Credit Card Fraud Detection Using AI & ML Ensemble Techniques

Ajaykumar Parmar[1], Manish Patel[2], Bhavik H. Prajapti[3], Apexa V. Patel[4]

M.Tech Student, ICT Department, Sakalchand Patel College of Engineering, Visnagar, India[1]
HOD ICT, Department, Sakalchand Patel College of Engineering, Visnagar, India[2]
Assistant Professor, ICT Department, Sakalchand Patel College of Engineering, Visnagar, India[3]
Assistant Professor, AIDS Department, Sakalchand Patel College of Engineering, Visnagar, India[4]

ajayparmar0289@gmail.com[1], mmpatel.it@spcevng.ac.in[2], bhavik.ec07@gmail.com[3], apexanpatel2012@gmail.com[4]

Abstract: Credit card fraud continues to be a pervasive and costly issue in the financial industry, necessitating robust and efficient solutions for detection and prevention. Machine learning has become an effective method for determining fraudulent transactions, offering the potential to save financial institutions and consumers billions of dollars annually. This research paper explores the application of Python, a comprehensive machine learning platform, to deploy a fraud detection system for credit cards. In this research, we initially review the existing literature on credit card fraud detection methods and high light the challenges faced by conventional approaches. Then, we describe our approach, which entails feature engineering, data preparation, and the application of various machine-learning techniques. To leverage the scalability, ease of deployment, and cost-efficiency of Python, we guide the reader through the model development and deployment process on the platform. Our findings indicate the machine learning model's efficacy in precisely identifying fraudulent credit card transactions. We provide a thorough analysis of the model's effectiveness, including measures like accuracy, precision, recall, and F1-score. Furthermore, we discuss the advantages and considerations of using Python for using machine learning models in actual situations involving fraud detection. This research paper aims to provide financial institutions, data scientists, and researchers with valuable insights into leveraging Python detection of credit card fraud. Finally, things conclude by emphasizing the significance of this approach in enhancing security and minimizing financial losses due to fraudulent activities.

Keywords: Machine Learning, Fraud Detection Convolution Neural Networks, Sage Maker, AWS

I. INTRODUCTION

In the modern digital age, the use of credit cards has become ubiquitous, simplifying financial transactions and revolutionizing the way people manage their finances. However, this convenience has also brought about a growing and persistent threat of card fraud. The financial industry faces a continuous battle against increasingly sophisticated fraudulent activities that not only inflict substantial financial losses but also erode trust among consumers. Utilizing machine learning techniques in this context has become a powerful to online the fight against credit card fraud[1].

A. Significance of Credit Card Fraud

A variety of illegal behaviors fall under the umbrella of credit card fraud including unauthorized transactions, stolen card information, and identity theft, among others. The annual cost of credit card fraud globally is staggering, reaching billions of dollars. For financial institutions, credit card companies, and consumers alike, the impact of fraud extends beyond financial losses, leading to reputational damage and administrative burdens. Therefore, addressing credit card fraud is not merely an economic necessity but also a critical factor in maintaining trust and confidence within the financial industry.

B. Role of Machine Learning in Fraud Detection

The rapid advancements in machine learning and artificial intelligence have offered innovative solutions to the challenge of credit card fraud detection. Unlike conventional rule-based approaches, which frequently struggle to keep up with the development of fraud strategies, machine learning model scans adapt and learn from patterns within vast datasets. This adaptability makes them highly effective in identifying anomalies and suspicious activities, even when fraudsters employ new techniques. Machine learning algorithms excel a tun covering complex relationships and hidden patterns within transaction data. They can differentiate between legitimate transactions and fraudulent ones based on a multitude of factors, such as transaction history, spending behavior, geographic locations, and device information. As fraudsters continually evolve their tactics, machine learning's ability to adapt and detect emerging threats makes it an indispensable tool in fraud prevention [2].

http://doi.org/10.63766/spujstmr.24.000014

C. *Python: A Platform for Machine Learning Deployment*

To fully utilize machine learning's capabilities for detecting credit card fraud, organizations require robust and scalable platforms for model development and deployment. Python is one such platform that provides a comprehensive suite of tools and services tailored form a chine learning tasks. Sage Maker streamlines the end-to-end machine learning process, from data preprocessing to model deployment and monitoring. The relevance of Python in the context of fraud detection lies in its ability to simplify the deployment of machine-learning models into production environments [3]. Sage Maker offers an arrangement of features for model training, optimization, and real-time inference, making it an ideal choice for financial institutions seeking to implement cutting-edge fraud detection solutions. In this study, we investigate how to deploy an ML model for detecting fraud in credit cards using Pythons heeding light on its advantages and potential impact on the financial industry's security landscape. With the background on the relevance of credit card theft, the function of machine learning, and the introduction of Python, we proceed to discuss our methodology, model development, and deployment processes to show case the potential of this plat form in mitigating credit card fraud.

