**Week 4**

Sandish Khoju Shrestha

Department of IT, Westcliff University

Presidential Graduate School

TECH 405: Artificial Neural Network & Deep Learning

Professor Acharya

November 17, 2024

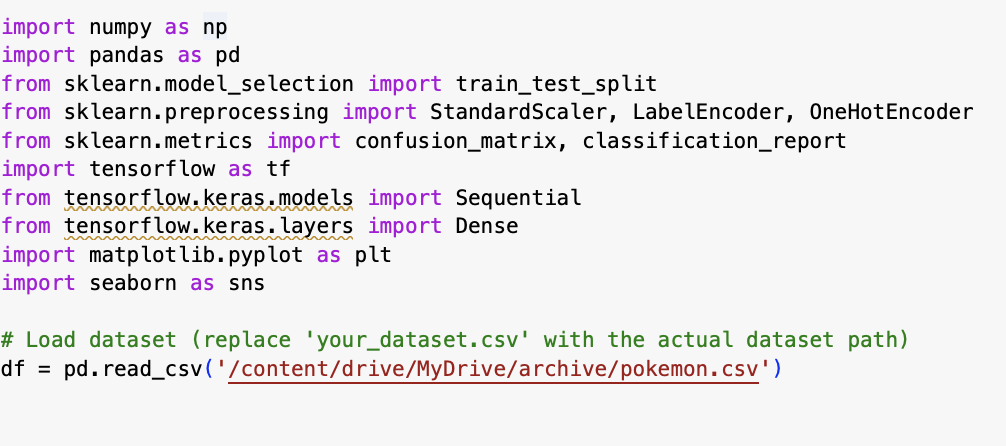
# **Introduction**

# This paper aims at using the artificial neural networks (ANN) with deep learning for binary classification tasks, as proposed in the introduction section. The paper’s goal is simple – to explain how to build a neural network model and how to evaluate its performances; however, it also underlines some preprocessing steps that play crucial roles, such as label encoding and one-hot encoding for categorical data. This talk provides an understanding of how classification problems are solved by using Neural Networks which consist of several layers, using activation functions like ReLU and sigmoid and how performance metrics are improved using methods like the Adam optimizer.

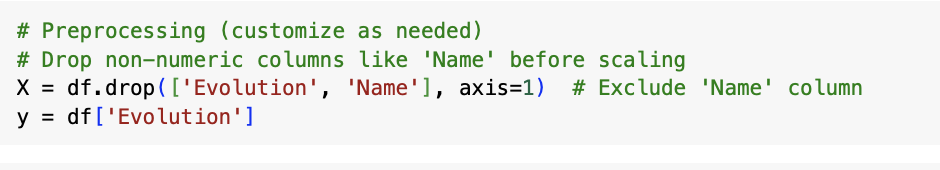
# **Related Work**

# In this context, previous work in the field of neural networks and deep learning is closely linked to different strategies and techniques with the aim of increasing the performance of classification models. Literature has it that several methods like the one-hot encoding and label encoding are used to prepare categorical features for machine learning models. Moreover, a lot of works have been dedicated to construction of neural networks and their architecture, to choice of the activation functions, methods of the models’ optimization and the assessment of their characteristics. Some works call for intermediate layer depth neural networks, while others pay attention to adaptable parameters like the learning rate, batch size and number of epochs. **Methodology**

**Importing the libraries**

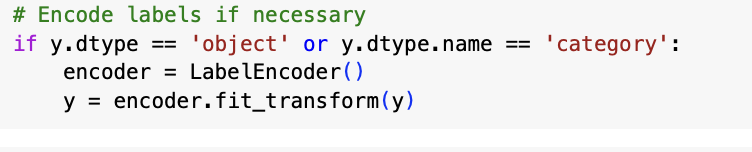


The shared Python code includes the libraries for data analysis and machine learning, the dataset to be used and prepares the work environment needed to create and train a neural network model.

