전복의 고리수 추정 신경망

강민진

단층 퍼셉트론

hyperparameter 변경으로 weight, bias를 변경시켜 학습을 다르게 하면서

최적의 accuracy를 찾아줘야 한다.

여기서 변경시킬 수 있는 hyperparameter는

1. minibatch size (미니배치 크기)

2. learning rate (학습률)

3. epoch (학습 횟수)

가 있다.

위의 3가지 딥러닝 모델의 구조나 학습 과정에 영향을 미치는 값을 변경해줌으로써 마음에 드는 학습 결과를 얻을 때까지 값을 바꾸어가면서 실험을 진행하였다.

abalone dataset에 들어가는 input값은 성별, 키, 지름, 높이, 전체 무게, 몸통 무게, 내장 무게, 껍질 무게가 있다. 이 8개의 input을 이용해서 신경망을 만들면 된다.

다른 값들과는 달리 성별 값은 I(Infant), M(Male), F(Female)로 표시되는데, 이 값을 숫자로 바꿔서 표현해도 선형성을 찾기 힘들기 때문에 one-hot vector를 사용해서, 해당 항목만 1, 나머지는 0으로 나타낸다. 단층 퍼셉트론에서 비선형성을 넣으면 예측 성능이 떨어지고, 원하는대로 동작하지 않을 수 있다.

이 방식을 이용해서 나타내면 input이 10개가 된다. 따라서 weight는 [10, 1] 형태의 행렬이 되며, bias는 [1, 1]의 vector가 되어야 한다.

Test 결과

	Test1	Test2	Test3	Test4	**Test5**	Test6	Test7	Test8	Test9	Test10
minibatch size	10	10	10	100	**500**	200	200	500	1000	500
learning rate	0.001	0.01	0.1	0.1	**0.1**	0.15	0.15	0.15	0.15	0.19
epoch_count	10	10	10	100	**500**	500	10000	500	500	500
Final accuracy	0.810	0.821	0.836	0.8418	**0.8439**	0.8426	0.8394	0.8454	0.8419	0.8469

	Test11	Test12	Test13	**Test14**	**Test15**	Test16	Test17	**Test18**	**Test19**	**Test20**
minibatch size	**500**	500	500	**500**	**500**	500	500	**500**	**500**	**500**
learning rate	**0.2**	0.21	0.22	**0.25**	**0.28**	0.3	0.33	**0.35**	**0.4**	**0.001**
epoch_count	**500**	500	500	**500**	**500**	500	500	**500**	**1000**	**500**
Final accuracy	**0.8472**	0.8476	0.8479	**0.8489**	**0.8495**	0.8495	0.8485	**NaN**	**NaN**	**0.8129**

의미 있는 Test

report = 20으로 조정하여 20 단위로 출력하도록 하였으며, 좀 더 정확한 accuracy를 찾기 위해서 accuracy를 소수 3자리까지 보이도록 한 것을 조정하여 소수 4자리까지 보이도록 하였다.

Test 5 (minibatch size = 500, learning rate 0.1, epoch_count = 500) accuracy : 0.8439

```
Epoch 320: loss=4.9679, accuracy=0.8382/0.8420
Epoch 340: loss=4.9445, accuracy=0.8378/0.8414
Epoch 360: loss=4.9287, accuracy=0.8383/0.8419
Epoch 380: loss=4.9130, accuracy=0.8383/0.8427
Epoch 400: loss=4.9055, accuracy=0.8399/0.8427
Epoch 420: loss=4.8827, accuracy=0.8392/0.8425
Epoch 440: loss=4.8778, accuracy=0.8401/0.8432
Epoch 460: loss=4.8634, accuracy=0.8405/0.8406
Epoch 480: loss=4.8644, accuracy=0.8403/0.8421
Epoch 500: loss=4.8568, accuracy=0.8390/0.8439

Final Test: final accuracy = 0.8439
```

Test 11 (minibatch size = 500, learning rate 0.2, epoch_count = 500) accuracy : 0.8472

```
Epoch 320: loss=4.8414, accuracy=0.8406/0.8425
Epoch 340: loss=4.8233, accuracy=0.8397/0.8436
Epoch 360: loss=4.8340, accuracy=0.8397/0.8411
Epoch 380: loss=4.8219, accuracy=0.8394/0.8464
Epoch 400: loss=4.8415, accuracy=0.8409/0.8451
Epoch 420: loss=4.8026, accuracy=0.8404/0.8430
Epoch 440: loss=4.8164, accuracy=0.8408/0.8442
Epoch 460: loss=4.7967, accuracy=0.8411/0.8412
Epoch 480: loss=4.8197, accuracy=0.8411/0.8402
Epoch 500: loss=4.8150, accuracy=0.8390/0.8472

Final Test: final accuracy = 0.8472
```