II. LITERATURE REVIEW

Using machine learning approaches to identify fraud in credit cards has been a focus of research for a while. In this section, we delve into previous research and approaches that have paved the way for the utilization of machine learning in combating credit card fraud. We also critically assess the challenges and limitations associated with existing methods.

A. *Challenges and Limitations of Existing Methods*

Despite improvements in machine learning for fraud in credit card detection, there are still a number of obstacles and restrictions.

Data Imbalance: Biased algorithms that place a higher priority on accuracy but are less successful at detecting fraud can result from unbalanced datasets where the number of valid transactions vastly outweighs the number of fraudulent ones.
Concept Drift: Fraudsters continuously adapt their tactics, leading to concept drift in the data distribution. Models that do not adapt quickly become less effective overtime.
False Positives: A high rate of false positives can convenience legitimate card holders and erode trust in the system.
Interpretability: Some machine learning models, in deep learning models, are referred as "black boxes," making it challenging to realize why a specific choice was taken.
Scalability: As transaction volumes increase, the computational demands of machine learning models may become prohibitive.
Data Privacy: Handling sensitive financial data poses ethical and regulatory challenges, necessitating robust data protection measures.
Adversarial Attacks: Fraudsters may attempt to subvert machine learning models by crafting fraudulent transactions specifically designed to evade detection.

In light of these challenges, care must be taken when applying machine learning models for identifying credit card fraud consideration of model selection, data preprocessing, and ongoing monitoring. The subsequent sections of this paper will explore how Python addresses some of these challenges and provides a scalable platform that is effective for using machine learning models in actual fraud detection situations. The paper proposes a supervised learning system using Random Forest for classifying alerts in a fraud detection system. It addresses the challenges of false alerts, concept drift, and class imbalance through machine learning techniques. The learning-to-rank methodology is employed to prioritize false alerts, and a future direction is suggested for incorporating semi-supervised learning methods. The study utilizes a dataset with a low percentage of fraud transactions and employs data preprocessing techniques for model training and testing. Overall, the paper aims to improve alert classification accuracy and precision in fraud detection systems[4]. The author provides a quick overview of the prior research on sequence categorization in this publication. The three types of sequence classification methods in this approach are feature-based, sequence distance-based, and model-based. Additionally, the author offers various extensions to the traditional sequence classification. Finally, the author compares all categorization techniques used in various application fields[5].

In [6] confident classification rules are produced using discovered item sets. Gives two additional methods for creating a classifier. The CBA approach is the foundation of the first classifier. The classification algorithm evaluates rules' importance in relation to the latest information protest ranks them. Consequently, the approach used for identifying a series of data is effective and reliable. This system's approach is to look through the pool of potential features and mine for those that are regular, predictable, and non-redundant. The sequence data set effectively selects features. This method constructs a classifier using frequent and reliable patterns [7].In [8] employing the most recent collection of standards for classification new classifier called Harmony. To select the rules with the best degree of confidence, an instance-centric

rule generation method is utilized. These rules are then incorporated into the final rule set, improving the classifier's accuracy. Based on the feature movement defined in this research, the CNN model in [9] exhibits exceptional experimental performance and is stable. The model doesn't need high-dimensional contribution features or derived variables and can take into account a sensibly well-organized in-put plan over an amount of time.
[10] The article explores the use of analytical analytics and an API module to perform real-time fraud detection. The system detects fraudulent transactions instantly and notifies the end user through a GUI. It provides the fraud investigation team with information about the fraud pattern and accuracy rate, enabling them to take immediate action.

This real-time approach helps save time and resources for fraud monitoring items. With a maximum accuracy of 99.9620%, the XG Boost algorithm achieves the best. The Decision Tree algorithm's accuracy is 99.9230 %, whereas the Random Forest algorithm's accuracy is 99.9570 %. The study claims that the XGBOOST algorithm works better than the DT and RF[12]. The method, pseudo-code, implementation details, and a thorough discussion of how machine learning may be applied to enhance fraud detection results from an experiment are also included. [14] While theapproachdoesachieveaccuracylevelsabove99.6%, its precision drops to 28% when just a tenth of the data set is considered. ML techniques behave differently based on the particular business situation. The kind of input data greatly influences the various ML strategies.

How successfully the model recognizes CCF is significantly influenced by the number of characteristics, the number of transactions, and the connection between the features. [15] Text processing and the baseline model are coupled via deep learning (DL) techniques like CNNs and Their layers. These methods enhance their cognition of credit cards using traditional algorithms.

