Scientific Research Publishing

Open Journal of Business and Management, 2024, 12, 1970-2001
https://www.scirp.org/journal/ojbm
ISSN Online: 2329-3292
ISSN Print: 2329-3284

Optimizing Bank Loan Approval with Binary Classification Method and Deep Learning Model

Abdalla Mahgoub

Department of Computing, Goldsmiths, University of London, London, UK
Email: Abdalla.mah@gmail.com

How to cite this paper: Mahgoub, A. (2024). Optimizing Bank Loan Approval with Binary Classification Method and Deep Learning Model. *Open Journal of Business and Management, 12,* 1970-2001.
https://doi.org/10.4236/ojbm.2024.123104

Received: January 29, 2024
Accepted: May 28, 2024
Published: May 31, 2024

Abstract

For any bank or financial institution, managing loans and controlling leverage is one of the most important tasks they have to undertake. A bank cannot function efficiently without a well-designed loan-to-deposit business model. As technology continues to evolve, the mechanism of handling and granting loans underwent a significant change with the introduction of use cases concerning machine learning and data science. Hence, this data-driven research utilized advanced machine learning techniques to analyze and manipulate the data, aiming to predict the best possible way to recommend a loan to a client. These predictions are based on modified yet unique features created from the data obtained from the client. The dataset was tested using two different methodologies: a logistic regression model and a Neural Network algorithm. Both of these methodologies produced high-level accuracy rates. However, the latter outperformed the currently used methodologies by over 20%, resulting in an accuracy of 90%. The model achieved its optimal accuracy level through the application of advanced deep learning techniques, optimization, and precise selection of the number of hidden layers, as well as the definition of input and output layers. The optimization process for the deep learning model prioritizes three major activation functions: RELU, Softmax, and Sigmoid. In addition to the previous statement, epochs and batch size were also considered based on the dataset size used in this research paper. The successful research results were obtained due to the use of a perfectly balanced, unbiased, and cleaned dataset, as well as the well-executed combination of activation functions for the Neural Network model. A performance assessment was conducted based on a confusion matrix evaluation to demonstrate its feasibility and performance.

Keywords

Bank Loan Prediction Model, Financial Prediction, Machine Learning Algorithms, Logistic Regression Algorithm, Deep learning Algorithms,

Advanced Data Science Techniques and Analysis, Deep Learning Activation Functions

1. Research Background

One of the most important objectives and main purposes for any financial institution is to capitalize on recent scientific discoveries and incorporate them into their operations to make sound decisions.

The purpose of this research paper is to figure out which standard classification algorithm or machine learning algorithm could be used to produce a suitable solution for a bank or financial institution. Also, we will experiment with deep learning algorithms to see if they would produce a better solution for this model.

This approach benefits the institution's financial position and simultaneously helps their customers reach their financial objectives.

2. Aims and Objective of This Project

The main objective and aim of this project are to leverage machine learning techniques, algorithms, and models to predict the creditworthiness of an individual or a corporation profile for a loan based on certain characteristics and checklists.

The successful implementation of this system could benefit the subject bank in various ways, which will be mentioned later in the results and discussion section of this project.

3. Literature Reviews

Now, let's analyze different literature reviews and report any research findings that could impact the quality of my project concerning relevant projects. Therefore, I will identify any research gaps arising from these literature reviews, expand on implemented methodologies, and attempt to improve the current project.

I have reviewed about three different relevant literature research papers, one of which underperformed in terms of the overall expected outcome. Moreover, I structured my literature review theoretically based on relevancy.

Notes: In this literature research, the research report is irrelevant and poorly addresses the topic. The main focus is on data mining and the impact of external engagements on the overall health of the economy, with no relation to our topic of loan approval prediction application.

On the technical side, the author managed to make an adequate attempt to use machine learning algorithms to dissect the dataset and arrive at a fair exploratory data analysis based on given parameters from the bank's customer data. The algorithm of choice was the decision tree algorithm, which, in my opinion, is the

best algorithm given the presented dataset.

The strengths of the author's approach include the use of two different algorithms and consideration of parameters such as the employment status, marital status, education, credit history, and the number of dependents as the main factors for their machine learning output/results. They also applied optimization techniques and algorithms, such as the cross-validation technique. The evaluation metric used in this case was the confusion matrix, which, in my opinion, is suitable for any classification algorithm, especially the one the author trained (Kulothungan & Gupta, 2021).

Notes: In this paper (Goel, Sheikh, & Kumar, 2020), the authors clearly stated the purpose of the project: to automate the loan approval process for the bank's customers by using a machine learning-driven application to accurately predict whether a customer deserves a loan or not based on given parameters.

The authors demonstrated mastery in the technical aspects of their loan approval project. In comparison to the previous literature survey, the authors successfully provided a well-explained and evaluated project, particularly highlighting its relevance to my project. This paper presented a more concise and improved pre-processing approach for handling loan prediction data. They utilized different initiatives such as log transformation, binning, and one-hot encoding, resulting in cleaner and more reliable data free of outliers. Additionally, they selected the appropriate methodology and algorithm for their model, which suited the data, and concluded that the problem is a classification problem.

The authors used a logistic regression algorithm, a simple yet powerful machine learning algorithm. They employed the logistic sigmoid function to predict the probability that a certain sample belongs to a particular class, resulting in an outcome of either 1 or 0, meaning either yes or no. However, it is unclear if they used optimization techniques like Gradient Descent or K-fold cross-validation to improve the model's results, providing an opportunity to enhance the overall accuracy of the model. The accuracy of this model was reported to be 81%.

The only shortcoming of this paper is the lack of detailed insights and arguments from a literature review standpoint, and the inefficient use of referencing. Consistent with the previous literature survey, the authors used the confusion matrix and heatmap to visualize and evaluate the model, which, in my opinion, is the right evaluation metric to use for the given dataset.

Notes: Dr. Sandeep R. Shinde and Amruta S. Aphale (Shinde & Aphale, 2020) utilized a range of machine learning algorithms to model and study the creditworthiness rating of customers. They compared seven different types of supervised machine learning algorithms and also employed deep learning algorithms in their research. These methodologies were tested using Matlab and the Python scikit-learn package to predict the creditworthiness of bank customers. The authors provided steps to perform data analytics and machine learning testing. They used a standard ratio of 0.66/0.33 representing Train/Test proportion to evaluate the model's performance. Additionally, the authors trained the model

with various algorithms such as Neural Network, Decision Trees, Naïve Bayes, and more. They utilized the Confusion Matrix for model evaluation and provided a comprehensive explanation of the evaluation process. The model achieved an accuracy rate ranging between 76% to 80%. Although a reference on feature selection was mentioned, no further explanation was provided.

After examining various literature reviews about loan approval models, it appears that different machine learning techniques were employed, resulting in overall accuracies between 70% to 81%. However, in the case of literature review-2, I believe the model was overfitting against the given data, which might affect its performance in test and real-life situations (Bell & Waters, 2014: pp. 87-104) (Kulothungan & Gupta, 2021: pp. 894-900) (Shinde & Aphale, 2020).

Various research platforms were used for obtaining and comparing datasets for these previous literature reviews. A brief list of research platforms used is mentioned below:

- Google, Google Scholar.
- University's library catalog.
- eBooks and online articles from various academic websites.

4. Methodology

I have outlined the steps I will take to conduct a thorough analysis and determine the most suitable model for the loan approval application:

4.1. Dataset Description and Data Collection

A list of datasets was compared, reviewed, and investigated; however, I have selected a dataset that is not only more relevant to real-life banking procedures but also a perfect fit for this project. Additionally, it is easy to find this particular dataset on the internet, and its static condition makes it easy to implement. For instance, a dynamic dataset might not always produce consistent results.

The dataset consists of 13 features (columns) and 614 instances related to every client at the bank, indicating whether they might or might not get approval for a loan.

These 13 attributes as shown in **Figure 1**. Moreover, feature selection could be a tedious feat due to its relevance to our project. This dataset is ideal for our project because of the sample feature named "Loan Status," which offers a binary outcome, either "Yes" or "No." Such a result is perfect for our logistic regression model and the neural network model.

Consequently, the dataset aligns with our research questions. Hence, implementing the relevant techniques to produce the anticipated outcome for this project should be challenging. The dataset was obtained from Kaggle (Rabinovich, 2020), The dataset was downloaded from Kaggle under the account name "DEB-DATTA CHATTERJEE" as referenced on reference (Rabinovich, 2020), a sample of the dataset can be downloaded from the link provided under "Data Availability and Access" at the end of the research paper.