**Processing the images and labels **

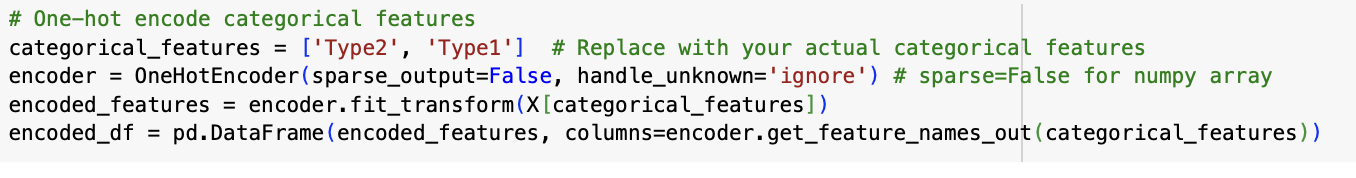
The code chooses numerical attributes for modeling which is represented by ‘X’ and the numerical quantitative variable which is to be predicted is ‘y’.

**Encoding labels**



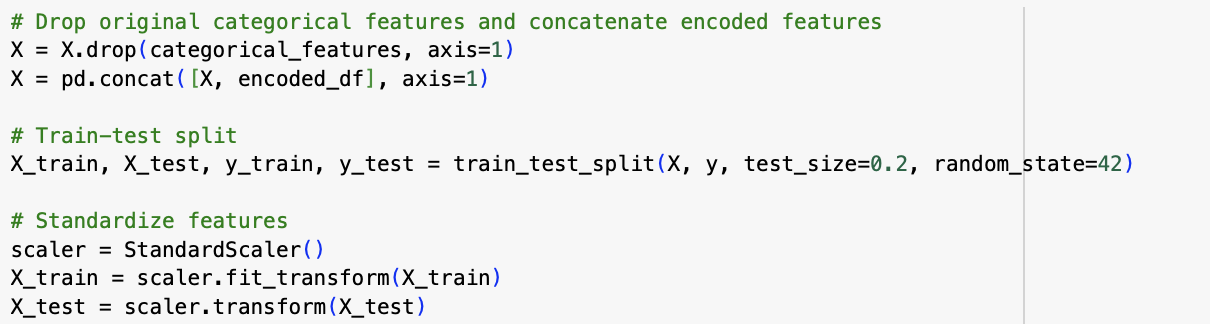
This code checks if the target variable `y` is of type `object` or `category`, and if so, it applies `LabelEncoder` to convert the categorical labels into numerical format. This transformation makes `y` suitable for machine learning algorithms that require numeric input.

**Encode categorical features**

****

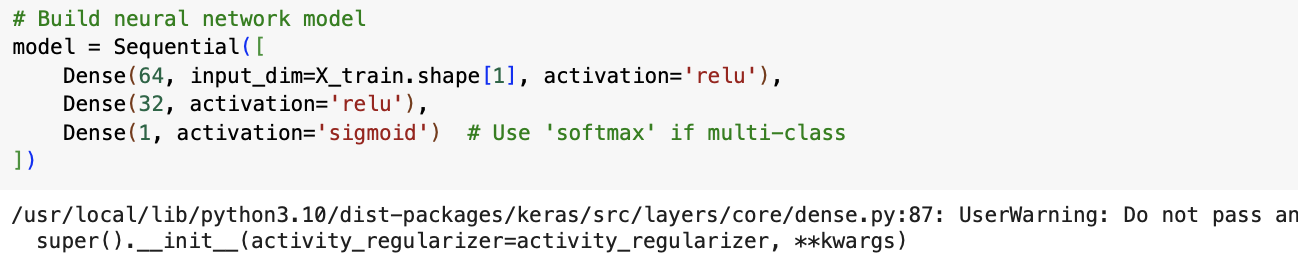
This code transforms the categorical features ‘Type2’ and ‘Type1’ in the DataFrame `X` into a series of binary indicators in a DataFrame `encoded\_df`. It successfully utilizes One Hot Encoder in order to treat unknown categories with appropriate handling, and it returns a dense NumPy array.

