Loan Approval Prediction

Kahf Hussain
Drexel University
Philadelphia, PA, USA
knh66@drexel.edu

ABSTRACT

The prediction of bank loan approval is essential to assess risk in financial decision making. This study explores and compares various machine learning models like Logistic Regression, SVMs, Ensemble Methods, and ANN on their predictive performance. Additionally, feature importance is analyzed to identify key contributors to loan approvals. Preprocessing steps included feature normalization, encoding, and correlation analysis to ensure data readiness. ANN achieved the highest accuracy, highlighting its ability to capture complex patterns. Logistic Regression and Linear SVM provided robust baselines, while Ensemble models balanced precision and recall. Results revealed income, debt ratio, and credit history as top predictive features. These findings provide a comparative analysis of models for predicting loan approvals and ways of improving machine learning applications in finance.

ACM Reference Format:
Kahf Hussain. 2024. Loan Approval Prediction. In . ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 BACKGROUND AND RELATED WORK

Bank loan approval prediction has been extensively studied using various machine learning methodologies, including ensemble techniques, binary classification models, and deep learning frameworks. Among these, Neural Networks consistently demonstrated superior performance due to their ability to effectively capture non-linear relationships in complex datasets. In our research, we explored the works of several authors who used diverse techniques on various datasets to address this problem. We have based our research by understanding the limitations and performances of various previous models.

The first paper we looked at was - Kumar and Singh [3]. They proposed an ensemble machine learning system that integrates Random Forest, Gradient Boosting, and Logistic Regression to enhance the accuracy of bank loan approval predictions. Their method used hard voting to combine model outputs, achieving a remarkable prediction accuracy of 91.8% on their dataset. An intriguing aspect of their study was that the Gradient-Boosting model outperformed Logistic Regression in isolating key decision-making factors, particularly those related to credit score and annual/monthly income. However, converging to suboptimal solution did to lead to lower performance of ensemble models. .

The second source, Mahgoub [4], explored a hybrid approach that combined binary classification and deep learning for predicting loan approvals. The proposed model integrated a Convolutional Neural Network (CNN) with a Multi-Layer Perceptron (MLP), achieving an impressive accuracy of 93.2%. This study showed that deep learning models significantly outperformed traditional methods in capturing non-linear relationships within the data, particularly when dealing with unstructured customer information. Notably, binary classification using logistic regression alone achieved an accuracy of 85%. Based on this finding, we adopted logistic regression as our foundational model to establish benchmark metrics for advanced models.

Our inspiration for applying Support Vector Machines (SVM) to the Loan Approval Prediction problem stemmed from the work of Yildiz [7]. The study demonstrated that fine-tuning hyperparameters such as kernel type and regularization parameter enabled the SVM model to effectively handle imbalanced datasets, achieving an accuracy of 87.5%. However, the author noted that SVM's performance declined significantly when applied to datasets with highly correlated features, emphasizing the importance of feature selection. Based on these findings, we incorporated strategies to remove highly correlated variables and applied Principal Component Analysis (PCA) to reduce dimensionality and optimize feature representation for polynomial kernels.

Finally, we referred to Alhamid [1]. The concept of hard and soft voting in ensemble methods discussed in the paper offers critical insights into improving model performance. Hard voting selects the class predicted by the majority of models, whereas soft voting uses weighted probabilities for final predictions. Alhamid emphasized that while hard voting is straightforward, soft voting is better suited for cases where individual models have varying levels of confidence in their predictions. We wanted to use this approach to combine models like SVM and Logistic Regression, as it balances the predictive strengths of each.

2 DATASET DESCRIPTION

This section provides an overview of the dataset used for developing the machine learning models. It includes details about the source of the dataset, its dimensionality, the target variable, and the features categorized by kind and type, as well as the dataset priors.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

2.1 Dataset Source

The dataset used in this study is publicly available and sourced from **Kaggle** [8]. It is a synthetic dataset comprising of records related to bank loan applications, containing information about customer demographics, financial status, and loan-related attributes.

2.2 Dataset Dimensionality

The dataset consists of **20,000** rows and **36** columns. Each row represents an individual loan application, while the columns correspond to the various attributes collected for analysis.

2.3 Dataset Target Variable

The target variable in the dataset is **LoanApproved**, which indicates whether a customer's loan application was approved (*1*) or denied (*0*). This binary variable serves as the outcome to be predicted by the machine learning models.

2.4 Dataset Features

2.4.1 By Kind of Feature. The dataset includes the following kinds of features:

- **Demographic features:** These include attributes such as age, gender, education level, marital status.
- **Financial features:** These cover income level, credit score, net worth, and existing debts.
- **Loan-related features:** These represent attributes such as loan amount requested, loan duration, and loan purpose.

2.4.2 By Type of Feature. The features of the dataset can be categorized as follows:

- **Continuous features:** Attributes such as annual income, credit score, and loan amount.
- **Categorical features:** Variables like gender, marital status, and loan purpose.
- **Binary features:** Attributes such as previous loan default status (*1* for defaulted, *0* for not defaulted).

Note: *For more information and a detailed list of the features, refer to table 10 at the end of the Paper.*

2.5 Dataset Priors

The dataset exhibits a class imbalance, with the proportion of approved loans (*1*) being **23.9**% and denied loans (*0*) being **76.1**%. This imbalance in the data set is natural, as more loans are denied than accepted.