Loan_ID	Gender	Married	Dependen	Education	Self_Empl	ApplicantIn	Coapplicar	LoanAmou	Loan_Amc	Credit_His	Property_/	Loan_Status
LP001002	Male	No	0	Graduate	No	5849	0		360	1	Urban	Y
LP001003	Male	Yes	1	Graduate	No	4583	1508	128	360	1	Rural	N
LP001005	Male	Yes	0	Graduate	Yes	3000	0	66	360	1	Urban	Y
LP001006	Male	Yes	0	Not Gradu	No	2583	2358	120	360	1	Urban	Y
LP001008	Male	No	0	Graduate	No	6000	0	141	360	1	Urban	Y
LP001011	Male	Yes	2	Graduate	Yes	5417	4196	267	360	1	Urban	Y
LP001013	Male	Yes	0	Not Gradu	No	2333	1516	95	360	1	Urban	Y
LP001014	Male	Yes	3	Graduate	No	3036	2504	158	360	0	Semiurban	N
LP001018	Male	Yes	2	Graduate	No	4006	1526	168	360	1	Urban	Y
LP001020	Male	Yes	1	Graduate	No	12841	10968	349	360	1	Semiurban	N
LP001024	Male	Yes	2	Graduate	No	3200	700	70	360	1	Urban	Y
LP001027	Male	Yes	2	Graduate		2500	1840	109	360	1	Urban	Y
LP001028	Male	Yes	2	Graduate	No	3073	8106	200	360	1	Urban	Y
LP001029	Male	No	0	Graduate	No	1853	2840	114	360	1	Rural	N
LP001030	Male	Yes	2	Graduate	No	1299	1086	17	120	1	Urban	Y
LP001032	Male	No	0	Graduate	No	4950	0	125	360	1	Urban	Y
LP001034	Male	No	1	Not Gradu	No	3596	0	100	240		Urban	Y

Figure 1. Excel format snapshot view of our dataset. **Source**: The dataset was obtained from Kaggle (Rabinovich, 2020).

As mentioned earlier, the data was obtained from the world-renowned website "Kaggle" (Rabinovich, 2020). The dataset comprises standardized data related to a list of approved and rejected loans by various applicants. The data in the dataset is considered a well-structured data type.

4.2. Data Cleaning and Pre-Processing

For this stage, the dataset in question was selected due to its relevance and perfect alignment with the project's final outcome. To begin, I unloaded the dataset into our integrated development environment (IDE) to run various experiments for our analysis. During the first stage of data cleaning, unnecessary feature data such as "Loan ID" was dropped since it wasn't necessary to include it in the overall dataset.

Subsequently, the second stage of data cleaning involved dealing with missing data and replacing them with either the mode or the mean. By adopting this technique, we achieved balanced data without irregularities that could impact our machine learning experiment in later stages, as shown in **Figure 2**.

The data we received is raw, and in real life, we usually have to deal with data of different data types altogether. Therefore, our case is no different. The last stage of cleaning involves converting our categorical variable data types from an object format into integers or another data type. This will allow us to have a workable data type for our experiments. Basically, I have managed to map these categorical variables to integer data types.

At this juncture, I will provide a preliminary review of the data presented in the dataset, and I should be able to distinguish useful data points, prominent features, as well as outliers. This stage is not only an important phase for data preparation but also will guide me in the right direction to clean the data properly for the feature selection and model selection stages.

```
[10]: df1['Gender'].fillna(df1['Gender'].mode()[0],inplace=True)
      df1['Married'].fillna(df1['Married'].mode()[0],inplace=True)
      df1['Dependents'].fillna(df1['Dependents'].mode()[0],inplace=True)
      df1['Self_Employed'].fillna(df1['Self_Employed'].mode()[0],inplace=True)
      df1['Loan_Amount_Term'].fillna(df1['Loan_Amount_Term'].mode()[0],inplace=True)
      df1['Credit_History'].fillna(df1['Credit_History'].mode()[0],inplace=True)

[11]: df1.LoanAmount = df1.LoanAmount.fillna(df1.LoanAmount.mean())

[12]: df1.isnull().sum() # as we see all missing data are replaced with either the mode or mean

[12]: Gender              0
      Married             0
      Dependents          0
      Education           0
      Self_Employed       0
      ApplicantIncome     0
      CoapplicantIncome   0
      LoanAmount          0
      Loan_Amount_Term    0
      Credit_History      0
      Property_Area       0
      Loan_Status         0
      dtype: int64
```

Figure 2. A code snippet of dealing with missing data in python. **Source**: Code snippet generated from Jupyter Notebook.

Aside from cleaning missing data and dealing with outliers, I will focus on using important categorical variables that might be crucial to include in the feature engineering and selection stage. For example, if I plan to include marriage status in the overall model training, I might as well use One-Hot encoding techniques in my ML model.

4.3. Feature Selection

There are many feature selection techniques that I can choose from to select intrinsic properties of the features represented in my loan approval dataset. However, I would like to focus on filtering methods that emphasize information gain or using the variance threshold technique, which focuses on removing all features whose variance doesn't meet a certain threshold or number.

Hence, in this phase, I will select a list of relevant features and parameters that will be useful to build a model for predicting a loan based on these selected features. By adding and selecting relevant features and variables, the overall complexity of the model will increase, and the generalization capability will also improve.

In this approach, I've created a list of new independent variables or features based on the given variables, which will have an impact on providing a better understanding and reliability of the model. Our original list of features, including our modified features, is briefly shown in Table 1.

We can understand from the table as shown in Table 1, that I've added 12 features to our dataset which in my opinion will add value to the overall performance of our model. These new features or variables are derived from our existing features, for example, the feature named "combined application income" was created and modified from simply the sum of data points of application of

Table 1. A list of original and modified features.

No.	Feature name	Data type	Feature status	Parameters
1	**Gender**	Integer	Original	-
2	**Married**	Integer	Original	-
3	**Dependents**	Integer	Original	-
4	**Education**	Integer	Original	-
5	**Self_Employed**	Integer	Original	-
6	**ApplicantIncome**	Integer	Original	-
7	**CoapplicantIncome**	Integer	Original	-
8	**LoanAmount**	Float	Original	-
9	**Loan_Amount_Term**	Float	Original	-
10	**Credit_History**	Integer	Original	-
11	**Property_Area**	Float	Original	-
12	**combined_application_income**	Float	Modified	[Application income], [co application income]
13	**EMI**	Float	Modified	["LoanAmount"], ["Loan_Amount_Term"]
14	**loan_per_income**	Float	Modified	["LoanAmount"], ["combined_application_income"]
15	**dependents_mean_per_EMI**	Float	Modified	["Dependents"], ["EMI"]
16	**EMI_Per_Term**	Float	Modified	[EMI], ["Loan_Amount_Term"]
17	**Loan_Amount_Term_Per_Income**	Float	Modified	["Loan_Amount_Term"], ["combined_application_income"]
18	**EMI_Per_loanamount**	Float	Modified	["EMI"], ["LoanAmount"]
19	**dependents_loan_sum**	Float	Modified	["Dependents"], ["LoanAmount"]
20	**Credit_history_income_sum**	Float	Modified	["Credit_History"], ["combined_application_income"]
21	**property_area_loanamount_per_totalincome**	Float	Modified	["Property_Area"], ["loan_per_income"]
22	**Loanamount_term_binning**	Float	Modified	["Loan_Amount_Term"]
23	**total_income_binning**	Float	Modified	["combined_application_income"]

Source: Author's own work.

"Applicationincome" and "Coapplicationincome" to have a wider view of the total income per applicant, as an applicant could be an individual with joint bank accounts with his or her spouse, therefore a wider view on total income is essential for a better prediction model. All equations related to these newly modified features are stated in [**Appendix A.1**].

In my opinion, all these features are essential to provide a better working machine learning model with optimal accuracy and effective performance on real

life test data.

On the other hand, if you notice, I haven't mentioned an important independent feature which is the "Loan status". This particular variable is indeed the target variable which we will use to understand our model and to learn patterns based on this historical data.

Data binning:

In this section, I've utilized an advanced level of outliers control and management known as discretization or binning. Essentially, binning is the process of transforming continuous variables and functions into discrete variables and forms. This is achieved by creating a set of bins or connecting intervals that span across the range of our desired variables. Adopting this technique will make our model more maintainable and faster overall, by effectively managing variables. This technique is especially useful when dealing with linear data that is updated in real-time through systems like Spark Streaming, Kafka, or Apache Storm. Hence, it would be efficient if a banker can quickly read and evaluate data provided by a live feed system on newly approved and rejected applications. Binning can enhance the quality and accuracy of overall performance, considering this level of variables management.

Data Split:

The final version of the cleaned data is now ready for splitting. Therefore, I have used sklearn to split our data into training and testing datasets with an initial split ratio of 80%/20%, representing the train and test datasets, respectively.

4.4. Model Selection and Training

There are many angles for improving this project. These improvements can be experimented with based on the observed methodologies used in previous related work, as mentioned earlier in the literature review section, starting from choosing the best methodology that can maximize our results. I used the logistic regression model for my project; however, I will also employ gradient descent or related optimization techniques to see if I can achieve better results.

As an alternative approach, I used a Neural Network algorithm, initializing random weights across the given features and trying different layers and techniques to achieve better overall accuracy. Now, let's discuss how well these methodologies will impact the overall outcome.

Logistic Regression Model

A logistic regression algorithm is a supervised classification algorithm used to describe data and explain the relationship between one dependent binary variable and one or more independent variables. The logistic equation is designed in such a way that the outcome probability value can be simply mapped to classes, and the values can only be presented between 0 and 1.

The function used to produce these two binary outcomes is the Sigmoid Function. The sigmoid function is given by the formula below, Sigmoid formula, also

used in Sigmoid activation function.

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

The sigmoid function is the core mathematical function for our classifier, which takes our features, multiplies each one by weight, sums them up, and includes them in the sigmoid function.

The sigmoid function produces an output of either 0 or 1. It is well-known for its characteristic sigmoid curve, as shown in **Figure 3**. In the context of my project, this algorithm is a perfect match due to the limited outcome, restricted to either "yes" or "no," corresponding to 0 and 1. (Geron, 2017) (Bach & Alpaydin, 2014).

