Machine Learning Assignment 3
-Pankil Kalra, 2018061

Question 1

Neural network from scratch was implemented in the Q1.py file.

Question 2

a)

Relu:

```
Training Accuracy for Relu: 1.0
Testing Accuracy for Relu: 0.9734
```

Test Accuracy for Relu: 0.9734

Sigmoid

```
Training Accuracy for Sigmoid: 0.9451
Testing Accuracy for Sigmoid: 0.9349
```

Test Accuracy for Sigmoid: 0.9349

Linear:

```
Training Accuracy for Linear: 0.92615
Testing Accuracy for Linear: 0.9181
```

Test Accuracy for Linear: 0.9181

Tanh:

```
Training Accuracy for Tanh: 0.9599166666666666
Testing Accuracy for Tanh: 0.9542
```

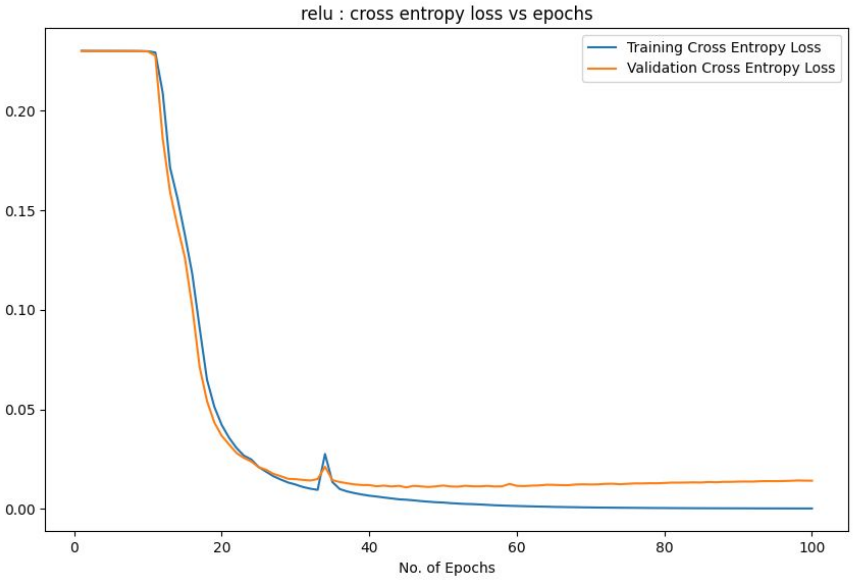
Test Accuracy for Tanh: 0.9542

Relu activation function gace the best test accuracy.

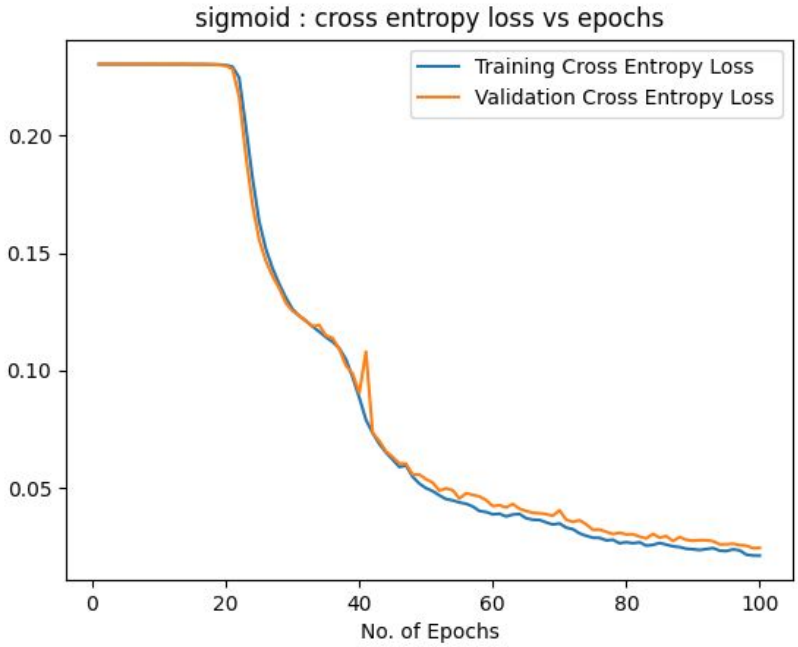
The weights and biases for the models are saved in the Weights/ folder.

