face_classification_starter

May 2, 2022

[212]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy.io import loadmat
from sklearn.metrics import accuracy_score
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import train_test_split
dataset = loadmat('face_emotion_data.mat')

X, y = dataset['X'], dataset['y']
n, p = np.shape(X)

y[y==-1] = 0  # use 0/1 for labels instead of -1/+1
X = np.hstack((np.ones((n,1)), X))  # append a column of ones
```

0.0.1 1)

a)

[202]:
```
#Logistic function => f(z) = 1/(1+e^(-z))
```

[205]:
```
## Train NN
Xb = np.hstack((np.ones((n,1)), X))
q = np.shape(y)[1] #number of classification problems
M = 32 #number of hidden nodes

## initial weights
V = np.random.randn(M, q);
W = np.random.randn(p+2, M);

def logsig(_x):
    return 1/(1+np.exp(-_x))

H = logsig(Xb@W)
Yhat = logsig(H@V)
```

[194]:
```
Yhat = [1 if i > 0.5 else 0 for i in Yhat ]
Yhat
```

[194]: [1,
 1,
 1,
 1,
 1,
 0,
 1,
 0,
 1,
 0,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 0,
 1,
 1,
 1,
 1,
 1,
 0,
 0,
 1,
 1,
 1,
 0,
 1,
 0,
 1,
 1,
 1,
 1,
 1,
 1,
 0,
 1,
 1,
 0,
 1,
 1,
 1,
 1,
 1,
 0,
 1,

1,
1,
1,
1,
0,
1,
1,
0,
1,
1,
1,
0,
1,
0,
1,
0,
1,
1,
0,
0,
0,
1,
1,
1,
1,
0,
1,
0,
1,
1,
1,
1,
1,
1,
0,
1,
1,
1,
0,
1,
0,
1,
1,
0,
1,
0,
1,

```
 0,
 1,
 0,
 0,
 1,
 0,
 1,
 1,
 1,
 1,
 1,
 1,
 0,
 1,
 1,
 0,
 1,
 0,
 1,
 0,
 0,
 0,
 1,
 1,
 0,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1]
```

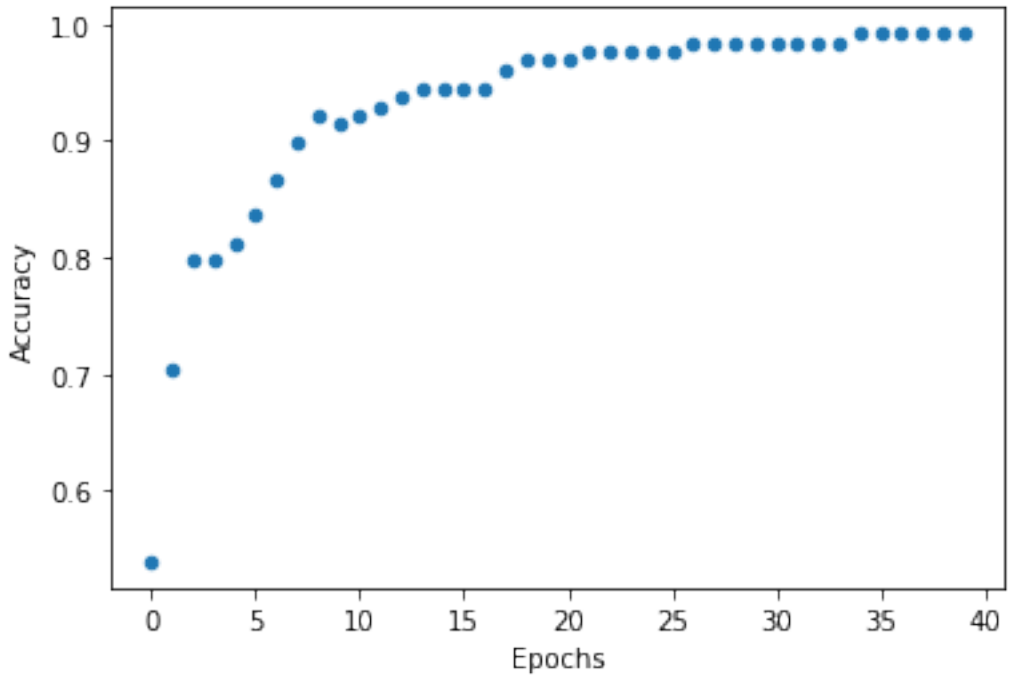
b)

[170]:
```
L = 40
## initial weights
V = np.random.randn(M+1, q);
W = np.random.randn(p+2, M);

alpha = 0.05
accuracy = []
epochs = []
for epoch in range(L):
    ind = np.random.permutation(n)
    epochs.append(epoch)
```

