```
import torch
from torch.autograd import Variable


x_data = Variable(torch.Tensor([[1.0], [2.0], [3.0]]))
y_data = Variable(torch.Tensor([[2.0], [4.0], [6.0]]))
class LinearRegressionModel(torch.nn.Module):

    def __init__(self):
        super(LinearRegressionModel, self).__init__()
        self.linear = torch.nn.Linear(1, 1)  # One in and one out

    def forward(self, x):
        y_pred = self.linear(x)
        return y_pred
our_model = LinearRegressionModel()


criterion = torch.nn.MSELoss(size_average = False)
optimizer = torch.optim.SGD(our_model.parameters(), lr = 0.01)
```

```
/usr/local/lib/python3.7/dist-packages/torch/nn/_reduction.py:42: UserWarning: size_average and reduce args will be deprecated, please use reduction='sum' instead.
  warnings.warn(warning.format(ret))
```

- Perform a forward pass bypassing our data and finding out the predicted value of y.
- Compute the loss using MSE.
- Reset all the gradients to 0, perform a backpropagation and then, update the weights.

```
for epoch in range(500):

    # Forward pass: Compute predicted y by passing
    # x to the model
    pred_y = our_model(x_data)

    # Compute and print loss
    loss = criterion(pred_y, y_data)

    # Zero gradients, perform a backward pass,
    # and update the weights.
    optimizer.zero_grad()
    loss.backward()
    optimizer.step()
    print('epoch {}, loss {}'.format(epoch, loss.item()))
```

```
epoch 442, loss 0.0004222563584335148
epoch 443, loss 0.00041618343675509095
epoch 444, loss 0.00041020161006599665
epoch 445, loss 0.00040430985973216593
epoch 446, loss 0.0003984967479482293
epoch 447, loss 0.000392764835851267
epoch 448, loss 0.0003871191001962870
epoch 449, loss 0.000381567602744326
epoch 450, loss 0.0003760764666367322
epoch 451, loss 0.0003706721472553909
epoch 452, loss 0.0003653437015600502
epoch 453, loss 0.0003600890631787479
epoch 454, loss 0.000354915246134623
epoch 455, loss 0.0003498234436847269
epoch 456, loss 0.000344479759051464
epoch 457, loss 0.0003398359112907201
epoch 458, loss 0.000334954173556730
epoch 459, loss 0.0003301441611256450
epoch 460, loss 0.0003253985487390309
epoch 461, loss 0.0003207206318620592
epoch 462, loss 0.000316161086351610
epoch 463, loss 0.0003115608415100723
epoch 464, loss 0.000307082664221525
epoch 465, loss 0.000302673317492008
epoch 466, loss 0.000298320956062525
epoch 467, loss 0.000294035009574145
epoch 468, loss 0.0002898028760682791
epoch 469, loss 0.0002856475475709885
epoch 470, loss 0.0002815339830704033
epoch 471, loss 0.000277488987194374
epoch 472, loss 0.000273499201284721
epoch 473, loss 0.0002695707953533502
epoch 474, loss 0.0002657030127011239
epoch 475, loss 0.0002618857542984187
epoch 476, loss 0.000258122279774397
epoch 477, loss 0.0002544092712923884
epoch 478, loss 0.0002507571480236947
epoch 479, loss 0.000247150688664987
epoch 480, loss 0.000243596645304933
epoch 481, loss 0.0002400971716269850
epoch 482, loss 0.0002366426924709230
epoch 483, loss 0.000233245227718725
epoch 484, loss 0.0002298934705322608
epoch 485, loss 0.0002265922375954687
epoch 486, loss 0.0002233261620858684
epoch 487, loss 0.000220118323341012
epoch 488, loss 0.0002169516665162518
epoch 489, loss 0.0002138462150469422
epoch 490, loss 0.0002107682230416685
epoch 491, loss 0.0002077393291983753
epoch 492, loss 0.000204756521270610
epoch 493, loss 0.000201815084437839
epoch 494, loss 0.0001989105221582576
epoch 495, loss 0.0001960538647836074
epoch 496, loss 0.000193234052858315
epoch 497, loss 0.0001904611417558044
epoch 498, loss 0.0001877249887911602
epoch 499, loss 0.0001850260450737550
```

```
new_var = Variable(torch.Tensor([[4.0]]))
pred_y = our_model(new_var)
print("predict (after training)", 4, our_model(new_var).item())
```

```
predict (after training) 4 7.984364032745361
```

✓ 0s completed at 7:58 PM