

Hyper Parameter

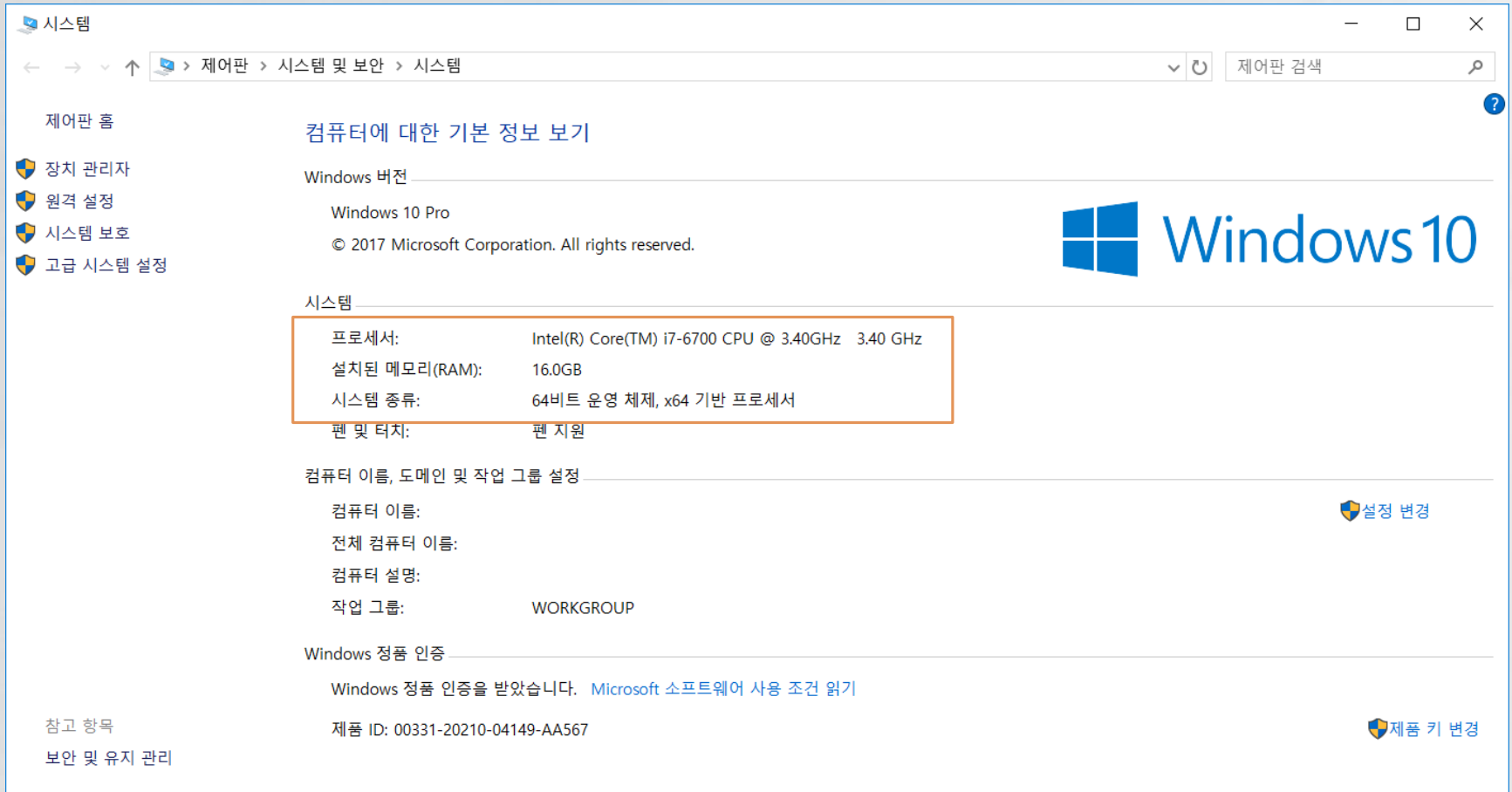
(Learning rate, Generation, Batch size)

작성자 : 김진성

목차

1. Test 환경
2. L1 vs L2 cost function
3. Learning rate
4. Generation(epoch)
5. Batch size

1. Test 환경



The screenshot shows the Windows 10 'System' settings window. The title bar reads '시스템' (System). The breadcrumb navigation shows '제어판 > 시스템 및 보안 > 시스템'. The left sidebar contains links to '장치 관리자', '원격 설정', '시스템 보호', and '고급 시스템 설정'. The main content area is titled '컴퓨터에 대한 기본 정보 보기'. It displays 'Windows 버전' as 'Windows 10 Pro' with copyright information. The '시스템' section is highlighted with an orange box and contains the following details:

프로세서:	Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz	3.40 GHz
설치된 메모리(RAM):	16.0GB	
시스템 종류:	64비트 운영 체제, x64 기반 프로세서	
팬 및 터치:	팬 지원	

Below this, the '컴퓨터 이름, 도메인 및 작업 그룹 설정' section shows fields for '컴퓨터 이름:', '전체 컴퓨터 이름:', '컴퓨터 설명:', and '작업 그룹:' (set to 'WORKGROUP'). A '설정 변경' link is present. The 'Windows 정품 인증' section shows a message about genuine Windows and a link to 'Microsoft 소프트웨어 사용 조건 읽기', along with the '제품 ID: 00331-20210-04149-AA567' and a '제품 키 변경' link.

