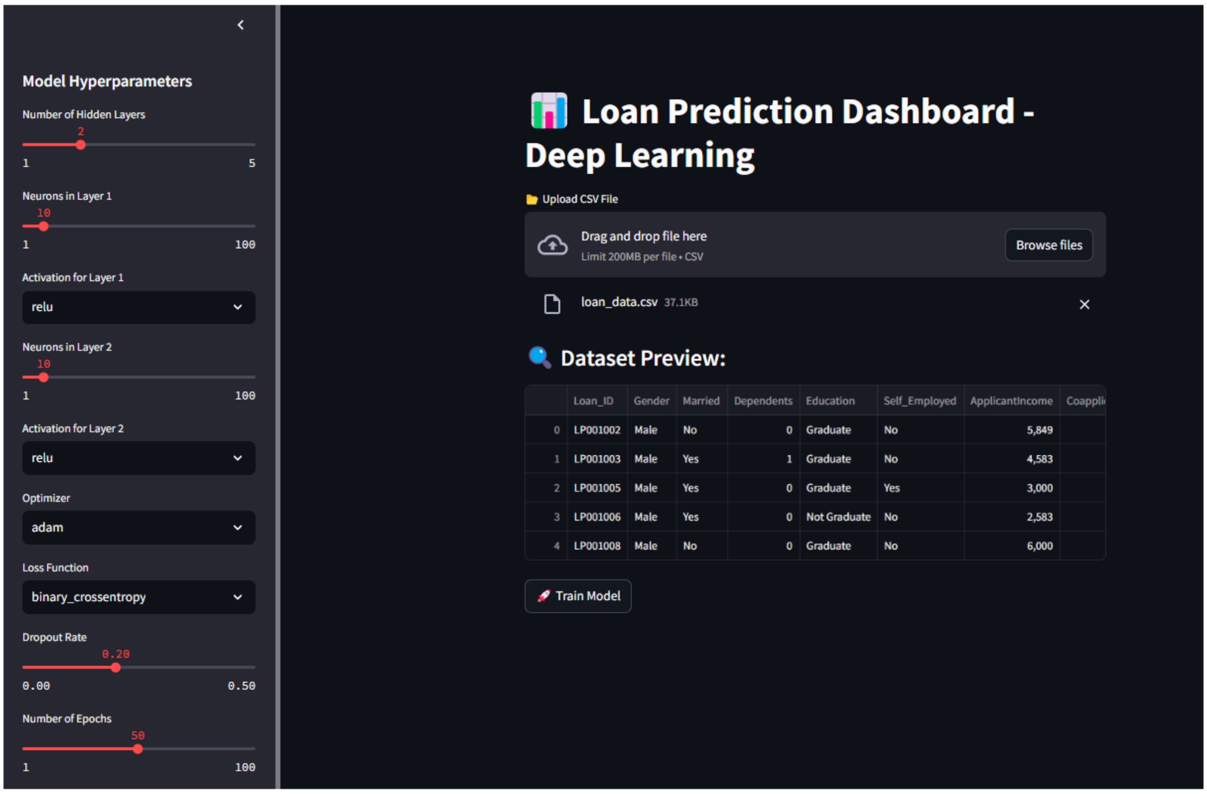
Loan Prediction Using Artificial Neural Networks (ANN) - Dashboard Report

Anjala 055005 | Itighne Sinha 055015

1. Introduction

This project involves the development of a **Loan Prediction Dashboard** using **Streamlit** and an **Artificial Neural Network (ANN)**. The dashboard allows users to upload a dataset and train a deep learning model to predict loan approvals based on various applicant attributes. The model is interactive, enabling users to adjust hyperparameters and visualize training performance.