3 DATA PREPARATION

3.1 Preprocessing

Our overall preprocessing included cleaning, standardizing, one-hot encoding, and removing correlated feautres from the dataset. This included the following steps:

(1) **Dropping unnecessary columns:** Redundant columns such as *ApplicationDate* and *RiskScore* were removed to simplify the data set and focus on relevant features.

(2) **Dataset summary:** Numerical and categorical features were summarized to provide an understanding of the dataset's structure, including the unique values in categorical features and their respective counts.

(3) **Class priors:** The proportions of each class in the target variable *LoanApproved* were calculated to assess any class imbalance.

(4) **Categorical feature analysis and encoding:** Categorical features such as *EmploymentStatus*, *EducationLevel*, *MaritalStatus*, and *HomeOwnershipStatus* were ordinally encoded based on a predefined mapping. Furthermore, one-hot encoding was applied to the *LoanPurpose* feature to convert it into multiple binary columns.

(5) **Normalization:** Continuous numerical features, excluding binary and ordinal features, were normalized using z-score standardization to ensure they are on a similar scale and improve model convergence.

(6) **Correlation analysis:** A correlation matrix heatmap was generated to visualize relationships between numerical features. Features with a correlation above a threshold of 0.7 were flagged, and multi-collinear columns such as *Experience*, *AnnualIncome*, and *LoanAmount* were dropped to reduce redundancy.

(7) **Saving the dataset:** The preprocessed dataset was saved as a new CSV file named *PreprocessedDataset.csv*, ensuring reproducibility and ease of use in subsequent modeling steps.

4 MODEL 1: LOGISTIC REGRESSION

4.1 Why Logit?

Logistic Regression was chosen as the first model due to its simplicity and interpretability for binary classification. It helped us understand the data better and set a benchmark for comparing accuracies with advanced models [4]. This section discusses the underlying principles of Logistic Regression, its implementation and training details, results, relevant features, and its limitations with suggestions for future improvements.

4.2 Principle Behind Logit

The Logistic Regression model is built upon the logistic (sigmoid) function:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

where $z = w^T x + b$ is a linear combination of the input features and their respective weights. This function maps predictions to a probability range of $[0, 1]$. The model is trained by minimizing the log-loss function, which penalizes incorrect predictions more heavily as their confidence increases:

$$J = -\frac{1}{n} \sum_{i=1}^{n} [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)]$$

where:

- n: Number of samples in the dataset.
- y_i: True label of the i-th sample (0 or 1).
- $\hat{y}_i$: Predicted probability for the positive class ($y = 1$).

We used Gradient-Descent to iteratively update the weights w and bias b to minimize the loss function:

$$w \leftarrow w - \eta \frac{\partial J}{\partial w}, \quad b \leftarrow b - \eta \frac{\partial J}{\partial b}$$

where η is the learning rate.

4.3 Implementation and Training

Key steps in the implementation and training process for the model are as follows:

(1) The dataset was split into training (67%) and validation (33%) sets.
(2) A bias term was added to the feature matrix.
(3) The model was trained using a learning rate of 0.1 for 3000 epochs, during which the weights and bias were iteratively updated to minimize the log-loss function.
(4) Log-loss values for both training and validation datasets were recorded at each epoch to monitor convergence.

For further testing - the Python code for this implementation is modular and stand-alone, and allows easy adjustments to parameters such as the learning rate, number of epochs, and feature selection.

4.4 Results

This subsection presents the performance metrics of the Logistic Regression model and visualizes the log-loss trends for training and validation data over the training epochs.

4.4.1 Metrics. The Logistic Regression model achieved high performance on both the training and validation datasets. The results are summarized in Table 1.

Metric	Training	Validation
Accuracy	0.9457	0.9472
Precision	0.8969	0.8901
Recall	0.8775	0.8797
F1-Score	0.8871	0.8849

Table 1: Performance metrics for Logistic Regression on training and validation datasets.

4.4.2 Log-Loss Diagram. Figure 1 illustrates the log-loss for both training and validation datasets over the training epochs. The convergence of both curves demonstrates that the model successfully minimizes the loss function, indicating stable training and generalization. The close alignment between the training and validation loss curves suggests that the model is neither overfitting nor underfitting, providing confidence in its predictive capabilities on unseen data.

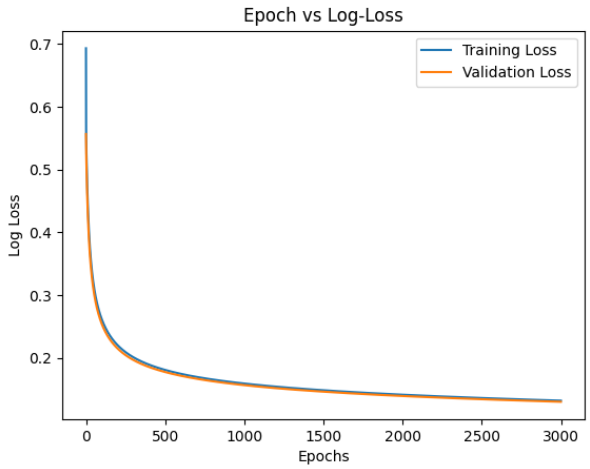

Figure 1: Epoch vs Log-Loss for Logistic Regression.

4.5 Relevant Features

Logistic Regression provides insight into the importance of features through learned weights. Table 2 lists the top 10 features ranked by their absolute weight values.