참고 항목
보안 및 유지 관리

2. L1 vs L2 cost(loss) function

model 비용/손실 함수 : MAE, MSE

- L1 cost function

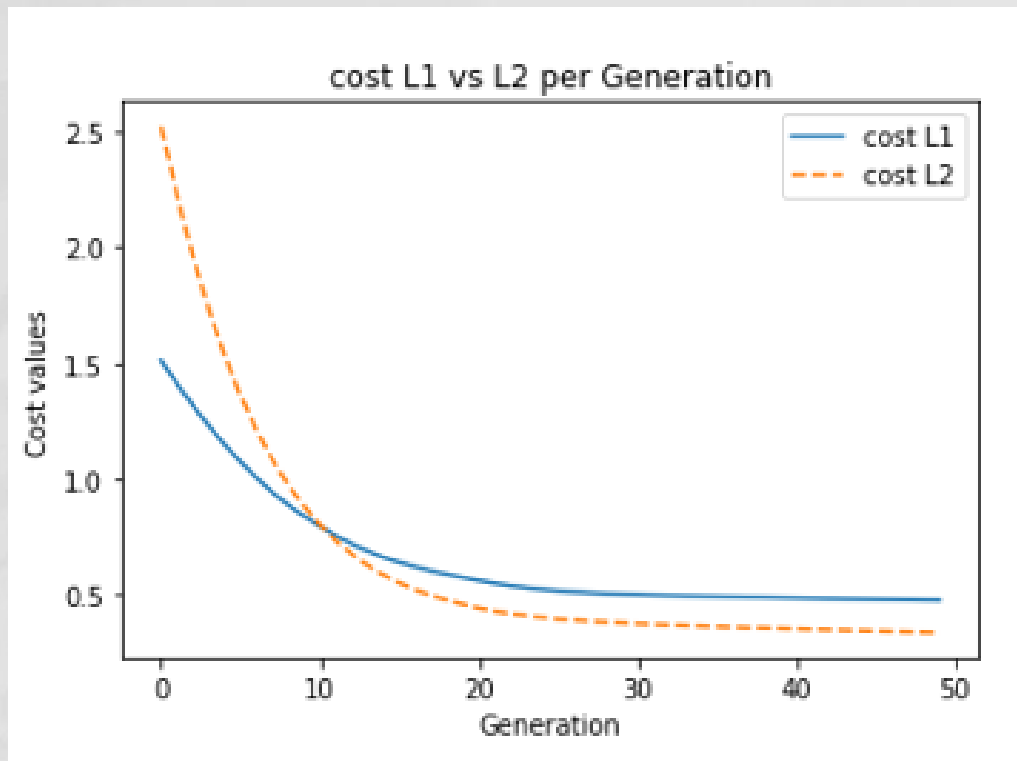
- ✓ 라쏘(Lasso) 회귀
- ✓ MAE(mean absolute error)
- ✓ $\text{cost_L1} = \text{tf.reduce_mean}(\text{tf.abs}(Y - \text{model_output}))$

- L2 cost function

- ✓ 릿지(Lidge) 회귀
- ✓ MSE(mean square error)
- ✓ $\text{cost_L2} = \text{tf.reduce_mean}(\text{tf.square}(Y - \text{model_output}))$

3. Learning rate

- Learning rate 적정한 경우



cost values

L1 = [0.48023006, 0.4791867, 0.4781492, 0.4771173, 0.476091]

L2 = [0.3443453, 0.3427307, 0.34113768, 0.33956522, 0.33801302]

