International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Analysis of Data Security Prognostic Method Utilizing Cognitive Machine Learning Behavior

Mr.Prabhanjan Chaudhari[1], Dr. Guddi Singh[2], Dr. Amit Bhusari[3]

Submitted: 27/01/2024 Revised: 05/03/2024 Accepted : 13/03/2024

Abstract: Machine Learning and Big Data of today's IT sector. Large volumes of data are reviewed and information extracted using big data storage. Machine learning, on the other hand, refers to a computer's ability to learn and develop without being explicitly taught. Decision trees and neural networks are used in combination with machine learning methods for these reasons. Many sectors have seen amazing development as a result of the dominating mix of Machine Learning and big data. One of these industries is the e-commerce industry. Financial analysts may use predictive analytics to track and exchange critical information about the various economic problems. They automatically retain data on their daily transactions, payments and linked systems, allowing customers to remotely access and manage the financial transactions using the concept of cognitive behaviour of Machine Learning. Along with this we will cover the part of intrusion or any other vulnerabilities. We will use Privacy-preserving techniques using Deep learning models with TensorFlow privacy-preserving method for financial data. we would implement the neural net. Beyond that, we will need to explore various RNNs models to determine appropriate data and context. We will test the trained model and evaluate performance using Accuracy, Precision, recall and F score.

Keywords: *Machine Learning, Big Data, Financial Security, Machine Learning,* TensorFlow*;*

1. Introduction

There are two blue-chips: Machine Learning and Big Data of today's IT sector. Large volumes of data are reviewed and information extracted using big data storage. Machine learning, on the other hand, refers to a computer's ability to learn and develop without being explicitly taught. Automatic data processing and decision-making algorithms are the pillars of machine learning that learn from their past experiences and improve at each stage of their job. "Evolve via learning," to put it another way. To keep up with the ever-growing and ever-changing stream of data in the context of Big Data, Machine Learning is applied in order to provide constantly evolving and relevant insights. Machine learning algorithms explain and detect patterns in the incoming data, they are then translated into other languages with actionable insights that can be incorporated into business processes. Many more decision-making processes were then automated by use of the algorithms. [1]

Decision trees and neural networks are used in combination with machine learning methods for these reasons. Many sectors have seen amazing development as a result of the dominating mix of Machine Learning and big data. One of these industries is the e-commerce industry. Integrating statistical models with data is assisting financial analysts in determining the solutions for various financial crises and to decide the remedies to overcome from it and for future perfectness.

Financial analysts may use predictive analytics to track and exchange critical information about the various economic problems. They automatically retain data on their daily transactions, payments and linked systems, allowing customers to remotely access and manage the financial transactions using the concept of cognitive behavior of Machine Learning [2].

2. Related Work

Rutvij H. Jhaveri (2022) [3] The digital environment of the Industry 5.0 revolution is rife with massive volumes of data. Despite the need for data analysis and interpretation, machine learning is showing promise in a number of fields, including intelligent control, decision-making, speech recognition, natural language processing, computer graphics, and computer vision. Recent years have seen a wide recognition and use of deep learning & machine learning techniques by many real-time engineering applications due to their remarkable performance. Designing automated and intelligent applications that can manage data in domains like health, cyber-security, and intelligent transportation systems requires a solid understanding of machine learning. In the topic of machine learning, there are many different approaches, such as supervised algorithms, semi-supervised algorithms, unsupervised algorithms, & reinforcement learning. study offers a thorough examination of managing machine learning-powered real-time engineering applications, which

[1]*Research Scholar, Kalinga University-492101, New Raipur, Chhattisgarh, India.*
ORCID ID: 0009-0003-6178-4750.
[2]*Department of CS&E, Kalinga University-492101, New Raipur, Chhattisgarh, India.*
ORCID ID: 0000-0002-5845-2558
[3]*Department of MCA, Trinity Academy of Engineering, Pune, 411048, Maharashtra, India*
ORCID ID: 0009-0000-2784-5191
[1] *Corresponding Author Email: prabhanjan1111@gmail.com*
[2] *Corresponding Author Email: guddi.singh@kalingauniversity.ac.in*
[3] *Corresponding Author Email: aabhusari@gmail.com*

will raise the capabilities & intelligence of an application. This research advances our knowledge of how different machine learning techniques can be applied in practical settings, including intelligent transportation systems, cyber security, or healthcare. The goals of this study are to shed insight on the challenges that machine learning techniques face while handling practical applications. Academics & experts in the business will use this study as a point of reference, and from a technical perspective, decision-makers on a variety of application domains & real-world situations will use it as a benchmark.

