[머신러닝 스터디 2~4강 정리 9/6]

# 2강\_ Image Classification

이미지 -> 3D array(height \* width \* color channel)

문제점 : 보는 시각에 따라, 조명에 따라, 감춰진 정도에 따라 등등 여러 요인에 따라 다르게 보일 수 있다.

* Nearest Neighbor Classifier (Nonparametric approach)

: 모든 학습용 이미지와 레이블을 메모리상에서 기억, 테스트 이미지를 학습 이미지와 하나하나 비교해보며 가장 비슷한 학습 이미지 label을 test label이라 predict

Test 이미지와 학습 이미지 비교 -> 비교기준 = 거리(hyper parameter), 거리 같으면 같은 이미지

-L1 distance =

-L2 distance =

* K Nearest Neighbor Classifier(kNN) -> test 시간 대비 성능이 안 좋아 안 쓴다.

: k개의 가장 가까운 이미지 찾고 k개의 이미지가 다수결로 vote, 가장 많이 나오는 것으로 해당 test 이미지가 어떤 학습 이미지와 유사한지 판단

Hyper parameter를 계속 바꾸며 Test 데이터 셋에 적용 -> 절대 하면 안된다.

Test data set은 성능 평가용, 학습할 때 절대 사용하면 안됨

hyper parameter 설정을 위해 여러 번의 실험을 해야함 -> Train data의 일부 (20%정도)를 validation set data로 설정하여 해당 hyper parameter에 대해 어느 정도의 성능이 나오는지 판단 후 튜닝

Train data가 적은 경우에는 Cross validation 사용

: Train data set을 n개의 fold로 나누어 n-1개의 fold는 학습 진행, 1개의 fold는 검증. 검증에 사용되지 않은 fold 1개는 검증에 사용 나머지 fold와 그 전 검증에서 사용한 fold는 학습, 이 과정을 반복한다.

* Linear Classifier

: 이미지 내 픽셀 값들에 대해 가중치를 곱해 처리, parametric approach

f(x,W) = Wx + b (W=가중치, b= bias, x=input)

# 3강\_ Loss function

SVM(Hinge Loss)

Weight Regularization

L1 Regularization

L2 Regularization

Regularization -> Train에 대한 정확도는 안 좋아지지만 Test에 대한 성능은 좋아진다

L2 Regularization의 경우 spread out된 weight를 더 선호한다 -> 모든 input feature를 고려하기위해

Softmax classifier

Score 정의 : class를 log화할 확률 & unnormalized

목적: 제대로 된 class의 log확률 최대화 -> 정확한 class의 -log확률 최소화

Optimization : loss를 최소화하는 weight 찾는 과정

전략 – 랜덤 search를 피한다, slope를 따라간다(gradient)

analytic기반으로 코딩 후 제대로 진행되고 있는지 numerical로 확인한다.

스터디 중 나왔던 질문 2

Q : 가중치의 변화율(gradient)를 구할 때 왜 Loss값 차이를 step size(learning rate)로 나누는 지, 가중치를 update할 때 왜 -step size \* gradient를 더해주는지

A : W(t+1) = W(t) - -> 델타 룰에 의해 step size는 update하는 크기, 미분된 값은 방향을 결정, gradient descent에서 cost function(loss) 최저 값으로 가기 위해서는 빼줘야함

#4강 Backpropagation

Gradient = global gradient \* local gradient

Backpropagation을 하는 이유 : Forward Pass만 사용할 경우 input의 인자가 사라질 수 있는데 Backpropagation을 하면 chain rule에 의해 input의 인자가 남아있게 되어 output에 대한 input의 영향을 알 수 있다.

Local gradient : Forward pass시 구하고 메모리에 저장. 현재연산결과 / input

Global gradient : Backpropagation 시 구한다. 최종Output / input

Add gate : gradient distributor : 각 input에 output값 그대로 전달

Max gate : gradient router : 큰 값의 local gradient = 1, 작은 값의 local gradient = 0

Mul gate : gradient switcher : xy를 x에 대해 미분하면 y -> input의 local gradient 값을 서로 바꾸어 넣어준다