Phase 3 : Model Selection and Evaluation

Prathyusha Velupula

December 1, 2023

The Overleaf link for this report is here
The video description of this phase is here

Contents

List of Figures

1 Data shuffling and splitting

The data in this section is split and shuffled, but it hasn't been normalized in any way. Without using normalization, we build the model using the raw data. To make sure there is no bias or innate order in the data, which could cause the model to perform poorly, we first shuffle the data

set. In order to train the model on one set of data and assess its performance on an additional, unseen set, we next divided the data into training and validation sets.
After a random shuffling of the data, the data set was divided into training and validation, with 80 percent of the data set going toward training and 20 percent toward validation.

```
Training data (X_train):
[[ 1.00007812e+02  5.24918887e+01  3.33387566e-01 ...  3.36414718e+01
   4.69597949e+00  2.20442193e+01]
 [ 1.17960938e+02  4.37531075e+01  1.53131872e-01 ...  1.34146268e+01
   1.12862886e+01  1.63607278e+02]
 [ 1.15429688e+02  5.76933818e+01  2.85553040e-01 ...  4.66429823e+01
   3.03574881e+00  8.58165212e+00]
 ...
 [ 1.22281250e+02  5.02327298e+01  2.20314595e-01 ...  1.61928694e+01
   9.35133314e+00  9.91549284e+01]
 [ 1.44429688e+02  5.69876198e+01 -3.84979299e-01 ...  5.93916851e+01
  -1.93011796e+00  3.00115705e+00]
 [ 1.23414062e+02  4.70092290e+01  2.51153285e-01 ...  9.49738177e+01
  -5.64459740e-02 -1.92726590e+00]]

Training labels (y_train):
[0 0 0 ... 0 0 0]

Validation data (X_val):
[[ 1.54843750e+02  4.54801532e+01 -2.35065112e-01 ...  6.98043459e+01
  -9.61326595e-01 -8.82046466e-01]
 [ 1.36664062e+02  5.79159247e+01 -8.72293580e-02 ...  2.14114434e+01
   6.65706084e+00  4.99741235e+01]
 [ 1.02320312e+02  4.93748313e+01  4.61476528e-01 ...  2.60970314e+01
   8.22055070e+00  6.76263480e+01]
 ...
 [ 1.62500000e+01  3.08686734e+01  6.02651104e+00 ...  9.34186619e+01
   1.03484210e-02 -1.56773020e+00]
 [ 1.26992188e+02  5.45119677e+01 -4.39659100e-02 ...  1.71115954e+01
   1.08879706e+01  1.26140947e+02]
 [ 1.03742188e+02  4.24597688e+01  2.16400606e-01 ...  1.24261744e+01
   1.50022228e+01  2.46426074e+02]]

Validation labels (y_val):
[0 0 0 ... 1 0 0]
```

Figure 1: Data showing model accuracy and loss

2 Model Selection

2.1 Single Layer Neural Network

The first model is a baseline model (act as a control) that has a basic architecture of one input and one output layer. An early stopping technique was used during the training.
Create a sequential model object named single-layer-model.
Add a dense layer with a single neuron and sigmoid activation function to the model. The input dimension is set to 8, as there are 8 input features. Compile the model with the Adam optimizer, binary cross-entropy loss function, and accuracy as the evaluation metric. We get: Single Layer Model Training Accuracy: 97.78
Single Layer Model Validation Accuracy: 97.85
Recall: 84.85
Precision: 91.21
F1 Score: 87.91