Shiv Hari Tewari (2021) [4] Among the many technological and operational advancements in cybersecurity in recent years, data science has emerged as a driving force. Automating and enhancing a security system requires the extraction of security event patterns or insights from cybersecurity data and the creation of a data-driven model. Data science is the study and analysis of real-world events utilizing a variety of scientific methodology, machine learning techniques, processes, and systems. Data science, its development, and its applications in cloud security are briefly presented in this study by the Researcher, as well as how cybersecurity data science came to be, the benefits provided by Cybersecurity Data Science (CSDS), and the steps involved, such as gathering data from relevant cybersecurity sources and combining it with analytics to provide more effective security solutions. Thoughts of cybersecurity data science provide more intelligent, actionable computing compared to traditional cybersecurity computing. After that, the researcher went through the numerous potential issues that may arise as a result of the widespread use of CSDS, as well as how machine learning and deep learning may be applied to it and the different sorts of algorithms that can be used. As a result, in addition to examining the history of Data Science and its current applications in cybersecurity, the research also examines how a system that relies on data-driven intelligent decision-making might protect our system from both known and unknown cyber threats.

Kosrat Dlshad Ahmed (2021) [5] Users benefit from enhanced experiences and higher service quality from a variety of angles thanks to IoT technologies & connectivity. In this regard, it is necessary to assure the recent growth of technological prospects & management of sufficient aspects for the delivery of performance. Broadly connected features, systems, data storage facilities, management procedures, applications, devices, users, gateways, services, and thousands of other components are all connected in the context of the Internet of Things idea. IoT applications have become increasingly important in recent years, which has created enormous development & management potential. Users' attention has recently been drawn to cybersecurity and protecting user privacy. An increasing number of people are connecting as social media platforms gain popularity. As opportunities for connectivity rise, people require more safe spaces for connectivity. This article covers a variety of cybersecurity topics, including developing and managing cybersecurity, comprehending security & privacy concepts, and utilizing deep learning models to analyze machine learning concepts. In order to illustrate the comprehension of cybersecurity within Internet of Things networks, several deep learning models, including CNN, MLP, LSTP, and a hybrid model combining CNN & LSTP, have been examined. Prospective study opportunities have also been suggested to aid in the learning process.

Brian Schwartz (2020) [6] In order to recommend the best course of action for patients based on their pre-treatment characteristics, this study intends to develop a treatment selection algorithm that combines statistical inference or machine learning. The study examined a naturalistic, disorder-heterogeneous sample of N = 1,379 outpatients receiving either cognitive behavioral therapy or psychodynamic therapy. The training data (n = 966) was used to model the varying treatment response, which indicates each person's ideal treatment, using a combination of random forest and linear regression. Personalized recommendations were assessed using a different holdout dataset (n = 413). Regarding the training data, there was a significant difference in the outcomes between patients who received their optimal treatment and those who did not (b = –0.043, p =.280). However, this difference was not significant in the holdout data. fortunately, the average percentage of change on the BSI in the holdout data was 52.6% for their optimal or 38.4% for their non-optimal treatment (p =.017; d = 0.33 [0.06, 0.61]) for the 50% of patients with the greatest predicted benefit of receiving their optimal treatment. A treatment selection algorithm that supports therapists' clinical decision-making & based on a blend of machine learning and statistical inference may enhance treatment outcomes for certain outpatients but not all of them.

Ouissem Ben Fredj (2020) [7] The frequency of cybersecurity attacks is rising exponentially, which renders current detection methods inadequate or increases the need to develop more pertinent prediction models and strategies. Since current attack prediction models are unable to keep up with the vast volume or diversity of attacks, this problem remains unresolved. Researchers have recently focused a lot of attention on machine learning approaches, particularly deep learning techniques, due to their exceptional performance in various prediction-based fields. This paper investigates the use of deep learning techniques for cybersecurity attack prediction in this context. Specifically, it suggests a new set of meticulously crafted LSTM, RNN, or MLP based models to forecast the kind of attack that might occur. A recently released dataset called CTF was used to validate the suggested models, and the outcomes

were encouraging, particularly for the LSTM model, which had an f-measure of more than 93%.

