Masters Programmes: Assignment Cover Sheet

Student Number:	**5530970, 2289197, 5526396, 2038061, 5583974, 5558022, 5538208**
Module Code:	**IB98D0**
Module Title:	**Advanced Data Analysis**
Submission Deadline:	**18 March 2024**
Date Submitted:	**17 March 2024**
Word Count:	**1,998 words**
Number of Pages:	**17 pages**
Question Attempted: *(question number/title, or description of assignment)*	**Group Assignment – Deep Learning**
Have you used Artificial Intelligence (AI) in any part of this assignment?	**No**

Contents

1. Executive Summary

This report explores the application of deep learning in the consumer loan industry, focusing on predicting loan defaults for Lending Club. The report aims to build an automated solution for loan default prediction by constructing a deep-learning model. Data pre-processing covers several crucial steps, including feature selection, visualisation, and missing value imputation. Following this, a deep neural network model is developed using Python, with extensive parameter tuning to mitigate overfitting risks. Performance evaluation relies on critical metrics such as precision, recall, and expected value, particularly emphasising minimising a failure to predict a loan default case. The model provides an automated solution to loan classification and demonstrates significant potential by predicting correctly 95% of the default cases, enabling Lending Club to mitigate risk. By calculating expected loss for defaulted loan cases, the model offers valuable insights into potential financial losses, allowing for more effective recourse allocation and proactive management.

Based on our findings, we recommend implementing the optimised deep learning model to streamline loan approval processes and minimise default risks. While our model demonstrates promising accuracy rates, it is essential to acknowledge its limitations, including computational complexity and lack of transparency. To fully harness its potential, ongoing research and development efforts are imperative. Addressing these limitations will optimise model efficacy, enhance loan prediction processes, and facilitate more informed decision-making within the financial sector.

2. Introduction

In today's fast-paced financial landscape, advanced analytics holds immense value for companies in all fields, especially the consumer loan industry, which relies on accurate risk assessment and efficient loan approvals. Deep learning could deliver a transformative opportunity for lending practices by enabling automatic loan classification thereby reducing the need for manual intervention and enhancing prediction efficiency.

Lending Club, a renowned peer-to-peer lending platform, connects investors seeking returns with borrowers needing capital. However, the company faces challenges in optimising investor returns, minimising default risks, and enhancing customer experience. This report constructs a deep learning model that predicts loan defaults, i.e., bad loans. Subsequently, it evaluates the model's performance, identifies limitations, and proposes potential solutions applicable to real-world implementation.

3. Overview of the Dataset and Data Preparation

3.1. Data Overview

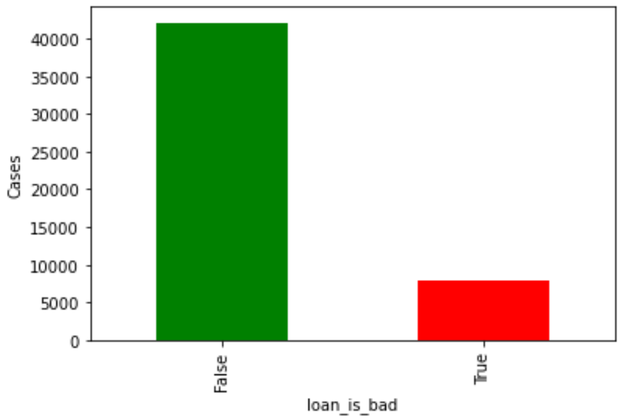


Figure 3.1 The distribution of the target variable in the dataset

Lending Club has provided a dataset comprising 50,000 records extracted from previous loan offerings. The dataset includes 53 explanatory variables, with 'id' and 'member_id' serving as unique identifiers for each record. 'loan_is_bad' represents the target variable denoting the loan's default status. The dataset comprises numerical and categorical data, including ordinal and nominal variables.

Several variables exhibit missing values, notably 'emp_length', 'mths_since_last_delinq', 'mths_since_last_record', 'mths_since_last_major_derog', 'tot_coll_amt', 'tot_cur_bal', and 'total_credit_rv'. Moreover, the target variable, 'loan_is_bad', reveals a substantial imbalance between instances of loan defaults (7,814 cases) and non-default cases (42,186 instances).

3.2. Data Preparation

Feature Selection: We commenced by meticulously examining all variables and their corresponding descriptions. Features deemed irrelevant or non-informative to the credit card default prediction were excluded, including id, member_id, address_state, and other variables. Details on feature selection are in Appendix 1. After feature selection, 24 variables remained.

