International Journal of Safety and Security Engineering
Vol. 14, No. 6, December, 2024, pp. 1647-1665
Journal homepage: http://iieta.org/journals/ijsse

Securing Online Job Platforms: A Distributed Framework for Combating Employment Fraud in the Digital Landscape

Hassan I. Sayed Ahmed[1], Sherif A. Naiem[2], Ghada F. Elkabbany[1], Mohamed S. Abdallah[1,3,4], Young-Im Cho[4*]

[1] Informatics Department, Electronics Research Institute (ERI), Cairo 11843, Egypt
[2] Faculty of Engineering, Ain Shams University, Cairo 11845, Egypt
[3] DeltaX Co., Ltd., AI Laboratory, 590 Gyeongin-ro, Guro-gu, Seoul 08213, Republic of Korea
[4] Department of Computer Engineering, Gachon University, Seongnam 13415, Republic of Korea

Corresponding Author Email: yicho@gachon.ac.kr

https://doi.org/10.18280/ijsse.140601

Received: 9 September 2024
Revised: 11 October 2024
Accepted: 23 October 2024
Available online: 31 December 2024

Keywords:
social media cybersecurity, big data analytics, machine learning, distributed processing, Apache Spark

ABSTRACT

Job scams have existed for a while; advancements in technology have made them more accessible and profitable. By creating fake company websites and posting spurious job listings on well-known online job forums, cybercriminals impersonate actual employers and deceptively interview application victims. They then proceed to request personal information or money from their victims. To effectively address this issue, this work proposes an efficient framework that combines distributed processing, big data analytics, and machine learning to detect such attacks and combat emerging cyber threats. The proposed model employs mining rules to identify and prevent cyber threats in real-time, thereby enhancing user safety and fostering trust in online platforms. The proposed model accurately and efficiently detects fraud by leveraging the power of distributed processing and machine learning. The proposed framework, which provides a reliable approach for identifying fraudulent activity in job postings, is a hybrid approach that incorporates two different analyzing methods based on the data volume and dimension. The first method utilizes Conventional Machine Learning (CML) algorithms, while the second leverages Distributed Machine Learning (DML) algorithms on a distributed platform. The decision-making process regarding CML or DML ultimately depends on the specific application requirements, including the data characteristics, the need for distributed computing, and the operating environment. While CML models are a suitable choice for small datasets, DML models are beneficial when working with big datasets. For CML algorithms, the results demonstrate the effectiveness of the Random Forest (RF) algorithm in detection tasks. On the other hand, when the DML algorithms are implemented and evaluated to assess their performance, the experiments confirm that the Distributed Random Forest (DRF) algorithm exhibits high performance in this context. This research contributes to ongoing efforts to strengthen cybersecurity measures on digital platforms, thereby creating a safer online environment for both users and organizations.

1. INTRODUCTION

Using social media for job searching has facilitated connections but increased opportunities for scammers. This leads to significant financial losses and cybersecurity risks. According to the FTC and BBB reports, there has been an increase in employment fraud cases. In 2017, the total number of victims was around 14,975, and losses exceeding $20 million. By 2021, reported losses rose to $46.7 million. In early 2023, scam losses reached approximately $840,000. These numbers highlight the urgent need for cybersecurity in online job search and recruitment [1-5]. To combat these threats, job seekers should verify the legality of the employment offer by researching the employer's background, checking their website and social media accounts, and directly contacting them. Additionally, it is crucial to be cautious with unwanted emails and stay informed about the best cybersecurity practices, such as using strong passwords, using two-factor authentication, updating software, and avoiding public Wi-Fi [6]. The Federal Trade Commission (FTC) provides guidance on reporting and recovering from employment scams [7].

To avoid famous cybersecurity attacks such as phishing attacks, job scams, ransomware and malware attacks, social engineering attacks, fake recruitment agencies, insider threats and data breaches, and fake interview scams accurately and efficiently in real-time, this research proposes a hybrid framework called Hybrid Job Posting-Analytical and Detection Framework (HJP-ADF) that depends on distributed processing, big data analytics, and machine learning.

While *phishing attacks* trick job seekers into revealing personal information, leading to identity theft and financial loss, *identity theft* can result in unauthorized access to accounts, credit fraud, or financial instability. On the other hand, *job scams* deceive users with fake job offers, causing direct financial loss and erosion of trust in the platform. *Ransomware and malware attacks* can lock users out of their systems or steal sensitive data. Moreover, *social engineering attacks* manipulate job seekers into sharing confidential information or making harmful decisions, often with lasting financial impacts. Additionally, *Fake recruitment agencies* exploit users by requesting upfront payments for services that never materialize, leaving them financially vulnerable. Furthermore, *Insider threats and data breaches* expose users' private information, leading to potential misuse or sale on the dark web. Finally, *fake interview scams* gain job seekers' trust only to demand payments or sensitive documents under pretenses, leaving victims both financially and emotionally strained. These attacks not only harm individuals but also damage the reputation and reliability of the job platforms themselves.

The HJP-ADF focuses on several key research questions. Firstly, it aims to determine the effectiveness of job details features in identifying recruitment scams and fraud. Secondly, it seeks to identify the machine learning model that demonstrates the highest performance in detecting job scams. Thirdly, it leverages the strengths of the distributed architecture (Apache Spark) to enable real-time distributed detection and classification of anomalous behavior (fake jobs). Lastly, it aims to identify statistically significant subsets of the most reliable features for job scams and fraud identification. Therefore, it aims to examine the impact of different features on job fraud detection and build the most effective detection. A public dataset provided by Kaggle [8] is used for training and testing. The collected data includes textual and numerical information from job posts sourced from online job hiring platforms. Data analysis is performed to gain insights into the distribution and characteristics of real and fake jobs. Python libraries are utilized for data analysis and visualization. Several popular conventional machine learning models including Decision Trees (DT), Random Forest (RF), Logistic Regression (LR), Boosting with Decision Trees (BDT), Bagging with Decision Trees (BagDT), Boosting with Random Forest (BRF), Bagging with Random Forest (BagRF), Stochastic Gradient Descent (SGD), Nearest Centroid (NC) are considered. Additionally, for distributed manner Distributed Decision Trees (DDT), Distributed Random Forest (DRF), Distributed Logistic Regression (DLR), Distributed Gradient-Boosted Tree (DGBT), Distributed Linear Support Vector Machine (DLSVM), Distributed Naive Bayes (DNB), and Distributed Factorization Machines (DFM) models are deemed.

Two evaluation metrics, namely k-fold cross-validation and accuracy learning curves, are applied to assessing the performance of both CML and DML models. The research results contribute to developing a robust model for job recruitment fraud detection based on the effective use of distributed processing, big data analytics, and machine learning by identifying influential features for job fraud detection (such as salary range, company profile, organization name, required education, and the presence of multiple jobs). It concludes that while CML models are a useful choice for small datasets that do not require distributed computation, DML models are suitable when working with datasets that are too big (big data) to fit in a single machine. Besides, the findings establish the effectiveness of the Random Forest (RF) algorithm in detection tasks. On the other hand, when the DML algorithms are implemented and evaluated to assess their performance, the experiments confirm that the Distributed Random Forest (DRF) algorithm exhibits high performance in this context.

This paper is organized as follows: Section 2 presents an overview of various approaches for real-time cyber threat detection and prevention. Then, Section 3 outlines the methodology for designing and implementing the proposed cybersecurity framework and highlights its potential benefits. In Section 4, the experimental results are presented. Finally, Section 5 concludes the work with a discussion of the next steps for further research and development.

2. LITERATURE REVIEW

Cybersecurity threats have become more common in recent years, especially on social media platforms. They are particularly vulnerable to fraudulent activities such as job scams. Fake job scams harm individuals and damage the reputation of legitimate job opportunities. Combating these scams requires collaboration among individuals, organizations, and law enforcement, promoting awareness, sharing information, and taking proactive measures. Researchers are investigating real-time methods to detect and prevent cyber threats on social media. Some approaches involve using Natural Language Processing (NLP) techniques to identify malicious content on social media platforms. Wang et al. [9] presented an NLP-based approach to detect fake job postings on LinkedIn. Their study showed the effectiveness of their strategy in accurately identifying fraudulent job postings, but it may have scalability and balance challenges.

Furthermore, other researchers have explored using DL to avoid pushing attacks [10-12]. Benavides et al. classified the deep learning algorithms used to prevent pushing attacks [10]. Wu et al. [11] presented a DL-based approach to detecting phishing attacks on Twitter by analyzing user behavior patterns. Another research explored using mining rules to detect phishing emails based on features like message content and sender information [12].

Additionally, researchers investigated data fusion techniques to enhance the accuracy and effectiveness of detecting cyber threats. Li et al. [13] introduced a data fusion-based approach that integrates various data sources such as user behavior, content, and network structure data to identify fake accounts on social media platforms. Likewise, graph-based techniques can detect fake news on social media by analyzing relationships between users and content [14].

Besides, alternative methods entail detecting and preventing online recruitment fraud using machine learning approaches [15-20]. For instance, Quihui-Rubio et al. [15] in their work compared the performance of Logistic Regression (LR), Multi-layer Perceptron, Random Forest (RF), and Decision Trees (DT) algorithms in distinguishing between legitimate job advertisements and fake postings. They used the Employment Scam Aegean Dataset (EMSCAD) to train and test their models. Their findings indicated that the Multi-layer Perceptron and LR models accurately classify fake job posts. However, the study did not mention any potential biases or limitations of the EMSCAD dataset.

Moreover, some studies have combined various techniques to detect cybersecurity threats [21-23]. For instance, Anita et al. [21] focused on detecting and differentiating fake job postings from real ones using advanced ML and DL techniques. For ML classification purposes, they used logistic regression, KNN, and RF classifiers. On the other hand, Bi-Directional LSTM-DL was used to train the neurons. Although the authors claim that the Bi-Directional LSTM is the most accurate in detecting fake jobs, the paper did not provide specific details about the ML and DL algorithms. Further, the evaluation metrics and experimental results were not presented. Another work was done by Khan et al. [22], they presented an ensemble model that combines ML and rule-based techniques to identify fake news on social media platforms. According to their statement, the suggested approach improved the accuracy compared to individual models. The drawback of this method is the complexity and computational cost. This could make implementation and deployment more challenging, especially in real-time scenarios where fast decision-making is required.

By utilizing a combination of ML algorithms and the latest advancements in big data analytics it is possible to enhance the security and safety of online users and promote increased trust in online platforms. This work proposes a hybrid framework called HJP-ADF that takes advantage of big data analytics, machine learning, and distributed processing to analyze vast amounts of data from various sources, including user behavior patterns, network structure data, and content data. This aims to identify potential job scams on social media platforms. Moreover, the HJP-ADF framework is customized to develop mining rules that align with the specific requirements of each platform, thereby enhancing its efficacy in detecting and preventing cyber threats. Consequently, this contributes to fortifying the security and safety of online users and promotes a more favorable and trustworthy environment within social media platforms. The subsequent section delves into an elaborate discussion of the proposed fraud detection framework.

3. THE PROPOSED JOB FRAUD DETECTION - ANALYTICAL AND DETECTION FRAMEWORK (HJP-ADF)

Nowadays, detecting job fraud via social media is one of the most important research areas. This work proposes a framework called Hybrid Job Posting-Analytical and Detection Framework (HJP-ADF) that combines big data analytics, machine learning, and distributed processing to detect job scams effectively and improve system performance. HJP-ADF uses ML models (both CML and DML) to identify patterns and trends indicative of fraudulent activities by analyzing extensive data from different sources like social media, job boards, and recruitment websites.

CML models are a practical choice for small (tiny) datasets that do not require distributed computation. It is popular among the Python ML community due to its ease of use, flexibility, and extensive collection of algorithms. Additionally, it offers tools for exploratory data analysis (EDA) and model interpretability, making it suitable for applications that require a thorough understanding of the model and its attributes. In addition, it provides simple and efficient pipeline capabilities for CML workflows that involve preprocessing, feature engineering, and model training. On the other hand, DML models are well-suited for large-scale data. It is designed for distributed computing and is particularly useful when working with datasets too big (big data) to fit in a single machine. They are a strong option for data processing workloads that require parallel and distributed computing. Moreover, these models integrate effectively with the larger ecosystem, making it easy to interact with other big data platforms and technologies (Big Data Ecosystem Integration). In addition, these models offer scalable feature engineering and data transformation tools, which are beneficial for large-scale preprocessing tasks (Data Transformation and Feature Engineering at Scale). Moreover, DML models are frequently used in cloud computing environments like AWS and Azure, where scalability and distributed computation are crucial.

As shown in Figure 1, the proposed HJP-ADF takes in job posting data as input. This data contains information about advertised job vacancies and opportunities, including job title, job description, salary, required qualifications, skills, experience, location, and application instructions. This data is collected from job boards, company websites, and online recruiting platforms. Analyzing this data helps to identify acquisition trends, market demand for specific skills, and potential job opportunities in different industries and locations. Job posting data is categorized as traditional and highly dimensional data (big data). While data preprocessing is the first step for traditional processing methods, big data analytics platforms are the corresponding ones for big data. This work uses the Apache Spark platform for big data analytics [24]. Apache Spark, an open-source cluster computing framework, enables the analysis and extraction of insights from large volumes of data. Furthermore, it provides tools and technologies for efficiently storing, processing, and analyzing big data sets. This capability enables the extraction of valuable insights, supports data-driven decision-making, and provides a competitive advantage [25, 26].