The weights indicate the influence of each feature on the likelihood of a loan being approved. Positive weights suggest that increasing the feature value increases the likelihood of approval, while negative weights suggest the opposite. These weights align closely with what we would expect in terms of financial knowledge. The top 5 features are listed below:

Feature	Weight
MonthlyIncome	3.1663
InterestRate	-2.5248
TotalDebtToIncomeRatio	-2.0674
BankruptcyHistory	-1.5101
MonthlyLoanPayment	-1.4636
PreviousLoanDefaults	-1.3756
NetWorth	1.3352
CreditScore	-1.1501
LengthOfCreditHistory	0.8338
EmploymentStatus	-0.5027

Table 2: Top 10 Relevant Features from Logistic Regression.

- **MonthlyIncome (3.1663):** This feature has the highest positive weight, reflecting its strong contribution to loan approval. Higher monthly income naturally reduces the perceived risk to the lender, indicating the applicant's ability to repay the loan.

- **InterestRate (-2.5248):** A significant negative weight suggests that higher interest rates are strongly associated with

loan denials. High interest rates might indicate a higher risk assigned to the applicant, making approval less likely.

- **TotalDebtToIncomeRatio (-2.0674):** This feature measures the proportion of an applicant's income that goes toward debt repayment. A high ratio signals financial instability, reducing the chances of approval.

- **BankruptcyHistory (-1.5101):** A negative weight here indicates that applicants with a history of bankruptcy are less likely to have their loans approved due to the high risk they represent.

- **MonthlyLoanPayment (-1.4636):** Larger loan payment amounts relative to income reduce the likelihood of approval, as they suggest a higher financial burden on the applicant.

- **CreditScore (-1.1501):** This was a surprising result for us as it's generally the opposite in real life. We believe that the small negative weight might indicate that the relationship between credit score and loan approval in this dataset is not linear or might interact with other features.

4.6 Limitations and Future Improvements

While the Logistic Regression model performs well, it has certain limitations:

- **Linear Decision Boundary:** Logistic Regression assumes a linear relationship between features and the log-odds of the target, which may not capture complex patterns in the data.
- **Sensitivity to Outliers:** Large feature values can disproportionately influence the model despite standardization.

Future improvements include:

- Exploring non-linear models, such as polynomial Logistic Regression or Support Vector Machines with kernels, to capture more complex relationships.
- Utilizing regularization techniques like L1 or L2 to enhance generalization and reduce the impact of less informative features.

5 MODEL 2: SUPPORT VECTOR MACHINE (LINEAR KERNEL)

5.1 Why Linear SVM?

The first Support Vector Machine (SVM) model uses a linear kernel and is trained using a gradient-based approach. Linear SVM was chosen for this task due to its ability to effectively handle high-dimensional feature spaces by finding a separating hyperplane that maximizes the margin between classes. Unlike the SVMs we used in class that use Quadratic Programming to optimize the objective function, this implementation utilizes a gradient-based optimization method, which is computationally efficient for larger datasets.

This section discusses the mathematical principles behind the linear kernel, implementation details, results, relevant features, and the model's limitations with potential improvements.

5.2 Principle Behind Linear SVM

The objective of Linear SVM is to find a hyperplane that maximally separates data points belonging to two classes. The mathematical formulation of this problem is as follows:

5.2.1 Optimization Objective. The optimization problem is defined as minimizing the cost function:

$$J(\mathbf{w}, b) = \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{n}\sum_{i=1}^{n}\max(0, 1 - y_i(\mathbf{w}\cdot\mathbf{x}_i - b))$$

where:

- $\mathbf{w}$: Weight vector defining the hyperplane.
- b: Bias term, which adjusts the position of the hyperplane.
- λ: Regularization parameter controlling the trade-off between maximizing the margin and minimizing misclassification.
- $y_i \in \{-1, 1\}$: Target label for the i-th data point.
- $\mathbf{x}_i$: Feature vector of the i-th data point.
- n: Number of training samples. [5]

5.2.2 Gradient Descent for Optimization. This implementation of Linear SVM uses gradient descent to iteratively optimize the weight vector $\mathbf{w}$ and bias term b. The gradients are derived based on the following cases:

- For correctly classified samples with sufficient margin ($y_i(\mathbf{w}\cdot\mathbf{x}_i - b) \geq 1$):

$$\frac{\partial J}{\partial \mathbf{w}} = \lambda\mathbf{w}, \quad \frac{\partial J}{\partial b} = 0$$

- For misclassified samples or those with insufficient margin ($y_i(\mathbf{w}\cdot\mathbf{x}_i - b) < 1$):

$$\frac{\partial J}{\partial \mathbf{w}} = \lambda\mathbf{w} - y_i\mathbf{x}_i, \quad \frac{\partial J}{\partial b} = -y_i$$

The weights and bias are updated iteratively using the following equations:

$$\mathbf{w} \leftarrow \mathbf{w} - \eta\frac{\partial J}{\partial \mathbf{w}}, \quad b \leftarrow b - \eta\frac{\partial J}{\partial b}$$

where η is the learning rate. This approach ensures gradual convergence to an optimal solution. [6]

5.3 Implementation and Training

The training process involved the following steps:

(1) The dataset was split into training (67%) and validation (33%) sets.
(2) Labels were transformed to -1 and 1 to align with the SVM formulation.
(3) The weights $\mathbf{w}$ and bias b were initialized to zeros.
(4) The weights and bias were updated iteratively using the gradient equations for 300 epochs.
(5) Regularization was applied using $\lambda = 0.005$ to penalize large weight values and prevent overfitting.

