Team #	**Team - 16**
Title of Paper	**An Intuitive Projective Model with Probabilities of PSO+GA on Visegrad features with Multi domain Applicative Fraud Detections**
Member Names	**Sai Shilpa Padmanabula** Padmanabulashilpa@gmail.com 9033558148 **SnehaSri Manda** Snehasrimanda@gmail.com 4302953518 **Rahul Gaddipati** Rahulgaddipati99@gmail.com 4804924567
Date of Submission	**05/02/2024**
Name of Checker	**Sai Shilpa Padmanabula**
Youtube Link	https://youtu.be/MrVW2XWWpcI

Checklist:

Items to check	Yes/No
Is your paper in IEEE/APA styles, single line-spaced with 11- or 12-points Times Roman fonts, in single column?	☒
Did you follow paper/section templates/sample papers?	☒
Is your paper in Word type?	☒
All figures/tables are numbered with captions (titles), and mentioned/elaborated in the text?	☒
If the figurers/tables are copied, are they cited with the source information? Best is to adopt in your own way and be put.	☒
Are all references cited in the article; 50% or more should be less than 5 years old; Are all references are in either IEEE style or APA style?	☒
Was methodology explained with diagrams and figures, and proposed as your own method and described as in model paper?	☒
Was appropriate technical name given for the proposed methodology?	☒
Were all parts/sections are written in order and included as intended?	☒
Did you paraphrase not coping someone else's work without citation?	☒
Were spell and grammar check done?	☒
Others? (Specify – All the changes that are suggested on the Literature review and methodology parts are included)	☒

Team #: 16

Title: An Intuitive Projective Model with Probabilities of PSO+GA on Visegrad features with Multi domain Applicative Fraud Detections.

Names: Sai Shilpa Padmanabula

Sneha Sri Manda

Rahul Gaddipati

Abstract

In the realm of financial transactions, fraud detection is of paramount importance for maintaining trust and security. Traditional methods have proven inadequate in handling the ever-evolving tactics of fraudulent activities. This study presents an innovative projective model integrating Probabilities of PSO+GA on Visegrad features for Multi-domain Applicative Fraud Detection. Leveraging the strengths of Particle Swarm Optimization (PSO) and Genetic Algorithms (GA), the model offers a comprehensive approach to fraud detection across diverse domains. By applying this intuitive framework to Visegrad features, the model demonstrates enhanced accuracy and efficiency in identifying fraudulent activities across multiple domains. This research contributes to advancing the efficacy of fraud detection systems through the fusion of probabilistic optimization techniques and multi-domain applicative approaches.

The proposed system leverages AI algorithms such as machine learning and deep learning to analyze vast amounts of transactional data efficiently and accurately. The research is underpinned by the ambition to achieve a prediction and classification accuracy exceeding 90%. The AI model encompasses a fusion of traditional machine learning algorithms and metaheuristic techniques such as Genetic Algorithms, Particle Swarm Optimization, and Simulated Annealing. The project not only aims to detect fraudulent activities effectively but also addresses the imbalanced nature of financial datasets, ensuring fairness and accuracy in the classification of potential fraud instances. With a focus on transparency and interpretability, the model incorporates techniques like LIME and SHAP to elucidate the decision-making process, thereby fostering user trust and understanding. This research is poised to contribute significant insights into the realm of fraud detection, offering a robust and advanced solution for financial security within the Visegrad Group.

1. Introduction

In today's rapidly evolving technological landscape, the prevalence of fraudulent activities across various domains poses significant challenges for businesses and organizations worldwide. Detecting and mitigating fraud requires sophisticated methodologies that can adapt to the dynamic nature of fraudulent behaviors across diverse domains. In response to this pressing need, researchers and practitioners have continuously sought innovative approaches to enhance fraud detection systems. The financial landscape is continually evolving, marked by technological advancements, digitalization, and an increasing volume of data. With these changes comes the persistent threat of financial fraud, necessitating innovative and adaptive approaches for detection

and prevention. The use of machine learning (ML) and artificial intelligence (AI) in the realm of fraud detection has gained considerable attention due to its potential to enhance accuracy and efficiency.

This introduction provides a comprehensive overview of the current state of fraud detection in finance, emphasizing recent research efforts and advancements in leveraging metaheuristic techniques.In recent years, numerous studies have explored novel algorithms and frameworks to fortify fraud detection mechanisms. Ashfaq [1] proposed a mechanism integrating machine learning and blockchain for efficient fraud detection, offering a holistic solution to combat fraudulent activities.The imperative role of digitalization in firms' finance was highlighted by [2], emphasizing the technological perspective and underscoring the necessity of embracing digital transformations to bolster financial security. [3] delved into credit card-not-present fraud detection and prevention using big data analytics algorithms, addressing the challenges posed by evolving payment methods. As financial markets continue to integrate AI and machine learning, [4] conducted a comprehensive analysis, unveiling the influence of these technologies on trading, risk management, and financial operations. Furthermore, [5] delved into the challenges and opportunities presented by data science in finance, shedding light on the evolving landscape and the need for sophisticated analytical approaches.

In the context of these advancements, this research proposes an innovative fraud detection framework for listed companies in the Visegrad Group. Leveraging metaheuristic techniques, the proposed model aims to surpass the 90% prediction and classification accuracy threshold, addressing the unique challenges posed by the dataset covering six sectors over the period of Q1 2017–Q1 2021. Building upon the foundations laid by existing studies, the proposed framework seeks to contribute to the evolving landscape of AI-enhanced fraud detection in finance. Furthermore, the model employs a hybrid optimization technique combining Particle Swarm Optimization (PSO) and Genetic Algorithms (GA). This hybrid approach harnesses the strengths of both optimization methods, enabling efficient feature selection and parameter optimization to enhance the model's performance in detecting fraudulent activities. Moreover, the model's applicative approach to fraud detection emphasizes its practical utility across multiple domains. By addressing the specific needs and challenges of different sectors, the model offers a versatile solution capable of adapting to diverse fraud scenarios.

The proposed model represents a significant improvement in the field of fraud detection, promising enhanced accuracy, adaptability, and practical utility in combating fraudulent activities across various sectors.

1.1 Problem Statement

Despite advancements in fraud detection technologies, traditional methods have struggled to keep pace with the increasingly sophisticated tactics employed by fraudulent actors in the realm of financial transactions. The existing approaches often fall short in effectively addressing the multi-dimensional nature of fraudulent activities across diverse domains within the Visegrad Group. In the context of listed companies within the Visegrad Group (Czech Republic, Hungary, Poland, Slovakia) [6], the existing fraud detection mechanisms lack the necessary precision and efficiency. Furthermore, the inherent imbalance in financial datasets poses a significant challenge, leading to suboptimal detection accuracy and fairness in classification outcomes.

Consequently, there is a pressing need for innovative and comprehensive fraud detection systems

capable of accurately identifying fraudulent transactions while addressing data imbalance issues and ensuring transparency and interpretability in the decision-making process. The challenge is intensified by the intricate nature of financial transactions and the imbalanced distribution of fraudulent instances in the datasets. This study aims to bridge this gap by proposing an intuitive projective model integrating Probabilities of PSO+GA on Visegrad features for Multi-domain Applicative Fraud Detection, thereby advancing the efficacy of fraud detection mechanisms in the financial domain.

1.2 Objectives:

Improve Detection Accuracy
Enhance fraud detection accuracy in financial transactions of Visegrad Group listed companies by integrating AI models with traditional machine learning and metaheuristic techniques like Genetic Algorithms, Particle Swarm Optimization, and Simulated Annealing, aiming for over 90% prediction and classification accuracy.

Address Imbalanced Datasets
Develop a model to handle imbalanced financial datasets by implementing strategies within the AI model to mitigate biases and effectively identify and classify potential fraud instances, ensuring fairness and accuracy in classification.

Enhance Transparency and Interpretability
Incorporate techniques such as LIME and SHAP to enhance transparency and interpretability in the decision-making process of the AI model, fostering user trust and enabling stakeholders to understand the rationale behind fraud classifications.

2. Literature Survey

Fraud detection became a critical area for research and application across various domains, including finance, healthcare, e-commerce, and telecommunications. In recent years, researchers have explored diverse methodologies and techniques to enhance the effectiveness of fraud detection systems. This literature review provides an overview of the important studies and approaches in the field, highlighting the evolution of fraud detection methodologies and the challenges that remain.

In this work, [7] adopt the Nonlinear Activated Beetle Antennae Search Algorithm for Fraud Detection of Public Trading Companies. The authors contribute a novel computational finance approach, showcasing improved fraud detection in public trading companies. [8] present Financial Fraud Detection and Prediction in Listed Companies using SMOTE and Machine Learning Algorithms. Their work involves the application of SMOTE and machine learning algorithms, leading to enhanced fraud detection and prediction in listed companies. [1] propose A Machine Learning and Blockchain-Based Efficient Fraud Detection Mechanism. The authors integrate machine learning and blockchain to establish an efficient fraud detection mechanism, showcasing advancements in fraud detection techniques.

[2] explore the Imperative Role of Integrating Digitalization in the Firms Finance: A Technological Perspective. Their work emphasizes the importance of integrating digitalization in firms' finance, discussing both opportunities and challenges in this technological shift. [9] contribute to the field

with the Robust Financial Fraud Alerting System based on the Cloud Environment. They develop a reliable fraud alerting system in a cloud environment, showcasing advancements in financial fraud detection systems. [3] focus on Credit Card-Not-Present Fraud Detection and Prevention Using Big Data Analytics Algorithms. The authors utilize big data analytics algorithms to effectively prevent credit card-not-present fraud, highlighting advancements in fraud prevention techniques.

[4] undertake a Comprehensive Analysis of AI Applications in Trading, Risk Management, and Financial Operations. The authors unveil the influence of artificial intelligence and machine learning on financial markets, providing an in-depth understanding of AI applications in various financial domains. [10] present Research on Financial Fraud Detection Models Integrating Multiple Relational Graphs. Their work contributes to enhanced fraud detection models by integrating multiple relational graphs, suggesting further exploration in this direction for improved fraud detection. In the exploration of Data Science in Finance: Challenges and Opportunities, [5] identify challenges and opportunities in applying data science in finance, providing insights for overcoming challenges and maximizing opportunities in this domain.

In their work, [11] utilize Heterogeneous Graph Representation Learning for Fraud Detection. The authors enhance fraud detection accuracy based on heterogeneous graph representation learning, showcasing advancements in fraud detection methodologies. [12] explore Big Data-Driven Banking Operations: Opportunities, Challenges, and Data Security Perspectives. Their work identifies opportunities and challenges in big data-driven banking operations, calling for further investigation into data security aspects in big data banking. [13] present an innovative Approach for Fraud Detection in Blockchain Based Healthcare Networks Using Machine Learning. Their work introduces a novel approach for fraud detection in healthcare networks using machine learning, highlighting potential gaps in current methods for fraud detection in healthcare.

