AI-Driven Loan Approval using PSO-Enhanced ANN & GBM

Akash Saraswat[1], Anushika Gupta[2], Divya Pandey[3] and Ishitta[4]

1B.Tech. III Year Centre for Artificial Intelligence, MITS, Gwalior, India

2.B.Tech. III Year Centre for Artificial Intelligence, MITS, Gwalior, India

3.B.Tech. III Year Centre for Artificial Intelligence, MITS, Gwalior, India

4.B.Tech. III Year Centre for Artificial Intelligence, MITS, Gwalior, India

Corresponding Author: divyapandey1113@gmail.com

AIM:

The primary motive of this study is to develop and evaluate Artificial Neural Networks (ANN's) and Gradient Boosting Machines (GBMs) that are PSO-optimized in order to analyse loan approval outcomes from borrowers' according there financial status. The aim of this Research is to explore the impact of PSO on improving the accuracy and efficiency of ANN and GBM models, and hence risk management as well as loan payback rates.

OBJECTIVE:

The various purposes that this paper caters to are:

Feature Enhancement & Data Analysis: Enhancing features at the data collection and extraction level to improve model accuracy and efficiency while analyzing and visualizing dataset features to observe their relation with loan repayment ability and identify key factors influencing loan approval.

Model Implementation & PSO Optimization: Implementing Artificial Neural Networks (ANN) and Gradient Boosting Machines (GBM) on loan data to forecast loan approval results while utilizing Particle Swarm Optimization (PSO) to optimize the hyperparameters of GBM and ANN for improved prediction capability.

Performance Comparison: Comparing the performance of ANN and GBM models with and without PSO based on performance metrics such as accuracy, precision, recall, and F1-score.

Future Scope: Identifying potential improvements in machine learning algorithms for financial applications and addressing future loan prediction challenges.

ABSTRACT:

In today's era, precise loan approval forecasting has become crucial for institutions and banks. This process benefits risk management, margins, and customers. Organizations need to identify areas of improvement, meet new people, and serve existing ones in order to expand.Machine learning helps greatly in discovering insights and patterns, assisting institutions in making sounder decisions.

This paper investigates the application of Artificial Neural Networks (ANNs) and Gradient Boosting Machines (GBMs) — optimized using Particle Swarm Optimization (PSO) for loan approval prediction.ANNs are effective in handling non-linear relationships between data, whereas GBMs are suitable with structured data and identifying subtle features interactions. PSO is used to optimize the hyperparameters of the two models, enhancing their accuracy and robustness by steering them to best solutions. Both models perform well in prediction with a dataset consisting of 396,030 records and 27 attributes, with PSO offering extra contributions to enhancements.The research points out the promise of combining optimization methods with machine learning to transform loan approval processes and proposes an efficient framework for new financial technology innovations.The objectives of this study are to suggest new features at the data extraction and collection level to improve the efficiency and accuracy of the models, analyzing and mapping various features of the dataset and their relation to loan payment ability, using ANN and GBM machine learning models on historical loan data, and testing the performance of the models on metrics such as accuracy, precision, recall, and F1-score. In addition, this study explores future innovations and challenges in machine learning models. Machine learning models, such as the ones discussed in this paper, can greatly assist in realizing these objectives through the detection of valuable insights and trends.

Keywords: Loan Approval Prediction, Artificial Neural Networks (ANNs) and Gradient Boosting Machines (GBM) , Particle Swarm Optimization (PSO)

1. INTRODUCTION

In this financial market, loan approval forecasting is highly crucial to banks because it directly influences risk management, profitability margins, and customer satisfaction. Proper loan forecasting minimizes defaults, improves utilization of financial assets, and leads to streamlined decision-making processes. Loan approvals used to depend on manual appraisals — a process that is not only time-consuming but also susceptible to human inconsistency. As the number and complexity of loan applications increase, financial institutions need more sophisticated methods to improve the accuracy and efficiency of their approval systems.Machine learning has proven to be an invaluable tool in this case, offering computer-based processes that examine and identify patterns within information,improve predictions, and optimize decision-making strategies.Among the several ML methods, Artificial Neural Networks (ANNs) [1] and Gradient Boosting Machines (GBMs) have specifically performed well in the prediction of whether a loan is sanctioned. ANNs, patterned after the functioning of the human brain, are optimal in identifying intricate, non-linear interactions among variables from large datasets, and thus suitable for identifying subtle financial trends. On the other hand, GBMs use ensemble learning by aggregating many decision trees, making predictions at each iteration, and improving performance, particularly with structured data such as loan applications.Their performance, nonetheless, is much dependent on hyperparameter tuning a technique that can become tiresome and ineffective when it is carried out manually. To compensate for this, Particle Swarm Optimization (PSO) is utilized as a metaheuristic optimization algorithm that ensures automatic selection of the optimal hyperparameters to be used with ANN and GBM models. PSO has been inspired by the birds' social behaviour flocking or fish schooling, PSO optimizes models through the direction of particles (potential solutions) to the optimal performance by iterative learning.[2] Integrating PSO into these models can result in increased accuracy, faster convergence, and enhanced efficiency, hence greater reliability to predict loan approvals.It is on this premise that the predictive performance of ANN and GBM models with and without PSO is evaluated in this research to establish the impact of optimization on predictive performance.The primary objectives of this research are to enhance model efficiency and accuracy by introducing new features at the data collection and extraction level, exploring the relationship between loan repayment ability and dataset features, and utilizing PSO-optimized ANN and GBM models on historical loan data.Performance is evaluated by testing on metrics such as accuracy, precision, recall, and F1-score to identify the optimal approach. In addition, this paper elaborates on upcoming advances and challenges in machine learning for financial decision-making, providing insightful information on maximizing loan approval prediction.