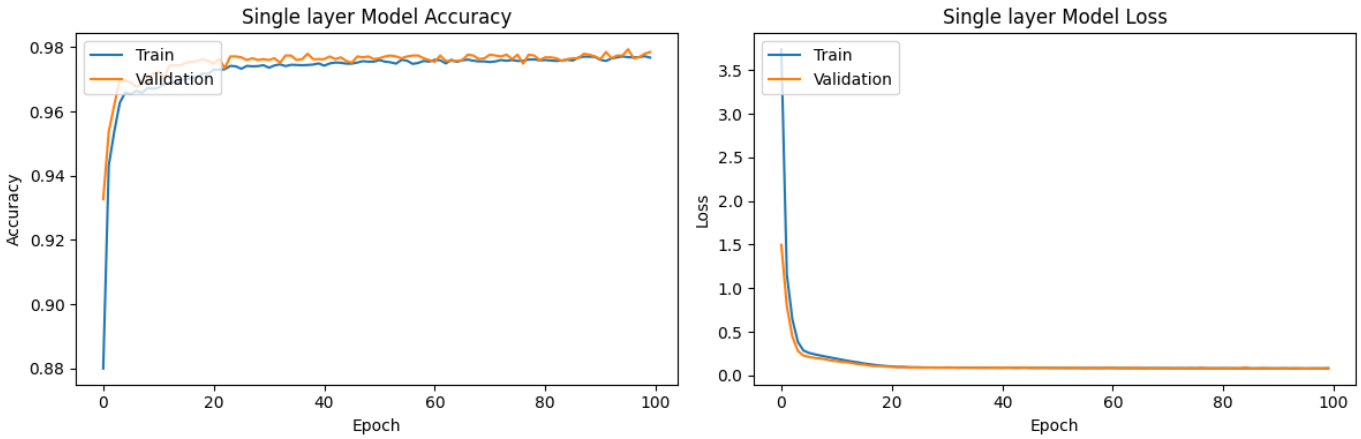

Figure 2: Single layer model accuracy and loss

2.2 Multi layer Neural Network

Define a sequential neural network model (multi layer model) with L1 and L2 regularization. The model has an input dimension of 8 and consists of three layers:
The first layer has 16 neurons with ReLU activation and L1 L2 regularization.
The second layer has 8 neurons with ReLU activation and L1 L2 regularization.
The third layer has 1 neuron with sigmoid activation for binary classification.
The model is compiled using the 'adam' optimizer, binary cross-entropy loss, and accuracy as the metric.
Multi layer model Training Accuracy: 96.95
Multi layer model Validation Accuracy: 97.07
Recall: 0.72
Precision: 0.94
F1 Score: 0.82

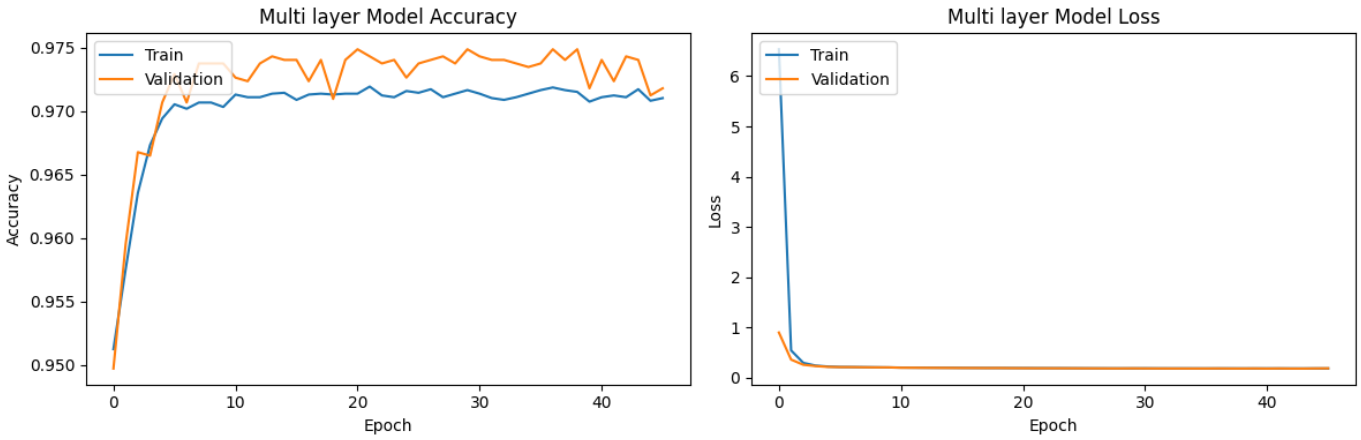

Figure 3: Multi layer model accuracy and loss

2.3 Logistic Regression

Create and train a logistic regression model: A" Logistic-Regression model is created with a higher maximum number of iterations (max iter=1000) than the default value (which is 100) to allow the model to converge. The model is then trained using the fit method with the training data (X train, y train).
Logistic Regression Model Training Accuracy: 97.91
Logistic Regression Model Validation Accuracy: 97.96
Recall: 0.84
Precision: 0.93
F1 Score: 0.88