b)

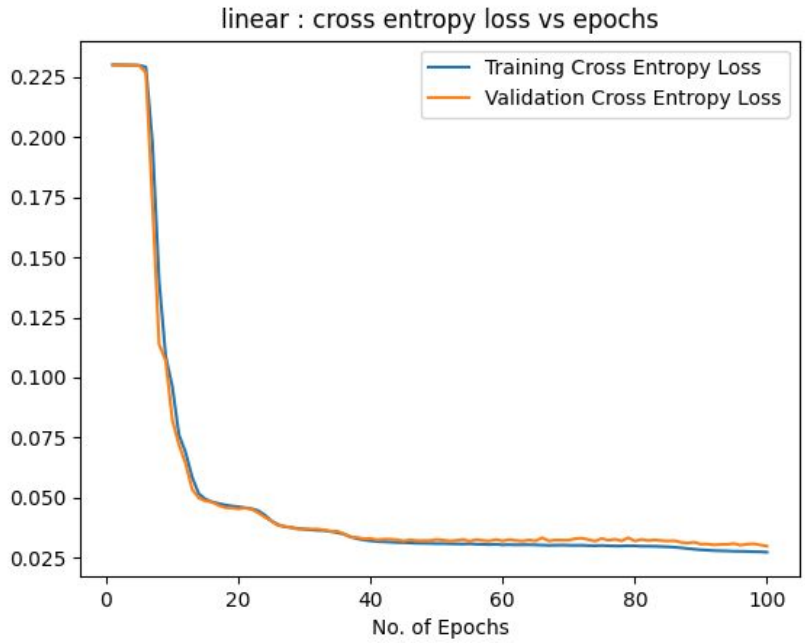
Relu:



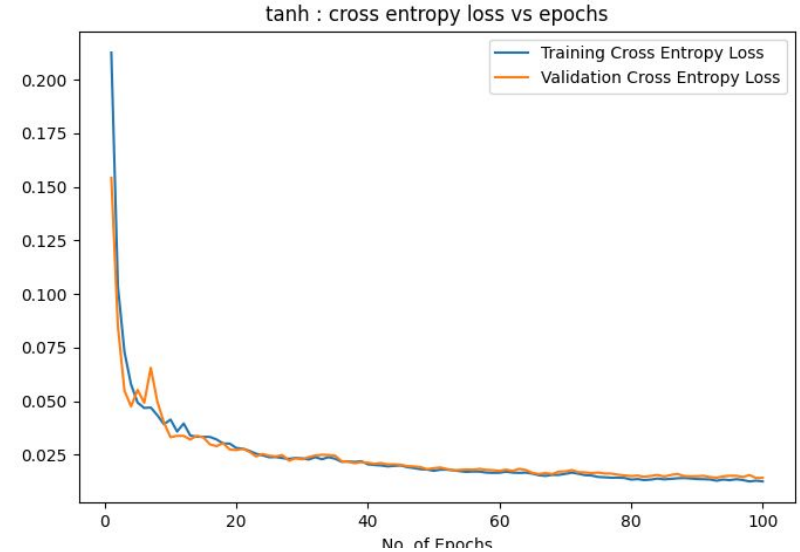
Sigmoid



Linear:


linear : cross entropy loss vs epochs
Training Cross Entropy Loss
Validation Cross Entropy Loss
0.225
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0
20
40
60
80
100
No. of Epochs

Tanh:


tanh : cross entropy loss vs epochs
Training Cross Entropy Loss
Validation Cross Entropy Loss
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0
20
40
60
80
100

