Predictive Modeling for Credit Card Application Outcomes

Name: Shubham Damodar Kalekar
ID: 240931696

1. Introduction and Problem Statement

The credit card application sector faces major obstacles when assessing the financial reliability of individual applicants. The evaluation process may produce several mistakes like classification errors which cause important consequences. Financial institutions face major monetary penalties when they extend credit to individuals who will not fulfil their repayment obligations. Approved credit applicants might experience rejected applications through unfair procedures which damage their financial prospects and their trust for credit systems. Credit application assessment demands precise predictive models because current challenges reveal significant evaluation issues.

Through the application of machine learning methods this research project aims to forecast credit application results. Researchers use multiple features to determine if the credit card application will be accepted or rejected. The project endeavours to reduce credit evaluation mistakes and improve financial decision-making through its adoption of sophisticated computational methods.

To achieve this, three machine learning models are evaluated and compared: Logistic Regression, K-Nearest Neighbors (KNN), and Neural Networks. The choice of each model rested upon its distinctive strengths for processing data alongside making predictive analyses. Through its ability to identify linear relationships between input features and target variable Logistic Regression enables user interpretations of specific feature contributions to decision-making processes. The non-parametric K-Nearest Neighbors model forms part of this system because its performance strengths emerge from how it identifies points within datasets with complex patterns by finding local similar instances. Machine learning practitioners select Neural Networks to handle sophisticated non-linear data relationships because these architectures perform well when working with complex high-dimensional datasets.

Models require high-quality inputs which the evaluation process delivers through data preprocessing plus statistical analysis procedures. The critical data preparation techniques involve tackling missing data values while normalizing numerical features and encoding categorical variables plus resolving class imbalances within the dataset. Performance measurements including accuracy estimates alongside precision results recall statistics with F1-scores provide detailed evaluation of predictive strength for every model tested.

This project conducts an evaluation of model capabilities and constraints to discover the best practical credit card application prediction method. Results from this research may improve fairness and precision in credit decisions to create advantages for both

lending organizations and credit applicants. The frameworks and understanding from this research will act as the groundwork for upcoming credit risk evaluation and financial decision-making progressions.

2. Analysis of the Dataset

The analysed dataset consists of 1,000 entries which have 15 features and use a binary target variable called Rejected. The target variable records credit card application results and assigns the number 1 to applications that were rejected and 0 to those that were approved. The binary nature of the dataset enables its use in classification projects focused on the prediction of application results.

Numerical Features

Various numerical features within the dataset work together to yield distinct insights about application results. The annual income feature reveals positive skewness because most recorded values pile up in the lowest income levels demonstrating that most applicants belong to lower-income category. The existence of high financial backgrounds becomes evident from dataset elements where annual income surpasses 100,000 dollars and establishes financial diversity among the applicants. Summary statistics verify the positive skewness of annual income because the mean substantially exceeds the median.

The distribution of employed days reveals a major grouping at -1,200 which probably denotes applicants who are currently unemployed. Analysis of the applicant distribution indicates that unemployment stands out as a common trait which affects their chances of rejection. Analysis of applicants' job histories becomes more apparent through binned visual displays that accurately show work history clustering patterns and distributions.

Categorical Features

The dataset's categorical elements deliver further information about applicants' characteristics. According to the Gender feature information 60% of the applicants are male while the remaining 40% are female. Potential biases during prediction analysis will appear from this observed imbalance if researchers fail to address it. The dataset shows 30% of applicants own a car while 70% do not demonstrating potential financial stability indicators through car ownership. The bar charts function to visually present categorical feature distributions using distinct patterns.

Statistical Summary

Numerical feature statistics enable complete comprehension of the data-set characteristics. Multiple financial records for applicants demonstrate a mean income about 45,000 but also considerable variation as reflected by a 20,000 standard deviation and a lower median of 35,000. The statistical analysis for employment duration shows

an average of negative 800 days alongside a midpoint figure of negative 1,200 days and variability of standard deviation 300 which signals widespread unemployment.