The learning rate ($\eta = 0.0005$) and number of epochs were chosen through experimentation to ensure convergence while avoiding instability or excessive training time.

5.4 Results

This subsection presents the performance metrics of the Linear SVM model and visualizes the log-loss trends for training and validation data over the training epochs.

5.4.1 Metrics. The Linear SVM achieved the following performance metrics, as summarized in Table 3.

Metric	Training	Validation
Accuracy	0.9549	0.9527
Precision	0.8958	0.8915
Recall	0.9194	0.9099
F1-Score	0.9075	0.9006

Table 3: Performance metrics for Linear SVM on training and validation datasets.

5.4.2 Log-Loss Diagram. Figure 2 illustrates the log-loss for both training and validation datasets over the training epochs. The convergence of both curves demonstrates that the model successfully minimizes the loss function, indicating stable training and generalization. The close alignment between the training and validation loss curves suggests that the model is neither overfitting nor underfitting, providing confidence in its predictive capabilities on unseen data.

Figure 2: Epoch vs Log-Loss for Linear SVM.

5.5 Relevant Features

The learned weight vector $\mathbf{w}$ provides insights into the importance of individual features. Table 4 lists all features and their weights.

The top 5 features are detailed below:

- **MonthlyIncome (1.5457):** This feature has the highest positive weight, indicating that higher monthly income increases the likelihood of loan approval. This aligns with domain knowledge, as lenders consider higher income as a sign of financial stability.

Feature	Weight
MonthlyIncome	1.5457
InterestRate	-1.2328
TotalDebtToIncomeRatio	-1.1135
BankruptcyHistory	-0.7499
MonthlyLoanPayment	-0.6994
NetWorth	0.6779
PreviousLoanDefaults	-0.6081
CreditScore	-0.5570
LengthOfCreditHistory	0.3728
EducationLevel	0.3335

Table 4: Feature weights for Linear SVM.

- **InterestRate (-1.2328):** A significant negative weight suggests that higher interest rates are associated with lower approval probabilities. This reflects the perception of higher interest rates as indicative of higher risk.
- **TotalDebtToIncomeRatio (-1.1135):** A higher debt-to-income ratio decreases the chances of loan approval, as it signals financial strain and limited repayment capacity.
- **BankruptcyHistory (-0.7499):** The negative weight reflects the reduced likelihood of approval for applicants with a history of bankruptcy due to perceived risk.
- **MonthlyLoanPayment (-0.6994):** Higher monthly loan payments relative to income reduce approval probabilities, as they indicate a higher financial burden.

5.6 Limitations and Future Improvements

While the Linear SVM performed well, there are several limitations:

- **Hyperparameter Sensitivity:** The learning rate (η) and regularization parameter (λ) must be carefully tuned to ensure convergence.
- **Feature Scaling Dependence:** The gradient-based approach relies heavily on standardized features to produce consistent updates.
- **Linear Decision Boundary:** The model assumes that the data is linearly separable, which may not hold true for our dataset.

Future improvements include:

- Exploring other kernelized SVMs (e.g., polynomial or RBF kernels) to handle non-linear relationships in the data.
- Implementing more adaptive learning rates to improve convergence efficiency.
- Converting the Primal Problem of SVM into a Dual Problem and solving the Dual Problem.

6 MODEL 3: SUPPORT VECTOR MACHINE (POLYNOMIAL KERNEL)

6.1 Why Polynomial SVM?

The Support Vector Machine (SVM) with a polynomial kernel was chosen to explore non-linear decision boundaries. Polynomial kernels are well-suited for datasets with correlated features that influence the target variable, allowing the model to learn more complex relationships compared to a linear SVM. This section explains mathematical foundation, implementation, results, and future directions for Polynomial SVM.

6.2 Principle Behind Polynomial SVM

The polynomial kernel introduces non-linearity into the SVM by transforming the feature space. The kernel function for degree d is given by:

$$K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j + 1)^d$$

where:

- $\mathbf{x}_i, \mathbf{x}_j$: Feature vectors of the i-th and j-th samples.
- d: Degree of the polynomial kernel, which controls the complexity of the decision boundary.
- The constant 1: Ensures that the kernel remains positive and allows for bias in the feature space.

Using this kernel, the optimization objective for SVM remains the same as with the linear kernel:

$$J(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

subject to the constraints:

$$0 \leq \alpha_i \leq C, \quad \sum_{i=1}^{n} \alpha_i y_i = 0$$

where:

- α_i: Lagrange multipliers for each sample.
- C: Regularization parameter controlling the margin-width trade-off.

6.3 Implementation and Training

The polynomial SVM model was implemented using custom Python code with the following key steps:

(1) The dataset was preprocessed and dimensionality was reduced using Principal Component Analysis (PCA) to retain 95% of the variance.
(2) Polynomial kernels of degrees 2 and 3 were applied to transform the feature space.
(3) The kernel matrix was computed for the training data, and Lagrange multipliers (α) were calculated using matrix inversion.
(4) Predictions on validation data were obtained by computing the decision function with the kernel matrix.

The regularization parameter $C = 1.0$ was used to balance margin maximization and misclassification penalty.

6.4 Results

The performance of the polynomial SVM model with degrees 2 and 3 is summarized in Table 5.