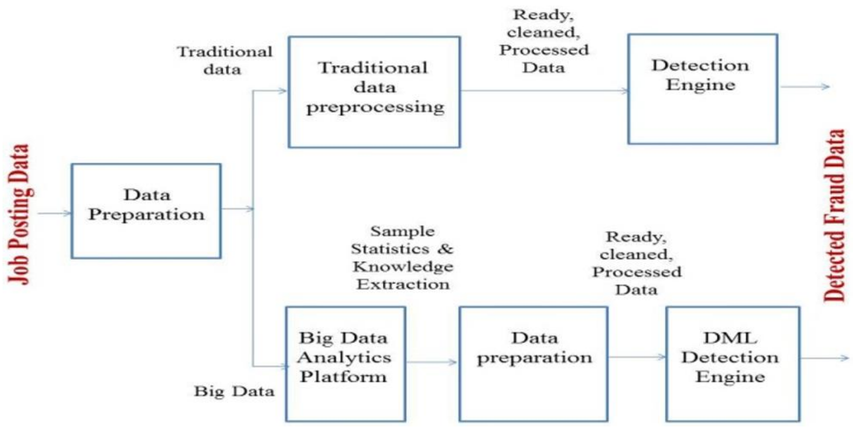

Job Posting Data
Data Preparation
Traditional data
Traditional data preprocessing
Ready, cleaned, Processed Data
Detection Engine
Big Data
Big Data Analytics Platform
Sample Statistics & Knowledge Extraction
Data preparation
Ready, cleaned, Processed Data
DML Detection Engine
Detected Fraud Data

Figure 1. The proposed framework (HJP-ADF)

The detection engine is the machine learning method utilized for data processing and detection. In the context of fake job detection, it is deployed to train algorithms on relevant data, enabling the models to learn patterns and features that distinguish real jobs and other fake jobs. There are two components in the HJP-ADF framework acting as detection engines as shown in Figure 1. Concerning traditional analyzing and detection, the CML detection engine is based on a high-performance CML algorithm. The other component concerned with big data analytics, the DML detection engine is employed for classification the of job postings based on a highly accuracy DML algorithm. In the detection engine module, experiments are applied to the appropriate model for CML and DML. Moreover, Figure 2 explains the components of the proposed HJP-ADF framework. It illustrates the workflow of the framework in detail.

Figure 3 presents a flowchart of the proposed HJP-ADF framework. The flowchart emphasizes the framework steps as follows:

- Data collection: HJP-ADF emphasizes the importance of obtaining a reliable dataset. The selected "real/fake job posting dataset" [8] ensures that subsequent analyses are based on relevant and representative data.
- Simple statistics and knowledge extraction from the data: The dataset is subjected to simple statistics and knowledge extraction using DML resulting in a comprehensive report. This step provides initial insights into the characteristics of the dataset and prepares the foundation for further exploitation.
- Initial data analysis: The framework advances to an in-depth analysis phase, facilitated by DML. The "big data analytics report" refines the dataset and enables a more comprehensive understanding of its complexities and potential discrepancies.
- Modeling and evaluation: Following data preparation, the framework goes to the modeling step. In this step, machine learning techniques are used to create a fraud detection model. The performance of different models is carefully evaluated to ensure their accuracy and reliability deployment.
- Feedback analysis: The selected validated model (obtained from the experiments) is used to detect fraud in real-time.

Finally, the flowchart depicts a feedback loop that follows the deployment phase; this iterative process ensures that real-world outcomes drive continuous improvement.

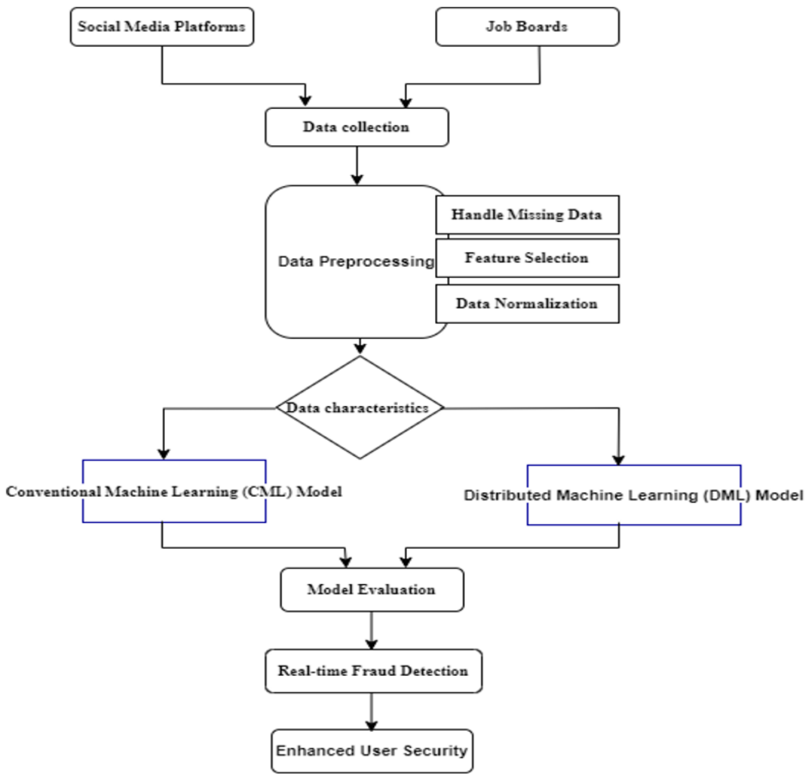


Figure 2. The workflow of the proposed framework

3.1 Real/Fake job posting prediction: Dataset of real and fake job postings

This research is conducted using a dataset of real and fake job postings obtained from Kaggle [8, 21]. The chosen dataset contains approximately 18,000 real and fake job postings, making it ideal for training models to detect fraudulent postings in real-world scenarios. Its diverse features, including textual and numerical data, enable comprehensive analysis, but the dataset has biases and limitations. It is highly imbalanced, with around 95% real and 5% fake jobs (800), potentially skewing the model toward real jobs unless techniques like oversampling are used. Additionally, 84% of the records have missing values, which could affect the model's accuracy. The dataset may also lack geographic and industry diversity, limiting its applicability across different contexts. However, our proposed Hybrid Job Posting-Analytical and Detection Framework (HJP-ADF) leverages advanced machine learning techniques, providing robust capabilities to classify and detect fraudulent job postings accurately. Furthermore, HJP-ADF is designed to handle the increasing volume and velocity of fraudulent job postings being generated in real-world scenarios, ensuring the framework remains scalable and effective as cybercriminals evolve their tactics.

The dataset includes a variety of features connected to job adverts, where each feature gives unique information. These characteristics include a unique job ID (job_id), job title (Title), geographical location (Location), corporate department (department), an estimated pay range (salary_range), a brief company profile (company_profile), a thorough job description (description), position requirements (requirements), and a list of employer benefits (benefits). Additional features include if the position supports telecommuting (telecommuting), whether the organization has a logo (has_company_logo), and whether screening questions are required throughout the application process (has_questions). The employment type (employment_type), required experience level (required_experience), and educational credentials are also provided (required_education). Furthermore, the dataset includes information about the industry (Industry) and functions connected with each job (Function), and the "fraudulent" feature acts as a classification attribute, indicating whether the job posting is tagged as potentially fraudulent.

Selecting meaningful features is crucial in classification problems as it contributes to enhancing the accuracy of classification using suitable machine learning models. In a study conducted by Mehboob and Malik [27], it was found that feature selection plays a significant role and offers advantages in addressing classification problems. The selected dataset features can be classified into three different categories namely, categorical, textual, and numerical features. Categorical features include Title, Location, department, and telecommuting, has_company_logo, has_questions, employment_type, required_education, required_education, Industry, and Function. Textual features include company_profile, description, job requirements, and benefits. Numerical features include job_id and salary_range.

3.1.1 Categorical features

The selected dataset consists of common categorical features, each with a specific number of categories. These features include: Title (11,231), Location (3,106), department (1,338), telecommuting (2), has_company_logo (2), has_questions (2), employment_type (6), required_experience (8), required_education (14), Industry (132), and Function (38).

3.1.2 Textual features

There are four common text data indicators for distinguishing between real and fake job opportunities: company_profile, job description, job requirements, and job benefits [8].

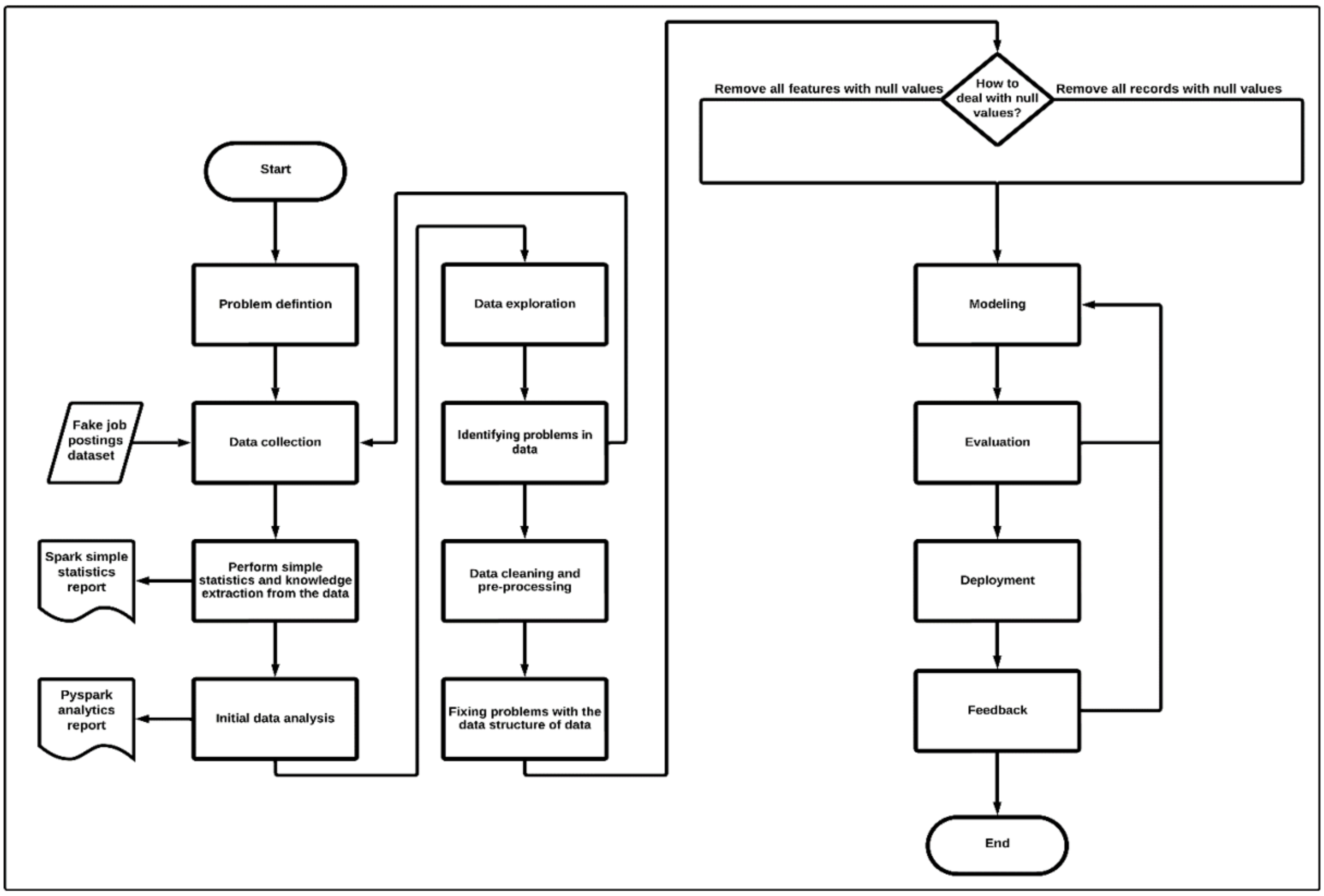

Start
Problem defintion
Fake job postings dataset
Data collection
Spark simple statistics report
Perform simple statistics and knowledge extraction from the data
Pyspark analytics report
Initial data analysis
Data exploration
Identifying problems in data
Data cleaning and pre-processing
Fixing problems with the data structure of data
Remove all features with null values
How to deal with null values?
Remove all records with null values
Modeling
Evaluation
Deployment
Feedback
End

Figure 3. Flowchart of the proposed HJP-ADF framework

3.1.3 Numerical features

In the selected dataset, the most important numerical feature is the salary_range feature. Salary statistics indicate that real job postings have higher Mean and median minimum salaries than fake postings, indicating a higher overall starting pay in real job opportunities. Minimum salary indicates that genuine job postings offer more competitive base pay than fake job listings. Real-world job postings have significantly higher Standard deviations for both minimum and maximum salaries, indicating broader salary ranges and greater variability.

In the next subsection, the pre-processing stages of the proposed HJP-ADF framework are discussed in detail.

3.2 Pre-processing stages of the proposed framework

Data preprocessing plays a crucial role in preparing data for machine learning tasks. Its primary objective is to clean and structure the data in a way that ensures accurate classification, training, and evaluation. Through preprocessing, the data is organized consistently and meaningfully, addressing issues such as missing values and outliers, and improving overall data quality. Effective preprocessing facilitates the identification of patterns and insights, enhances the interpretability of the data, and establishes a reliable foundation for successful machine learning models. Given that the selected dataset comprises three types of features; categorical, textual, and numerical; distinct preprocessing techniques are employed for each feature type.