Correlation Analysis

Through the observation of a correlation heatmap we can identify how different features interact with the target variable. A moderate positive correlation of 0.45 between annual income and application acceptance demonstrates its status as one of the dataset's strongest predictors. The direct predictive value of employed days towards the outcome variable is restricted because its correlation results are insignificant. Pairwise correlations among dataset features reveal that applicants with high incomes tend to possess longer employment durations which could exert indirect influence on their application acceptance probabilities.

Target Variable Distribution

Approval rates show a class distribution where 60% of cases fall into accepted applications as target = 0 and 40% end up rejected with target = 1. Proportional comparisons in bar charts help display the application acceptance imbalance where they compare accepted personnel against those deemed unsuitable.

3. Methods

3.1 Preprocessing

Before training machine learning models' data preprocessing emerges as an essential phase to confirm dataset quality and reliability. The preparation of the dataset involved multiple methods that handled missing values while standardizing numerical data and selecting essential analytical features.

Handling Missing Values: The dataset handled missing numerical and categorical feature values through individual procedures to ensure data integrity. The mean value of each column served as the substitution for missing numerical feature values. The implemented method maintains vital statistical characteristics throughout the dataset while limiting alterations to numerical data structures. The categorical missing data within columns received modal category replacements representing each column's most prevalent value. This replacement strategy maintains categorical data representative accuracy while avoiding introduction of bias.

Feature Scaling: Prior to running machine learning models, all numerical features received normalization through Z-score scaling to maintain cross-feature consistency and across-model comparability. This standardization technique transforms features so that their distribution centers around zero with one measuring unit of dispersion. K-Nearest Neighbors (KNN) models along with Neural Networks require normalizing features because they perform poorly when input feature values have differing ranges

of magnitudes. Data scaling decreases dominance from larger range features allowing models to rely more heavily on the actual data relationships.

Feature Selection: The model training process became more efficient by only retaining features with an absolute correlation over 0.1 to the target variable during dimensionality reduction. This boundary is designed to permit just those features which display a substantial relation to the target variable to influence the research outcomes. Models achieve improved prediction while benefiting from less computation complexity when irrelevant features along with weakly connected elements are taken out from the dataset.

3.2 Binary Logistic Regression

Logistic Regression is a parametric model commonly used for binary classification problems. It predicts the probability of a positive outcome (class 1) based on a feature vector and learned model parameters. The model hypothesis is represented as:

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

where x ∈Rn in the feature vector, and θ ∈Rn represents the model parameters or weights. This hypothesis function outputs a value between 0 and 1, which can be interpreted as the probability of belonging to class 1.

The decision rule for logistic regression involves setting a threshold of 0.5. If hθ (x)≥0.5, the predicted label y is assigned to class 1. Otherwise, it is assigned to class 0. Mathematically, this can be expressed as:

$$\hat{y} = \begin{cases} 1 & \text{if } h_\theta(x) \geq 0.5 \\ 0 & \text{otherwise} \end{cases}$$

During model training logistic regression works by minimizing binary cross-entropy loss which quantifies discrepancies between predicted probabilities and true class destinations. The loss function is defined as:

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} \left[y^{(i)} \log(h_\theta(x^{(i)})) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)})) \right]$$

where m is the number of training examples, x^(i) is the feature vector for the i-th training example, and y^(i) is its corresponding label. By minimizing this loss function using techniques such as gradient descent, the model learns optimal values for θ, enabling it to make precise predictions.

3.3 K-Nearest Neighbors (KNN)

K-Nearest Neighbors functions as a non-parametric instance-based learning approach that finds extensive use in classification tasks. KNN functions differently than parametric models because it avoids any assumption regarding explicit data distribution. KNN bases its predictions on how similar individual data points appear to each other.

The fundamental principle of KNN assigns a new sample its label based on the labels of its k-closest samples within the feature space. The similarity between data points is measured using the Euclidean distance, defined as:

$$d(x, x') = \sqrt{\sum_{j=1}^{n} (x_j - x'_j)^2}$$

The points x and x′ which are examined as data pair constitute locations within n-dimensional Euclidean space.