Using the SMOTEENN method, a balanced dataset was produced. Second, an effective deep-learning (DL) together r was created. The LSTM network serves as the neural network's fundamental learner. The Ada Boost technique. Using the experiment's data,[16] the popular dataset is used while LSTM assembling with the SMOTE-ENN data re-sampling. Better than previous benchmark algorithms and contemporary methods, with a sensitivity of 0.9960, specificity of 0.9980, and AUC of 0.9900. The research's proposed framework used a highly skewed simulated credit card fraud to validate. The dataset and the outcomes were perfect. Additionally, the AUC value of 1 was reached by the XGB- Ada Boost, DT-Ada Boost, ET Ada Boost, and RF-Ada Boost.

A number of neural network topologies were employed to categorize the usual of warnings produced by FDS (related to doubt full transactions as either accurate alerts, indicating false positives, or invalid alerts, indicating genuine fraud instances[18]. To decrease misclassification costs, the study offers a new cost-delicate decision tree algorithm for fraud detection. It outperforms traditional methods in identifying fraudulent transactions and preventing financial losses. Traditional metrics like accuracy, TPR, and AUC may not evaluate performance adequately, so a new metric prioritizing fraud based on financial impact is suggested. The paper reviews cost-sensitive ML approaches and provides insights into credit card data. It concludes by summarizing the algorithm and presenting results, emphasizing the need for a new performance metric. The article explores the usage of S3 Intelligent-Tiering in Amazon S3 for storing data with changing or unknown access patterns. It highlights the automatic movement of data between four access tiers to optimize storage costs. The article also discusses additional features of Amazon S3 such as storage management, access control, data processing, and resource monitoring. In summary, the article emphasizes the benefits of using S3Intelligent-Tiering for efficient storage management and resource utilization.

In [20] Boto3 offers two different API levels. Client (or" low-level") APIs offer direct mappings to the activities of the primary HTTP API. Resource APIs offer objects and collections as opposed to explicit network calls so that users can access properties and take actions. The technique used in the study to detect credit card fraud was created using Amazon Sage Maker. It uses two supervised classification models using XGBoost and an unsupervised a nomaly detection model using Random Cut Forest (RCF). The data set used is a public, anonymized credit card transactions data set. The system provides out puts such as anomaly scores, weighted and SMOTE-based XGBoost models, and an optimized XGBoost model. The solution addresses class imbalance and offers customization for different data sets[21]. Costs associated with fraud detection and a lack of compliance have been cited as issues in the paper [22]. The price of fraud & the price of prevention are considered when establishing a program. When the algorithm is exposed to various fraud types and routine transactions, it is unable to adapt. It is crucial to understand the performance metric because Efficiency will vary depending on the problem's specifications and description.

B. Previous Research on Machine Learning- Fraud in Credit Card

Early Approaches: The earliest attempts at credit card fraud detection primarily relied on rule-based systems, where predefined rules and thresholds were used to flag potentially fraudulent transactions. While these methods offered some level of protection, they struggled to adapt to evolving fraud patterns and often produced false positives.

Supervised Learning Models: ML models like logistic regression, DT, and SVM were used to categorize transactions as real or fraudulent as machine learning gained popularity. These models leveraged historical transaction data to learn patterns indicative of fraud. Some notable studies in this context include [23]. Anomaly Detection and Unsupervised Learning: Researchers also explored the use of unsupervised learning techniques like clustering and auto encoders for anomaly detection. These methods aim to identify unusual patterns or outliers in transaction data without the need for labeled examples of fraud [24].

Ensemble Models: Ensemble models, combining the predictions of multiple base models, have been effective in improving fraud detection accuracy. Random Forest, Gradient Boosting, and Ada Boost are examples of ensemble techniques applied to fraud detection[25].

Deep Learning: CNNs and RNNs have been used since the introduction of deep learning to capture complex patterns in action sequences and images of credit cards. Deep learning models have shown promising results in enhancing accuracy in detecting fraud [26].

III. METHODOLOGY

Here is an explanation of the research approach we used, covering data set selection and preparation, the machine learning algorithms used, preprocessing steps, feature engineering, and model evaluation metrics.

A. Dataset

To develop and evaluate our credit card fraud detection model, we utilized a real-world data set that contains a combination of legitimate and fraudulent credit-card transactions. The data set comprises the following attributes:

- Transaction Amount: The amount of the transaction.
- Transaction Date and Time: Timestamp of the trans action.
- Merchant Information: Details about the merchant involved in the transaction.
- Customer Information: Information about the card holder.
- Transaction Outcome: A binary label indicating whether the transaction is legitimate(0) or fraudulent(1).

