Logistic Regression - Demo

Input: Breast Cancer Data from sklearn.datasets

Code:

1. Importing Libraries

```
import torch
import torch.nn as nn
import numpy as np
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

#model
#loss and optimizer
#training loop
```

2. Loading and preparing data

```
#preparing data - taking data from sklearn datasets
bc=datasets.load_breast_cancer()
X,y=bc.data,bc.target   # X contains features, y contains labels 0 or 1

n_samples, n_features=X.shape
print(n_samples, n_features)
#splitting data in sets
# 80% training and 20% testing
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=1234)
```

3. Scaling Features

```
#scaling features
sc=StandardScaler()
X_train=sc.fit_transform(X_train)
X_test=sc.transform(X_test)
```

```
#converting numpy arrays to PyTorch tensors of type float32
X_train=torch.from_numpy(X_train.astype(np.float32))
X_test=torch.from_numpy(X_test.astype(np.float32))
y_train=torch.from_numpy(y_train.astype(np.float32))
y_test=torch.from_numpy(y_test.astype(np.float32))

#reshape y tensor
#Reshape y_train and y_test to be column vectors
y_train=y_train.view(y_train.shape[0],1)
y_test=y_test.view(y_test.shape[0],1)
```

4. Defining the model

```
#setting up model
class LogisticRegression(nn.Module):
    def __init__(self,n_input_features):
        super(LogisticRegression,self).__init__()
        self.linear=nn.Linear(n_input_features,1)

    def forward(self,x):
        #sigmoid - logistic function
        y_pred=torch.sigmoid(self.linear(x))
        return y_pred   #returns 0 or 1

model=LogisticRegression(n_features)   #30 input features, 1 output feature
```

5. Loss and Optimizer

```
#loss and optimizer
criterion=nn.BCELoss()            #binary cross entropy loss
optimizer=torch.optim.SGD(model.parameters(),lr=0.01)  #learning rate is 0.01    #using
stochastic gradient descent
```

6. Training Loop

```
#training loop
num_epochs=100
for epoch in range(num_epochs):
    #forward pass and loss
    y_pred=model(X_train)
```

```
    loss=criterion(y_pred,y_train)

    #backward pass and optimizer step
    loss.backward()

    if (epoch + 1) % 10 == 0:
        print(f'Epoch {epoch + 1} - Before updating: Weights:
{model.linear.weight.data.numpy()}')

    #update weight
    optimizer.step()

    # Print the values of theta (weights) after updating
    if (epoch + 1) % 10 == 0:
        print(f'Epoch {epoch + 1} - After updating: Weights:
{model.linear.weight.data.numpy()}')



    #empty gradient
    optimizer.zero_grad()

    if(epoch+1)%10==0:    #after every 10th step
        print(f'epoch:{epoch+1}, loss={loss.item():.4f}')
```

7. Evaluation

```
with torch.no_grad():
    y_predicted=model(X_test)
    #if y is greater than 0.5 than 1 otherwise 0
    y_predicted_cls=y_predicted.round()
    #calculating accuracy
    acc=y_predicted_cls.eq(y_test).sum()/float(y_test.shape[0])
    print()
    print(f'accuracy={acc.item():.4f}')
```

The code demonstrates a full machine learning pipeline using PyTorch: data loading, preprocessing, model definition, training, and evaluation. It uses logistic regression to classify breast cancer data and achieves this by defining a model class, using loss functions and optimizers, and iterating through training epochs while monitoring the model's performance.