[14] measure the Effect of Fraud on Data-Quality Dimensions. Their work examines the impact of fraud on data-quality dimensions, identifying opportunities for improving data-quality dimensions in the presence of fraud. Al- [15] propose Towards a Secure Technology-Driven Architecture for Smart Health Insurance Systems: An Empirical Study. The authors develop a secure technology-driven architecture for smart health insurance systems, emphasizing the evaluation of the proposed architecture's effectiveness and practicality. [16] focus on Fraud Detection in Healthcare Insurance Claims Using Machine Learning. They utilize machine learning for improved fraud detection in healthcare insurance claims, identifying challenges and limitations in current fraud detection methods in healthcare.

[17] investigate Spice and Herb Frauds: Types, Incidence, and Detection: The State of the Art. The authors provide a comprehensive understanding of spice and herb frauds and detection methods, highlighting potential gaps in current spice and herb fraud detection techniques. [18] introduced An Improved Strategy for High Recall Using KNN, LDA, and Linear Regression. Their work enhances recall strategy in credit card fraud detection using KNN, LDA, and linear regression, prompting an evaluation of the proposed strategy and its applicability in diverse scenarios. [19] present FinChain-BERT: A High-Accuracy Automatic Fraud Detection Model Based on NLP Methods for Financial Scenarios. Their work develops a high-accuracy automatic fraud detection model for financial scenarios using NLP methods, highlighting potential limitations or challenges in deploying the FinChain-BERT model.

[20] explore Exploration of Metrics and Datasets to assess the Fidelity of Images generated by the Generative Adversarial Networks. Their work provides insights into assessing the fidelity of images generated by GANs using different metrics and datasets, identifying gaps in current metrics and datasets for assessing image fidelity in GANs. [21] contribute to the development of Seven New dPCR Animal Species Assays and a Reference Material for quantitative ratio measurements of food and feed products. Their work is aimed at assessing the performance and accuracy of the developed assays and reference material.

[22] implement Anomaly Detection Using an Ensemble of Multi-Point LSTMs, showcasing successful anomaly detection using an ensemble of LSTMs. They also discuss potential scenarios where the ensemble of LSTMs may face challenges or limitations. [23] implement Active Learning in the Detection of Anomalies in Cryptocurrency Transactions. Their work showcases improved anomaly detection in cryptocurrency transactions using active learning and emphasizes the assessment of the effectiveness and efficiency of active learning in different cryptocurrency scenarios.

[24] conducts An Analysis of an Open Source Binomial Random Variate Generation Algorithm. The work involves an analysis of an open-source binomial random variate generation algorithm, providing insights into its characteristics and performance. [25] propose A Novel Unsupervised Outlier Detection Algorithm Based on Mutual Information and Reduced Spectral Clustering. The authors introduce an effective unsupervised outlier detection algorithm and discuss the evaluation of the algorithm's performance and applicability in diverse datasets. [26] develop Decentralized Federated Learning-Enabled Relation Aggregation for Anomaly Detection. Their work focuses on enhancing anomaly detection using decentralized federated learning, prompting an investigation of potential challenges and limitations in decentralized federated learning for anomaly detection.

[27] present NLP Sentiment Analysis and Accounting Transparency: A New Era of Financial Record Keeping. Their work involves the application of NLP sentiment analysis for enhancing accounting transparency, prompting exploration of potential limitations or challenges in implementing NLP sentiment analysis for accounting transperency.

[28] focus on Enhan cing Credit Card Fraud Detection: An Ensemble Machine Learning Approach. They contribute to improved credit card fraud detection through an ensemble machine learning approach, prompting an assessment of the performance and robustness of the proposed ensemble approach in diverse scenarios. [29] develop Secure Internet Financial Transactions: A Framework Integrating Multi-Factor Authentication and Machine Learning. Their work contributes to the development of a secure framework for internet financial transactions, emphasizing the evaluation of the effectiveness and practicality of the proposed framework. [30] conduct an Experimental Evaluation of Possible Feature Combinations for the Detection of Fraudulent Online Shops. Their work provides insights into effective feature combinations for detecting fraudulent online shops, prompting exploration of potential limitations or challenges in current feature combinations for fraud detection.

[31] Proposed a hybrid method with a Dynamic Weighted Entropy measurement for improving the efficiency of the whole model learning, and the divide & conquer idea is used.Comparing the concept of auto loan fraud with credit card fraud, the author thought that they have some similarities. So this paper tried a new technique that has not been employed in financial fraud detection yet. It is named Dominance-based Rough Set Balanced Rule Ensemble(DRSA-BRE), and it does have some advantages compared with traditional methods. The author introduces two

new statistic and machine learning techniques - Graph Attention Network and Graph Convolutional Network. This paper summarizes some new exploration in the financial fraud detection area. A typical example is credit card fraud.

Despite the advancements in fraud detection methodologies, several challenges persist, including the detection of previously unseen fraud patterns, handling imbalanced datasets, and ensuring model interpretability and transparency. Future research directions may involve exploring deep learning architectures, incorporating explainable AI techniques, and leveraging emerging technologies such as blockchain and federated learning to further enhance the effectiveness and reliability of fraud detection systems.

Comprehensive Analysis of AI Applications in Trading, Risk Management, and Financial Operations [4]	Findings	The findings offer insights into the AI and ML adoption in finance, highlighting trends, challenges, and regulatory concerns for professionals and policymakers to consider.
	Contributions	Presenting quantitative survey results demonstrating the increasing use of AI and ML technologies in financial institutions.
	Research Gap	The research highlights a gap in addressing the need for comprehensive regulatory oversight, workforce transformation, and ethical standards in the adoption of AI and ML technologies within financial markets.
Research on Financial Fraud Detection Models Integrating Multiple Relational Graphs [10]	Findings	The study introduces Tri-RGCN-XGBoost, a model enhancing fraudulent user detection in digital finance by analyzing user association patterns while identifying key behavioral associations for fraud pattern mining.
	Contributions	Demonstrating significant improvements in fraudulent user detection performance, particularly in reducing underreporting rates, through the proposed model.
	Research Gap	Further exploration of integrating diverse relational graphs for improved fraud detection
Ensemble Model of the Financial Distress Prediction in Visegrad Group Countries [6]	Findings	Proposed ensemble model significantly outperforms traditional machine learning methods in credit card fraud detection, addressing issues such as data imbalance and achieving superior accuracy, precision, recall, and F1-score metrics.
	Contributions	Proposes an effective credit card fraud detection model addressing data imbalance, demonstrates computational efficiency of ensemble models, and compares performance of various machine learning models for fraud detection.

	Research Gap	The research gap identified revolves around the model is relatively fast, specifically in comparison to ANN and SVM. However, the iteration process was demanding more on computational time due to computer equipment.
Deep Learning Techniques in Financial Fraud Detection [31]	Findings	Proposed a hybrid method with a Dynamic Weighted Entropy measurement for improving the efficiency of the whole model learning.
	Contributions	Author tried a new technique that has not been employed in financial fraud detection yet. It is Dominance-based Rough Set Balanced Rule Ensemble(DRSA-BRE), and it does have some advantages compared with traditional methods
	Research Gap	The research gap identified concerns the need for effective solutions to address challenges such as data skewness, concept drift, and short-time system response in financial fraud detection.
An Intuitive Projective Model with Probabilities of PSO+GA on Visegrad features with Multi domain Applicative Fraud Detections. **(Proposed)**	Findings	Development of PSO+ Heuristic Approach on Gen-AI models with ML and DL custom functionality generations on multi-objective approach
	Contributions	Enhanced performance metrics and other Frauds types detection with new conditions and changes for the prediction analysis for anomalies detected and predicted analytically
	Research Gap	More accurate design and its functional Deployments and testing analysis

Table 1 :- Comparison of the State – of – the – art works and the proposed model

3. Methodology

The methodology proposed in this study aims to develop an advanced fraud detection system capable of effectively identifying fraudulent activities across multiple domains. This section provides a brief overview of the key steps involved in the methodology, which integrates intuitive projective modeling with probabilities, optimization techniques like PSO+GA, and multi-domain applicative features for enhanced fraud detection. In the pursuit of enhancing fraud detection

accuracy in financial transactions within the Visegrad Group, a comprehensive methodology is outlined. This methodology encompasses data acquisition and preprocessing, model development with integrated AI techniques, handling imbalanced datasets, enhancing transparency and interpretability, and evaluation and validation.

The proposed approach leverages a Hybrid Probabilistic method integrating Particle Swarm Optimization (PSO) and Genetic Algorithms (GA) to enhance fraud detection accuracy in financial transactions within the Visegrad Group. The Visegrad Group dataset, encompassing multi-domain fraud instances, presents a complex and diverse set of fraudulent activities, making traditional detection methods insufficient. The Hybrid Probabilistic approach aims to address this challenge by combining the strengths of PSO and GA to optimize the fraud detection model across multiple objectives. These methods help us figure out the best settings for catching fraud. At its core, the Hybrid Probabilistic approach begins with the initialization of PSO and GA parameters, tailored to the characteristics of the Visegrad Group dataset. PSO helps us search through lots of possibilities quickly, like exploring a big maze efficiently. Then, GA helps us improve our search by trying out different ideas over and over, like evolving solutions over time. This initialization phase sets the stage for the subsequent optimization process, where the algorithm iteratively refines the model parameters to maximize fraud detection accuracy across various domains.

Following initialization, the Hybrid Probabilistic method undergoes an iterative optimization process driven by PSO and GA. PSO optimizes the model parameters based on particle movements guided by the best-performing solutions, allowing for rapid convergence towards promising regions of the solution space. PSO quickly finds good solutions by following the best ones found so far. It's like quickly finding your way through a maze by following someone who knows the right path. Meanwhile, GA introduces diversity through genetic operators such as crossover and mutation, facilitating exploration of alternative solutions and preventing premature convergence. GA adds variety by trying out different ideas and preventing us from getting stuck too early. It's like trying different routes in the maze to make sure we don't miss anything.This iterative optimization continues until convergence criteria are met, resulting in a robust fraud detection model tailored to the specific characteristics of the Visegrad Group dataset. The final model is tested to make sure it's good at spotting fraud in different situations, giving us confidence that it can help reduce financial risks in the Visegrad Group.

The proposed approach also incorporates Explainable AI (XAI) features, particularly the SHapley Additive exPlanations (SHAP) method, to enhance the interpretability of the fraud detection model. SHAP provides insights into the contribution of each feature to the model's predictions, enabling stakeholders to understand the rationale behind the model's decisions. For example, if our model flags a transaction as fraudulent, SHAP can tell us which factors, like transaction amount or location, contributed the most to that decision. This helps everyone involved understand why the model made that call.

