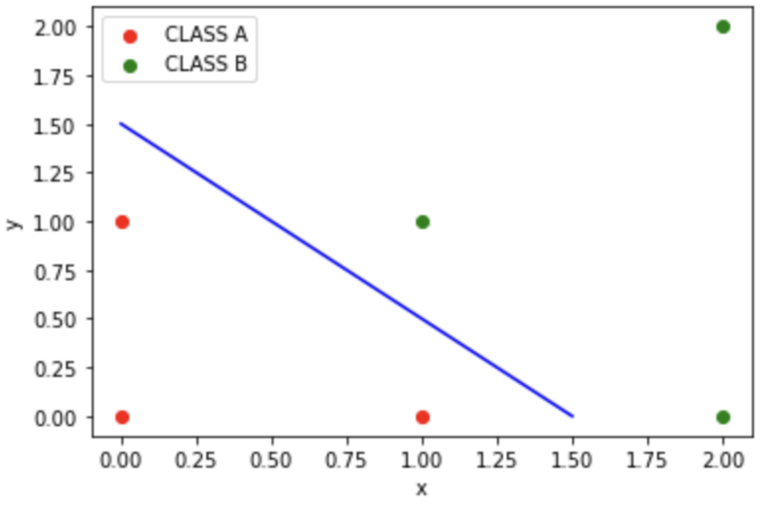
SECTION A

1.)



The above graph shows that the points in classes A and B are linearly separable by an infinite number of decision boundaries.
In this example, the best optimal hyperplane is evident because the shortest distance between the two closest points from different classes is one unit.
→ between (1,0) and (1,1), between (0,1) and (1,1) etc.

2.) We know that the equation of the line is y-y1 = [(y1-y2)/(x1-x2)] * (x-x1) where (x1,y1) & (x2,y2) are the midpoints of any two closest points of two different classes. Suppose we consider (x1,y1)=(1,0.5)[mid-point of (0,1) and (1,1)] and (x2,y2)=(1,0)[the midpoint of (1,0) & (2,0)], after simplification, the equation of the optimal hyperplane turns out to be **x + y - 1.5 = 0.** In this case, (0,1), (1,1),(1,0), and (2,0) become the support vectors as these vectors are the closest to the line from different classes. From the equation of the line which we got above, we can see that the weight vector comes out to be $(1, 1)^T$ as the coefficients of x and y in the line is

3.) As we can see, there are 4 support vectors. Each support vector is 0.5 units away from the hyperplane, and the margin becomes 1 unit as the distance between the nearest points is one unit. Suppose we remove (1,1) or (1,0) from the particular dataset. In that case, the margin will increase, while on the other hand, if we remove the other two, the optimal decision boundary will be as it is. There will be no change in the optimal hyperplane, so the margin will remain unchanged.We can verify the same by calculating the value of the distance from all the support vectors by the formula of the distance of a point from a line, i.e., d =[|Ax1 + By1 + C|]/ √(A2 + B2).

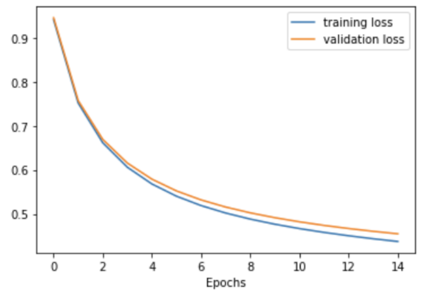
4.)We get an optimal value at least as good as the previous one when we remove some constraints from a constrained maximisation problem. This is because the set of candidates satisfying the initial (more significant, more robust) set of constraints is a subset of the candidates fulfilling the current (smaller, weaker) set of constraints. The old optimal solution is still available, and there may be better alternatives. In SVM, we maximise the margin while keeping the constraints imposed by training points in mind. Depending on the dataset, dropping any constraints can cause the margin to increase or remain the same. Generally, in the case of realistic datasets, it is expected that the margin will increase when we drop support vectors. But in this data, we saw that removing or keeping the support vectors can increase or preserve the margin unchanged, as we saw above.