Neural Network model

Deep learning or Artificial Neural network(ANN) is one of the hottest fields of Data science right now, as many case studies concerning ANN has an interesting result in image recognition, robotics, and autonomous driving application, so why not try to experiment with our Loan approval dataset and see if we can get a better prediction result than logistic regression.

Simply put, ANN is used in machine learning to mimics the function process of the human brain, where many things are connected to perform a specific task. This process is consist of layers with input and output, these inputs and outputs are determined by perceptrons or in another word "Neurons" known as "Nodes" see **Figure 4**.

Basically, these neurons are mathematical units that connect weighted inputs of layers through an activation function which helps the neuron to produce results for the next neuron on the following layer. Weights are determined and randomly initialized as well as optimized during the training session of the model to minimize the loss function of the model (Chollet, 2017) (Willems, 2019), see below figure (**Figure 5** is for illustration purposes only)

Then, a propagation technique is used during model training by propagating the error back into the neurons or nodes and updating the weights and other parameters.

Many activation functions could be used to produce optimal predictions and results for our model, such as hyperbolic tangent, Sigmoid, ReLU, and softmax function. However, for my project, I used ReLU function, softmax, and sigmoid for my final output results. Hence, this perfect mixture of functions in this case might be a perfect fit for our dataset, specifically the softmax function.

The reason why I plan to choose this function is that it is similar to the sigmoid function concept, where it will return the target with the highest probability and generate an output between 0 and 1.

The primary difference between the two approaches is that the sigmoid function is used for binary classification challenges, while the softmax function is used for multivariate classification challenges (Chollet, 2017) (Willems, 2019).

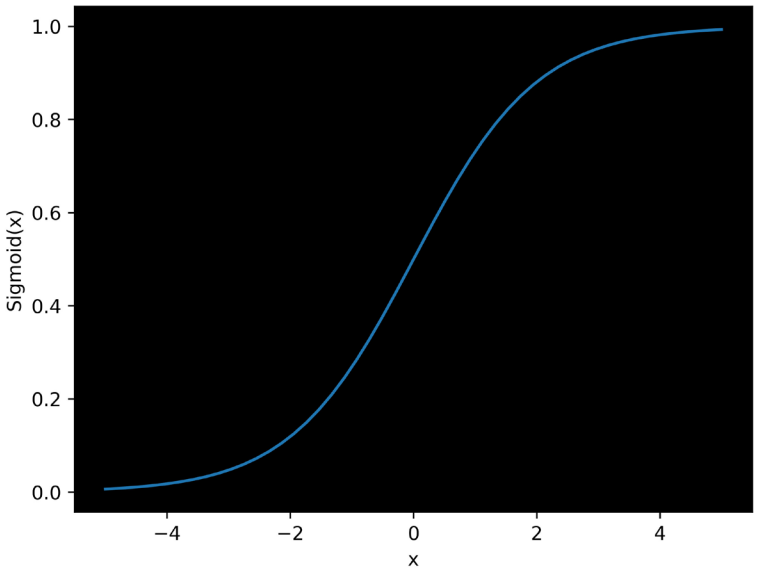


Figure 3. Visualization of Sigmoid Curve. **Source**: Author's own work.

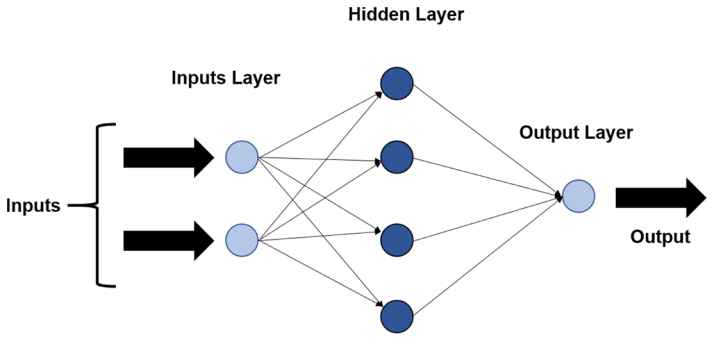


Figure 4. Neural Network structure design. **Source**: Author's own work.

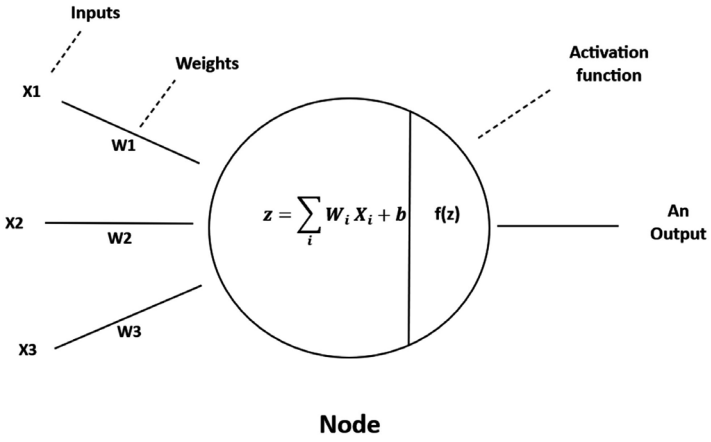


Figure 5. Structure design of one perceptron. **Source**: Author's own work.

Splitting data into training and test data

This is a very important part of the project, splitting our cleaned dataset into training and test datasets will not only help us train our model in a training en-

vironment, but also will assist me to test it on the test dataset to see how well the model might perform on a real dataset if it were used effectively.

Consequently, the proper split ratio in these cases is 20/80 or 30/70, reflecting test/training dataset, this is an essential part of feature engineering for our model development. The dataset was split into training dataset representing a shape of (491, 21), while testing dataset was split to the shape of (123, 21), since we have a total observation points of 614.

4.5. Model Evaluation

The most important part of this stage is implementing an evaluation metric that suits our classification problem. Since I will present a classification challenge for this project, the evaluation metric techniques that will be a perfect fit and can be implemented in our case will be the confusion matrix. This matrix will provide us with a list of classification prediction matrices that can be used to derive three main metrics: accuracy, precision, and recall.

4.6. Model Analysis and Reporting

This is simply the final section of this research, where I will analyze and communicate the results of my experiments. I will also state my findings and explain how the statistical methods or logical techniques I've chosen made a difference and produced better outcomes than peers. These pieces of information will be mentioned in the results and discussion section of this report.

5. Ethical Consideration for the Project

I cannot think of anything more important or interesting to discuss than considering the implementation of regulations and an ethical code of conduct for an AI-driven application in the financial industry (Boddington, 2017).

AI and ML applications have been implemented in the financial services industry for a while now, as they successfully increased optimization processes, reduced overall costs, and improved the efficiency of financial services. Big players in the industry, such as JP Morgan, Goldman Sachs, Morgan Stanley, and BlackRock, have all adopted some form of AI application to enter the new era of efficiency and make faster and better decisions related to capital markets, financial services, etc. These applications could be related to customer service and experience, portfolio management, asset management, wealth advisors, algorithmic trading, and more.

From an ethical standpoint, these applications posed many challenges, which were shared by regulators. For my project, I can only think of the following ethical considerations that could affect and impact the quality of the project:

5.1. Bias and Equal Treatment of all Participants

Ideally, the loan ML model should be free of bias and should not prefer or select

a specific group of loan applicants. For example, if the system kept choosing males over females or certain applicants with specific job statuses without considering their ability to pay back the loan over a period of time based on their history, it shows a clear lack of understanding of how the algorithm makes these decisions and should be investigated.

5.2. Transparency

Ethical considerations are interlinked with the previous point I brought up about bias, as the model could suffer from a lack of transparency. We always fear what we don't understand; in our case, a potential client will not trust and understand the loan approval process if it is handled by an AI algorithm. Hence, it's our job to be transparent about how this algorithm works and how it uses and interprets the given data. This step will make them more trustworthy and alleviate any unjustified fear. The model needs to be transparent and fair, and it shouldn't favor one group of people over another (Swapna, 2018) (Patel, 2020).

6. Results and Outcome

I would prefer to categorize the findings into three specific subcategories:

6.1. Exploratory Data Analysis, Which Encompasses Bivariate Analysis

After cleaning the given data, we initially settled with the following observation, as shown in **Figure 6**.

There are more approved loans compared to rejected loan applications, representing over 68% more than the number of rejected loan applications. Additionally, it seems that you will have a better chance of getting a loan if you are married, educated or graduated, and have a good credit history.

Furthermore, it is not necessary to be self-employed to be considered for a loan. However, unfortunately, the data represented in this dataset is mostly geared in favor of male applicants compared to females. Moreover, many applicants, about 50% of them, do not have dependents or kids. Based on all these observations from the above figure, I can confidently state that the bank is biased in favor of educated, working-class bachelors with no parental responsibilities.

Moving on to analyzing a mixture of features from our cleaned data using a variety of visualization techniques such as box plots, bar charts, and scatter plots. Basically, I took two or more relevant variables to see if these meaningful data are of quality for the project and also will give us a better view of the data at hand. This technique is considered multivariate, as it allows us to review relevant multiple variables together.

Based on the above figure (see **Figure 7**), we can gather that applicant income plays an important role in obtaining a loan from the subject bank. From the above figure, it is evident that the majority of the applicants who received a loan

from the bank are males, with a median total income closer to USD 10,000, while the median total income for female applicants is below that of their male counterparts. The upper quartile or upper whiskers for both males and females are closer to USD 20,000, representing 25% of our total observed applicants.

