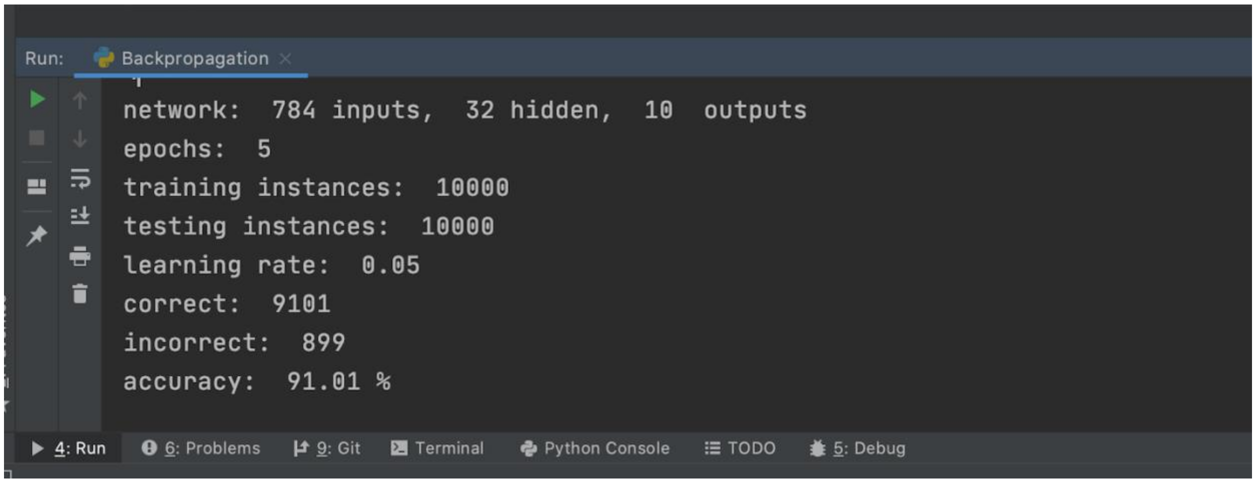
Vitaliy Stepanov
Assignment 3
Machine Learning

The topology of my network consists of 1 layer of input nodes on the left, 1 layer of hidden nodes in the middle, and 1 layer of output nodes on the right. Only the number of hidden nodes may be modified. There are weights going from input to hidden layer and weights going from hidden to output layer. The input and hidden layer both contain a bias. The function propagate forward goes through the network from left to right one image at a time. The function propagate backward calculates the error of the outputted images compared to the actual target label then the function update weights updates the weights from input to hidden and from hidden to output. Then it repeats. A testing function is used on never before seen images. Many fields are global and on top to easily change parameters.

The parameters I used for training are epochs, learning rate, network hidden, training instances, and testing instances. When I doubled the hidden nodes from 16 to 32, my accuracy went from 87% to 91%. I also tried 64 hidden nodes but with half the training data for 89%. Most time-consuming test was increasing epochs to 10, changing learning rate from 0.05 to 0.02, 32 hidden nodes, and 10k training and 10k testing which surprisingly resulted in only 89% accuracy. It seems like making the learning rate lower made accuracy worst in that one instance. I use numpy data structure to speed up computation and can train in 10 minutes with an accuracy of 91%.



```
network:  784 inputs,  32 hidden,  10  outputs
epochs:  5
training instances:  10000
testing instances:  10000
learning rate:  0.05
correct:  9101
incorrect:  899
accuracy:  91.01 %
```