Metric	Degree 2	Degree 3
Training Accuracy	0.9268	0.9608
Validation Accuracy	0.9208	0.9095
Training Precision	0.9463	0.9580
Validation Precision	0.9249	0.8400
Training Recall	0.7378	0.8757
Validation Recall	0.7218	0.7605
Training F1-Score	0.8292	0.9150
Validation F1-Score	0.8108	0.7982

Table 5: Performance metrics for Polynomial SVM models with degrees 2 and 3.

The SVM with a degree 3 polynomial kernel performed best on the training data, achieving a higher F1-score and better recall compared to degree 2. However, its validation performance was slightly worse than degree 2, indicating potential overfitting.

6.5 Limitations and Future Improvements

While the polynomial kernel SVM demonstrated strong performance, the following limitations were observed:

- **Overfitting with Higher Degrees:** The degree 3 kernel exhibited signs of overfitting, as evidenced by its higher training accuracy but lower validation performance compared to degree 2.
- **Computational Complexity:** Polynomial kernels, especially with higher degrees, increase the computational burden due to the growth of the kernel matrix size and complexity.
- **Limited Improvement over Linear SVM:** The performance improvements from polynomial kernels were marginal, suggesting that the dataset may not contain significant non-linear patterns.

Note: *An RBF kernel was also evaluated, but the results were significantly worse. This indicates that the RBF kernel may not align well with the characteristics of this dataset.*

Future work could address these limitations by:

- Exploring ensemble methods, such as combining polynomial kernels of different degrees, to reduce overfitting while capturing complex patterns.
- Investigating advanced regularization techniques, such as elastic net regularization, to improve generalization.
- Incorporating kernel optimization methods to dynamically choose the best kernel type and parameters based on the dataset.

7 MODEL 4: ENSEMBLE METHOD (LOGISTIC REGRESSION AND LINEAR SVM)

The Ensemble model combines the strengths of Logistic Regression and Linear SVM through two distinct voting mechanisms: hard voting and soft voting. This section discusses the motivation, mathematical foundation, implementation, results, and potential future improvements for the Ensemble model.

7.1 Why Ensemble Method?

The Ensemble model was chosen to leverage the complementary strengths of Logistic Regression and Linear SVM. Logistic Regression provides interpretability and probabilistic outputs, while Linear SVM excels in finding optimal decision boundaries. Thus, combining the two models helps reduce the individual biases and errors, leading to improved generalization.

Both hard and soft voting were employed to explore different ensemble strategies. While hard voting relies on majority rule based on model predictions; soft voting combines model probabilities to make a weighted prediction.

7.2 Principle Behind Ensemble Method

The Ensemble model integrates Logistic Regression and Linear SVM predictions in the following ways:

7.2.1 Hard Voting. Hard voting predicts the final label based on the majority vote:

$$\text{Ensemble Prediction} = \text{mode}(\text{Logit Prediction}, \text{SVM Prediction})$$

where the SVM predictions $(-1, 1)$ are first converted to binary labels $(0, 1)$ for consistency with Logistic Regression.

7.2.2 Soft Voting. Soft voting averages the probabilities from Logistic Regression and the decision values from Linear SVM (converted into probabilities using the sigmoid function):

$$\text{Combined Probability} = \frac{\text{Logit Probability} + \text{SVM Probability}}{2}$$

The sigmoid function used for SVM decision values is defined as:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

The final ensemble prediction is derived by thresholding the combined probability:

$$\text{Ensemble Prediction} = \begin{cases} 1, & \text{if Combined Probability} \geq 0.5 \\ 0, & \text{otherwise} \end{cases}$$

7.3 Implementation and Training

The Ensemble model was implemented in the following way:

- Training Logistic Regression and Linear SVM models separately using their respective methods.
- Obtaining predictions from Logistic Regression (probabilities and binary predictions) and Linear SVM (decision values and binary predictions).
- Converting SVM predictions to match that of the Logit model.
- Applying hard and soft voting to combine predictions from both models.
- Evaluating the ensemble predictions using accuracy, precision, recall, and F1-score.

7.4 Results

The performance metrics of the Ensemble model for both hard and soft voting strategies are presented in Table 6.

Metric	Hard Voting	Soft Voting
Accuracy	0.9502	0.9481
Precision	0.9408	0.8942
Recall	0.8368	0.8791
F1-Score	0.8858	0.8866

Table 6: Performance metrics for the Ensemble model with hard and soft voting.

Hard voting slightly outperformed soft voting in terms of accuracy and precision, whereas soft voting achieved higher recall. Both approaches demonstrated balanced performance, indicating that the Ensemble model effectively combines the strengths of Logistic Regression and Linear SVM.

7.5 Limitations and Future Improvements

While the Ensemble model improves prediction performance, it has certain limitations:

- **Dependence on Component Models:** The Ensemble model's performance is constrained by the individual capabilities of Logistic Regression and Linear SVM.
- **Lack of Scalability:** Combining models adds computational overhead, which may become a bottleneck for larger datasets or more complex base models.

Future work could explore:

- Adding more diverse base models, such as polynomial SVM or decision trees, to create a more robust ensemble.
- Using weighted voting schemes that dynamically adjust weights based on model confidence or performance on specific subsets of the data.
- Experimenting with more advanced ensemble techniques, such as stacking, to further enhance predictive performance.

8 MODEL 5: ARTIFICIAL NEURAL NETWORK (ANN)

8.1 Why ANN?

The Artificial Neural Network (ANN) was chosen as the high standard to compare the performance of our hardcoded models against a more complex deep learning approach. ANNs have been widely used in related works, demonstrating strong performance in binary classification tasks such as loan approval prediction. This section details the mathematical principles, implementation, results, relevant features, and future improvements for the ANN model.