Data visualisation: We conducted a visualisation process to understand the data distribution better and identify visualisation anomalies. (Figures 3.2 and 3.3)

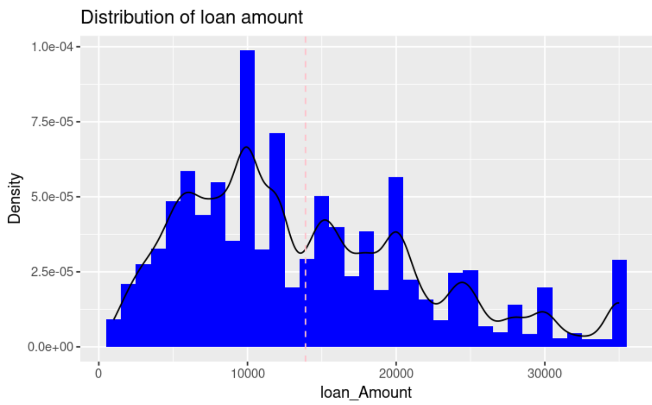
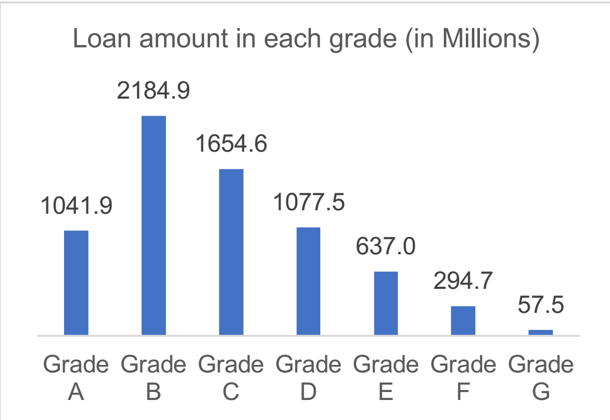


Figure 3.2 Distribution of the loan amount

The average loan amount applied for by the borrower is £13,901.20, with a standard deviation of £8,086.40. Most of the loans (81%) have a duration of 36 months, while the rest (19%) have a duration of 60 months. Loans are categorised into six grades, from 'Grade A' to 'Grade G', with most loans (36%) credited to 'Grade B' (see Figure 3.3).



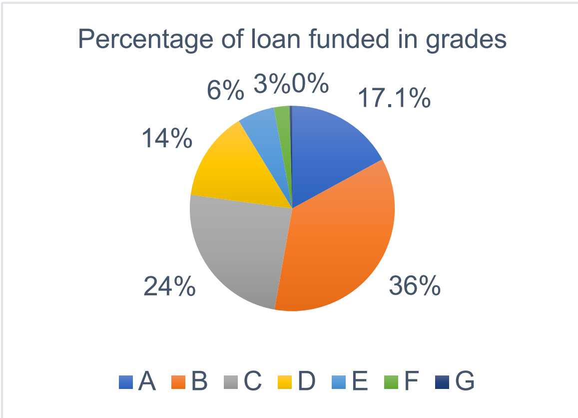


Figure 3.3 Distribution of the number of loans in each grade

Duplicate checks: The data was examined for duplicate values, and none were found.

Missing Value Imputation: We addressed missing values using a strategic approach tailored to the specific variable characteristics. For features that was possible, we employed mode imputation, such as total collection amount (tot_coll_amt) and total current balance (tot_cur_bal), or filled all NA values with zero, such as revolving line utilisation rate (revol_util). Conversely, features exhibiting an unusually high proportion of missing values (>14,000), such as 'mths_since_last_major_derog' and 'total_credit_rv', were excluded from the analysis (see Appendix 1). This decision was made because imputing values in such instances could introduce artificial patterns and distort the actual data distribution, and using mean imputation would not be appropriate.

Data Standardisation: To ensure all numerical features contributed equally to the model and prevent biases caused by different scales, we standardized these values using z-scores. Moreover, the standardised input data can improve the speed of the backpropagation process of the model (Al-Faiz, Ibrahim & Hadi, 2018).

Categorical Feature Encoding: As deep learning models require numerical inputs, we transformed all categorical features into numerical representations using either label encoding or one-hot encoding.

