.97
Ling Spam	94.9	97.5	0.96	0.97	0.96
Sentiment 140	72.5	81.5	0.8	0.81	0.79
Breast Cancer	94.7	94.7	0.94	0.94	0.94
Load Digits	84.1	93.8	0.94	0.94	0.94
Fashion MNIST	79.4	80.3	0.8	0.8	0.8
BBC News	83.2	97	0.97	0.95	0.96
AG News	70.4	84.2	0.85	0.84	0.84
Hate Speech	99.6	100	1.00	1.00	1.00

complex patterns within the datasets, leading to improved accuracy and performance metrics.

Comparing the existing classifiers with our proposed methods, it is evident that the proposed techniques consistently outperform the traditional approaches. In particular, the ensemble learning approach, which combines Naive Bayes and Random Forest using the enriched features, demonstrates remarkable results across various datasets. This enhanced ensemble model showcases the potential of hybrid models in significantly improving classification outcomes.

Table 13 and Figure 5 provides the classification accuracies for classifiers with 10-fold cross-validation using the proposed feature engineering techniques. The results demonstrate the consistency and effectiveness of our proposed methods, indicating their reliability even under rigorous cross-validation settings.

TABLE 11: Performance Metrics for RF Classifier

Datasets	Accuracy (Non-enhanced) (%)	Accuracy (Proposed) (%)	Precision	Recall	F1-Score
SMS Spam	97.6	98.6	0.99	0.96	0.97
Ling Spam	97.7	98.4	0.99	0.96	0.97
Sentiment 140	76.5	82.2	0.81	0.82	0.8
Breast Cancer	96.4	96.5	0.97	0.96	0.96
Load Digits	97.2	98	0.95	0.95	0.95
Fashion MNIST	88.2	89.7	0.88	0.88	0.88
BBC News	96.6	98.3	0.98	0.97	0.98
AG News	82.8	87.5	0.87	0.86	0.88
Hate Speech	100	100	1.00	1.00	1.00

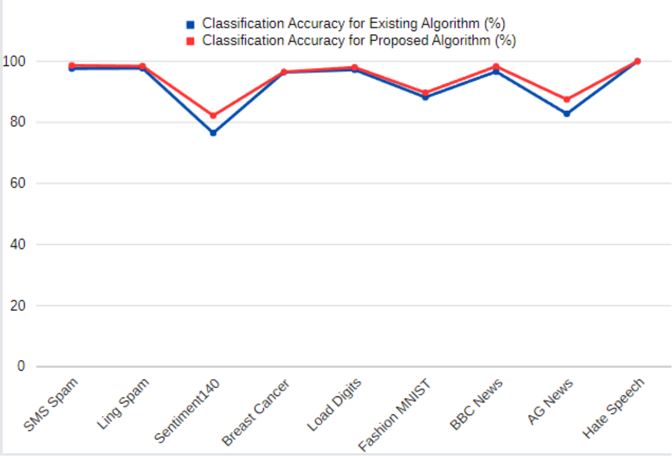

FIGURE 6: Performance Line Graph for RF Classifier

TABLE 12: Performance Metrics for Ensemble Learning Classifier

Datasets	Accuracy (Non-enhanced) (%)	Accuracy (Proposed) (%)	Precision	Recall	F1-Score
SMS Spam	98.5	98.7	0.99	0.96	0.97
Ling Spam	95.1	98.5	0.99	0.96	0.97
Sentiment 140	76.6	82.3	0.81	0.82	0.8
Breast Cancer	96.4	97.8	0.96	0.92	0.93
Load Digits	93.1	95.8	0.92	0.91	0.91
Fashion MNIST	79.6	86.7	0.86	0.87	0.86
BBC News	96.7	98.4	0.97	0.98	0.98
AG News	83.22	87.68	0.89	0.86	0.87
Hate Speech	100	100	1.00	1.00	1.00