Dataset : iris

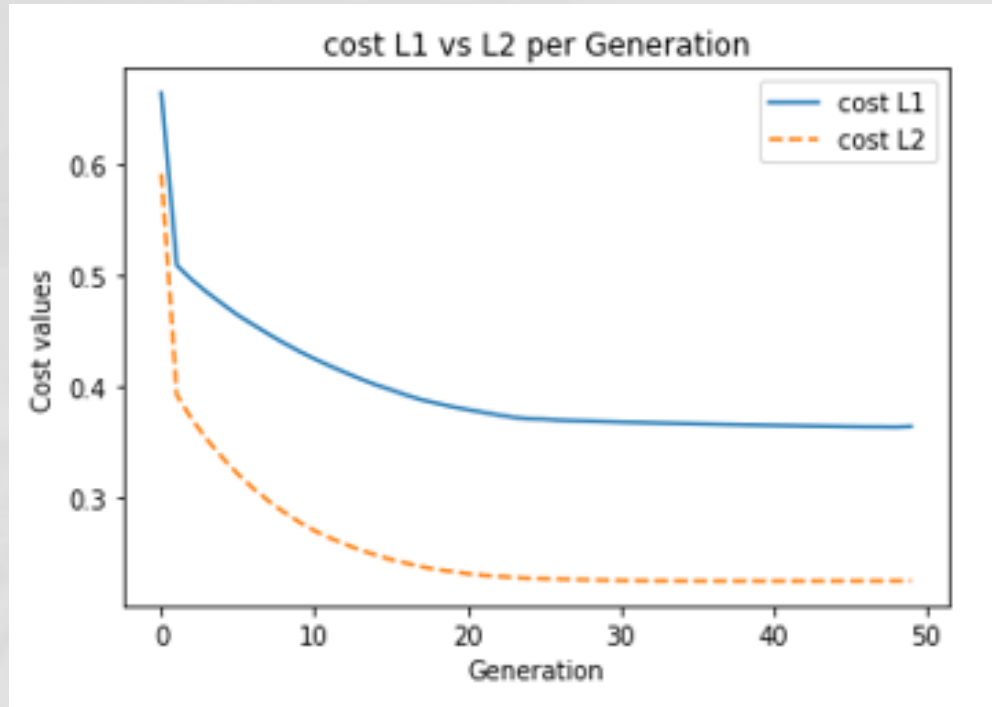
learning_rate = 0.01

iter_size = 50

학습률을 적정하게
지정한 경우 반복 학
습이 증가하면서 cost
가 점진적으로 0에
수렴한다.

L1 방식에 비해서 L2
방식이 0에 더 가깝
게 수렴한다.

● Learning rate 큰 경우(비교적)



Dataset : iris

learning_rate = 0.1

iter_size = 50

학습률을 적절한 크기 보다 높이면 0에 수렴하는 속도가 빨라진다.

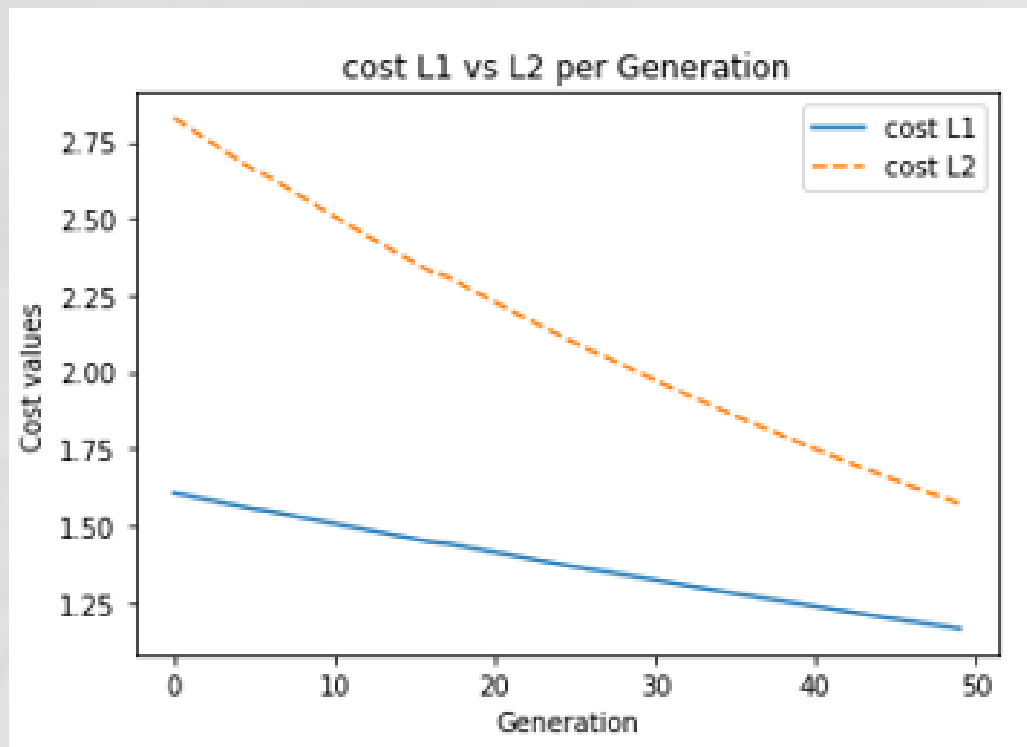
적은 반복학습으로 cost의 최솟점(0)을 찾을 수 있지만, 정확도는 낮다.

cost values

L1 = [0.36515275, 0.36493295, 0.36472228, 0.3645204, 0.36432663]

L2 = [**0.2257019**, 0.22572665, 0.22575803, 0.22579534, 0.22583748]

● Learning rate 작은 경우(비교적)



cost values

L1 = [1.1923388, 1.1844356, 1.1765758, 1.1687598, 1.1611265]

L2 = [1.6404339, 1.6210787, 1.6019807, 1.583138, 1.564547]