By visualizing the impact of individual features on fraud detection outcomes, SHAP makes the model's decisions more transparent and trustworthy. In the context of the Hybrid Probabilistic approach, SHAP analysis is integrated into the model evaluation phase to verify its effectiveness in detecting fraudulent activities within the Visegrad Group dataset. By examining the SHAP values associated with different features, analysts can identify which variables have the greatest influence on the model's predictions and assess whether the model aligns with domain knowledge and expectations. Additionally, SHAP analysis enables the identification of potential biases or

inconsistencies in the model's decision-making process, allowing for iterative refinement and improvement of the fraud detection model.

Overall, the integration of XAI features such as SHAP enhances the transparency, interpretability, and trustworthiness of the fraud detection model developed using the Hybrid Probabilistic approach. By providing stakeholders with actionable insights into the model's inner workings, SHAP enables informed decision-making and validation of the model's effectiveness in mitigating financial risks within the Visegrad Group.

Data Acquisition and Preprocessing

It is the first step of the financial fraud detection process. In this phase, financial transaction data is sourced from reputable open data repositories such as Kaggle, Data.world, and other relevant websites. This data may include information about financial transactions conducted by various entities. These datasets typically comprise transaction details of listed companies within the Visegrad Group, including transaction amounts, timestamps, and types.

The data, often available in CSV format, undergoes preprocessing to ensure consistency and suitability for analysis. We fix any missing information and make sure all the data is consistent and suitable for our analysis. We also adjust the numerical features so they're all on the same scale, which helps our analysis be more accurate.

Some of the information might be in words instead of numbers, like transaction types. We change these into numbers so our computer programs can understand and analyze them better. Sometimes, there are way more legitimate transactions than fraudulent ones, which can make it hard for our analysis to spot the fraud.

So, we use techniques like adding more fraudulent transactions or removing some legitimate ones to make the data more balanced. Overall, this step is about getting the data ready for our analysis by cleaning it up, making sure everything is in a format our programs can understand, and balancing out the different types of transactions.

Exploratory Data Analysis (EDA)

After preprocessing, we take a closer look at it to understand it better. This step is called exploratory data analysis (EDA). We use statistics and graphs like histograms, box plots, and scatter plots to see what the data looks like and if there are any unusual patterns. Through statistical summaries and visualizations, such as histograms, box plots, and scatter plots, the underlying patterns and anomalies in the data are explored. For example, if we're looking at financial data, we might use these graphs to see if there are any unusual spikes in transactions or if certain types of transactions happen more often at certain times.

Feature engineering techniques are then employed to extract relevant features from the dataset. These features may include transaction amounts, timestamps, and other metadata that could provide valuable information for fraud detection. By doing this, we can identify trends, outliers, missing values, and potential problems in the data that could be signs of fraud.

This helps us to figure out what steps to are needs to be taken next in our analysis, like cleaning up the data or choosing the best methods for finding fraud. Overall, this process is really important because it helps us understand the data better and make smart decisions as we try to catch fraud.

3.1 Block Diagram

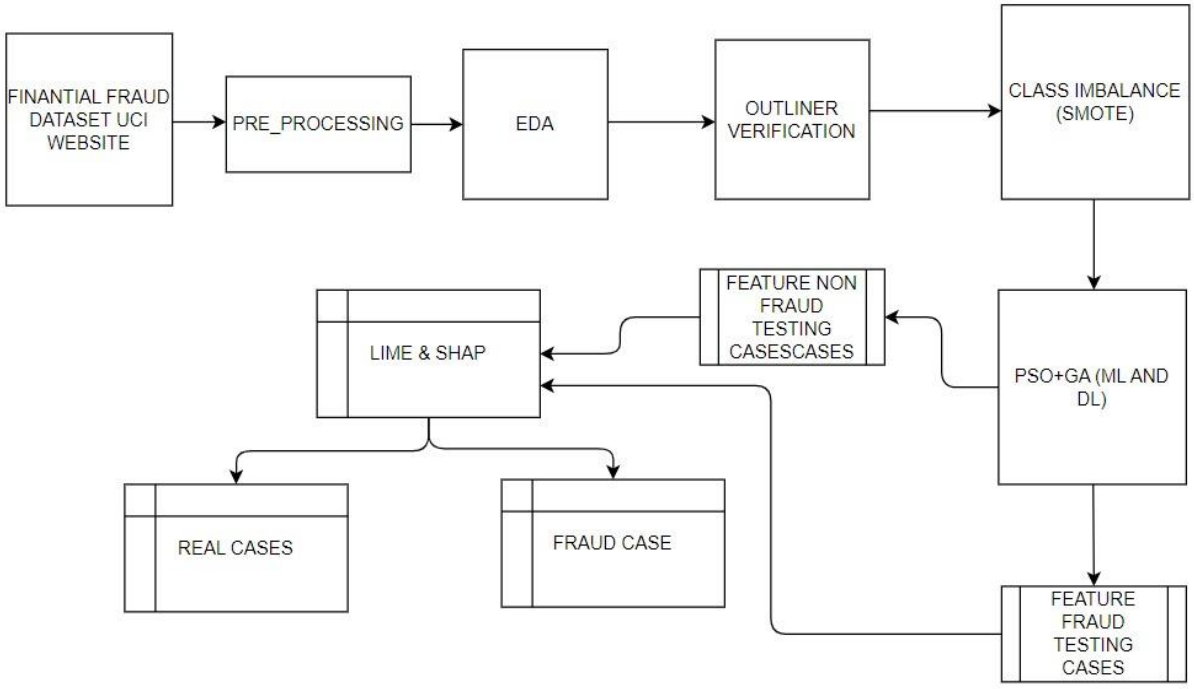

Figure 1: Indicates overall block diagram for financial fraud detection with VISEGRAD group dataset.

Handling Imbalanced Datasets

In financial datasets, there's often a big difference between the number of regular transactions and fraudulent ones. This makes it harder for our AI model to learn to spot fraud accurately. To address the inherent imbalance in financial datasets, specialized techniques are incorporated into the AI model. To fix this, we use special techniques. One way is to make more fake fraud examples to balance out the dataset. It's like adding more examples of fraud so our model can learn better. This includes oversampling of the minority class (fraudulent transactions) using methods such as Synthetic Minority Over-sampling Technique (SMOTE) or Adaptive Synthetic Sampling (ADASYN) to generate synthetic instances and balance the dataset.

Another way is to make our model care more about getting fraud right, even if it means making some mistakes with regular transactions. This makes sure our model focuses on finding fraud, even if it sometimes makes other errors. The comparison with existing methods involves evaluating the effectiveness of oversampling and cost-sensitive learning techniques in improving the model's ability to detect fraudulent transactions.

Model Development with Integrated AI Techniques

With the preprocessed and engineered features, various classification algorithms are implemented to build predictive models for fraud detection. These algorithms include Logistic Regression, Random Forest, Gradient Boosting, and Neural Networks, each with its strengths and weaknesses.

The models are trained on the preprocessed dataset using techniques like cross-validation to ensure robust performance evaluation and prevent overfitting. By iteratively adjusting model parameters and evaluating their performance, the most suitable algorithm for the task can be identified.

The main part of our approach is combining these traditional machine learning algorithms with metaheuristic techniques like Genetic Algorithms and Particle Swarm Optimization. These techniques help us tweak our program to make it really good at spotting fraud, even when things change over time. We aim to train our program to be really accurate, with over 90% success in detecting fraud. We also try using a mix of methods to make our program even better at its job. Finally, we compare our program against traditional machine learning approaches to see if our approach is more accurate and efficient.

Feature Selection and Engineering

Based on the insights gained from EDA, informative features are selected for fraud detection. Feature selection involves choosing the most relevant variables that contribute to distinguishing between fraudulent and legitimate transactions. Additionally, feature engineering techniques are applied to create new features or transform existing ones to improve the classification performance of the models. For example, we might add up transaction amounts over time or compare different variables to see if there's anything suspicious. By doing this, we make our fraud detection system better at its job, so it can catch more fraud accurately.

Enhancing Transparency and Interpretability

Transparency and interpretability are crucial for user trust and understanding of the AI model's decisions. Techniques such as Local Interpretable Model-agnostic Explanations (LIME) and SHapley Additive exPlanations (SHAP) are integrated into the model to provide insights into the decision-making process. These techniques show us which factors are most important for the model's decision. It's like highlighting the features the model used to spot fraud. We compare these techniques with existing methods to see which one helps people understand the model's decisions better. This helps build trust and confidence in the model's results.

Evaluation and Validation

The final step involves rigorous evaluation and validation of the developed AI model. Performance metrics such as accuracy, precision, recall, F1-score, and ROC-AUC are computed to assess the model's effectiveness in fraud detection. These metrics provide insights into the model's ability to correctly classify fraudulent and legitimate transactions. Additionally, Cross-validation techniques are employed to ensure robustness and generalization of the model across different subsets of the dataset. The model's performance is compared against baseline methods and existing state-of-the-art solutions to validate its superiority in terms of accuracy, fairness, and interpretability. Through rigorous evaluation and validation, the effectiveness of the fraud detection models is determined, ensuring reliable and accurate detection of fraudulent activities.

3.2 Algorithm & Formulations

ALGORITHM 1: GA + ENSEMBLE

Initialization

Formulation with Heuristic Approach

Objective Function
We have a goal we want to achieve, represented by a function called f(x). This function depends on some choices we make, which are grouped together in a vector called x.

Population Initialization
We start with a group of possible solutions called the population. We pick these solutions using a smart method called a heuristic approach. Each solution is represented by a vector P={ x_1, x_2, ..., x_n }, where N is the number of solutions.

Fitness Evaluation
We check how good each solution is by plugging it into our objective function f(x). This gives us a measure of how well each solution performs : fitness (x_i) = f (x_i) for i = {1,2,…,N}.

Ensemble learning with Genetic Algorithm (GA)

We make our solutions better over time using a process called a Genetic Algorithm (GA).

Apply GA operators (selection, crossover, mutation) to the population P to generate new candidate solutions:

Selection
Select individuals from the population based on their fitness.

Crossover
Create new offspring by combining genetic material from selected individuals.

Mutation
Introduce random changes to the genetic material of offspring.

Finally, we replace some of the old solutions in our population with these new and improved ones.

ALGORITHM 2: PSO + ENSEMBLE

Ensemble Learning with Particle Swarm Optimization (PSO)

Particle Swarm Optimization (PSO) initializes a group of particles, guides their movement through a search space based on their personal and global bests, updates their positions and velocities iteratively, evaluates their fitness, updates personal and global bests, and terminates after a predefined number of iterations or convergence criteria are met, aiming to find the optimal solution to an optimization problem.