For both male and female loan applicants, the median income is less than USD 10,000, approximately USD 5000, and around 69% of total applicants successfully obtained a loan from the bank.

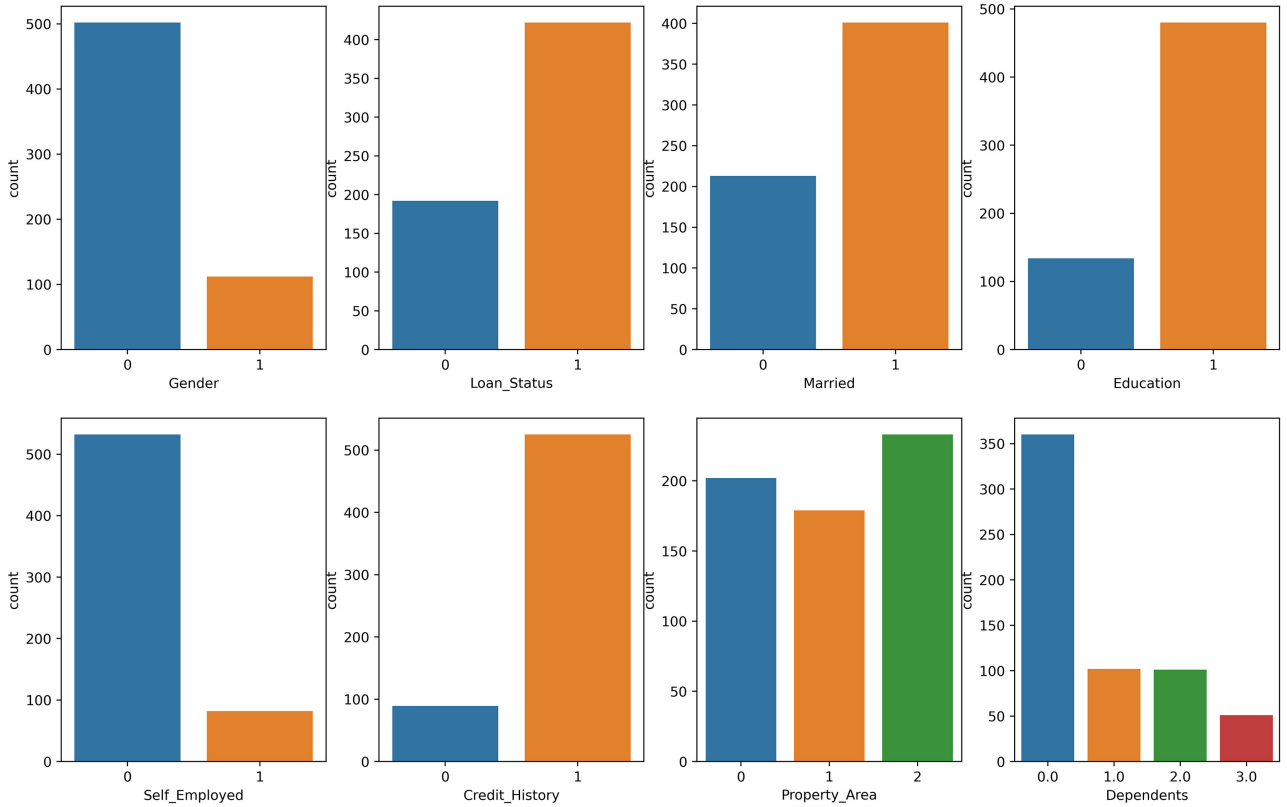
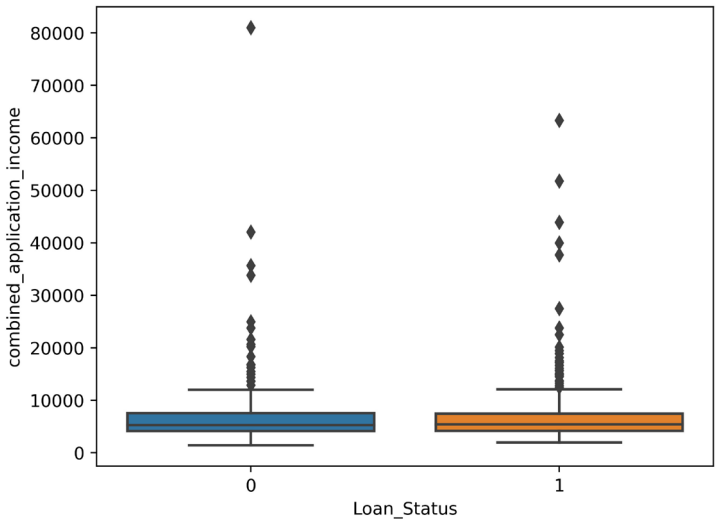


Figure 6. Main features description bar charts. **Source**: Bar charts generated from Jupyter Notebook.



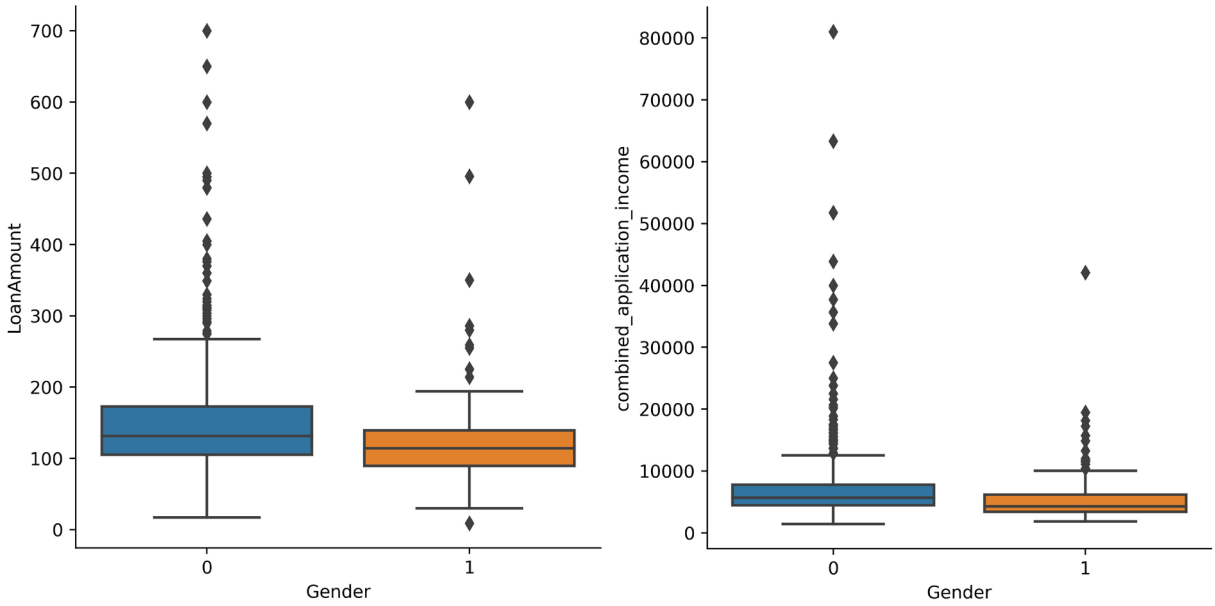


Figure 7. Multivariate boxplot chart for Loans status and Gender. **Source**: Results and charts generated from Jupyter Notebook.

In addition to the above, the scatter plot as shown in **Figure 8** shows that most of the applicants are married, and managed to get a loan amount between 600 to 100, with the majority getting the loan below the 200 mark. Their total income is mostly concentrated at the USD10,000 mark and below.

Obviously from **Figure 9**, We can see that between male and female applicants, graduated males are much more in numbers than graduated females, also male applicants as we mentioned earlier, they are not only educated but they are also making more money than females.

Also, see **Figure 10**, we observe that graduated male applicants are receiving larger loan amounts compared to graduated female applicants. Moreover, this difference makes sense since males generally have much higher total incomes than females.

As shown in **Figure 11**, I have employed an advanced visualization technique that allows me to combine three independent variables. This technique will enable us to observe the relationship between Gender, the total income of applicants, and the type of property area. For example, we can determine if residing in an urban city has a significant impact on total income compared to living in a rural area, which, in turn, could affect the overall loan application. The three property types are categorized as Urban, Rural, and Semi-urban areas, represented in the figure as property area 0, property area 1, and property area 2, respectively.

Based on the figure, we can presume that the majority of male and female applicants who were successfully approved for a loan are residing in urban and semi-urban areas, while a smaller number of approved applicants live in rural areas.

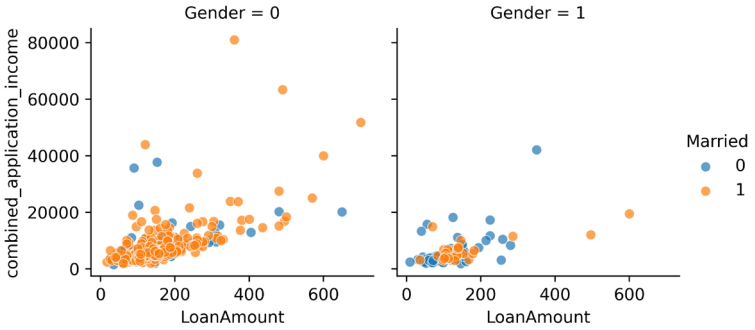


Figure 8. Scatter plot of Loan amount and total income data per Gender. **Source**: Charts generated from Jupyter Notebook.