Dataset : iris

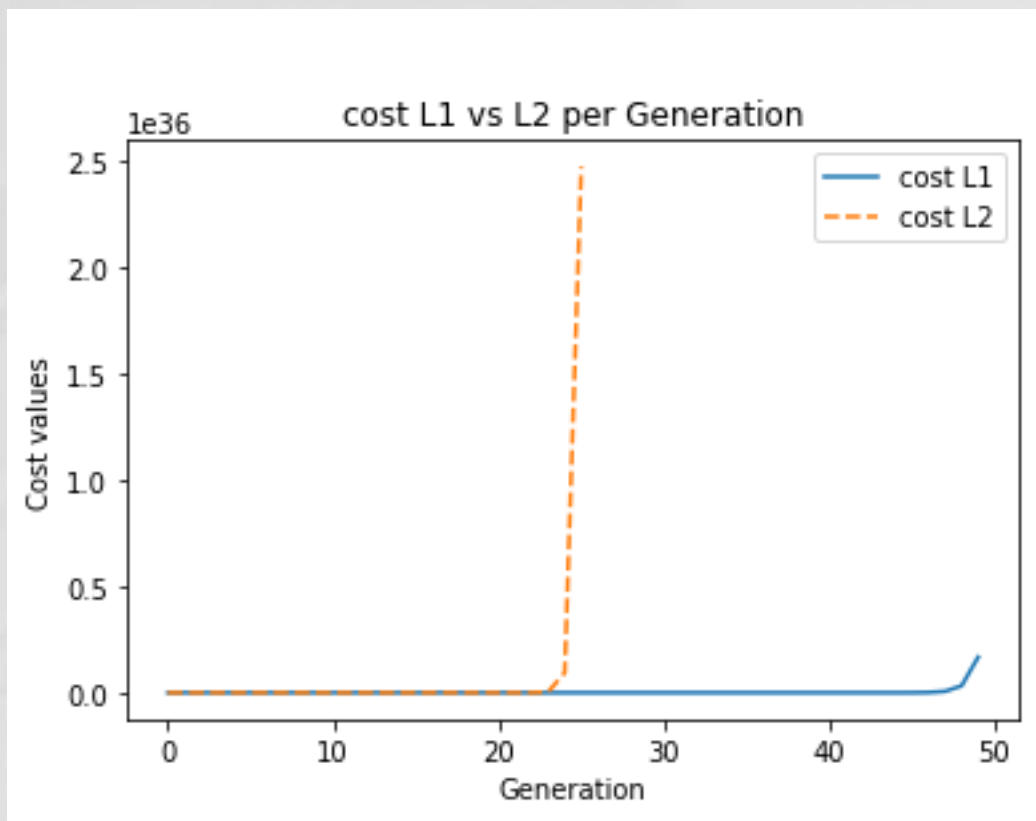
learning_rate = 0.001

iter_size = 50

학습률을 적정한 크기 보다 작게 하면 0에 수렴하는 속도가 느려진다.

학습률을 지나치게 낮추면 0에 수렴하는 속도가 느려지기 때문에 반복학습을 늘려야 한다.

● Learning rate 적정하지 않은 경우



learning_rate = 0.9
iter_size = 50

학습률을 지나치게
높이면 0에 수렴하는
속도가 너무 빨라지
고, cost값이 비정상
적으로 만들어진다.

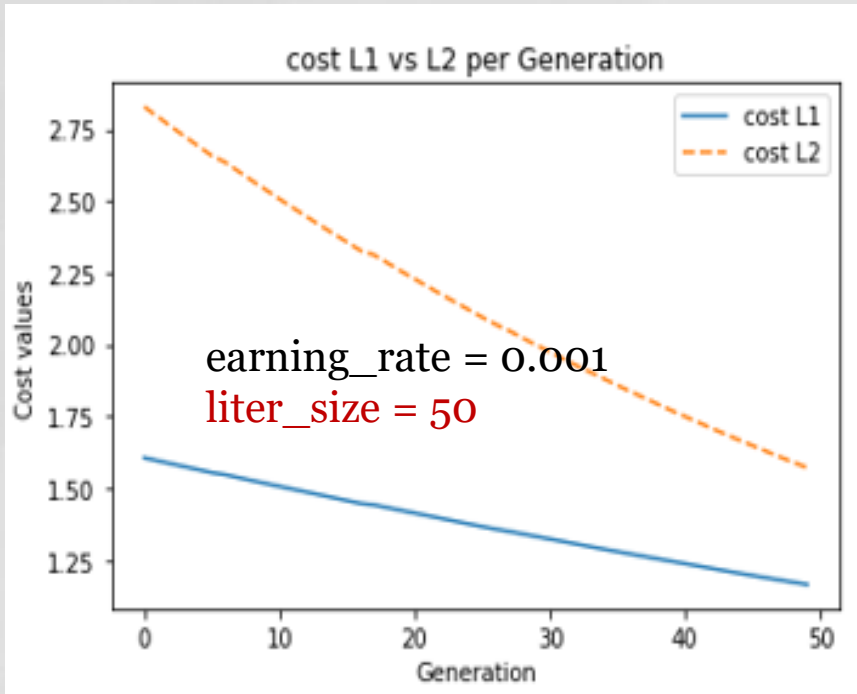
cost values

L1 = [6.37634e+30, 2.8822617e+31, 1.3028528e+32, 5.889213e+32, 2.662068e+33]