2. LITERATURE OVERVIEW

2.1 Machine Learning in Financial Decision-Making

Machine learning has significantly transformed financial decision-making by accelerating processes and improving precision, particularly loan approval prediction. Traditional models such as Decision Trees, Naive Bayes, Logistic Regression, and Support Vector Machines (SVM) have been used everywhere for this. Decision Trees [3]simplify decision-making by defining outcomes through a tree-like structure, and Naive Bayes applies probability theory under feature independence to classify data. Logistic Regression models binary outcomes by predicting probabilities using a logistic function, while SVM seeks the best hyperplane that discriminates between data points based on categories. Comparisons of performance among these models yielded diverse results—Decision Trees registered 93.648%, Random Forests 83.388%, and Logistic Regression 80.945%. Although less accurate, Random Forests performed better in generalization across cross-validation, thus being more credible when working with unseen data.[4]

2.2 Artificial Neural Networks (ANNs) and Gradient Boosting Machines (GBMs)

Gradient Boosting Machines (GBMs) and Artificial Neural Networks (ANNs) have become increasingly important in the last few years for loan approval predictions. ANNs are capable of extracting non-linear relationships and understanding complex patterns of data and, therefore, they are ideal for big data. It has been

proven through studies that ANNs are more accurate and robust than traditional models due to the capability of ANNs to learn the complex features present in the data.[5] In contrast, GBMs exploit ensemble learning by combining multiple decision trees and iteratively refining predictions to reduce errors. Through closer monitoring of hard cases, GBMs have achieved spectacular gains in prediction accuracy, often beating single-tree models.[6]

2.3 Particle Swarm Optimization (PSO) in Model Optimization

Although ANNs and GBMs are powerful predictors, hyperparameter tuning by hand is cumbersome and time-consuming. To resolve this, Particle Swarm Optimization has become prominent as a metaheuristic optimization method that makes the process automatic. Using the birds' flocking or schools' behavior for inspiration, PSO models a swarm of particles, where each particle is a possible solution. These particles traverse the solution space by moving according to two important considerations: Personal Best (Pbest) , or the best known solution for each particle to date, and Global Best (Gbest) , or the best solution found by the whole swarm. Through repeated learning from these optimal solutions, PSO converges toward ideal hyperparameter values, improving model performance and convergence rate. Combining PSO with ANN and GBM has the potential to drastically enhance predictive accuracy, loan approval predictions becoming more accurate and streamlined. The paper seeks to investigate the effect of PSO on both ANN and GBM in assessing their ability to enhance loan approval predictions and comparing performance both with and without PSO.[7]