**Train-Test Split**



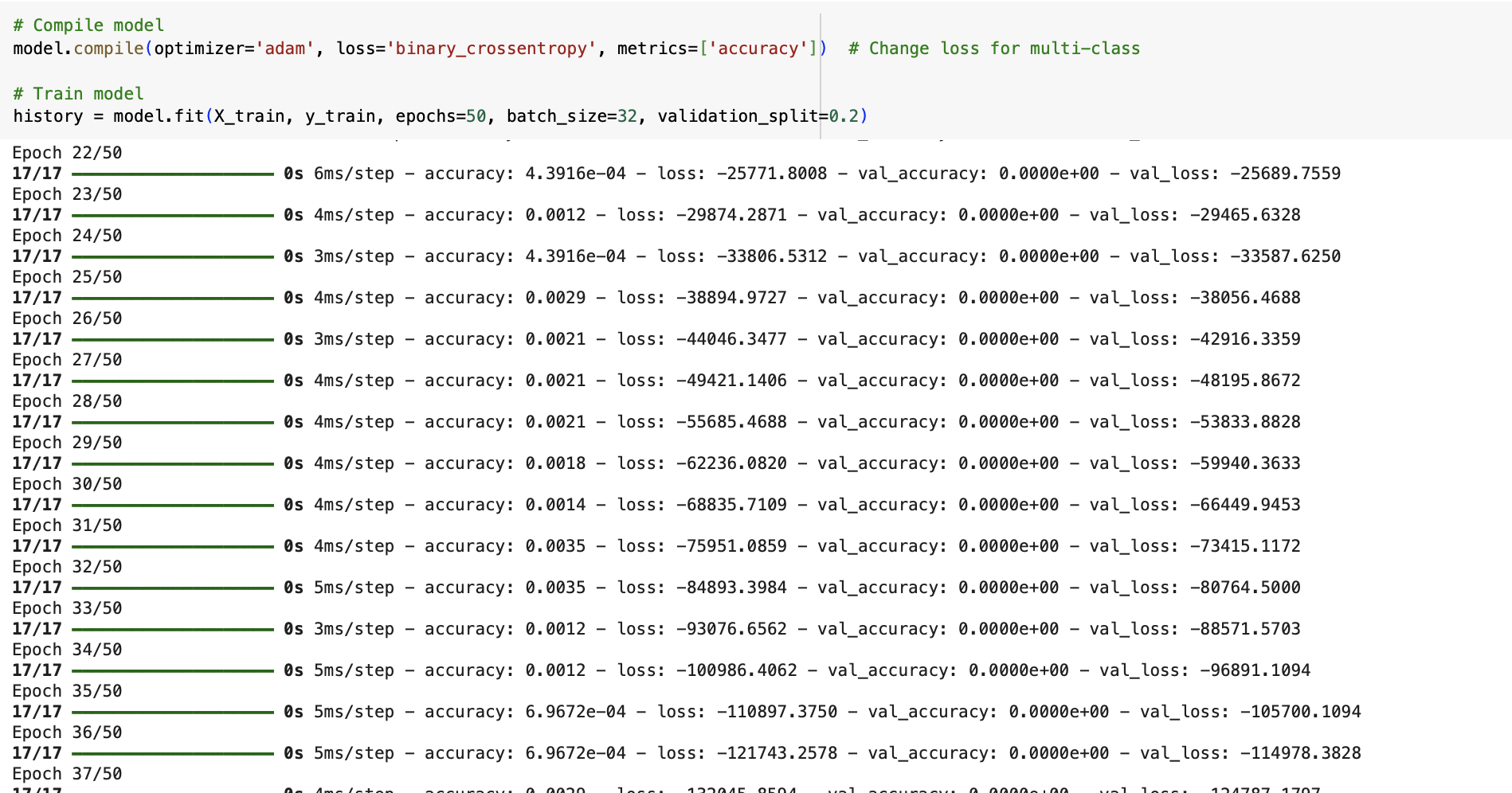
This code removes the original categorical features from X and adds the features that are one hot encoded. It then divides the data into a testing set and a training set and scales the features using StandardScaler to optimise model outcome.