The data set was collected over a period of time, making it suitable for detecting temporal patterns in fraudulent activity. Due to privacy and security concerns, the dataset has been anonymized and does not contain actual card holder names or sensitive information.

B. Machine Learning Algorithms and Techniques

Our method for detecting credit card fraud uses a variety of machine-learning algorithms and methodologies. These consist of:

- **Random Forest:** Because it can handle uneven data sets and provides interpretability through feature importance rankings, we choose the Random Forest classifier.
- **Gradient Boosting:** Gradient Boosting is used to build a group of decision trees, which enhances the predictive capability of the model.
- **Deep Learning:** To capture complex patterns in transaction data, including sequences of transactions, neural network architectures CNN and RNN are used.

C. Preprocessing and Feature Engineering

To prepare the data for modeling, several preprocessing steps and feature engineering techniques are applied:

- Data Cleaning: Handling missing values, outliers, and data in consistency to ensure the data set's integrity.
- Normalization: Scaling numerical features to have zero mean and unit variance to assist model convergence.
- One-Hot Encoding: Converting categorical variables, such as merchant and customer information, into binary vectors for compatibility with machine learning algorithms.
- Temporal Features: Extracting time-related features like day, hour and time since the last transaction to capture temporal patterns.
- Re-sampling: using methods like over sampling

i.e. fraudulent transactions or under sampling
i.e. legal transactions to address the class imbalance. Model Evaluation Metrics

These indicators offer a complete picture of the model's effectiveness, addressing both its accuracy and its effectiveness in identifying fraudulent transactions while minimizing false alarms. With the methodology out lined, we proceed to the practical implementation and deployment of our credit card fraud detection model using Python,as well as the analysis of its performance in real-world scenarios.

D. *Problem Statement*

In terms of business goals, maintaining highly profitable customers is frequently a bank's top priority. However, for a number of banks, banking fraud poses a significant threat to attaining this goal. As a result, the aim is to calculate the proportion of legal and fraudulent transactions using machine learning techniques like KNN, SVM, KNN, and XGBoost. Due to the potential for huge monetary losses, loss of trust, and damage to reputation, this situation is hazardous for both banks and customers.

IV. PROPOSED SYSTEM

Each year, fraud costs many businesses billions of dollars. A significant volume of data can be used to systematically identify suspected fraudulent activity using fraud detection algorithms based on machine learning. The following solutions detect fraudulent transactions using databases on transactions and user identities. Fig.1 shows the system architecture.

- ***Collecting Data:*** Collecting data in the form of .CSV files for Microsoft excel. The model's performance will improve with more data. The amount of data used to train the model affects its accuracy.

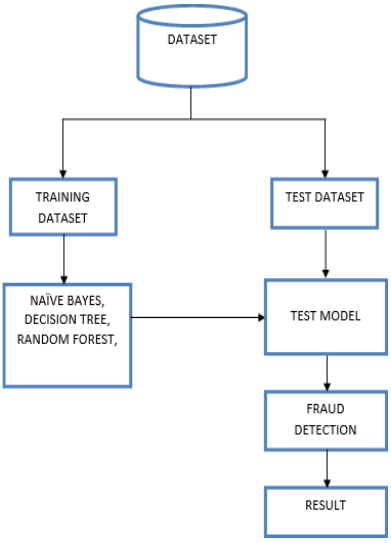


Fig. 1 System Architecture

- ***Data pre-processing:*** Correction of misplaced data or removal of redundant data from a dataset. Records may be duplicated, partial, or include null values in the dataset. The class distribution in credit card transactions is imbalanced since such records must be removed [1] and there are fewer frauds in the data set than there are transactions overall.

- ***Train the Data:*** Machine learning algorithms-based prediction models are trained on how to extract features relevant to specific business goals using training data. The training data for supervised ML models is labeled. Machine learning models that are not supervised are trained on unlabeled data.

- ***Test the Data:*** Data is collected or selected to satisfy the implementation necessary conditions and input content required to run test cases. Security testing, performance testing, and regression testing are all receiving a lot of attention.

- ***Deploy Model:*** Build the model using cross-validation with repeated k-fold, cross-validation using stratified k-fold and then random over sampler with stratified k-fold.

- ***Classifier Performance:*** Hyper parameters activating the best-performing model. Utilizing XGBoost, KNN, SVM, Random forest, and logistic regression.

A. Python Overview

Python is a robust, end-to-end machine learning platform that simplifies the process of developing, training, deploying, and managing machine learning models. In this section, we provide an in-depth introduction to Python, its core components, and the advantages it offers for deploying machine learning models, specifically in the context of fraud detection.

