

11주차 과제

2021254008 최준혁

1. 프로그램 4-4를 수행하여 결과를 정리하고, 프로그램의 동작을 설명하시오.

```
1 from sklearn.datasets import fetch_openml
2 from sklearn.neural_network import MLPClassifier
3 import matplotlib.pyplot as plt
4 import numpy as np
5
6 # MNIST 데이터셋을 읽고 훈련 집합과 테스트 집합으로 분할
7 mnist=fetch_openml('mnist_784')
8 #0~1 정규화
9 mnist.data=mnist.data/255.0
10 # train : 6만개, test : 1만개
11 x_train=mnist.data[:60000]; x_test=mnist.data[60000:]
12 # 정수형 변환
13 y_train=np.int16(mnist.target[:60000]); y_test=np.int16(mnist.target[60000:])
14
15 # MLP 분류기 모델을 학습
16 #MLP 객체 생성, 미니배치 512
17 mlp=MLPClassifier(hidden_layer_sizes=(100), learning_rate_init=0.001, batch_size=512, max_iter=300, solver='adam', verbose=True)
18 #학습
19 mlp.fit(x_train,y_train)
20
21 # 테스트 집합으로 예측
22 res=mlp.predict(x_test)
23
24 # 혼동 행렬
25 conf=np.zeros((10,10), dtype=np.int16)
26 for i in range(len(res)):
27     conf[res[i]][y_test[i]]+=1
28 print(conf)
29
30 # 정확률 계산
31 no_correct=0
32 for i in range(10):
33     no_correct+=conf[i][i]
34 accuracy=no_correct/len(res)
35 print("테스트 집합에 대한 정확률은", accuracy*100, "%입니다.")
```

- 4-4 프로그램 코드

프로그램 4-4는 데이터를 읽은 후 각 데이터를 0~1 사이로 정규화하고, 데이터 앞쪽 ~ 6만 개는 훈련 집합과 나머지는 테스트 집합으로 분할 후 혼동행렬에서 사용하기 위해 정수형으로 변환합니다. 훈련 집합의 크기가 크기 때문에 미니배치를 512로 설정하고, 훈련 집합으로 학습을 하고 테스트 집합으로 예측을 진행해 결과를 출력합니다.