3.2.1 Categorical features preprocessing

Data preprocessing includes various tasks such as handling missing values through imputation, converting ordinal categorical features using label encoding, managing nominal categorical features using one-hot encoding, and addressing rare categories by renaming or merging them. These steps are crucial for achieving optimal model performance. They mitigate potential problems arising from missing values, enable the conversion of categorical data into a numerical format understandable by machine learning algorithms, and reduce data scarcity to enhance stability in model predictions.

3.2.2 Textual features preprocessing

The text preprocessing phase ensures that textual input is cleaned and standardized before converting it into a numerical format understandable by machine learning models. Textual data preprocessing offers several benefits, including noise elimination, data standardization, and dimensionality reduction to improve model generalization. Additionally, it enhances text consistency, readability, and tokenization while preserving the primary content focus by eliminating repetitive and non-contributing terms.

One of the most important steps of text preprocessing is character handling. This step involves removing non-alphanumeric and non-digit characters, converting text to lowercase, expanding contractions, performing stemming and lemmatization, removing tabs and redundant whitespaces, replacing empty strings with the string "missing", and tokenization. Tokenization entails splitting text into words, converting split words to lowercase, and excluding stop words such as "the," "a," "an," "he," and "have" that contributed little to the overall meaning.

3.2.3 Numerical features preprocessing

For numerical features optimization, data preprocessing includes several key steps: Handling missing values through imputation, managing outliers, and applying uniform scaling via Min-Max scaling or standardization. These processes help reduce bias, enhance model robustness, and prevent feature dominance. Furthermore, binning, feature engineering, log transformation, and optional normalization techniques can improve model interpretability and fit.

These preprocessing steps collectively play a vital role in preparing numerical data for effective model training and prediction. Vectorization techniques bridge the gap between textual and numerical data, facilitating the efficient handling and learning of machine-learning models from text input. The salary feature is a significant attribute in classifying job seekers based on their job interests, making it highly influential for users. Table 1 and Table 2 represent the Mean and Standard deviation [28, 29] of the numerical data both before and after the removal of outliers in the salary feature.

Table 1. Mean and standard deviation of numerical data before removing the outliers

P.O.C	Real Job Postings	Fake Job Postings
Mean of the minimum salary	47093.27	49033.76
Standard deviation of the minimum salary	59655.48	31935.31
Median of the minimum salary	35000	60000
Mean of the maximum salary	126561	68548.24
Standard deviation of the maximum salary	901307.4	38574.65
Median of the maximum salary	50000	75000

Table 2. Mean and standard deviation of numerical data after removing the outliers

P.O.C	Real Job Postings	Fake Job Postings
Mean of the minimum salary	47353.52	55404.07
Standard deviation of the minimum salary	25635.46	28209.27
Median of the minimum salary	40000	65000
Mean of the maximum salary	66320.14	77453.33
Standard deviation of the maximum salary	35960.56	31272.03
Median of the maximum salary	60000	75000

After removing outliers, the analysis of salary data shows increased minimum incomes for both real and fake job postings, along with higher mean and median values. This suggests improved opportunities for job seekers. Furthermore, the standard deviations for both minimum and maximum salaries decreased in both categories, indicating reduced salary variability within job postings. This implies that removing outliers led to more consistent and predictable salary ranges, giving job seekers better decision-making insights.

4. EXPERIMENTAL RESULTS AND EVALUATION

Moving on to the discussion of the result, it is considered that the dataset includes classes 0 (real job) and 1 (fake job), with a class rate of 95% and 5%, respectively. The initial DataFrame has 17880 records and 18 features. In subsequent experiments on the test data, all classifiers use a stratified split of 85% training data and 15% testing data to maintain a balanced distribution between real and fake job postings. Outliers are removed from the numerical features before conducting the experiments, and the evaluation metrics are presented through column charts that display precision, recall, and F1-score for both classes and overall accuracy.

In addition, this work utilizes both CML and DML models to assess their performance and scores. In the context of CML, various classifiers are employed including Decision Tress (DT), Random Forest (RF), Logistic Regression (LR), Boosting with DT (BDT), Bagging with DT (BagDT), Boosting with Random Forest (BRF), Bagging with Random Forest (BaggRF), Stochastic Gradient Descent (SGD), Nearest Centroid (NC), and Perceptron algorithms [30]. Within the DML several classifiers are employed including Distributed Decision Trees (DDT), Distributed Random Forest (DRF), Distributed Logistic Regression (DLR), Distributed Gradient-Boosted Tree (DGBT), Distributed Linear Support Vector Machine (DLSVM), Distributed Naive Bayes (DNB), and Distributed Factorization Machines (DFM) classifiers are applied [31, 32].

Table 3. Null count and percentage of categorical features

Categorical Features	null_count	null_percentage
Title	0	0%
Location	346	1.94%
Department	11547	64.58%
Telecommuting	0	0%
Has_company_logo	0	0%
Has_questions	0	0%
Employment_type	3471	19.41%
Required_experience	7050	39.43%
Required_education	8105	45.33%
Industry	4903	27.42%
Function	6455	36.10%

Table 4. Null count and percentage of textual features

Textual Features	null_count	null_percentage
company_profile	3308	18.5%
Description	1	0.01%
Requirements	2695	15.07%
Benefits	7210	40.32%

Table 5. Null count and percentage of numerical features

Numerical Features	null_count	null_percentage
job_id	0	0%
min_salary	15012	83.96%
max_salary	15012	83.96%

Five experiments are conducted for each detection engine, focusing on managing missing values and addressing categorical, textual, and numerical features. In the first experiment, all missing values are included without dropping any feature. In the second experiment, missing values of textual features are dropped while other missing values are included. In the third experiment missing values of categorical and textual features are dropped while numerical missing values are included. The fourth experiment drops all missing values. Finally, the fifth experiment drops features with many missing values and keeps relevant features. Through all experiments, it is assumed that if a textual or categorical feature has missing value, it will be replaced with the term 'missing'. However, any numerical feature with missing values is replaced with the number '0'.

Tables 3-5 illustrate null_count and null_percentages, which are used for categorical, textual, and numerical features in both CML and DML, respectively.

4.1 Using conventional machine learning (CML) classifiers

As mentioned earlier, the employed CML classifiers are Decision Trees (DT), Random Forest (RF), Logistic Regression (LR), Boosting with Decision Trees (BDT), Bagging with Decision Trees (BagDT), Boosting with Random Forest (BRF), Bagging with Random Forest (BagRF), Stochastic Gradient Descent (SGD), Nearest Centroid (NC), and Perceptron algorithms. The subsections describe a detailed account of the performance of various experiments conducted using different CML models.

4.1.1 Dealing with all missing values without dropping any features

This experiment aims to address missing values without dropping any features. The class distribution remains consistent, with Class 0 accounting for 95% and Class 1 for 5%. The resultant DataFrame maintains the same 18 features but has a modified shape, consisting of 4965 records. Figure 4 (a) illustrates scores of CML classifiers using all features. Notably, all classifiers showcase excellent recall, precision, and F1-score for Class 0, resulting in impressive overall accuracy of 99%. Regarding fake job postings, the BDT and BagDT classifiers slightly underperform compared to Class 0 but still maintain a high accuracy level of 99%. On the other hand, the SGD classifier presents comparatively weaker results, achieving a moderate accuracy of 94%. Besides, the NC and Perceptron classifiers display relatively minor results for both classes, especially for Class 1, with moderate accuracy levels (83% for Perceptron and 85% for NC).

4.1.2 Dropping missing values of textual features while handling other missing values

This experiment aims to systematically address the presence of missing data, contributing to a more comprehensive analysis of the dataset by mitigating the impact of missing values on the features' textual and numerical aspects. The class distribution reveals that Class 0 constitutes the majority, accounting for 96%, while Class 1 comprises a smaller proportion at 4%. The resulting DataFrame consists of 2846 records and includes 18 features. Figure 4 (b) shows that all classifiers showcase excellent recall, precision, and F1-score for Class 0. Regarding Class 1, while the DT, RF, LR, Boosting, and Bagging classifiers maintain consistent excellence in Class 1 and achieve a perfect accuracy rate of 100%, the BRF and BagRF classifiers have a slight reduction in recall. Otherwise, the SGD, NC, and Perception classifiers encounter challenges in Class 1. Although the SGD classifier slightly diminishes the F1-score, it maintains a commendable accuracy rate of 96%. On the other hand, the NC classifier achieves an overall accuracy of 81%. Besides, the Perceptron classifier struggles to identify Class 1, resulting in a 0% precision, recall, and F1-score, with a minor effect on the overall accuracy of 96%.

4.1.3 Dropping missing values of textual and categorical features while handling numerical missing values

In this experiment, missing values in categorical and textual features are removed, while numerical feature values are replaced by the numeric value '0'. Analysis of the class distribution exposes that Class 0 constitutes the majority of 94%, while Class 1 comprises a smaller proportion at 6%. The resultant DataFrame consists of 2207 records, and 18 features.

The CML classifier scores are illustrated in Figure 4 (c). This figure shows that all classifiers accurately identify Class 0, achieving flawless recall, precision, and F1-score. Concerning fake job postings (Class 0), the DT, RF, LR, and Boosting classifiers demonstrate excellent performance with high overall accuracy ranging between 99% and 100%. On the other hand, the BagDT, BagRF, NC, and Perceptron classifiers show a relatively lower performance. Whereas the BagDT and BagRF classifiers keep up a remarkable accuracy rate of 99%, the SGD achieves an overall accuracy of 94%, the NC classifier attains an overall accuracy of 73%, and the Perceptron classifier reaches an accuracy rate of 94%.

4.1.4 Dropping all missing values

This experiment handles missing values within all features and aims to create a more refined and balanced dataset for subsequent analysis and modeling. The resulting class distribution indicates 90% for Class 0 and 10% for Class 1. Consequently, the shape of the DataFrame is modified to comprise 720 records and 18 features. As shown in Figure 4 (d) Class 0 achieves complete accuracy across all classifiers. As regards to Class 1, the RF and Boosting classifiers achieve a 100% accuracy rate, while the LR maintains 97% accuracy. However, the NC and SGD face challenges, resulting in lower precision, recall, and F1-score.

4.1.5 Dropping features with many missing values and keeping relevant features

In this experiment, certain features such as job_id, title, location, department, telecommuting, has_company_logo, has_questions, employment_type, required_experience, industry, and function fetures are dropped and only company_profile, description, requirements, benefits, min_salary and max_salary are utilized in the training and testing of the CML models. The resultant class distribution reveals 16% for Class 0 and 84% for Class 1. Consequently, the resultant DataFrame consists of 4965 records and 9 features. As shown in Figure 4 (e) all classifiers demonstrate high recall, precision, and F1-score for Class 0. Concerning Class 1, the DT, RF, LR, Boosting, and Bagging classifiers attain high recall, precision, and F1-score, showcasing strong performance with high accuracy (reaching 99%). Besides, the SGD classifier shows moderate performance with a 96% accuracy rate. Both the NC and the Perceptron classifier classifiers face challenges, leading to a notable decrease in recall, precision, and F1-score, with an 85% accuracy rate.

4.2 Using distributed machine learning (DML) based on Apache Spark framework classifiers

This subsection describes the system performance of the DML classifiers, including Distributed Decision Trees (DDT), Distributed Random Forest (DRF), Distributed Logistic Regression (DLR), Distributed Gradient-Boosted Tree (DGBT), Distributed Linear Support Vector Machine (DLSVM), Distributed Naive Bayes (DNB), and Distributed Factorization Machines (DFM), based on Apache Spark when applying the above mentioned five experiments.

4.2.1 Dealing with all missing values without dropping any features

As mentioned above, this experiment aims to address missing values without dropping any features. Figure 5 (a) presents the scores of the DML classifiers when using all features. Similar to the CML classifiers, all DML classifiers showcase excellent recall, precision, and F1-score for Class 0, resulting in impressive overall accuracy. Regarding fake job identification, the DDT and DRF classifiers face challenges resulting in a lower F1-score due to compromised recall and precision. Besides, the performance of the DLR classifier, especially in recall, is moderate. Moreover, the DGBT and DLSVM classifiers display decent precision; however, their recall is relatively low. In contrast, the DNB classifier has lower overall metrics and accuracy. Lastly, the DFM classifier demonstrates balanced performance for both classes, achieving an accuracy rate of 95%.

4.2.2 Dropping missing values of textual features while handling other missing values

Figure 5 (b) represents the textual features based on DML classifier scores. As per the previous experiment, all classifiers exhibit outstanding performance with the highest F1-score, recall, and precision for Class 0, maintaining a high overall accuracy. Regarding the fake job identifying, the DDT, DRF, DLR, DGBT, and DLSVM classifiers demonstrate almost flawless metrics, with slightly lower performance but still retained a commendable accuracy of 99%, 98%, 98%, 98%, and 98% respectively. On the other hand, both the DNB and DFM classifiers encounter challenges with Class 1. While the DNB classifier has an overall accuracy rate of 82%, the DFM maintains an accuracy rate of 97%.