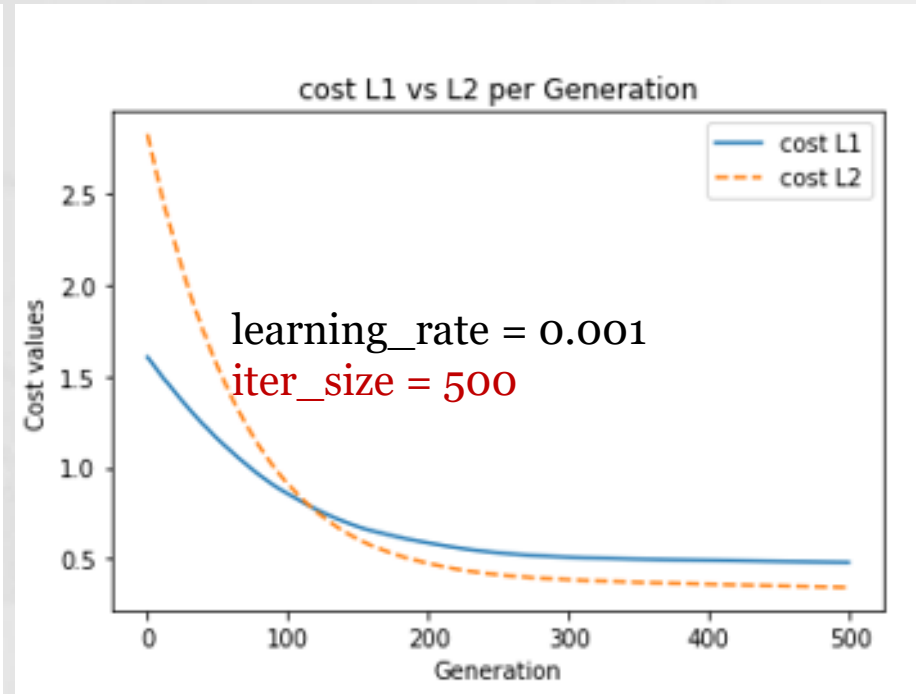
L2 = [inf, inf, inf, inf, inf]

4. Generation(epoch)

- 학습률이 낮은 경우 cost의 최솟점을 찾기 위해서 반복학습 회수를 늘린다.
- 학습횟수가 지나치게 많아져도 cost에는 큰 변화 없음



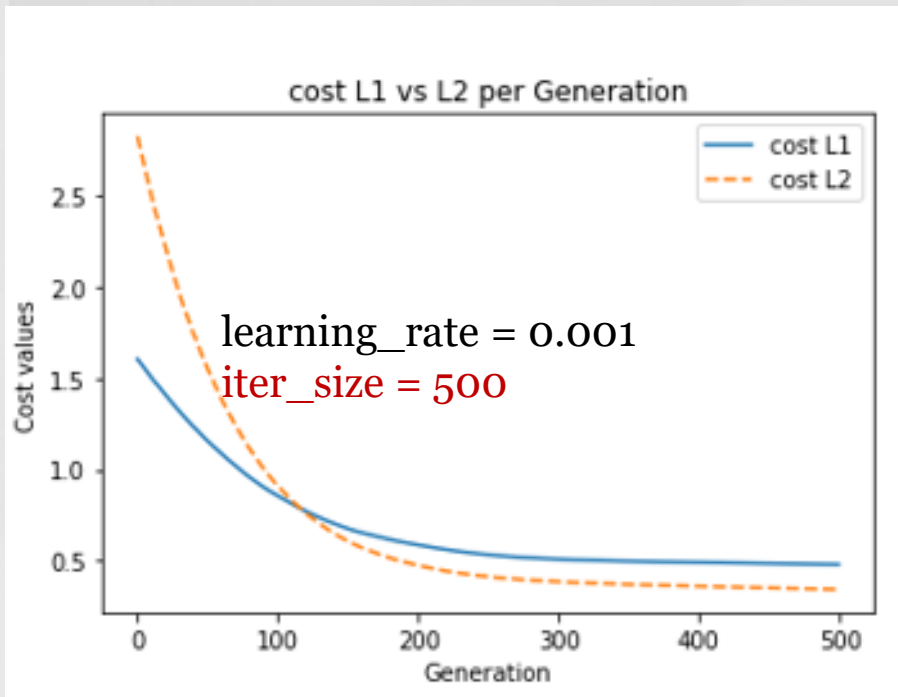
L1 = [1.1923388, 1.1844356, 1.1765758,
1.1687598, 1.1611265]
L2 = [1.6404339, 1.6210787, 1.6019807,
1.583138, 1.564547]



