Technical Report

A.Dataset used-Iris

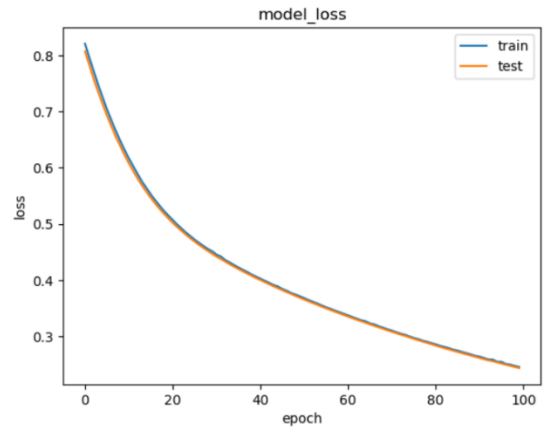
Algorithm: The implemented algorithm is a neural network model for multi-class classification using the Iris dataset.

The algorithm follows the following steps:

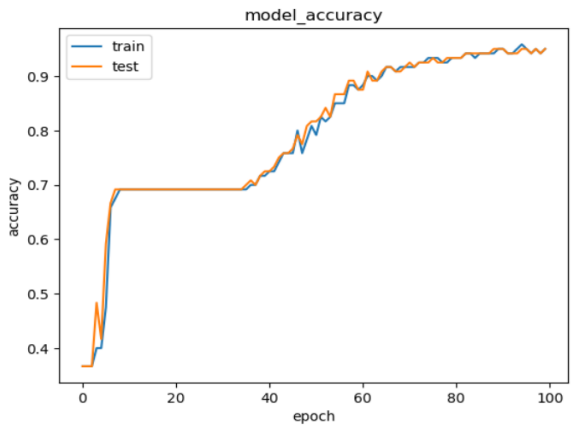
- Dataset Loading and Pre processing
- Data Splitting
- Data Scaling
- Model Building
    - Initial Settings
        - The random state is set to 1 for reproducibility in the data splitting process.
        - The batch size is set to 5, and the number of epochs is set to 100 for training the model.
        - The initial weights and biases of the neural network are initialized using the 'uniform' distribution.
        - The 'adam' optimizer is used for optimizing the model's weights during training.
        - The 'relu' activation function is used in the first dense layer, and 'softmax' is used in the output layer for multi-class classification.
- Model Evaluation
    - Model Train Accuracy is: 0.9666666666666667
    - Model test Accuracy is: 0.9666666666666667
    - Model Recall_score is: 0.9666666666666667
    - Model Precision_score is: 0.9714285714285714
    - Model f1_score is: 0.9672820512820512

Plots:

Train vs Test Loss Plot



Train vs Test Accuracy Plot



Observations: From both the plots, we can see that there is no much difference in train and test data, which suggests that there is no overfitting in the data. For epochs between 15 and 40, there was no change in the accuracy score, but after that there has been a positive change in the score.

Final Parameters:

```
Layer 1 - Weights Shape: (4, 8), Biases Shape: (8,)
Weights:
[[ 6.20050669e-01 -2.77791377e-02  6.75269008e-01  4.97486442e-03
   9.47589651e-02 -1.51394885e-02  1.06299622e-02  6.98686421e-01]
 [-2.40013078e-01 -3.63693126e-02 -2.63693154e-01 -3.17363366e-02
   1.12260377e+00  5.54857892e-04 -5.11212088e-02 -2.89577544e-01]
 [ 1.00283813e+00  3.92386429e-02  1.04491270e+00 -4.93623018e-02
  -7.74020553e-01 -3.10405940e-02  7.01285526e-03  1.08868361e+00]
 [ 1.18267763e+00 -3.91772874e-02  1.17962396e+00  2.06004642e-02
  -1.14501321e+00 -8.20564572e-04 -2.09459104e-02  1.23562491e+00]]
Biases:
[-0.18579331  0.         -0.1786885   0.          1.327746   -0.01109695
 -0.0103669  -0.17379849]

Layer 2 - Weights Shape: (8, 3), Biases Shape: (3,)
Weights:
[[-1.429983    0.06285403  0.57688475]
 [ 0.03748068  0.04589346  0.04628743]
 [-1.3981103   0.09726243  0.5672563 ]
 [ 0.00689896 -0.04306279 -0.04351223]
 [ 1.5804012  -0.06802545 -2.1458933 ]
 [-0.01712993 -0.0043772  -0.03561551]
 [ 0.01120028 -0.03608674 -0.02733093]
 [-1.3788185   0.1243028   0.5717949 ]]
Biases:
[ 0.39958107  0.21322224 -0.51645446]
```