**Building neural network**



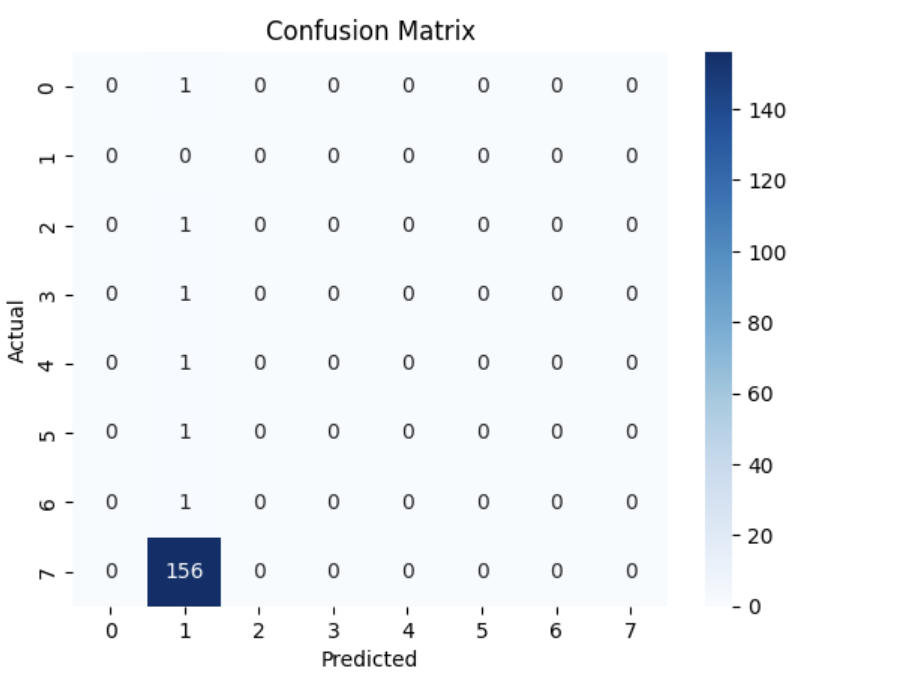
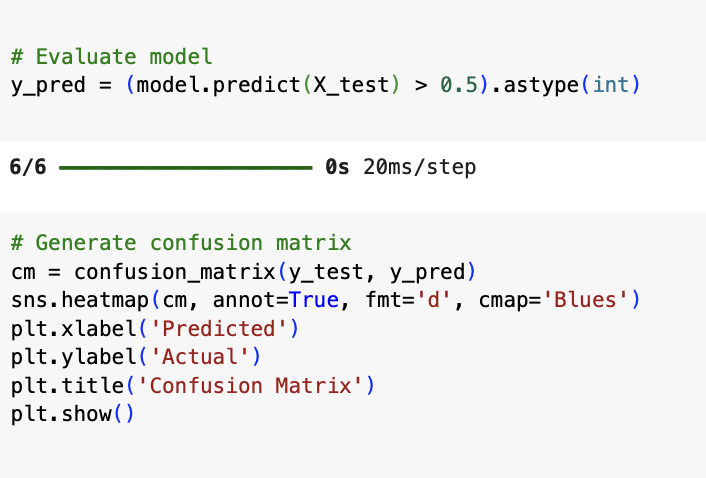
This code builds a neural network model using Keras with three layers: We test the network using an input layer of 64 units, a hidden layer of 32 units, and use logistic sigmoid activation for binary classification task. The model for the hidden layer utilizes ReLU activation.