Emiliano De Cristofaro (2020) [8] In recent years, service providers like Google, Microsoft, and Amazon have begun to give users access to software interfaces that make it simple for them to incorporate machine learning tasks into their apps. All things considered, businesses can now outsource complicated tasks like clustering, training classifiers, making predictions, and so on by using Machine Learning as a Service (MLaaS) engines. Additionally, they can allow third parties to query models that were trained using their data. Naturally, there are other contexts in which this approach can be applied (and is frequently recommended), such as government partnerships, citizen science initiatives, and business-to-business alliances. But if the data utilized for training these models could be recovered by hostile users, there would be major problems due to information leakage. If the model's internal parameters are thought to be confidential information, then access to the model shouldn't enable a rival to discover them. In this paper, we examine the privacy issues in this field, offering a methodical analysis of the pertinent research literature and considering potential solutions. More precisely, we give a thorough introduction to pertinent machine learning or privacy concepts. After that, we go over potential adversarial models or settings, go over a variety of attacks pertaining to the leakage of sensitive or private information, and examine recent findings that try to thwart these attacks. In closing, provide a list of unresolved issues that still need to be investigated. These issues include the need for improved assessments, more focused defenses, and research on the relationship between policy and data protection initiatives.

Shan Suthaharan (2014) [9] The specific issue of classifying network intrusion traffic using big data is the main focus of this paper. It talks about the difficulties that the Big Data issues related to network intrusion prediction pose for the system. Predicting a potential intrusion attack in a network necessitates the ongoing gathering of traffic data & quick learning of its attributes. The network's constant gathering of traffic data results in Big Data issues, which are brought on by the volume, variety, and velocity characteristics of Big Data. ML techniques that capture global traffic pattern knowledge are necessary for the learning of network characteristics. The implementation of machine learning frameworks will present significant system challenges due to the properties of big data. In this paper, geometric representation-learning techniques or contemporary Big Data networking technologies are used to address the issues and difficulties associated with handling Big Data classification. This paper specifically addresses the challenges associated with integrating machine learning, representation-learning, supervised learning, and big data technologies (such as Hadoop, Hive, or cloud) to address issues relating to network traffic classification.

Ishan Banerjee (2013) [10] GUI testing is system testing of a software that has a graphical-user interface (GUI) front-end. Because system testing entails that the entire software system, including the user interface, be tested as a whole, during GUI testing, test cases—modeled as sequences of user input events—are developed and executed on the software by exercising the GUI's widgets (e.g., text boxes and clickable buttons). More than 230 articles have appeared in the area of GUI testing since 1991.

3. Objectives of the study

1. Collect Amazon Review Dataset, which is publically available.

2. Pre-processing of the training data which include:

   a. Remove duplicate and star reviews

   b. Lemmatization

   c. Word Cloud

   d. Clean Text

   e. Stemming

3. Feature selection and engineering by TF-IDF method.

4. Fitting the data in different deep learning classification models.

5. Hyper tuning the model to obtain the best results in the terms of accuracy, precision, F1-score, etc.

6. Originate various plots to show the distribution of sentiment classes.

4. Methodology

Methodology is the systematic, theoretical analysis of the methods applied to a field of study. It comprises the theoretical analysis of the body of methods and principles associated with a branch of knowledge. This project is problem-driven and will involve a large amount of deep learning, RNN, or hybrid algorithm implementation.

Initially, experiment with data, gather from accessible sites, etc. Once this data is ready for the next preprocessing step, we can determine after numerous analyses which features are useful and which ones should be excluded to create a very useful, sharply driven, and finely tuned dataset. First, we will use Deep Learning RNN models, and once we extract the features, we will apply hybrid models. We anticipate a lot of features.

This data is structured using RNNs, which we understand to be good.

The neural net would then be put into use. Beyond that, in order to find relevant data and context, we will need to investigate different RNNs models.

Steps for work we will do

1. Implementation of the given topic "Prognostic Approach immunes the Data Security in Data science while analyzing Big Data specific to Financial Data using the concept Mature behavior of Machine Learning."
2. We will need to modify the methodology according to the title and redesign it.
3. To apply data security, I will use Privacy-preserving techniques using Deep learning models with TensorFlow privacy-preserving method for financial data.
4. Data collection, preprocessing, and Eda remain the same but neural network implementation will change according to the requirements.
5. We only focus on research work implementation (Results analysis, performance evaluation, and comparative analysis of algorithms with proposed work) and the remaining application part like GUI implementation based on client requirement.

Dataset- In this case, we will use financial fraud detection or credit card fraud detection data open access data.

Neural Network- In this case for defining mature behavior we will create our own custom transfer learning model for training data with the existing trained feature. To this, we will consider text data and implement recurrent neural networks to convey existing features based on mature behavior or transfer learning concepts.

Preprocessing- Pre-processing of the training data which includes:

a. Remove duplicate and star reviews

b. Lemmatization

c. Word Cloud

d. Clean Text

e. Stemming

Performance Evaluation- Test the trained model and evaluate performance using Accuracy, Precision, recall and F score.

