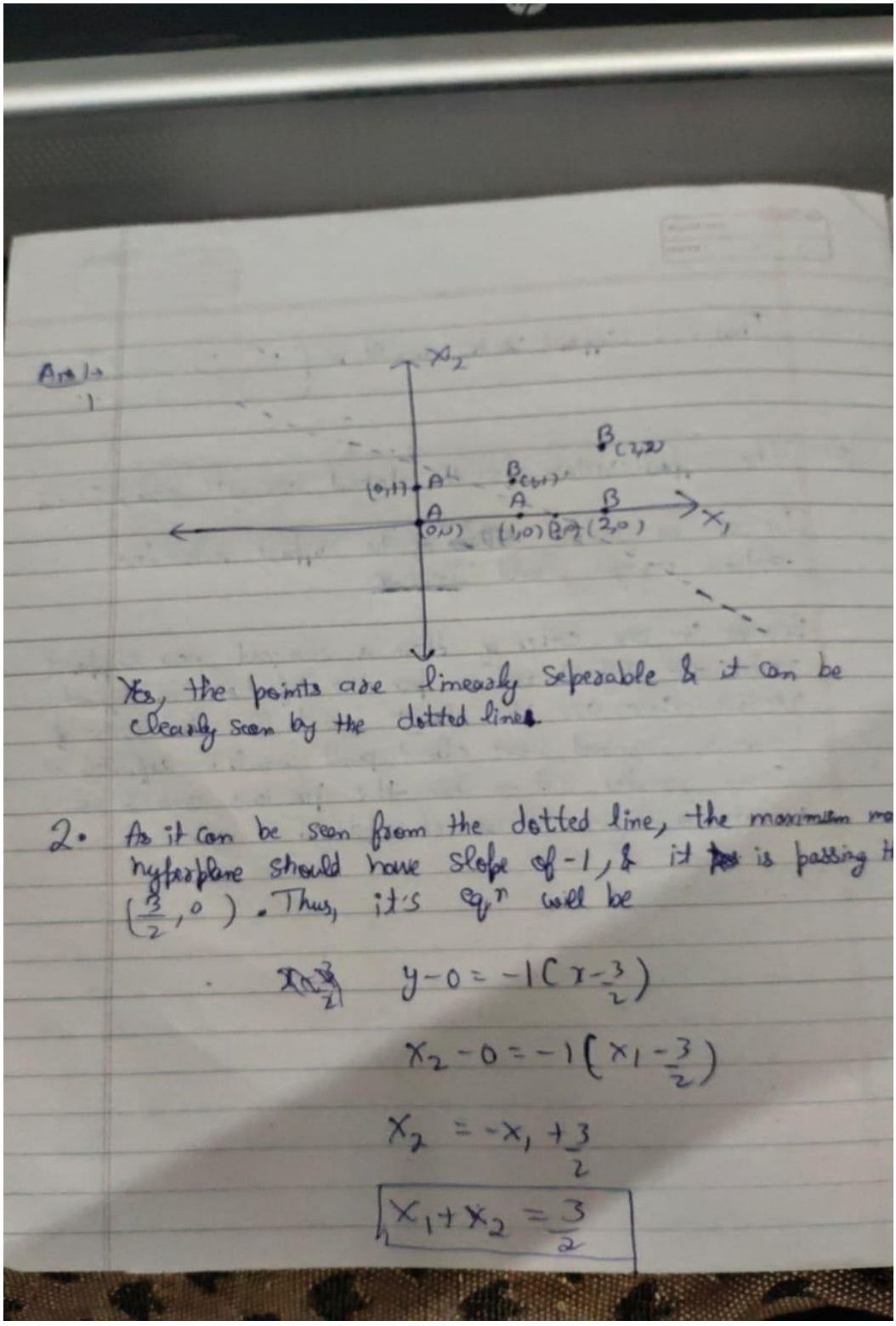
VAIBHAV RAJPAL

2020146

14-Nov-22

ASSIGNMENT-3

Ans 1



Ans 1 →

Yes, the points are linearly seperable & it can be clearly seen by the dotted lines.

2. As it can be seen from the dotted line, the maximum ma hyperplane should have slope of -1, & it is passing th $(\frac{3}{2}, 0)$. Thus, it's eqⁿ will be

$$y - 0 = -1(x - \frac{3}{2})$$

$$x_2 - 0 = -1(x_1 - \frac{3}{2})$$

$$x_2 = -x_1 + \frac{3}{2}$$

$$\boxed{x_1 + x_2 = \frac{3}{2}}$$

Thus, it's support vector would be $\begin{bmatrix} 1 \\ 1 \end{bmatrix} \Rightarrow (1,1)^T$

3. The support vectors for the dataset are A(1,0) & B(1,2)

In case we remove one of the support vector the optimal margin would Increase

Because in case either of them is removed new support vector are defined and as per the definition of support vectors these are the closest points & when either of them is removed, next closest point would be definitely at some greater distance than the previous one & which would result in increase of optimal margin.