**Compling and training model**



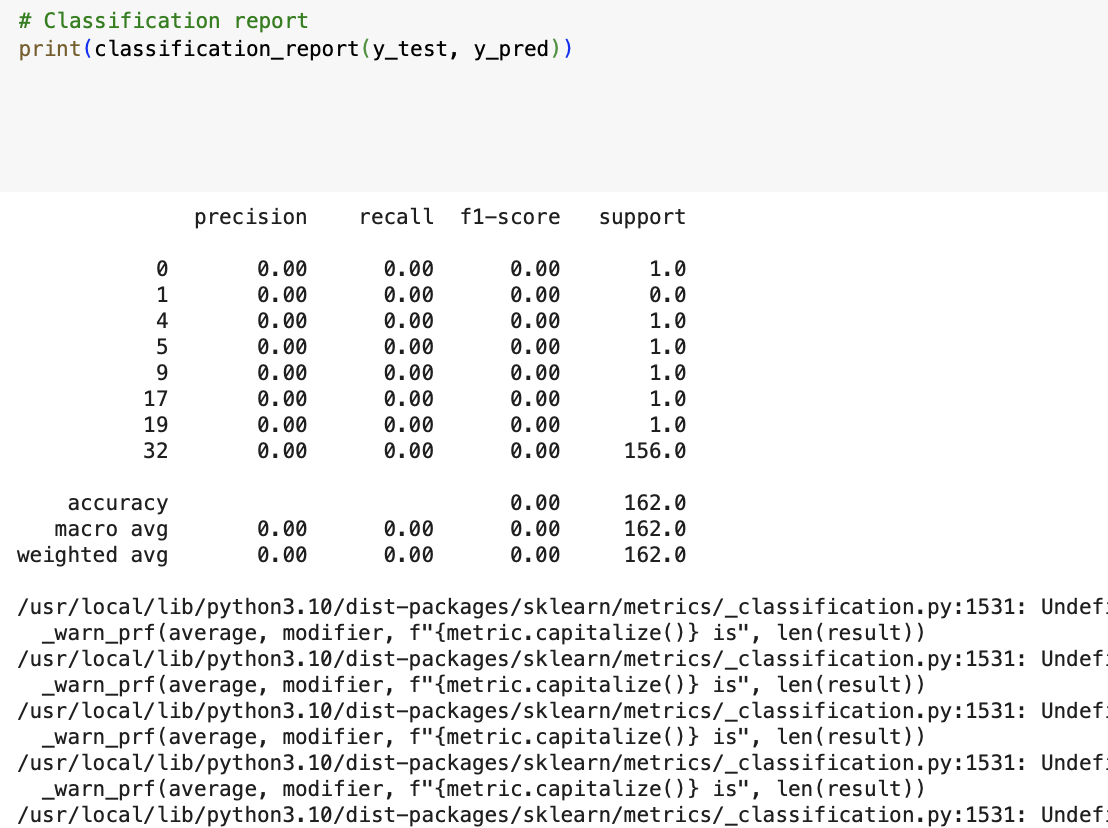
This code builds up the neural network model using Adam optimizer and binary cross-entropy loss function and subsequently fit the training data in to the model for 50 epochs with batch size of the training set defined as 32 and we segregate 20% of the training data for validation.

**Evalutating model and genertaing confusion matrx**

****

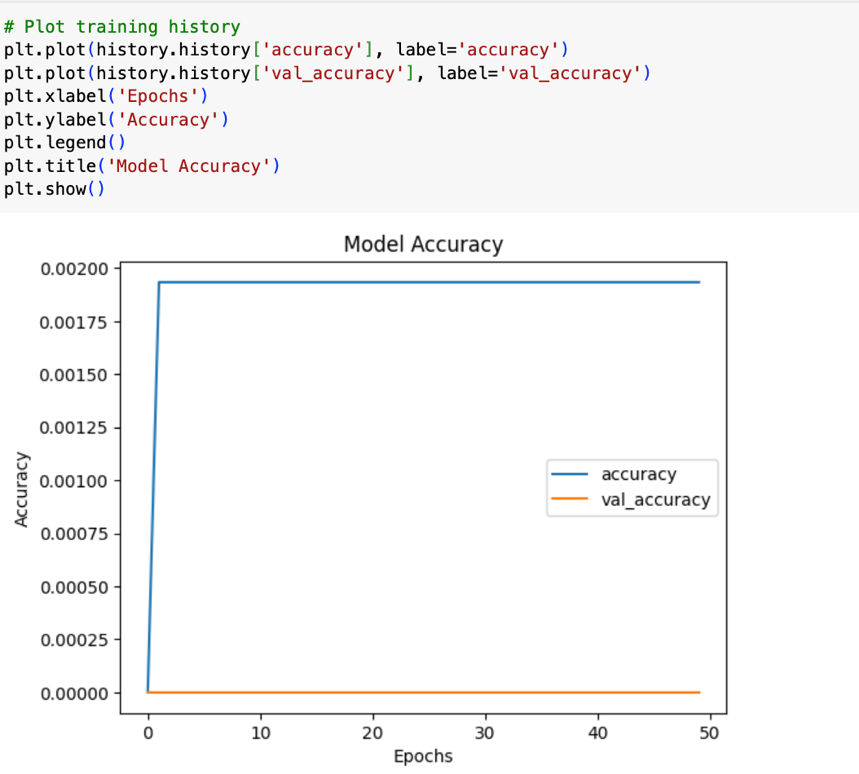
This code fits the model to the test set and then just uses the predict method to make predictions on the test data then converts the continuous output of the model using the threshold of 0.5 to make the output binary. The predictions are then converted to integers either 0 or 1. This code computes visualization of confusion matrix by using the true labels and the predicted labels from (y\_test and y\_pred respectively). It then creates a heatmap of the matrix and labels the axes, as well as gives the heatmap a title for clarity.