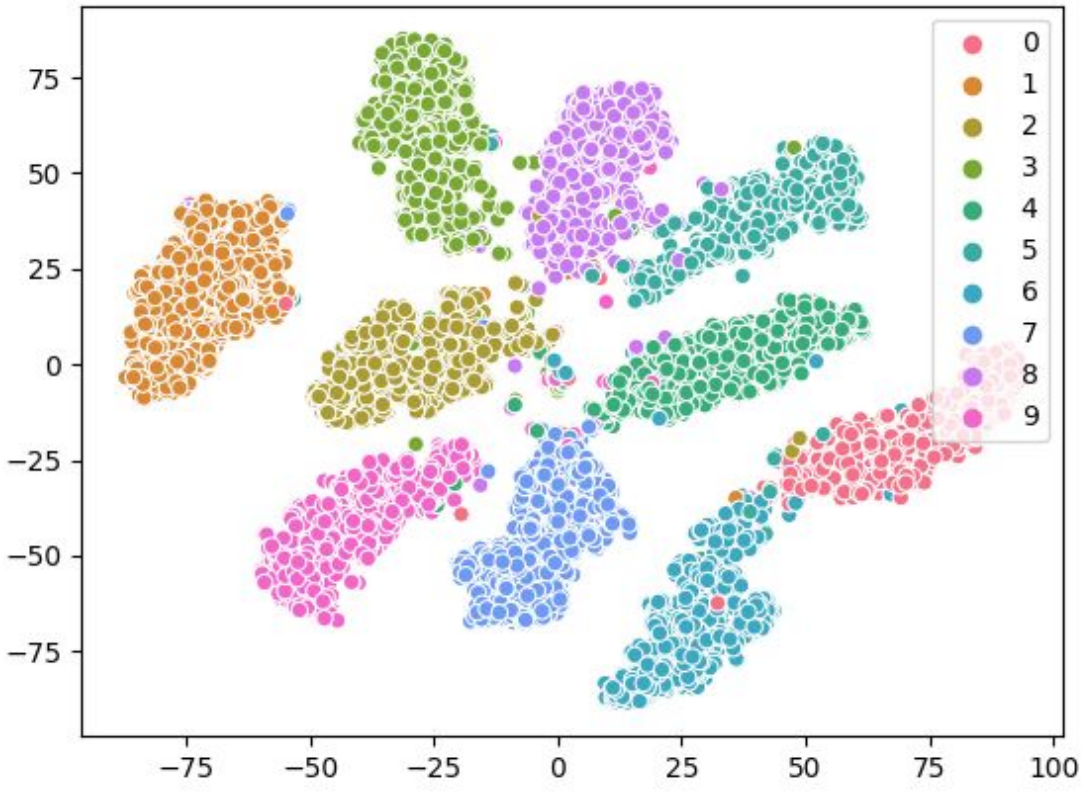
c) In every case, **softmax** softmax activation function should be used for the output layer. We are dealing with multiclass classification. Before applying softmax, we get the outputs of different classes from 0-1 but all these outputs do not sum to 1. We use softmax on the output layer, so that the final values given by our models simulate a probability distribution. That is, it gives us the probability that the data point belongs to each class and all these probabilities add to 1.

d)

The total number of layers is 5, while the number of hidden layers is 3.

The number of neurons in the 3 hidden layers are 256, 128 and 64 respectively.

e)
Relu activation function gave the highest test accuracy. Below is the TSNE plot for the given by the final hidden layer(with 64 neurons) on test data.