8.2 Principle Behind ANN

The ANN used for this study is a feedforward neural network comprising three fully connected layers with ReLU activation and dropout regularization. [2]

8.2.1 Mathematical Representation. The model's forward propagation can be described as:

$$\mathbf{a}^{(1)} = \text{ReLU}(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1)$$
$$\mathbf{a}^{(2)} = \text{ReLU}(\mathbf{W}_2\mathbf{a}^{(1)} + \mathbf{b}_2)$$
$$\hat{y} = \sigma(\mathbf{W}_3\mathbf{a}^{(2)} + \mathbf{b}_3)$$

where:

- $\mathbf{W}_i$ and $\mathbf{b}_i$: Weight matrices and bias vectors for layer i.
- $\mathbf{a}^{(i)}$: Activation output of layer i.
- $\sigma(z) = \frac{1}{1+e^{-z}}$: Sigmoid function for output layer, mapping predictions to probabilities.

8.2.2 Loss Function. The Binary Cross-Entropy (BCE) loss function is used:

$$J = -\frac{1}{n}\sum_{i=1}^{n}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)]$$

where y_i is the true label and $\hat{y}_i$ is the predicted probability for the i-th sample.

8.2.3 Optimization. The model is optimized using the Adam optimizer, which adaptively adjusts learning rates for each parameter. Regularization is applied through:

- Dropout layers to prevent overfitting by randomly deactivating neurons during training.
- Early stopping to halt training once validation loss fails to improve for 5 consecutive epochs.

8.3 Implementation and Training

The ANN was implemented using PyTorch, and the training process included:

(1) Splitting the dataset into training (67%) and validation (33%) sets.
(2) Training the model for a maximum of 100 epochs with a learning rate of 0.001, batch size of 32, and patience of 5 for early stopping.
(3) Monitoring Binary Cross-Entropy loss for both training and validation datasets during training.
(4) Saving the best-performing model based on validation loss.

8.4 Results

The performance metrics of the ANN model on the training and validation datasets are summarized in Table 7.

Figure 3 illustrates the Binary Cross-Entropy loss for training and validation datasets across epochs. The convergence of the curves demonstrates stable training and validation processes.

8.5 Relevant Features

The importance of input features was determined based on the absolute weights of the first hidden layer. Table 8 lists the top 10 features ranked by their importance scores.

The top five features and their contributions to the model are:

Metric	Training	Validation
Accuracy	0.9775	0.9639
Precision	0.9499	0.9276
Recall	0.9550	0.9254
F1-Score	0.9525	0.9265

Table 7: Performance metrics for ANN on training and validation datasets.

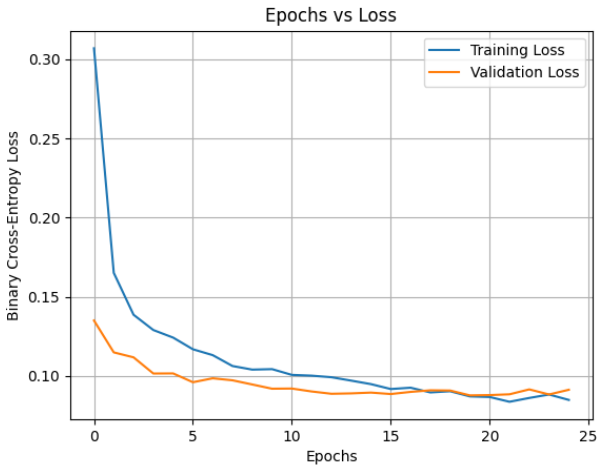

Figure 3: Epochs vs Loss for ANN.

Feature	Importance
MonthlyIncome	24.9299
BankruptcyHistory	24.8756
InterestRate	19.5580
TotalDebtToIncomeRatio	17.3426
PreviousLoanDefaults	15.7918
MonthlyLoanPayment	12.3033
NetWorth	12.1042
Age	10.8356
CreditScore	9.9946
LengthOfCreditHistory	8.4824

Table 8: Top 10 Relevant Features from ANN.

- **MonthlyIncome (24.9299):** The most influential feature, highlighting the significance of stable income in loan approval.
- **BankruptcyHistory (24.8756):** Reflects the lender's preference for applicants without a history of bankruptcy.
- **InterestRate (19.5580):** Indicates that lower interest rates are associated with higher loan approval chances.
- **TotalDebtToIncomeRatio (17.3426):** Demonstrates the importance of financial stability in assessing loan eligibility.
- **PreviousLoanDefaults (15.7918):** Suggests that past defaults are critical in determining loan approval likelihood.

8.6 Limitations and Future Improvements

While the ANN model outperformed other approaches, it has certain limitations:

- **Computational Cost:** Training deep learning models requires significant computational resources compared to simpler models.
- **Interpretability:** Unlike Logistic Regression, ANNs are less interpretable as there might be hidden layers within the neural network.

Future improvements include:

- Experimenting with additional layers and neurons to capture even more complex patterns in the dataset.
- Incorporating advanced regularization techniques, such as L2 regularization or learning rate scheduling, to further enhance performance.

9 RESULTS AND ANALYSIS

This section provides a comparative analysis of the performance of the models implemented in this study. The analysis focuses on the overall results, identifies the best-performing model, and discusses why it outperforms the others.