5. Implementation

To implement this concept, we will use financial fraud detection or credit card fraud detection data open access data.

Step 1 Dataset collection

Link https://www.kaggle.com/datasets/ealaxi/paysim1/data

Dataset Description

NOTE: Transactions which are detected as fraud are cancelled, so for fraud detection these columns (oldbalanceOrg, newbalanceOrig, oldbalanceDest, newbalanceDest) must not be used.

Headers

Here is an example of one row with an explanation of the headers:

	step	type	amount	nameOrig	oldbalanceOrg	newbalanceOrig	nameDest	oldbalanceDest	newbalanceDest	isFraud	isFlaggedFraud
0	1	PAYMENT	9839.64	C1231006815	170136.0	160296.36	M1979787155	0.0	0.0	0	0
1	1	PAYMENT	1864.28	C1666544295	21249.0	19384.72	M2044282225	0.0	0.0	0	0
2	1	TRANSFER	181.00	C1305486145	181.0	0.00	C553264065	0.0	0.0	1	0
3	1	CASH_OUT	181.00	C840083671	181.0	0.00	C38997010	21182.0	0.0	1	0
4	1	PAYMENT	11668.14	C2048537720	41554.0	29885.86	M1230701703	0.0	0.0	0	0

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6362620 entries, 0 to 6362619
Data columns (total 11 columns):
 #   Column          Dtype
---  ------          -----
 0   step            int64
 1   type            object
 2   amount          float64
 3   nameOrig        object
 4   oldbalanceOrg   float64
 5   newbalanceOrig  float64
 6   nameDest        object
 7   oldbalanceDest  float64
 8   newbalanceDest  float64
 9   isFraud         int64
 10  isFlaggedFraud  int64
dtypes: float64(5), int64(3), object(3)
memory usage: 534.0+ MB
```

1,PAYMENT,1060.31,C429214117,1089.0,28.69,M1591654462,0.0,0.0,0,0

step - displays a real-world time interval. One step in this instance equals one hour. 744 steps in total (30 days of simulation).