4.2.3 Dropping missing values of textual and categorical features while handling numerical missing values

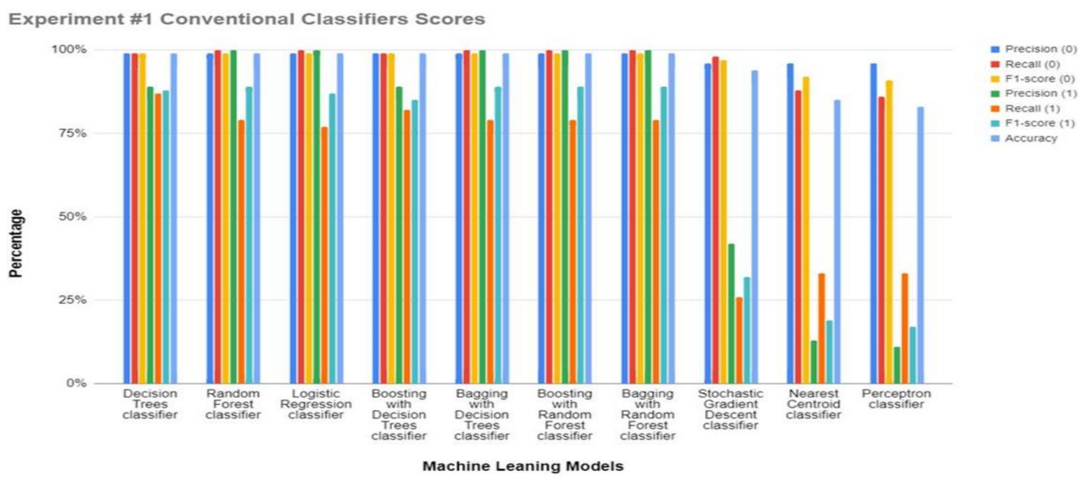
Figure 5 (c) illustrates the performance of various DML classifiers. Missing values of textual and categorical features are dropped while any missing numerical feature values are replaced with '0'.

4.2.4 Dropping all missing values

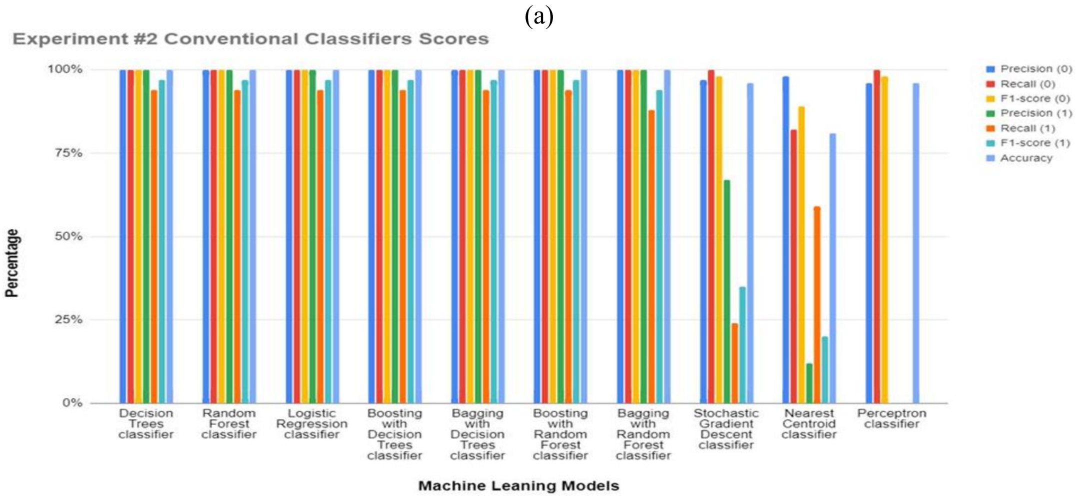
Figure 5 (d) illustrates the performance of various DML classifiers after dropping all missing values of textual, categorical, and numerical features. Like the previous experiments, all classifiers exhibit outstanding performance with the highest F1-score, recall, and precision for Class 0, maintaining a high overall accuracy. However, for Class 1, the DDT and DRF classifiers demonstrate a moderate performance, resulting in an overall accuracy rate of 97%. Besides, the DLR and DGBT classifiers exhibit variable performance. The DLSVM classifier struggles to achieve 91% accuracy, and the DNB classifier has low recall and precision, leading to 55% accuracy. Lastly, the DFM classifier fails to identify Class 1, slightly impacting its overall accuracy of 91%.

4.2.5 Dropping features with many missing values and keeping relevant features

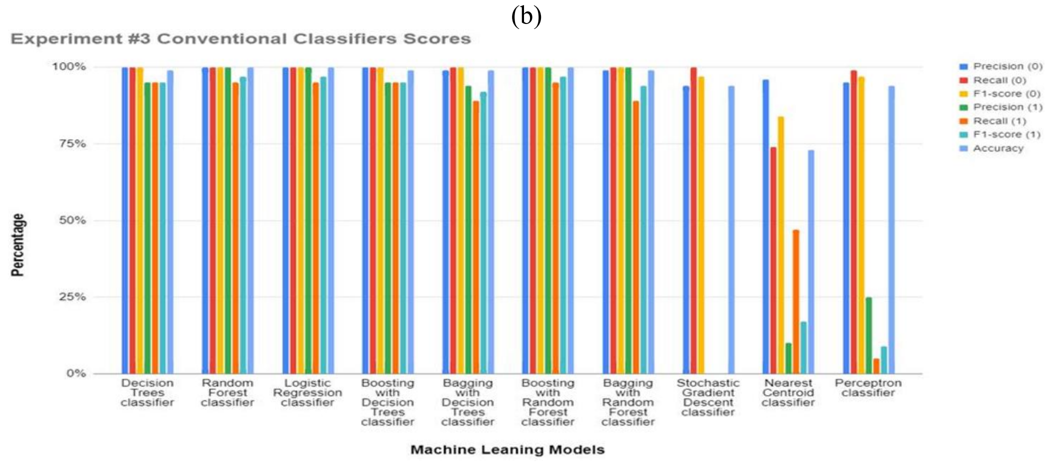
Different DML classifiers are evaluated when missing values are dropped, as shown in Figure 5 (e). All classifiers show impressive performance metrics for Class 0. Regarding Class 1, the performance of the DDT and DRF classifiers is moderate, with lower recall, F1-score, and precision. While the DLR, DGBT, and DLSVM classifiers have varied performance, the DLR classifier has comparatively worse precision, recall, and F1-score, resulting in an accuracy rate of 97%. On the other hand, the DNB classifier comes across challenges, impacting its overall accuracy (rate 92%). Besides, the DFM classifier struggles significantly with Class 1, affecting its overall accuracy at 85%.


Experiment #1 Conventional Classifiers Scores
Precision (0)
Recall (0)
F1-score (0)
Precision (1)
Recall (1)
F1-score (1)
Accuracy
Percentage
100%
75%
50%
25%
0%
Decision Trees classifier
Random Forest classifier
Logistic Regression classifier
Boosting with Decision Trees classifier
Bagging with Decision Trees classifier
Boosting with Random Forest classifier
Bagging with Random Forest classifier
Stochastic Gradient Descent classifier
Nearest Centroid classifier
Perceptron classifier
Machine Leaning Models

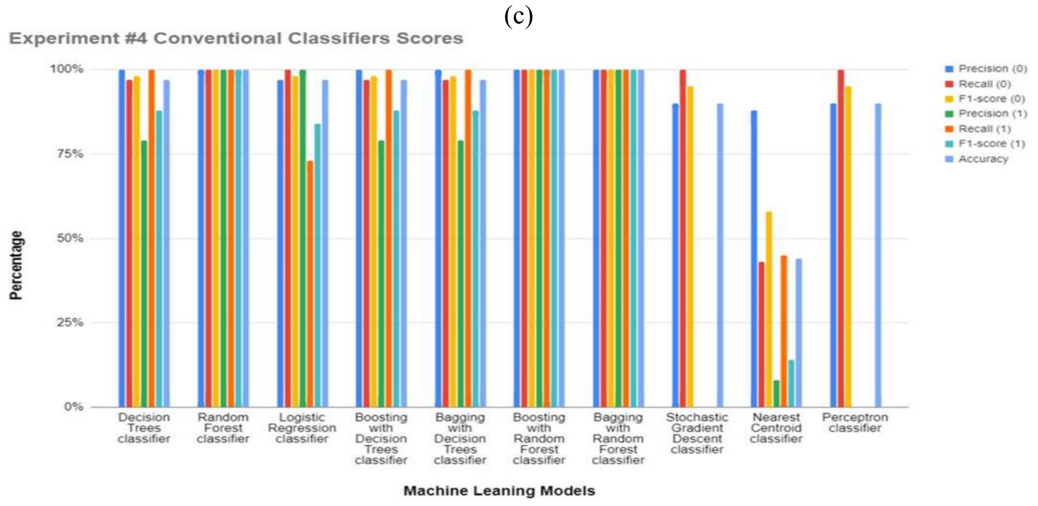
(a)


Experiment #2 Conventional Classifiers Scores
Precision (0)
Recall (0)
F1-score (0)
Precision (1)
Recall (1)
F1-score (1)
Accuracy
Percentage
100%
75%
50%
25%
0%
Decision Trees classifier
Random Forest classifier
Logistic Regression classifier
Boosting with Decision Trees classifier
Bagging with Decision Trees classifier
Boosting with Random Forest classifier
Bagging with Random Forest classifier
Stochastic Gradient Descent classifier
Nearest Centroid classifier
Perceptron classifier
Machine Leaning Models

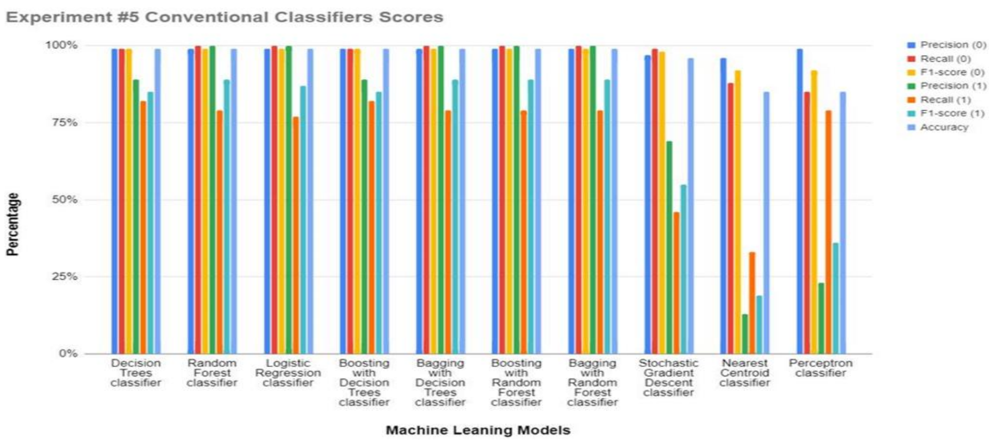
(b)


Experiment #3 Conventional Classifiers Scores
Precision (0)
Recall (0)
F1-score (0)
Precision (1)
Recall (1)
F1-score (1)
Accuracy
Percentage
100%
75%
50%
25%
0%
Decision Trees classifier
Random Forest classifier
Logistic Regression classifier
Boosting with Decision Trees classifier
Bagging with Decision Trees classifier
Boosting with Random Forest classifier
Bagging with Random Forest classifier
Stochastic Gradient Descent classifier
Nearest Centroid classifier
Perceptron classifier
Machine Leaning Models

(c)

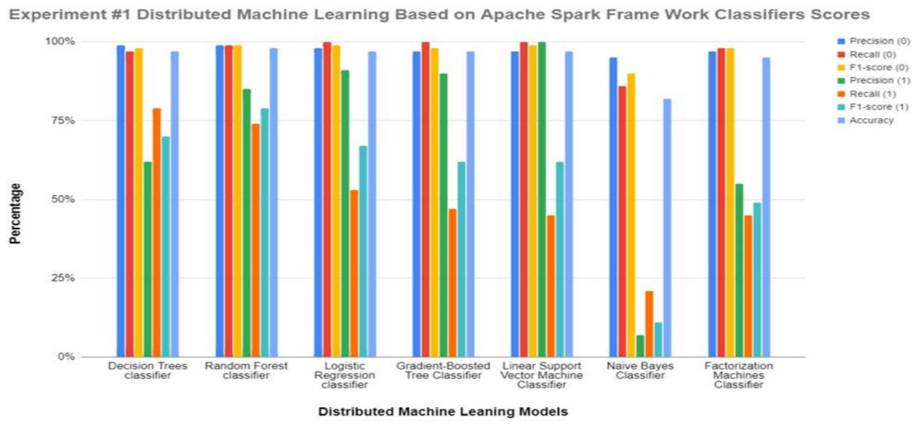

Experiment #4 Conventional Classifiers Scores
Precision (0)
Recall (0)
F1-score (0)
Precision (1)
Recall (1)
F1-score (1)
Accuracy
Percentage
100%
75%
50%
25%
0%
Decision Trees classifier
Random Forest classifier
Logistic Regression classifier
Boosting with Decision Trees classifier
Bagging with Decision Trees classifier
Boosting with Random Forest classifier
Bagging with Random Forest classifier
Stochastic Gradient Descent classifier
Nearest Centroid classifier
Perceptron classifier
Machine Leaning Models

(d)