Data Splitting and Imbalance Handling: Finally, we partitioned the pre-processed data into training, validation, and test sets using a 70%, 10%, and 20% split, respectively. Given the class imbalance in the target variable (loan_is_bad), we employed the Synthetic Minority Oversampling Technique (SMOTE) within the training data to address this issue and enhance model performance (Dablain, Krawczyk & Chawla 2023).

4. Modelling

From prior research, a Deep Learning Neural Network demonstrates the best predictive power for predicting loan defaults compared to other models (Li, B. 2022). Therefore, we have chosen to focus on developing a deep-learning NN model for this project.

We have used Python and the 'Keras' and 'TensorFlow' libraries to build our deep neural network. The 'ReLU' activation function was used as well as the sigmoid function to determine the probabilities for the output. Moreover, we selected 'AdamW' as the optimiser and Binary Cross Entropy as the loss function due to the binary classification nature of the problem. The model was trained on the balanced training data, and we tuned various model hyperparameters by assessing precision and recall on the validation dataset. Despite balancing the data, overfitting remained a concern. Consequently, we introduced dropout layers and L2 regularisation to mitigate it, as suggested by the literature (Shunk, 2022).

Our final model consists of an input layer with 24 nodes, followed by a dense layer with 24 nodes, activated by 'ReLU'. Subsequently, there are three more dense layers with 16,8 and 8 nodes, respectively, with the first and third having L2 regularisation with a penalty of 0.001 to prevent overfitting. Finally, the output layer is a single-node dense layer with sigmoid activation for binary classification predictions. The model is compiled using 'AdamW' optimiser over 50 epochs with a learning rate of 0.001 and a weight decay of 0.7. The code can be found in Appendix 5.

5. Evaluation

We evaluated model performance using Precision and Recall values from the confusion matrix. Additionally, we considered the Expected Value, focusing on the False Negative rate since loan defaults have a high cost in the banking industry. We calculated the average loss for each default case and the expected loss for our models (Refer to Appendix 2 for details). The best results of our models are displayed in Table 5.1 below. For the entire table, see Appendix 3.

Model	**Learning Rate**	**No. of Epochs**	**Batch number**	**Drop out layer**	**Precision Value**	**Recall Value**	**Expect Loss**
10	0.001	50	1000	NO	0.83	0.96	£677
11	0.001	50	2000	YES	0.86	0.96	£724
13	0.001	100	1000	YES	0.79	0.97	£560

Table 5.1 The best three performance model on validation dataset

After testing our models' performance with the validation dataset, we found the top-performing models listed above. We prioritized recall as our primary selection criterion as it indicates the model's ability to detect loan default cases accurately, aligning with the company's goal of avoiding misclassification of defaulted loans as good.

The next factor we considered was expected loss. We decide to include more lucid criteria in terms of business. Expected loss shows a possibility that the lender is expected to experience per each customer. After selecting the three models with the best performance from testing on the validation dataset, we further test these models with the test data set. The result from the final testing can be exhibited in Table 5.2 below.

Model	**Learning Rate**	**No. of Epochs**	**Batch number**	**Drop out layer**	**Precision Value**	**Recall Value**	**Expect Loss**
10	0.001	50	1000	NO	0.84	0.96	£818
11	0.001	50	2000	YES	0.89	0.95	£888
13	0.001	100	1000	YES	0.86	0.95	£888

Table 5.2 The comparison among best three models

As observed from the recall value and expected loss, model 10 has the best performance. Therefore, we recommend using these parameter values to develop our model's performance. In Appendix 4, we plot an AUC chart to illustrate the performance of the best three models compared to a random classifier.

6. Limitations and Recommendations

6.1. Limitations

High Cost: Deep learning's time-consuming nature poses a limitation for predictions on large datasets. The model entails multiple hyperparameters, including learning rate, batch size, and epochs, whose optimisation can be time-consuming and costly, necessitating hardware upgrades for efficient execution. In that case, applying such a model may impose significant time and financial costs on the Lending Club, particularly in processing substantial volumes of information. Furthermore, deep learning requires large datasets to effectively learn, which as be challenging and costly to acquire.

Application Accuracy: Overfitting, a notable issue in deep learning, arises from the model's high complexity. Despite excelling in training data, the model may need to improve in new, unseen data predictions, which impairs automatic and transfer learning effectiveness, limiting the model's ability to generalise patterns across different datasets (Tu, 1996). Consequently, this hampers the accurate identification of bad loans and the assessment of transaction risks with new customer data.