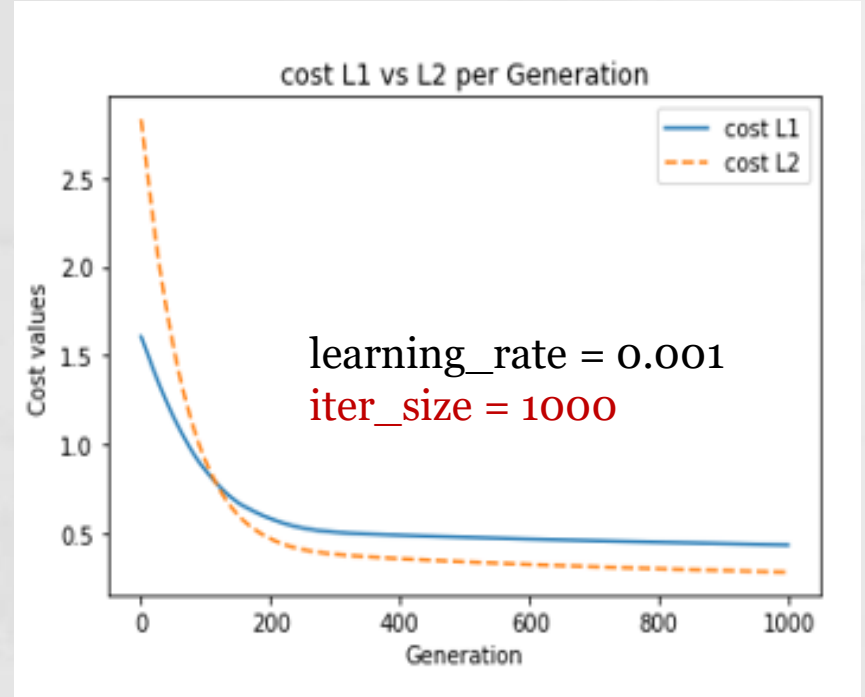
L1=[0.47846994, 0.4783665,
0.47826314, 0.4781599, 0.47805664]
L2=[0.34162125, 0.34146267, 0.3413043,
0.34114614, 0.3409882]

● 학습횟수가 지나치게 많은 경우

✓ cost에는 큰 변화 없음, 과적합(overfitting) 발생



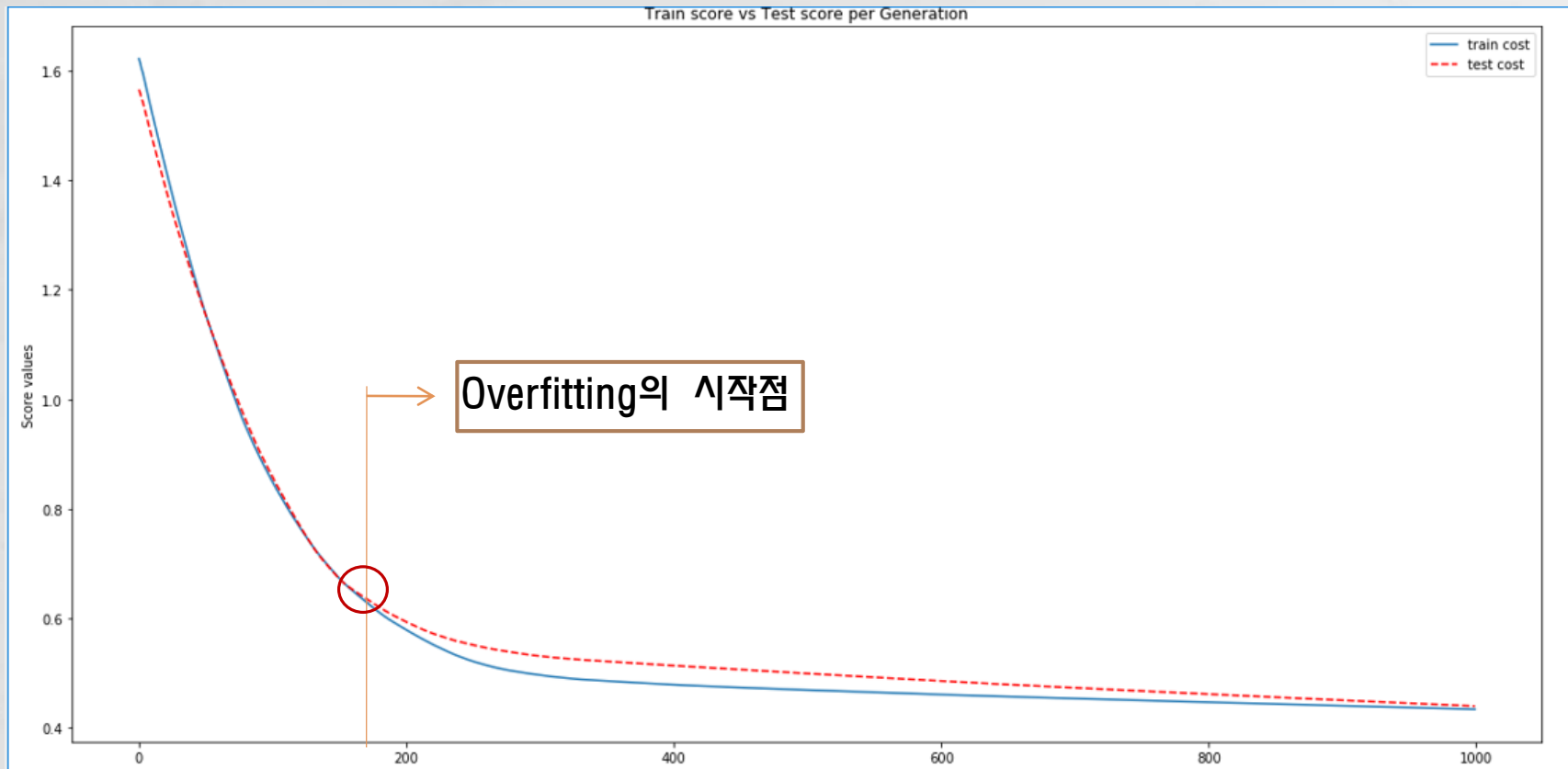
L1=[0.47846994, 0.4783665,
0.47826314, 0.4781599, 0.47805664]
L2=[0.34162125, 0.34146267, 0.3413043,
0.34114614, 0.3409882]