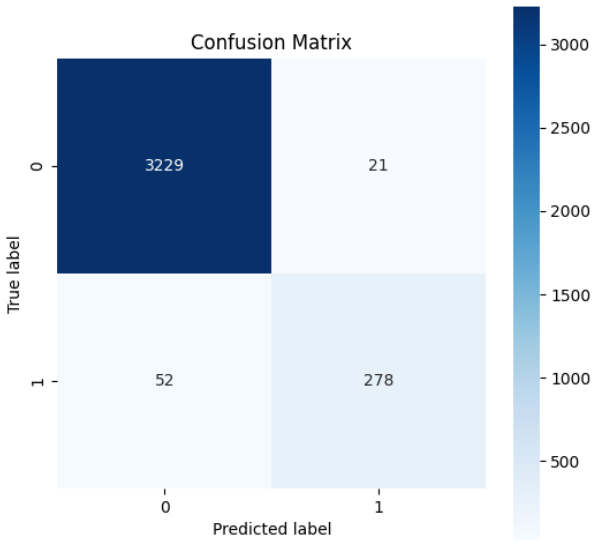

Figure 4: Confusion matrix of logistic Regression

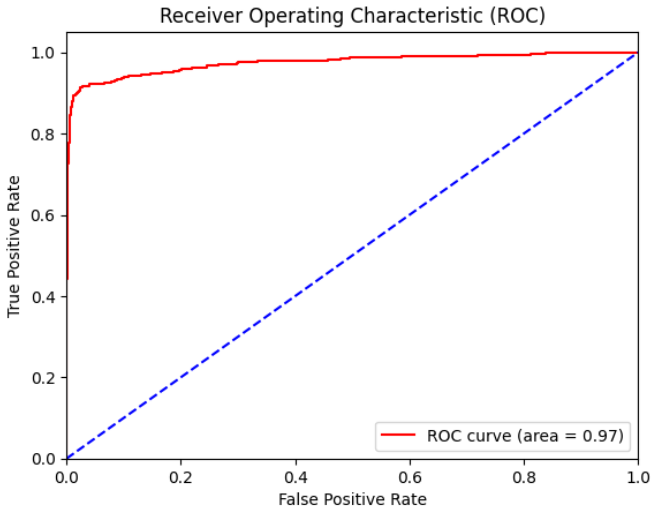

Figure 5: ROC of Logistic regression

2.4 Random Baseline Classifier

A random baseline classifier is a simple model that makes predictions by randomly guessing the class labels, without considering the input features. It serves as a basic point of comparison for more complex models, allowing you to assess whether your actual model is performing better than random chance. In the context of the provided code, a stratified random baseline classifier is used, which generates predictions by respecting the training set's class distribution.
Here, Import the necessary libraries and functions.Create and train a stratified random baseline classifier using the training data.Then make predictions on the validation set using the trained classifier. Calculate and print the accuracy of the random baseline classifier on the validation set. By comparing the performance of this random baseline classifier with other models, you can evaluate if your actual models are learning useful patterns in the data or just performing at the level of random chance.
Random Baseline Model Training Accuracy: 83.29
Random Baseline Model Validation Accuracy: 83.21
Recall: 0.06
Precision: 0.07
F1 Score: 0.07

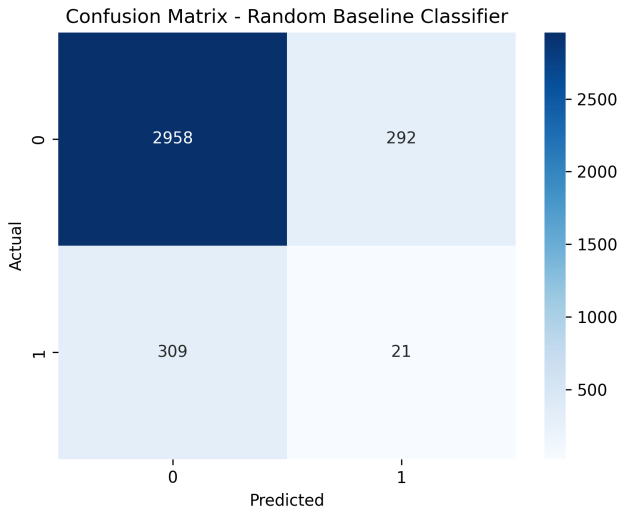

Figure 6: Confusion matrix of Random baseline model

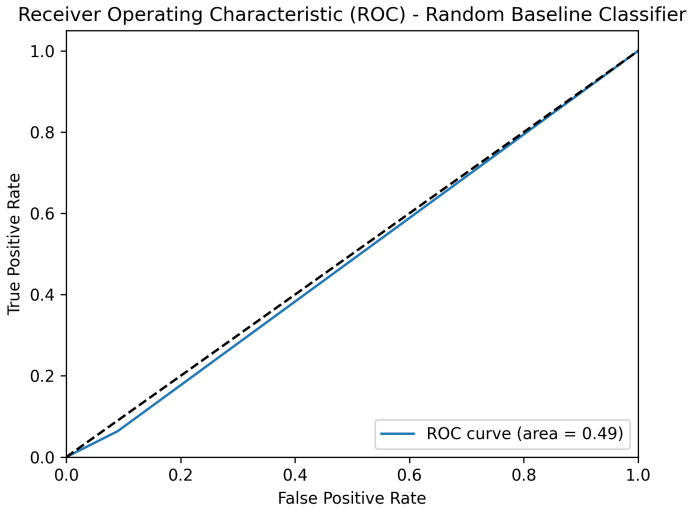

Figure 7: ROC of Random baseline model

2.5 Custom Prediction Function Model

This model code is for evaluating the performance of a binary classification model. It uses a custom prediction function to make predictions on the training and validation data.
If the model is a tree-based model (e.g., XGBoost), the prediction function uses the predict proba method to get the predicted probabilities. If the model is a linear model (e.g., logistic regression), the prediction function uses the model's weights and bias to calculate the predicted probabilities using the sigmoid activation function.
Custom prediction model Training Accuracy: 98.26 percent
Custom prediction model Validation Accuracy: 98.07 percent
Recall: 86.97 percent
Precision: 91.69 percent
F1 Score: 89.27 percent