Model Hyperparameters

Number of Hidden Layers: 2 (1–5)

Neurons in Layer 1: 10 (1–100)

Activation for Layer 1: relu

Neurons in Layer 2: 10 (1–100)

Activation for Layer 2: relu

Optimizer: adam

Loss Function: binary_crossentropy

Dropout Rate: 0.20 (0.00–0.50)

Number of Epochs: 50 (1–100)

📊 Loan Prediction Dashboard - Deep Learning

📁 Upload CSV File

Drag and drop file here
Limit 200MB per file • CSV

loan_data.csv 37.1KB

🔍 Dataset Preview:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Coappli
0	LP001002	Male	No	0	Graduate	No	5,849	
1	LP001003	Male	Yes	1	Graduate	No	4,583	
2	LP001005	Male	Yes	0	Graduate	Yes	3,000	
3	LP001006	Male	Yes	0	Not Graduate	No	2,583	
4	LP001008	Male	No	0	Graduate	No	6,000	

2. Dataset Overview

The dataset consists of loan applicant information with the following features:

- **Loan_ID** (Unique identifier, removed during preprocessing)
- **Gender** (Male/Female)
- **Married** (Yes/No)

- **Dependents** (0,1,2,3+)
- **Education** (Graduate/Not Graduate)
- **Self_Employed** (Yes/No)
- **ApplicantIncome** (Numeric)
- **CoapplicantIncome** (Numeric)
- **LoanAmount** (Numeric)
- **Loan_Amount_Term** (Loan repayment duration)
- **Credit_History** (Creditworthiness indicator)
- **Property_Area** (Urban/Semiurban/Rural)
- **Loan_Status** (Approved or Not)

3. Preprocessing Steps

- **Removing Unnecessary Columns**: Dropped `Loan_ID` as it has no predictive value.
- **Handling Missing Values**:
  - Categorical features filled with mode.
  - Loan amount filled with mean.
- **Encoding Categorical Variables**:
  - Used numerical mappings for features like Gender, Married, Education, etc.
- **Feature Scaling**:
  - Normalized `ApplicantIncome`, `CoapplicantIncome`, and `LoanAmount`.
- **Train-Test Split**:
  - **80% Training**, **20% Testing** (Stratified sampling applied).
  - Standardized numeric features using `StandardScaler`.

4. ANN Model Architecture

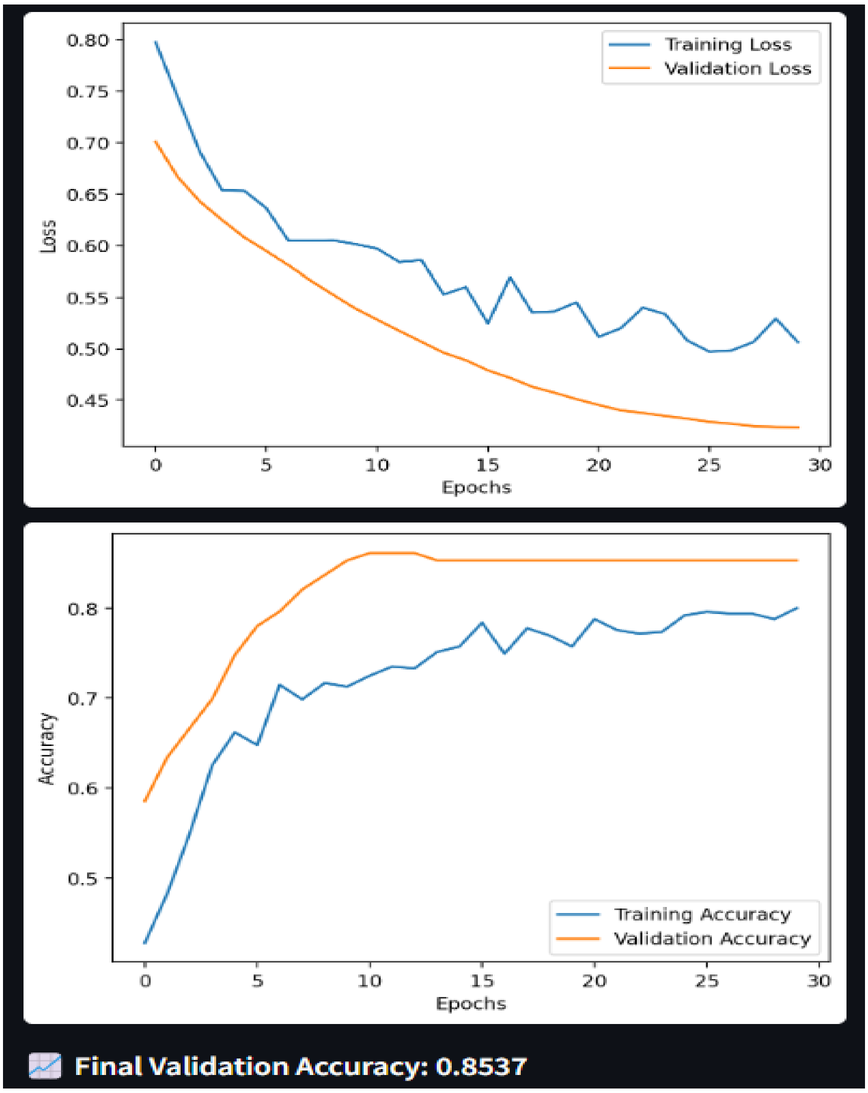
- **Input Layer**: Defined based on the number of features.
- **Hidden Layers**:
  - User-defined **number of layers (1-5)**
  - Customizable **neurons per layer**
  - Selectable **activation functions (ReLU, Tanh, Sigmoid)**
  - **Dropout layers** to prevent overfitting.
- **Output Layer**:
  - Single neuron with **sigmoid activation** (Binary classification: Loan Approved or Not).
- **Loss Functions**:
  - User can select from `binary_crossentropy`, `mean_squared_error`, or `hinge`.
- **Optimizers**:
  - `adam`, `sgd`, `rmsprop`.
- **Hyperparameters**:
  - User can adjust **dropout rate**, **epochs**, and **batch size**.