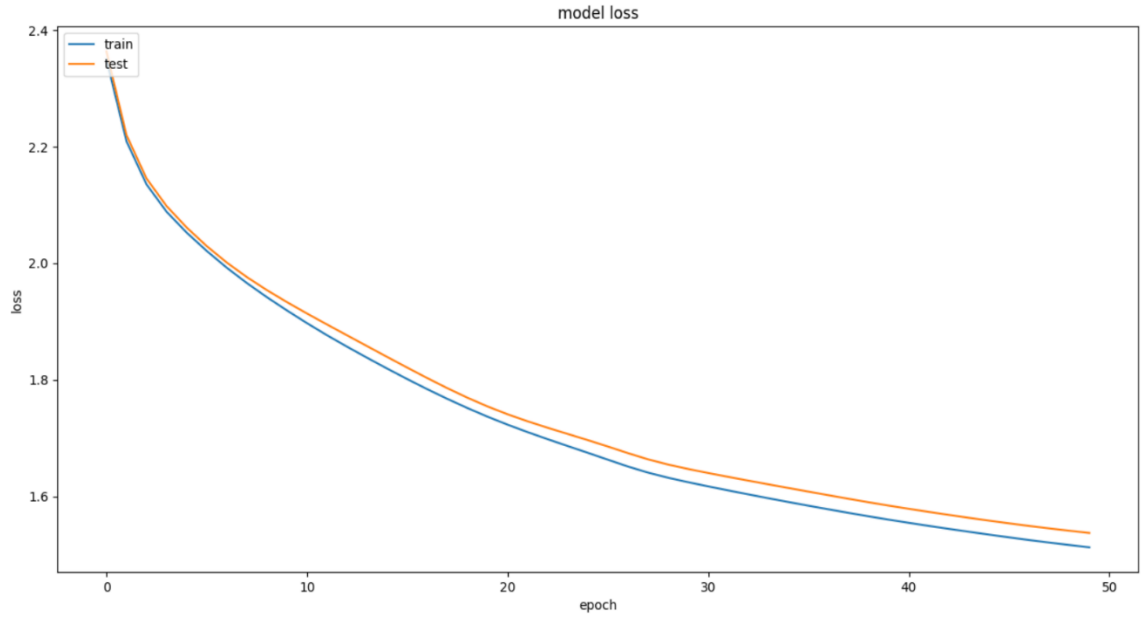
4. For any general dataset, on removing Support Vectors the optimal margin can "increase or remain the same".

Generally on removing dataset optimal margin increase but depending upon the distribution of dataset it could even remain the same.

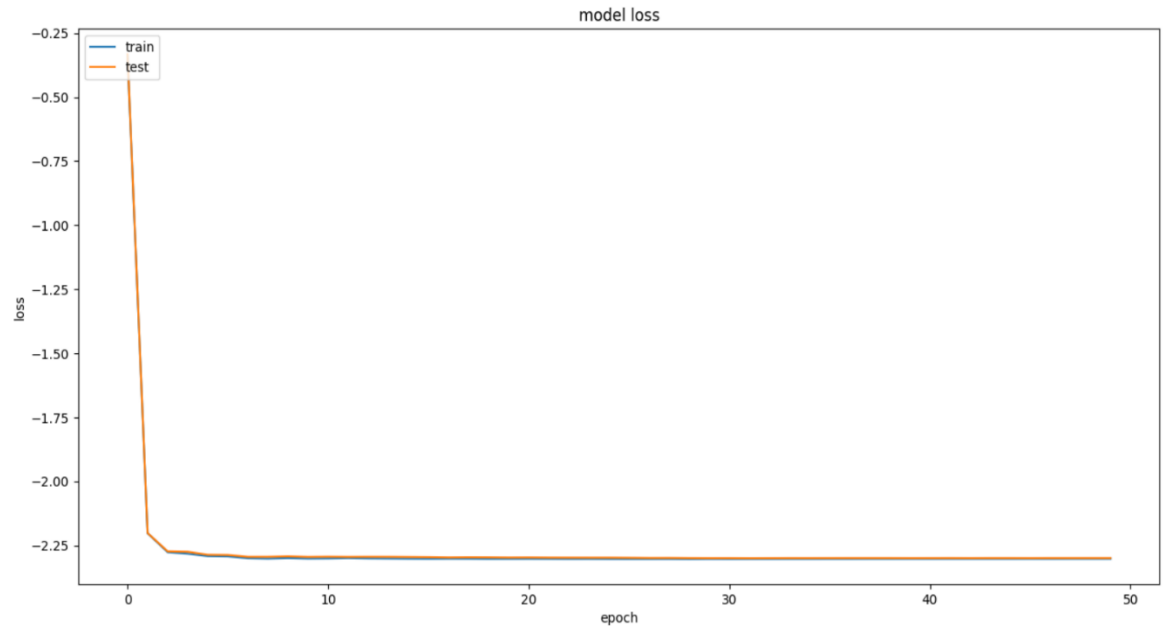
Ans 2:

4.

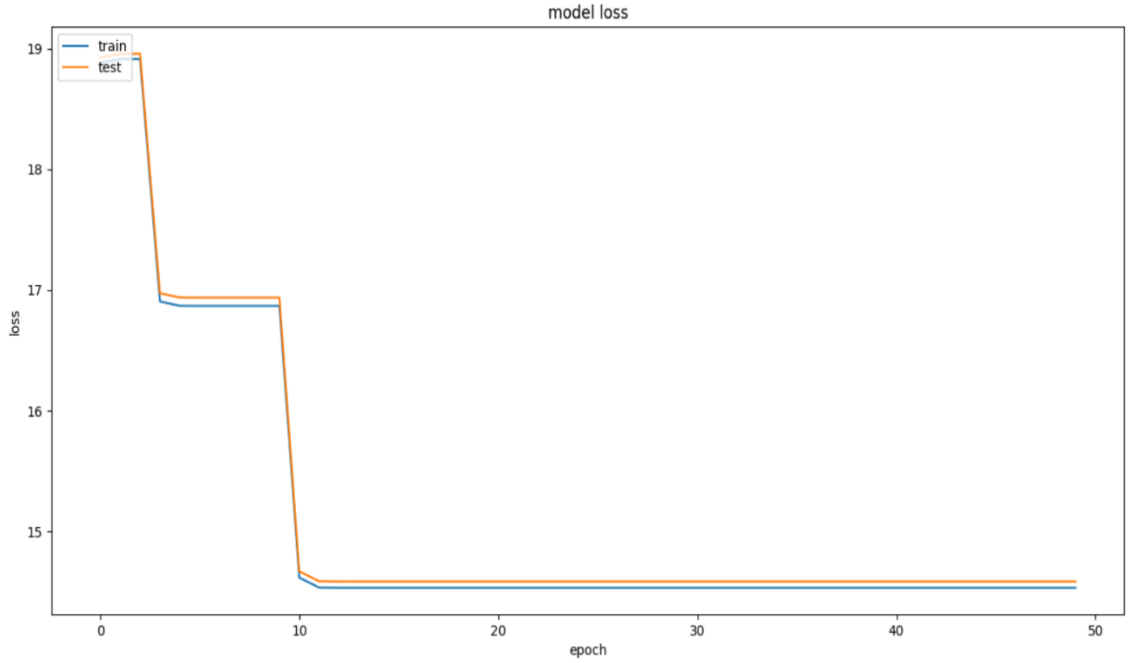
Sigmoid:



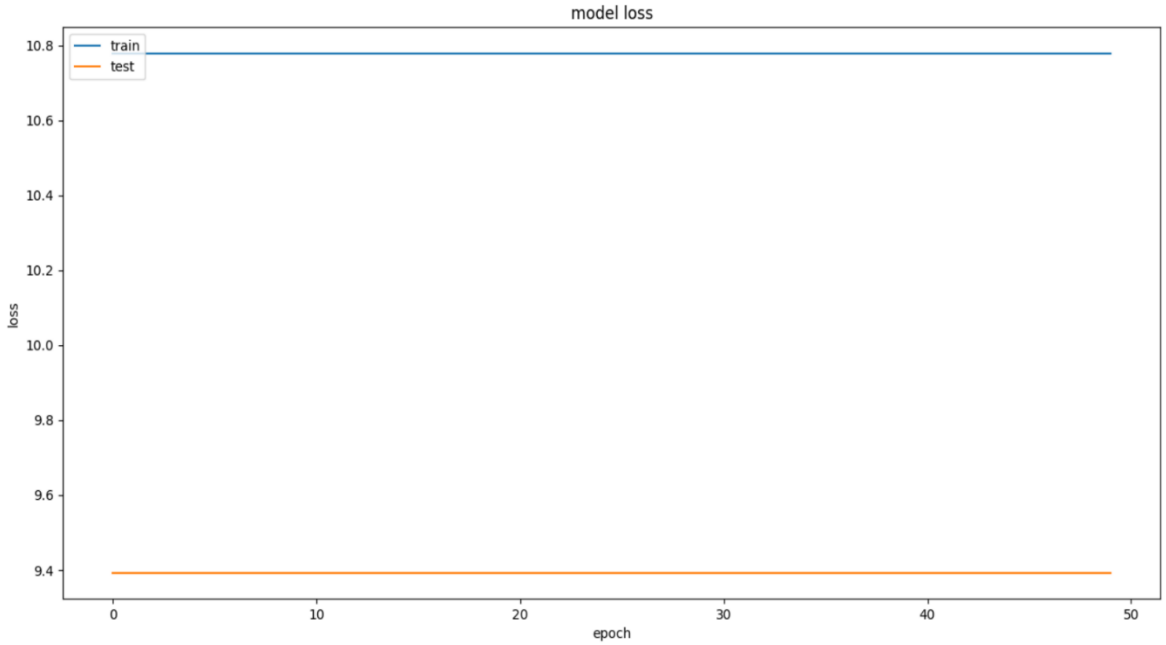
Tanh:



Relu:

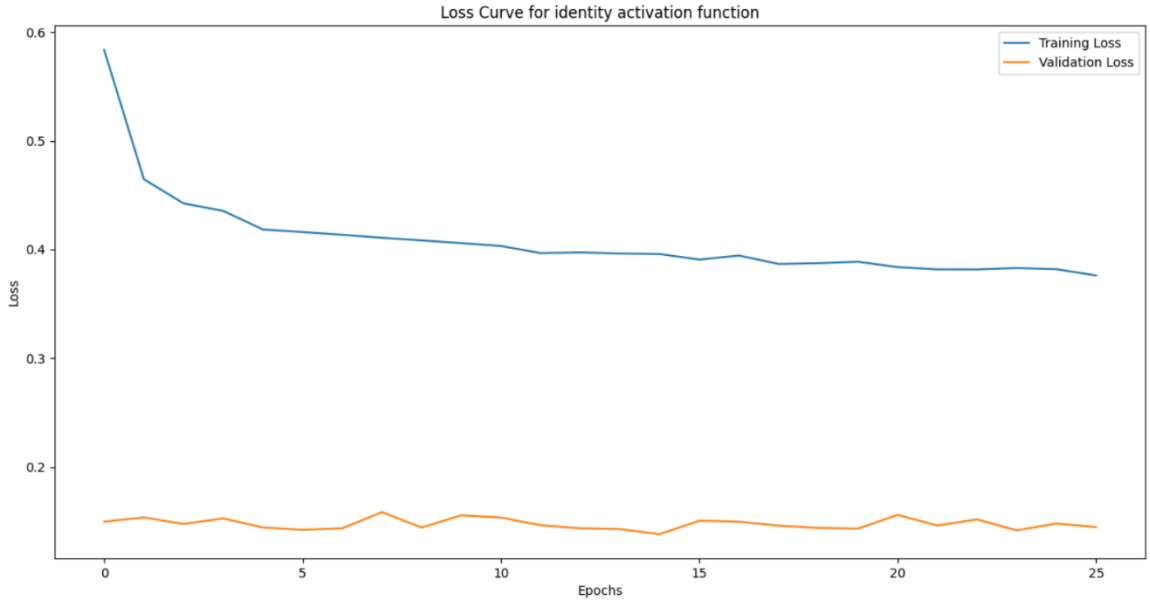


Softmax:



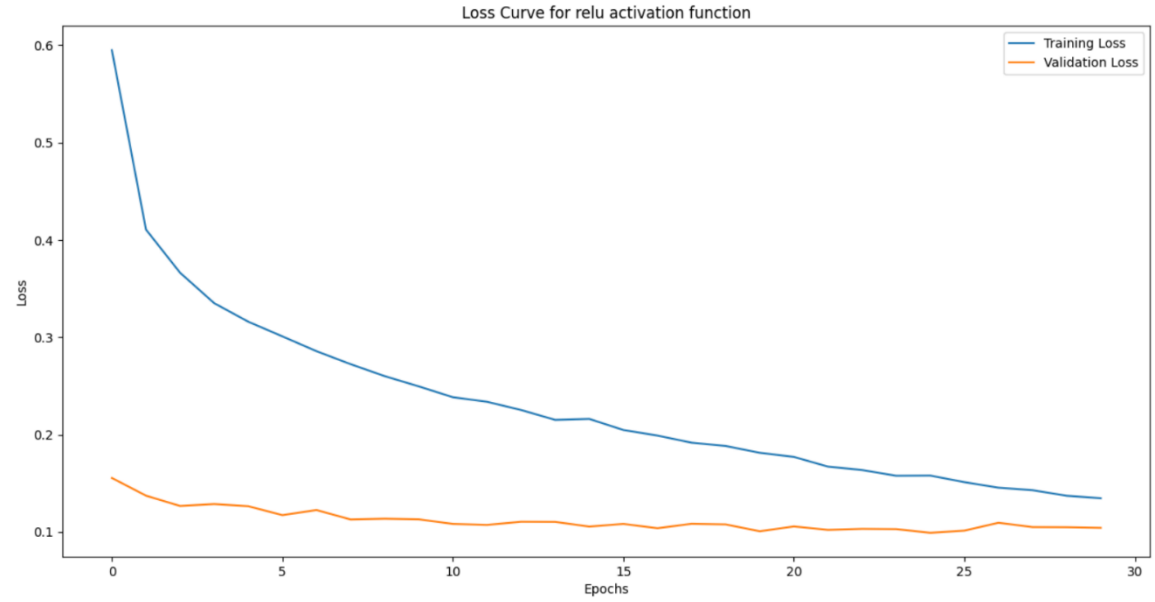
Ans 3:

1. Linear:



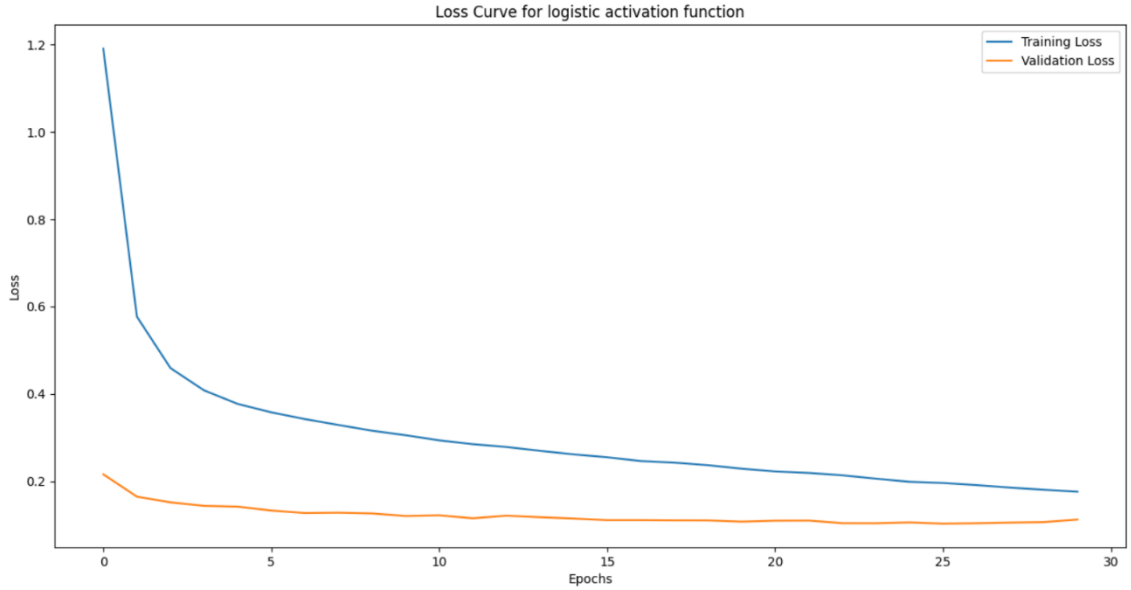

```
Accuracy for identityactivation function in training data is:  0.8690196078431373
Accuracy for identityactivation function in validation data is:  0.8515555555555555
```

Relu:



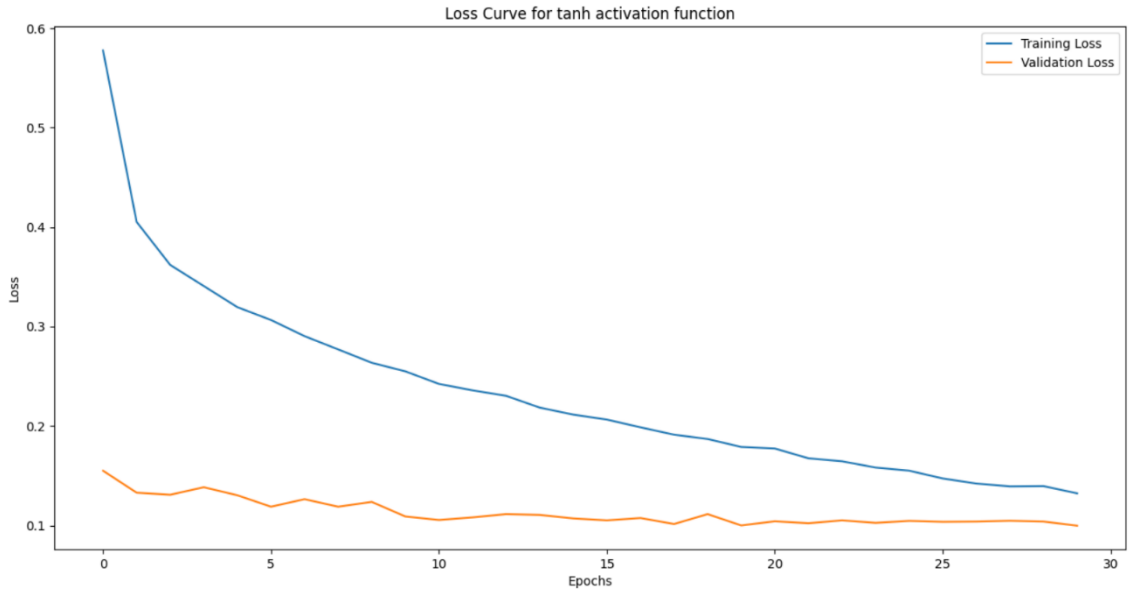

```
  warnings.warn(
Accuracy for reluactivation function in training data is:  0.9447254901960784
Accuracy for reluactivation function in validation data is:  0.8946666666666667
```