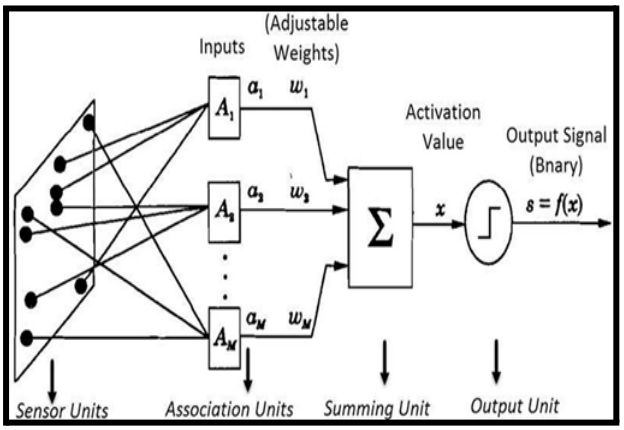

Fig.01-ANN Model

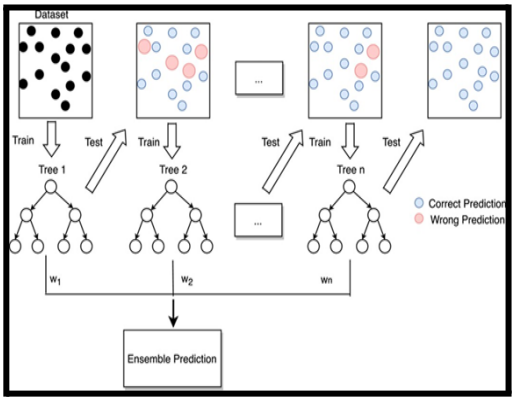

Fig.02- GBM Model

3. METHODOLOGY

This research will build Artificial Neural Networks (ANNs) and Gradient Boosting Loan approval prediction machines (GBMs) , with Particle Swarm Optimization (PSO) used for hyperparameter optimization. The approach includes a number of phases: Data Preprocessing, Model Architecture, PSO Optimization, and Performance Evaluation.

3.1 Features Provided

The data set contains some of the individual and financial characteristics of loan applicants that are significant in quantifying creditworthiness and loan approval. The significant financial characteristics include Loan Amount, the amount requested in dollars, and Term, the term of repayment of 36 or 60 months. Interest Rate is allocated on the basis of creditworthiness of the borrower, while Installments are the monthly payment. Creditworthiness is also assessed by Grade and Sub_Grade, from A to G and A1 to G5, respectively. Employment information, i.e., Employment Length, shows years for which the borrower has been working, while Home Ownership status, Annual Income, and Verification Status show further financial information about stability of income. The other required characteristics include the Debt-to-Income Ratio (DTI) , Open Accounts, Total Accounts, Revolving Balance, Revolving Line Utilization Rate, and Public Records, all of which help determine the financial history and risk profile of an applicant. Furthermore, characteristics like Purpose, Issue Date, Loan Status, and Application Type (individual or joint) speak more to the type of loan application.

3.2 Proposed Features for Future Information Extraction

In addition to further enhancing model accuracy, additional features are proposed to further assess an applicant's financial position and risk profile. Total Income Over Past Years (5, 3, and 1 year) is included to reflect the stability and direction of the growth in income and obtain valuable information regarding the consistency of earnings. Net Worth Over Previous Years is a quantitative measure of overall fiscal well-being, indicating an applicant's total accumulation of assets and liabilities over a period of time. Health and Insurance Policies are included to establish risk management methods, as adequate coverage can alleviate fiscal uncertainty. Additionally, Personal and Family Details such as marital status and dependents provide an insight into the applicant's financial obligations, helping refine risk assessment and loan authorization processes.

3.3 Data Preprocessing

To support high-quality data and increase model efficiency, various preprocessing methods are used. Numerical features' missing values are replaced using the mean or median to manage null values, and categorical features are replaced with the mode. One-Hot Encoding is used to transform categorical variables into an appropriate form for machine learning models. For numeric features such as Loan Amount and Debt-to-Income Ratio (DTI) , Standard Scaler is used to scale the data for improved model performance. Non-relevant features such as Address and Employee Title are also removed for increased computational speed.

3.4 Model Architecture

3.4.1 Artificial Neural Network (ANN)

Neural network structure is designed in such a way that it efficiently works on loan approval predictions. It starts with an **Input Layer** which accepts preprocessed features. It continues with **four Hidden Layers** each having 78, 39, 19, and 10 neurons where the ReLU activation function is employed for detecting intricate patterns in the data. To avoid overfitting, **Dropout Layers** with drop values ranging from 0.2 to 0.5 are used. The **Output Layer** employs a Sigmoid activation function because it is better suited to binary classification. The model is trained with **Binary Cross-Entropy** loss function to improve probability-based estimates and the **Adam Optimizer** to aid in structured and adaptive learning.[8]

3.4.2 Gradient Boosting Machine (GBM)

The Gradient Boosting Machine (GBM) model is trained for both accurate- and efficient loan approval prediction. It utilizes **Decision Trees** as the base learner, which take advantage of their capability to learn complex patterns in the data. The **learning rate** parameter is at 0.01, which means gradual learning and not radical updates that can cause instability. To regulate model complexity and prevent overfitting, the **maximum depth** of every tree is capped at 6. Moreover, an **80% subsample ratio** is used in every iteration so that the model learns from various segments of the data and remains diverse. Lastly, the model is trained using **1000 estimators**, progressively enhancing accuracy by minimizing errors across successive trees.[9]

3.5 Particle Swarm Optimization (PSO)

Particle Swarm Optimization (PSO) is used to optimize ANN and GBM hyperparameters and improve their predictability. Initialization is used initially, during which 20 particles, corresponding to different combinations of hyperparameters, are produced. The particles are then graded depending on their **Fitness** against accuracy and F1-score, which act as the measures to be optimized. By the **Update Mechanism**, every particle updates its hyperparameters by learning from its **Personal Best (Pbest)** and the **Global Best (Gbest)** solution found by the swarm. The optimization process stops when the **Stopping Criteria** are satisfied, either after 100 iterations or when a convergence threshold is reached where improvements are less than 0.01% for 10 consecutive iterations.[10]