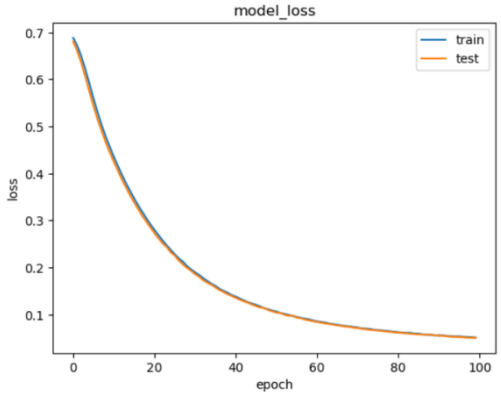
B.Dataset used-Banknote

Algorithm: The implemented algorithm is a neural network model for binary-class classification using the Banknote dataset.

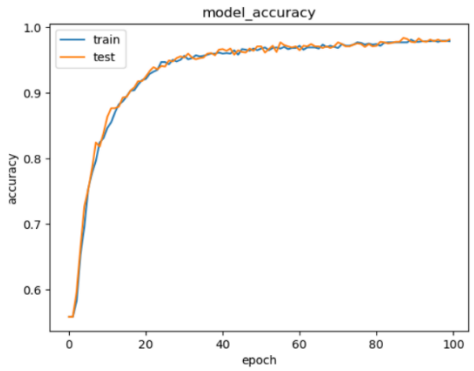
The algorithm follows the following steps:

- Dataset Loading and Pre processing
- Data Splitting
- Data Scaling
- Model Building
    - Initial Settings
        - The random state is set to 42 for reproducibility in the data splitting process.
        - The batch size is set to 10, and the number of epochs is set to 100 for training the model.
        - We have also used earlystopping in order to avoid unnecessary steps.
        - The initial weights and biases of the neural network are initialized using the 'uniform' distribution.
        - The 'adam' optimizer is used for optimizing the model's weights during training.
        - The 'relu' activation function is used in the first dense layer, and 'sigmoid' is used in the output layer for binary-class classification.
- Model Evaluation
    - Model Train Accuracy is: 0.9811320754716981
    - Model Test Accuracy is: 0.9805825242718447
    - Model Recall_score is: 0.9789473684210527
    - Model Precision_score is: 0.9789473684210527
    - Model f1_score is: 0.9789473684210527

- Plots:
  Train vs Test Loss Plot



  Train vs Test Accuracy Plot



Observations: From both the plots, we can see that there is no much difference in train and test data, which suggests that there is no overfitting in the data.

Final Parameters:

```
Layer 1 - Weights Shape: (4, 8), Biases Shape: (8,)
Weights:
[[-0.03966632  1.3254781   1.3569392  -1.1429539  -1.1488069   1.3070796
   1.2556143   0.02999468]
 [-0.04992492  1.0816566   1.0696799  -0.68932986 -0.69463336  1.0533218
   1.1127499   0.0124517 ]
 [-0.03853904  1.2078604   1.2115628  -0.940983   -0.90728515  1.2609135
   1.2555716  -0.02568509]
 [ 0.00294281 -0.11997388 -0.09058328  0.37602738  0.39840624 -0.05994143
  -0.07539572 -0.06306284]]
Biases:
[ 0.         -1.1480819  -1.1684375   1.4987128   1.4942158  -1.1943846
 -1.1962932  -0.01811425]

Layer 2 - Weights Shape: (8, 1), Biases Shape: (1,)
Weights:
[[-0.03284212]
 [-3.290249  ]
 [-3.233581  ]
 [ 3.814225  ]
 [ 3.69447   ]
 [-3.4170122 ]
 [-3.3940306 ]
 [ 0.01536426]]
Biases:
[0.9292983]
```

C.Dataset used- Wisconsin Breast Cancer

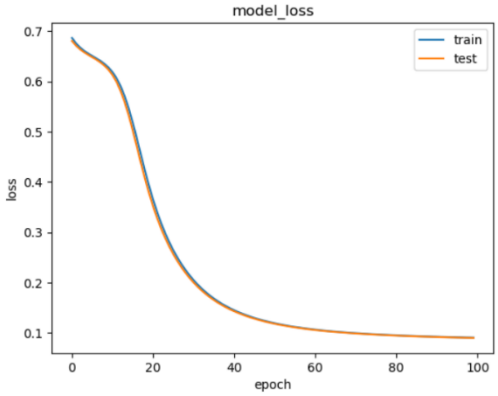
Algorithm: The implemented algorithm is a neural network model for binary-class classification using the Winconsin Breast Cancer dataset.