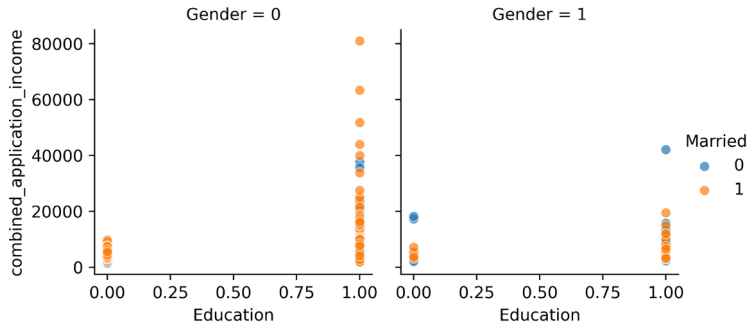


Figure 9. Scatter plot of total income and education data per Gender. **Source**: Charts generated from Jupyter Notebook.

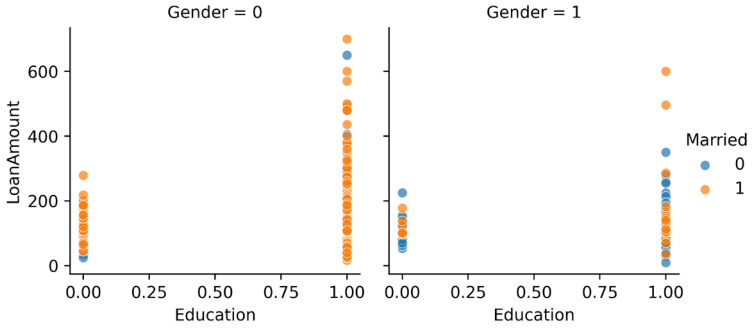


Figure 10. Scatter plot of Education and loan amount data per Gender. **Source**: Charts generated from Jupyter Notebook.

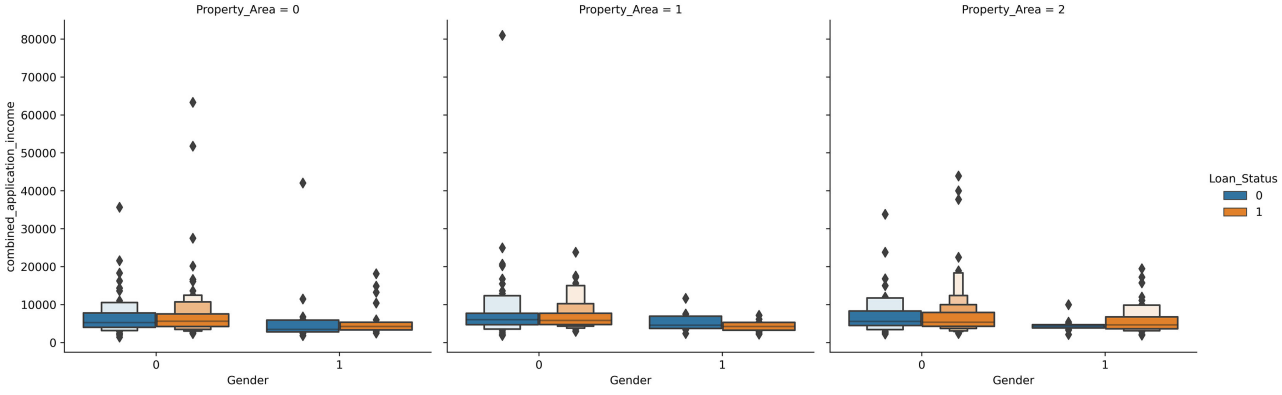


Figure 11. Advanced Boxplot of three features. **Source**: Charts generated from Jupyter Notebook.

Clearly, we can see that applicants living in urban areas have a higher income than applicants residing in semi-urban and rural areas.

In conclusion, our EDA analysis indicates that the overall data is of quality, with relevant and equal information distributed between male and female applicants. However, qualitative data such as loan amount, total income per applicant, and loan status slightly deviate or favor male applicants. Nevertheless, the dataset is of high quality, and it should provide fair results for our loan predictive model.

6.2. Outcomes of the Machine Learning Models

6.2.1. Model 1: Logistic Regression Algorithm

Let's elaborate on the implementation of the Logistic Regression algorithm mentioned earlier and interpret our results based on this algorithm.

We understand that the logistic regression model can interpret statistical output based on favorable criteria. For instance, it can distinguish between an applicant with sufficient total income and a suitable profile, compared to a rejected loan application that exhibits less favorable criteria for obtaining a bank loan.

In my model, we split the dataset into x and y, where x represents the training dataset, and y represents the testing dataset.

First let's understand the case of a single binary predictor, where:

$$x = \begin{cases} 1, & \text{if exposed to favorable factor} \\ 0, & \text{if not exposed to favorable factor} \end{cases},$$

$$y = \begin{cases} 1, & \text{yes loan is approved} \\ 0, & \text{no loan request is rejected} \end{cases}$$

We began our analysis by distinguishing the relationship between our features and target variables, as it is critical in deciding how well the model will perform, considering whether we might have a positive or negative relationship for our model.

Henceforth, I used the R-squared (R^2) and adjusted R-squared (Adjusted R^2) scores to define the relationship and get a sense of how the model will perform using the logistic algorithm as its foundation.

Interpretation of R-squared/Adjusted R-squared:

Before delving into interpreting the R-squared and adjusted R-squared results from the model, let's briefly explain them.

The R-squared score is a statistical method that measures the variations caused by the regression model and explained by the implementation of the coefficient of determination outcome from the relationship of the variables presented in the model. It quantifies the proportion of the variation in our dependent variable **Y** (the target in this case) that can be attributed to the independent variables **X**. The R-squared/Adjusted R-squared formula is explained in [**Appendix A.2**] (Singh, 2019).

On the other hand, the Adjusted R-squared score is simply a modified version of the R-squared score. It reacts when a new independent variable is added to the mix and only increases if the new predictor enhances the model beyond the obtained probability of the model. This is why the Adjusted R-squared is considered an important measure in determining how reliable the correlation of these variables is to the target variable and how much it is determined by the addition of new independent variables.

As shown in **Figure 12**, you can see that I've used a regression relationship technique with the assistance of the "Mutual info regression" library to determine if there is a positive relationship between these features and the target values. Therefore, we can observe that our credit history, income, loan amount, and credit history income have a strong relationship with the target variable, while the remaining features that we created and modified show no relationship with the target. However, this is not sufficient to have peace of mind regarding the relevance of these variables to the target. Hence, we will run a test by incrementally adding random features to our R-squared and Adjusted R-Squared scores and observe if there is any increase or decrease in the total score.

Logically speaking, an increase in features should result in an increase in R-squared, while the Adjusted R-squared should remain the same unless one of the newly added features contributes value to the model and improves its performance and accuracy (Nighania, 2018).

See **Figure 13**, I used an iterative technique to incrementally add a feature at each step. The numbered index denotes the number of features added to the model, and it measures the R-squared and the adjusted R^2, as shown in the figure above. We can observe that the more features we add to the model, the lower the score of the adjusted R-squared becomes, while the R-squared remains the same. The figure above shows that approximately 79% of the variation in the dependent variable is explained by the independent variables. These features are represented and highlighted in yellow (**Figure 13**). Therefore, the results of this test were expected and are in line with our assumption, given the relevance and reliability of the dataset.

Model Evaluation

Next, we move on to model evaluation. I utilized the confusion matrix for model evaluation as it is the best fit for our classification model challenge. In addition to the accuracy results, I also took into account the precision and recall outcomes for the model.

Based on the given results as shown in **Figure 14**, the model exhibits an accuracy of 82.9%, surpassing the 70% to 80% accuracy reported in previous research reviews. As we can see, the preprocessing procedures and feature selection mechanism that have been adopted made a significant difference in the overall performance of our prediction model, resulting in a better predictive classifier than those of our peers.

Logically speaking, having a high accuracy score does not guarantee perfect

```
[74]: Credit_history_income_sum                    0.185522
      Credit_History                               0.136795
      EMI_Per_loanamount                           0.066328
      dependents_mean_per_EMI                      0.029571
      Loan_Amount_Term_Per_Income                  0.026355
      Gender                                       0.021003
      Self_Employed                                0.006899
      combined_application_income                  0.005976
      Loanamount_term_binning                      0.000000
      property_area_loanamount_per_totalincome     0.000000
      dependents_loan_sum                          0.000000
      EMI_Per_Term                                 0.000000
      EMI                                          0.000000
      loan_per_income                              0.000000
      Married                                      0.000000
      Property_Area                                0.000000
      Loan_Amount_Term                             0.000000
      LoanAmount                                   0.000000
      Education                                    0.000000
      Dependents                                   0.000000
      total_income_binning                         0.000000
      dtype: float64
```

Figure 12. Regression relationship result of all features. **Source**: Output generated from Jupyter Notebook.