9.1 Overall Model Comparison

The performance of the models is summarized in Table 9. The metrics—Accuracy, Precision, Recall, and F1-Score—are presented for both the training and validation datasets. These metrics provide a comprehensive view of the models' ability to generalize to unseen data. The models with the best and worst performance in each metric of training and validation set are marked as green and red respectively.

9.2 Analysis of Results

The models exhibit varying performance levels based on their complexity and underlying assumptions. The key observations are as follows:

- **ANN as the Best-Performing Model:** The ANN achieved the highest metrics across all categories, indicating its superior ability to capture complex, non-linear patterns in the dataset. Its capacity to model feature interactions and handle high-dimensional data contributed to its high performance.
- **Logistic Regression and Linear SVM:** Both models performed the best among our hard-coded models. Linear SVM slightly outperformed Logistic Regression due to its ability to maximize the margin, ensuring better generalization but neither of them were overfitting or underfitting the model.
- **Ensemble Models:** The ensemble models combined the strengths of Logistic Regression and Linear SVM. Hard voting was slightly better than soft voting, as it effectively balanced precision and recall. But they weren't able to detect the minority class (Loan Approvals) as well as the above models.
- **Polynomial SVM (Degree 2 and 3):** While the degree-2 polynomial kernel achieved good precision, its recall was comparatively lower, leading to a reduced F1-score. The degree-3 polynomial kernel had even more disparity between it's training and validation metrics implying that the model was overfitting.

Model	Accuracy	Precision	Recall	F1-Score
Training Metrics				
Logistic Regression	0.9457	0.8969	0.8775	0.8871
Linear SVM	0.9549	0.8958	0.9194	0.9075
Polynomial SVM (Degree 2)	0.9268	0.9463	0.7378	0.8292
Polynomial SVM (Degree 3)	0.9608	0.9580	0.8757	0.9150
ANN	0.9775	0.9499	0.9550	0.9525
Validation Metrics				
Logistic Regression	0.9472	0.8901	0.8797	0.8849
Linear SVM	0.9527	0.8915	0.9099	0.9006
Polynomial SVM (Degree 2)	0.9208	0.9249	0.7218	0.8108
Polynomial SVM (Degree 3)	0.9095	0.8400	0.7605	0.7982
ANN	0.9639	0.9276	0.9254	0.9265
Ensemble (Hard Voting)	0.9502	0.9408	0.8368	0.8858
Ensemble (Soft Voting)	0.9481	0.8942	0.8791	0.8866

Table 9: Performance Comparison Across Models.

9.3 Why ANN Outperformed the Other Models?

The ANN's superior performance can be attributed to the following factors:

- **Complexity Handling:** Unlike linear models, the ANN could effectively model non-linear relationships and interactions among features, providing a more comprehensive understanding of the data.
- **Regularization:** Dropout layers and early stopping mitigated overfitting, allowing the ANN to generalize well to the validation dataset.
- **Adaptive Optimization:** The use of the Adam optimizer ensured efficient and stable training by dynamically adjusting learning rates.

10 DISCUSSION

Analyzing the features across all models provides significant insights into the factors most influential in predicting loan approval. Several features consistently appeared with high coefficients or weights in the models, indicating their strong predictive power. Below, we discuss these features and their implications in greater detail:

10.1 Monthly Income

Across all models, *MonthlyIncome* emerged as one of the most critical features. Its high positive coefficients in Logistic Regression, Linear SVM, and ANN signify that higher income is strongly associated with loan approval. This aligns with domain knowledge,

as lenders view high income as a key indicator of an applicant's ability to repay the loan.

10.2 Bankruptcy History

BankruptcyHistory consistently appeared with high negative coefficients in models like Logistic Regression and ANN. Its importance highlights the significant weight lenders place on an applicant's financial reliability. A history of bankruptcy is a strong predictor of loan denial, reflecting the risk-averse nature of financial institutions.

10.3 Interest Rate

The *InterestRate* feature consistently exhibited a strong negative relationship with loan approval across models. High interest rates often correlate with greater perceived risk on the part of the lender. This aligns with financial practices where higher-risk applicants are assigned higher rates, reducing their chances of approval.

10.4 Total Debt-to-Income Ratio

TotalDebtToIncomeRatio was another feature that consistently carried high importance. Its negative weight indicates that applicants with a higher proportion of their income allocated to existing debts are less likely to be approved. This feature serves as a measure of financial stability and the ability to manage additional credit.

10.5 Previous Loan Defaults

PreviousLoanDefaults frequently appeared as a top predictor with negative coefficients. This result is expected, as past loan defaults suggest a higher likelihood of future defaults, thereby reducing the probability of approval.

10.6 Opportunities for Further Exploration

While these features were consistently significant, further exploration into secondary features, such as *NetWorth* and *LengthOfCreditHistory*, could provide additional insights. Moreover, incorporating domain-specific adjustments, such as dynamic interest rates or regional economic conditions, could enhance the overall performance of models.

11 FUTURE WORK/EXTENSIONS

11.1 Exploring Advanced Regularization Techniques

Future work can involve incorporating advanced regularization techniques, such as L1 or elastic net regularization. These methods can further enhance model generalization by reducing overfitting and emphasizing the most relevant features. Elastic net, in particular, combines the benefits of both L1 and L2 regularization, making it suitable for datasets with correlated features.

11.2 Ensemble Model Optimization

While the ensemble models demonstrated balanced performance, there is room for improvement by optimizing their design. This includes experimenting with weighted voting mechanisms, where model predictions are assigned weights based on their performance. Additionally, incorporating more diverse base models, such as polynomial SVM or decision trees, could lead to a more robust ensemble.