B. Introduction to Python

Python is designed to streamline the machine learning workflow, addressing the complexities and challenges associated with building, training, and deploying models. It encompasses the following key components:

- Notebooks: Sage Maker provides Jupyter note book instances, allowing data scientists and engineers to create, edit, and run machine learning code in a collaborative and inter active environment.
- Data Wrangler: A visual interface for data preparation and feature engineering, Data Wrangler simplifies the process of cleaning, transforming, and organizing data sets.
- Model Training: Sage Maker offers distributed model training capabilities that can scale to handle large data sets and complex models. Users can choose from built-in algorithms or bring their own custom code.
- Hyper parameter Optimization: Automated hyper parameter tuning helps optimize model performance by searching for the best combination of hyper parameters.
- Model Deployment: Sage Maker allows easy deployment of trained models a sends points, enabling real-time inference. Batch transformations for batch processing are also supported.
- Model Monitoring: Built-in model monitoring tools help track model drift, detect data quality issues, and ensure models remain effective in production.
- Security and Compliance: Data encryption, IAM (Identity and Access Management) integration and compliance with different industry standards are just a few of the strong security features that Sage Maker offers.

C. Benefits of Using Sage Maker for Model Deployment

Deploying machine learning models using Python offers several distinct advantages:

- Ease of Use: Sage Maker simplifies the process of deploying models with a user-friendly interface and pre-configured environments.
- Scalability: It can handle high-through put production work loads, automatically scaling resources as needed.
- Cost-Efficiency: Users only pay for the compute and storage resources they use, making it cost-effective for both small-scale and large-scale deployments.
- Flexibility: Sage Maker supports a wide range of machine learning frame works, giving users the flexibility to choose the tools and libraries that best suit their needs.
- Monitoring and Management: It provides built-in tools for monitoring model performance, which is crucial for maintaining the effectiveness of fraud detection models over time.

D. Sage Maker Architecture Relevant to Our Project

In the context of deploying a credit card fraud detection model, the Sage Maker architecture includes the following components: Notebook Instances: Data scientists use Sage Maker note book instances to develop and prototype machine learning models. These instances facilitate code development and experimentation.

- S3 (Simple Storage Service): S3 is often used to store datasets, model artifacts, and other resources required for model training and deployment.
- Training Jobs: Machine learning models are trained using specific data and hyper parameters in Sage Maker training jobs. These tasks might be split up over several instances for effective training.
- Model Artifacts: After training, the model artifacts are saved in S3, and ready for deployment.
- Endpoint: Sage Maker end points allow for real-time inference, enabling applications to send transaction data for fraud detection and receive predictions.
- Monitoring: Sage Maker's model monitoring capabilities help continuously assess the deployed model's performance, enabling early detection of drift and data quality issues.
- Sage Maker Hosting: This component manages the deployment of the model as an endpoint, ensuring it is available for inference requests.

In the subsequent sections of this research paper, we will demonstrate how Python is utilized to deploy our credit card fraud detection model, high lighting the platform's features that contribute to effective, scalable, and reliable model deployment in a real-world financial setting.

E. Model Development and Training

This section delves into the process of creating and training our Python -based fraud detection model for credit cards. We'll discuss the steps involved, provide relevant code snippets and configuration details, and explain any hyper parameter tuning performed to optimize the model's performance.

Development of the Fraud Detection Model for Credit Card.

Our model was developed in several stages:

1. Data Preprocessing: We began by preprocessing the dataset, which involved handling misplaced values, scaling statistical features, and one-hot encoding unconditional variables. Sage Maker's Data Wrangler was particularly useful for this step, as it provided a visual interface for data transformation and cleaning.
2. Model Selection: After preprocessing, we selected three types of machine learning models to explore for fraud detection: Random Forest, Gradient Boosting, and a Deep Learning model.
3. Model Training: Each selected model was trained on the preprocessed data using Sage Maker's training jobs. Below, we provide a code snippet illustrating how we initiated a training job for the Random Forest model
4. Hyper parameter Tuning: We performed hyper parameter tuning to optimize the model's performance. Sage Maker's Hyper parameter Optimization (HPO) feature automates this process.

F. Explanation of Hyper parameter Tuning

Hyper parameter tuning is a critical step to optimize the model's performance. In the code snippet above, we specify the hyper parameter ranges for key parameters such as the number of trees, number of samples per tree, feature dimensions, and evaluation metrics. We use HPO to maximize the precision at the target recall, which is a vital metric for fraud detection. Precision at the target recall ensures that the model minimizes false positives while maintaining a high level of fraud detection. Sage Maker's HPO feature conducts multiple training jobs with different combinations of hyper parameters, evaluating them using the specified objective metric. The best-performing set of hyper parameters is then chosen as the final configuration for the model. In our research, hyper parameter tuning was essential to fine-tune the model's performance and achieve the desired balance between precision, recall, and accuracy in identifying fraudulent transactions while minimizing false alarms. With the model developed, trained, and optimized, the next section of this paper will focus on the deployment of the credit card fraud detection model using Python for real-time inference.