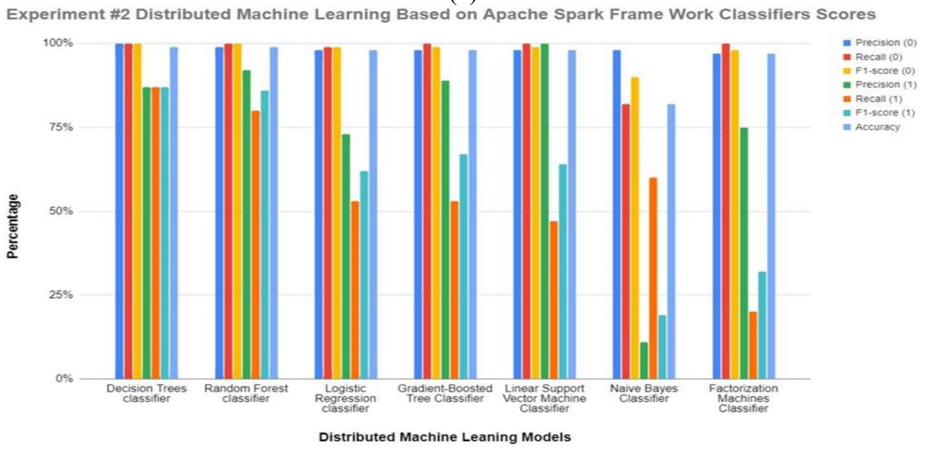


(e)

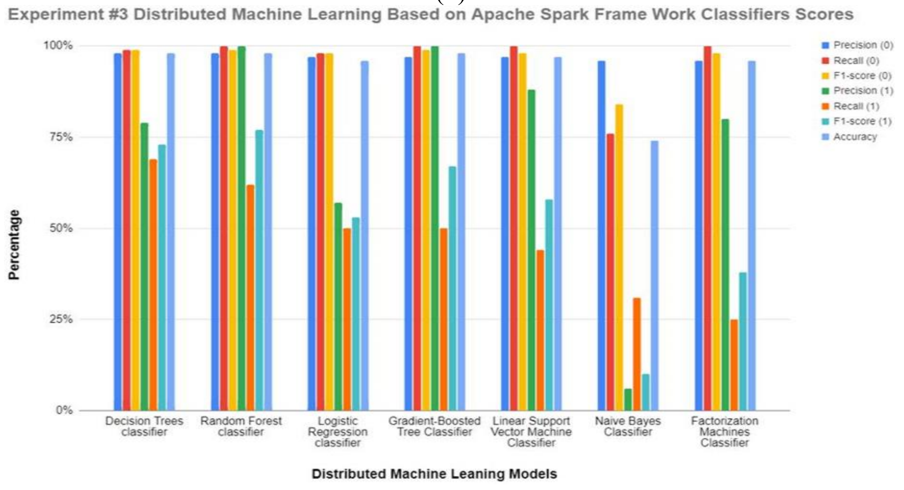
Figure 4. Conventional machine learning classifier scores



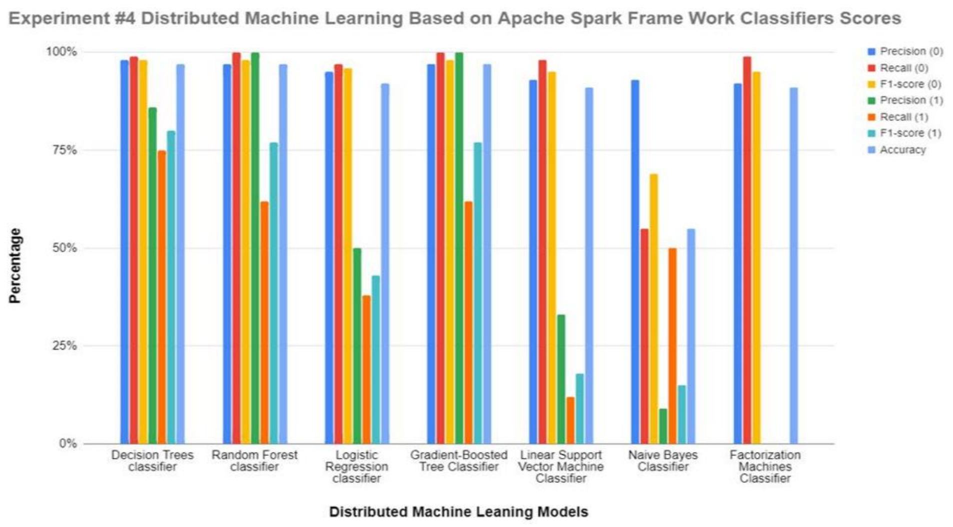
(a)



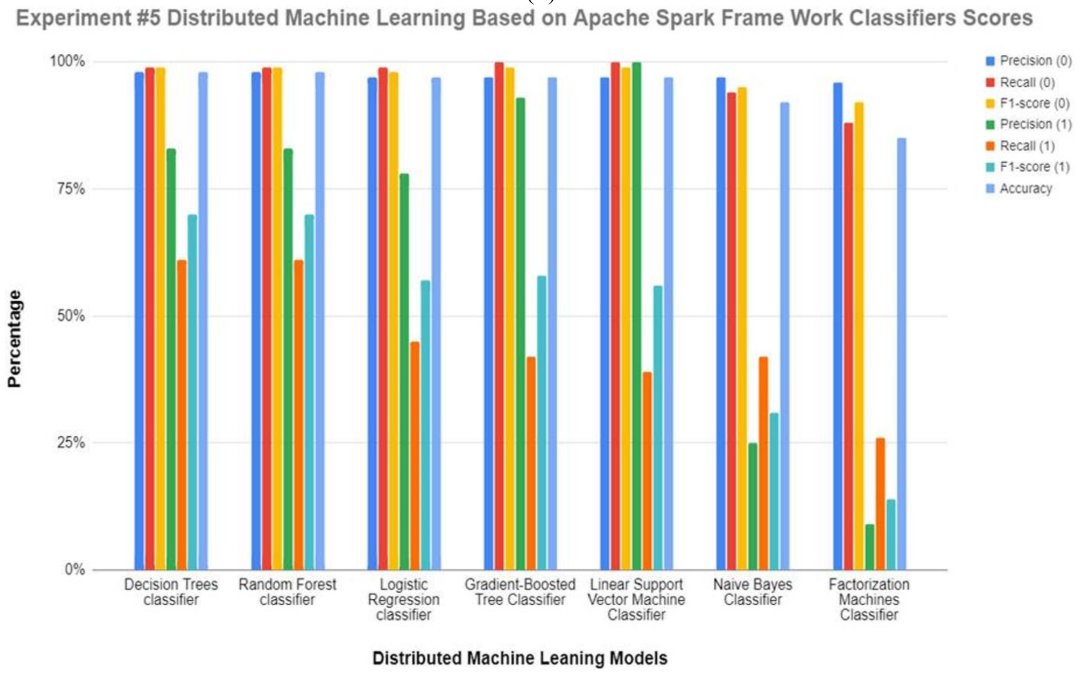
(b)



(c)



(d)



(e)

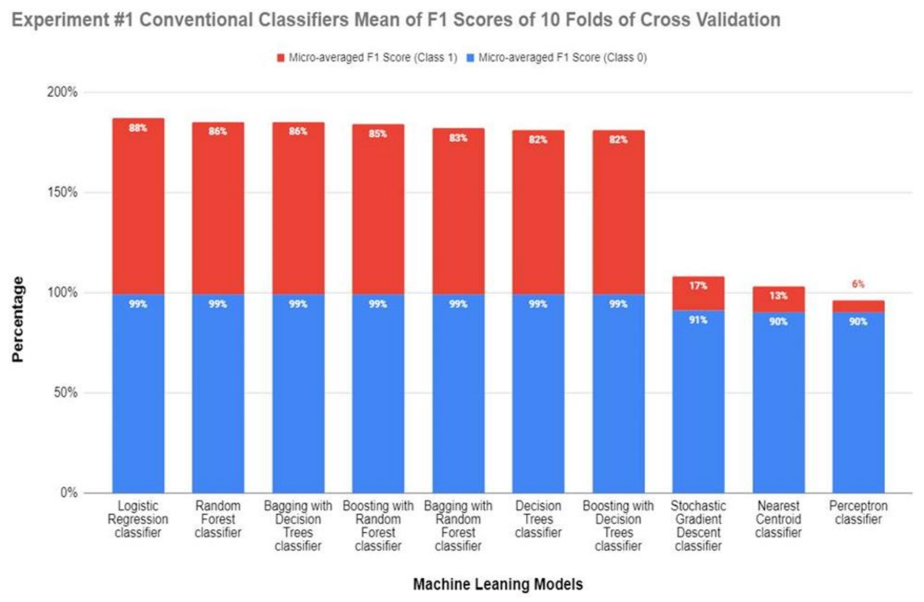
Figure 5. Distributed machine learning classifier scores

4.3 Assessing CML and DML performance through k-fold cross-validation and accuracy learning curves

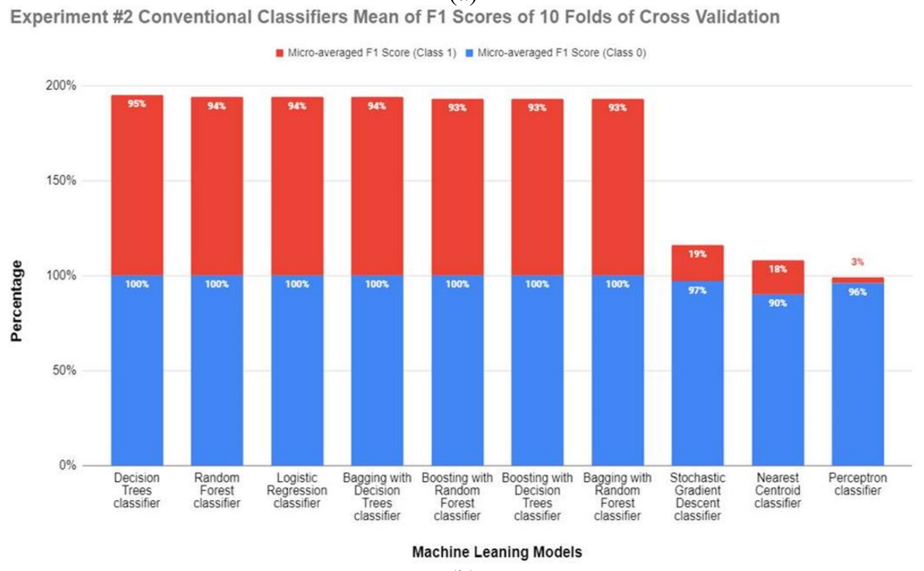
There are several evaluation methods that determine the accuracy of machine learning models; in this work, k-fold cross-validation and accuracy learning curves are used [33-37]. Since the dataset is imbalanced, F1-scores are used as a reliable metric for evaluating the best-performing machine learning classifier for both CML and DML.

4.3.1 Examining CML and DML performance using k-fold cross-validation

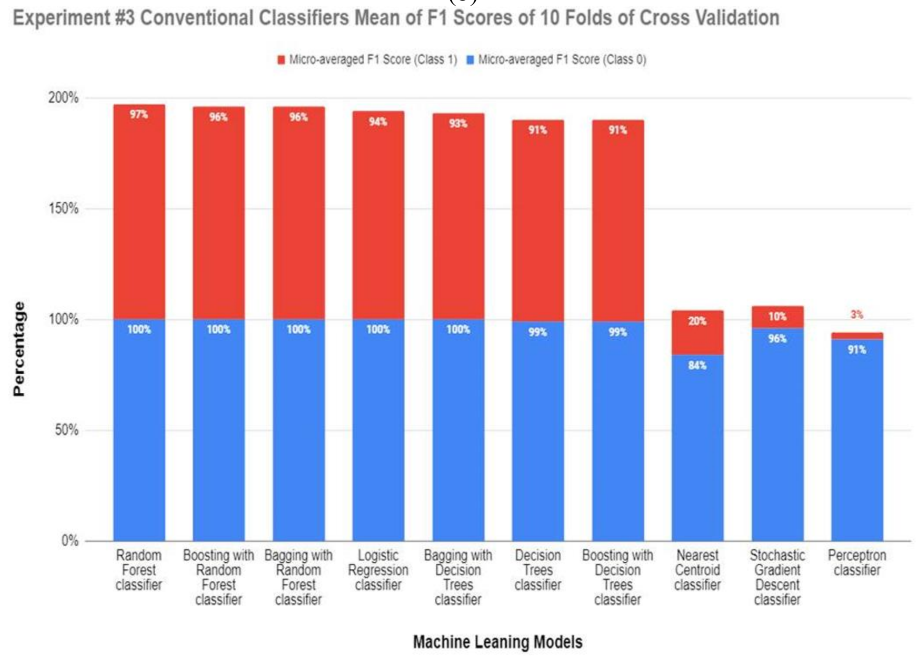
Cross-validation is a method for evaluating the predictive performance of ML models on unseen data [33-35]. It splits the data into ‘k’ subsets and creates training and validation sets. The purpose of cross-validation is to provide a more reliable and unbiased measure of different classifiers' performance, as compared to a single train-test split. Figure 6 illustrates a comparison of various classifiers applied to the five aforementioned experiments on the CML training data. Regardless of the experiment, the SGD, NC, and Perceptron classifiers consistently exhibit the lowest F1-scores. In Experiment #1, the LR classifier achieves the highest F1 scores, and the RF classifier ranks second. Additionally, the DT classifier takes third place with good F1-scores. However, the boosting and bagging for both the RF and the DT do not contribute to improvement. In Experiment #2, the DT classifier obtained the highest F1-scores, while the boosting and bagging could not enhance the scores. The RF classifier ranks second, but its boosting and bagging techniques do not improve the F1-scores. The LR classifier comes in third place. Experiment #3 shows that the RF classifier performs the highest F1-scores, but its boosting and bagging models could not contribute to improvement. The LR classifier has better F1-scores than the DT classifier but the boosting and bagging models of the DT classifier do not enhance the F1-scores. In Experiment #4, the RF classifier attained the highest F1-scores, while its boosting and bagging could not improve the scores. The DT classifier achieves better F1-scores than the LR classifier, and its boosting and bagging models help improve the F1-scores. Lastly, in Experiment #5, the LR classifier accomplishes the best F1-scores. The RF classifier ranks second, but its boosting and bagging classifiers could not improve F1 scores. The DT classifier performs well in terms of F1-scores, its bagging classifier helps improve the scores, but it's boosting one does not contribute to improvement.



