1.
```
a = torch.Tensor( [ [2,3], [4,8], [7,9] ] )
numpy_a = a.numpy()
b = np.array([2,3], [4,5])
torch_b = torch.from_numpy(b)
```

transform between numpy and torch, give us more possibilities and flexibilities
more importantly, The memory is shared between both. I can change the values in-place of one object, then the other will change as well.

2.
torch.sum(input, dim)
we can specify the dimension equal 0 or equal 1 according to the axis(rows/columns) that we want to go across.

3.
torch.view(a, b), parameters = -1
=> powerful, smart, automatically calculate the dimensions that we need.

4. .topk(k, dim=1),
when we get prediction_outputs, we could use .topk(k, dim=1), return the k highest values and its classes, which allows us to use it to compare with the true labels, to measure the performance of our alogorithm

5.
torch.save(model.state_dict(), 'file.name')
state_dict() return a dictionary object that contain the mapping relations, parameters like weights, and bias, then we could save them in a file, which we can load it next time without calculating again.