Output:

```
569 30
Epoch 10 - Before updating: Weights: [[-0.17073406 -0.1711105  -0.13568045
-0.02915274 -0.16110763 -0.09371217
   0.0763087  -0.02072659 -0.16975053 -0.0309427  -0.00556845 -0.03620943
   0.05817804  0.1473701   0.16461124  0.1604254   0.10457321 -0.06082732
  -0.11116552 -0.09127568 -0.16230337 -0.06632227 -0.20284383  0.07208515
  -0.12065051 -0.06398452 -0.04981894  0.11550106  0.16690008
-0.01826466]]
Epoch 10 - After updating: Weights: [[-0.17301682 -0.17222755 -0.13801351
-0.03139529 -0.16219965 -0.09567025
   0.07396628 -0.0232381  -0.17072085 -0.03096887 -0.00750027 -0.03602823
   0.05626854  0.14543271  0.16457392  0.15931483  0.10358009 -0.06227916
  -0.11111563 -0.09168877 -0.16476777 -0.06757452 -0.20533884  0.06973836
  -0.12209271 -0.06593622 -0.05205654  0.11287782  0.16553918
-0.01941744]]
epoch:10, loss=0.4870
Epoch 20 - Before updating: Weights: [[-0.19135179 -0.18135822 -0.15672535
-0.04937538 -0.1706881  -0.11100165
   0.0553793  -0.04327925 -0.17825574 -0.03072538 -0.02281404 -0.03443651
   0.04118913  0.13009918  0.16432303  0.15089591  0.09603389 -0.0735964
  -0.11044271 -0.0944775  -0.18462431 -0.07787882 -0.2254004   0.05086967
  -0.13376732 -0.08145928 -0.06994706  0.09181805  0.1544983
-0.02852602]]
Epoch 20 - After updating: Weights: [[-0.19318125 -0.18228605 -0.15858969
-0.05116667 -0.17150615 -0.11249018
   0.05354844 -0.04526543 -0.1789812  -0.03065387 -0.02432154 -0.03426554
   0.03971044  0.128592    0.16430229  0.15010852  0.09532756 -0.07468774
  -0.11034843 -0.09469938 -0.18661195 -0.0789325  -0.22740448  0.0489844
  -0.13494076 -0.08299085 -0.0717217   0.08971956  0.15338622 -0.0294174
]]
epoch:20, loss=0.4129
Epoch 30 - Before updating: Weights: [[-0.20822072 -0.19003777 -0.17389622
-0.06587987 -0.17802043 -0.12441476
   0.03867775 -0.06149107 -0.18475068 -0.02971203 -0.03658122 -0.03279112
   0.02772437  0.11634479  0.16415033  0.14403625  0.08988965 -0.08335247
  -0.10939056 -0.09608827 -0.20299526 -0.08777841 -0.24389474  0.03346243
  -0.14463726 -0.09543103 -0.08620635  0.07251409  0.14418341
-0.03659105]]
Epoch 30 - After updating: Weights: [[-0.20975463 -0.19084115 -0.17545538
-0.06737968 -0.17866327 -0.12559828
   0.03717995 -0.06313524 -0.18531898 -0.02957915 -0.03781933 -0.0326346
   0.02651779  0.11510852  0.16413592  0.14345896  0.08937427 -0.08420321
  -0.10927507 -0.09618411 -0.2046705  -0.08869925 -0.24557805  0.03187664
  -0.14563069 -0.09668304 -0.08767146  0.0707653   0.14323947
-0.03730547]]
epoch:30, loss=0.3643
```

```
Epoch 40 - Before updating: Weights: [[-0.22257762 -0.19765273 -0.18847445
-0.07991655 -0.18387742 -0.1352407
   0.02480417 -0.07679739 -0.18991724 -0.02818853 -0.04809534 -0.03128986
   0.01653086  0.10484661  0.16401388  0.13895679  0.08537385 -0.09105352
  -0.10818789 -0.09663111 -0.21870504 -0.09653374 -0.2596591   0.01859735
  -0.15396243 -0.10700937 -0.09981182  0.05620527  0.13531868
-0.04313461]]
Epoch 40 - After updating: Weights: [[-0.22390564 -0.1983681  -0.18982124
-0.08121511 -0.18440105 -0.13621299
   0.02353767 -0.07820367 -0.1903776  -0.02801538 -0.04915288 -0.03114728
   0.01550583  0.10379014  0.16400056  0.13852447  0.08499211 -0.09173524
  -0.10806326 -0.09664021 -0.22016157 -0.09735918 -0.2611183   0.01721956
  -0.15482788 -0.10806365 -0.10105728  0.05470413  0.13449574
-0.04372285]]
epoch:40, loss=0.3296
Epoch 50 - Before updating: Weights: [[-0.23513842 -0.20449613 -0.20120125
-0.09220279 -0.18870737 -0.1442326
   0.01294266 -0.0900318  -0.1941498  -0.02632674 -0.0580579  -0.02992082
   0.00689481  0.09488824  0.16387632  0.13513213  0.0820194  -0.09728307
  -0.10692288 -0.09642725 -0.23250403 -0.10444853 -0.27346686  0.00554434
  -0.16216666 -0.11685827 -0.11149379  0.04206502  0.12751916
-0.04857429]]
Epoch 50 - After updating: Weights: [[-0.23631428 -0.20514576 -0.20239131
-0.09335358 -0.18914555 -0.14505069
   0.01184583 -0.09126305 -0.19453202 -0.02612659 -0.05898652 -0.02979052
   0.00599892  0.0939592   0.16386178  0.13480477  0.08173524 -0.0978408
  -0.1067948  -0.09637449 -0.23379843 -0.10520193 -0.2747602   0.00431978
  -0.16293687 -0.11776582 -0.11257578  0.04074823  0.12678736
-0.04906899]]
epoch:50, loss=0.3035