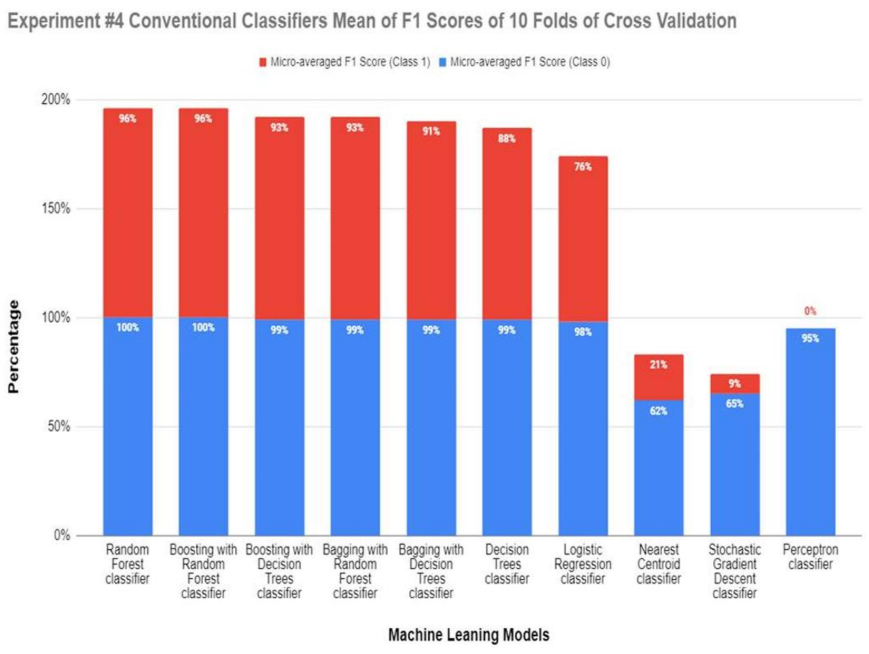
(a)



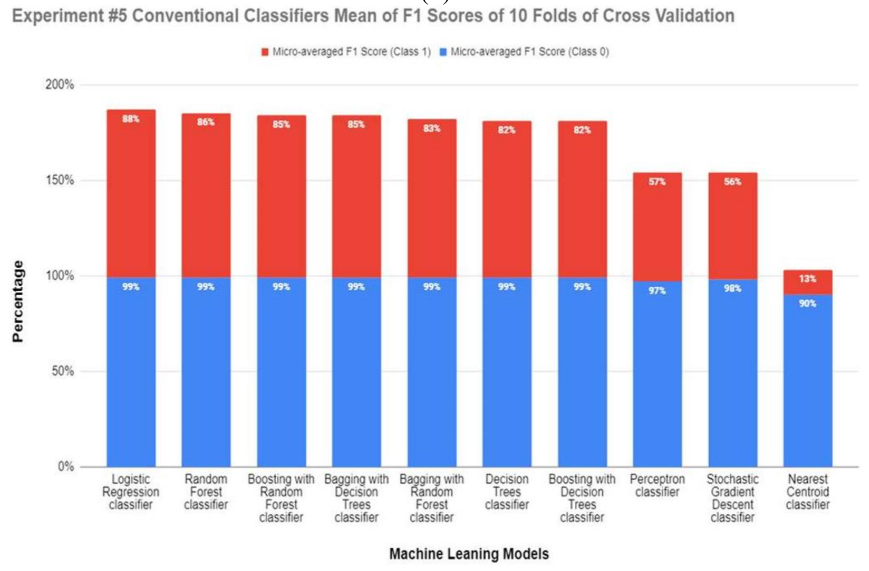
(b)



(c)

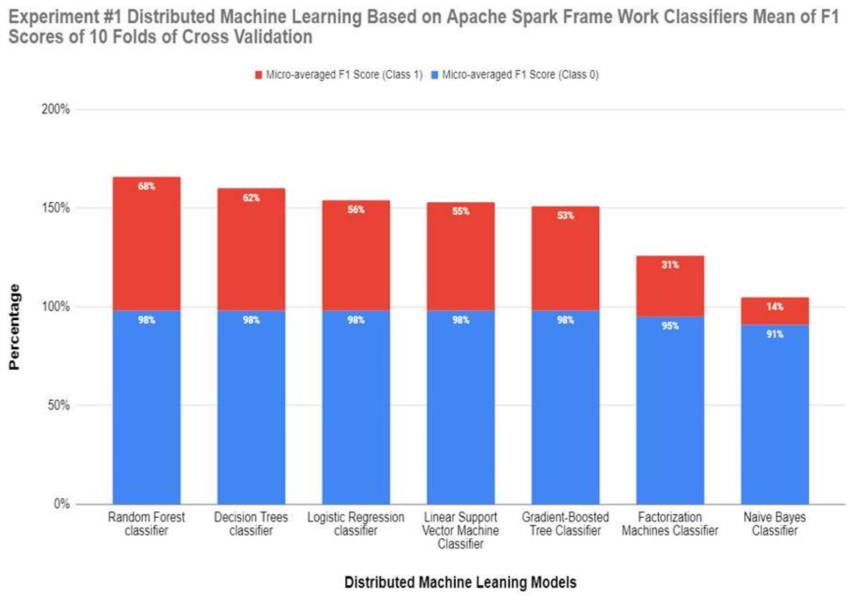


(d)

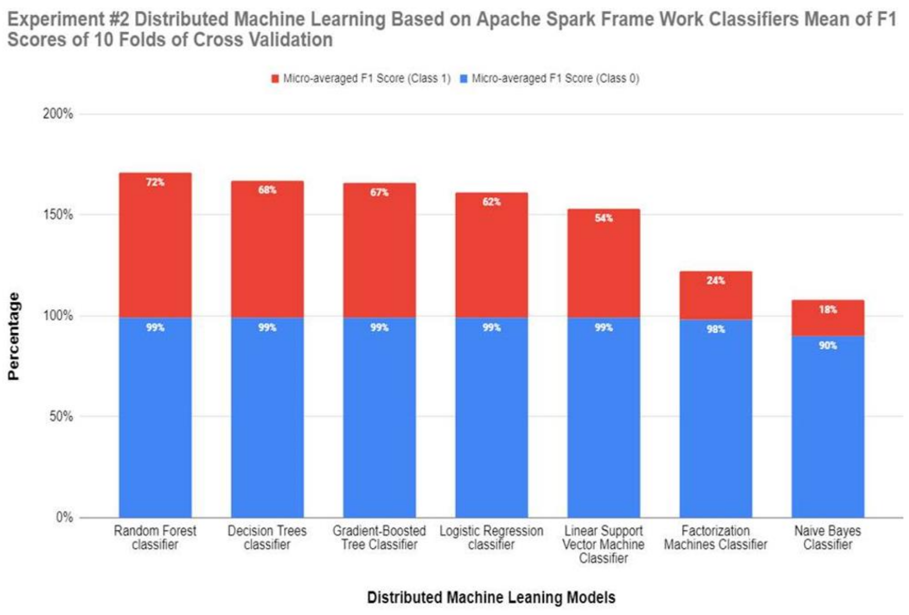


(e)

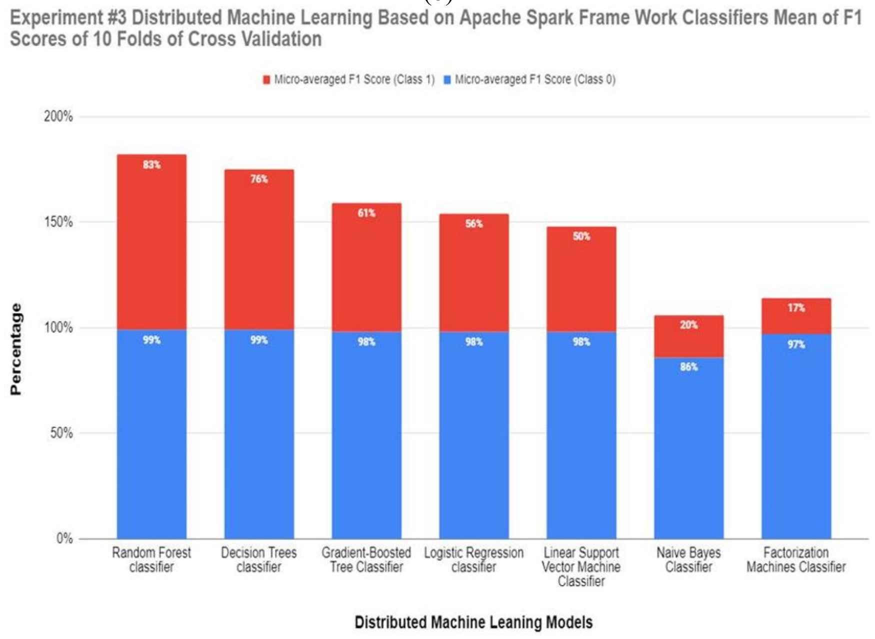
Figure 6. Conventional machine learning classifiers mean of F1-scores of 10 folds of cross-validation



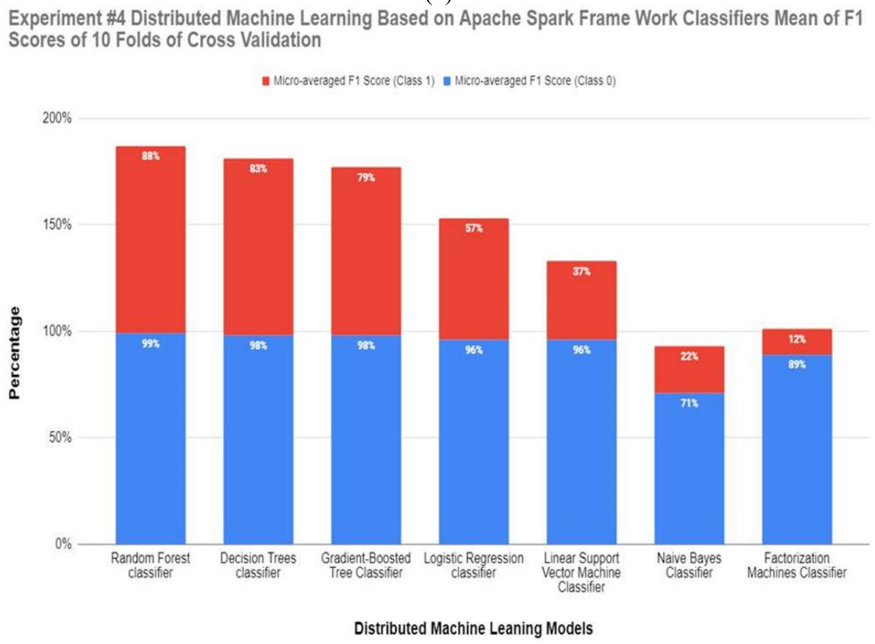
(a)


Experiment #2 Distributed Machine Learning Based on Apache Spark Frame Work Classifiers Mean of F1 Scores of 10 Folds of Cross Validation
Micro-averaged F1 Score (Class 1)
Micro-averaged F1 Score (Class 0)
200%
150%
100%
50%
0%
Percentage
72%
99%
68%
99%
67%
99%
62%
99%
54%
99%
24%
98%
18%
90%
Random Forest classifier
Decision Trees classifier
Gradient-Boosted Tree Classifier
Logistic Regression classifier
Linear Support Vector Machine Classifier
Factorization Machines Classifier
Naive Bayes Classifier
Distributed Machine Leaning Models

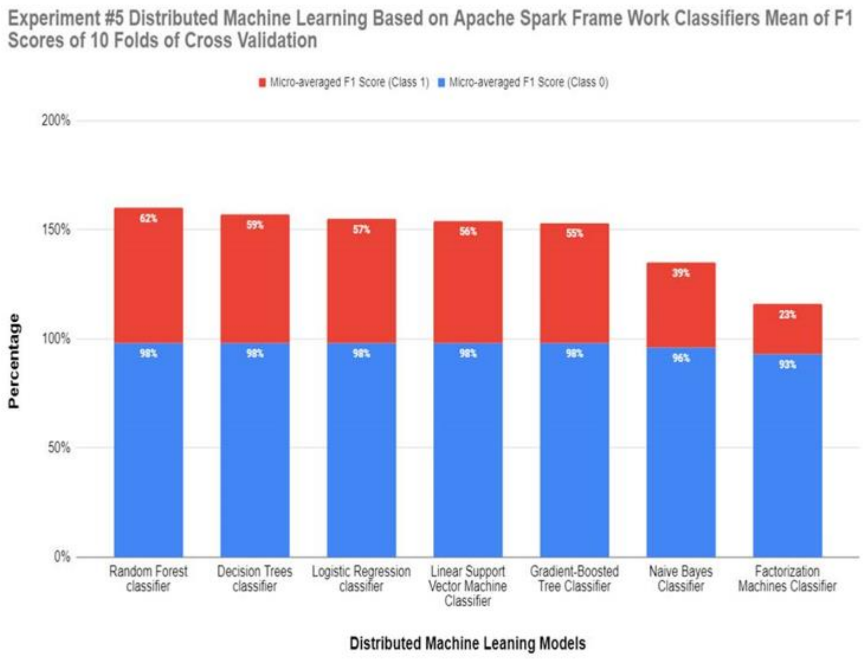
(b)


