Goal: Classifying Fashion-MNIST, Lesson 5.12, 5.13

- Load dataset from torchversion
- Define network architecture
    - Use nn.Module
        - Define __init__
        - Define forward
    - Or use nn.Sequential
- Create network
- Define criterion
- Define optimizer
    - Adam optimizer same as stochastic gradient descent but has nice property that use momentum which speed up the training / fitting process and it adjust learning rate for each individual parameter in model
- Train the network number of x time; x is called epochs
- Calculate the class probabilities (softmax) for image

Part 4 - Fashion-MNIST (Exercises).ipy...

File Edit View Insert Runtime Tools Help All chang

Comment

+ Code + Text

RAM Disk

```
[14] Training loss: 0.5116642041247028
     Training loss: 0.3925563087547893
     Training loss: 0.35695936826309926
     Training loss: 0.3329488458489177
     Training loss: 0.3165316684032554
```

```
%matplotlib inline
%config InlineBackend.figure_format = 'retina'

import helper

# Test out your network!

dataiter = iter(testloader)
images, labels = dataiter.next()
img = images[0]
# Convert 2D image to 1D vector
img = img.resize_(1, 784)

# TODO: Calculate the class probabilities (softmax) for i
ps = torch.exp(model(img))

# Plot the image and probabilities
helper.view_classify(img.resize_(1, 28, 28), ps, version=
```