Interpretability: Deep learning's lack of interpretability poses a major limitation, especially in finance. Financial institutions, driven by regulations or internal preferences, often favour highly interpretable models (Denadai, 2018). However, as a black box method, it is hard to tell how it arrived at its predictions, nor does it indicate which characteristics are most likely to influence whether a loan defaults.

6.2. Recommendation

Further optimisation of the model should be considered before practical application. Introducing new technologies, such as regularisation and interpretative techniques, can help mitigate the risk of overfitting and enhance model interpretability (Hoblitzell, 2023). Combining deep learning models with traditional statistical or machine learning models can also leverage their strengths and mitigate weaknesses.

Implementing this deep learning model enables Lending Club to identify loans at higher risk of default, streamlining its approval processes by prioritising these cases for closer scrutiny or deciding to raise a higher interest rate to respond to loans that have a chance of default to make sure that the interest amount which we receive from this borrower can cover the risk by using a range of prediction value from '0' to '1'. The closer the prediction value to '0', the higher the interest rate loan bearing the Lender Club should offer.

Furthermore, this model can analyse customer data, transaction history, and behavioural patterns, enabling personalised recommendations that enhance overall customer satisfaction.

7. Conclusion

In response to the need for better operation of Lending Club loan offerings, we developed a deep learning neural network to reliably predict and minimise the likelihood of providing loans that will default in the future. We performed feature selection to exclude unnecessary variables and any future information from our data set. Additionally, we standardised the input data to expedite the model process and encoded all categorical variables. After data preparation, we developed deep neural network models and chose the best model over the validation and test data set by considering the Recall Value and Expected loss.

Although our model has limitations such as high resource consumption, a high possibility of overfitting and lack of interpretability, it can bring many benefits to the firm, such as automation and more effective resource utilization. For future implementation, the company could consider incorporating optimisation techniques, such as using the model’s output to facilitate loan interest rate determination or even exploring the use of this deep learning model for other business activities.

References

Al-Faiz, M. Z., Ibrahim, A. A., & Hadi, S. M. (2018), The effect of Z-Score standardization (normalization) on binary input due the speed of learning in back-propagation neural network, *Iraqi Journal of Information and Communication Technology, 1(3)*, pp. 42-48.

Dablain, D., Krawczyk, B. & Chawla, N.V. 2023, DeepSMOTE: Fusing Deep Learning and SMOTE for Imbalanced Data, *IEEE transaction on neural networks and learning systems,* vol. 34, no. 9, pp. 6390-6404.

Denadai, E. P. (2018). Interpretability of deep learning models. *Towards Data Science*. Available at: https://towardsdatascience.com/interpretability-of-deep-learning-models-9f52e54d72ab (Accessed 12 March 2024).

Hoblitzell, A. (2023, February 10). *Understanding and Debugging Deep Learning Models: Exploring AI Interpretability Methods.* Available at: https://www.infoq.com/articles/deep-learning-models-ai-interpretability-methods/ (Accessed 13 March 2024).

LI, B., 2022. Online Loan Default Prediction Model Based on Deep Learning Neural Network. *Computational Intelligence and Neuroscience : CIN,* 2022.

SHUNK, J., 2022. *Neuron-Specific Dropout: A Deterministic Regularization Technique to Prevent Neural Networks from Overfitting & Reduce Dependence on Large Training Samples.* Ithaca.

Tu, J.V. 1996, Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes, *Journal of clinical epidemiology,* vol. 49, no. 11, pp. 1225.

Appendix

Appendix 1: Information for feature selection

Feature Dropped	Reason
Id	Non-informative
Member_id	Non-informative
installment	Non-informative
sub-grade	Parent classification is considered
emp_title	Non-informative
emp_length	Non-informative
home_ownership	Categorical variable leading to overfitting issue
issue_d	Non-informative
loan_status	Future value – Causing overfitting
pymnt_plan	Most of the values are ‘n’
desc	Non-informative
purpose	Non-informative
title	Non-informative
zip_code	Non-informative
addr_state	Non-informative
earliest_cr_line	Non-informative
inq_last_6mths	Non-informative
mths_since_last_delinq	28126 NA Values
mths_since_last_record	47468 NA Values
total_rec_late_fee	Non-informative
recoveries	Non-informative
collection_recovery_fee	Non-informative
last_pymnt_d	Non-informative
next_pymnt_d	Non-informative
last_credit_pull_d	Non-informative
collections_12_mths_ex_med	Same statistical value - 0
mths_since_last_major_derog	42880 NA Values
policy_code	Same statistical value - 1
acc_now_delinq	Most of the values are 0
total_credit_rv	14618 NA values