3.6 Performance Evaluation

To ensure the accuracy and reliability of predictions regarding loan approval, the model's performance is evaluated using a variety of performance measures. While **Precision** considers the percentage of correct approved loans, Accuracy measures the overall accuracy of the model's predictions. To ensure that deserving

applicants are not missed,Recall measures the accuracy of the model in detecting the true loan approvals. The F1-Score yields a balanced measurement by weighing recall and precision so as to combat class imbalances. Further, the model is fortified through 5-Fold Cross-Validation , decreasing bias and confirming that the model performs uniformly over different subsets of data.[11]

4. IMPLEMENTATION:

4.1 Remove irrelevant or high-cardinality features like 'Employee Title' and 'Title'. Discard non-predictive features like 'Address'.

```
[56]: df.columns

[56]: Index(['loan_amnt', 'term', 'int_rate', 'installment', 'sub_grade',
       'home_ownership', 'annual_inc', 'verification_status', 'issue_d',
       'loan_status', 'purpose', 'dti', 'earliest_cr_line', 'open_acc',
       'pub_rec', 'revol_bal', 'revol_util', 'total_acc',
       'initial_list_status', 'application_type', 'mort_acc',
       'pub_rec_bankruptcies', 'address'],
      dtype='object')
```

4.2 The image shows the count of missing values in a DataFrame using df.isnull( ) .sum( ) . Some columns, like emp_title, emp_length, title, revol_util, mort_acc, and pub_rec_bankruptcies, have missing values, while others have none. This helps identify data cleaning or imputation needs.

```
[34]: df.isnull().sum()

[34]: loan_amnt                0
      term                     0
      int_rate                 0
      installment              0
      grade                    0
      sub_grade                0
      emp_title            22927
      emp_length           18301
      home_ownership           0
      annual_inc               0
      verification_status      0
      issue_d                  0
      loan_status              0
      purpose                  0
      title                 1756
      dti                      0
      earliest_cr_line         0
      open_acc                 0
      pub_rec                  0
      revol_bal                0
      revol_util             276
      total_acc                0
      initial_list_status      0
      application_type         0
      mort_acc             37795
      pub_rec_bankruptcies   535
      address                  0
      dtype: int64
```

4.3 The image shows the output of df.columns, listing all column names in a DataFrame. It includes financial variables (e.g., loan_amnt, int_rate) , categorical fields (e.g., verification_status, application_type) , purpose-related features, and encoded values. The dataset contains object-type columns, indicating categorical string data.

```
[84]: df.columns

[84]: Index(['loan_amnt', 'term', 'int_rate', 'installment', 'annual_inc', 'dti',
       'open_acc', 'pub_rec', 'revol_bal', 'revol_util', 'total_acc',
       'mort_acc', 'pub_rec_bankruptcies', 'A2', 'A3', 'A4', 'A5', 'B1', 'B2',
       'B3', 'B4', 'B5', 'C1', 'C2', 'C3', 'C4', 'C5', 'D1', 'D2', 'D3', 'D4',
       'D5', 'E1', 'E2', 'E3', 'E4', 'E5', 'F1', 'F2', 'F3', 'F4', 'F5', 'G1',
       'G2', 'G3', 'G4', 'G5', 'verification_status_Source Verified',
       'verification_status_Verified', 'application_type_INDIVIDUAL',
       'application_type_JOINT', 'initial_list_status_w',
       'purpose_credit_card', 'purpose_debt_consolidation',
       'purpose_educational', 'purpose_home_improvement', 'purpose_house',
       'purpose_major_purchase', 'purpose_medical', 'purpose_moving',
       'purpose_other', 'purpose_renewable_energy', 'purpose_small_business',
       'purpose_vacation', 'purpose_wedding', 'OTHER', 'OWN', 'RENT',
       'cr_year', '05113', '11650', '22690', '29597', '30723', '48052',
       '70466', '86630', '93700', 'Fully Paid'],
      dtype='object')
```

4.4 Smote technique is used to balance the imbalance data

```
[103]: from collections import Counter

       print("Class distribution in y_train:", Counter(y_train))
       print("Class distribution in y_test:", Counter(y_test))

       Class distribution in y_train: Counter({True: 212937, False: 51859})
       Class distribution in y_test: Counter({True: 104759, False: 25664})

[104]: from imblearn.over_sampling import SMOTE

[105]: smote = SMOTE(sampling_strategy='auto', random_state=42)

[106]: X_train, y_train = smote.fit_resample(X_train, y_train)

[107]: from collections import Counter
       print("Class distribution after SMOTE:", Counter(y_train))

       Class distribution after SMOTE: Counter({True: 212937, False: 212937})
```