Epoch 60 - Before updating: Weights: [[-0.24634446 -0.21075211 -0.21253346
-0.10317601 -0.19278787 -0.15186132
   0.00258506 -0.101712   -0.19769438 -0.02423737 -0.06688683 -0.02866665
  -0.00160703  0.08604818  0.16372128  0.13223036  0.0795249  -0.10241697
  -0.1056373  -0.09568485 -0.24485853 -0.11171757 -0.2857979  -0.00614618
  -0.16952358 -0.1254026  -0.12172064  0.02956699  0.1205348  -0.0531846
]]
Epoch 60 - After updating: Weights: [[-0.24740265 -0.21135052 -0.2136025
-0.10421301 -0.19316229 -0.15256223
   0.00161799 -0.10280879 -0.19801779 -0.02401901 -0.0677185  -0.02854691
  -0.00240606  0.08521457  0.16370448  0.1319817   0.07931424 -0.10288067
  -0.10550851 -0.09558687 -0.24602738 -0.11241443 -0.286963   -0.0072526
  -0.17022039 -0.1261972  -0.12267646  0.02839279  0.1198741
-0.05360783]]
epoch:60, loss=0.2830
Epoch 70 - Before updating: Weights: [[-0.25648636 -0.2165438  -0.22277178
-0.11312157 -0.1963013  -0.15843983
```

```
  -0.00660583 -0.11218073 -0.20071423 -0.02199503 -0.07484785 -0.0275113
  -0.00924263  0.0780616   0.16354136  0.13002916  0.07768508 -0.10671018
  -0.10435173 -0.0945456  -0.2560774  -0.11847305 -0.29696962 -0.01676971
  -0.17621946 -0.13292965 -0.13081     0.01835623  0.11419427
-0.05715435]]
Epoch 70 - After updating: Weights: [[-0.25745037 -0.217101   -0.22374408
-0.11406774 -0.19662665 -0.15904893
  -0.00747047 -0.11317084 -0.20099205 -0.02176453 -0.07560366 -0.02740066
  -0.00996603  0.07730252  0.16352186  0.12984113  0.07753111 -0.10710069
  -0.10422363 -0.09441397 -0.25714573 -0.11912422 -0.29803216 -0.01778184
  -0.17685813 -0.13363478 -0.13166563  0.01729564  0.11359061
-0.05752164]]
epoch:70, loss=0.2663
Epoch 80 - Before updating: Weights: [[-0.26576626 -0.2219576  -0.23212524
-0.12223656 -0.19937345 -0.16418639
  -0.01486555 -0.12167708 -0.20332295 -0.01965166 -0.08212034 -0.02644142
  -0.01619235  0.0707513   0.16333318  0.12837262  0.07635409 -0.11034291
  -0.10307679 -0.09310968 -0.2663761  -0.12480877 -0.3072034  -0.0265312
  -0.18238625 -0.13964255 -0.13898712  0.00818213  0.10837533
-0.06061825]]
Epoch 80 - After updating: Weights: [[-0.26665285 -0.22248077 -0.23301813
-0.12310822 -0.19966011 -0.16472177
  -0.01564732 -0.12258034 -0.20356457 -0.0194133  -0.08281495 -0.02633869
  -0.01685484  0.07005233  0.16331074  0.1282323   0.07624459 -0.11067526
  -0.10295013 -0.09295277 -0.2673618  -0.12542208 -0.3081818  -0.02746602
  -0.18297778 -0.14027515 -0.13976143  0.00721422  0.10781843
-0.06094086]]
epoch:80, loss=0.2524
Epoch 90 - Before updating: Weights: [[-0.27433088 -0.22705656 -0.24074553
-0.1306634  -0.20209451 -0.16925925
  -0.02236534 -0.13037467 -0.20560251 -0.01724432 -0.08883145 -0.02544614
  -0.02258367  0.06399258  0.16309524  0.12714766  0.07542478 -0.11344698
  -0.10181822 -0.09145021 -0.2759115  -0.13079362 -0.31666014 -0.03557906
  -0.1881206  -0.14568987 -0.14641738 -0.00113887  0.10298782
-0.06367521]]
Epoch 90 - After updating: Weights: [[-0.2751525  -0.22755107 -0.24157187
-0.13147259 -0.20235002 -0.1697343
  -0.02307876 -0.13120581 -0.20581485 -0.01700117 -0.08947551 -0.02535037
  -0.02319592  0.0633433   0.16306978  0.12704545  0.07535065 -0.11373235
  -0.1016934  -0.09127419 -0.27682787 -0.13137493 -0.317568   -0.03644913
  -0.18867321 -0.14626256 -0.14712442 -0.00202969  0.10247002
-0.06396156]]
epoch:90, loss=0.2406
Epoch 100 - Before updating: Weights: [[-0.28229064 -0.23188816
-0.24874668 -0.13850884 -0.20453097 -0.17377664
  -0.02923374 -0.13840383 -0.20761387 -0.01479989 -0.09507484 -0.02451668
  -0.02851036  0.05769395  0.1628271   0.12626974  0.07481632 -0.11612135
```

```
  -0.10057913 -0.08962179 -0.28480145 -0.1364797  -0.3254607  -0.04402452
  -0.19349517 -0.15118356 -0.15322582 -0.0097453   0.0979634
-0.06639975]]
Epoch 100 - After updating: Weights: [[-0.2830568  -0.2323581  -0.24951631
-0.13926475 -0.20476101 -0.17420152
  -0.02988987 -0.13917406 -0.20780215 -0.01455422 -0.09567636 -0.02442706
  -0.02908037  0.05708657  0.16279863  0.12619841  0.07477058 -0.11636825
  -0.10045636 -0.08943135 -0.28565866 -0.13703352 -0.32630846 -0.04483942
  -0.1940151  -0.15170598 -0.1538764  -0.01057099  0.09747878
-0.06665625]]
epoch:100, loss=0.2305

accuracy=0.9123
```