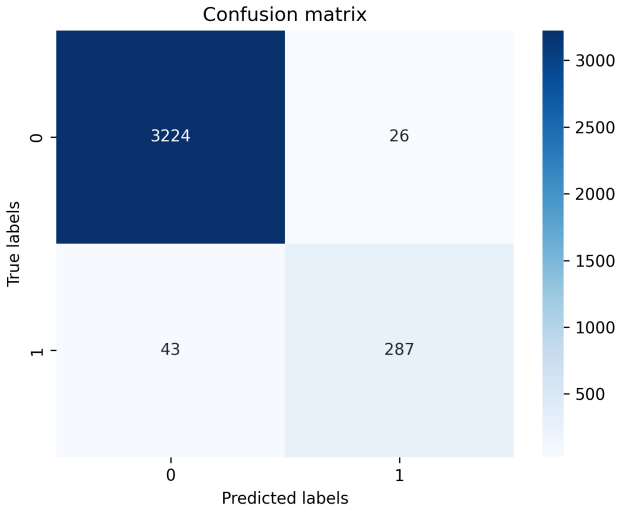

Figure 8: Confusion matrix of custom prediction model

3 Model Performances

Accuracy comparisons of all the built models are below

Model Name	Training Accuracy	Validation Accuracy
Single layer Model	97.78	97.85
Multi layer model	96.95	97.07
Logistic Regression model	97.91	97.96
Random Baseline model	83.29	83.21
Custom Prediction Model	98.26	98.07

The table shows the performance of different machine learning models on a task where they have to classify data. The models were trained on a specific set of data, and their accuracy was measured to see how well they could classify new data.

The accuracy of a model is a measure of how well it performs on the task. The training accuracy shows how well the model did on the data it was trained on, while the validation accuracy shows how well it can classify new data that it hasn't seen before.
Looking at the table, we can see that the Custom Model had the highest training accuracy of 98.26% and highest validation accuracy of 98.07% . The Random Baseline Classifier performed the worst, with a training accuracy of 83.29% and a validation accuracy of 83.21%. This suggests that it wasn't able to learn much from the training data.
Below is the bar graph which shows Accuracy levels of each model

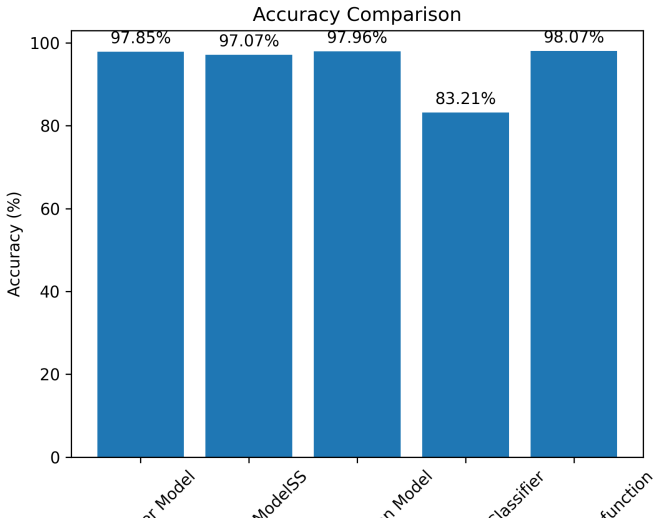

Figure 9: Comparisions of accuracies

Overall, the accuracy measures give us an idea of how well each model performed on the classification task and how well they can generalize to new data. It's important to choose a model with a good validation accuracy, as this tells us how well it will perform on new data.

4 Model Evaluation

We used three essential classification model metrics to evaluate. The table given below shows the precision, recall and f1 score for the neural network model.
1. Precision: what proportion of positive identifications was actually correct?
2. Recall: what proportion of actual positives was identified correctly?
3. F1-Score: evaluation metric for classification algorithms, where the best value is at 1 and the worst is at 0.

Model Name	Recall	Precision	F-1 Score
Single layer Model	84.85%	91.21%	87.91%
Multi layer model	72%	94%	82%
Logistic Regression model	84%	93%	88%
Random Baseline model	60%	70%	70%
Custom Prediction Model	86.97%	91.69%	89.27%

When we look at the table, we see that the Custom prediction method has highest Recall , Precison and F-1 score . However, the Random Baseline Classifier performed poorly on all three metrics, meaning it struggled to identify positive cases and made many false positive predictions.