[53]:

	r2	adj_r2
1	0.793598	0.793176
2	0.793598	0.792752
3	0.793598	0.792327
4	0.793598	0.791899
5	0.793598	0.791470
6	0.793598	0.791040
7	0.793598	0.790607
8	0.793598	0.790172
9	0.793598	0.789736
10	0.793598	0.789298
11	0.793598	0.788858
12	0.793598	0.788417
13	0.793598	0.787973
14	0.793598	0.787528
15	0.793598	0.787080
16	0.793598	0.786631
17	0.793598	0.786180
18	0.793598	0.785727
19	0.793598	0.785272
20	0.793598	0.784815
21	0.793598	0.784356

Figure 13. R^2 and R^2 Adjusted results. **Source**: Results generated from Jupyter Notebook.

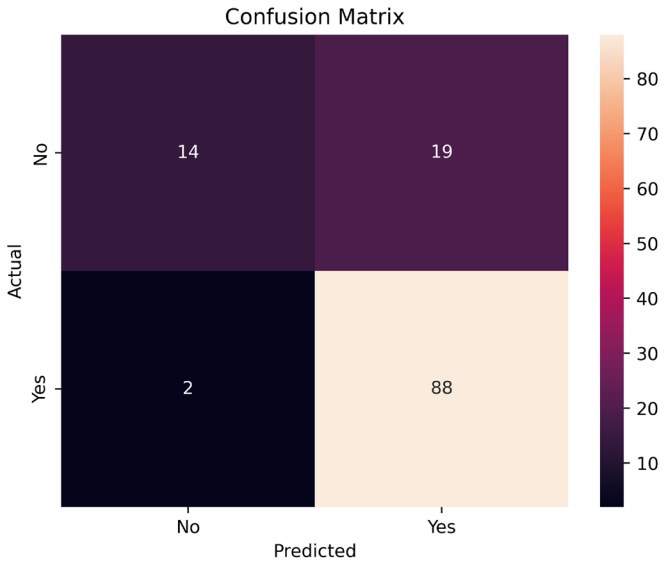


Figure 14. Model 1 confusion Matrix result. **Source**: Graph generated from Jupyter Notebook.

performance in a real-world environment. Therefore, we rely on other evaluation metrics to provide a better assessment of the model's readiness. Consequently, we will use the precision and recall evaluation methods to explain these logistic regression prediction results.

Our precision and recall scores are 82.2% and 97.8%, respectively. This means that from the features used in the model, the proportion of correctly predicted positive outcomes out of all positively predicted outcomes is 82.2% (representing the precision score), while the proportion of correctly positive outcomes predicted out of the total number of positive outcomes is 97.8% (represented by recall). Formulas related to the confusion matrix and equations for accuracy, precision, and recall can be found in [**Appendix A.3**] (Deepanshi, 2021).

6.2.2. Model 2: Neural Network Deep Learning Algorithm

Now that we know how the model will perform by using logistic regression algorithm, let's experiment and adopt deep learning algorithm to our predictive model and compare them to our findings from our first model.

For the successful implementation of this model, I employed various techniques to preprocess the dataset, preparing it for the new neural network-based algorithm using libraries such as TensorFlow. Some of the techniques utilized include SMOTE (Synthetic Minority Over-sampling Technique) for handling imbalanced data, scaling techniques, and selecting appropriate neural network layers like Sequential and Dense for classifier management.

Similar preprocessing methods were used to handle missing values and clean the dataset. However, for this particular model, I employed the following techniques to prepare our dataset for deep learning model testing.

• **Pandas.get dummies() method**: this method is used to convert all string data types and other data types to either 0 and 1 values. This is a common tech-

nique on deep learning application to save time and space for sizable dataset applications.

• **SMOTE() method**: stands for Synthetic Minority Oversampling Technique, basically it is a oversampling technique where you can select a sample from a minority class to assist you in imbalanced data, in our case we have an imbalance between how many yes and no from our target variables. As you see in **Figure 15**, we have an equal balance of yes and no in our model after implementing this technique, the reason I did this, because you need to test a fair model without any bias toward a certain criteria on your model.

• **MinMaxScaler():** This method is a scaling technique of which will help us transform our features by scaling each feature between 0 and 1 values.

• **Sequential()**: it is a class in Keras, that helps to create deep learning models by creating instance of the Sequential class, then create and add layers to it.

• **Dense()**: after defining a sequential class, a dense layer will be connected to the preceding layer which means the neuron from a layers will be connected to the next till you get to the output neuron, which in our case we have 1 output neuron as out result.

Expanding on my earlier definition of the deep learning algorithm, a Sequential class was built, and a total of 200 neurons were added to the model, representing the first layer. This was followed by adding 400 neurons, which can be considered an extension from the first layer to the second layer. Then, 200 neurons were added for the third layer, followed by 100 neurons for the fourth layer.

Subsequently, these 100 neurons were condensed into 4 neurons, representing the fifth layer, before the last single neuron was added to represent the output of our neural network model. In total, we've built 6 layers of perceptrons. These layers were constructed using different activation functions, which are presented in the following order of occurrences:

• Relu
• Softmax
• Sigmoid

The combination of these functions produced a unique yet robust blend for our dataset, resulting in an accuracy of over 87% with an error margin of 35%. We reached this optimal accuracy number by strategically adopting the propagation technique and assigning 100 epochs.

Basically, the propagation technique is a gradient descent algorithm, or in other words, an iterative optimization algorithm for finding a local minimum of a differentiable function. It begins by initializing the weights of these neurons with random values, usually set between 0 and 1. Then, the initial output is computed to calculate the error loss of the model. After that, the weights are adjusted, and the process propagates back to the first layer of the model because the goal is to minimize the loss. These steps are repeated until the optimal solution is reached, which minimizes the loss function. We know we've reached our optimal predictive model once all pre-defined epochs are exhausted in the model.

```
[53]: # now let's run a small analysis
      smote = SMOTE()
      X1, y = smote.fit_resample(dm_x, dm_y)
      sc = MinMaxScaler()
      X = sc.fit_transform(X1)

[54]: Counter (y) # as you can see now it is balanced

[54]: Counter({0: 332, 1: 332})
```

Figure 15. Code Snippet of smote technique. **Source**: Code snippet generated from Jupyter Notebook.

Simply put, from a neuron perspective, increasing the number of epochs leads to further propagation of the process, which updates the weights assigned to each neuron, all the way back to the first neuron. The process is then repeated, updating the results and model layer by layer until we arrive at the newly updated outcome.

The Architecture of the Neural Network

Based on the information concerning the Neural Network algorithm used in our model, we can provide an overview of the neural network architecture. Unfortunately, we cannot view all neurons from the model in a single figure or graph due to the large number of neurons represented for each layer. Hence, we can use the following factors to decide on a suitable architecture for the neural network algorithm at hand:

1) Independent variables: Representing our input layer, we have 480 independent variables for every feature we have.

2) Number of Nodes in hidden layers: Representing how many neurons per hidden layer, in our case, we set 5 hidden layers for our model.

3) Output node: Since our problem is a regression problem, our output shall be either 0 or 1; hence, only one node is required.

A conceptual figure of the above discussion can be represented by the figure introduced earlier, Figure 4.

Model Evaluation: For this stage, I experimented with different parameters on our neural network algorithm, which not only impact the overall performance and prediction capabilities of the model but also the speed of the process for the model in the testing environment. The factors that I used to play around with our classifier were the train-test size, activation functions, batch size, and the total number of neurons used in the model.

A number of experiments were conducted based on different activation functions, from Relu, softmax to sigmoid. However, for this project, I selected a mixed approach to see if we can achieve better performance than a stand-alone logistic regression-driven model.

These tests were carried out with different metrics and inputs. These parameters concern processes related to Neural Network, such as the number of hidden

layers per stage, test size, batch size, and Epochs size, to determine how many times each cycle should go through the propagation process.

As you can see from the above table (See Table 2) of these tests, the results varied in terms of performance and accuracy outputs. Between model 1 and Model 3, where a mixture of activation methods was used successfully, we can conclude that an average accuracy rate of (87.70 + 87.72 + 88.50)/3 = 87.97% could be obtained if we adopted this approach for our bank loan approval model. This is a better result than the 82% we managed to get in our logistic regression model 1 attempt.

As for the remaining models 4 and 5, we used the RELU and Sigmoid activation functions, respectively. As a scientific researcher, it is always a good idea to try different metrics to gauge and test these models, to reach an optimal conclusion on which metric could provide a suitable solution to our problem.

Model 4 exclusively implemented the RELU activation function. Relu is one of the most popular activation functions that operate on each input where all values less than zero are set to zero. It also offers better generalization in neural network modeling. This model produced an accuracy prediction of over 90%, which is impressive. As for model 5, the model used Sigmoid as its base for model learning and prediction, resulting in an accuracy of 86.44.

You can notice that all these models are using sigmoid as their output prediction activation model because this is the right way to handle how the model should learn and predict the probability-based output.

To conclude this stage, I suggest going with a Relu-based model with sigmoid activation function-based output in order to not only maximize our prediction potential but also make them easy to optimize with gradient-descent methods.

Table 2. Table of results of Neural Network model.