type - CASH-IN, CASH-OUT, DEBIT, PAYMENT and TRANSFER.

Amount - The transaction amount expressed in local currency.

OldbalanceOrg is the initial balance prior to the transaction, and nameOrig is the customer who initiated the transaction.

newbalanceOrig - new balance following the transaction.

nameDest: The client who will be receiving the transaction

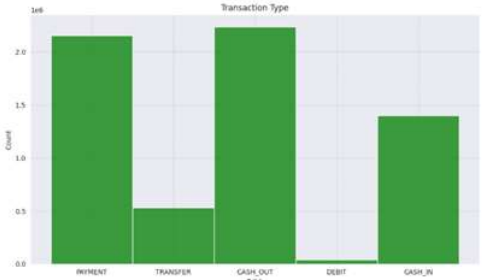
oldbalanceDest: The recipient of the starting balance prior to the transaction. Please take note that there is no information available for clients beginning with M (Merchants).

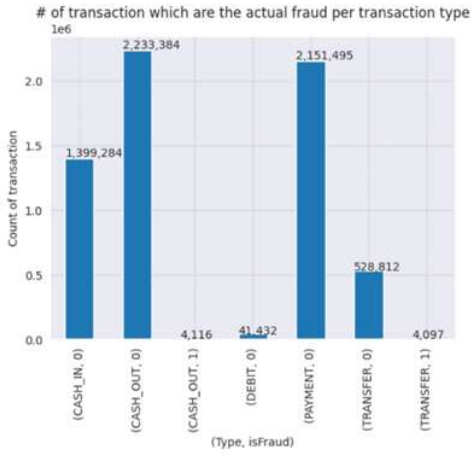
newbalanceDest: The recipient of the new balance following the transaction. Please take note that there is no information available for clients beginning with M (Merchants).

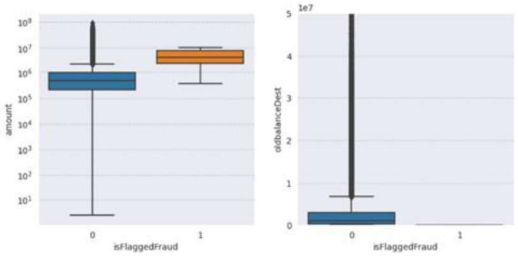
isFraud: These are the transactions that the fictitious agents inside the simulation make. The fraudulent activity of the agents in this particular dataset seeks to profit by seizing control of the customers' accounts, attempting to withdraw all of the money by moving it to another account, and then using the system to cash out.

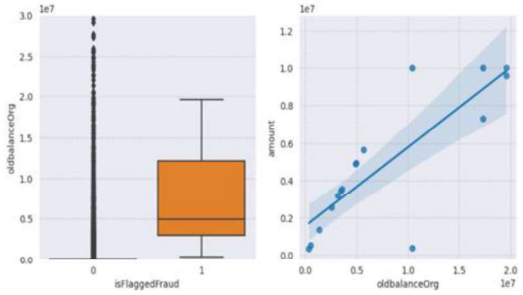
isFlaggedFraud: The goal of the business model is to prevent unauthorized attempts at transfer of large amounts of money between accounts. In this dataset, transferring more than 200,000 in a single transaction is considered illegal.

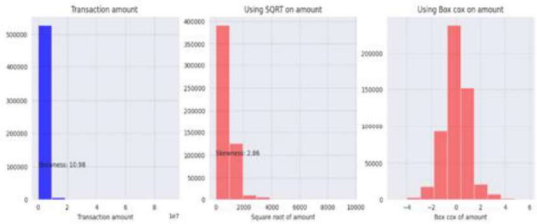
Step 2 Visualization and EDA

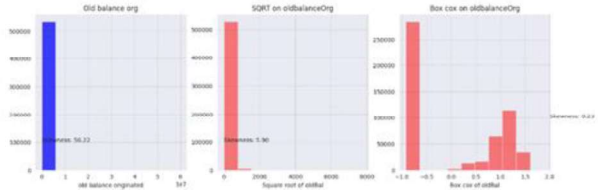


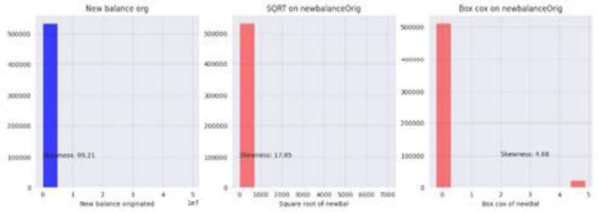


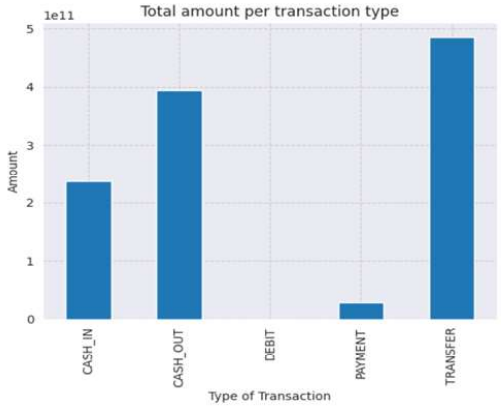












Step 3 Feature Selection