4.5 The code defines a function build_ann(params) that constructs an artificial neural network (ANN) using Keras. It extracts hyperparameters (learning_rate, neurons, layers, dropout) from params, ensuring neurons and layers are integers. The ANN is built using the Sequential model, starting with an input layer followed by multiple hidden layers, each with a specified number of neurons, ReLU activation, and dropout for regularization. The final layer has a single neuron with a sigmoid activation for binary classification. The model is compiled using the Adam optimizer with the specified learning rate, binary cross-entropy loss, and accuracy as the evaluation metric. The function returns the constructed ANN model.

```
def build_ann(params):
    learning_rate, neurons, layers, dropout = params
    neurons = int(neurons)
    layers = int(layers)

    model = Sequential()
    model.add(Dense(neurons, activation='relu', input_dim=X_train.shape[1]))
    model.add(Dropout(dropout))
    for _ in range(layers - 1):
        model.add(Dense(neurons, activation='relu'))
        model.add(Dropout(dropout))
    model.add(Dense(1, activation='sigmoid'))
    model.compile(optimizer=Adam(learning_rate=learning_rate),
                  loss='binary_crossentropy',
                  metrics=['accuracy'])
    return model
```

4.6 The function ann_objective(params) evaluates an artificial neural network (ANN) based on given hyperparameters. It starts by recording the time, extracts the parameters (learning_rate, neurons, layers, dropout) , and prints them. The function then builds the ANN using build_ann(params) , trains it on X_train and y_train for 10 epochs with a batch size of 32 and a validation split of 10%. After training, it evaluates the model on X_test and y_test, obtaining the loss and accuracy. The fitness score is calculated as 1 - accuracy (to be minimized in optimization) , stored in fitness_history, and the execution time is printed. Finally, the function returns the fitness score.

```
import time

fitness_history = []

def ann_objective(params):
    start_time = time.time()
    learning_rate, neurons, layers, dropout = params
    print(f"\nEvaluating ANN with params: LR={learning_rate:.5f}, Neurons={int(neurons)}, Layers={int(layers)}, Dropout={dropout:.2f}")
    model = build_ann(params)
    model.fit(X_train, y_train, epochs=10, batch_size=32, verbose=0, validation_split=0.1)
    loss, acc = model.evaluate(X_test, y_test, verbose=0)
    fitness_score = 1 - acc
    fitness_history.append(fitness_score)
    end_time = time.time()
    print(f"→ Accuracy: {acc:.4f}, Loss: {loss:.4f}, Fitness Score: {fitness_score:.4f}, Time Taken: {end_time - start_time:.2f} sec")
    return fitness_score
```

4.7 Used for tuning the hyperparameters of an XGBoost classifier using Particle Swarm Optimization (PSO). The function `gbm_objective(params)` evaluates different hyperparameter combinations, including `learning_rate`, `max_depth`, `n_estimators`, `subsample`, and `colsample`, printing the parameter values being tested. It initializes an `XGBClassifier` with these parameters, trains it on `X_train` and `y_train`, and evaluates its accuracy on `X_test`. The fitness score is computed as `1 - accuracy` (to be minimized), and execution time is logged. The fitness score is stored in `fitness_history_gbm` and returned for optimization.

GBM with PSO

[113]:
```
fitness_history_gbm = []

def gbm_objective(params):
    start_time = time.time()
    learning_rate, max_depth, n_estimators, subsample, colsample = params
    print("Evaluating: LR={:.5f}, max_depth={}, n_estimators={}, subsample={:.2f}, colsample={:.2f}".format(
        learning_rate, int(max_depth), int(n_estimators), subsample, colsample))

    model = xgb.XGBClassifier(
        learning_rate=learning_rate,
        max_depth=int(max_depth),
        n_estimators=int(n_estimators),
        subsample=subsample,
        colsample_bytree=colsample,
        eval_metric="logloss",
        n_jobs=-1
    )

    model.fit(X_train, y_train,
              eval_set=[(X_test, y_test)],
              verbose=False)

    acc = model.score(X_test, y_test)
    fitness_score = 1 - acc
    end_time = time.time()

    print("Result: Accuracy = {:.4f}, Fitness = {:.4f}\n".format(acc, fitness_score))
    print("Time Taken: {:.2f} sec".format(end_time - start_time))

    fitness_history_gbm.append(fitness_score)
    return fitness_score
```