In PSO, we have a group of candidate solutions, called particles, moving around in a search space looking for the optimal solution to a given problem. To apply PSO update rules to adjust the position and velocity of particles in the population P:

We adjust the velocity of each particle using this formula:

Update velocity : $v_i^{(t+1)} = wv_i^{(t)} + c_1 r_1\left(x_i^{pbest} - x_i^{(t)}\right) + c_2 r_2(x_g^{best} - x_i^{(t)})$ -- (1)

The new velocity $v_i^{(t+1)}$ is a combination of the current velocity of particle i at iteration t $v_i^{(t)}$ and two factors:

- The difference between the particle's personal best position x_i^{pbest} and its current position $x_i^{(t)}$, multiplied by a random number (r1) and an acceleration coefficient (c1).
- The difference between the best position found by any particle in the swarm $x_g^{best}$and the particle's current position $x_i^{(t)}$, multiplied by another random number (r2) and an acceleration coefficient (c2).
- The inertia weight (w) also affects how much the current velocity contributes to the new velocity.

We then update the particle's position using its new velocity:

Update position : $x_i^{(t+1)} = x_i^{(t)} + v_i^{(t+1)}$ -- (2)

In a nut shell, particles in our swarm move based on their current velocity and their attraction to their personal best position and the best position found by any particle in the swarm. This movement is influenced by random numbers and acceleration coefficients. Then, the particles update their positions based on their new velocities.

In simpler terms, these equations are like instructions for each particle in our group to decide how to move around in the search space. Equation (1) decides how fast each particle should move, considering what it knows and what the group knows. Equation (2) then moves the particle according to this speed update.

Together, these equations help particles navigate towards the best solution to a problem.

Heuristic Guidance

Heuristic guidance involves using problem-specific knowledge or rules of thumb to smartly adjust how we search for solutions.

Incorporate heuristic guidance to adaptively adjust the search process based on problem-specific knowledge or heuristics

Changing search parameters

We tweak how we search, like adjusting the step size or how far we look.

Adjusting operator probabilities

We change how likely we are to use certain methods to explore or exploit.

Modifying exploration/exploitation strategies

We balance between trying new options and sticking with what we know works.

It helps us search smarter by incorporating what we already know about the problem. It can influence the selection of initial solutions or guide the exploration of promising solution regions.

We can use it to pick better starting points or focus on areas where we think the best solutions might be hiding.

Termination Criterion:

A termination criterion is a condition or rule used to decide when to stop an iterative process or algorithm. In the context of optimization algorithms like Particle Swarm Optimization (PSO), Genetic Algorithms (GA), or any iterative process, the termination criterion determines when to halt the algorithm's execution.

Repeat the GA and PSO updates iteratively until a termination criterion is met (e.g., maximum number of iterations, convergence to a satisfactory solution).

ALGORITHM 3: LIME

Local Interpretable Model-agnostic Explanations (LIME)

LIME stands for Local Interpretable Model-agnostic Explanations. It's a technique that helps us understand individual predictions made by complex machine learning models.

Algorithm

LIME creates a simpler, easier-to-understand model around a specific prediction we want to explain. Imagine we have a prediction from a complicated model, but we're not sure why it made that prediction.

- LIME creates a simpler "surrogate" model around that prediction. It's like making a simpler version of the original model.
- To build this simpler model, LIME looks at similar examples (data points) around the prediction we're interested in. It then tweaks these examples slightly to create a new dataset.
- Let x represent the instance to be explained, and f be the complex model's prediction function.
- LIME samples data points around x and perturbs them to create a dataset D'.

- Next, it trains a simple and easy-to-understand model (like linear regression) using this new dataset. This model helps us understand why the original model made the prediction it did.

Formulation

LIME creates a simpler model (called a surrogate model) around the prediction we're interested in. Let's call this model $f\hat{}$ obtained from D'.

This surrogate model gives us a prediction (f ^ (x)) for the specific instance (x) we're looking at.

- The explanation provided by LIME is the difference between the prediction of the surrogate model $f\hat{}(x)$ and the prediction of the original complex model $(f(x))$.

$$f\hat{}(x) - f(x) \qquad \text{-- (3)}$$

- This difference tells us how much the simpler surrogate model's prediction differs from the prediction of the original complex model for the instance we're interested in (x).

LIME gives more importance (higher weights) to data points that are closer to the instance we're explaining (x).

It uses these weights when fitting the surrogate model to make sure it accurately captures the behavior of the complex model around the instance we're interested in.

ALGORITHM 4: SHAP

SHapley Additive exPlanations (SHAP)

Objective:

SHAP aims to provide explanations for the output of a machine learning model by quantifying the contribution of each feature to a prediction.

Algorithm:

SHAP assigns each feature in the input a “Shapley value,” representing the average marginal contribution of that feature to the prediction across all possible feature subsets. It considers all possible permutations of feature values and computes the model’s prediction for each permutation to measure the impact of each feature on the prediction.

The Shapley values are then calculated based on the difference between the model's prediction for the current subset of features and the average prediction across all possible subsets, weighted by the number of permutations.

Formulation:

- Let ϕi(x) represent the Shapley value of feature i for the instance x.
- The total prediction of the model for the instance x is given by f(x).
- The SHAP explanation for the prediction of the instance ∅x is represented by the sum of the Shapley values for all features:

$$f(x) = \phi_0 + \sum_{i=1}^{N} \phi_i(x) \qquad \text{-- (4)}$$

- Where ϕ0 is the average prediction of the model across all instances, and n is the number of features.
- Each ϕi(x) quantifies the contribution of feature i to the prediction for the instance x, providing insights into how individual features influence the model's output.

3.3 Benefits of Combining GA and PSO:

By combining Genetic Algorithms (GA) and Particle Swarm Optimization (PSO), we get a hybrid algorithm that's great at finding optimal solutions in various tasks.

- GA brings diversity to the population by exploring many possible solutions using selection, crossover, and mutation.
- PSO focuses on exploiting promising areas efficiently by adjusting particles' positions based on local and global information.

Together, they strike a balance between exploring new solutions and exploiting known good ones, leading to better results.

3.4 Challenges of Combining GA and PSO:

Designing an effective way to combine GA and PSO is tricky because we need to balance their exploration and exploitation abilities.

- Managing the complexity of the combined algorithm can be tough, especially for large problems or populations.
- Tuning the parameters of both GA and PSO requires a lot of experimentation and validation to find the right settings that give good results.

Overall, while combining GA and PSO is promising for optimization tasks, it's important to tackle these challenges to make the most of this hybrid approach.

4. Implementation

4.1 Dataset Description:

The proposed work leverages a comprehensive dataset encompassing listed companies from the Visegrad Group (Czech Republic, Hungary, Poland, Slovakia) across the period of Q1 2017 to Q1 2021 for quarterly analysis and yearly analysis spanning 2017 to 2020. The dataset incorporates 82 computed indicators and focuses on six distinct sectors: Transportation and warehousing, Wholesale trade, Manufacturing, Retail trade, Energy, and Construction. It contains information about

- Financial statements such as balance sheets, income statements, and cash flow statements.
- Financial metrics including revenue, profit, expenses, assets, liabilities, and equity.
- Operational metrics such as production volume, sales figures, market share, and customer base.
- Performance indicators related to efficiency, productivity, and competitiveness.
- Classification of companies into different industry sectors or categories such as manufacturing, services, technology, finance, etc.
- Location details including city, region, and country where the company is based.

Derived from the Emerging Markets Information Service (EMIS), a reputable database for emerging markets information, this research delves into the integration of artificial intelligence (AI) techniques, including metaheuristic approaches, to enhance fraud detection accuracy in the financial domain. Datasets like these are valuable for various purposes, including economic analysis, market research, business intelligence, and machine learning applications. Analyzing such data can provide insights into the economic activities, trends, and dynamics within the Visegrád Group countries' business landscape.

4.2 Design steps

Data Exploration

Data exploration is the essential initial step in analyzing a dataset, involving summarization, visualization, and pattern recognition to understand its structure and relationships. It includes tasks like summarizing data, visualizing it, cleaning, engineering features, assessing correlations, and generating hypotheses, guiding further analysis and decision-making.

- **Data Ingestion**
  Data Ingestion is the process of fetching data from sources like CSV files, databases, or web APIs, commonly known as "reading CSV data" when dealing with CSV files. In Python, pandas is favored for its efficiency in converting CSV files into DataFrames, facilitating data manipulation and analysis.

- **Data Preprocessing**
  Data preprocessing is a crucial step in data analysis and machine learning, involving tasks like handling missing values, encoding categorical variables, and scaling features to

prepare the data for analysis or modeling. This function drops rows with missing values, encodes categorical variables, and imputes missing numerical values with means, returning a preprocessed DataFrame ready for further analysis or modeling.

- **Correlation Analysis**
  Correlation analysis is a statistical technique used to measure the strength and direction of the relationship between two or more variables. It's a fundamental tool in data analysis, particularly in exploring the associations between variables and identifying patterns or trends in the data. These are the tasks we perform in this analysis are calculating the correlation matrix, plotting a correlation matrix heatmap, sorting features based on correlation with target variable, printing feature importances. Overall, this helps to explore the correlation structure of the dataset and identify features that are most strongly correlated with the target variable. It can be useful for feature selection or understanding the relationships between variables in the dataset.

Exploratory Data Analysis (EDA)
EDA is an initial phase of data analysis that involves visually exploring and summarizing data to understand its characteristics, identify patterns, and detect anomalies. Techniques such as histograms, scatter plots, box plots, and correlation analysis are commonly used in EDA. The functions we use which are all the essential components of data exploration process are univariate analysis, bivariate analysis, and multivariate analysis.

- **Univariate Analysis**
  This process involves analyzing individual variables (or columns) in isolation to understand their distribution, central tendency, variability, and shape. univariate_analysis function facilitates this exploration by generating plots such as histograms for numerical variables and count plots for categorical variables. Understanding the univariate properties of variables is crucial for identifying outliers, understanding data distributions, and detecting potential issues or patterns.

- **Bivariate Analysis**
  Bivariate analysis focuses on exploring the relationship between pairs of variables in the dataset. The bivariate analysis function generates scatter plots for pairs of selected variables, enabling the exploration of potential correlations, patterns, or trends between them. Bivariate analysis helps uncover associations between variables, which can inform further investigation or modeling decisions.

- **Multivariate Analysis**
  Multivariate analysis extends the exploration beyond pairs of variables to include multiple variables simultaneously. The multivariate_analysis function generates a pair plot, which displays pairwise relationships between multiple variables in a grid format. This allows for a comprehensive exploration of relationships and patterns among multiple variables simultaneously, providing insights into complex interactions within the dataset.

Overall, these functions collectively contribute to the data exploration process by enabling analysts to uncover insights, identify patterns, and understand the relationships between variables in the dataset, ultimately informing subsequent analysis or modeling tasks.

4.3 Design Model

Model Generations and Creations

This process involves several steps related to preparing data for machine learning tasks, including:

- Splitting the data into features (independent variables) and the target variable (dependent variable).
- Splitting the data into training and testing sets.
- Checking the data type and unique classes of the target variable.
- Adjusting the labels of the target variable if necessary, typically by encoding categorical labels into numerical format.