Iteration 1, loss = 0.62422579	Iteration 43, loss = 0.00823320	<pre> Training loss at test: [[962 0 5 0 2 1 6 1 3 0] [0 1121 0 0 0 0 2 3 0 2] [1 4 999 3 5 0 3 7 0 0] [2 1 1 977 0 9 1 1 3 3] [2 0 3 1 953 2 3 0 3 6] [2 0 0 6 0 856 3 0 1 3] [3 1 1 0 5 6 936 0 2 1] [3 1 7 5 1 0 0 1007 2 9] [3 7 15 7 3 13 4 4 958 5] [2 0 1 11 13 5 0 5 2 980]] 테스트 집합에 대한 정확률은 97.49 %입니다. </pre>
Iteration 2, loss = 0.26320898	Iteration 44, loss = 0.00773622	
Iteration 3, loss = 0.20797085	Iteration 45, loss = 0.00728825	
Iteration 4, loss = 0.17508509	Iteration 46, loss = 0.00662892	
Iteration 5, loss = 0.15115085	Iteration 47, loss = 0.00640001	
Iteration 6, loss = 0.13345659	Iteration 48, loss = 0.00608273	
Iteration 7, loss = 0.11849987	Iteration 49, loss = 0.00556292	
Iteration 8, loss = 0.10585587	Iteration 50, loss = 0.00513541	
Iteration 9, loss = 0.09551129	Iteration 51, loss = 0.00474864	
Iteration 10, loss = 0.08719583	Iteration 52, loss = 0.00440422	
Iteration 11, loss = 0.07993754	Iteration 53, loss = 0.00428661	
Iteration 12, loss = 0.07441903	Iteration 54, loss = 0.00398580	
Iteration 13, loss = 0.06817040	Iteration 55, loss = 0.00364066	
Iteration 14, loss = 0.06260953	Iteration 56, loss = 0.00354734	
Iteration 15, loss = 0.05764234	Iteration 57, loss = 0.00317351	
Iteration 16, loss = 0.05388849	Iteration 58, loss = 0.00306641	
Iteration 17, loss = 0.05040489	Iteration 59, loss = 0.00282706	
Iteration 18, loss = 0.04638953	Iteration 60, loss = 0.00267598	
Iteration 19, loss = 0.04282182	Iteration 61, loss = 0.00242628	
Iteration 20, loss = 0.04029984	Iteration 62, loss = 0.00223967	
Iteration 21, loss = 0.03781362	Iteration 63, loss = 0.00221184	
Iteration 22, loss = 0.03486418	Iteration 64, loss = 0.00216038	
Iteration 23, loss = 0.03324714	Iteration 65, loss = 0.00194813	
Iteration 24, loss = 0.03058012	Iteration 66, loss = 0.00188386	
Iteration 25, loss = 0.02833491	Iteration 67, loss = 0.00173555	
Iteration 26, loss = 0.02678874	Iteration 68, loss = 0.00166969	
Iteration 27, loss = 0.02490817	Iteration 69, loss = 0.00157920	
Iteration 28, loss = 0.02284012	Iteration 70, loss = 0.00152115	
Iteration 29, loss = 0.02168660	Iteration 71, loss = 0.00140974	
Iteration 30, loss = 0.01983893	Iteration 72, loss = 0.00130444	
Iteration 31, loss = 0.02039843	Iteration 73, loss = 0.00127198	
Iteration 32, loss = 0.01712677	Iteration 74, loss = 0.00119601	
Iteration 33, loss = 0.01614813	Iteration 75, loss = 0.00115672	
Iteration 34, loss = 0.01510573	Iteration 76, loss = 0.00113160	
Iteration 35, loss = 0.01450581	Iteration 77, loss = 0.00105342	
Iteration 36, loss = 0.01349074	Iteration 78, loss = 0.00104501	
Iteration 37, loss = 0.01250711	Iteration 79, loss = 0.00097295	
Iteration 38, loss = 0.01184983	Iteration 80, loss = 0.00090489	
Iteration 39, loss = 0.01078066	Iteration 81, loss = 0.00085651	
Iteration 40, loss = 0.01000241	Iteration 82, loss = 0.00110553	
Iteration 41, loss = 0.00918703	Iteration 83, loss = 0.01075155	
Iteration 42, loss = 0.00883512		

- 프로그램 4-4 수행결과

프로그램 4-4 는 학습 횟수가 늘어날수록 오차율이 감소하지만, 일정횟수를 넘어가면서 비슷한 오차율을 보이고, 소요시간은 2분 17초, 정확률은 97.49%가 출력되었습니다.

2. batch size를 128로 하고, 은닉층 사이즈를 50인 경우에 수행하여 결과를 비교하시오.

batch size: 512, 은닉층 : 100	batch size: 128, 은닉층 : 50
<pre> Training loss at test: [[962 0 5 0 2 1 6 1 3 0] [0 1121 0 0 0 0 2 3 0 2] [1 4 999 3 5 0 3 7 0 0] [2 1 1 977 0 9 1 1 3 3] [2 0 3 1 953 2 3 0 3 6] [2 0 0 6 0 856 3 0 1 3] [3 1 1 0 5 6 936 0 2 1] [3 1 7 5 1 0 0 1007 2 9] [3 7 15 7 3 13 4 4 958 5] [2 0 1 11 13 5 0 5 2 980]] 테스트 집합에 대한 정확률은 97.49 %입니다. </pre>	<pre> [[963 0 4 0 3 1 3 1 1 0] [0 1121 4 1 1 0 2 5 1 2] [5 4 1004 6 4 0 1 14 2 0] [1 0 2 981 1 11 1 1 6 5] [0 2 3 1 939 2 4 3 6 9] [3 1 0 7 0 862 5 0 2 3] [4 1 3 0 11 4 937 0 0 0] [0 2 3 4 5 2 0 997 3 3] [3 4 8 7 1 8 5 1 947 5] [1 0 1 3 17 2 0 6 6 982]] 테스트 집합에 대한 정확률은 97.33000000000001 %입니다. </pre>
소요시간 : 2분 17초	소요시간 : 2분 41초

batch 와 은닉층 사이즈를 변경하고 수행하였을 때, 이전보다 정확률이 소폭 낮아진 것을 확인할 수 있습니다. batch size가 512에서 128로 줄었기 때문에 소요시간이 늘어난 것을 확인할 수 있습니다.