Sigmoid:




```
Accuracy for logisticactivation function in training data is:  0.9298039215686275
Accuracy for logisticactivation function in validation data is:  0.8908888888888888
```

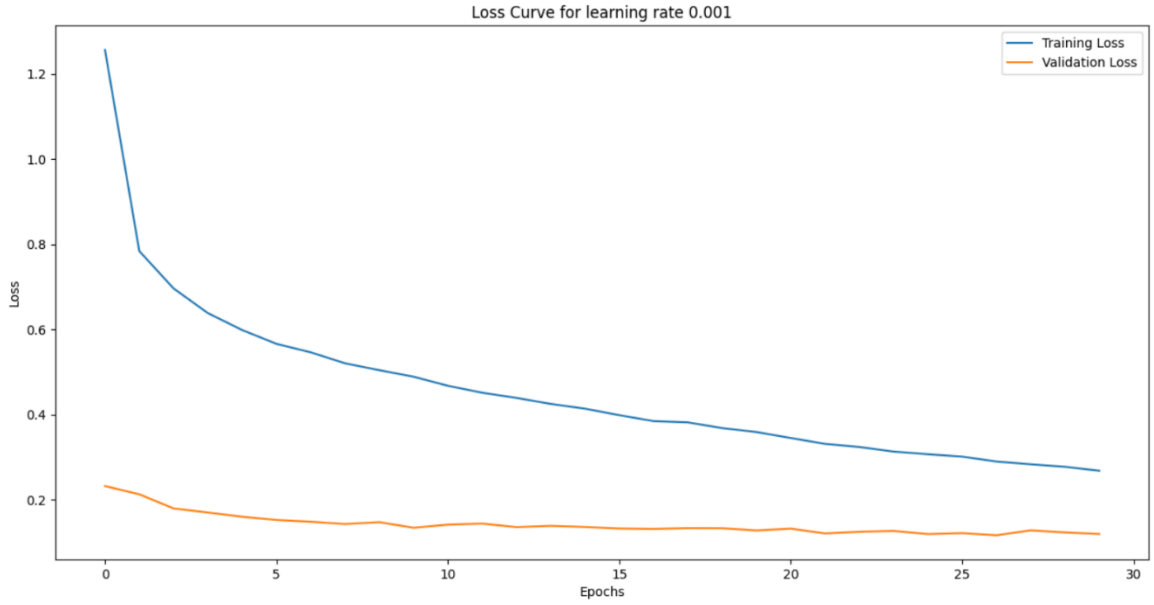
Tanh:



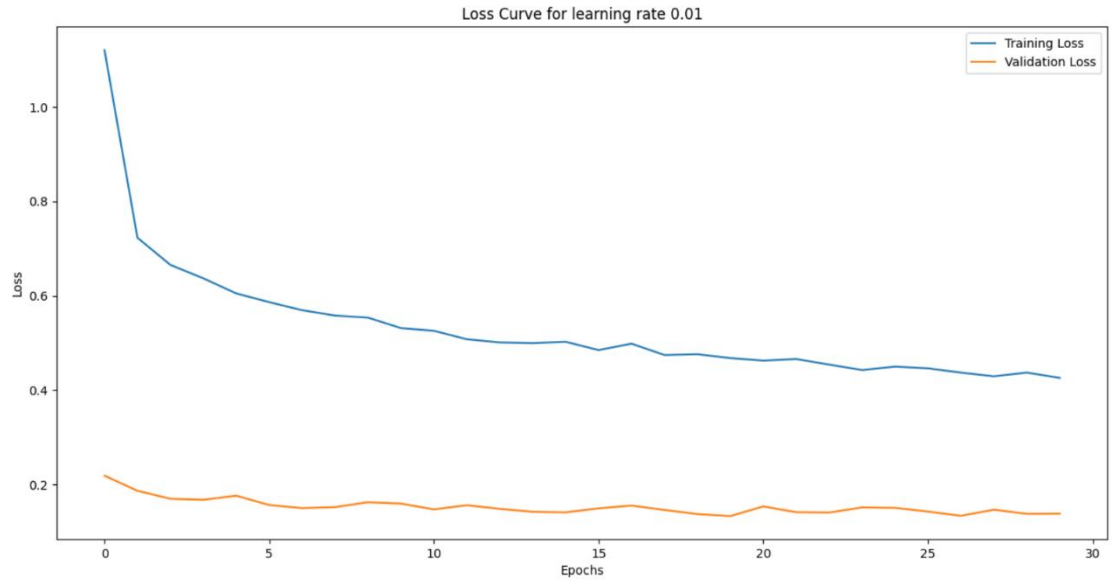

```
Accuracy for tanhactivation function in training data is:  0.930921568627451
Accuracy for tanhactivation function in validation data is:  0.8876666666666667
```

From all of above we can see that Relu gives the highest accuracy.