The algorithm follows the following steps:

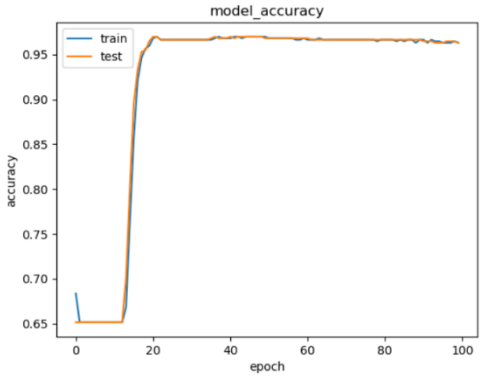
- Dataset Loading and Pre processing
  - Data cleaning in terms of replacing '?' in 'bare_nucleoli' column to 0 and changing the datatype to integer has been done. Target class has also been changed from 2,4 to 0,1 representing benign and malignant cases respectively.
- Data Splitting
- Data Scaling

- Model Building
    - Initial Settings
        - The random state is set to 1 for reproducibility in the data splitting process.
        - The batch size is set to 10, and the number of epochs is set to 100 for training the model.
        - We have also used earlystopping in order to avoid unnecessary steps.
        - The initial weights and biases of the neural network are initialized using the 'uniform' distribution.
        - The 'sgd' optimizer is used for optimizing the model's weights during training.
        - The 'relu' activation function is used in the first dense layer, and 'sigmoid' is used in the output layer for binary-class classification.
- Model Evaluation
    - Model Train Accuracy is: 0.9629629629629629
    - Model Test Accuracy is: 0.9428571428571428
    - Model Recall_score is: 0.8529411764705882
    - Model Precision_score is: 0.9666666666666667
    - Model f1_score is: 0.90625
- Plots:

Train vs Test Loss Plot



Train vs Test Accuracy Plot



Observations: From both the plots, we can see that there is no much difference in train and test data, which suggests that there is no overfitting in the data. From accuracy curve, we can note that there has been no significant change in accuracy after 20 epochs.

D.Dataset used-MNIST

Algorithm: The implemented algorithm is a neural network model for multi-class classification using the MNIST dataset.

The algorithm follows the following steps:

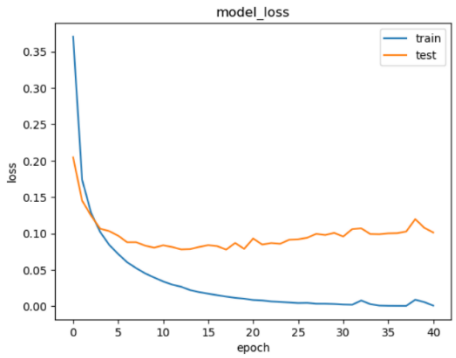
- Dataset Loading and Pre processing
- Data Splitting
- Data Normalising to get values between 0 and 1
- Model Building
    - Initial Settings
        - The random state is set to 1 for reproducibility in the data splitting process.

- The batch size is set to 128, and the number of epochs is set to 10 for training the model.
- We have built the model- with and without convolution layer
- We have also used earlystopping in order to avoid unnecessary steps.
- The initial weights and biases of the neural network are initialized using the 'uniform' distribution.
- The 'adam' optimizer is used for optimizing the model's weights during training.
- The 'relu' activation function is used in the first dense layer, and 'softmax' is used in the output layer for multi-class classification.

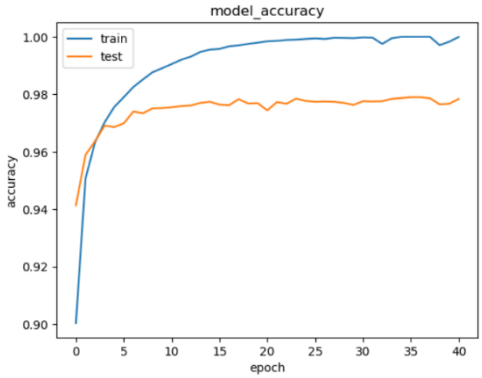
- Model Evaluation
  - Train Accuracy: 0.9991333333333333
  - Test Accuracy: 0.9893
  - F1 score: 0.9892963551619991
  - Recall Score: 0.9893
  - Precision Score: 0.989318086733273

- Plots:

Train vs Test Loss Plot



Train vs Test Accuracy Plot



Observations: From both the plots, we can see that there is difference in train and test data, which suggests that there is overfitting in the data.