Test 14 (minibatch size = 500, learning rate 0.25, epoch_count = 500) accuracy : 0.8489

```
Epoch 320: loss=4.8414, accuracy=0.8406/0.8405
Epoch 340: loss=4.8120, accuracy=0.8398/0.8450
Epoch 360: loss=4.8355, accuracy=0.8395/0.8395
Epoch 380: loss=4.8135, accuracy=0.8393/0.8479
Epoch 400: loss=4.8647, accuracy=0.8409/0.8442
Epoch 420: loss=4.7914, accuracy=0.8407/0.8425
Epoch 440: loss=4.8343, accuracy=0.8403/0.8448
Epoch 460: loss=4.7864, accuracy=0.8412/0.8412
Epoch 480: loss=4.8202, accuracy=0.8413/0.8373
Epoch 500: loss=4.8177, accuracy=0.8384/0.8489

Final Test: final accuracy = 0.8489
```

Test 15 (minibatch size = 500, learning rate 0.28, epoch_count = 500) accuracy : 0.8495

```
Epoch 320: loss=4.8430, accuracy=0.8405/0.8388
Epoch 340: loss=4.8078, accuracy=0.8399/0.8461
Epoch 360: loss=4.8490, accuracy=0.8391/0.8394
Epoch 380: loss=4.8095, accuracy=0.8393/0.8483
Epoch 400: loss=4.9028, accuracy=0.8403/0.8432
Epoch 420: loss=4.7863, accuracy=0.8408/0.8421
Epoch 440: loss=4.8924, accuracy=0.8389/0.8463
Epoch 460: loss=4.7819, accuracy=0.8412/0.8410
Epoch 480: loss=4.8300, accuracy=0.8413/0.8358
Epoch 500: loss=4.8269, accuracy=0.8378/0.8495

Final Test: final accuracy = 0.8495
```

Test 18 (minibatch size = 500, learning rate 0.1, epoch_count = 500) accuracy : NaN

```
Epoch 20: loss=2221.0004, accuracy=-3.7923/-3.6707
Epoch 40: loss=32781.9534, accuracy=-17.4216/-18.3162
Epoch 60: loss=755865.1777, accuracy=-87.5030/-86.0996
Epoch 80: loss=12875300.4656, accuracy=-363.4510/-393.9569
Epoch 100: loss=268160061.6336, accuracy=-1661.7308/-1710.8603
Epoch 120: loss=5274886969.4115, accuracy=-7384.2404/-7758.7938
Epoch 140: loss=92502928743.1032, accuracy=-30905.6082/-32150.0643
Epoch 160: loss=1858002080411.6250, accuracy=-138536.6934/-142269.8004
Epoch 180: loss=36685173762389.0078, accuracy=-615893.8655/-666879.1246
Epoch 200: loss=744714693872121.6250, accuracy=-2770887.8826/-2970427.0106
Epoch 220: loss=15158838566050122.0000, accuracy=-12525832.6723/-12515262.595
0
```

Learning rate가 너무 높아 epoch 횟수가 증가함에도 불구하고 loss값이 커지는 것을 확인할 수 있다.

Test 19도 마찬가지의 결과가 나왔다. 이 abalone 데이터셋에서는 Learning rate가 0.35이상이면 이렇게 epoch가 증가함에도 loss값이 커지는 것을 확인할 수 있었다.

Test 20

```
Epoch 320: loss=7.2192, accuracy=0.8094/0.8125
Epoch 340: loss=7.1977, accuracy=0.8094/0.8125
Epoch 360: loss=7.1773, accuracy=0.8094/0.8126
Epoch 380: loss=7.1579, accuracy=0.8094/0.8126
Epoch 400: loss=7.1393, accuracy=0.8094/0.8127
Epoch 420: loss=7.1216, accuracy=0.8094/0.8127
Epoch 440: loss=7.1046, accuracy=0.8094/0.8128
Epoch 460: loss=7.0884, accuracy=0.8095/0.8128
Epoch 480: loss=7.0728, accuracy=0.8094/0.8129
Epoch 500: loss=7.0579, accuracy=0.8094/0.8129

Final Test: final accuracy = 0.8129
```

Learning rate가 너무 낮아 위의 Test 경우들에 비교해서 같은 Epoch 횟수에 비해 Loss 값이 큰 것을 확인할 수 있다

결론

학습 횟수에 걸친 loss 값을 분석하여 적합한 learning rate 를 찾아내는 과정이 필요하다. 실습을 통하여 minibatch size가 증가하면 정확성이 떨어지는 경우도 있으나, 학습 속도가 증가하는 것을 확인할 수 있었다. epoch 횟수를 증가하면 학습 시간은 오래 걸리지만 minibatch size와 learning rate에 따라 어떤 값으로 수렴하는지를 확인하기 수월했다.

Test 15의 Final Test에서 0.8495라는 accuracy를 찾아내서 얼핏 보기에는 제일 정확한 값을 찾은 듯 보이나, learning rate가 높아서 정확한 수렴값을 찾지 못하고 크게 주변을 맴돌기 때문에 Epoch 460, Epoch 480, Epoch 500이 비교적 큰 폭으로 차이가 난다. 마지막 결과값만 Test 15를 좋은 Learning rate를 설정한 최적의 구조라고 말하기 어렵다. 그렇다고 Test 5 를 최적의 결과값으로 말하기에는 정확성이 비교적 많이 떨어진다. 그러므로 Test 5와 Test 15를 어느정도 절충한 Test인 Test 11이 평균적인 정확성이나 수렴값을 보았을 때 가장 정확하다고 판단하였다.