```
    for i in ind:
        # Forward-propagate
        H = logsig(np.hstack((np.ones((1,1)), Xb[[i],:]@W)))
        Yhat = logsig(H@V)
         # Backpropagate
        delta = (Yhat-y[[i],:])*Yhat*(1-Yhat)
        Vnew = V-alpha*H.T@delta
        gamma = delta@V[1:,:].T*H[:,1:]*(1-H[:,1:])
        Wnew = W - alpha*Xb[[i],:].T@gamma
        V = Vnew
        W = Wnew
    H = logsig(np.hstack((np.ones((n,1)), Xb@W)))
    Yhat = logsig(H@V)
    Yhat = [1 if i > 0.5 else 0 for i in Yhat ]
    accuracy.append(accuracy_score( Yhat, y ))
df = pd.DataFrame({"Epochs":epochs,"Accuracy":accuracy})
df.plot.scatter("Epochs","Accuracy")
```

[170]: <AxesSubplot:xlabel='Epochs', ylabel='Accuracy'>



It takes about 35 epochs to reach 0% of error in the training set.

c)

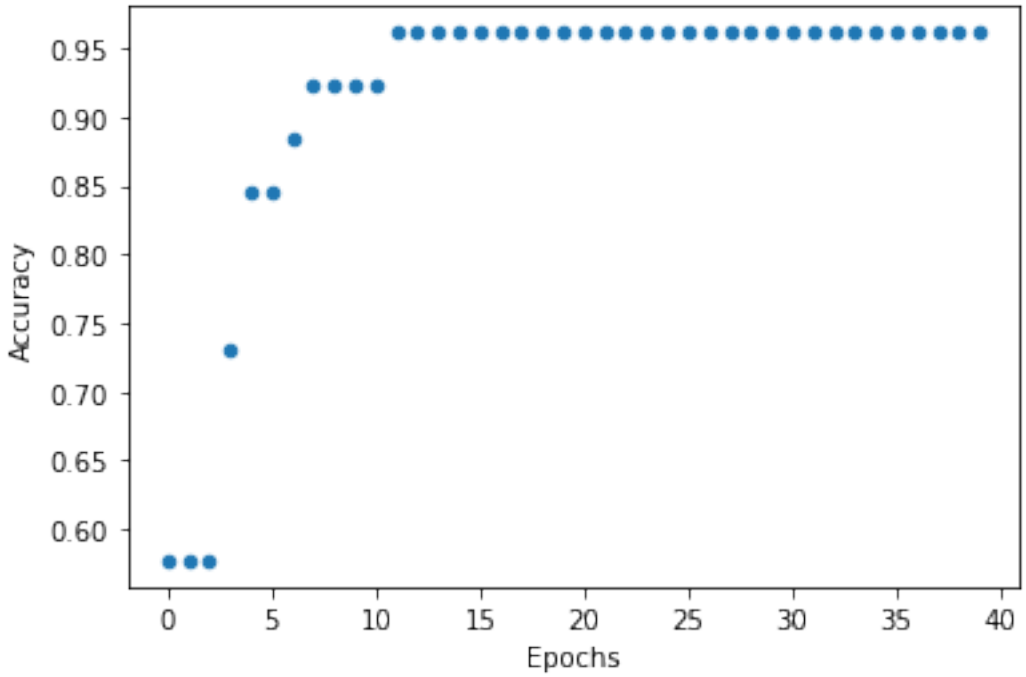
[216]:
```
X_train, X_test, Y_train, Y_test = train_test_split(Xb, y, test_size = 0.2)
q = np.shape(Y_train)[1]
M = 32
V = np.random.randn(M+1, q);
W = np.random.randn(p+2, M);

L = 40

alpha = 0.05
accuracy = []
epochs = []
for epoch in range(L):
    ind = np.random.permutation(len(X_train))
    epochs.append(epoch)
    for i in ind:
        # Forward-propagate
        H = logsig(np.hstack((np.ones((1,1)), X_train[[i],:]@W)))
        Yhat = logsig(H@V)
         # Backpropagate
        delta = (Yhat-Y_train[[i],:])*Yhat*(1-Yhat)
        Vnew = V-alpha*H.T@delta
        gamma = delta@V[1:,:].T*H[:,1:]*(1-H[:,1:])
        Wnew = W - alpha*X_train[[i],:].T@gamma
        V = Vnew
        W = Wnew
    H = logsig(np.hstack((np.ones((len(X_test),1)), X_test@W)))
    Yhat = logsig(H@V)
    Yhat = [1 if i > 0.5 else 0 for i in Yhat ]
    accuracy.append(accuracy_score( Yhat, Y_test ))
```

[217]:
```
df = pd.DataFrame({"Epochs":epochs,"Accuracy":accuracy})
df.plot.scatter("Epochs","Accuracy")
```

[217]:
```
<AxesSubplot:xlabel='Epochs', ylabel='Accuracy'>
```



Now as wee can see on the graph, we reach 0% of testing error by approximately 12 epochs