4.8 This code initializes and trains an **XGBoost classifier** for binary classification. The model is created using XGBClassifier with random_state=101 for reproducibility, use_label_encoder=False (which is now deprecated and unnecessary) , and eval_metric='logloss' to use **logarithmic loss** for evaluation. When calling fit(X_train, y_train) , a warning appears indicating that use_label_encoder is ignored since it is no longer needed in recent versions of XGBoost. Despite the warning, the model successfully initializes with default parameters and starts training on the given dataset.

```
[138]: model_XGB = XGBClassifier(random_state=101, use_label_encoder=False, eval_metric='logloss')

[139]: model_XGB.fit(X_train, y_train)

/opt/anaconda3/lib/python3.11/site-packages/xgboost/core.py:158: UserWarning: [19:32:03] WARNING: /U
ner.cc:740:
Parameters: { "use_label_encoder" } are not used.

  warnings.warn(smsg, UserWarning)

[139]: XGBClassifier

XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, device=None, early_stopping_rounds=None,
              enable_categorical=False, eval_metric='logloss',
              feature_types=None, gamma=None, grow_policy=None,
              importance_type=None, interaction_constraints=None,
              learning_rate=None, max_bin=None, max_cat_threshold=None,
              max_cat_to_onehot=None, max_delta_step=None, max_depth=None,
              max_leaves=None, min_child_weight=None, missing=nan,
              monotone_constraints=None, multi_strategy=None, n_estimators=None,
              n_jobs=None, num_parallel_tree=None, random_state=101, ...)
```

4.9 This PSO implementation trains and evaluates two models: an Artificial Neural Network (ANN) and a Gradient Boosting Machine (GBM) using XGBoost. The ANN is built using hyperparameters optimized by PSO, trained on X_train, and predicts on X_test, converting predictions to binary form (>0.5 → 1) before calculating accuracy. Similarly, the GBM model is configured with PSO-optimized parameters, trained on X_train, and tested on X_test, with accuracy printed for comparison. This approach assesses the effectiveness of PSO in optimizing hyperparameters for both models.

PSO implementation

```
final_ann = build_ann(best_params)
final_ann.fit(X_train, y_train, epochs=100, batch_size=32, verbose=1)
y_pred_ann = (final_ann.predict(X_test) > 0.5).astype(int)
print("ANN Accuracy:", accuracy_score(y_test, y_pred_ann))
```

```
final_gbm = xgb.XGBClassifier(learning_rate=best_params_gbm[0], max_depth=int(best_params_gbm[1]),
                              n_estimators=int(best_params_gbm[2]), subsample=best_params_gbm[3],
                              colsample_bytree=best_params_gbm[4])
final_gbm.fit(X_train, y_train)
y_pred_gbm = final_gbm.predict(X_test)
print("GBM Accuracy:", accuracy_score(y_test, y_pred_gbm))
```

5. RESULT

5.1

In ANN With pso we are highly improving our model as we are using only 4 layers and 36 neurons and accuracy is incremented by 4%n from 84 to 88**.** In GBM with pso there is only slight improvement and also the graph converges initially only.

Model	Accuracy	Precision	Recall	F1 Score
ANN without PSO	84	75.5	78.5	77
ANN with PSO	89	77.5	80.5	78.5
GBM without PSO	89	89	73.5	78
GBM with PSO	89	89.5	74	78.5

5.2 Exploratory Data Analysis (EDA)

Fig.03 Shows the distribution of loans across different verification statuses, split by loan status (e.g., Fully Paid vs. Charged Off) .

Fig.04 indicates the count of loans for various purposes, categorized by loan status.

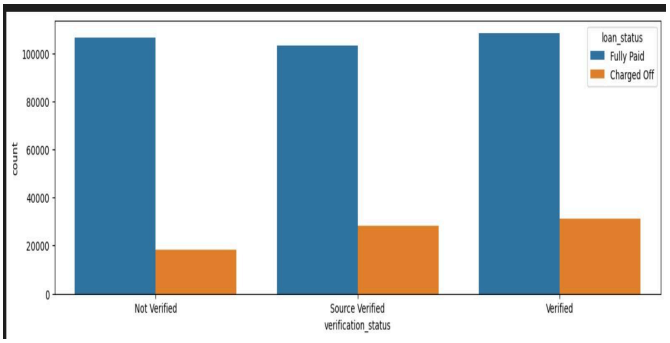

Fig.03 distribution of loans

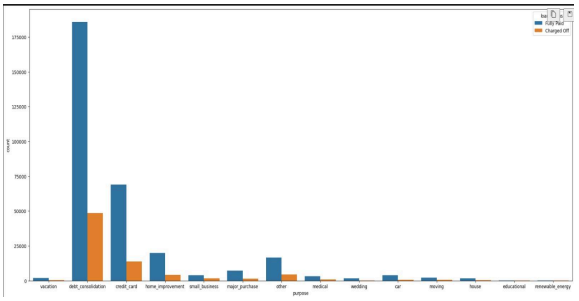

Fig.04 count of loans for various purposes

Fig.05 indicates the relationship between home ownership categories and loan repayment status, showing that most loans are fully paid, especially for those with mortgages, while a smaller proportion is charged off.