By representing its nearest k points from the dataset KNN assigns a class to an input sample based on the most frequent label among these neighbors. The predicted label y is then assigned as the mode of the labels of the nearest neighbors:

$$\hat{y} = \text{mode}\{y^{(1)}, y^{(2)}, \ldots, y^{(k)}\}$$

where y(i) is the label of the i-th nearest neighbor. The parameter k is chosen based on cross-validation to balance the trade-off between model complexity and generalization. A small k may lead to overfitting, while a large k could result in underfitting.

KNN remains easy to use yet yields good results especially for limited data volume or intricate separations. When the size of the dataset expands the computational expense of KNN goes up because you must measure distances between the predicted sample and every existing training instance.

3.4 Neural Networks

The powerful computational models known as Neural Networks derive their design from the structural and functional principles of the human brain. Neural networks create non-linear data pattern recognition with multiple neurons connected layers. The project implementation used a two-layer neural network structure which consisted of one input layer followed by a hidden layer and finished with an output layer. During forward propagation input data flows through each network layer to determine the network's final output value. A linear transformation within the hidden layer transforms the input feature vector x according to its definition.

$$z^{[1]} = W^{[1]}x + b^{[1]}$$

where W^ [1] and b^ [1] are the weight matrix and bias vector for the hidden layer. The resulting z^ [1] values are then passed through the ReLU activation function,

$$a^{[1]} = \max(0, z^{[1]})$$

to introduce non-linearity and enhance the network's capacity to model complex patterns. The output layer receives transformed inputs from the hidden layer to complete its specialized processing tasks. It applies another linear transformation,

$$z^{[2]} = W^{[2]}a^{[1]} + b^{[2]}$$

where W^ [2] and b^ [2 represent the weights and biases for the output layer. The final output, a^ [2], is computed using the sigmoid activation function,

$$a^{[2]} = \frac{1}{1 + e^{-z^{[2]}}}$$

which converts the output into a probability score ranging between 0 and 1. This probability indicates the likelihood of the input belonging to the positive class. During network training the binary cross-entropy loss function evaluates the differences between actual labels and predicted probabilities. The loss function is defined as:

$$J(W, b) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log(a^{[2](i)}) + (1 - y^{(i)})\log(1 - a^{[2](i)})\right],$$

where m is the number of training examples, a^[2](i) is the predicted probability for the i-th example, and y^(i) is its corresponding true label.

Through iterative updates to weights and biases backpropagation and optimization algorithms like gradient descent enable the neural network to minimize the loss function. This approach leads the neural network to learn the best parameters that will result in accurate predictions. Both modeling techniques sharply define intricate data interconnections which allows the neural network to produce dependable classification results.

4. Results of the Prediction Task

Model Performance

The performance of the predictive models was evaluated based on cross-validation accuracy and test accuracy. The results are summarized as follows:

- Logistic Regression achieved a cross-validation accuracy of 89% and a test accuracy of 87%. This performance indicates that Logistic Regression is well-suited for the binary classification task in this dataset, with a balance of simplicity and predictive accuracy.

- K-Nearest Neighbors (KNN) demonstrated a cross-validation accuracy of 86% and a test accuracy of 86%. While its accuracy is slightly lower compared to Logistic Regression and Neural Networks, KNN remains a reliable model for classification tasks.
- Neural Networks matched the performance of Logistic Regression, achieving a cross-validation accuracy of 89% and a test accuracy of 87%. The results indicate successful learning and data generalization by the neural network.

Discussion

Logistic Regression and Neural Networks reached equal performance levels with 89% cross-validation accuracy according to the results. Logistic Regression stands out for efficiency because its parametric structure creates low computational costs and clarity in understanding results. Applications that need fast and straightforward solutions find Logistic Regression exceptionally useful because it works best for binary classification tasks.

The K-Nearest Neighbors algorithm despite its solid 86% accuracy needs more computational power. The distance calculation between all samples throughout prediction phases reveals its non-parametric characteristic. The method loses usability as datasets increase in size or when fast forecasting becomes essential.