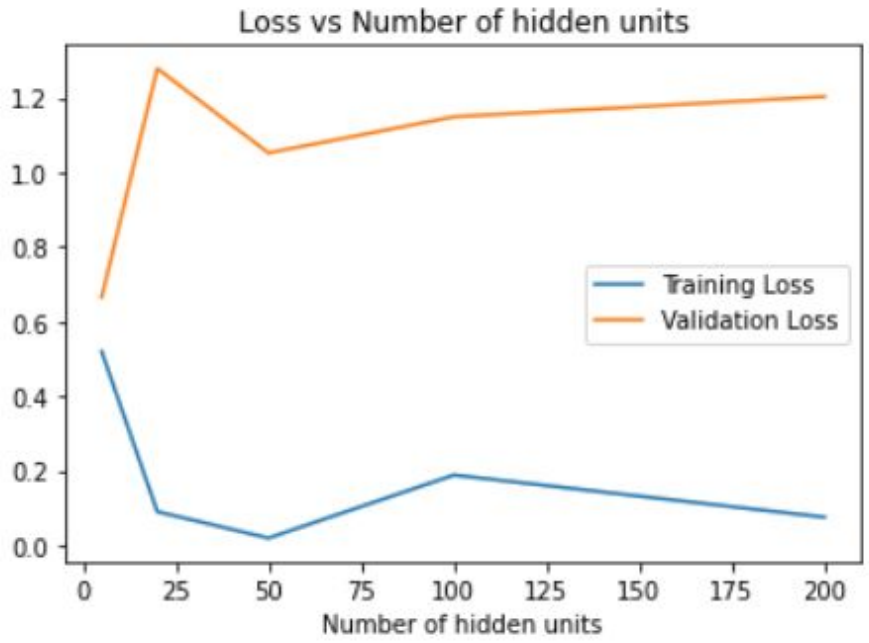
f)

```
Sklearn with Relu Testing Accuracy: 0.9822
Sklearn with Sigmoid Testing Accuracy: 0.9194
Sklearn with Linear Testing Accuracy: 0.8905
Sklearn with Tanh Testing Accuracy: 0.7269
```

With the **same parameters,** the test accuracies of relu, sigmoid and linear activation based MLP Classifier was similar to what my implementation achieved except for tanh where the sklearn implementation had a lower accuracy than my implementation.

Question 3

1.a)



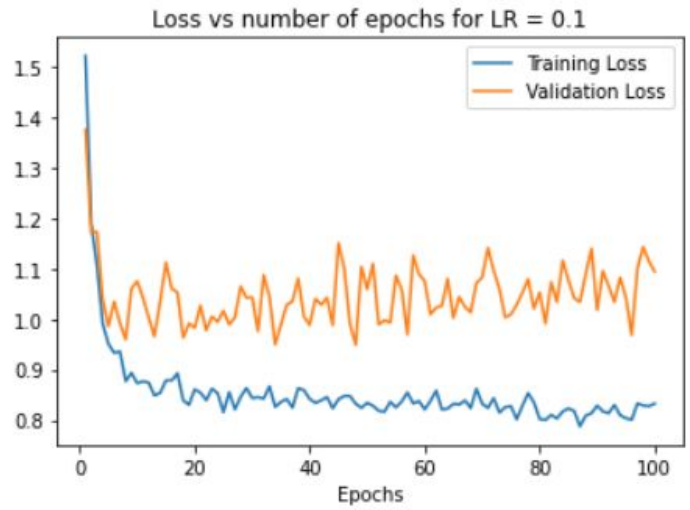
1.b)

Clearly as the number of hidden units increases, the validation loss first decreases then remains constant while the training loss keeps decreasing then remains close to zero.