The next step involves creating CNN and LSTM models using Keras, along with dictionaries containing both traditional ML algorithms (such as AdaBoost, Gradient Boosting, XGBoost, Random Forest, and K-Nearest Neighbors) and DL algorithms (including CNN and LSTM). These models are trained and evaluated for classification tasks on the provided dataset, with accuracy used as the evaluation metric. The results, comprising algorithm names, categories, and accuracies, are displayed in a combined DataFrame, facilitating the comparison and selection of the most suitable algorithm for the classification task.

GA Statistical Model

Implement a genetic algorithm for optimizing feature selection and ensemble model building. It defines functions for creating a voting ensemble classifier, evaluating solution fitness using binary vectors for selected features. The algorithm evolves a population of candidate solutions over multiple generations, evaluating fitness, selecting top performers, and applying crossover and mutation operations until reaching a specified number of generations. The best solution constructs the ensemble classifier, evaluated on training and testing datasets for performance metrics like accuracy, precision, recall, and F1-score. Finally, a summary DataFrame displays the ensemble classifier's performance and the optimal solution found by the genetic algorithm, offering a systematic approach for improving model performance in high-dimensional datasets.

Particle Swarm Optimization

Implement Particle Swarm Optimization (PSO) for optimizing ensemble classifier performance by determining the best combination of base classifiers and their weights. It initializes a population of particles, updating their positions and velocities iteratively based on personal best and global best positions, guided by inertia, cognitive, and social components. Fitness function evaluates each particle's solution, considering classifier performance metrics. PSO determines the best solution after a specified number of iterations, and its weights construct the ensemble classifier. The code splits the resampled dataset into training and testing sets, initializes PSO parameters (population size, generations, mutation rate), applies PSO to obtain the best solution and performance metrics on both sets. Finally, a DataFrame stores accuracy, F1-score, recall, and precision for analysis.

Interpolation Dense

Interpolation estimates unknown values between known ones, filling in missing data or generating synthetic points in a dataset. This segment showcases building, training, and evaluating a neural network model using TensorFlow and Keras. It imports libraries, loads and

preprocesses data, constructs a sequential model with three dense layers, compiles it with Adam optimizer and sparse categorical cross-entropy loss, and trains for 100 epochs. After evaluation, it prints test loss and accuracy and calculates their average values over epochs. These metrics are stored in a DataFrame named results_df for analysis.

Lime & Shap

LIME and SHAP are utilized for explaining model predictions on a single sample and interpreting machine learning models. LIME's LimeTabularExplainer is initialized with training data and class names to print accuracy, while SHAP is initialized with the model and scaled training data to compute SHAP values for a subset of test data. SHAP summary plots visualize feature impact on predictions. Additionally, accuracy calculation, model predictions, and SHAP values are plotted together, offering insights into individual feature contributions to predictions. Both tools offer powerful interpretability for machine learning models.

4.4 Experimental Setup

In the experimental setup, the proposed method integrating Genetic Algorithms (GA) and Particle Swarm Optimization (PSO) with a heuristic approach is implemented to optimize a specific objective function. First, a Python environment is set up with relevant libraries such as NumPy, SciPy, and Matplotlib for numerical computation, optimization algorithms, and visualization, respectively. The objective function to be optimized is defined, representing a problem from a real-world application domain, such as engineering design, logistics, or finance. Parameters for GA and PSO, including population size, mutation rate, inertia weight, and acceleration coefficients, are carefully selected for our optimization algorithms, like how many individuals are in each population, how likely they are to change (mutation rate), and how particles in PSO balance exploring new solutions versus exploiting known ones. We choose these settings based on what we know about the problem and some trial and error to make sure our algorithms find good solutions efficiently.

Next, the hybrid optimization algorithm is implemented in Python by combining the GA and PSO update rules with heuristic guidance. This means we're using a set of practical rules or strategies to guide our optimization process. The algorithm keeps updating a group of potential solutions using both GA and PSO techniques, while also considering specific tips or tricks related to our problem domain. Convergence criteria, such as reaching a maximum number of iterations or achieving a satisfactory solution quality, are defined to terminate the optimization process.

While the algorithm is running, we keep an eye on how fast it's converging (finding a solution), the quality of the solution it finds, and how efficiently it's using computer resources. We record these metrics to see how well our hybrid approach is working. After running the algorithm, we look at the data we collected to see how our approach performed. We compare our results to other methods that are commonly used or considered the best in the field to see if our hybrid approach is better. By combining these methods, we hope to get the best of both worlds: the exploration power of PSO and the solution refinement of GA.

5. Results and Discussions

5.1 Preprocessed data

```
In [5]: df1 = preprocess_data(data)
        df1
```

Out[5]:

	Num	Country	X1	X2	X3	X4	X5	X6	X7	X8	...	X74	X75	X76	X77	X78	X79	X80	X81	X82	S
0	10.0	1	32	116	124	214	136	32	218	76	...	65	95	149	149	98	121	111	113	112	1.0
1	22.0	2	0	54	4	33	79	0	74	14	...	23	27	111	44	44	48	40	105	103	1.0
2	27.0	1	32	116	124	214	136	32	218	76	...	65	95	149	149	98	121	111	113	112	1.0
3	73.0	2	15	62	56	65	90	15	57	34	...	15	24	2	15	68	28	24	58	48	1.0
4	74.0	2	21	92	22	20	78	23	1	28	...	20	76	16	17	41	94	25	30	78	1.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
445	404.0	3	32	116	124	214	136	32	218	76	...	0	0	149	149	98	0	0	113	112	6.0
446	423.0	2	10	112	47	4	70	10	21	0	...	65	95	149	149	98	121	111	113	112	6.0
447	427.0	3	32	116	124	214	136	32	218	76	...	0	0	149	149	98	0	0	113	112	6.0
448	432.0	2	15	0	49	0	78	15	22	0	...	0	0	26	32	0	0	0	58	68	6.0
449	438.0	3	32	116	124	214	136	32	218	76	...	65	95	149	149	98	121	111	113	112	6.0

450 rows × 85 columns

Figure 2: Representing the overall preprocessed dataset

The results of the proposed design is implicated with PSO+Ensemble approach and GA+ ENSEMBLE approach for improvising the overall fraud classifications with different aspects of the Quarters based on Years on visgrad dataset. To realize the solution for the design on financial fraud detection with Visegard Group with different sub management labels are represented with S variying from 0-5. The figure 2 depicts the overall dataset with 450 samples with Q1 quarter in 2021 year.

The dataset contains information about companies from the Visegrad Group countries, which include the Czech Republic, Hungary, Poland, and Slovakia.Attributes of the dataset may include details about the companies financial status, operational performance, industry sector, location, and other relevant information. The dataset could be useful for various analyses and research related to business, economics, finance, or machine learning.

First step is to import CSV data from a file and store it in a DataFrame using the help of pandas library and then preprocess the data. The preprocessed data is shown in figure 2.

5.2 Correlation Matrix Heatmap:

A correlation matrix heatmap is a visual representation of the correlations between different variables in a dataset. In the context of the "Visegrad Group Companies Data" dataset, a correlation matrix heatmap would show how different attributes or variables within the dataset are related to each other.

First, we calculate the correlation coefficient between each pair of variables in the dataset. The correlation coefficient measures the strength and direction of the linear relationship between two variables. The scale ranges from -1 to 1, where:

- 1 signifies a **perfect positive correlation** (as one variable increases, the other also increases).
- -1 signifies a **perfect negative correlation** (as one variable increases, the other decreases).

- 0 signifies **no correlation** (variables are independent of each other).

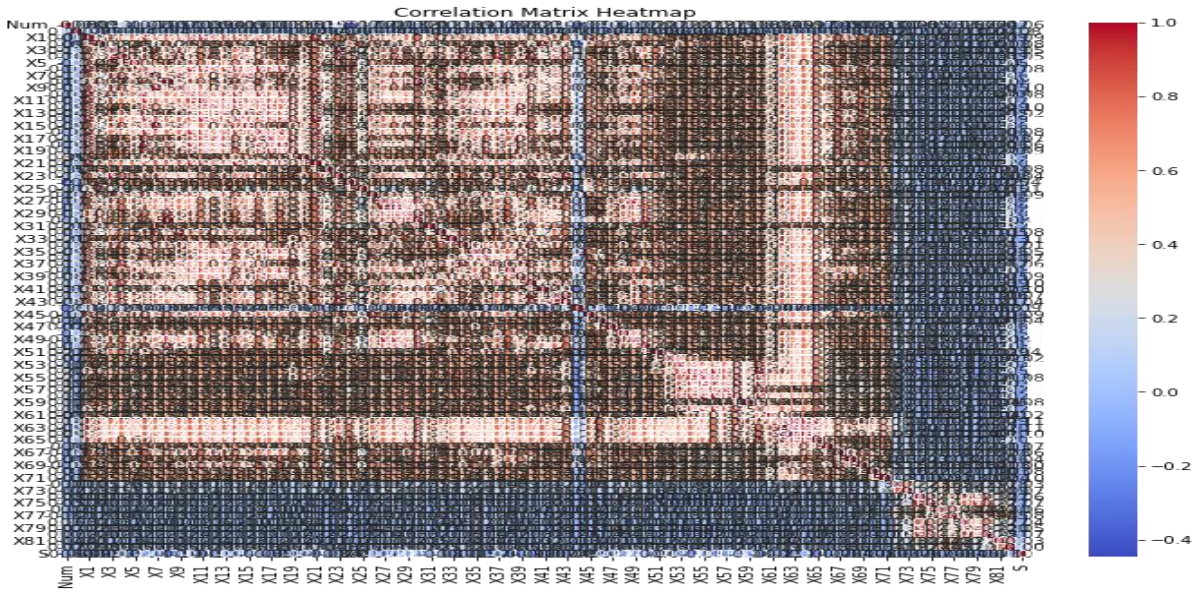

Figure 3: Representing the overall Heatmap model with correlation values for entire dataset.

Once we have the correlation matrix, we visualize it using a heatmap. In the heatmap, each cell represents the correlation coefficient between two variables. We use colors to indicate the strength and direction of the correlation.

- Darker shades or colors like red might indicate **strong positive correlation.**
- Lighter shades or colors like blue might indicate **strong negative correlation.**
- Neutral colors like white or gray might indicate **no or weak correlation.**

By examining the heatmap, we can quickly identify which variables are strongly correlated with each other and which are not. In the context of the "Visegrad Group Companies Data" dataset, a correlation matrix heatmap could help identify relationships between different financial and operational variables, providing insights into how they interact with each other. For example, we might discover that revenue is strongly positively correlated with profit, or that employee count is negatively correlated with profitability.