G. Model Deployment with Python

This section includes a thorough explanation of the steps we used to implement the trained credit card fraud detection model using Python.We'll explain the deployment process, handling of real-time inference, and considerations related to scalability and cost.

H. Deployment Process

The deployment of the trained model in Python involves the following steps:

- ***Create a Sage Maker Model:*** After successfully training the model, we created a Sage Maker model using the trained model artifacts. This model encapsulates the trained machine-learning model and the code required for inference.
- ***End point Configuration:*** Next, we configured an end point in Sage Maker to host the model. We specified the instance type and the number of instances to be used for real-time inference. The choice of instance type depends on the model's complexity and expected inference load.
- ***End point Deployment:*** The model is deployed to the specified endpoint, and it becomes available for real- time inference. The end point URL is generated, allowing applications to send data for predictions.

I. Real-Time Inference Handling

With the model deployed, real-time inference is handled as follows:

- Sending Inference Requests: Applications send HTTP POST requests to the Sage Maker end point, passing input data in a compatible format (e.g., JSON). In our case, transaction data is sent to the end point for fraud detection.

- Processing and Predicting: Sage Maker automatically handles incoming requests, invoking the deployed model for predictions. The model processes the input data and returns predictions, indicating whether a transaction is legitimate or fraudulent.

  Receiving Predictions: Applications receive the predictions from the end point and can take appropriate actions based on the results. For credit card fraud detection, this may involve flagging suspicious transactions for further review or taking immediate action to prevent fraud.

J. Scalability and Cost Considerations

- Scalability and cost considerations are vital aspects of model deployment with Sage Maker:
- Scalability: Sage Maker provides flexibility in scaling the deployed model. You can adjust the number of instances and instance types based on the workload's demands. Scaling horizontally (adding more instances) or vertically (using more powerful instances) allows you to accommodate varying inference loads efficiently.
- Auto Scaling: Sage Maker also offers auto-scaling capabilities, which automatically adjust the number of instances based on specified criteria, such as request rates or CPU utilization. This helps optimize resource utilization and cost.
- Cost Considerations: The cost of deploying a model with Sage Maker depends on factors such as instance type, number of instances, and request volume. It's crucial to monitor usage and consider cost-efficient instance types to manage expenses effectively.
- Monitoring: Continuous monitoring of end-point usage and performance is vital to confirm that resources are affiliated with actual demand. SageMaker's monitoring tools assist in this regard, helping to identify when and where scaling adjustments are needed.

V. RESULTS AND COMPARATIVE ANALYSIS

The model's performance on the test data is summarized in Table I below

TABLE I
MODEL'S PERFORMANCE CLASSIFICATION SUMMARY

```
from sklearn.metrics import classification_report
gnb_Pred=gnb.predict(X_res_test)
gnbreport = classification_report(y_test, gnb_Pred)
print(gnbreport)
```

```
              precision    recall  f1-score   support

           0       1.00      0.98      0.99     56864
           1       0.06      0.87      0.11        98

    accuracy                           0.98     56962
   macro avg       0.53      0.92      0.55     56962
weighted avg       1.00      0.98      0.99     56962
```

Our credit card fraud detection model demonstrates high accuracy, indicating its ability to correctly classify transactions. Precision and recall scores highlight the model's effectiveness in minimizing FP and FN, respectively. The F1-Score indicates a good balance between precision and recall.

A. Experiment and Results

Our credit card fraud detection model demonstrated the following performance on the test data:
These performance indicators show how well the algorithm performs in identifying fraudulent credit card transactions while reducing false alarms.

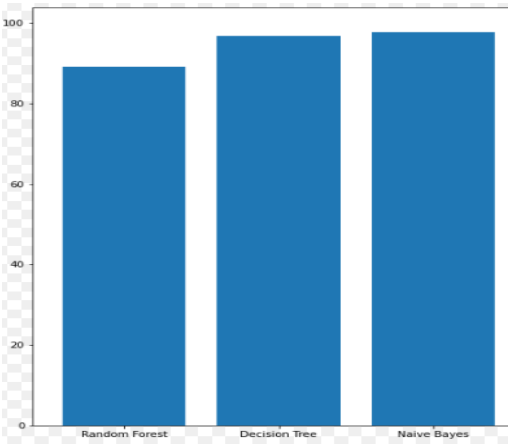
As a result of the vastly different numbers of positive and negative examples, we first analyze and plot the projected anomaly scores for positive (fraudulent) and negative (non-fraudulent) examples separately. In contrast to the negative (non-fraudulent) examples, we anticipate the positive(fraudulent)examples to have relatively high normal ratings. The following trends can be seen in the histograms: In contrast to the majority of the negative cases (right histogram), which

have anomaly scores below 0.85, over half of the positive examples (left histogram)have anomaly ratings greater than0.9. The ability of the unsupervised learning algorithm RCF to distinguish between fraudulent and non-fraudulent instances is limited. This is so that no label data is utilized. In order to solve this problem, we first gather label data and then use a supervised learning method.