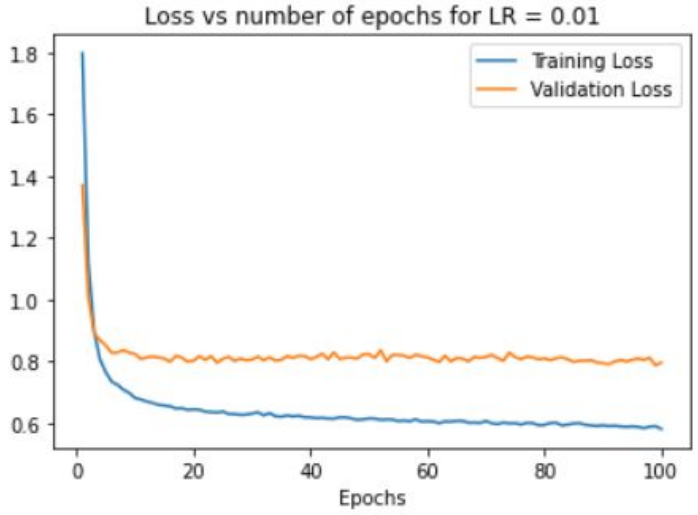
The constant nature of validation shows that the model has overfitted on the training data. The model does not perform on the validation set as it does on the training set. This shows that the model has high variance.

2. a)

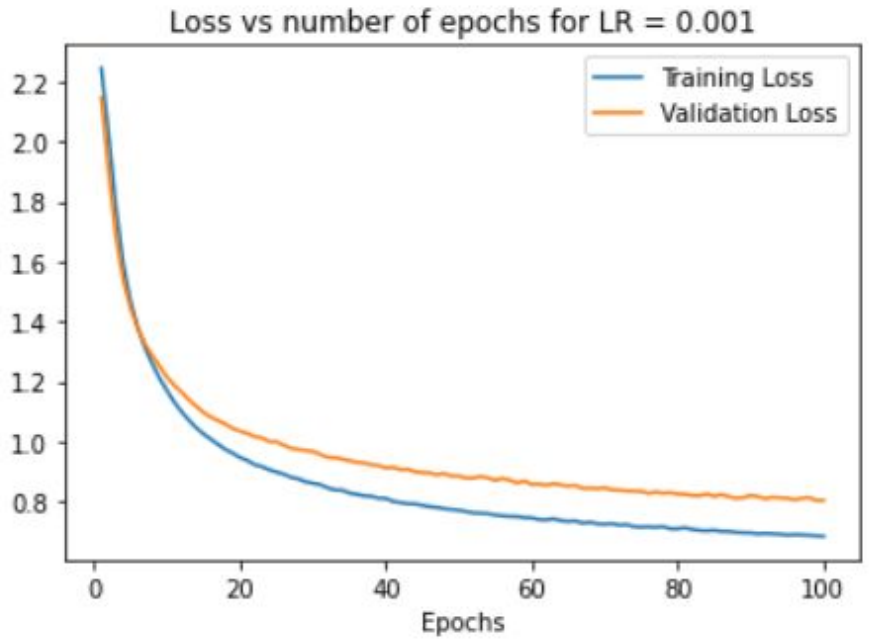
LR = 0.1



LR = 0.01



LR = 0.001



2. b)
Observations:

i) LR = 0.1
The loss plot was very unstable and was spiky in nature. The model did not converge and was very unstable in nature(visible by the validation plots). There was a huge difference between the training and validation losses.

ii) LR = 0.01

The plot was fairly smooth and stable with little spikes in the loss here and then. It was able to converge easily within 100 epochs(at around 30-40 epochs).

iii) LR = 0.001

The loss plots of this model were the smoothest plots out of the 3 learning rates. At the end 100 epochs, it can be clearly seen that the model was about to converge in the next 20-30 epochs.

0.01 was the best learning rate.

Summarising the convergence for the 3 learning rates, at 0.1 LR the model was unstable and did not converge, at 0.01 LR the model converged easily within 100 epochs, while at 0.001 LR the model is about to converge at 100 epochs.