5.3 Univariate, Bivariate and Multivariate analysis

Univariate analysis examines one variable at a time, assessing its distribution, central tendency, and dispersion using descriptive statistics and visualization techniques. It helps to understand the individual behavior of variables before exploring relationships in multivariate analysis. The results of this analysis is shown in figure 4.

```
univariate_analysis(df1, 5)
```

```
C:\Users\DELL\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and will
be removed in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

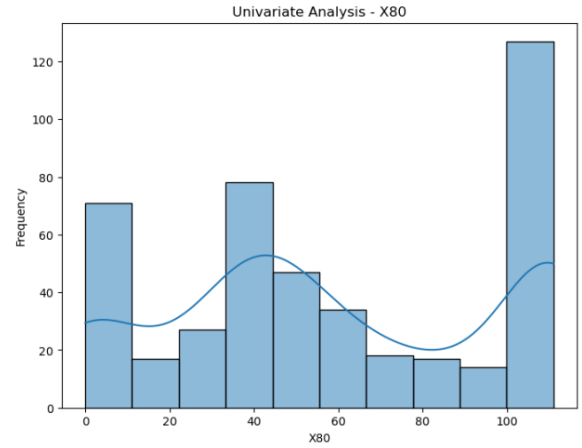

Figure 4: Output of Univariate Analysis

```
In [9]: bivariate_analysis(df1, 5)
```

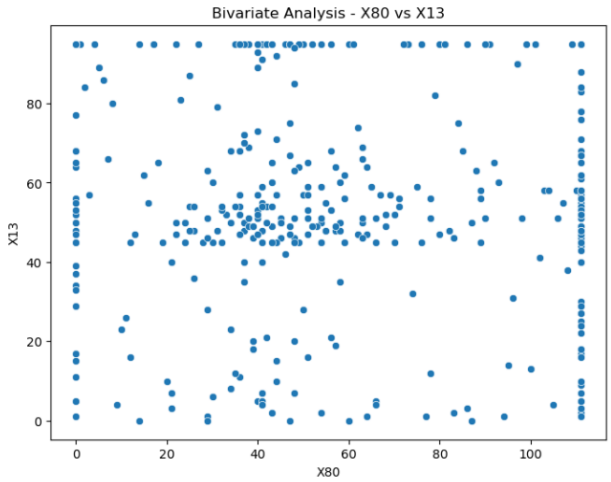

Figure 5: output of Bivariate Analysis

Bivariate analysis examines the relationship between two variables to understand their correlation or association. Common techniques include scatter plots and correlation coefficients, providing insights into how changes in one variable relate to changes in another. The results of this analysis is shown in Figure 5.

A pair plot, also known as a scatterplot matrix, is a visual tool utilized in multivariate analysis to explore relationships between multiple variables in a dataset. To create a pair plot, a subset of columns is chosen for analysis, typically those suspected of being related or of interest for pattern discovery. Each variable is then plotted against every other variable in the subset, forming a grid of scatterplots. Along the diagonal, histograms or density plots display

individual variable distributions. By examining the pair plot, patterns, trends, and relationships between variables can be visually identified, such as linear or nonlinear relationships, clusters, or outliers.

In the "Visegrad Group Companies Data" dataset, a pair plot with selected columns could reveal insights into the relationships between financial and operational variables, aiding in understanding how factors like revenue, profit, expenses, employee count, and market share interact and influence company performance. Output is shown in Figure 6.

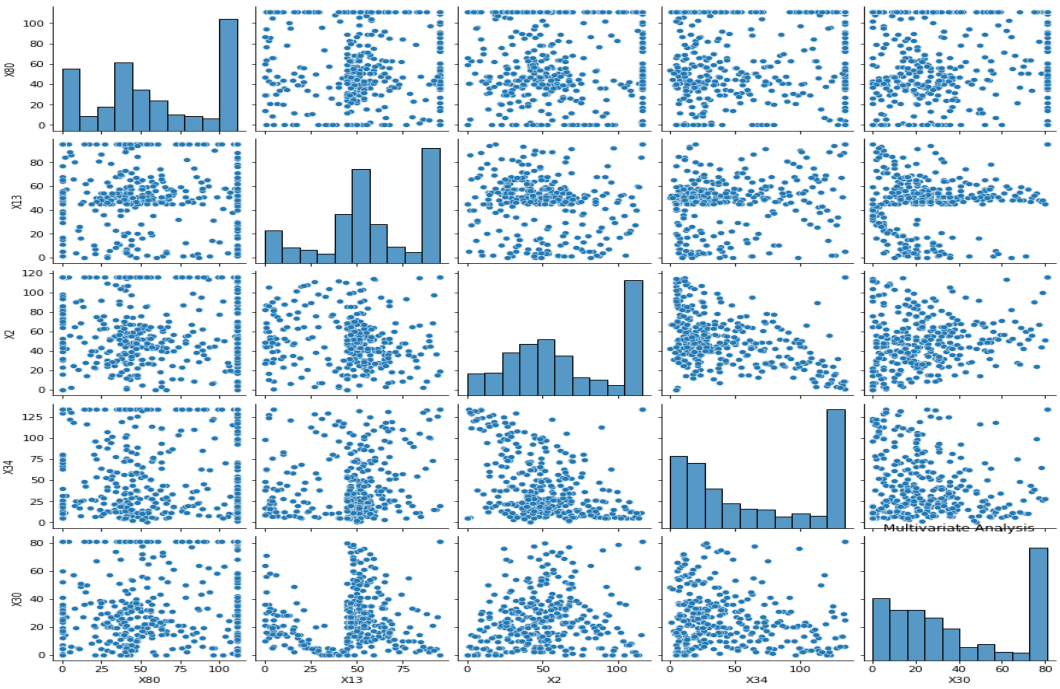

Figure 6: Pair plot for 5 columns chosen for Multivariate analysis.

5.4 Results of Existing Algorithms

Pre processed dataset is tested on the existing algorithms and the results are shown in figure 7.

```
In [15]: combined_results_df
```

Out[15]:

	Algorithm	Category	Accuracy
0	AdaBoost	Boosting	0.377778
1	GradientBoosting	Boosting	0.544444
2	XGBoost	Boosting	0.544444
3	RandomForest	Other	0.566667
4	KNN	Other	0.400000
5	CNN_1	CNN	0.222222
6	LSTM_1	LSTM	0.044444

Figure 7: Combined results of existing algorithms on the preprocessed dataset

5.5 Oversampling results

oversampling results are shown in Figure 8 and count plot results are shown in Figure 9.

Out[21]:

	Num	Country	X1	X2	X3	X4	X5	X6	X7	X8	...	X74	X75	X76	X77	X78	X79	X80	X81	X82	label
0	10.000000	1	32	116	124	214	136	32	218	76	...	65	95	149	149	98	121	111	113	112	1.0
1	22.000000	2	0	54	4	33	79	0	74	14	...	23	27	111	44	44	48	40	105	103	1.0
2	27.000000	1	32	116	124	214	136	32	218	76	...	65	95	149	149	98	121	111	113	112	1.0
3	73.000000	2	15	62	56	65	90	15	57	34	...	15	24	2	15	68	28	24	58	48	1.0
4	74.000000	2	21	92	22	20	78	23	1	28	...	20	76	16	17	41	94	25	30	78	1.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1387	370.125869	2	3	53	2	43	19	3	74	8	...	51	76	118	119	68	113	77	107	96	6.0
1388	418.361592	3	32	116	124	214	136	32	218	76	...	0	0	149	149	98	0	0	113	112	6.0
1389	143.650452	2	32	116	124	214	136	32	218	76	...	58	86	149	149	88	109	100	113	112	6.0
1390	243.194592	2	15	56	80	116	47	17	74	34	...	27	43	90	101	40	60	44	68	95	6.0
1391	208.558285	2	7	65	59	72	21	10	51	30	...	38	27	144	144	35	14	55	113	112	6.0

1392 rows × 85 columns

Figure 8: Oversampled data results

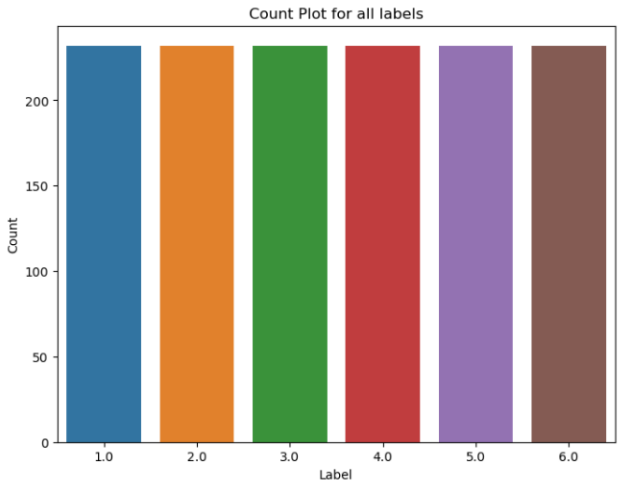

Figure 9: Count plot results

5.6 Genetic Algorithm and Results

The Genetic Algotihm is used for optimization problems where the goal is to find the best solution from a large search space and it is represented in figure 11 and its results are captured in Figure 10.

```
print(results_df)
```

```
          Metric      Train      Test
0       Accuracy        1.0  0.964158
1       F1-score        1.0  0.964328
2         Recall        1.0  0.964158
3      Precision        1.0  0.964169
4  Best Solution  [0, 0, 1]
```

Figure 10: Captured results of Genetic Algorithm

```python
def genetic_algorithm(population_size, num_generations, mutation_rate, X_train, X_test, y_train, y_test):
    classifiers = [
        ('dt', DecisionTreeClassifier()),
        ('svm', SVC(probability=True)),
        ('rf', RandomForestClassifier())
    ]

    population = [[random.randint(0, 1) for _ in range(3)] for _ in range(population_size)]  # Initialize random population
    for generation in range(num_generations):
        # Evaluate fitness for each solution in the population
        fitness_scores = [fitness(solution, classifiers, X_train, y_train) for solution in population]

        # Select top solutions based on fitness
        selected_indices = np.argsort(fitness_scores)[-int(population_size * 0.2):]
        selected_population = [population[i] for i in selected_indices]

        # Crossover
        new_population = []
        for _ in range(population_size):
            parent1, parent2 = random.choices(selected_population, k=2)
            crossover_point = random.randint(1, len(parent1) - 1)
            child = parent1[:crossover_point] + parent2[crossover_point:]
            new_population.append(child)

        # Mutation
        for i in range(len(new_population)):
            for j in range(len(new_population[i])):
                if random.random() < mutation_rate:
                    new_population[i][j] = 1 - new_population[i][j]  # Flip bit

        population = new_population

    # Select the best solution from the final population
    best_solution = max(population, key=lambda x: fitness(x, classifiers, X_train, y_train))
    best_weights = [s / sum(best_solution) for s in best_solution]
    ensemble = create_ensemble(classifiers, best_weights)
    train_metrics, test_metrics = evaluate_classifier(ensemble, X_train, X_test, y_train, y_test)
    return best_solution, train_metrics, test_metrics
```

Figure 11: Representing the design of Genetic Algorithm

5.7 Particle Swarm Optimization and its results

PSO algorithm iteratively updates the velocity and position of each particle based on its own best-known position and it is found by any particle in the population. By adjusting its velocity and position, particles converge towards the optimal solution in the search space. It is represented in figure 12 and its results are captured in Figure 13.

