알고리즘 ⇒ 어떤 문제를 푸는 완전히 절차.

↳ Input에 대해 always 정답제공

Machine Learning ⇒ Data-driven approach로 access.

- 1) Model 학습 절차
- 2) unseen data 예측 절차.

Nearest Neighbor

가장 가까운 pixel들을 사용해서 predict 해보자. 즉, 주변 이웃의 pixel들을 보고 가장 많은 pixel로 해당 pixel의 class를 결정하는 아이디어.

1) 학습 ⇒ test case를 받기 이전에 모든 data를 memory에 올려놓고 기억하게 전부이다.

2) 예측 ⇒ 모든 pixel에 대해 비교하여 class 예측.

이미지를 끼리 비교하는 기준이 뭘까?

∴ 두 이미지의 유사도를 수치화하는 function이 필요하다.

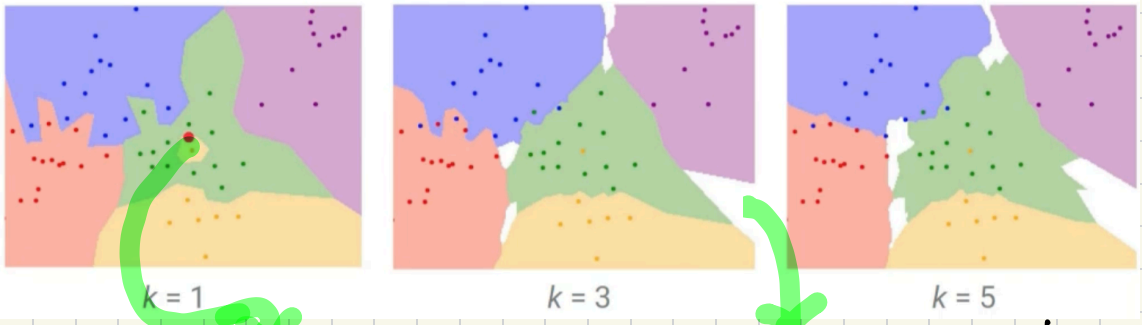
$$sim(img1, img2) = ?$$

① 모든 이미지 pixel 끼리의 차이 절대값 $L_1(A,B) = \sum_{i=1}^{n}\sum_{j=1}^{m} |A_{ij} - B_{ij}|$

(img 사이즈 동일 가정) OR 제곱사용

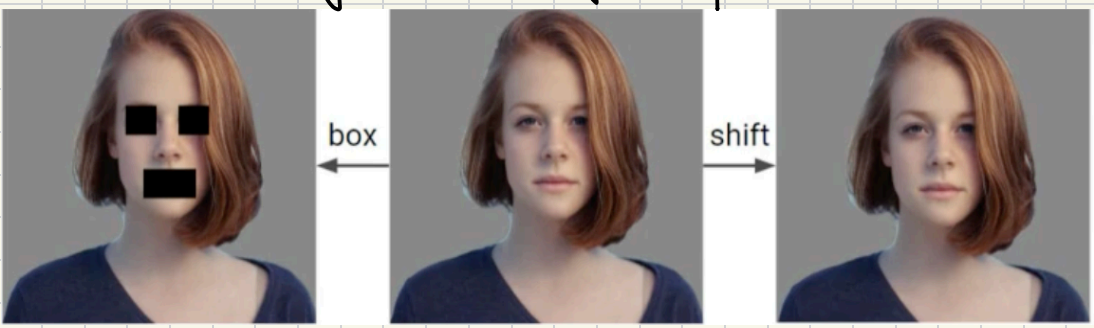
→ 학습시 $O(1)$ & 예측시 $O(n)$ ⇒ 효율적 최악. predict 소요시간 ↓ 해야해 효율적이기에.

→ 가장 L이 작은 이웃 사용 ⇒ k=1인 case.



k가 작으면 Noise에 약함, k↑ 연산량 많아짐. k는 어떻게 정함?
⇒ Cross-validation.

Nearest Neighbor는 Image에 적용 불가능한 이유.



사람은 shift가 유사했지만 model는 완전히 유사하다고 판단. ⇒ pixel 값들만 비교하기에

즉, 사람이 이해하는 의미적 해석이 잘 되지 않는다.

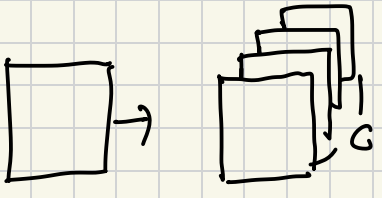
그리고 이미지가 (224 x 224 x 3) 인 경우 pixel 비교는 computation 많이↑

Parametric Approach

이미지를 받고 [이미지 → f(x) → 결과] 인 함수를 저장하자.

f(x)를 Linear하게 해보자 ⇒ Linear classifier.