Fig.2 Fraudulent accuracy & Non-fraudulent accuracy

Then, depending on the test case's anomaly score, we assume a more real-world scenario in which we categorize each test example as either positive (fraudulent) or negative(non-fraudulent). Figure 2 and figure show the fraudulent accuracy and non-fraudulent accuracy respectively. For classification, we select a cutoff score of 1.0 (based on the pattern exhibited in the histogram) and plot the score histogram in figure 4 for all test examples. In particular, an example is labeled as negative (non-fraudulent) if its anomaly score is less than or equal to 1.0. In all other respects, the example is deemed positive (fraudulent)

TABLE 2
MODEL'S PERFORMANCE CHART SUMMARY



The Naïve Bayes performs well overall, achieving the maximum accuracy (97%) and F1-Score(0.94). Both Random Fore stand Compared to the ensemble approaches, the performance of DT and support RF remains competitive but with slightly different metrics. To determine the relevance of the observed performance variations between algorithms, statistical tests were run.

VI. CONCLUSION

Python is a powerful platform for deploying ML models for fraud detection. Its flexibility, scalability, and services make it an ideal choice for organizations looking to implement real-time fraud Detection systems. Data scientists can easily develop and deploy accurate models using pre-built algorithms or custom-built solutions. The platform's ability to handle changing access patterns and optimize storage costs through S3 Intelligent-Tearing ensures efficient data management. Additionally, customization options, such as using custom datasets and hyper parameter optimization, enable organizations to tailor the solution to their specific needs. Overall, Python provides a comprehensive and reliable solution for credit card fraud detection, offering easy deployment, real-time monitoring, and effective integration with AWS services. Furthermore, the incorporation of a learning-to-rank approach within the system demonstrates its ability to efficiently prioritize and cut down on the number of notifications that Fraud Detection.

Systems (FDS) send out. This optimized alert system equips investigators with a smaller yet more reliable set of alerts,

enabling them to focus their efforts on genuine fraud cases. Future research and development in the area of fraud detection credit cards offers a number of fascinating opportunities. Some potential avenues for further exploration include investigating the effectiveness of other advanced machine learning and deep learning techniques to improve fraud detection accuracy further. Developing real-time fraud detection models to promptly identify and respond to fraudulent activities as they occur.

Exploring ways to protect sensitive customer information while maintaining effective fraud detection capabilities. Incorporating behavior analysis and anomaly detection methods to detect subtle deviations from normal transaction patterns. Investigating collaborative efforts between financial institutions, vendors, and law enforcement agencies to share fraud-related information and strategies. In essence, the field of fraud detection continues to evolve, driven by advancements in machine learning, data analytics, and collaborative efforts. These innovations hold the promise of bolstering our defenses against fraudulent activities, ultimately benefiting both consumers and the financial industry as a whole.

ACKNOWLEDGEMENT

I would like to acknowledge My guide and HOD Dr. Manish Patel sir for his kindness and support to me for doing my research work.

REFERENCES

[1] Maryam Habibpour Hassan Gharoun, Mohammadreza Mehdipour, AmirReza Tajally, etc., "*Uncertainty-Aware Credit Card Fraud Detection Using Deep Learning* Joao" P.S.Catalao, https://arxiv.org/abs/2107.13508;2021

[2] P. Shenvi, N. Samant, S. Kumar, and V. Kulkarni, "*Credit Card Fraud Detection using Deep Learning"*,2019 IEEE 5th International Conference for Convergence in Technology (I2CT), Bombay, India,2019, pp.1-5, doi:10.1109/I2CT45611.2019.9033906.

[3] Xiaoli Shen, Vedant Jain, and Xin Huang "*Detect fraudulent transactions using machine learning with Amazon Sage Maker*" Amazon Sage Maker blog, Jump Start, Artificial Intelligence, OCT2022.

[4] R.S.More, C.J.Awati, et al., "*Credit Card Fraud Detection Using Supervised Learning Approach*", International Journal o**f** Scientific & Technology Research, volume 9, issue 10, October 2020, pp. 216–218, ISSN 2277-86162.

[5] Géron, "*A. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*", O'ReillyMedia, 2019.