Experiment #3 Distributed Machine Learning Based on Apache Spark Frame Work Classifiers Mean of F1 Scores of 10 Folds of Cross Validation
Micro-averaged F1 Score (Class 1)
Micro-averaged F1 Score (Class 0)
200%
150%
100%
50%
0%
Percentage
83%
99%
76%
99%
61%
98%
56%
98%
50%
98%
20%
86%
17%
97%
Random Forest classifier
Decision Trees classifier
Gradient-Boosted Tree Classifier
Logistic Regression classifier
Linear Support Vector Machine Classifier
Naive Bayes Classifier
Factorization Machines Classifier
Distributed Machine Leaning Models

(c)


Experiment #4 Distributed Machine Learning Based on Apache Spark Frame Work Classifiers Mean of F1 Scores of 10 Folds of Cross Validation
Micro-averaged F1 Score (Class 1)
Micro-averaged F1 Score (Class 0)
200%
150%
100%
50%
0%
Percentage
88%
99%
83%
98%
79%
98%
57%
96%
37%
96%
22%
71%
12%
89%
Random Forest classifier
Decision Trees classifier
Gradient-Boosted Tree Classifier
Logistic Regression classifier
Linear Support Vector Machine Classifier
Naive Bayes Classifier
Factorization Machines Classifier
Distributed Machine Leaning Models

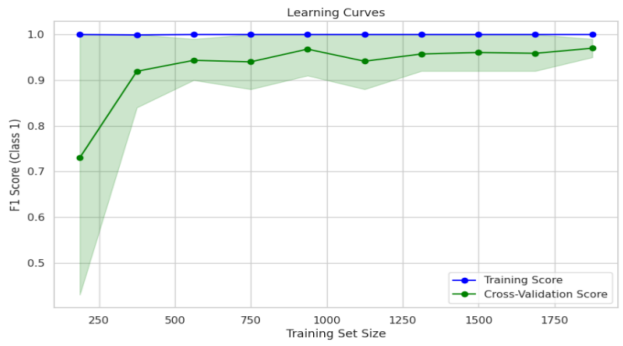
(d)



(e)

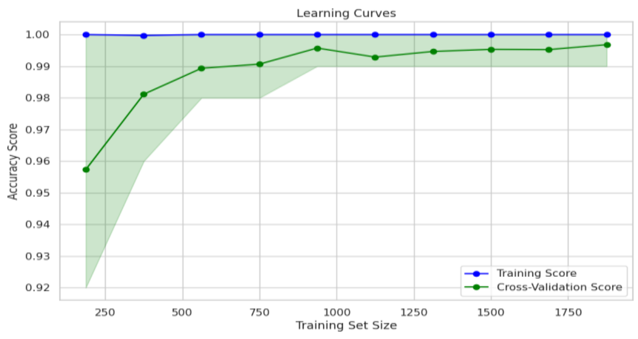
Figure 7. Distributed machine learning classifiers mean of F1-scores of 10 folds of cross-validation

On the other hand, Figure 7 compares different DML classifiers based on the results of the five experiments conducted on Apache Spark. The DRF classifier achieves the highest F1-scores for all experiments, followed by the DDT classifier in second place. The DFM and DNB classifiers have the lowest F1-scores for different experiments. In Experiment #1, the DLR classifier comes in the third place. The F1-scores of DLSVM and DGBT classifiers are comparable and close to the DLR classifier. In Experiments #2, #3, and #4 the DGBT classifier comes in third place. The DLR classifier shows moderate F1-scores, which are better than the F1-scores of the DLSVM classifier. Lastly, in Experiment #5, the DLR classifier fails in third place. The F1-scores of the DLSVM and DGBT classifiers are similar and close to the DLR classifier.

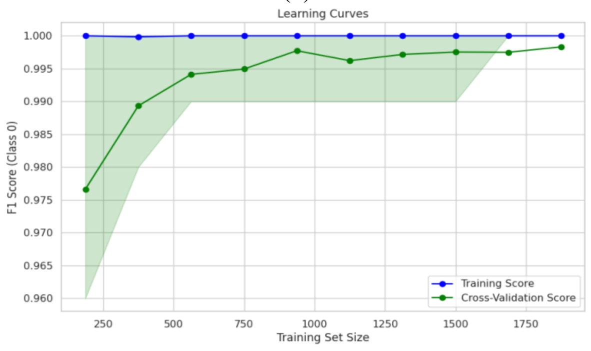


(c)

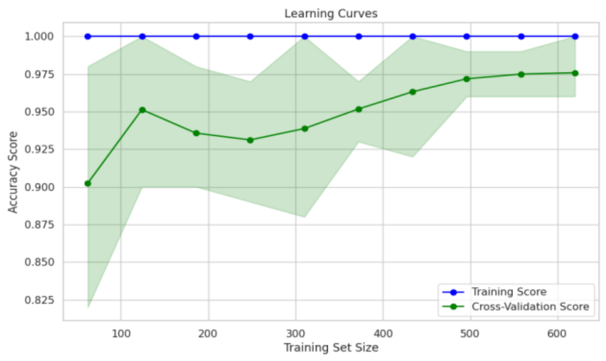
Figure 8. Random Forest (RF) classifier learning curve



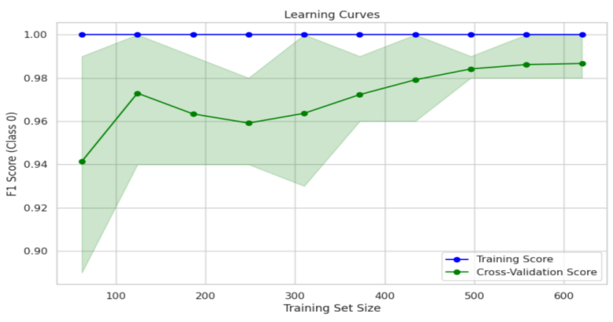
(a)



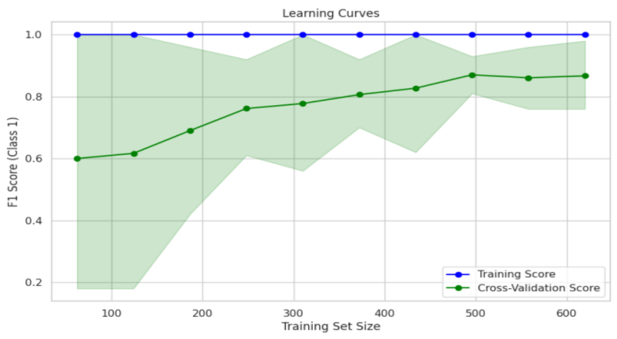
(b)



(a)



(b)



(c)

Figure 9. Distributed Random Forest (DFR) classifier accuracy learning curve

4.3.2 Analyzing CML and DML performance via accuracy learning curves

The accuracy learning curves metric provides useful information regarding how a learner can generalize based on the size of the training data. It serves as a useful tool for model selection, predicting the impact of additional training data, and reducing the computational requirements associated with model training and hyperparameter tuning [36, 37]. In most of the experiments discussed in Figure 4 and Figure 5, the RF/DRF classifier demonstrates the best F1-scores. To gain further insights into the performance of the RF and the DRF classifiers, this subsection examines their accuracy learning curves. Figure 8 illustrates the accuracy learning curves of the RF classifier, while Figure 9 depicts the accuracy learning curves of the DRF classifier.

Figure 8 illustrates the accuracy learning curves of the RF classifier. In Figure 8 (a), the training score remains consistently high at 100%, indicating the model's proficiency. On the other hand, the cross-validation score starts at around 96% accuracy rate, highlighting an initial imbalance in the dataset. However, as the training set size increases, the accuracy steadily improves, reaching its peak at approximately 600 and then plateauing after 1000 training samples. The final accuracy on the cross-validation set is approximately 99.68%. Figure 8 (b) focuses on the F1-score learning curve for real job postings. This figure shows that the training score remains at 100% since the entire dataset is used during training. Regarding the cross-validation score, it initially shows a high F1-score of around 97% with a small training set, emphasizing the dataset's imbalance towards real job postings. As the training set size increases, the F1-score is steadily improved, reaching 75% before plateauing after 1000 training samples. Additional data may not significantly enhance the performance of the model. The final evaluation of the cross-validation set yields an impressive F1-score of approximately 99.83%, affirming the model's overall proficiency and effectiveness.

Figure 8 (c) represents the F1-score learning curve for fake job postings. Similar to the previous figures, the training score remains at 100%. According to the cross-validation score, the initial F1-score is around 70% with a limited training set and it is steadily improved as the training set size increases. The accuracy peaks at approximately 600 and plateaus after 1250 training samples. Additional data may not yield significant enhancements in the performance of the model. The final evaluation of the cross-validation set results in an impressive F1-score of approximately 96.99%, indicating commendable overall performance.

Moving on to Figure 9, it demonstrates the accuracy learning curves of the DRF classifier. Similar to the RF classifier, the training score remains high at 100% as shown in Figure 9 (a). As regards the cross-validation score, the accuracy starts at around 90% with a small training set but it is improved steadily as the training set size increases. The accuracy reaches its peak at approximately 500 and then plateaus after 550 training samples, suggesting that additional data might not significantly enhance the performance of the model. The final accuracy on the cross-validation set is approximately 97.58%.

Figure 9 (b) concentrates on the F1-score learning curve for real job postings with the DRF classifier. Like the RF classifier, the model's training score remains consistently 100%. As regards the cross-validation score, the initial cross-validation phase yields an F1-score of approximately 94% with a small training set, which increases steadily to around 94.5% as the training set size grows. The learning curve plateaus after 500 training samples, indicating that the additional data might not yield significant improvements in the model's performance. The final evaluation of the cross-validation set results in an F1-score of approximately 98.67%, indicating a well-performing model.

Lastly, Figure 9 (c) depicts the F1-score learning curve for fake job postings with the DRF classifier. Like the RF classifier, the training score remains consistently 100%. Regarding the cross-validation score, the initial cross-validation phase with a small training set yields an F1-score of around 60%, which increases steadily to approximately 80% as the training set size grows. The learning curve plateaus after 500 training samples, suggesting that additional data might not significantly enhance the model's performance. The final evaluation of the cross-validation set results in an F1-score of approximately 86.70%, indicating a reasonable performance by the model.

4.4 Discussion of results

Tables 6-9 summarize the results of the above five experiments for both CML and DML classifiers. Based on the findings presented in Tables 6-9 for both CML and DML classifiers, it is evident that the Random Forest (RF) classifier is the most effective classifier in the CML category. Throughout all five experiments, RF consistently achieves high precision, recall, and F1-scores for both classes (0 and 1), along with impressive accuracy rates. Meanwhile, in the DML category, The Distributed Random Forest (DRF) classifier is the top performer. DRF demonstrates superior F1-scores in most experiments and outperforms other classifiers, including the Decision Tree (DDT) when considering the mean F1-scores from 10-fold cross-validation. Notably, DRF shows robust performance in terms of F1-score for class 1 and accuracy, making it the preferred choice for DML tasks. Generally, while RF excels in CML settings, DRF proves to be the go-to classifier for DML, showcasing the importance of selecting the appropriate model based on the specific context and task requirements.

The high accuracy and F1-scores suggest that the framework can effectively detect fake job postings, improving user security and trust across known platforms like LinkedIn and Indeed. Its adaptability to handle imbalanced data, such as the 95% real and 5% fake job ratio, is critical in real-time operations where fraudulent postings are less common but harmful. The scalability provided by Distributed Machine

Learning (DML) models based on big data analytics, allows the framework to handle high traffic on large platforms, making it suitable for platforms with a significant volume and velocity of job postings. Furthermore, the feedback loop enables continuous learning, ensuring that the framework evolves with emerging scam tactics, thus offering long-term protection. This could lead to substantial economic benefits for job platforms by improving their reputation and user retention, while also enhancing user safety by reducing the risk of fraud. However, real-world deployment might face challenges such as false positives or negatives, which need further refinement to minimize errors. Future research should focus on testing the framework across different platforms and datasets to ensure its generalizability and robustness in diverse environments.