Neural Networks successfully mapped complex non-linear patterns within the data but reached only similar performance outcomes as Logistic Regression. The extensive computational resources required by the neural network throughout training period remained high because its iterative optimization process worked alongside its many parameters. The capacity of neural networks to effectively model complex patterns allow their implementation as the preferred option for datasets which involve complex associations undetectable by basic models

While Neural Networks deliver similar performance to Logistic Regression, they also excel at capturing complex data relationships explicitly whereas Logistic Regression stands out through its superior performance alongside its easily understandable outputs. When comparing alternatives to achieve similar goal performance KNN stands as a valid substitute because it delivers marginally lower accuracy while consuming more computational resources. These findings provide essential information to guide model selection according to the projected needs and limits of upcoming prediction endeavours

.

5. Conclusion

The research project provided clear evidence that machine learning approaches can predict credit card application outcomes successfully. It sought to determine application results by distinguishing between accepted and rejected submissions using various

applicant characteristics. It explored binary classification applications by applying three machine learning models which are Logistic Regression together with K-Nearest Neighbors and Neural Networks.

Based on the research findings Logistic Regression along with Neural Networks emerged as the most effective techniques by obtaining 89% cross-validation accuracy and 87% ability to generalize to unseen test data. Because of its parametric nature Logistic Regression achieved both excellent performance characteristics like computational efficiency and ease of understanding along with clear model interpretability which made it fit well for similar prediction scenarios. Although Neural Networks required greater computational power, they showed capability for modeling complex relationships and delivered comparable performance results to Logistic Regression.

K-Nearest Neighbors methodology demonstrated strong performance because it stayed competitive despite generating only 86% accuracy during cross-validation and testing processes. During prediction KNN requires distance calculations between all samples which creates computational overhead that prevents its direct use in larger databases or when real-time responses are needed. As an effective baseline for comparison KNN implements its instance-based methodology despite these restrictions.

Research outcomes reveal machine learning as an effective solution to financial risk assessment problems. Logistic Regression proved to be the most practical approach because it provided straightforward implementation despite performing efficiently alongside Neural Networks which provide dependable outcomes through more intricate structures. The value of these findings leaps dramatically across fields that require dependable forecasts to minimize financial losses while enhancing decision-making capabilities.

Summary of the Problem Tackled

The project worked to determine how likely applicants will effectively manage credit through predictions of credit application results. Erroneously assigned predictions will result in financial consequences when they block creditworthy persons from accessing necessary credit or generate overhead costs. Data-driven decision making in credit risk assessment received a demonstration through machine learning models in this project. This research reveals how machine learning technology functions in financial applications while creating a strong foundation for upcoming technological progress in the industry.

6. References

1. James, G., Witten, D., Hastie, T., & Tibshirani, R. (2021). An Introduction to Statistical Learning: With Applications in R (2nd ed.). Springer. [Chapter 4: Classification Methods].

2. Cover, T. M., & Hart, P. E. (1967). "Nearest neighbor pattern classification." IEEE Transactions on Information Theory, 13(1), 21–27. https://doi.org/10.1109/TIT.1967.1053964

3. Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press. [Chapter 6: Feedforward Deep Networks]. Available online at: https://www.deeplearningbook.org

4. Kuhn, M., & Johnson, K. (2013). Applied Predictive Modeling. Springer. [Chapter 3: Data Pre-Processing]. https://doi.org/10.1007/978-1-4614-6849-3

5. LeCun, Y., Bengio, Y., & Hinton, G. (2015). "Deep learning." Nature, 521(7553), 436–444. https://doi.org/10.1038/nature14539

6. Lessmann, S., Baesens, B., Seow, H.-V., & Thomas, L. C. (2015). "Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research." European Journal of Operational Research, 247(1), 124–136. https://doi.org/10.1016/j.ejor.2015.05.030

7. Zheng, A., & Casari, A. (2018). Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists. O'Reilly Media.

8. Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). "Learning representations by back-propagating errors." Nature, 323(6088), 533–536. https://doi.org/10.1038/323533a0

9. Dataset provided by the course instructor, "Machine Learning with python".