We saw an implementation of MLP in numpy and PyTorch. You may have noticed that the weights are initialized randomly. What happens if we we set all weights and biases to 0

Assume we are using ReLU activation

(A) **[Ans]** Weights do not change while training

(B) No problem: the model will converge nicely

(C) Overfitting issue

(D) Underfitting issue

(E) None of these

We saw this implementation of MLP model in PyTorch:

```
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        # Initialize all the layers with learnable parameters
        self.fc1 = nn.Linear(2, 2, True)
        self.fc2 = nn.Linear(2, 1, True)

    def forward(self, x):
        # Write the forward pass
        x = self.fc1(x)
        x = torch.sigmoid(x)
        x = self.fc2(x)
        x = torch.sigmoid(x)
        return x
```

The output of this model is

(A) **[Ans]** Always greater than zero

(B) **[Ans]** Always less than one

(C) Can be negative as well as positive

(D) Always zero

(E) None of these

We saw this implementation of MLP in PyTorch:

```
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        # Initialize all the layers with learnable parameters
        self.fc1 = nn.Linear(2, 2, True)
        self.fc2 = nn.Linear(2, 1, True)

    def forward(self, x):
        # Write the forward pass
        x = self.fc1(x)
        x = torch.sigmoid(x)
        x = self.fc2(x)
        x = torch.sigmoid(x)
        return x
```

What happens if we remove the 2 lines with code x=torch.sigmoid(x) in the forward function

Assume that we are working with the XOR data as given in the notebook shared.

(A) Syntax error

(B) Math error

(C) **[Ans]** Model becomes Single layer perceptron

(D) Model remains multi layer perceptron

(E) None of these

We saw the implementation of MLP in PyTorch with XOR data. What happens if we add 10 more hidden layers with 100 weights each with non-linear activation and train the model till loss is minimized.

(A) Results in the same decision boundary

(B) **[Ans]** Results in a different decision boundary but still able to classify all 4 samples correctly.

(C) Can not classify all 4 samples correctly.

(D) None of these

Suppose instead of XOR data, we now want to work on NAND data. Model 1 is a MLP with a hidden layer with 2 neurons as we saw. Model 2 is a SLP.

(A) Model 1 can classify all 4 samples correctly but not model 2

(B) Model 2 can classify all 4 samples correctly but not model 1

(C) **[Ans]** Both model 1 and model 2 can classify all 4 samples correctly.

(D) Neither model 1 nor model 2 can classify all 4 samples correctly.

(E) None of these