SECTION C

1.)

For tanh:

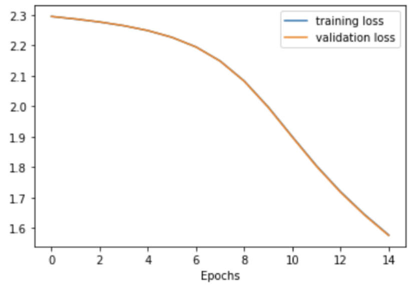
```
For  tanh  Activation Function:
```



```
Training set accuracy score(in %):  84.71960784313725
Validation set accuracy score(in %):  84.04444444444444
Test set accuracy score(in %):  83.43

******************************
```

For sigmoid:

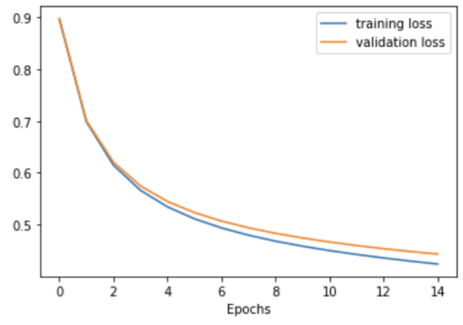
```
For  logistic  Activation Function:
```



```
Training set accuracy score(in %):  51.76274509803922
Validation set accuracy score(in %):  52.022222222222226
Test set accuracy score(in %):  52.59
******************************
```

For relu:

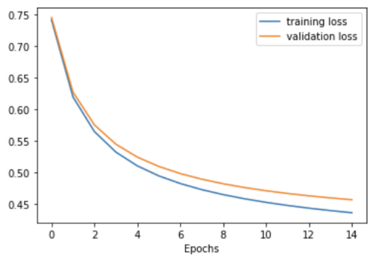
```
******************************

For  relu  Activation Function:
```



```
Training set accuracy score(in %):  85.31176470588235
Validation set accuracy score(in %):  84.66666666666667
Test set accuracy score(in %):  83.89999999999999

******************************
```

For identity:

```
For  identity  Activation Function:
```



```
Training set accuracy score(in %):  84.97450980392158
Validation set accuracy score(in %):  84.41111111111111
Test set accuracy score(in %):  83.39
```

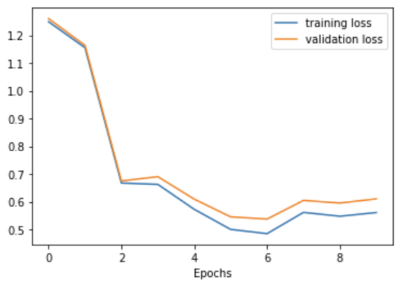
We can see from the accuracy score that relu activation function has performed the best for the given data set as it has the highest test set accuracy i.e. 83.89999999999999%

Relu performs the best as it's computational cost is the lowest as it just have to calculate max(0,x) while on the other hand function like sigmoid have to bear expensive exponential operations,similarly it is also better than tanh as the drawbacks like Vanishing Gradient Problem is completely removed in this activation function.It is also better than the linear activation function as the linear just returns the same x which is the input and the gradient is always 1 in every case which might not be the best.

2.)

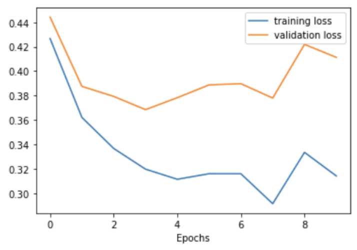
We get the best accuracy at learning rate =0.001 as we know when the learning rate is small it converges towards the minima in each iteration but it may be slow,while bigger learning rate leads to divergence and it sometimes can go even far away by just skipping the global minima. Hence we got the best results at the lowest learning rate. But reducing the learning rate further can lead to overfitting of data and also it takes alot of time to converge so choosing an optimal learning rate is essential

```
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

For learning Rate:  0.1
```