Fig.06 indicates plot compares loan terms (36 months vs. 60 months) across loan statuses, showing more loans with 36-month terms and a higher proportion of "Charged Off" loans in the 60-month term.

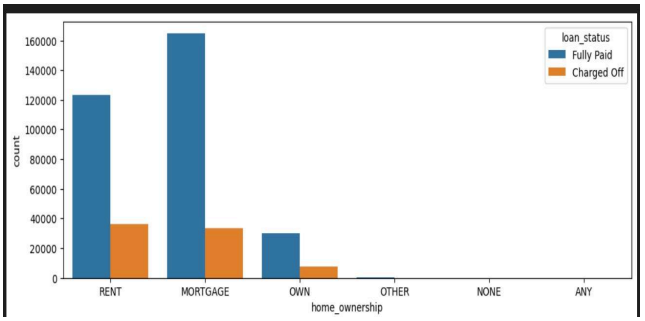

Fig.05 Home ownership and loan repayment relationship

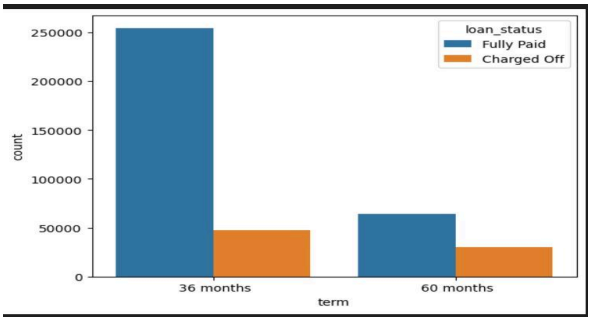

Fig.06 loan terms across loan statuses

Fig.07 indicates most loans are in mid-risk sub-grades (B2, B3) , with fewer loans in the highest-risk categories

Fig.08 indicates the image is a heatmap representing the correlation matrix of financial variables, where colors indicate the strength and direction of relationships between them.

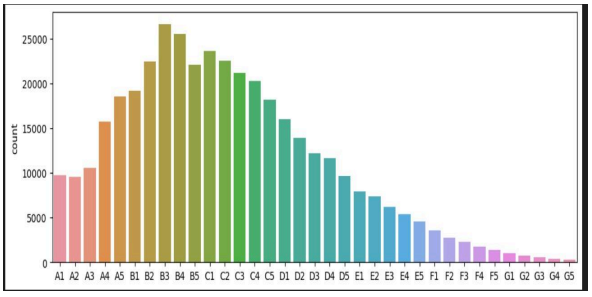

Fig.07 loan histogram

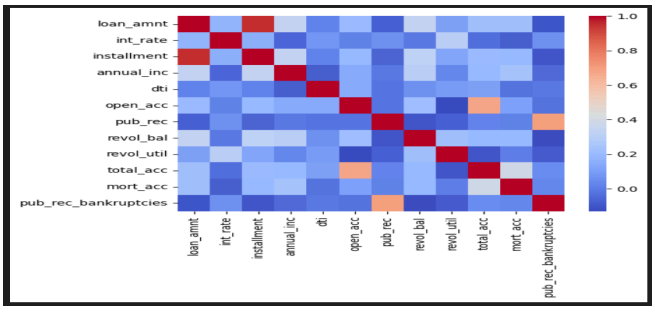

Fig.08 heatmap representing the correlation matrix

6.CONCLUSION:

We compared the performance of Artificial Neural Networks (ANNs)and Gradient Boosting Machines (GBMs) for loan approval prediction in this study. The comparison clearly illustrates how the two models both enhance the prediction precision and speed immensely compared to conventional models. The GBM slightly fared better as it has an easy time managing intricate, non-linear interactions as well as having lesser tendencies to overfit. The study confirms again that machine learning techniques can yield considerable banking sector improvements, automating the loan sanctioning process and reducing defaults. This not only benefits the financial institutions by saving losses but also raises the level of customer satisfaction by making faster decisions.

6.1 Key Findings

Comparison of ANNs and GBMs brings out the advantages and disadvantages of both methods. ANNs are very good at picking up complex patterns in large datasets, hence being very powerful in predicting loan approvals if enough data is provided. However, GBMs have greater interpretability and are often less computationally intensive, and hence more suited for real-world applications where speedy decision-making is important. The results indicate that GBMs have a slight edge over ANNs, especially when dealing with imbalanced datasets, where most of the loans are approved or rejected and not so many borderline ones.