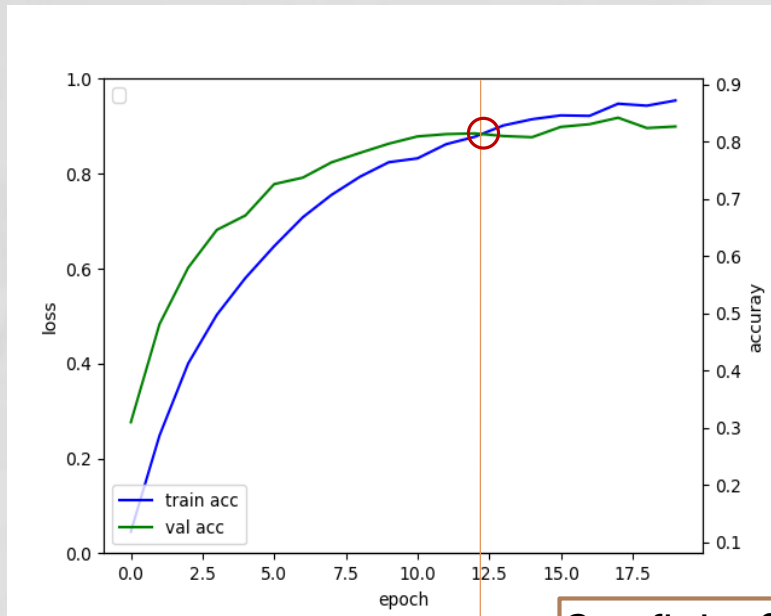
L1=[0.43492034, 0.4348461, 0.4347719,
0.43469766, 0.43462357]
L2=[0.28194258, 0.28185663,
0.2817708, 0.28168502, 0.28159946]