Question 4

1)
Class distribution

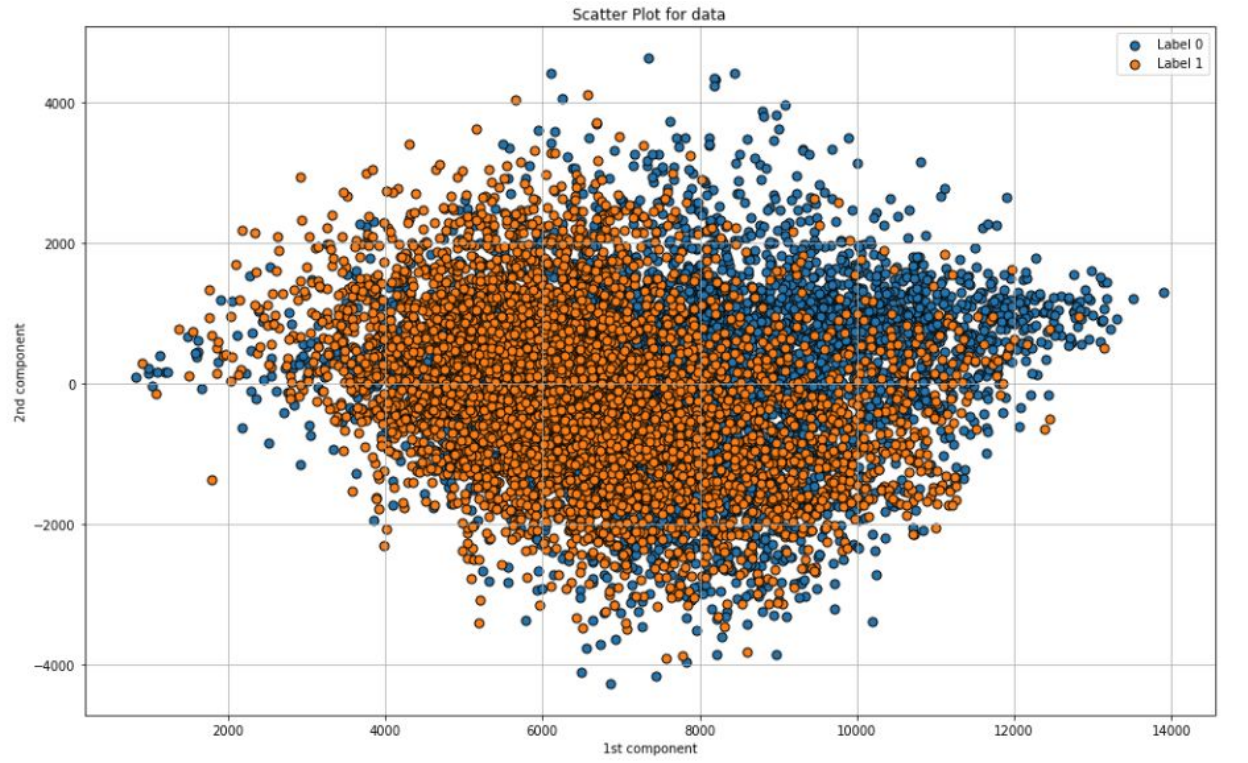
```
[11] np.unique(y_train, return_counts = True)

     (array([0, 1]), array([5000, 5000]))
```

Clearly, the dataset had equal class distribution of the two labels.

Scatter Plot:

I used SVD to reduce the dataset to two dimensions and then visualised them using a scatter plot. Clearly the dataset is well distributed across 2 dimensions.



2) I used the pretrained AlexNet model from Pytorch and extracted features for the images of the CIFAR dataset. Now my datapoint has 1000 features.

3) Trained a Neural Network with 2 hidden layers of sizes 512 and 256 neurons respectively.

Test accuracy: 0.9096

```
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')

model = model.to(device)
AlexNet.to(device)
criterion = criterion.to(device)
num_epochs = 1


for i in range(num_epochs):
  train_loss, train_acc = train(model, train_iterator, optimizer, criterion, device)
  val_loss, val_acc = evaluate(model, test_iterator, criterion, device)
  print("Train acc: ", train_acc, " Test acc: ", val_acc)
```

```
1000 512
512 256
256 2
Train acc:  0.8597730891719745  Test acc:  0.90966796875
```