**Classification Report**



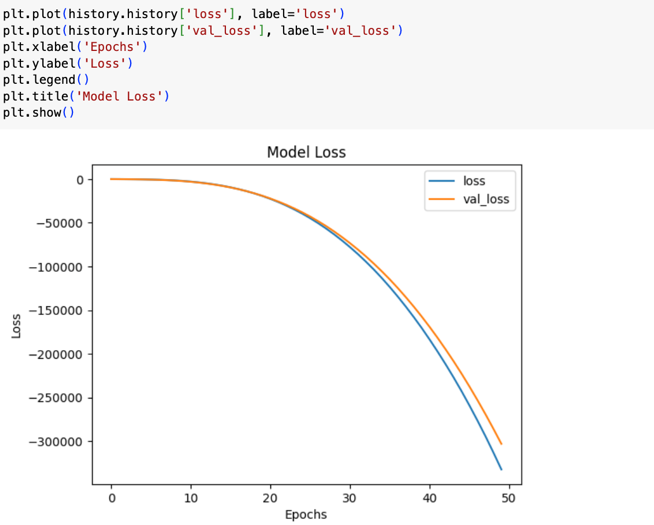
It produces and prints a classification report for binary classification model that demonstrates precision, recall, F1 score and support. It also determines the correctness of the model in the test set (Actual values ‘y\_test’ vs. predicted ‘y\_pred’).

**Plotting Traing history**



This code is to draw the training and validation accuracy from the history of the model. It actually shows a graphical display of how the performance of the model changes or/and increases as training progresses.

**Plotting**



This code shows a figure with the training and validation loss from the history of a model’s training session. It describes the learning process of the model and how it affects the loss in order to make an assessment of either improvement or deterioration.

**Experiment**

# The experiment in this paper aims at developing and testing a neural network model for binary classification. As a first step, the required library files are imported, next, the data set is loaded for the analysis, and the categorical and target variables are encoded by label encoding and one-hot encoding techniques, respectively. The features in the dataset are scaled following which the dataset is divided into a training set and a test set. Neural network is formed with input, hidden and output layers and activation functions involves ReLU and sigmoid activation functions. The model is optimized with Adam optimizer and computed by the binary cross entropy loss function and trained with 50 epochs. After training, the MNIST model predicts outcomes on a test set and assesses it with a confusion matrix, classification report and figures of accuracy and loss across epochs. These steps give one an outlook of training and testing phases of the model as well as its learning ability.

# **Conclusion**

The conclusion of the paper successfully discusses the potentials of artificial neural networks (ANNs) as well as deep learning solutions in binary classification problems. After properly preprocessing and balancing the data through label encoding and one hot encoding the model trains and is then tested. The application of the neural network of one hidden layer with ReLU and sigmoid activation functions and the Adam optimizer and binary cross-entropy loss function proved the possibility of learning and predicting conditions by the model.

Github link:

# **References**

Kaggle Dataset for Pokémon Images**:** <https://www.kaggle.com/datasets/kvpratama/pokemon-images-dataset>

‌