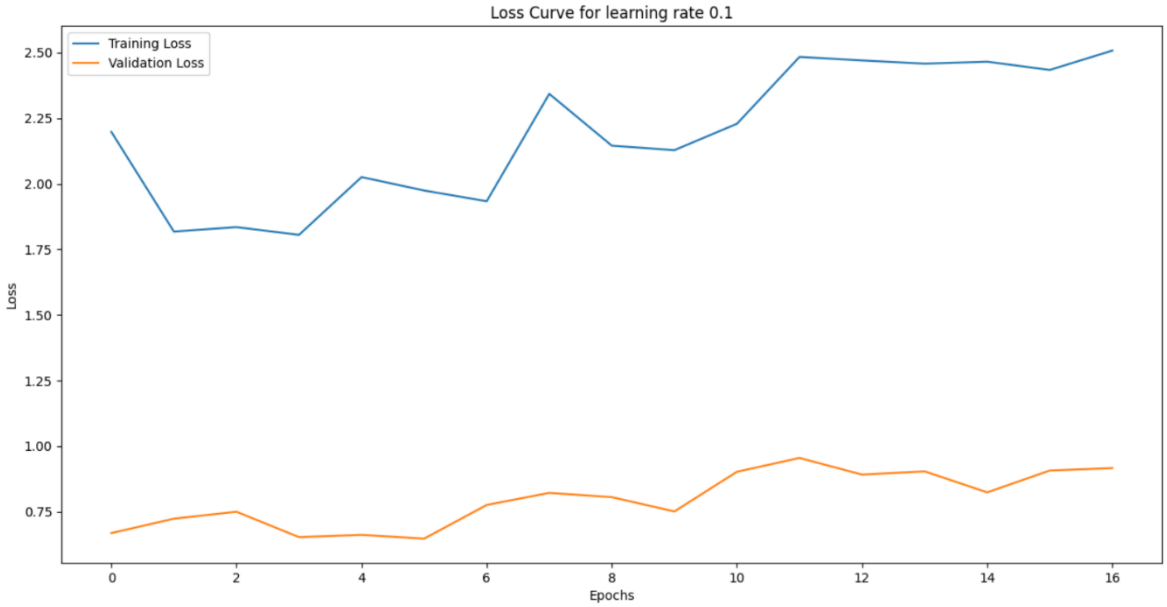
2. For 0.001:



For 0.01:



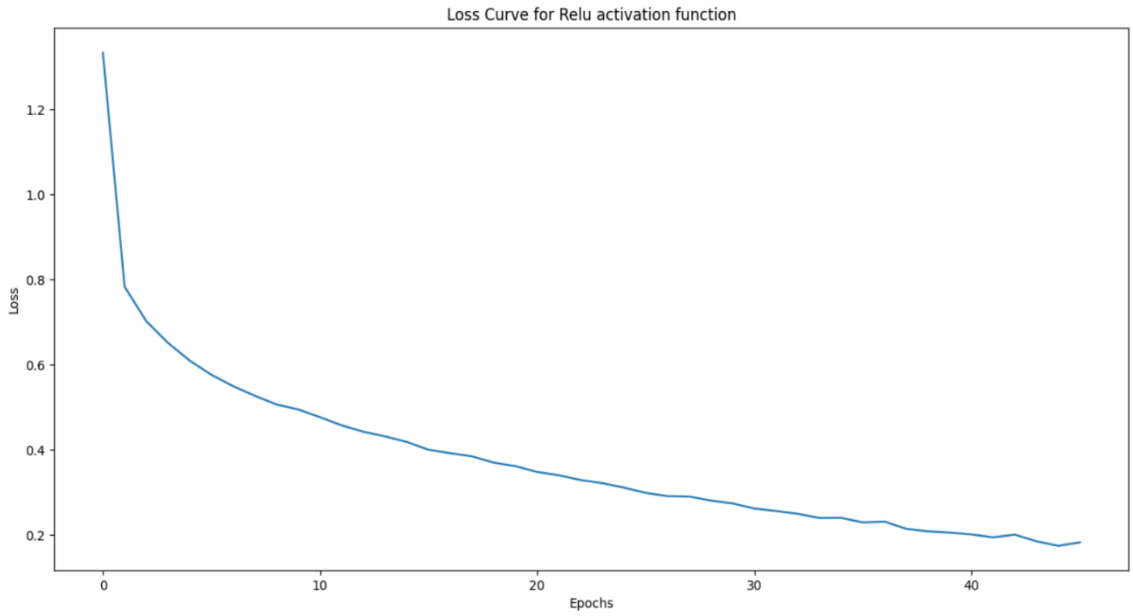
For 0.1:



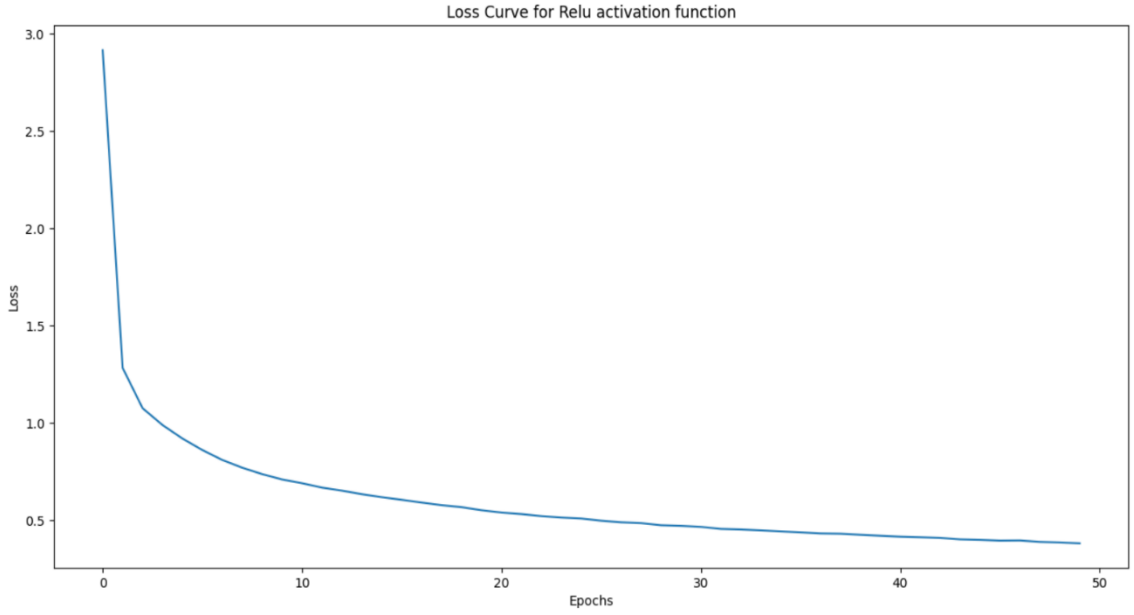

```
Accuracy of the model for learning rate 0.1 is 0.3451111111111111
C:\Users\91991\AppData\Local\Programs\Python\Python38\lib\site-packages\sklea
nd the optimization hasn't converged yet.
  warnings.warn(
Accuracy of the model for learning rate 0.01 is 0.857
C:\Users\91991\AppData\Local\Programs\Python\Python38\lib\site-packages\sklea
nd the optimization hasn't converged yet.
  warnings.warn(
Accuracy of the model for learning rate 0.001 is 0.8768888888888889
```