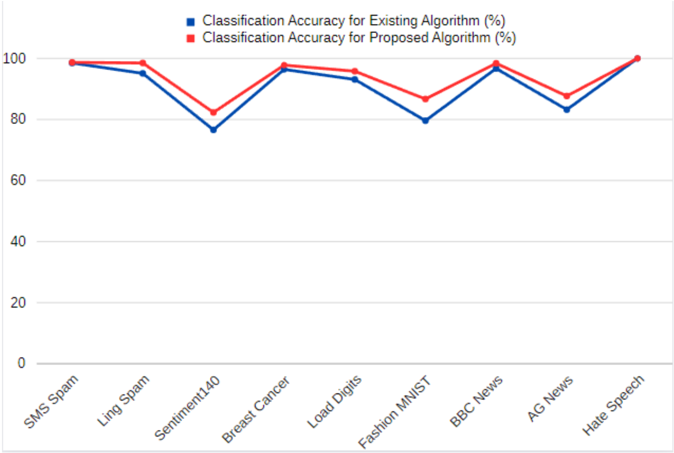

FIGURE 7: Performance Line Graph for Ensemble Learning Classifier

The high accuracy and performance across different datasets underscore the adaptability and strength of the proposed techniques. By integrating classifier probabilities as features, our approach enhances the models' ability to capture nuanced patterns in the data, leading to superior classification outcomes.

In summary, our experiments and analysis highlight the effectiveness of the proposed feature engineering techniques, demonstrating their superiority over existing methods. These techniques are particularly valuable in scenarios where accurate classification is crucial, such as spam detection, sentiment analysis, and medical diagnosis.

VI. CONCLUSION AND FUTURE DIRECTION

In this study, we delved into the challenges posed by sparse data in the context of classification tasks. Sparse data, characterized by a large number of features with limited occurrences, is a common phenomenon in various fields, including natural language processing, bioinformatics, and image recognition. Addressing the complexities associated with sparse datasets, we explored innovative strategies and techniques aimed at improving classification accuracy and robustness. Our key insights and contributions are - (a) Feature Engineering for Sparse Data: our study emphasized the power of advanced feature engineering methods like TF-IDF (Term Frequency-Inverse Document Frequency) and word embeddings. By transforming raw data into meaningful representations, we minimized the impact of high-dimensional sparse features. (b) Balancing Imbalanced Classes: Dealing with class imbalance in sparse datasets, we employed techniques like oversampling and class-weighted training. This ensured our classifiers didn't favor majority classes,

TABLE 13: Classification Accuracies for Classifiers with 10-fold Cross Validation

Datasets	Processed NB Classifier (%)	Processed DT Classifier (%)	Processed RF Classifier (%)	Processed Ensemble Learning (%)
SMS Spam	97.9	96.9	97.5	99.4
Ling Spam	90.1	95.4	97.6	96.1
Sentiment140	73.2	78.2	78.6	78.2
Breast Cancer	89.2	91.9	96.8	100
Load Digits	89.9	92.4	97.9	91.9
Fashion MNIST	66.8	79.6	88.1	79.5
BBC News	98.7	97.6	98.8	98
AG News	88.5	83.8	88.4	87.8
Hate Speech and Offensive Language	99.5	100	100	100

leading to more accurate predictions, especially for minority classes. (c) Robust Evaluation Methods: Rigorous confusion matrix and cross-validation ensured our models were robust. Besides, the implications and future directions of our study are - (a) Future Prospects with Deep Learning: While traditional methods showed promising results, the exploration of customized neural networks for sparse inputs holds potential. Tailored deep learning architectures and attention mechanisms could further enhance accuracy and efficiency in sparse data handling. (b) Optimization for feature engineering with Big Data: Our proposed algorithm has shown limitations in regards to creating the enhanced dataset from Big Data which includes the Baysian probabilities as features. So, there remains an avenue for developing more computation-efficient algorithms that deal with Big Data. (c) Interdisciplinary Collaboration: Collaborating with domain experts is key. Their insights guide feature selection, ensuring engineered features align with the data's context. This interdisciplinary approach tailors solutions to specific sparse data challenges in diverse applications.

Our research showcases significant strides in handling sparse data challenges. By combining innovative techniques, advanced algorithms, regularization, and interdisciplinary collaboration, we've laid a foundation for more precise and reliable classification models. These advancements not only enhance current decision-making processes but also pave the way for future innovations in data analysis.

ACKNOWLEDGEMENTS

REFERENCES

[1] S. Chen, G. I. Webb, L. Liu, and X. Ma, "A novel selective naïve bayes algorithm," Knowledge-Based Systems, vol. 192, p. 105361, 2020.