11.3 Hyperparameter Tuning

A key area for improvement is rigorous hyperparameter tuning across all models. Techniques such as grid search or randomized search can be employed to systematically explore combinations of learning rates, regularization parameters, and other hyperparameters. This can lead to better-converging models with improved performance metrics.

11.4 Incorporating Dynamic Features

The dataset currently uses static features for training. Future work could include dynamic features, such as time-series data representing changes in credit scores or income levels over time. These temporal insights can enhance predictive accuracy by capturing trends that static features cannot reflect.

11.5 Addressing Class Imbalance with SMOTE

To tackle the significant class imbalance in the dataset, future efforts could involve using the Synthetic Minority Over-sampling Technique (SMOTE). SMOTE generates synthetic samples for the minority class by interpolating between existing samples, which can help balance the class distribution. This method could improve model performance, particularly recall, by ensuring that the minority class is adequately represented during training.

REFERENCES

[1] M. Alhamid. 2020. How to Attain a Deep Understanding of Soft and Hard Voting in Ensemble Machine Learning Methods. *Towards Data Science* (2020).

[2] Jason Brownlee. 2021. Develop Your First Neural Network With PyTorch Step-By-Step. https://machinelearningmastery.com/develop-your-first-neural-network-with-pytorch-step-by-step/ Accessed: 2024-12-11.

[3] S. Kumar and R. Singh. 2023. An ensemble machine learning based bank loan approval predictions system with a smart application. *Journal of Financial Technology and Innovation* 5, 2 (2023), 123–135.

[4] A. Mahgoub. 2024. Optimizing Bank Loan Approval with Binary Classification Method and Deep Learning Model. *Open Journal of Business and Management* 12, 3 (2024), 1970–2001.

[5] S. Peng, Q. Hu, J. Dang, and Z. Peng. 2017. Stochastic Sequential Minimal Optimization for Large-Scale Linear SVM. In *Neural Information Processing. ICONIP 2017 (Lecture Notes in Computer Science, Vol. 10634)*, Derong Liu, Shengli Xie, Yirong Li, Dzmitry Zhao, and El-Sayed M. El-Alfy (Eds.). Springer, Cham, 257–265. https://doi.org/10.1007/978-3-319-70087-8_30

[6] S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter. 2011. Pegasos: Primal Estimated sub-GrAdient SOlver for SVM. *Mathematical Programming* 127, 1 (2011), 3–30. https://doi.org/10.1007/s10107-010-0420-4

[7] B. Yildiz. 2022. Predicting acceptance of the bank loan offers by using support vector machines. *International Advanced Researches and Engineering Journal* 6, 2 (2022), 142–147. https://doi.org/10.35860/iarej.1058724

[8] Lorenzo Zoppelletto. 2023. Financial Risk for Loan Approval. https://www.kaggle.com/datasets/lorenzozoppelletto/financial-risk-for-loan-approval/data?select=Loan.csv

Feature Name	Description	Type	Used/Dropped
ApplicationDate	Loan application date	Date	Dropped
Age	Applicant's age	Continuous	Used
AnnualIncome	Yearly income	Continuous	Used
CreditScore	Creditworthiness score	Continuous	Used
EmploymentStatus	Job situation	Categorical	Used
EducationLevel	Highest education attained	Categorical	Used
Experience	Work experience	Continuous	Dropped
LoanAmount	Requested loan size	Continuous	Dropped
LoanDuration	Loan repayment period	Continuous	Used
MaritalStatus	Applicant's marital state	Categorical	Used
NumberOfDependents	Number of dependents	Continuous	Used
HomeOwnershipStatus	Homeownership type	Categorical	Used
MonthlyDebtPayments	Monthly debt obligations	Continuous	Used
CreditCardUtilizationRate	Credit card usage percentage	Continuous	Used
NumberOfOpenCreditLines	Active credit lines	Continuous	Used
NumberOfCreditInquiries	Credit checks count	Continuous	Used
DebtToIncomeRatio	Debt to income proportion	Continuous	Used
BankruptcyHistory	Bankruptcy records	Binary	Used
LoanPurpose	Reason for loan	Categorical	Used
PreviousLoanDefaults	Prior loan defaults	Binary	Used
PaymentHistory	Past payment behavior	Continuous	Used
LengthOfCreditHistory	Credit history duration	Continuous	Used
SavingsAccountBalance	Savings account amount	Continuous	Used
CheckingAccountBalance	Checking account funds	Continuous	Used
TotalAssets	Total owned assets	Continuous	Dropped
TotalLiabilities	Total owed debts	Continuous	Used
MonthlyIncome	Income per month	Continuous	Used
UtilityBillsPaymentHistory	Utility payment record	Continuous	Used
JobTenure	Job duration	Continuous	Used
NetWorth	Total financial worth	Continuous	Used
BaseInterestRate	Starting interest rate	Continuous	Dropped
InterestRate	Applied interest rate	Continuous	Used
MonthlyLoanPayment	Monthly loan payment	Continuous	Used
TotalDebtToIncomeRatio	Total debt against income	Continuous	Used
LoanApproved	**Loan approval status**	Binary	**Target Variable**
RiskScore	Risk assessment score	Continuous	Dropped

Table 10: Dataset Features: Names, Descriptions, Types, and Whether They Were Used or Dropped.