Table 6. CML classifiers scores

Exp. No.	Best Class.	Perc. (0)	Rec. (0)	F1-Score (0)	Perc. (1)	Rec. (1)	F1-Score (1)	Acc.
1	RF	99%	100%	99%	100%	79%	89%	99%
2	RF	100%	100%	100%	100%	94%	97%	100%
3	RF	100%	100%	100%	100%	95%	97%	100%
4	RF	100%	100%	100%	100%	100%	100%	100%
5	RF	99%	100%	99%	100%	79%	89%	99%

Table 7. CML classifiers mean of F1-score of 10 folds of cross-validation

Exp. No.	Best Class.	Perc. (0)	Rec. (0)	F1-Score (0)	Perc. (1)	Rec. (1)	F1-Score (1)	Acc.
1	LR	99%	100%	99%	100%	78%	88%	99%
2	DT	100%	100%	100%	95%	96%	95%	100%
3	RF	100%	100%	100%	100%	94%	97%	100%
4	RF	99%	100%	100%	100%	93%	96%	99%
5	LR	99%	100%	99%	100%	78%	88%	99%

Table 8. DML classifiers scores

Exp. No.	Best Class.	Perc. (0)	Rec. (0)	F1-Score (0)	Perc. (1)	Rec. (1)	F1-Score (1)	Acc.
1	DRF	99%	99%	99%	85%	74%	79%	98%
2	DRF	100%	100%	100%	87%	87%	87%	99%
3	DRF	98%	100%	99%	100%	62%	77%	98%
4	DRF	98%	99%	98%	86%	75%	80%	97%
5	DRF	98%	99%	99%	83%	61%	70%	98%

Table 9. DML classifiers mean of F1-score of 10 folds of cross-validation

Exp. No.	Best Class.	Perc. (0)	Rec. (0)	F1-Score (0)	Perc. (1)	Rec. (1)	F1-Score (1)	Acc.
1	DRF	98%	99%	98%	81%	60%	68%	97%
2	DRF	99%	99%	99%	83%	65%	72%	98%
3	DRF	99%	99%	99%	90%	78%	83%	98%
4	DRF	98%	99%	99%	95%	83%	88%	98%
5	DRF	97%	99%	98%	80%	51%	62%	97%

4.5 Scalability of the proposed framework

The scalability of the HJP-ADF is a crucial aspect, particularly in real-world scenarios where online job platforms handle vast amounts of data that grow continuously. As cybercriminals increase the volume and velocity of fraudulent job postings, it becomes essential that the framework can scale efficiently to maintain high performance. HJP-ADF is designed to process large datasets through its integration of distributed processing capabilities, leveraging Apache Spark for Distributed Machine Learning (DML) models. This ensures that as the dataset size increases, the framework can distribute the workload across multiple nodes, enabling parallel processing and reducing the computational time required for fraud detection. This distributed architecture allows HJP-ADF to handle a growing number of job postings, regardless of the size of the dataset, without experiencing significant degradation in performance.

The framework is built to manage both batch and real-time data processing, allowing it to scale as the number of incoming job postings increases. In high-traffic environments, such as large job boards or social media platforms, the ability to handle real-time data streams ensures that fraudulent postings can be detected and flagged immediately. This scalability in real-time processing makes HJP-ADF effective in dynamic, fast-paced environments where job postings and potential scams are generated rapidly.

HJP-ADF also incorporates resource optimization strategies, allowing it to scale without excessive use of computational resources. By dynamically switching between Conventional Machine Learning (CML) for smaller datasets and Distributed Machine Learning (DML) for larger datasets, the framework optimizes resource usage based on the volume of data it processes. This adaptability ensures that even with limited computational resources, the framework can still perform effectively.

Experimental results have shown that the Distributed Random Forest (DRF) model employed in the framework performs well even as the dataset size increases. The model's ability to maintain high accuracy and low latency while processing larger datasets highlights its robustness in scaling across different data volumes. Furthermore, the feedback loop in the framework ensures continuous learning from new data, allowing the system to evolve with the growing volume of fraudulent job postings, and maintaining its effectiveness over time.

The scalability of HJP-ADF ensures that the framework is well-equipped to handle increasing data volumes in real-world applications, maintaining both efficiency and accuracy in detecting fraudulent job postings. This scalability is particularly critical as the digital job market expands and cybercriminals exploit the growing opportunities for fraud.

5. CONCLUSIONS

The growth of social media platforms has opened up new means of communication, collaboration, and engagement. Nonetheless, this increased interconnectivity has resulted in new vulnerabilities to cyber threats, including phishing attacks, malware distribution, and job scams. Combating employment fraud on social media is of paramount importance for cybersecurity and infrastructure security agencies. This work contributes to ongoing efforts aimed at enhancing cybersecurity measures on digital platforms, thereby creating a safer online environment for both users and organizations.

In this work, a real-time cybersecurity framework called Hybrid Job Posting-Analytic and Detection Framework (HJP-ADF) that leverages big data analytics and distributed machine learning is proposed to combat emerging cyber threats. The decision to switch between classification based on conventional and distributed strategies in the HJP-ADF is determined by the dimension and volume of the data. For relatively smaller datasets or less complex data structures, the framework employs a Conventional Machine Learning (CML) approach, utilizing the Random Forest (RF) model for classification tasks. RF is well-known for its effectiveness in handling structured data, offers high precision, recall, and F1-scores, making it suitable for such scenarios. However, as the data dimensionality and volume increase, indicating the need for distributed computing and scalability, the framework seamlessly transitions to Distributed Machine Learning (DML) approach. The Distributed Random Forest (DRF) model is then employed, leveraging its capability to handle large-scale datasets and distributed computing environments efficiently. By dynamically adapting between CML and DML based on the data characteristics, the HJP-ADF ensures high performance and scalability, enabling users and organizations to effectively analyze and detect patterns in job postings across varying scales and complexities.

Future research could explore integrating various data sources, such as social media and dark web information, to capture evolving fraud tactics. Refining machine learning models with advanced techniques like deep learning and natural language processing (NLP) can improve fraud detection accuracy. Incorporating contextual features such as geographical location and industry trends may enhance performance. Addressing the challenge of distinguishing between similar job postings through advanced clustering or anomaly detection can reduce false positives. Additionally, optimizing real-time detection and scalability will help tackle the increasing volume of fraudulent job postings.

ACKNOWLEDGMENT

This paper is supported by Korean Agency for Technology and Standards under the Ministry of Trade, Industry and Energy in 2023 (Grant No.: 1415181638) (Establishment of standardization basis for BCI and AI Interoperability), 1415181629 (Development of International Standard Technologies based on AI Learning and Inference Technologies), and 1415181629 (Development of International Standard Technologies based on AI Model Lightweighting Technologies).

REFERENCES

[1] Network, C.S. (2021). Databook 2020. Consumer Sentinel Network/Federal Trade Commission. https://iapp.org/resources/article/consumer-sentinel-network-databook/.

[2] Better Business Bureau. (2022). Employment Scams. https://www.bbb.org/article/scams/23372-employment-scams.

[3] Better Business Bureau. (2018). Better Business Bureau Study Shows Job Scams Continue to Plague the Unemployed. https://www.bbb.org/globalassets/local-bbbs/council-113/media/news-releases/2018/job-scams-report-bbb-2018.pdf.

[4] Abioye, T., Adegboye, F., Salawu, A. (2020). Cybercrime and its implications on job seekers in Nigeria. International Journal of Business and Management Invention, 9(11): 22-27.

[5] Internet Organised Crime Threat Assessment (IOCTA) Europol. (2020). https://www.europol.europa.eu/publications-events/main-reports/internet-organised-crime-threat-assessment-iocta-2020.

[6] Patchin, J., Hinduja, S. (2020). Cybercrime victimization and mental health outcomes: Results from a nationwide survey. International Journal of Environmental Research and Public Health, 7(9): 3121.

[7] Shteynberg, G., Johnson, K. (2017). Employment scams and the federal trade commission: Impacts on financial and emotional well-being. Journal of Financial Counseling and Planning, 28(1): 106-119.

[8] Real/Fake Job Posting Prediction Dataset from Kaggle. https://www.kaggle.com/datasets/shivamb/real-or-fake-fake-jobposting-prediction.

[9] Wang, C., Zhang, X., Zou, D., Guo, Q. (2020). Detecting fake job postings on LinkedIn: An NLP-based framework. Expert Systems with Applications, 166: 113729.

[10] Benavides, E., Fuertes, W., Sanchez, S., Sanchez, M. (2020). Classification of phishing attack solutions by employing deep learning techniques: A systematic literature review. Developments and Advances in Defense and Security: Proceedings of MICRADS 2019, 152: 51-64. https://doi.org/10.1007/978-981-13-9155-2_5

[11] Wu, Y., Li, G., Lin, J. (2020). A deep learning approach for phishing detection in social media. Future Generation Computer Systems, 107: 150-161.

[12] Wang, Y., Guan, R., Ren, K. (2020). Mining rules for detecting phishing emails. Journal of Information Security and Applications, 54: 102551.

[13] Li, Y., Liang, X., Li, S., Hu, Y. (2021). A data fusion-based method for detecting fake accounts on social media platforms. Journal of Network and Computer Applications, 181: 103039.

[14] Wang, Y., Chen, L., Wang, C., Yu, J., Zhou, A. (2019). Detecting rumors from microblogs with recurrent neural networks. IEEE Transactions on Knowledge and Data Engineering, 31(3): 556-569.

[15] Quihui-Rubio, P., Espinosa, G., V'azquez, D. (2023). Fake job detection with machine learning: A comparison. https://www.researchgate.net/publication/37150873.

[16] Naudé, M., Adebayo, K.J., Nanda, R. (2023). A machine learning approach to detecting fraudulent job types. AI & Society, 38(2): 1013-1024. https://doi.org/10.1007/s00146-022-01469-0

[17] Ullah, Z., Jamjoom, M. (2023). A smart secured framework for detecting and averting online recruitment fraud using ensemble machine learning techniques. PeerJ Computer Science, 9: e1234. https://doi.org/10.7717/peerj-cs.1234

[18] Gupta, G., Raja, K., Gupta, M., Jan, T., Whiteside, S.T., Prasad, M. (2023). A comprehensive review of deepfake detection using advanced machine learning and fusion methods. Electronics, 13(1): 95. https://doi.org/10.3390/electronics13010095

[19] Taherdoost, H. (2023). Enhancing social media platforms with machine learning algorithms and neural networks. Algorithms, 16(6): 271. https://doi.org/10.3390/a16060271

[20] Krishnan, L.P., Vakilinia, I., Reddivari, S., Ahuja, S. (2023). Scams and solutions in cryptocurrencies—A survey analyzing existing machine learning models. Information, 14(3): 171. https://doi.org/10.3390/info14030171

[21] Anita, C.S., Nagarajan, P., Sairam, G.A., Ganesh, P., Deepakkumar, G. (2021). Fake job detection and analysis using machine learning and deep learning algorithms. Revista Geintec-Gestao Inovacao e Tecnologias, 11(2): 642-650.

[22] Khan, S., Qadir, J., Ahmad, F., Bashir, F. (2020). Fake news detection on social media: A data mining perspective. IEEE Access, 8: 99466-99484.

[23] Silva, M.C.D., Tavares, G.M., Gritti, M.C., Ceravolo, P., Barbon Junior, S. (2023). Using process mining to reduce fraud in digital onboarding. FinTech, 2(1): 120-137. https://doi.org/10.3390/fintech2010009

[24] Testas, A. (2023). Distributed Machine Learning with PySpark: Migrating Effortlessly from Pandas and Scikit-Learn.

[25] Ahmadvand, H., Goudarzi, M., Foroutan, F. (2019). Gapprox: Using gallup approach for approximation in big data processing. Journal of Big Data, 6: 1-24. https://doi.org/10.1186/s40537-019-0185-4

[26] Shi, J., Qiu, Y., Minhas, U.F., Jiao, L., Wang, C., Reinwald, B., Özcan, F. (2015). Clash of the titans: Mapreduce vs. spark for large scale data analytics. Proceedings of the VLDB Endowment, 8(13): 2110-2121. https://doi.org/10.14778/2831360.2831365

[27] Mehboob, A., Malik, M.S.I. (2021). Smart fraud detection framework for job recruitments. Arabian Journal for Science and Engineering, 46(4): 3067-3078. https://doi.org/10.1007/s13369-020-04998-2

[28] Brownlee, J. (2020). How to remove outliers for machine learning. Data Preparation.

[29] Andrade, C. (2020). Understanding the difference between standard deviation and standard error of the mean, and knowing when to use which. Indian Journal of Psychological Medicine, 42(4): 409-410. https://doi.org/10.1177/0253717620933419

[30] Li, H. (2024). Machine learning methods. Tsinghua University China, © Tsinghua University Press.

[31] Chatterjee, B. (2024). Distributed machine learning. In ICDCN '24: Proceedings of the 25th International Conference on Distributed Computing and Networking, Chennai, India, pp. 4-7. https://doi.org/10.1145/3631461.3632516

[32] Verbraeken, J., Wolting, M., Katzy, J., Kloppenburg, J., Verbelen, T., Rellermeyer, J.S. (2020). A survey on distributed machine learning. Acm Computing Surveys (CSUR), 53(2): 1-33. https://doi.org/10.1145/3377454

[33] Kundu, R. (2022). F1 score in machine learning: Intro & calculation. V7labs. https://www.v7labs.com/blog/f1-score-guide.

[34] Buhl, N. (2023). F1 Score in Machine Learning. https://encord.com/blog/f1-score-in-machine-learning/.

[35] Manorathna, R. (2020). K-fold cross-validation explained in plain English (For evaluating a model's performance and hyperparameter tuning).

[36] Viering, T., Loog, M. (2022). The shape of learning curves: A review. IEEE Transactions on Pattern Analysis and Machine Intelligence, 45(6): 7799-7819.

[37] Perlich, C. (2010). Learning curves in machine learning. Springer Science+Business Media LLC, pp. 577-580.