[2] P. Chandrasekar and K. Qian, "The impact of data preprocessing on the performance of a naive bayes classifier," in 2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC), vol. 2, pp. 618–619, IEEE, 2016.

[3] M. Sahami, "Learning limited dependence bayesian classifiers.," in KDD, vol. 96, pp. 335–338, 1996.

[4] J. R. Quinlan, "Induction of decision trees," Machine learning, vol. 1, pp. 81–106, 1986.

[5] A. Tsymbal, M. Pechenizkiy, and P. Cunningham, "Dynamic integration with random forests," in Machine Learning: ECML 2006: 17th European Conference on Machine Learning Berlin, Germany, September 18-22, 2006 Proceedings 17, pp. 801–808, Springer, 2006.

[6] K. Ji and Y. Kwon, "New spam filtering method with hadoop tuning-based mapreduce naïve bayes.," Computer Systems Science & Engineering, vol. 45, no. 1, 2023.

[7] M. Granik and V. Mesyura, "Fake news detection using naive bayes classifier," in 2017 IEEE first Ukraine conference on electrical and computer engineering (UKRCON), pp. 900–903, IEEE, 2017.

[8] L. Jiang and C. Li, "Scaling up the accuracy of decision-tree classifiers: A naive-bayes combination.," J. Comput., vol. 6, no. 7, pp. 1325–1331, 2011.

[9] B. P. Salmon, W. Kleynhans, C. P. Schwegmann, and J. C. Olivier, "Proper comparison among methods using a confusion matrix," in 2015 IEEE International geoscience and remote sensing symposium (IGARSS), pp. 3057–3060, IEEE, 2015.

[10] A. Ng and M. Jordan, "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes," Advances in neural information processing systems, vol. 14, 2001.

[11] I. Rish et al., "An empirical study of the naive bayes classifier," in IJCAI 2001 workshop on empirical methods in artificial intelligence, vol. 3, pp. 41–46, 2001.

[12] J. Chen, H. Huang, S. Tian, and Y. Qu, "Feature selection for text classification with naïve bayes," Expert Systems with Applications, vol. 36, no. 3, pp. 5432–5435, 2009.

[13] D. Ruta, "Robust method of sparse feature selection for multi-label classification with naive bayes," in 2014 Federated Conference on Computer Science and Information Systems, pp. 375–380, IEEE, 2014.

[14] X. Li, C. X. Ling, and H. Wang, "The convergence behavior of naive bayes on large sparse datasets," ACM Transactions on Knowledge Discovery from Data (TKDD), vol. 11, no. 1, pp. 1–24, 2016.

[15] D. M. Chickering and D. Heckerman, "Fast learning from sparse data," arXiv preprint arXiv:1301.6685, 2013.

[16] H. K. Sok, M. P.-L. Ooi, and Y. C. Kuang, "Sparse alternating decision tree," Pattern Recognition Letters, vol. 60, pp. 57–64, 2015.

[17] X. Hu, C. Rudin, and M. Seltzer, "Optimal sparse decision trees," Advances in Neural Information Processing Systems, vol. 32, 2019.

[18] P. Sathiyanarayanan, S. Pavithra, M. S. Saranya, and M. Makeswari, "Identification of breast cancer using the decision tree algorithm," in 2019 IEEE International conference on system, computation, automation and networking (ICSCAN), pp. 1–6, IEEE, 2019.

[19] I. H. Sarker, A. Colman, J. Han, A. I. Khan, Y. B. Abushark, and K. Salah, "Behavdt: a behavioral decision tree learning to build user-centric context-aware predictive model," Mobile Networks and Applications, vol. 25, pp. 1151–1161, 2020.

[20] S. Nandhini and J. M. KS, "Performance evaluation of machine learning algorithms for email spam detection," in 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), pp. 1–4, IEEE, 2020.

[21] J. Xiao and O. Bulut, "Evaluating the performances of missing data handling methods in ability estimation from sparse data," Educational and Psychological Measurement, vol. 80, no. 5, pp. 932–954, 2020.

[22] N. M. Kebonye, P. C. Agyeman, and J. K. Biney, "Optimized modelling of countrywide soil organic carbon levels via an interpretable decision tree," Smart Agricultural Technology, vol. 3, p. 100106, 2023.