```
X = data_fraud.drop(['isFraud'],axis=1)
y = data_fraud[['isFraud']]
```

data_fraud appears to be a DataFrame or a dataset that contains information related to fraud detection or a similar binary classification problem.

data_fraud.drop(['isFraud'], axis=1) is used to separate the features (input variables) from the target variable (label). Here's what's happening:

data_fraud.drop(['isFraud'], axis=1) is calling the drop method on the data_fraud

DataFrame. It specifies that we want to drop the column labeled 'isFraud'.

axis=1 indicates that we are specifying the column axis (i.e., columns are dropped).

After this operation, X will contain all the columns from the original data_fraud DataFrame except for the 'isFraud' column. In other words, X contains the feature variables used for making predictions.

y = data_fraud[['isFraud']] is used to extract the target variable (the variable you want to predict). Here's what's happening:

data_fraud[['isFraud']] is used to select the 'isFraud' column from the data_fraud DataFrame.

After this operation, y contains the values of the 'isFraud' column, which typically represents whether a transaction is fraudulent (1) or not fraudulent (0).

Step 4 Data splitting, Standard Scaling, and reshaping and Create tensorflow data for prognostic approach for immune the data for privacy preserving

i. Splitting the Dataset:

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

This code uses the train_test_split function from scikit-learn to split the dataset into a training set & testing (validation) set.

X represents the features or input data, and y represents the labels or target variable.

test_size=0.2 specifies that 20% of the data will be used for testing, and the rest (80%) for training.

random_state=42 sets a random seed for reproducibility.

ii. Standardizing the Data:

scaler = StandardScaler()

The StandardScaler from scikit-learn is created to standardize (normalize) the data.

X_train = scaler.fit_transform(X_train)

The fit_transform method is used to compute the mean & standard deviation of the training data and then transform it to have a mean of 0 & standard deviation of 1. This standardization helps the model learn more effectively.

X_test = scaler.transform(X_test)

The same transformation is applied to the test data, using the mean & standard deviation computed from the training data.

iii. Reshaping Data for Model Input:

X_train = np.reshape(X_train, (X_train.shape[0], 1, X_train.shape[1])

This code reshapes the training data to be compatible with a Model layer.

It changes the shape from (batch_size, sequence_length) to (batch_size, sequence_length, input_features), which is the required input shape for LSTM layers.

X_test = np.reshape(X_test, (X_test.shape[0], 1, X_test.shape[1])

The same reshaping operation is applied to the test data to match the model's input requirements.

iv. Creating TensorFlow Datasets:

train_dataset = tf.data.Dataset.from_tensor_slices((X_train, y_train)

This code creates a TensorFlow dataset from the training data. TensorFlow datasets are an efficient way to handle and batch data.

train_dataset = train_dataset.shuffle(buffer_size=1024).batch(batch_size)

The dataset is shuffled to randomize the order of samples in each epoch. Shuffling helps prevent the model from memorizing the order of training samples.

It's then batched into mini-batches of size batch_size. Batching is used to train the model more efficiently.

Step 5 Modeling Neural network

I) LSTM Model Details

1. **Create a Sequential Model:**

model_LSTM = Sequential(): This line initializes a Sequential model in Keras. A Sequential model allows you to create a linear stack of layers.

2. **Add LSTM Layers:**

model_LSTM.add(LSTM(128, input_shape=(1, X.shape[1], return_sequences=True)): This line adds an LSTM layer to the model.

LSTM(128): It creates an LSTM layer with 128 units (or neurons) in this layer. LSTM layers are commonly used for handling sequential data.

input_shape=(1, X.shape[1]): Specifies the input shape for the first LSTM layer. In this case, it expects input data with one time step and the number of features represented by X.shape[1].

return_sequences=True: Indicates that this LSTM layer should return sequences (output for each time step) instead of just the final output. This is useful when you're stacking multiple LSTM layers.

3. **Add More LSTM Layers:**

The next three lines (model_LSTM.add(LSTM(128, return_sequences=True)), model_LSTM.add(LSTM(64, return_sequences=True)), and model_LSTM.add(LSTM(64, return_sequences=True))) add additional LSTM layers to the model.

These layers are similar to the first LSTM layer but without the input_shape parameter since it has already been specified in the first layer.

All of these LSTM layers return sequences.

4. **Add Dense Layers:**

model_LSTM.add(Dense(64, activation='relu'), model_LSTM.add(Dense(32, activation='relu'), and model_LSTM.add(Dense(16, activation='relu')): These lines add Dense layers to the model.

Dense layers are fully connected layers with the specified number of units (64, 32, and 16) and use the ReLU (Rectified Linear Unit) activation function. They are used for nonlinear feature processing.

5. **Add the Output Layer:**

model_LSTM.add(Dense(1, activation='sigmoid')): This line adds the output layer to the model.

The output layer has a single unit, typically used for binary classification tasks, and uses the sigmoid activation function to produce binary output values (0 or 1).

II) Hybrid Model details

1. **Model Type and Architecture:**

model_hybrid = Sequential(): Initializes a sequential model using Keras, which is a linear stack of layers.

