1. Build a MLP to learn regression *y*=*f*(*x*)

|  |  |  |  |
| --- | --- | --- | --- |
| **Training data** | x | 100 equally spaced points between -1 and 1. | |
| y | *y* = *x*3 + 2*x*2 + 0.3*b*. | |
| **NN design** | Hidden layers | |  |
| Activation function | |  |
| Loss function | |  |
| Optimization function | |  |
| Learning rate | |  |
| **Train** | epochs | |  |

1. Run one epoch step by step.

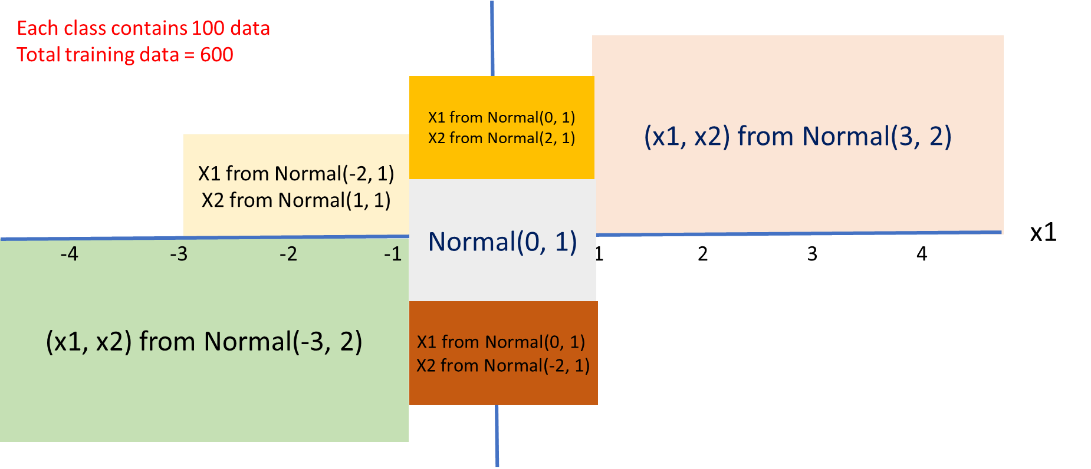
|  |
| --- |
| Forward propagation to calculate loss |
| Print loss |
| Backward propagation to update weights |

2) Use loop to run *N* epochs.

* Animate the learning process.
* Print out loss value.
* Plot loss curve after the loop.

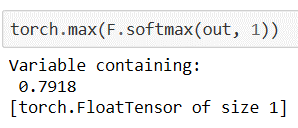
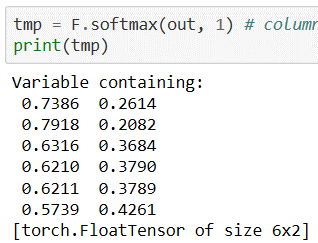
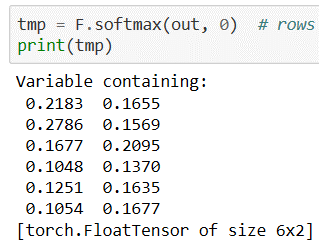
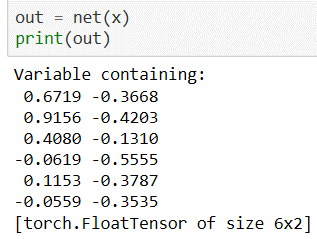
1. Build a MLP to learn classification

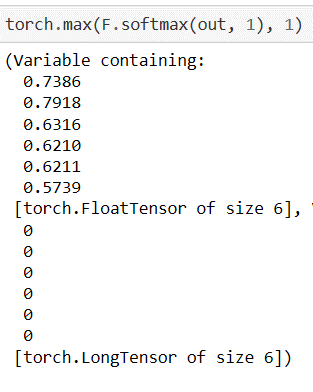
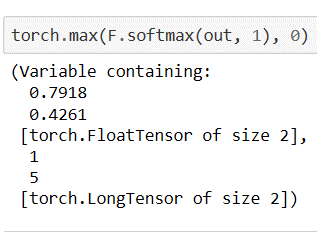
|  |  |  |  |
| --- | --- | --- | --- |
| **Training data** | X | 6 classes of 2-d data | |
| Y | 0, 1, 2, 3, 4, 5, 6 | |
| **NN design** | Hidden layers | |  |
| Activation function | |  |
| Loss function | |  |
| Optimization function | |  |
| Learning rate | |  |
| **Train** | epochs | |  |