[23] R. Koyama, M. Suzuki, Y. Nakamura, T. Mimura, and S. Ishiguro, "Estimating future human trajectories from sparse time series data," in Proceedings of the 1st International Workshop on the Human Mobility Prediction Challenge, pp. 26–31, 2023.

[24] L. Breiman, "Random forests," Machine learning, vol. 45, pp. 5–32, 2001.

[25] P. Xu and F. Jelinek, "Random forests and the data sparseness problem in language modeling," Computer Speech & Language, vol. 21, no. 1, pp. 105–152, 2007.

[26] V. Y. Kulkarni, P. K. Sinha, and M. C. Petare, "Weighted hybrid decision tree model for random forest classifier," Journal of The Institution of Engineers (India): Series B, vol. 97, pp. 209–217, 2016.

[27] J. L. Speiser, M. E. Miller, J. Tooze, and E. Ip, "A comparison of random forest variable selection methods for classification prediction modeling," Expert systems with applications, vol. 134, pp. 93–101, 2019.

[28] L. Demidova, I. Klyueva, and A. Pylkin, "Hybrid approach to improving the results of the svm classification using the random forest algorithm," Procedia Computer Science, vol. 150, pp. 455–461, 2019.

[29] X. Li, W. Chen, Q. Zhang, and L. Wu, "Building auto-encoder intrusion detection system based on random forest feature selection," Computers & Security, vol. 95, p. 101851, 2020.

[30] W. Feng, C. Ma, G. Zhao, and R. Zhang, "Fsrf: an improved random forest for classification," in 2020 IEEE International Conference on Advances in Electrical Engineering and Computer Applications (AEECA), pp. 173–178, IEEE, 2020.

[31] A. H. Sakib, A. K. Shakir, S. Sutradhar, M. A. Saleh, W. Akram, and K. B. M. B. Biplop, "A hybrid model for predicting mobile price range using machine learning techniques," in 2022 The 8th International Conference on Computing and Data Engineering, pp. 86–91, 2022.

[32] R. Polikar, "Ensemble learning," Ensemble machine learning: Methods and applications, pp. 1–34, 2012.

[33] X.-C. Yin, K. Huang, H.-W. Hao, K. Iqbal, and Z.-B. Wang, "A novel classifier ensemble method with sparsity and diversity," Neurocomputing, vol. 134, pp. 214–221, 2014.

[34] H. Li, Y. Cui, Y. Liu, W. Li, Y. Shi, C. Fang, H. Li, T. Gao, L. Hu, and Y. Lu, "Ensemble learning for overall power conversion efficiency of the all-organic dye-sensitized solar cells," IEEE Access, vol. 6, pp. 34118–34126, 2018.

[35] I. D. Mienye, Y. Sun, and Z. Wang, "An improved ensemble learning approach for the prediction of heart disease risk," Informatics in Medicine Unlocked, vol. 20, p. 100402, 2020.

[36] M. AlJame, I. Ahmad, A. Imtiaz, and A. Mohammed, "Ensemble learning model for diagnosing covid-19 from routine blood tests," Informatics in Medicine Unlocked, vol. 21, p. 100449, 2020.

[37] Q. R. S. Fitni and K. Ramli, "Implementation of ensemble learning and feature selection for performance improvements in anomaly-based intrusion detection systems," in 2020 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT), pp. 118–124, IEEE, 2020.

[38] H. E. Kiziloz, "Classifier ensemble methods in feature selection," Neurocomputing, vol. 419, pp. 97–107, 2021.

[39] D. M. Farid, L. Zhang, C. M. Rahman, M. A. Hossain, and R. Strachan, "Hybrid decision tree and naïve bayes classifiers for multi-class classification tasks," Expert systems with applications, vol. 41, no. 4, pp. 1937–1946, 2014.

[40] D. M. Farid, N. Harbi, and M. Z. Rahman, "Combining naive bayes and decision tree for adaptive intrusion detection," arXiv preprint arXiv:1005.4496, 2010.

[41] J. R. Quinlan, C4. 5: programs for machine learning. Elsevier, 2014.

[42] T. G. Dietterich, "Ensemble methods in machine learning," in International workshop on multiple classifier systems, pp. 1–15, Springer, 2000.

[43] G. Louppe, "Understanding random forests: From theory to practice," arXiv preprint arXiv:1407.7502, 2014.

• • •