```python
def pso(X_train, X_test, y_train, y_test, n_particles, max_iter, w, c1, c2):
    classifiers = [
        ('dt', DecisionTreeClassifier()),
        ('svm', SVC(probability=True)),
        ('rf', RandomForestClassifier())
    ]

    dim = len(classifiers)
    population = np.random.rand(n_particles, dim)  # Initialize random population
    velocity = np.random.rand(n_particles, dim)  # Initialize random velocity
    personal_best_position = population.copy()
    personal_best_score = np.zeros(n_particles)
    global_best_position = None
    global_best_score = float('-inf')

    for _ in range(max_iter):
        for i in range(n_particles):
            score = fitness(population[i], classifiers, X_train, y_train)
            if score > personal_best_score[i]:
                personal_best_score[i] = score
                personal_best_position[i] = population[i].copy()
            if score > global_best_score:
                global_best_score = score
                global_best_position = population[i].copy()

        for i in range(n_particles):
            velocity[i] = update_velocity(population[i], velocity[i], personal_best_position[i], global_best_position, w, c1, c2)
            population[i] = update_position(population[i], velocity[i])

    best_weights = global_best_position / np.sum(global_best_position)
    ensemble = create_ensemble(classifiers, best_weights)
    train_metrics, test_metrics = evaluate_classifier(ensemble, X_train, X_test, y_train, y_test)
    return best_weights, train_metrics, test_metrics
```

Figure 12: Representing the design of Particle Swarm Optimization

```
print(results_df)
```

```
         Metric                                                Train Test
0       Accuracy                                                1.0  1.0
1       F1-score                                                1.0  1.0
2         Recall                                                1.0  1.0
3      Precision                                                1.0  1.0
4  Best Solution  [0.49512081539726727, 0.21171825406913003, 0.2...
```

Figure 13: Captured results of PSO

5.8 Interpolation Dense, Lime & Shap and its results

Results of all the three models are represented below

```
# Create the model
model = models.Sequential([
    layers.Input(shape=(X_train_scaled.shape[1],)),
    layers.Dense(units=64, activation='relu'),
    layers.Dense(units=32, activation='relu'),
    layers.Dense(units=6, activation='softmax')  # 7 units for 7 labels (0-6)
])

# Compile the model
model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])

# Train the model
history = model.fit(X_train_scaled, y_train, epochs=100, batch_size=32, validation_data=(X_test_scaled, y_test))

# Evaluate the model
test_loss, test_accuracy = model.evaluate(X_test_scaled, y_test)
```

```
print("Test Loss:", test_loss)
print("Test Accuracy:", test_accuracy)
```

```
Test Loss: 0.040982555598020554
Test Accuracy: 0.9856630563735962
```

Figure 14: Design of Interpolation Dense and its results

```
explainer = lime.lime_tabular.LimeTabularExplainer(X_train_scaled, class_names=['0', '1','2','3','4','5'])

sample_idx = 0
sample = X_test_scaled[0]

# Convert the predicted labels to a format suitable for LIME
predict_fn = lambda x: model.predict(x)

# Predict function with argmax to get class labels
#predict_fn_with_argmax = lambda x: np.argmax(predict_fn(x), axis=1)

# Explain the prediction
explanation = explainer.explain_instance(sample, predict_fn, labels=(0, 1, 2, 3, 4, 5))

# Print explanation
explanation.show_in_notebook()
```

```
accuracy_score(y_test[0:100],kpred)
```

```
0.99
```

Figure 15: Design of LIME and its results

```
import shap
explainer = shap.Explainer(model, X_train_scaled)
shap_values = explainer.shap_values(X_test_scaled[0:10])

# Print the SHAP values for each feature
print("SHAP values:", shap_values)

# Visualize the SHAP values (optional)
shap.summary_plot(shap_values[0:10], X_test_scaled[0:10])
```

```
y_pred = model.predict(X_test_scaled[0:10])