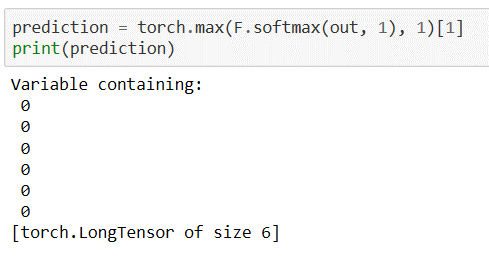


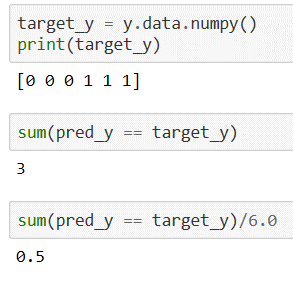
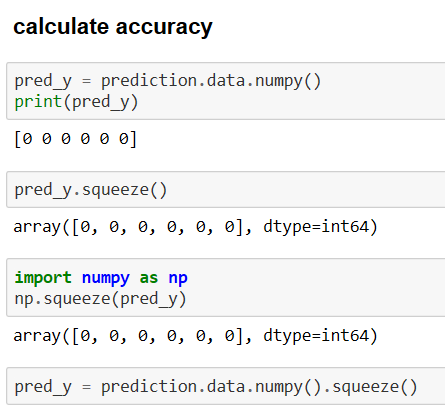
1. Run one epoch step by step

|  |
| --- |
| Forward propagation to calculate loss |
| Print loss |
| Use Softmax to convert output to label 0~5 |
| Calculate accuracy and print |
| Backward propagation to update weights |









2) Use loop to run *N* epochs.

* Animate the learning process.
* Print loss and accuracy values during loop.
* Plot loss curve after loop.
* Plot accuracy curve after loop.

1. Save/Load NN

* Save the weight of an NN
* Save the structure and weights of an NN
* Load the structure and weights of an NN
* Load the weights of an NN.

1. Repeat problem 2, practice mini-batch

|  |  |  |  |
| --- | --- | --- | --- |
| **Training data** | X | 6 classes of 2-d data | |
| Y | 0, 1, 2, 3, 4, 5, 6 | |
| **NN design** | Hidden layers | |  |
| Activation function | |  |
| Loss function | |  |
| Optimization function | |  |
| Learning rate | |  |
| **Train** | Epochs | |  |
| Batch size | |  |

1. Run one batch step by step.

|  |
| --- |
| Forward propagation to calculate loss |
| Print loss |
| Use Softmax to convert output to label 0~5 |
| Calculate accuracy and print |
| Backward propagation to update weights |

1. Build a MLP to learn classification

|  |  |  |  |
| --- | --- | --- | --- |
| **Training data** | X | The 25,000 cat and dog image files from Kaggle: <https://www.kaggle.com/c/dogs-vs-cats/data>. | |
| Y | Cat = 0, dog =1 | |
| **NN design** | Hidden layers | |  |
| Activation function | |  |
| Loss function | |  |
| Optimization function | |  |
| Learning rate | |  |
| **Train** | Epochs | |  |
| Batch size | |  |

1. Run one batch step by step.

|  |
| --- |
| Forward propagation to calculate loss |
| Print loss |
| Use Softmax to convert output to label 0 and 1 |
| Calculate accuracy and print |
| Backward propagation to update weights |

2) Use loop to run *N* epochs, for each epoch, the batch size = *M*

* Print loss and accuracy values during each batch.
* Plot loss curve after loop.
* Plot accuracy curve after loop.