● 과적합(overfitting)을 고려한 반복횟수 결정

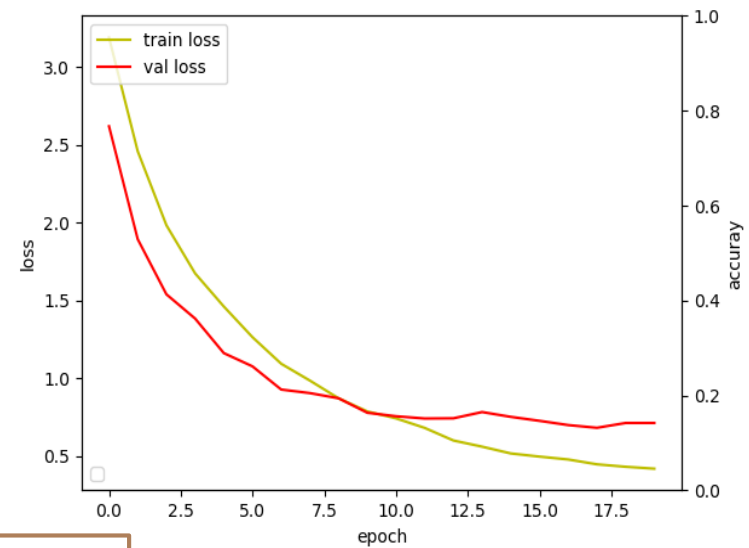
✓ Train 예측치와 Test 예측치의 교차점



● 분류정확도(Accuracy) 예



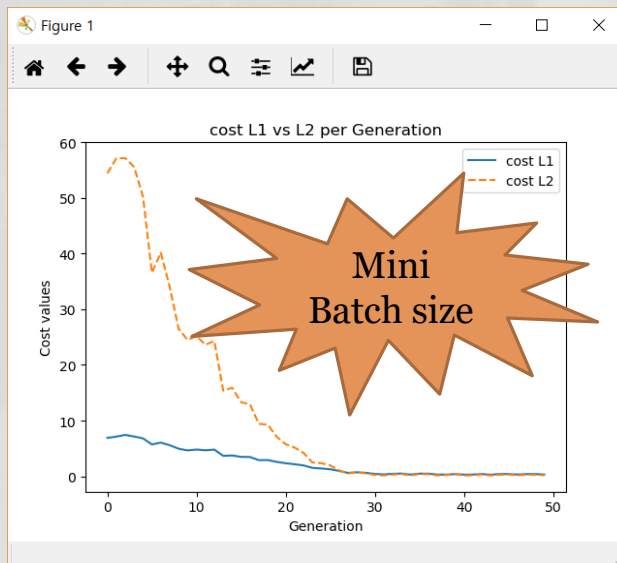
Overfitting의 시작점



5. Batch size(Mini batch vs Full batch)

- 전체 데이터 셋을 한 번에 학습(full batch)한 후 기울기와 절편을 수정하는 것 보다, Batch size를 지정하여 Model을 학습하는 것이 더 좋다.

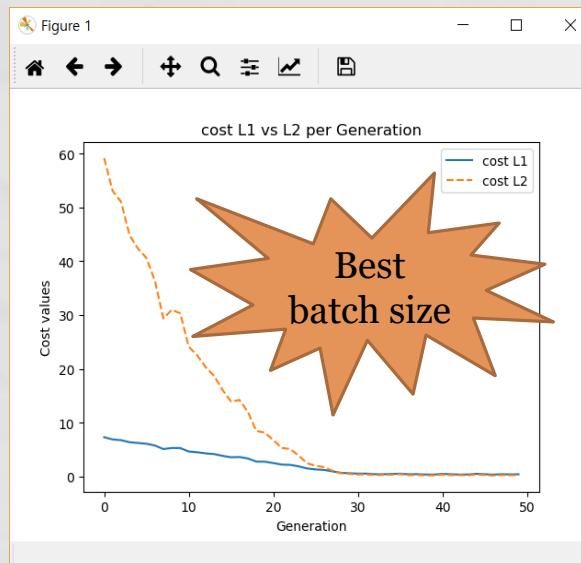
batch_size = 25



cost values

[0.52573496, 0.44184673,
0.43498725, 0.4324601,
0.42966926]
[0.414955, 0.32474276,
0.31981355, 0.353978,
0.30057228]

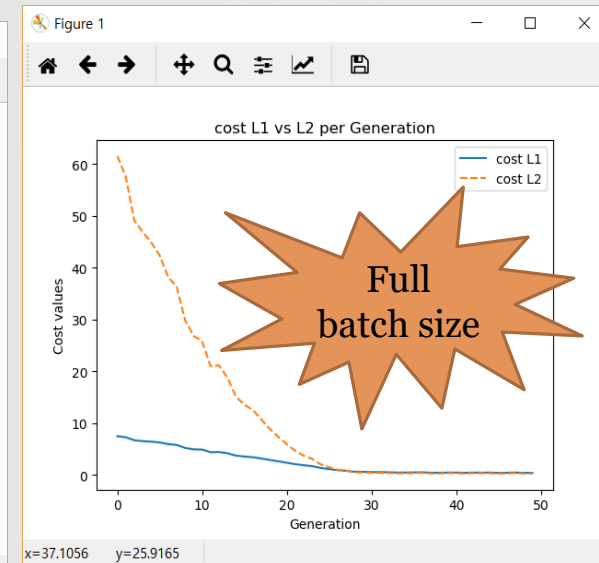
batch_size = 50



cost values

[0.4643185, 0.352369,
0.33250573, 0.4167205,
0.3862848]
[0.30322018, 0.20713636,
0.17424262, 0.26978245,
0.21953487]

batch_size = 150



cost values

[0.51659864, 0.5049254,
0.4648637, 0.46590352,
0.46129212]
[0.40130436, 0.40553337,
0.31910434, 0.3517762,
0.3480786]

● California Housing 데이터 셋 예

```
X, y = fetch_california_housing(return_X_y=True)
print(X.shape) # (20640, 8)

x_train, x_test, y_train, y_test = train_test_split(
    x_data, y_data, random_state=123) # test_size = 0.25
print(x_train.shape) # (15480, 8)
```

Full batch (batch_size = 15480)



*** model 평가(test set : full batch) ***

MSE = 0.32020

MAE = 0.45852

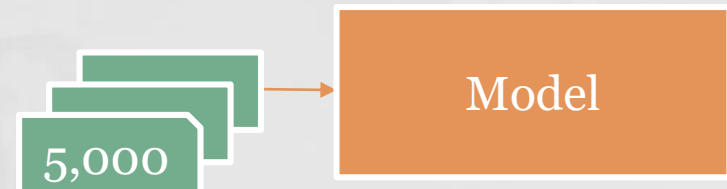
예측값 평균 = 0.5545497

실제값 평균 = 0.56379884

=====

실행시간 : 141.72010612487793

Mini batch (batch_size = 5,000)



*** model 평가(test set : mini batch) ***

MSE = 0.32057

MAE = 0.45835

예측값 평균 = 0.5744557

실제값 평균 = 0.56379884

=====

실행시간 : 42.69244194030762