Experiment	Model 1	Model 2	Model 3	Model 4	Model 5
Inputs	480 Variables	480 Variables	480 Variables	480 Variables	480 Variables
Test Size	20%	30%	20%	20%	20%
Hidden Layer (1)	Relu 200 nodes	Relu 300 nodes	Relu 500 nodes	Relu 500 nodes	Sigmoid 500 nodes
Hidden Layer (2)	Relu 400 nodes	Relu 500 nodes	Relu 600 nodes	Relu 600 nodes	Sigmoid 600 nodes
Hidden Layer (3)	Relu 200 nodes	Relu 300 nodes	Relu 200 nodes	Relu 200 nodes	Sigmoid 200 nodes
Hidden Layer (4)	Softmax 100 nodes	Softmax 100 nodes	Softmax 150 nodes	Relu 150 nodes	Sigmoid 150 nodes
Hidden Layer (5)	Softmax 4 nodes	Softmax 2 nodes	Softmax 2 nodes	Relu 2 nodes	Sigmoid 2 nodes
Output Layer	Sigmoid 1 node	Sigmoid 1 node	Sigmoid 1 node	Relu 1 node	Sigmoid 1 node
Batch Size	20	30	60	20	20
Epochs	100	150	200	200	200
Accuracy	**87.70%**	**87.72%**	**88.50%**	**90.96%**	**86.44%**

Source: Author's own work.

6.3. Conclusive Results and Discoveries

In this section, I will answer the related research questions and give a detailed analysis of my findings. These findings will help me compare my results to the overall performance of other people's work and point out any areas of improvement that were implemented.

One of the major findings from the above analysis is that our original independent variables are positively correlated with the target variable, which is an amazing find and solidifies our assumption about the quality of our dataset for this project. As we gathered from the R-squared and adjusted R-squared test run, the dataset variables have a positive relationship with the target variables and can be easily explained. To answer the first research question of this paper concerning the use of a classification algorithm for this experiment, we need to look into the facts gathered so far from experimenting with a logistic regression algorithm as a traditional classification solution for our problem.

The model I designed and wrote successfully produced accuracy and a performance level better than its peers. Moreover, not only did the model deliver better results, but it also performed better in real-life environments compared to the other research papers in this field. Moving on to the second model, we adopted and built a neural network algorithm to assist us in producing better prediction results. Thus, for this experiment, we built about 6 layers of neurons and selected a blend of activation functions. However, I plan to use the softmax activation function as it will assist us in logically controlling our dataset from overfitting. Also, I will use Relu and Sigmoid activation functions as output functions.

Expanding on all the trials and experiments conducted by using several neural network activation functions as shown in Figure-18, the findings from this experiment were very interesting, especially if we took a closer look at "Model 4" and "Model 3". Compared to the other models, we can see that an increase in the number of nodes leads to a better reading from our output sigmoid-driven node. Also, an increase in the number of propagation epochs translates into a better weighting setup, which leads to better prediction capabilities for the model.

All these facts helped the model to learn from its mistakes. By adding more layers and using activation functions, we increase the network's expressive power and predict a better output based on the trained target variables.

In this research project, I implemented two different methodologies to address our emphasizing research questions. The first methodology involved implementing the logistic regression algorithm on our dataset to predict the best output for potential loan applicants, while the second methodology involved implementing a neural network algorithm to produce the same predicted output. Both methodologies produced accuracy rates of 82.90% and 90.96%, respectively.

Both of these approaches yielded fair and positive results overall. However, I would like to classify the best findings based on the metrics used in this research. By analyzing the above table as shown in **Table 3**, we can reach the following findings:

Table 3. Table of results of Neural Network model.

	Logistic regression model	Neural Network driven model
Modified Variables	Yes	No
Possibility of Bias	Yes	No
Worst recorded accuracy	78%	82.90%
Best recorded accuracy	**82.90%**	**90.96%**
Diff %	4.90%	8.06%
Algorithm model used	Logistic Regression	Neural Network/Relu, Softmax and Sigmoid

Source: Author's own work.

In the logistic regression model approach, I conducted a comprehensive cleaning and pre-processing of the dataset, including creating modified variables from the existing original variables, to assist and come up with clearer and more relevant observations. On the other hand, I did not conduct the same level of data preparation for the neural network model approach, as the neural network model has some built-in pre-processing techniques that I've adopted, such as SMOTING.

In the first methodology, there is a possibility that my data might be biased since the observed data for males are more than females after dealing with missing data, while in the second methodology, the usage of many relevant techniques helped me remove this risk by rebalancing the dataset to have an equal observation between males and females.

With a margin difference of 4% for the logistic regression model against an 8% difference for the neural network model, we can see that the neural network, in this case, is much superior in terms of producing a high level of prediction result.

From a scientific point of view, the logistic regression approach uses the sigmoid formula [**Appendix A.4**] to come up with the needed output. Therefore, in some way, it's similar to the neural network, as a neural network in my model uses a sigmoid formula for the output. The only difference is that in the neural network model, the model takes into account a mixture of different activation functions [**Appendix A.4**] to tackle different datasets that are fed to the system.

7. Discussion

For this section, I would like to discuss how these results could play out in the real world if this model is adopted at a bank. In addition to discussing these results, keep in mind that both models exercised a decent amount of positive operations on the given dataset. However, in my humble opinion, based on the above results, the neural network has a better application than a stand-alone logistic regression algorithm. The neural network model has more room to grow in terms of possible automation in real life. Moreover, multiple datasets can be

fed into the system to produce better performance results, and the way the model deals with biased data is better than the first approach using a logistic regression model.

Let's dig deeper into the neural network model results, as shown in Table 2. Even though I suggested "model 4" as our winner for this research project, in my opinion, a Sigmoid-driven model will be better for a real-life environment. This means I would rather take an accuracy of 86% as shown in "model 5" (Table 2) rather than an accuracy of 90% from "model 4," and I will state my reasons.

A higher accuracy does not necessarily mean the model predicts all of the labels correctly in a real-life situation, as new data observations might vary from the existing ones. That's why further work on evaluation metrics should be considered. Hence, experimenting with a lower accuracy, such as 86%, to train and improve to capture new observation data from a real-live environment, has a better chance to predict the correct labels than a 90% accuracy since it doesn't appear to be an overfit model overall.

8. Conclusion

In my report, I have demonstrated the use of advanced pre-processing techniques in dealing with a complex biased dataset. Additionally, I managed to leverage machine learning algorithms to successfully predict the correct and right decisions for a bank manager to determine whether a particular client deserves a loan or not. We tested our dataset on two different machine learning models: one is the logistic regression algorithm model, and the other is the neural network algorithm model.

These decisions are quantitative in nature and follow an unbiased end-product dataset, which not only provides a highly accurate solution but also ensures fair and ethical regulated data free of biases. We demonstrated that the more data we fed the model, the better the R^2 adjusted and R metric, which confirms our hunch about the first model (logistic regression algorithm). The first model is a perfect fit for this classification problem. This also proves that the model will perform with fair accuracy on test and real environment data.

On the other hand, our neural network algorithm model offered a better way to tackle our dataset by using a combination of different activation functions, such as Relu, Softmax, and Sigmoid. The sigmoid activation function, as a binary output function, has proven to be prominent compared to the other activation functions. Moreover, this combination impacted the overall accuracy of the model and managed to offer an accuracy of 90.

I also explained why a neural network with a sigmoid activation focus will perform better than the other activation models, even though the accuracy rate is less than the others. Simply because a less accurate model, in this case, has a better chance to perform than a high accuracy model, due to the fear of an overfit model case.

Both of these models can be improved by simply considering adding more re-

latable data points and features. Also, in the case of the neural network model, we can experiment with other classification supervised algorithms such as Random Forest, Naive Bayes, Decision trees, etc. The use case of this project can provide added value to a financial institution or a bank. Of course, such a project should be taken in conjunction with complementary projects, such as Fraud detection and prevention, portfolio management, or even assessment and management of credit risks.

9. Further Work

The beauty of data science and machine learning, is that there are many ways where you can clean, train and implement different pipeline methodologies to your data.

In our case, we can always improve our work from the following perspectives:

9.1. L.1 from Data Research Perspective

Firstly, I would consider using multiple datasets for one project. For example, in the banking loan project, it can be complemented with the bank's current provision against loans. I would create a risk management safe process to alert the bank if they can loan out these facilities and provide loans to their customers based on their current provisions and resources. Doing so will safeguard the bank from classifying their account as an overdraft, and it will also protect their customers from exceeding their loan limit.

In other words, using much more data will help predict the best possible output for a customer. A good data scientist will not only provide an optimal solution for a problem but also look into mitigating and minimizing risks for every approach.

9.2. L.2 from Data Science Methodologies Perspective

Lastly, there are many ways to improve our methodologies, including adopting different pipelines and algorithms such as the GPU-accelerated XGBoost algorithm, Random tree, SVM algorithm, or even the probability-driven algorithm Naive Bayes algorithm. Additionally, various activation functions can be tested on the neural network model, such as the Tanh activation function, Leaky ReLU activation function, and more.

While these algorithms could potentially generate better results than our current methodologies, it's important to note that they might not necessarily outperform the tested neural network model. Therefore, further experiments and trials should take place to determine if these other classification algorithms are worth implementing or not. Also, it won't be a bad idea to implement an automation process to complement the neural network model approach, by simply finalizing and completing the NN (Neural network) classifier based on the model I've created. Lastly, we can experiment with different evaluation techniques to get better evaluation results for our models, such as the ROC Curve or

the F1 Score evaluation technique [**Appendix A.3**] (Bruce & Bruce, 2017).

Acknowledgements

I would like to express my deep appreciation to my family members, especially my father, also Nozima Nazarova, whose inspiration has fuelled my desire to contribute to the data science community and advance knowledge in the field of data science research.