Appendix 2: Calculation of Expected loss

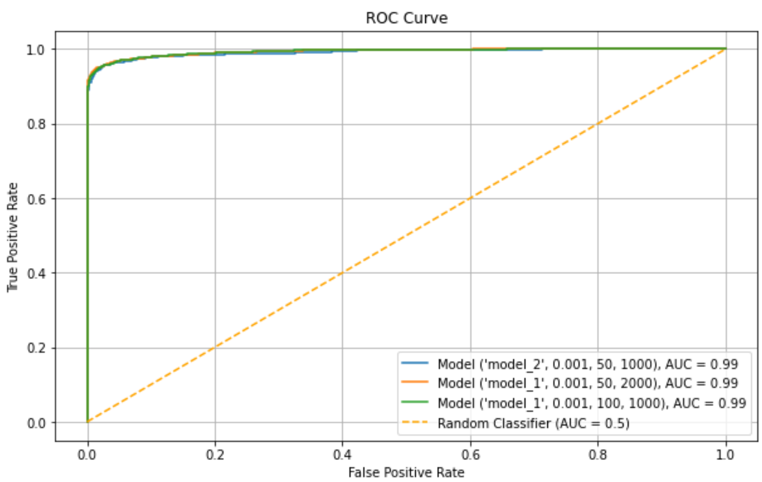
$$Expected\ loss = \frac{False\ Negative}{Total\ number\ of\ Predictions} * 166{,}847$$

*116,847 is the average of the current loan balance of all loan default case (From data set)

Appendix 3: Comparison among Models performance over validation data set

Model	Learning Rate	No. of Epochs	Batch number	Drop out layer	Precision Value	Recall Value	Expect Loss*
1	0.01	50	1000	YES	0.92	0.91	£1,730
2	0.01	50	1000	NO	0.91	0.93	£1,215
3	0.01	50	2000	YES	0.84	0.95	£935
4	0.01	50	2000	NO	0.92	0.93	£1,239
5	0.01	100	1000	YES	0.88	0.93	£1,355
6	0.01	100	1000	NO	0.93	0.93	£1,379
7	0.01	100	2000	YES	0.83	0.95	£911
8	0.01	100	2000	NO	0.82	0.94	£1,122
9	0.001	50	1000	YES	0.91	0.95	£934
10	0.001	50	1000	NO	0.83	0.96	£677
11	0.001	50	2000	YES	0.86	0.96	£724
12	0.001	50	2000	NO	0.89	0.95	£911
13	0.001	100	1000	YES	0.79	0.97	£560
14	0.001	100	1000	NO	0.88	0.95	£911
15	0.001	100	2000	YES	0.87	0.96	£818
16	0.001	100	2000	NO	0.89	0.95	£981

Appendix 4: AUC chart of the top three model performance over test data set



Appendix 5: Python code

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from imblearn.over_sampling import SMOTE
from imblearn.combine import SMOTEENN
from imblearn.under_sampling import RandomUnderSampler
from sklearn.preprocessing import LabelEncoder, StandardScaler
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras import regularizers
from sklearn.metrics import precision_score, recall_score, confusion_matrix

# Set seed for reproducibility
np.random.seed(123)
tf.random.set_seed(123)

# Read data
loan_data = pd.read_csv("loan_data_ADA_assignment.csv")
loan_data = loan_data.replace(r'^\s*$', np.nan, regex=True)

# Drop unnessesary columns
loan_data.drop(columns=['id', 'member_id', 'installment','sub_grade', 'emp_title',
                    'emp_length', 'issue_d', 'loan_status','pymnt_plan', 'desc',
                    'purpose', 'title', 'zip_code','addr_state', 'inq_last_6mths',
                    'earliest_cr_line', 'mths_since_last_delinq',
                    'mths_since_last_record', 'total_rec_late_fee', 'recoveries',
                    'collection_recovery_fee', 'last_pymnt_d', 'policy_code',
                    'last_credit_pull_d', 'acc_now_delinq', 'next_pymnt_d',
                    'home_ownership', 'collections_12_mths_ex_med',
                    'mths_since_last_major_derog','total_credit_rv'], inplace=True)
```