Thus we can observe that we get maximum accuracy for 0.001. Learning rate tells how quickly model can be trained. Smaller the learning rate more epochs would be required by it and because of it there would be less change in weights of the model thus giving higher accuracy than higher learning rate.

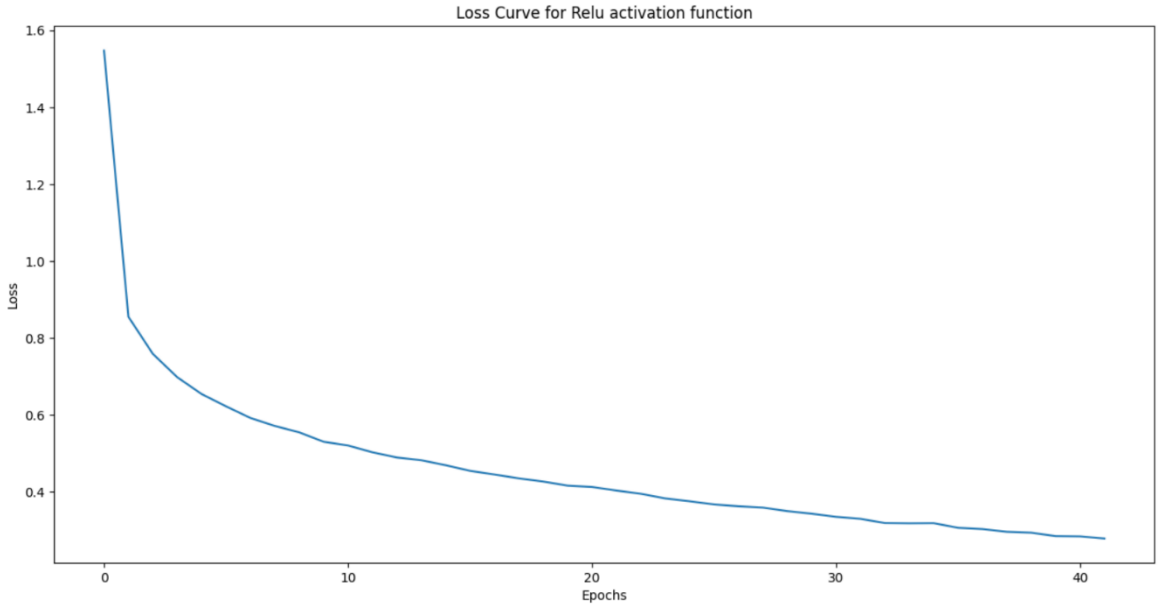
3. (256,32):


Loss Curve for Relu activation function
Loss
Epochs
1.2
1.0
0.8
0.6
0.4
0.2
0
10
20
30
40

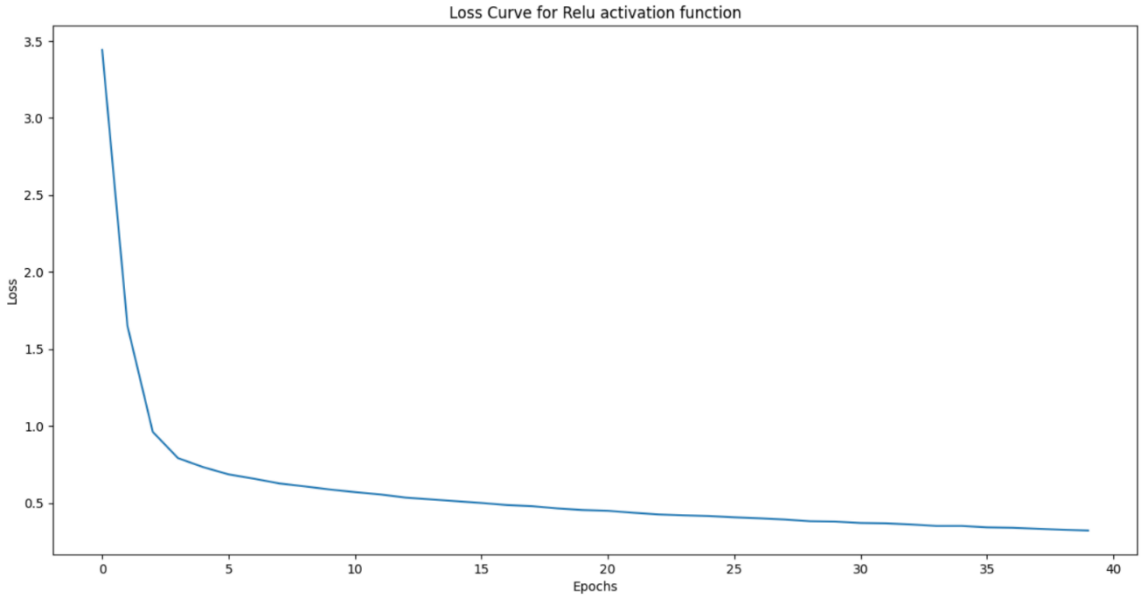
(128,32):