The model consists of a combination of Bidirectional LSTM & Bidirectional GRU (Gated Recurrent Unit) layers, along with Dense layers for feature processing and nonlinearity.

2. **Bidirectional LSTM and GRU Layers:**

Bidirectional layers process input data in both forward and backward directions, which allows the model to capture information from past and future time steps. The model includes multiple stacked Bidirectional LSTM and Bidirectional GRU layers.

model_hybrid.add(Bidirectional(LSTM(128, input_shape=(1, X.shape[1]), return_sequences=True))): The first Bidirectional LSTM layer with 128 units and input shape specified for the first layer. It returns sequences as output.

model_hybrid.add(Bidirectional(GRU(128, return_sequences=True))): The second Bidirectional GRU layer with 128 units, returning sequences.

model_hybrid.add(Bidirectional(GRU(64, return_sequences=True))): The third Bidirectional GRU layer with 64 units, returning sequences.

model_hybrid.add(Bidirectional(GRU(64))): The fourth Bidirectional GRU layer with 64 units. It does not return sequences.

3. **Dense Layers:**

After the Bidirectional LSTM and GRU layers, the model includes Dense layers for feature processing and nonlinearity.

model_hybrid.add(Dense(64, activation='relu')): The first Dense layer with 64 units and ReLU (Rectified Linear Unit) activation function.

model_hybrid.add(Dense(32, activation='relu')): The second Dense layer with 32 units and ReLU activation.

model_hybrid.add(Dense(16, activation='relu')): The third Dense layer with 16 units and ReLU activation.

4. **Output Layer:**

model_hybrid.add(Dense(1, activation='sigmoid')): This is the final Dense layer with a single unit, using the sigmoid activation function. It produces binary output values (0 or 1) and is suitable for binary classification tasks.

5. **Compiling the Model:**

model_hybrid.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy']): This line compiles the model, specifying the loss function ('binary_crossentropy' for binary classification), the optimizer ('adam'), and the metric to monitor during training ('accuracy').

Step 6 Privacy Preserving for Immune the data security using DPSDG Optimizer

It is designed to provide differential privacy guarantees to the training process by adding noise to the gradients during optimization. Let's break down how this part of the code is used for privacy-preserving and detail each parameter:

l2_norm_clip:

l2_norm_clip is a hyperparameter that sets an upper limit on the L2 (Euclidean) norm of the gradients of the model's loss function. It represents the maximum allowable magnitude of the noise that can be added to the gradients.

noise_multiplier:

noise_multiplier is another hyperparameter that controls the amount of noise added to the gradients. A higher noise_multiplier introduces more noise, which increases privacy but can negatively affect model utility (accuracy).

num_microbatches:

num_microbatches refers to the number of mini-batches into which each batch of data is divided. Differentially private stochastic gradient descent (DP-SGD) computes gradients for each mini-batch separately, adds noise to the gradients, and then aggregates the noisy gradients to update model parameters. Smaller values of num_microbatches lead to stronger privacy guarantees.

target_delta:

target_delta is the desired privacy guarantee. It is a measure of the risk of privacy breaches. A smaller target_delta corresponds to a stronger guarantee, but it may require more noise to achieve.

learning_rate:

learning_rate is the step size used in the optimization process. It controls how quickly the model updates its parameters in the direction of minimizing the loss. A typical hyperparameter for training neural networks, though it can affect both model utility and privacy.

optimizerDP_SGD:

optimizerDP_SGD is an instance of the DPKerasSGDOptimizer from TensorFlow Privacy.

It is specifically designed for use with differential privacy. This optimizer computes noisy gradients using the parameters you set (l2_norm_clip, noise_multiplier, num_microbatches) and then updates the model's weights using these noisy gradients.

The DPKerasSGDOptimizer helps you train a deep learning model while providing differential privacy guarantees to protect sensitive data. It combines the benefits of privacy and utility by adding carefully calibrated noise to the gradients during training.

Step 7 Continuous Training to achieving the mature behavior of models

The provided code snippet represents a training loop that continuously retrains a machine learning model for credit card fraud detection across multiple iterations. The goal of this approach is to achieve mature behavior for the model. Let's break down the key components and explain how this process contributes to the model's maturity:

i. Loop for Continuous Retraining:

The loop iterates for a specified number of total iterations. In each iteration, the model is retrained using new data.

ii. Loading and Combining New Data:

In each iteration, new data is loaded from a CSV file (credit_card_transactions_iterationX.csv, where X is the iteration number).

The new data is combined with the initial data. This combination ensures that the model adapts to new patterns and potentially changing characteristics of the data. It allows the model to continuously learn from fresh data.

iii. Data Preprocessing:

After combining the data, features and labels are separated.

The combined dataset is split into training and testing sets for model training and evaluation.

Data standardization is applied using a StandardScaler. Standardization ensures that the data has a mean of 0 and a standard deviation of 1, which is a common preprocessing step for many machine learning algorithms.

iv. Reshaping Data for Model Input:

The data is reshaped to fit the input requirements of the model. In this case, the data is reshaped for a Bidirectional GRU model, which expects input data in the form of (batch_size, sequence_length, feature_dim). The reshaping is necessary to match the model's architecture.

v. Privacy Guarantee Calculation:

The code calculates the privacy guarantee for the current iteration. It uses the compute_dp_sgd_privacy function to estimate the privacy loss. The epsilon value represents the privacy budget expended. Privacy is a critical concern in applications involving sensitive data, such as fraud detection. This step ensures that the model's training process adheres to privacy regulations and protects individuals' data.

vi. Differential Privacy (DP) Optimizer Configuration:

An instance of the DPKerasSGDOptimizer is configured with parameters that control the privacy preservation aspects, such as l2_norm_clip, noise_multiplier, and num_microbatches. This optimizer introduces privacy-aware noise during gradient computation.

vii. Model Compilation and Training:

The machine learning model (presumably a Bidirectional GRU model) is compiled using the DP optimizer. The loss function, optimizer, and evaluation metrics are specified.

The model is trained for a fixed number of epochs (in this case, 10) to update its parameters based on the new data and privacy constraints.

viii. Model Evaluation:

After training, the model is evaluated on a separate testing dataset.

Various evaluation metrics, as accuracy, precision, recall, and F1 score, are calculated. These metrics assess the model's performance in identifying credit card fraud. It's essential to continuously monitor and evaluate the model's effectiveness.

ix. Maturity and Continuous Learning:

By repeating this process over multiple iterations, the model continuously adapts to evolving data patterns, privacy requirements, and the changing characteristics of the credit card transaction data.

This continuous learning approach aims to achieve mature behavior for the model by keeping it up-to-date and privacy-compliant.

A continuous learning and privacy-preserving approach for a credit card fraud detection model. It incorporates new data, adapts to changing conditions, and ensures privacy compliance while continuously monitoring the model's performance. This iterative process is designed to maintain the model's effectiveness and reliability over time.

6. Results

Privacy-preserving techniques using Deep learning models with TensorFlow privacy-preserving method for financial data, will follow the TensorFlow DP-SGD optimizer process. we were considering text data and implement recurrent neural networks to convey existing features based on mature behavior or transfer learning concepts. Test the trained model and evaluate performance using Accuracy, Precision, recall and F score.

Table 1: Results Evaluation

Model	Accuracy	Precision	Recall	F Score
LSTM	99.95	99.81	67.90	80.61
GRU	99.85	40.75	31.11	35.32
Hybrid Approach	99.95	98.98	66.48	79.54

7. Conclusion

We were enhanced data security within the field of data science through the application of a prognostic approach that leverages the concept of cognitive behavior in machine learning. We were studied to using a prognostic approach, which involves predicting future security threats based on historical data and patterns, can enhance data security in the field of data science. This approach leverages the cognitive behavior of machine learning, implying that ML algorithms can learn from past security incidents to proactively identify and mitigate potential risks, ultimately strengthening data security measures. We will begin with Deep Learning RNN models and then implement hybrid models on the features we ultimately extract. Privacy-preserving techniques using deep learning models

References

[1] Pecht, Michael G. (2008). Prognostics and Health Management of Electronics Wiley. ISBN 978-0-470-27802-4.

[2] Chaudhari Prabhanjan, and Dr. Amit Bhusari. "Transfer Optimistic Outcome-based Learning for Mature Behavior of Machine in Deep Learning." 2020 IEEE International Conference on Advent Trends in Multidisciplinary Research and Innovation (ICATMRI). IEEE, 2020.

[3] Jhaveri, R. H., Revathi, A., Ramana, K., Raut, R., & Dhanaraj, R. K. (2022). A review on machine learning strategies for real-world engineering applications. Mobile Information Systems, 2022.

[4] Shiv Hari Tewari (2021) on "Necessity of data science for enhanced Cybersecurity", International Journal of Data Science and Big Data Analytics, Volume 1, Issue 1, ISSN: 2710-2599

[5] Ahmed, K. D., & Askar, S. (2021). Deep learning models for cyber security in IoT networks: A review. International Journal of Science and Business, 5(3), 61-70.

[6] Schwartz, B., Cohen, Z. D., Rubel, J. A., Zimmermann, D., Wittmann, W. W., & Lutz, W. (2021). Personalized treatment selection in routine care: Integrating machine learning and statistical algorithms to recommend cognitive behavioral or psychodynamic therapy. Psychotherapy Research, 31(1), 33-51.

[7] Ben Fredj, O., Mihoub, A., Krichen, M., Cheikhrouhou, O., & Derhab, A. (2020, November). CyberSecurity attack prediction: a deep learning approach. In 13th International Conference on Security of Information and Networks (pp. 1-6).

[8] De Cristofaro, E. (2020). An overview of privacy in machine learning. arXiv preprint arXiv:2005.08679.

[9] Suthaharan, S. (2014). Big data classification: Problems and challenges in network intrusion prediction with machine learning. ACM SIGMETRICS Performance Evaluation Review, 41(4), 70-73.

[10] Banerjee, I., Nguyen, B., Garousi, V., & Memon, A. (2013). Graphical user interface (GUI) testing: Systematic mapping and repository. Information and Software Technology, 55(10), 1679-1694.