```
#Check for duplicate observations
duplicates = loan_data.duplicated().sum()
print(duplicates)

# Label encoding for the grade variable
loan_data['grade'] = loan_data['grade'].map({'A': 1, 'B': 2, 'C': 3, 'D': 4,
                                           'E': 5, 'F': 6, 'G': 7})

# One-hot encoding for categorical variables, in this case only verification_status
loan_data = pd.get_dummies(loan_data, columns=['verification_status'])

# Handle missing values by replacing with mode
loan_data['tot_coll_amt'].fillna(loan_data['tot_coll_amt'].mode()[0], inplace=True)
loan_data['tot_cur_bal'].fillna(loan_data['tot_cur_bal'].mode()[0], inplace=True)

# Fill missing values with 0, convert and replac % for 'revol_util' variable
loan_data['revol_util'] = loan_data['revol_util'].fillna(0)
loan_data['revol_util'] = loan_data['revol_util'].astype(str).str.replace('%', '').astype(float)
loan_data['revol_util'] = loan_data['revol_util'].astype(str).str.replace('%',
'').astype(float).astype(int)

#Columns to standardize
columns_to_standardize = [
   'loan_amnt', 'funded_amnt', 'funded_amnt_inv','int_rate',
   'annual_inc', 'dti', 'revol_bal', 'total_pymnt', 'total_pymnt_inv',
   'total_rec_prncp', 'total_rec_int', 'last_pymnt_amnt', 'tot_cur_bal',
   'term', 'open_acc', 'revol_util', 'total_acc', 'tot_coll_amt'
]

# Initialize the StandardScaler
scaler = StandardScaler()
```

```
# Loop through columns and replace each one with standardized values
# Overwrite original column with standardized values
for col in columns_to_standardize:
    col_values = loan_data[[col]]
    standardized_values = scaler.fit_transform(col_values)
    loan_data[col] = standardized_values.flatten()

# Split data into training and test sets
X = loan_data.drop(columns=['loan_is_bad']).astype('float')
y = loan_data['loan_is_bad'].astype('float')

# Encoding target variable
#label_encoder = LabelEncoder()
#y = label_encoder.fit_transform(y)

# Split data into training and test sets (70% training, 10% validation and 20% testing)
X_train_val, X_test, y_train_val, y_test = train_test_split(X, y, test_size=0.2,
random_state=123)
X_train, X_val, y_train, y_val = train_test_split(X_train_val, y_train_val, test_size=0.125,
random_state=123)

# Apply smoteenn to balance the training data
smote_enn = SMOTEENN(random_state=123)
X_train_balanced, y_train_balanced = smote_enn.fit_resample(X_train, y_train)

# Convert datasets into TensorFlow datasets
train_dataset = tf.data.Dataset.from_tensor_slices((X_train_balanced, y_train_balanced))
val_dataset = tf.data.Dataset.from_tensor_slices((X_val, y_val))
test_dataset = tf.data.Dataset.from_tensor_slices((X_test, y_test))

# Batch datasets
train_batch = train_dataset.batch(1000)
```

```
features, labels = next(iter(train_batch))

#Define the model
model = tf.keras.Sequential([
   tf.keras.layers.Dense(24, activation=tf.nn.relu, input_shape=(24,)),
   tf.keras.layers.Dropout(0.2),
   tf.keras.layers.Dense(16, activation=tf.nn.relu, kernel_regularizer=regularizers.l2(0.001)),
   tf.keras.layers.Dense(16, activation=tf.nn.relu),
   tf.keras.layers.Dense(8,  activation=tf.nn.relu, kernel_regularizer=regularizers.l2(0.001) ),
   tf.keras.layers.Dense(1,  activation='sigmoid')
])

#Complie the model
model.compile(optimizer=tf.keras.optimizers.AdamW(learning_rate=0.001,
weight_decay=0.7),
            loss=tf.keras.losses.BinaryCrossentropy(), #loss function as cross entropy
            metrics=['accuracy', tf.keras.metrics.Precision(), tf.keras.metrics.Recall(),
tf.keras.metrics.FalseNegatives()])

#Model training
model.fit(X_train_balanced, y_train_balanced, epochs=50)

#Cretae validation batches - here we are validating with the whole validation set
val_batch = val_dataset.batch(len(val_dataset))
val_features, val_labels = next(iter(val_batch))