I am also grateful to my alma mater at Goldsmiths, University of London, for providing me with the foundation in Data Science and Artificial Intelligence.

I extend special thanks to Prof. Zimmer and all the faculty and lecturers at the university for their invaluable roles in stimulating our intellectual growth.

Authors Contribution Statement

The Author conceived and designed the study, collected and analyzed the data, interpreted the results, and drafted the manuscript. Also, the Author provided valuable guidance and critical revisions throughout the research process. The author read and approved the final version of the manuscript.

Ethical and Informed Consent for Data Used

Not applicable.

Data Availability and Access

The datasets generated during and/or analyzed during the current study are available in the Zenodo repository at the following link: DOI-link https://zenodo.org/records/10041577

Funding

No funding to declare.

Consent to Participate

Not applicable.

Consent for Publication

Not applicable.

Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

References

Bach, F., & Alpaydin, E. (2014). *Introduction to Machine Learning* (3rd ed., pp. 34-37). MIT Press.

Bell, J., & Waters, S. (2014). *Doing Your Research Project* (6th ed., pp. 87-104). Open University Press.

Boddington, P. (2017). Introduction: Artificial Intelligence and Ethics. In B. O'Sullivan, & M. Wooldridge (Eds.), *Towards a Code of Ethics for Artificial Intelligence Research* (pp. 1-5). Springer. https://doi.org/10.1007/978-3-319-60648-4_1

Bruce, P., & Bruce, A. (2017). *Practical Statistics for Data Scientists* (pp. 250-252). O'Reilly Media.

Chollet, F. (2017). *Deep Learning with Python* (pp. 1-28). Manning Publications Co. LLC.

Deepanshi (2021). *In-Depth Understanding of Confusion Matrix*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/05/in-depth-understanding-of-confusion-matrix/

Geron, A. (2017). *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques for Building Intelligent Systems* (pp. 134-136). O'Reilly.

Goel, A. K., Sheikh, M. A., & Kumar, T. (2020). An Approach for Prediction of Loan Approval Using Machine Learning Algorithm. In *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)* (pp. 490-494). Institute of Electrical and Electronics Engineers. https://doi.org/10.1109/ICESC48915.2020.9155614

Kulothungan, A. N. H., & Gupta, K. (2021). Loan Forecast by Using Machine Learning. *Turkish Journal of Computer and Mathematics Education, 12,* 894-900.

Nighania, K. (2018). *Various Ways to Evaluate a Machine Learning Model's Performance. Towards Data Science*. https://towardsdatascience.com/various-ways-to-evaluate-a-machine-learning-models-performance-230449055f15

Patel, M. (2020). *The Ethics of AI: AI in the Financial Services Sector: Grand Opportunities and Great Challenges*. https://thefintechtimes.com/the-ethics-of-ai-ai-in-the-financial-services-sector-grand-opportunities-and-great-challenges/

Rabinovich, Y. (2020). *Loan Prediction Dataset Machine Learning Project Dataset [Data Set]*. https://www.kaggle.com/code/yonatanrabinovich/loan-prediction-dataset-ml-project/notebook

Shinde, S. R., & Aphale, A. S. (2020). Predict Loan Approval in Banking System: Machine Learning Approach for Cooperative Banks Loan Approval. *International Journal of Engineering Research and Technology (IJERT), 9,* 991-995.

Singh, A. (2019). *R-Squared vs Adjusted R-Squared*. Medium. https://medium.com/analytics-vidhya/r-squared-vs-adjusted-r-squared-a3ebc565677b

Swapna, M. (2018). *Ethics of Using AI in the Financial/Banking Industry*. Medium. https://medium.datadriveninvestor.com/ethics-of-using-ai-in-the-financial-banking-industry-fa93203f6f25

Willems, K. (2019). *Keras Tutorial: Deep Learning in Python*. https://www.datacamp.com/tutorial/deep-learning-python

List of Abbreviations

AI: Artificial Intelligence
ANN: Artificial Neural Network
CNNs: Convolutional Neural Networks
eBooks: Electronic books
DL: Deep Learning
EDA: Exploratory Data Analysis
GPU: Graphics Processing Unit
IDE: Integrated Development Environment
ML: Machine Learning
R^2: R-Squared or the Coefficient of Determination
No.: Number
Relu: Rectified Linear Unit
ROC Curve: Receiver Operating Characteristic Curve
sklearn: Scikit-Learn
SMOTE: Synthetic Minority Over-Sampling Technique
Tanh: Hyperbolic Tangent
USD: United States Dollar

Appendices

A.1. Modified Features Calculation

These modified features are calculated from our existing features which is amounted to 11 features of which we've created an additional 12 features, to help us on predicting a better output based on given label variables.

- **Formula for Combined application income**

Combined application income = application income/Coapplication Income

- **Formula for EMI (EMI stands for Equated Monthly Installment)**

$$\text{EMI} = P^* [r^* (1 + r)^n/(1 + r)^n - 1]$$

where:

P = Principal amount borrowed

r = Periodic monthly interest rate

n = Total number of monthly payments

- **Formula for Loan per Income**

Loan per Income = Loan Amount/combined application income

- **Formula for dependents mean per EMI**

Basically it is the dependents mean of the total applicants

- **Formula for EMI per Term**

EMI per Term = EMI/Loan amount term

- **Formula for Loan amount term per Income**

Loan amount term per Income = Loan Amount Term/combined application income

- **Formula for EMI per loan amount**

EMI per loan amount = EMI/Loan Amount

• **Formula for dependents loan sum**

Basically it is Dependents sum of loan amount wise

• **Formula for Credit history Income sum**

Credit history sum of total income

• **Formula for Property area Loan amount per total Income**

Categorical variables wise mean of loan amount per total income

• **Formula for loan amount term binning & Formula for Total Income binning**

Sklearn library were used to get these information by using advance binning techniques

A.2. R-Squared and Adjusted R-Squared

Formula for **R-Squared** is mentioned below:

$$R^2 = 1 - \frac{\sum\left(y_i - \hat{y}_i\right)^2}{\sum\left(y_i - \bar{y}_i\right)^2}$$

where:

R-squared = 1 − (sum of squares residuals error (SSE)/sum of squares total (SST))

Formula for **Adjusted R-Squared** are mentioned below:

$$\text{Adjusted } R^2 = 1 - \frac{\left(1 - R^2\right)\left(N - 1\right)}{N - p - 1}$$

where:

N = number of records in the data set.

p = number of independent variables.

A.3. Accuracy, Precision and Recall Formula

Accuracy

Accuracy is dividing the total number of correct predictions by all the predictions.

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN}$$

where:

TP: is True Positive instances

TN: is True Negative instances

FP: is False Positive instances

FN: is False Negative instances

Recall

The recall is the measure to check correctly positive predicted outcomes out of the total number of positive outcomes.

$$\text{Recall} = \frac{TP}{TP + FN}$$

Precision

Precision checks how many outcomes are actually positive outcomes out of the total positively predicted outcomes.

$$\text{Precision} = \frac{TP}{TP + FP}$$

F1 score

It is the harmonic mean of recall and precision. Both of these metrics are used to calculate the F1 score, F1 score ranges between 0% to 100% and higher the F1 score value means a better quality classifier. Moreover, a model does well in F1 score if the positive predicted are actually positives (precision) and doesn't miss out on positives and predicts them negative (recall).

$$\frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{recall}}$$

ROC Curve

ROC stands for receiver operating is a graphical plot that demonstrates the analytical ability of a binary classifier system as its discrimination threshold is varied.

$$\text{True Position Rate}(TPR) = \text{Recall} = \frac{TP}{TP + FN}$$

$$\text{False Position Rate}(FPR) = 1 - \text{Specificity} = \frac{TP}{TP + FP}$$

A.4. List of Activation Functions

There are many activation function for neural network and it's a critical task to choose the most suitable one for your project, however for this project I've focused on three of them, and that is Sigmoid, Relu and Softmax.

Sigmoid Activation Function

The sigmoid function is a non-linear activation function used primarily in feedforward neural networks. It is a differentiable real function, defined for real input values, and containing positive derivatives everywhere with a specific degree of smoothness. The sigmoid function appears in the output layer of the deep learning models and is used for predicting probability-based outputs. The sigmoid function is represented as:

$$f(x) = \frac{1}{\left(1 + \exp^{-x}\right)}$$

Softmax Function

The softmax function is another activation function type used in neural networks to compute probability distribution from a vector of real numbers. This

function generates an output that ranges between values 0 and 1 and with the sum of the probabilities being equal to 1. The softmax function is represented as follows:

$$f\left(x_i\right)=\frac{\exp\left(x_i\right)}{\sum_j \exp\left(x_j\right)}$$

Rectified Linear Unit (ReLU) Function

One of the most popular activation function in deep learning models, the rectified linear unit (ReLU) function, is a fast-learning AF that promises to deliver state-of-the-art performance with stellar results. Compared to other AFs like the sigmoid and tanh functions, the ReLU function offers much better performance and generalization in deep learning. The function is a nearly linear function that retains the properties of linear models, which makes them easy to optimize with gradient-descent methods.

The ReLU function performs a threshold operation on each input element where all values less than zero are set to zero. Thus, the ReLU is represented as:

$$f\left(x\right)=\max\left(0,x\right)=\begin{cases}x, & \text{if } x\geq 0\\ 0, & \text{if } x<0\end{cases}$$