Loss Curve for Relu activation function
Loss
Epochs
3.0
2.5
2.0
1.5
1.0
0.5
0
10
20
30
40
50

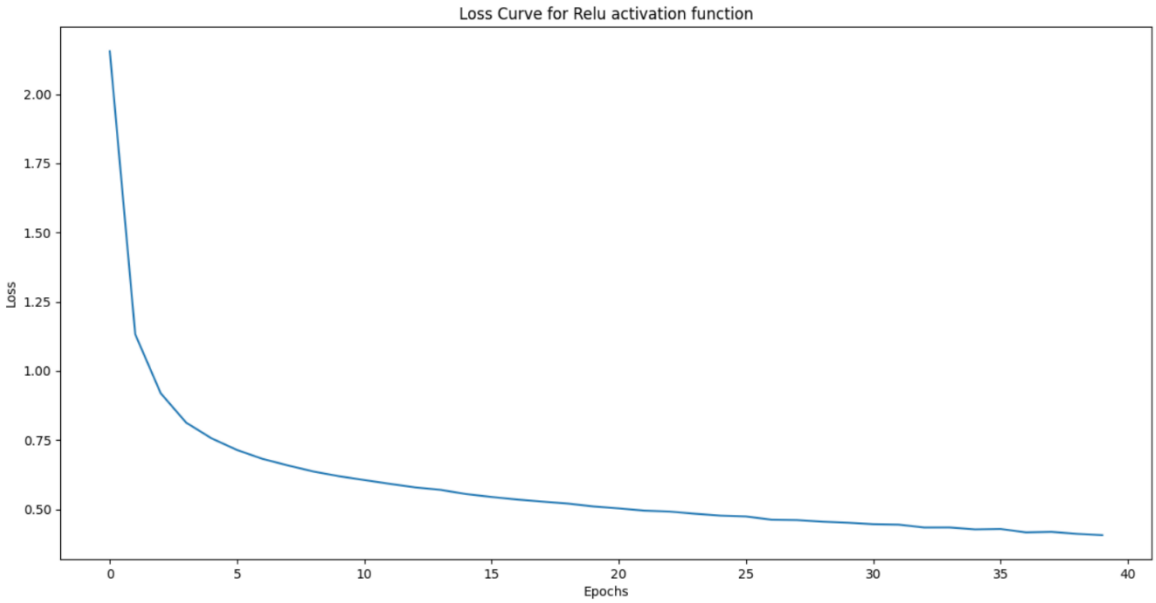
(128,16):


Loss Curve for Relu activation function
Loss
Epochs
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0
10
20
30
40

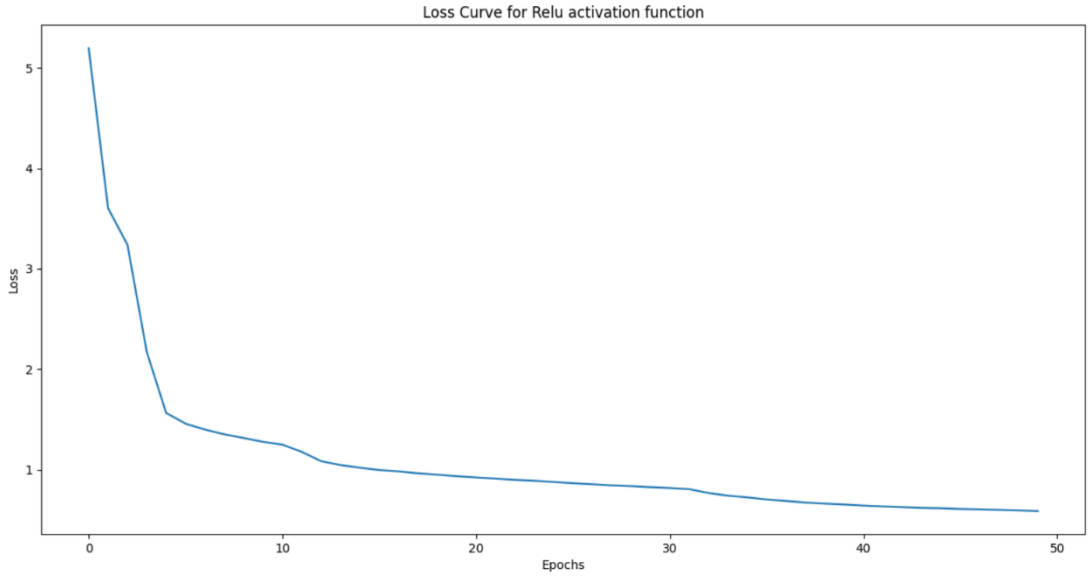
(128,8):


Loss Curve for Relu activation function
Loss
Epochs
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0
5
10
15
20
25
30
35
40

(64,8):



(32,4):



As it can be clearly seen from the loss curves that as the number of neurons in each layer is decreasing the accuracy of the model is also decreasing. This is because with the decrease in the number of neurons the classification of the model becomes less extensive thus not classifying all features more accurately as per their weightage in the model, thus making it less accurate.

4. Up to 10 iterations:

```
  warnings.warn(
0.8527777777777777
{'activation': 'tanh', 'alpha': 0.01, 'hidden_layer_sizes': 14, 'max_iter': 10, 'solver': 'adam'}
```

Up to 50 iterations:

```
  warnings.warn(
0.8714444444444445
{'activation': 'tanh', 'alpha': 0.1, 'hidden_layer_sizes': 15, 'max_iter': 50, 'solver': 'adam'}
```

Up to 80 iterations:

```
0.8696666666666667
{'activation': 'tanh', 'alpha': 0.1, 'hidden_layer_sizes': 15, 'max_iter': 80, 'solver': 'adam'}
```

From the above results we can see that for larger number of iterations 0.1 learning rate is good. However for all of them tanh is the best activation function, with 15 as size of hidden layer and adam be the solver.

We are getting tanh as the best activation function as it is a one of the best performing activation function and maximum iteration are always the desired number of iterations because more the number of iterations, more accurate the model becomes.