#Use the model to make predictions for the validation dataset
# Convert probabilities to binary predictions using threshold 0.5
y_pred = model.predict(val_features)
y_pred_binary = (y_pred > 0.5).astype(int)

# Compute Precision and Recall and the confusion matrix
precision = precision_score(val_labels, y_pred_binary)
```

```
recall = recall_score(val_labels, y_pred_binary)
conf_matrix = confusion_matrix(val_labels, y_pred_binary)
print('Precision:', precision)
print('Recall:', recall)
print('Confusion Matrix:')
print(conf_matrix)
#Confusion Matrix:
#[[TN  FP]
#[FN  TP]]

#Print the predicted and actual values of the validation set for checking
#for pred, real in zip(y_pred, val_labels):
#   print(f"Predicted: {pred[0]};    Real: {real}")


#After tuning parameters test using test set
test_batch = test_dataset.batch(len(test_dataset))  # the whole dataset
test_features, test_labels = next(iter(test_batch))
y_pred_test = model.predict(test_features)
y_pred_binary_test = (y_pred_test > 0.5).astype(int)
test_labels_flat = np.ravel(test_labels)
precision_test = precision_score(test_labels_flat, y_pred_binary_test)
recall_test = recall_score(test_labels_flat, y_pred_binary_test)
conf_matrix = confusion_matrix(test_labels_flat, y_pred_binary_test)
print('Precision:', precision_test)
print('Recall:', recall_test)
print('Confusion Matrix:')
print(conf_matrix)
```

Appendix 6: Meeting Minutes

- Meeting Date: February 22nd 2024
    - Participants: 5530970, 2289197, 5526396, 2038061, 5583974, 5558022, 5538208
    - Meeting Goal: Introduction and assignment topic confirmation
    - Meeting Agenda
        - Meeting team members
        - Deciding on the assignment topic/question
        - Conduct initial research on deep learning
        - Aligning on overall goals and deadlines
    - Action Points: All group members to work on understanding the topic in assignment brief and conducting some research on deep learning
    - Meeting Date: February 29th 2024
    - Participants: 5530970, 2289197, 5526396, 2038061, 5583974, 5558022, 5538208
    - Meeting Goal: Discussion and allocation of the tasks
    - Meeting Agenda
        - Understanding objective and discussion on how to work on the model
        - Division of tasks (report writing/data preparation/model construction)
        - Setting weekly targets to complete the assignment
    - Action Points: Group members were equally split with 3 members on the report, 2 members on data cleaning & preparation, and rest working on the deep learning model
- Meeting Date: March 2nd 2024
    - Participants: 5530970, 2289197, 5526396, 2038061, 5583974, 5558022, 5538208
    - Meeting Goal: Data Cleaning & Preparation
    - Meeting Agenda
        - Data Preparation and Selection of Variables
        - Defining the process and completing the report summary
    - Action Points: Group members were assigned tasks involving the data cleaning & preparation, or worked on the report
- Meeting Date: March 5th 2024
    - Participants: 5530970, 2289197, 5526396, 2038061, 5583974, 5558022, 5538208
    - Meeting Goal: Progress Update and Review
    - Meeting Agenda

    - Discussion of overall progress and aligning on the report work
    - Reviewing the deep learning model
  - Action Points: All group members were involved in reviewing the progress of the assignment

- Meeting Date: March 12th 2024
  - Participants: 5530970, 2289197, 5526396, 2038061, 5583974, 5558022, 5538208
  - Meeting Goal: Model Evaluation and Discussion of Results
  - Meeting Agenda
    - Evaluating the model and analysing the results
    - Interpretation of the results and suggesting potential changes
  - Action Points: All group members were involved in reviewing the model and evaluating the results

- Meeting Date: March 14th 2024
  - Participants: 5530970, 2289197, 5526396, 2038061, 5583974, 5558022, 5538208
  - Meeting Goal: Reviewing the code and finalizing the report
  - Meeting Agenda
    - Reviewing and fine-tuning the deep learning model's code
    - Completing all remaining sections of the report
  - Action Points: All group members were equally involved in either reviewing the code or finalizing the report work

- Meeting Date: March 16th 2024
  - Participants: 5530970, 2289197, 5526396, 2038061, 5583974, 5558022, 5538208
  - Meeting Goal: Final discussion to wrap-up the assignment
  - Meeting Agenda
    - Going over the report and code as a final check
  - Action Points: All group members were involved in final discussions