6.2 Future Works

Subsequent research in this field should integrate additional attributes into the prediction models to enhance accuracy more. Such attributes should be more comprehensive financial metrics, behavioral data, and even social media activity, which can provide more insight into the financial health of a borrower. Further, investigating the capabilities of hybrid models that integrate ANNs and GBMs can produce better performance by taking advantage of strengths of both methods. Examining the interpretability of such models is just as important to ensure that financial institutions are able to rely and use these sophisticated systems properly in their decision-making.

7. Challenges and Future Directions

While a lot has been done in loan approval prediction with machine learning, there are still some challenges. The biggest challenge is data quality. Inconsistent or incomplete data can lead to biased models and erroneous predictions. Having access to detailed data with multiple financial and personal variables is important to developing more reliable models. Feature selection is also a top priority area. Selecting the most appropriate characteristics to predict loan approvals are imprecise, lacking precise values especially when dealing with vast datasets. Feature selection using even more sophisticated techniques such as genetic algorithms or reinforcement learning could even improve model accuracy. A second important challenge is model explainability. Powerful models like ANNs and GBMs, though incredibly powerful, are often "black boxes," and it is not clear what goes into their decision-making process. Development of techniques to enhance model transparency will be crucial to generate trust between regulators and institutions. Future directions should also cover the construction of hybrid models in combining different machine learning techniques, such as ANNs, GBMs, and Support Vector Machines (SVMs) . The hybrid techniques can detect a larger range of patterns in the data, leading to more accurate and stable predictions. As the financial sector continues to march towards digitalization, such machine learning advances will be leading the way in making loan approval procedures more efficient, ultimately to the benefit of lenders and borrowers. Also we should try other different optimisation techniques on GBM as graph converges on initial phase only.

8.REFERENCES

[1]Y. Yoon and L. L. Peterson, “Artificial neural networks: an emerging new technique,” Jan. 1990, doi: https://doi.org/10.1145/97709.97738.

[2] H. M. ELRAGAL, “IMPROVING NEURAL NETWORKS PREDICTION ACCURACY USING PARTICLE SWARM OPTIMIZATION COMBINER,” *International Journal of Neural Systems*, vol. 19, no. 05, pp. 387–393, Oct. 2009, doi: https://doi.org/10.1142/s0129065709002099.

[3]T. Pranckevičius and V. Marcinkevičius, “Comparison of Naive Bayes, Random Forest, Decision Tree, Support Vector Machines, and Logistic Regression Classifiers for Text Reviews Classification,” *Baltic Journal of Modern Computing*, vol. 5, no. 2, 2017, doi: https://doi.org/10.22364/bjmc.2017.5.2.05.

[4] A. Gupta, V. Pant, S. Kumar, and P. K. Bansal, “Bank Loan Prediction System using Machine Learning,” *2020 9th International Conference System Modeling and Advancement in Research Trends (SMART)* , Dec. 2020, doi: https://doi.org/10.1109/smart50582.2020.9336801

[5]Mr Rushikesh and Limbraj Kashimpure, “Artificial neural networks (ANNs, also shortened to neural networks (NNs) or neural nets),” *International Research Journal of Modernization in Engineering Technology and Science*, Jul. 2023, doi: https://doi.org/10.56726/irjmets43049.

[6] A. Natekin and A. Knoll, “Gradient boosting machines, a tutorial,” *Frontiers in Neurorobotics*, vol. 7, no. 21, 2013, doi: https://doi.org/10.3389/fnbot.2013.00021.

[7]Keisuke Kameyama, “Particle Swarm Optimization-A Survey,” *IEICE Transactions on Information and Systems*, vol. E92-D, no. 7, pp. 1354–1361, Jan. 2009, doi: https://doi.org/10.1587/transinf.e92.d.1354.

[8] Introduction to Artificial Neural Network, by A.D.Dongare, R.R.Kharde, Amit D.Kachare, International Journal of Engineering and Innovative Technology (IJEIT) Volume 2, Issue 1, July 2012.

[9] Muhammad, I., Dahlia, R., Muhammad Ifan Rifani Ihsan, Lisnawanty, & Rabiatus Sa’adah. (2024) . Performance Analysis of Ensemble Learning and Feature Selection Methods in Loan Approval Prediction at Banks. Journal of Artificial Intelligence and Engineering Applications (JAIEA) , 3(2) , 557–564. https://doi.org/10.59934/jaiea.v3i2.426

[10] Ovat Friday Aje and Anyandi Adie Josephat, “The particle swarm optimization (PSO) algorithm application – A review,” *Global Journal of Engineering and Technology Advances*, vol. 3, no. 3, pp. 001–006, Jun. 2020, doi: https://doi.org/10.30574/gjeta.2020.3.3.0033.

[11] A. Lavie, Kenji Sagae, and S. Jayaraman, “The Significance of Recall in Automatic Metrics for MT Evaluation,” *Lecture notes in computer science*, pp. 134–143, Jan. 2004, doi: https://doi.org/10.1007/978-3-540-30194-3_16.