

▼ Perceptrons - Training

Note for

드라이브에서 찾기

실습 모드에서 열기

• M

새 Python 3 노트

새 Python 2 노트

def prec

bias

acti

if a

else

노트 열기... Ctrl+O

노트 업로드...

이름 바꾸기...

• E

휴지통으로 이동

Gradient descent

드라이브에 사본 저장...

def trai

#wei

weig

prin

prin

GitHub Gist로 사본 저장...

dataset이 아니다

없음

GitHub에 사본 저장...

저장 Ctrl+S

vb =

vw0

vw1

버전 저장 및 고정 Ctrl+M S

for

업데이트 기록

.ipynb 다운로드

.py 다운로드

드라이브 미리보기 업데이트

인쇄 Ctrl+P

error * error * row[i] #에러가 0이면 값 유지 #ro

vw0.append(weights[1])

vw1.append(weights[2])

print('epoch={}, error={}'.format(epoch, sum_error))

return weights,vb,vw0,vw1

#training set for AND gate

dataset = [[0,0,0],

[0,1,0],

[1,0,0],

[1,1,1]]

• Hyperparameters

l_rate = 0.1

n_epoch = 5

weights, vb,vw0,vw1 = train_weights(dataset, l_rate, n_epoch)



```
print(weights)#에러 빼기=미분
```



```
pred = predict([0.51,1], weights)
print(pred)
```



```
pred = predict([1,0.0000000000000001], weights)
print(pred)
```



```
import matplotlib.pyplot as plt
```

```
plt.plot(pred)
```



```
plt.plot(vb)
```



```
plt.plot(vw0)
```



```
plt.plot(vw1)
```



```
plt.plot(prediction)
```



- Why ?

partial derivative with respect to m

#derivative=미분

$$\begin{aligned}\frac{\partial J(m, b)}{\partial m} &= \frac{1}{n} \sum_{i=1}^n -2x^{(i)}(y_i - (mx^{(i)} + b)) \\ &= \frac{2}{n} \sum_{i=1}^n x^{(i)}((mx^{(i)} + b) - y^{(i)}) \\ &= \frac{2}{n} \sum_{i=1}^n x^{(i)}(\hat{y}^{(i)} - y^{(i)})\end{aligned}$$

partial derivative with respect to b

$$\begin{aligned}\frac{\partial J(m, b)}{\partial b} &= \frac{1}{n} \sum_{i=1}^n -2(y^{(i)} - (mx^{(i)} + b)) \\ &= \frac{-2}{n} \sum_{i=1}^n (y^{(i)} - (mx^{(i)} + b)) \\ &= \frac{2}{n} \sum_{i=1}^n (\hat{y}^{(i)} - y^{(i)})\end{aligned}$$

m은 x가 붙어있음 바이오스 = 맨아래부분 차이점 x붙어있음

Partial derivatives : <https://www.mathsisfun.com/calculus/derivatives-partial.html>

- References

<https://machinelearningmastery.com/implement-perceptron-algorithm-scratch-python/>