5. Model Training and Evaluation

- **Training the Model**:
  - ANN model is trained on user-defined settings.
  - Uses validation data (`X_test`, `y_test`) to monitor performance.
- **Performance Visualization**:
  - Loss curve (Training vs Validation Loss).
  - Accuracy curve (Training vs Validation Accuracy).
- **Final Model Evaluation**:
  - Displays **Final Validation Accuracy** after training.
  - Uses model evaluation metrics to measure performance.

6. Experiments

Configuration 1:

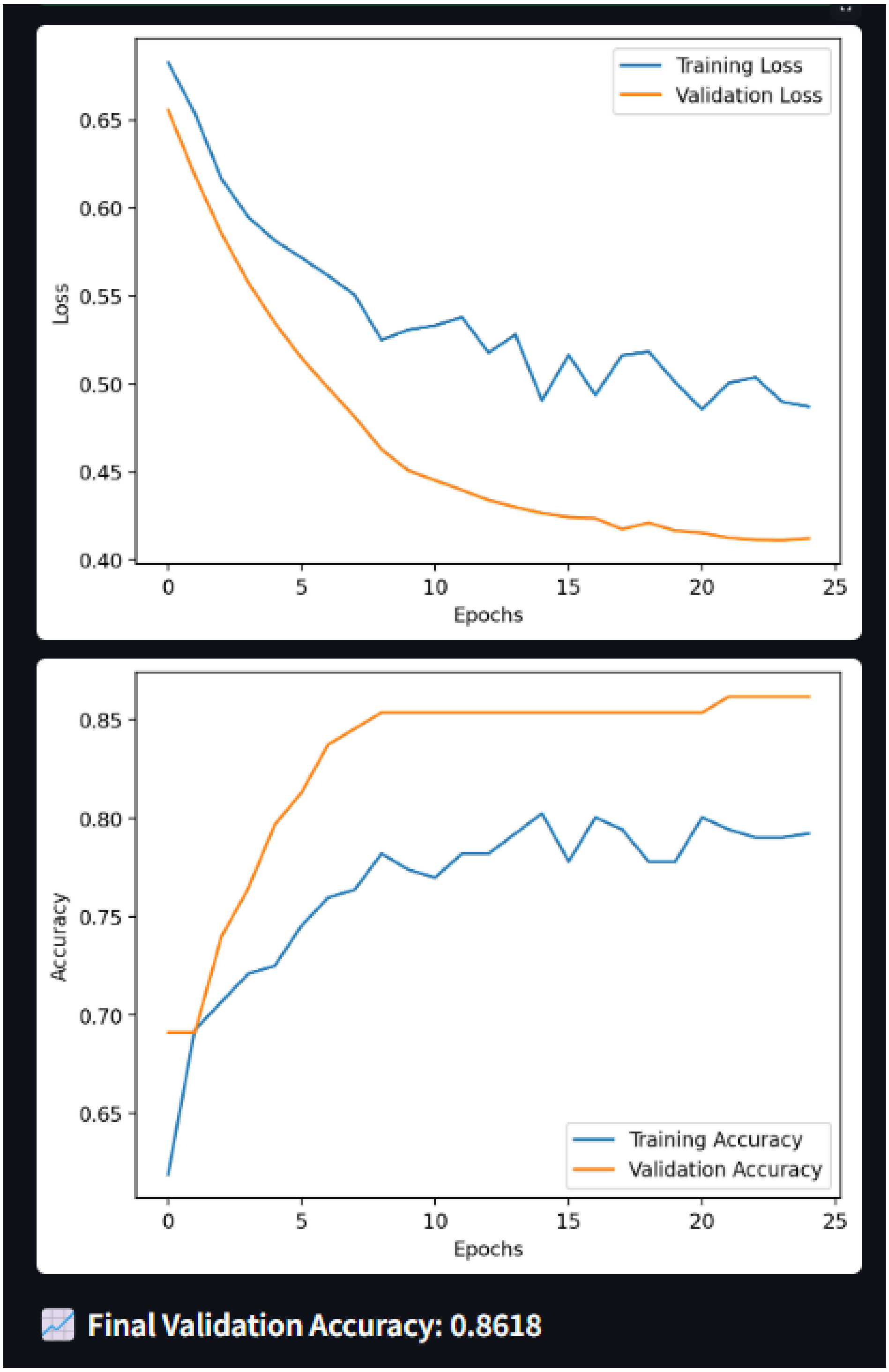


📈 Final Validation Accuracy: 0.8537

Parameters: 2 Hidden Layers (16, 16 neurons), ReLU, Adam, Binary Crossentropy, Dropout 0.2, 30 Epochs

Result: Validation accuracy of 0.8537. Good learning curves, some minor overfitting.

Configuration 2:

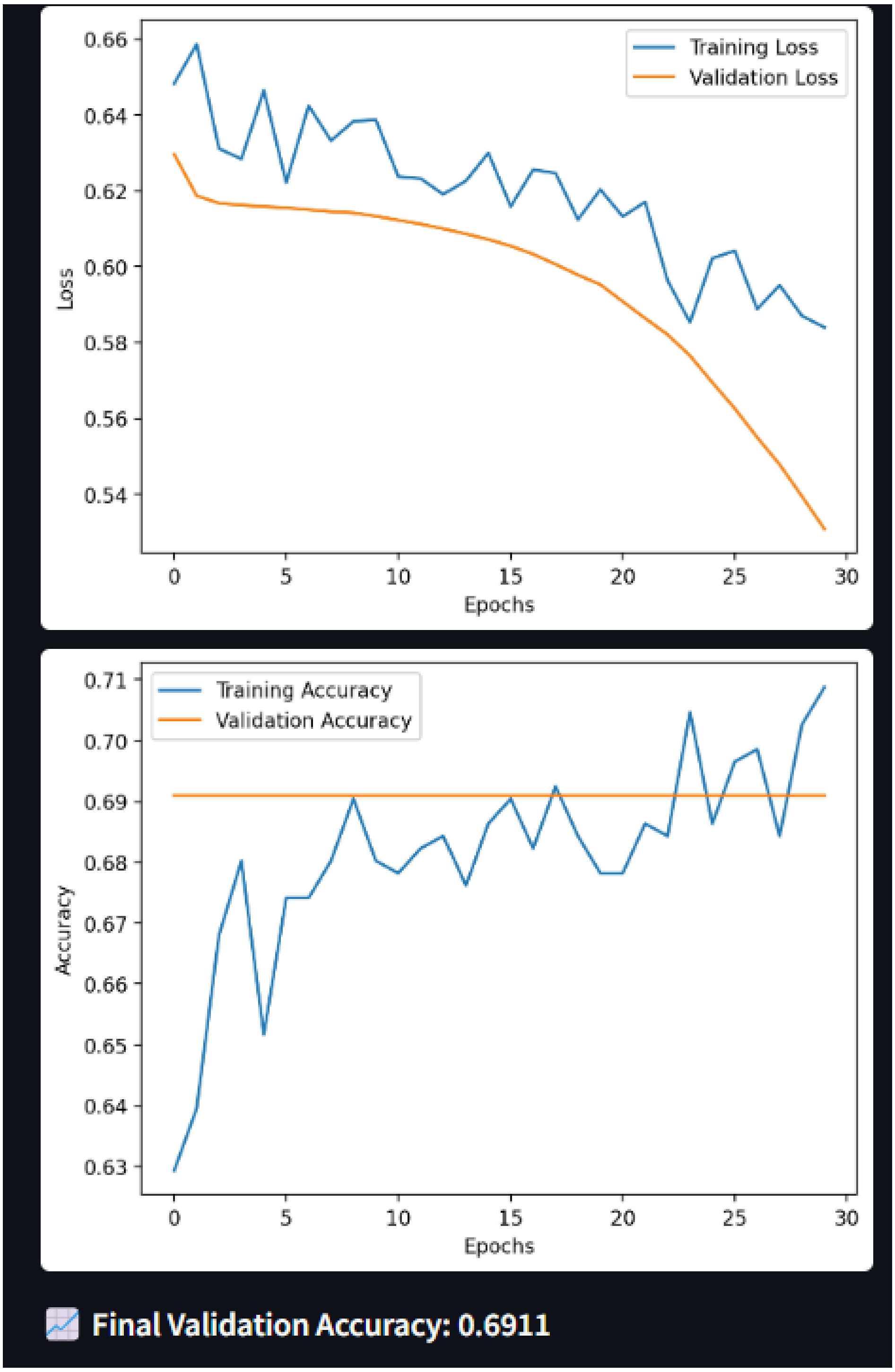


📈 Final Validation Accuracy: 0.8618

Parameters: 3 Hidden Layers (32, 32, 16 neurons), ReLU, Adam, Binary Crossentropy, Dropout 0.2, 25 Epochs

Result: Validation accuracy of 0.8618. Highest accuracy achieved, but more pronounced overfitting.

Configuration 3:

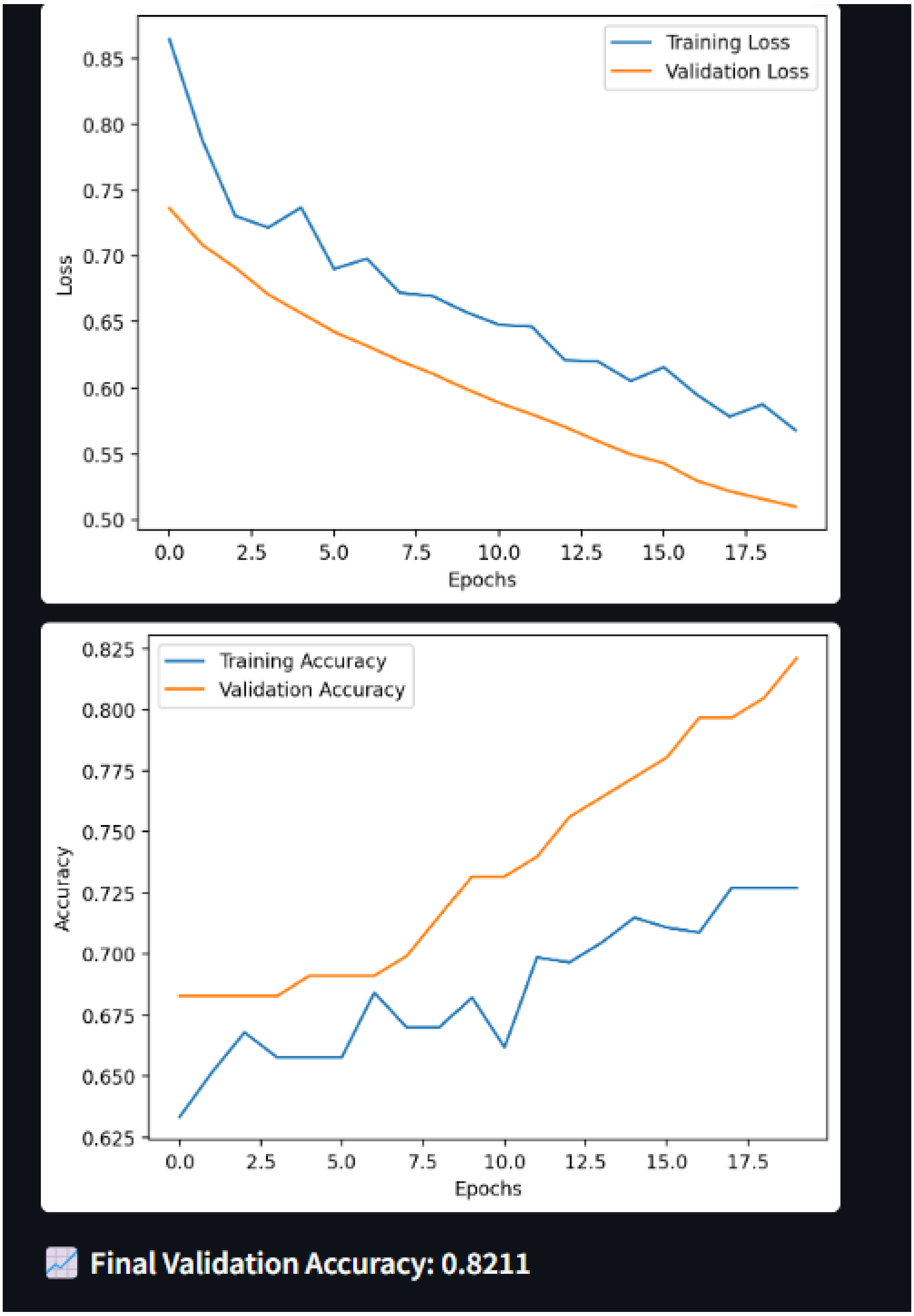


📈 **Final Validation Accuracy: 0.6911**

Parameters: 4 Hidden Layers (32, 32, 16, 16 neurons), Sigmoid, Adam, Binary Crossentropy, Dropout 0.2, 30 Epochs

Result: Validation accuracy of 0.6911. Significantly lower accuracy, poor learning.

Configuration 4:



📈 **Final Validation Accuracy: 0.8211**

Parameters: 2 Hidden Layers (16, 16 neurons), ReLU, RMSprop, Binary Crossentropy, Dropout 0.3, 20 Epochs

Result: Validation accuracy of 0.8211. Underfitting, validation accuracy higher than training accuracy.

Overall Interpretations:

- ReLU is Superior: ReLU activation functions consistently outperformed Sigmoid. Sigmoid in Configuration 3 led to very poor results, indicating it might not be suitable for this type of problem.
- Overfitting is a Key Challenge: Several configurations exhibited signs of overfitting, where the model performs better on the training data than on the validation data. This is evident from the increasing divergence of training and validation accuracy/loss curves. Dropout was used to combat this.
- More Layers Don't Always Mean Better: Increasing the number of hidden layers didn't consistently improve performance. The 3-layer model achieved the best validation accuracy, but the 4-layer model performed the worst. There is a sweet spot.
- Optimization Matters: The choice of optimizer (Adam vs. RMSprop) influences performance. Adam generally performed well, while RMSprop in Configuration 4 showed signs of underfitting.
- Hyperparameter Tuning is Critical: The results demonstrate the importance of careful hyperparameter tuning. Small changes in parameters can significantly impact model performance.

Key Takeaways:

- The optimal ANN architecture for this loan prediction problem appears to involve ReLU activation functions and a limited number of hidden layers (2-3).
- Overfitting is a significant concern, requiring techniques like dropout to regularize the model.
- Further improvements might be achieved through feature engineering, more refined data preprocessing, and learning rate optimization.

7. Conclusion

Based on our experiments, the best configuration where the model exhibited highest accuracy (0.8618) consisted of the following parameters:

1. **Number of Hidden Layers:** 3
2. **Neurons per Layer:** 32, 32, and 16
3. **Activation Function**: ReLU for all hidden layers
4. **Optimizer:** Adam
5. **Loss Function:** Binary Crossentropy
6. **Dropout Rate:** 0.2
7. **Epochs**: 25

- It is important to note however, that while the model seemed the most fitted in this config, it still had symptoms of overfitting.
- Another interesting observation was that accuracy levels at multiple configuration tended to be similar, we have only added the different ones in our project.
- As it is static data, there seems to be a limit to how well fit the model can be, resulting in these insights.