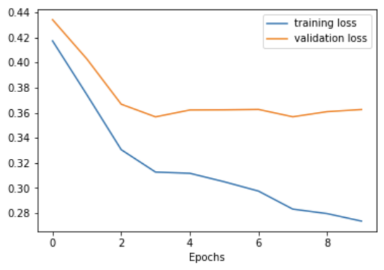
```
Training set accuracy score(in %):  82.35882352941177
Validation set accuracy score(in %):  81.38888888888889
Test set accuracy score(in %):  80.71000000000001
```

```
For learning Rate:  0.01
```



```
Training set accuracy score(in %):  88.60588235294118
Validation set accuracy score(in %):  86.34444444444445
Test set accuracy score(in %):  85.8

******************************
```

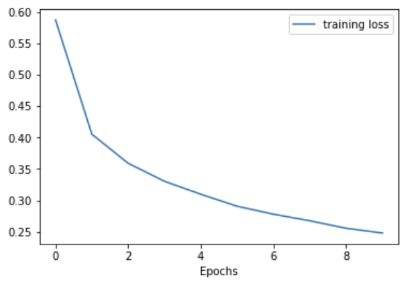
```
For learning Rate:  0.001
```



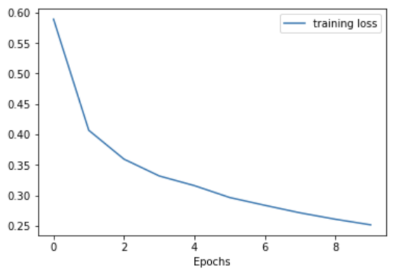
```
Training set accuracy score(in %):  89.78823529411764
Validation set accuracy score(in %):  87.5
Test set accuracy score(in %):  86.61999999999999

******************************
```

3.)

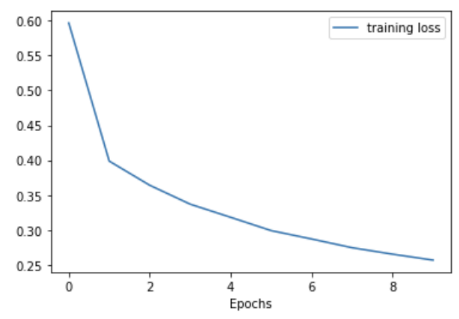
For 245 and 30



For 210 and 25



For 190 and 20



If we increase the number of neurons in hidden layers and the number of hidden layers, the network's complexity and efficiency may increase depending on the problem.

The model becomes more adaptable, allowing it to learn finer details. This may sometimes lead to overfitting. And our classifier might not perform better on the 'next dataset.' As the model becomes more susceptible to over-fitting, the generalisation of your classification model may suffer.

I observed the highest accuracy at hidden layer 245,30. The more the numbers of neurons in the hidden layer the more the accuracy as the model is trained well as it increases the recognition rate.,similarly in this data set the best accuracy was observed when the number of neurons was the highest. But increasing the number of neurons doesnt help everytime as more neurons cause the increase in risk of over fitting so it is important to choose the most optimal number of neurons in a hidden layer.

4.) When i ran grid search i got this output → The most optimal parameters observed are : {'activation': 'relu', 'alpha': 0.001, 'hidden_layer_sizes': (245, 30), 'learning_rate': 'adaptive', 'solver': 'adam'}

As i explained above relu is much faster and less complex type of activation function that is why we got relu as our best activation function

The best learning rate comes out to be 0.001 that is also explained above. We know that a smaller learning rate might slows down the process of learning but generally it increases accuracy of the model

The highest number of neurons which we took came out to be the most optimal parameters as explained above. This increases accuracy as the complexity is increased and hence the accuracy is also increased.

The adaptive learning rate is the best in this data set.
Adaptive learning rates can help to accelerate training and relieve problem associated with selecting a learning rate and learning rate schedule. The learning rate can increase or decrease depending on what we need to achieve the minima.

Adam optimiser is the best in this data set. The Adam optimizer produces better results than other optimization algorithms in general because it computes faster and requires fewer tuning parameters.