Artificial Intelligence: Project 2 보고서

201614868 이성민

(모든 코드 실행결과(html) 파일에서 전체 코드 및 실행 결과 확인가능합니다.)

1. **Numpy: dot, einsum**

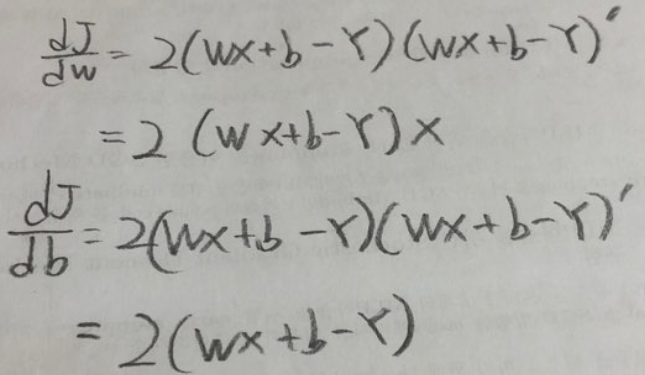
dot - a, b가 2차원 이하에서는 내적, 행렬곱으로 연산이 됨을 알 수 있고,

a가 고차원 배열, b가 2차원 이상의 배열일 때, a의 마지막 축과 b의 뒤에서 두번째 축과의 내적으로 계산됨을 알 수 있다.

einsum – 직관적으로 행렬의 계산이 가능하다, 그리고 dot과 einsum을 이용한 행렬곱의차이는 dot은 계산시 직관적으로 shape을 얻지 못한다는 단점이 있는 것 같다. Pytorch, tensorflow 등 여러 라이브러리에서 einsum을 지원하는 걸로 보아 einsum을 사용하는 것이 좋은 것 같다.

1. **Linear regression**

Linear regression을 위한 식 유도



Linear regression학습을 위한 SGD pseudo code

While True:

Batch = next\_train\_batch(train\_data, 256)

W = W – Ir \* (2WX+b-Y)X)

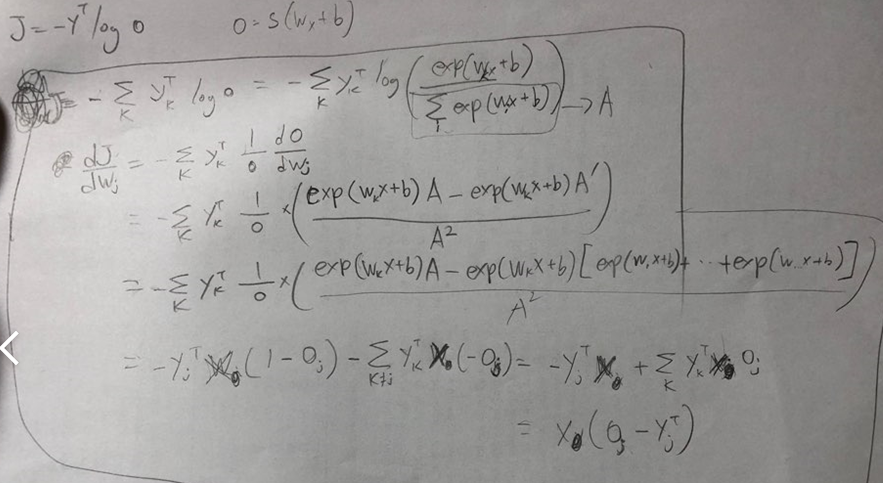
b = b – Ir \* (2(WX+b-Y))

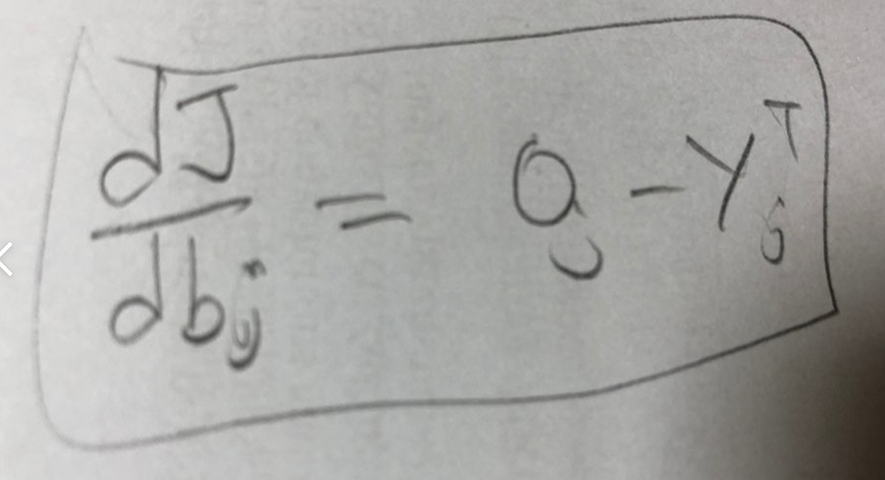
if Early\_Stopping(Acc): # acc 변화 에 따라 Early Stopping 적용

break

1. **Logistic regression**

Logistic regression을 위한 식 유도





Logistic regression학습을 위한 SGD pseudo code

While True:

Batch = next\_train\_batch(train\_data)

W = W – Ir \* (X\*(softmax(WX+b)-Y))

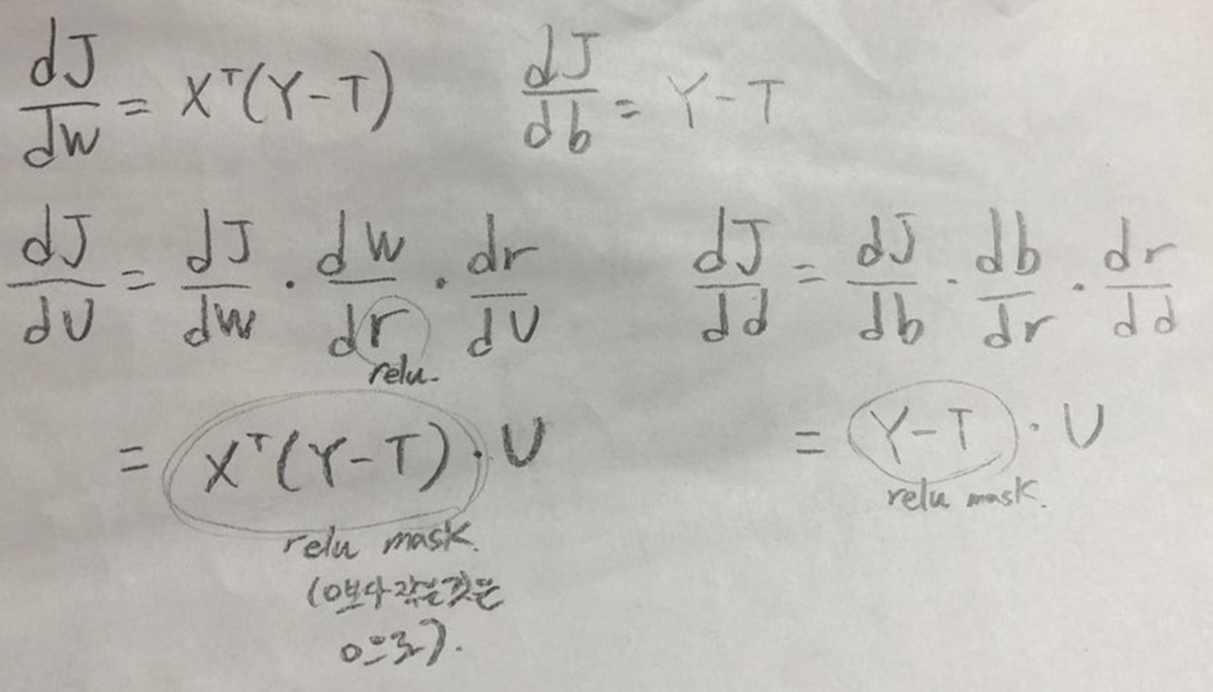
b = b – Ir \* (softmax(WX+b)-Y)

if Early\_Stopping(Acc): # acc 변화에 따라 Early Stopping 적용

break

1. **Multi-layer perceptron**

MLP를 위한 식 유도



MLP학습을 위한 SGD pseudo code

While True:

Batch = next\_train\_batch(train\_data) --배치뽑기

grad = MLP.backpropagation(train\_data)

--역전파를 이용해 미분값 구하기(반복문을 이용해 구현)

for key in Parameter: --파라미터 업데이트

network.params[key] -= learning\_rate \* grad[key]

if Early\_Stopping(Acc): # accuracy변화에 따라 Early Stopping 적용

break

은닉층의 개수 별 accuracy변화

은닉층의 개수를 3개로 늘렸을 때 acc가 가장 높게 나왔다.(95%) 학습시간은 더 오래걸렸다. 은닉층의 개수를 4개로 늘려보았는데, vanishing gradient가 일어나는듯한지 학습이 제대로 이루어지지 않았다. Vanishing gradient문제를 해결할 방법이 필요해 보인다.