Final Report

1. Personal Information

Yiu Wai Ming
mankoyiu@gmail.com

2. Objective

The objective of this project is to develop predictive models to classify individuals as infected or not infected based on a set of features. The models aim to identify patterns in the training dataset and provide predictions for a test dataset. The ultimate goal is to compare the performance of different machine learning and neural network architectures to determine the most effective model for this classification task.

3. Datasets

3.1 List of Datasets

1. Training Dataset: `firstData.txt`
This dataset contains both feature columns and the target variable (`Infected`), which is used to train the models.

2. Test Dataset: `secondData.txt`
This dataset contains only the feature columns, and predictions are made for this dataset after training the models.

3.2 Steps of Dataset Transformations

The following transformations were applied to both datasets:

1. Column Naming:
   The raw datasets did not have column names. These were assigned for clarity:

`Fever`, `Tiredness`, `Dry-Cough`, `Difficulty-in-Breathing`, `Sore-Throat`, `Pains`, `Nasal-Congestion`, `Runny-Nose`, `Diarrhea`, `Age`, `Gender`, `Country`, and `Infected` (for the training dataset only).

2. Categorical Variable Encoding:
The columns `Age`, `Gender`, and `Country` were treated as categorical features and converted into dummy variables using encoding. This transformation expanded each categorical feature into multiple binary columns.

3. Feature Scaling:
All numerical features were scaled to the range [0, 1] using `MinMaxScaler` to normalize the data. This ensures that all features contribute equally to the model training, avoiding issues caused by differences in feature magnitudes.

4. Dataset Splitting:
From the training dataset, the features (`X_train`) and target variable (`y_train`) were separated.
The test dataset (`X_test`) was prepared with the same transformations as the training dataset.

4. Models

4.1 List of Implemented Models

Five neural network models were implemented using the Keras library. Each model was trained on the transformed training dataset (`X_train` and `y_train`) and used to generate predictions on the test dataset (`X_test`).

Model 1: Simple Neural Network
- Model Type: Neural Network
- Description: A simple neural network with one hidden layer for basic classification tasks.
- Parameters:
  - Input Layer: 64 neurons, activation function `ReLU`
  - Hidden Layer: 32 neurons, activation function `ReLU`
  - Output Layer: 1 neuron, activation function `Sigmoid`

  - Optimizer: Adam, learning rate = 0.001
  - Loss Function: Binary Crossentropy
  - Epochs: 100
  - Batch Size: 32
- Dataset Used: Training data (`X_train` and `y_train`)

Model 2: Deeper Neural Network with Dropout
- Model Type: Neural Network
- Description: A deeper neural network with two hidden layers and dropout regularization to prevent overfitting.
- Parameters:
  - Input Layer: 128 neurons, activation function `ReLU`
  - First Hidden Layer: 64 neurons, activation function `ReLU`, dropout rate = 0.3
  - Second Hidden Layer: 32 neurons, activation function `ReLU`, dropout rate = 0.3
  - Output Layer: 1 neuron, activation function `Sigmoid`
  - Optimizer: Adam, learning rate = 0.001
  - Loss Function: Binary Crossentropy
  - Epochs: 100
  - Batch Size: 32
- Dataset Used: Training data (`X_train` and `y_train`)

Model 3: Neural Network with Tanh Activation
- Model Type: Neural Network
- Description: A neural network that uses the `Tanh` activation function instead of `ReLU`, which can be beneficial for handling certain data distributions.
- Parameters:
  - Input Layer: 64 neurons, activation function `Tanh`
  - Hidden Layer: 32 neurons, activation function `Tanh`
  - Output Layer: 1 neuron, activation function `Sigmoid`
  - Optimizer: Adam, learning rate = 0.001
  - Loss Function: Binary Crossentropy
  - Epochs: 100 (with early stopping)
  - Batch Size: 32

- Dataset Used: Training data (`X_train` and `y_train`)

Model 4: Wide Neural Network
- Model Type: Neural Network
- Description: A wide neural network with a large number of neurons in the hidden layers, designed to capture complex patterns in the data.
- Parameters:
  - Input Layer: 256 neurons, activation function `ReLU`
  - Hidden Layer: 128 neurons, activation function `ReLU`
  - Output Layer: 1 neuron, activation function `Sigmoid`
  - Optimizer: Adam, learning rate = 0.001
  - Loss Function: Binary Crossentropy
  - Epochs: 100
  - Batch Size: 32
- Dataset Used: Training data (`X_train` and `y_train`)

Model 5: Shallow Neural Network
- Model Type: Neural Network
- Description: A shallow neural network with only one hidden layer, designed as a lightweight model for comparison.
- Parameters:
  - Input Layer: 16 neurons, activation function `ReLU`
  - Output Layer: 1 neuron, activation function `Sigmoid`
  - Optimizer: Adam, learning rate = 0.001
  - Loss Function: Binary Crossentropy
  - Epochs: 100
  - Batch Size: 32
- Dataset Used: Training data (`X_train` and `y_train`)

5. Results

5.1 Results for Each Model
- Model 1: Achieved high accuracy on the training set and reasonable validation performance. Predictions were well generalized.
- Model 2: Improved generalization due to dropout layers, with slightly lower training accuracy but better validation results.

- Model 3: Performed similarly to Model 1 but with slower convergence due to the `Tanh` activation function.
- Model 4: Captured complex patterns in the data, achieving the highest training accuracy but slightly overfitted despite early stopping.
- Model 5: Performed well for a shallow model, but its simplicity limited its ability to capture complex relationships in the data.

5.2 Summary of Results

The deeper models (Model 2 and Model 4) performed better in terms of generalization, while the simpler models (Model 1 and Model 5) were faster to train but had limited capacity. Dropout regularization (Model 2) helped mitigate overfitting, and `Tanh` activation (Model 3) provided an alternative performance profile.

6. Conclusion

This project successfully implemented and evaluated five neural network models for infection prediction. The results demonstrate the trade-offs between model complexity, training time, and generalization. Model 2, with dropout regularization, provided the best balance between accuracy and generalization. These findings highlight the importance of model selection and preprocessing in predictive tasks.