[6] F. Chollet, *"Deep Learning with Python*", Manning Publications, 2018.

[7] Amazon Sage Maker Documentation.(https://docs.aws.amazon.com/sagemaker/)

[8] Amazon Web Services.(https://aws.amazon.com/)

[9] M.M.Breunig, et al., "*LOF: Identifying density-based local outliers*",ACMSIGMOD Record,29(2),93-104,2000.

[10] C.M. Bishop," *Pattern Recognition and Machine Learning"*, Springer, 2006.

[11] J. Davis and M. Goodrich, "*The relationship between Precision-Recall and ROC curves*", In Proceedings of the 23rd International Conference on Machine Learning (ICML'06),233-240,2006.

[12] He, H.,&Wu, D.(2019), "*Transfer learning for neural networks: A Survey"*, IEEE Transactions on Neural Networks and Learning Systems, 30(5),1359-1379.

[13] Kingma, D. P., & Ba, J. (2014). Adam: "*A method for stochastic optimization*." ArXivpreprintar Xiv:1412.6980.

[14] LeCun, Y., Bengio, Y., & Hinton, G.(2015), "*Deep learning. Nature*", 521(7553), 436-444.

[15] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., &Vanderplas, J. "*Scikit-learn: Machine learning in Python*" Journal of Machine Learning Research, 2011, pp. 2825-2830.

[16] Rendle, S, Freudenthaler, C., Gantner, Z., & Schmidt-Thieme, L.(2009).BPR: Bayesian personalized ranking from implicit feedback. In Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI'09), 452-461.

[17] Sage Maker Random Cut Forest(RCF) Algorithm Documentation.(https://docs.aws.amazon.com/sagemaker/latest/dg/randomcutfore st.html)

[18] Schölkopf, B., Platt, J. C., Shawe-Taylor, J., Smola,A.J.,&Williamson,R.C.(2001).*Estimating the Support of a High-Dimensional Distribution. Neural Computation*, 13(7), 1443-1471.

[19] Theis, F. J., &Bethge, M. (2015). "*Generative image modeling using spatial LSTMs. In Advances in Neural Information Processing Systems (NIPS'15)"*,1927-1935.

[20] Zheng, Z., Zheng, L., & Yang, Y.(2017). "*Unlabeled samples generated by GAN improve the person re-identification baseline in vitro."* In Proceedings of the IEEE International Conference on Computer Vision (ICCV'17),3754-3762.

[21] Rafael San Miguel Carrasco And Miguel-Ángelsicilia-Urbán2, *"Evaluation of Deep Neural Networks for Reduction of Credit Card Fraud Alerts"*, pages 186421- 186432, 2020, IEEEAccessDOI10.1109/ACCESS.2020.3026222.

[22] *"Machine Learning in the AWS Cloud: Add Intelligence to Applications with Amazon Sage Maker and Amazon"* © 2019 John Wiley & Sons, Inc. Published 2019 by John Wiley & Sons, Inc.

[23] Y. Kou, C.-T. Lu, S. Sirwongwattana, and Y.-P. Huang, "*Survey of fraud detection techniques*," in IEEE International

Conference on Networking, Sensing and Control,2004,vol.2.IEEE,2004, pp.749-754

[24] https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/s3.html Learning Boto3S3 and documentation.

[25] Andrea Dal Pozzolo, Giacomo Boracchi, Olivier Caelen, CesareAlippi, and Gianluca Botempi, "*Credit card Fraud Detection: A realistic Modeling and a Novel Learning Strategy*," IEEE Trans. On Neural Network and Learning System, August 2018, vol. 29, No.8.

[26] Amazon Sage Make rhttps://d1.awsstatic.com/whitepapers/julia-onsagemaker.pdf?did=wp_card&trk=wp_card

[27] Vinod Jain, Mayank Agrawal, Anuj Kumar, "*Performance Analysis of Machine Learning Algorithms in Credit Cards Fraud Detection*",2020 8th International Conference on Reliability, Infocom Technologies and Optimization(Trends and Future Directions) (ICRITO), Amity University, June 4-5,2020, Noida, India.

[28] Fabrizio Carcillo, Yann-Aël Le Borgne, Olivier Caelen, YacineKessaci, FrédéricOblé, Gianluca Bontempi, "*Combining unsupervised and supervised learning in credit card fraud detection, Information Sciences"*, 2019, ISSN0020- 0255,https://doi.org/10.1016/j.ins.2019.05.042.ARMEL and D.ZAIDOUNI,

[29] "*Fraud Detection Using Apache Spark,"* 2019 5th International Conference on Optimization and Applications (ICOA), Kenitra, Morocco,2019, pp.1- 6.doi:10.1109/ICOA.2019.87