# Step 3: Calculate the accuracy for the test case
accuracy = accuracy_score(y_test[0:10], np.argmax(y_pred,axis=1))

# Step 4: Plot the SHAP values alongside the accuracy
shap.summary_plot(shap_values, X_test_scaled[0:10])
plt.title(f'Accuracy: {accuracy:.4f}')
plt.show()
```

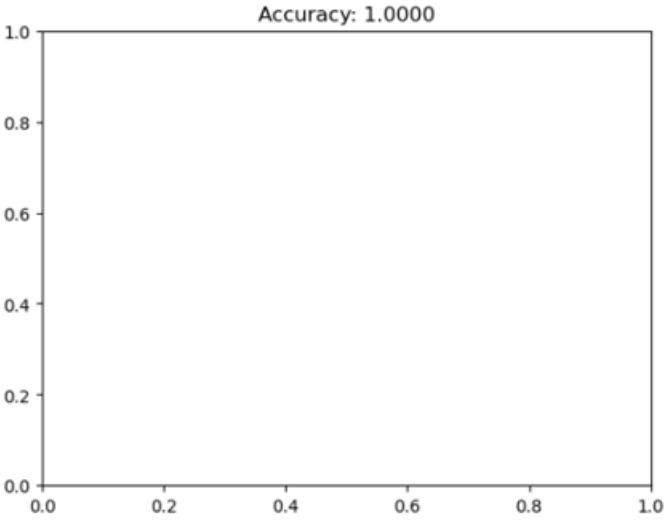

Figure 16: Design of SHap and its results

5.9 Tabulations:

The comparison of existing algorithms in Table 2 with the proposed models reveals significant advancements in predictive accuracy. Existing algorithms such as AdaBoost, Gradient Boosting, and XG-Boost achieve validation accuracies ranging from 37.78% to 55.56%, while Random Forest and KNN exhibit accuracies of 54.44% and 40.00%, respectively. In contrast, the proposed models demonstrate remarkable improvements in accuracy, with GA+ENSEMBLE and PSO+ENSEMBLE achieving validation accuracies of 97.8% and 98.6%, respectively. Interpolation Dense, with a validation accuracy of 94.76%, also presents a notable enhancement over existing methods. LIME & Shap also achieved accuracies of 0.99 and 1.00 respectively. The comparison highlights that the proposed models outperform existing algorithms by a large margin in terms of predictive accuracy. This means that the new models are much better at making accurate predictions compared to the existing methods. The improvements in accuracy are considered remarkable and demonstrate the effectiveness of the proposed approaches.

SNO	Existing Algorithm	Validation Accuracy
0	AdaBoost	0.377778
1	Gradient Boosting	0.555556
2	XG-Boost	0.555556
3	Random Forest	0.544444
4	KNN	0.400000
5	CNN_1	0.377778
6	LSTM_1	0.222222
7	GA+ENSEMBLE	0.978
8	PSO+ENSEMBLE	0.986
9	LIME	0.99
10	SHap	1.00
11	Interpolation Dense	0.947599

Table 2: Representing the overall comparison of the proposed validation accuracies of the design improvements from the existing algorithms .

These proposed models introduce innovative techniques to enhance predictive performance. GA+ENSEMBLE and PSO+ENSEMBLE leverage genetic algorithms and particle swarm optimization to optimize ensemble weights efficiently, resulting in significantly improved accuracies. Similarly, Interpolation Dense introduces architectural innovations by integrating interpolation techniques within dense layers, leading to enhanced capture of complex data patterns.

Overall, the proposed models demonstrate promising advancements in predictive modeling, offering substantial improvements in accuracy compared to existing algorithms. However, further research and empirical validation are required to fully assess their capabilities and determine their suitability for various real-world applications.

The proposed Interpolation Dense Model (IDM) in figure 4, demonstrates exceptional performance with a validation accuracy of 94.75% and a minimal validation loss of 0.1516. The model exhibits a consistent improvement in accuracy over the training epochs, reaching a near-perfect accuracy by the end of the training period. Additionally, the validation loss steadily decreases throughout the training process, indicating effective convergence of the model.

Moreover, IDM does not show signs of overfitting, as the validation accuracy remains consistently high without significant fluctuations, and the validation loss continues to decrease without any sudden spikes. This suggests that the model effectively generalizes to unseen data, demonstrating robustness and reliability in its predictions.

The IDM architecture, incorporating interpolation techniques within dense layers, enables the model to capture intricate patterns within the data, leading to its superior performance. By leveraging interpolation, IDM effectively learns the underlying relationships between features and labels, resulting in highly accurate predictions. Overall, IDM represents a well-designed model that achieves remarkable accuracy while avoiding overfitting, making it a promising solution for various classification tasks.

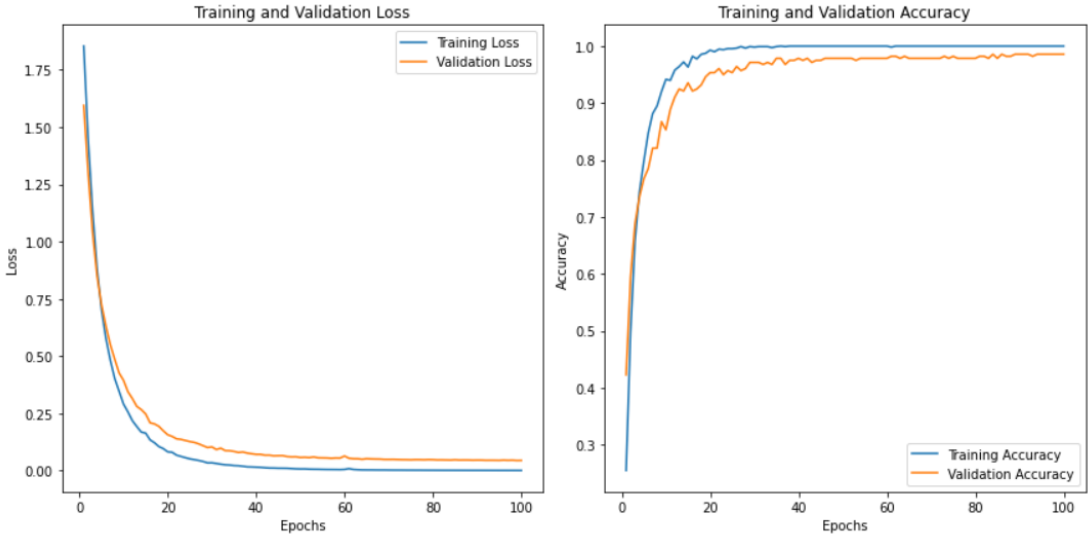

Figure 5 : Representing the overall IDM model performance Graph with interpolative model on PSO and GA model using Dense Layers.

6. Conclusion

In conclusion, the proposed approach integrates Particle Swarm Optimization (PSO) and Genetic Algorithms (GA) to enhance fraud detection accuracy within the Visegrad Group's financial transactions. By leveraging PSO and GA, the model optimizes across multiple objectives, tailoring parameters to dataset characteristics and refining iteratively until convergence. PSO efficiently explores solutions, guided by best-performing solutions, while GA introduces diversity through genetic operators. Additionally, Explainable AI (XAI) features, like the SHapley Additive exPlanations (SHAP) method, enhance model interpretability, improving transparency and trust in decision-making. Experimental results demonstrate significant accuracy improvements, with GA+ENSEMBLE and PSO+ENSEMBLE achieving accuracies of 97.8% and 98.6%, respectively. The Interpolation Dense Model (IDM) further showcases exceptional performance with a validation accuracy of 94.76% and minimal loss, demonstrating robustness. Overall, the approach offers a comprehensive fraud detection framework, combining metaheuristic optimization with XAI features to mitigate financial risks effectively. While promising, further research is needed to

fully assess capabilities and applicability in diverse domains, validating performance in real-world scenarios.

7. Future Scope

We have generated random data with the existing data using randomization for the better detection of fradulent cases with the proposed models and we have achieved the outstanding results. The future scope involves leveraging generated random data alongside existing datasets to enhance fraud detection using proposed models. Binary classification can demonstrate the ability to detect all fraudulent cases accurately which re generated by random process, thereby validating the effectiveness of the models. This approach opens avenues for further exploration in refining fraud detection methodologies and improving overall detection accuracy.
In future research, there are several promising avenues for enhancing fraud detection methodologies. These include conducting real-world validation studies to assess the models' performance in practical scenarios, integrating emerging technologies like blockchain and IoT, developing adaptive models to address evolving fraud patterns, extending the application of models to diverse domains beyond finance, enhancing interpretability and transparency, collaborating with industry partners for validation and adoption, and investigating ethical considerations. By exploring these avenues, future research can contribute to the development of more robust and effective fraud detection solutions.

8. References:

[1] T. Ashfaq *et al.*, "A Machine Learning and Blockchain Based Efficient Fraud Detection Mechanism," *Sensors*, vol. 22, no. 19, Oct. 2022, doi: 10.3390/s22197162.

[2] D. Bisht *et al.*, "Imperative Role of Integrating Digitalization in the Firms Finance: A Technological Perspective," *Electronics (Switzerland)*, vol. 11, no. 19. MDPI, Oct. 01, 2022. doi: 10.3390/electronics11193252.

[3] A. Razaque *et al.*, "Credit Card-Not-Present Fraud Detection and Prevention Using Big Data Analytics Algorithms," *Applied Sciences (Switzerland)*, vol. 13, no. 1, Jan. 2023, doi: 10.3390/app13010057.

[4] M. El Hajj and J. Hammoud, "Unveiling the Influence of Artificial Intelligence and Machine Learning on Financial Markets: A Comprehensive Analysis of AI Applications in Trading, Risk Management, and Financial Operations," *Journal of Risk and Financial Management*, vol. 16, no. 10, Oct. 2023, doi: 10.3390/jrfm16100434.

[5] X. Zheng, E. Gildea, S. Chai, T. Zhang, and S. Wang, "Data Science in Finance: Challenges and Opportunities," *AI*, vol. 5, no. 1, pp. 55–71, Dec. 2023, doi: 10.3390/ai5010004.

[6] M. Pavlicko, M. Durica, and J. Mazanec, "Ensemble model of the financial distress prediction in visegrad group countries," *Mathematics*, vol. 9, no. 16, Aug. 2021, doi: 10.3390/math9161886.

[7] B. Liao, Z. Huang, X. Cao, and J. Li, "Adopting Nonlinear Activated Beetle Antennae Search Algorithm for Fraud Detection of Public Trading Companies: A Computational Finance Approach," *Mathematics*, vol. 10, no. 13, Jul. 2022, doi: 10.3390/math10132160.

[8] Z. Zhao and T. Bai, "Financial Fraud Detection and Prediction in Listed Companies Using SMOTE and Machine Learning Algorithms," *Entropy*, vol. 24, no. 8, Aug. 2022, doi: 10.3390/e24081157.

[9] B. Stojanović and J. Božić, "Robust Financial Fraud Alerting System Based in the Cloud Environment," *Sensors*, vol. 22, no. 23, Dec. 2022, doi: 10.3390/s22239461.

[10] J. Li and D. Yang, "Research on Financial Fraud Detection Models Integrating Multiple Relational Graphs," *Systems*, vol. 11, no. 11, Nov. 2023, doi: 10.3390/systems11110539.

[11] X. Zheng, C. Feng, Z. Yin, J. Zhang, and H. Shen, "Research on Fraud Detection Method Based on Heterogeneous Graph Representation Learning," *Electronics (Switzerland)*, vol. 12, no. 14, Jul. 2023, doi: 10.3390/electronics12143070.

[12] M. Hasan, A. Hoque, and T. Le, "Big Data-Driven Banking Operations: Opportunities, Challenges, and Data Security Perspectives," *FinTech*, vol. 2, no. 3, pp. 484–509, Jul. 2023, doi: 10.3390/fintech2030028.

[13] M. A. Mohammed, M. Boujelben, and M. Abid, "A Novel Approach for Fraud Detection in Blockchain-Based Healthcare Networks Using Machine Learning," *Future Internet*, vol. 15, no. 8, Aug. 2023, doi: 10.3390/fi15080250.

[14] S. Brahimi and M. Elhussein, "Measuring the Effect of Fraud on Data-Quality Dimensions," *Data (Basel)*, vol. 8, no. 8, Aug. 2023, doi: 10.3390/data8080124.

[15] F. Al-Quayed, M. Humayun, and S. Tahir, "Towards a Secure Technology-Driven Architecture for Smart Health Insurance Systems: An Empirical Study," *Healthcare (Switzerland)*, vol. 11, no. 16, Aug. 2023, doi: 10.3390/healthcare11162257.

[16] E. Nabrawi and A. Alanazi, "Fraud Detection in Healthcare Insurance Claims Using Machine Learning," *Risks*, vol. 11, no. 9, Sep. 2023, doi: 10.3390/risks11090160.

[17] R. Velázquez, A. Rodríguez, A. Hernández, R. Casquete, M. J. Benito, and A. Martín, "Spice and Herb Frauds: Types, Incidence, and Detection: The State of the Art," *Foods*, vol. 12, no. 18. Multidisciplinary Digital Publishing Institute (MDPI), Sep. 01, 2023. doi: 10.3390/foods12183373.

[18] J. Chung and K. Lee, "Credit Card Fraud Detection: An Improved Strategy for High Recall Using KNN, LDA, and Linear Regression," *Sensors*, vol. 23, no. 18, Sep. 2023, doi: 10.3390/s23187788.

[19] X. Yang *et al.*, "FinChain-BERT: A High-Accuracy Automatic Fraud Detection Model Based on NLP Methods for Financial Scenarios," *Information (Switzerland)*, vol. 14, no. 9, Sep. 2023, doi: 10.3390/info14090499.

[20] C. N. Valdebenito Maturana, A. L. Sandoval Orozco, and L. J. García Villalba, "Exploration of Metrics and Datasets to Assess the Fidelity of Images Generated by Generative Adversarial Networks," *Applied Sciences (Switzerland)*, vol. 13, no. 19, Oct. 2023, doi: 10.3390/app131910637.

[21] K. R. Griffiths *et al.*, "Development of Seven New dPCR Animal Species Assays and a Reference Material to Support Quantitative Ratio Measurements of Food and Feed Products," *Foods*, vol. 12, no. 20, Oct. 2023, doi: 10.3390/foods12203839.

[22] G. Lee, Y. Yoon, and K. Lee, "Anomaly Detection Using an Ensemble of Multi-Point LSTMs," *Entropy*, vol. 25, no. 11, Nov. 2023, doi: 10.3390/e25111480.

[23] L. L. Cunha, M. A. Brito, D. F. Oliveira, and A. P. Martins, "Active Learning in the Detection of Anomalies in Cryptocurrency Transactions," *Mach Learn Knowl Extr*, vol. 5, no. 4, pp. 1717–1745, Dec. 2023, doi: 10.3390/make5040084.

[24] V. A. Cicirello, "An Analysis of an Open Source Binomial Random Variate Generation Algorithm," *Engineering Proceedings*, vol. 56, no. 1, 2023, doi: 10.3390/ASEC2023-15349.

[25] Y. Huang, W. Liu, S. Li, Y. Guo, and W. Chen, "A Novel Unsupervised Outlier Detection Algorithm Based on Mutual Information and Reduced Spectral Clustering," *Electronics (Switzerland)*, vol. 12, no. 23, Dec. 2023, doi: 10.3390/electronics12234864.

[26] S. Shuai, Z. Hu, B. Zhang, H. Bin Liaqat, and X. Kong, "Decentralized Federated Learning-Enabled Relation Aggregation for Anomaly Detection," *Information (Switzerland)*, vol. 14, no. 12, Dec. 2023, doi: 10.3390/info14120647.

[27] A. Faccia, J. McDonald, and B. George, "NLP Sentiment Analysis and Accounting Transparency: A New Era of Financial Record Keeping," *Computers*, vol. 13, no. 1, p. 5, Dec. 2023, doi: 10.3390/computers13010005.

[28] A. R. Khalid, N. Owoh, O. Uthmani, M. Ashawa, J. Osamor, and J. Adejoh, "Enhancing Credit Card Fraud Detection: An Ensemble Machine Learning Approach," *Big Data and Cognitive Computing*, vol. 8, no. 1, Jan. 2024, doi: 10.3390/bdcc8010006.

[29] A. M. Aburbeian and M. Fernández-Veiga, "Secure Internet Financial Transactions: A Framework Integrating Multi-Factor Authentication and Machine Learning," *AI*, vol. 5, no. 1, pp. 177–194, Jan. 2024, doi: 10.3390/ai5010010.

[30] A. Janavičiūtė, A. Liutkevičius, G. Dabužinskas, and N. Morkevičius, "Experimental Evaluation of Possible Feature Combinations for the Detection of Fraudulent Online Shops," *Applied Sciences*, vol. 14, no. 2, p. 919, Jan. 2024, doi: 10.3390/app14020919.

[31] K. Gu, "Deep Learning Techniques in Financial Fraud Detection," in *ACM International Conference Proceeding Series*, Association for Computing Machinery, Sep. 2022, pp. 282–286. doi: 10.1145/3558819.3565093.