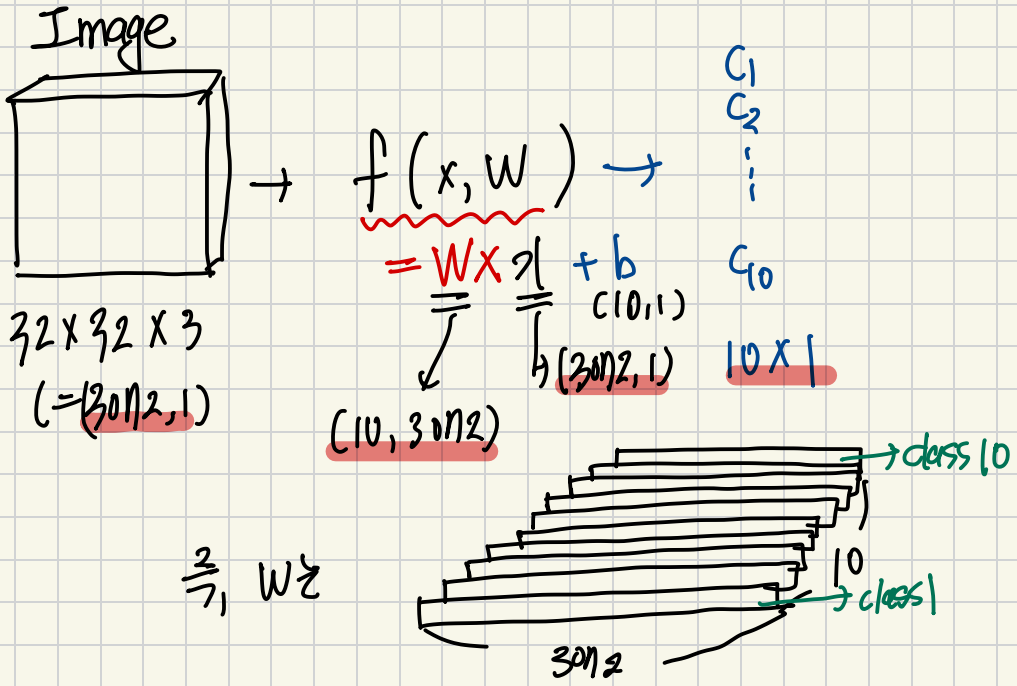
모든 pixel에 가중치를 곱해서 결과를 스코어로



⇒ max(이미지·w_1, 이미지·w_2, ··· 이미지·w_C)해서 C select.

이미지 / 각 class마다 w matrix 존재

$f(x, W) = x$랑 W를 모두 곱해서 라서 scalar 를 얻는다.



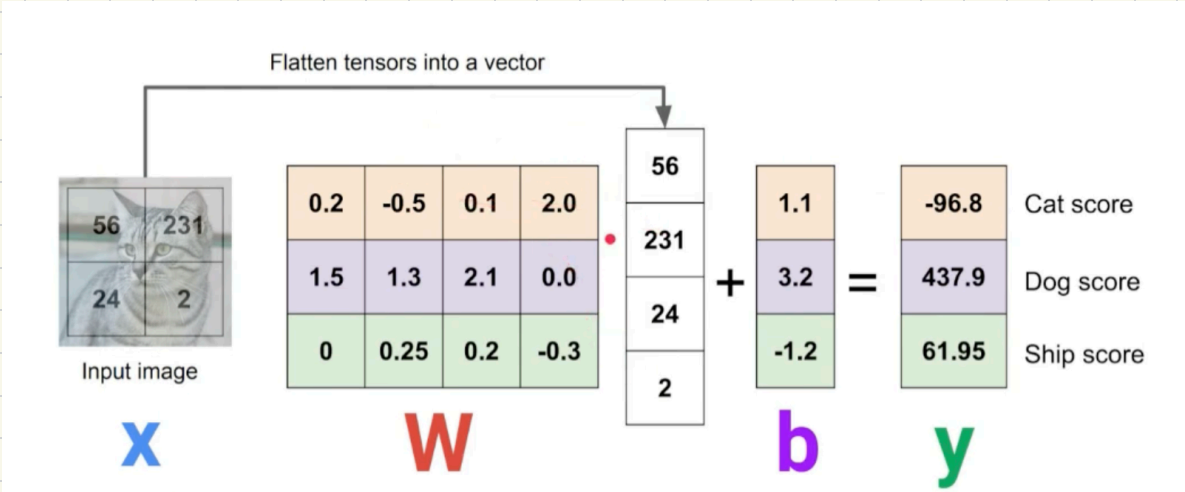
⇒ 입력의 각 pixel이 각 class에 얼마나 영향력 미치는지.

& bias를 $f(W,x)$에 넣기위해 $W = (10, 3073)$, $x = (3073, 1)$로 변경.

$f(x,W) = Wx$로 간단하게 변경가능.

Nearest neighbor는 predict시 train data수 만큼 compare.

Linear classifier는 W랑 class수 만큼 compare



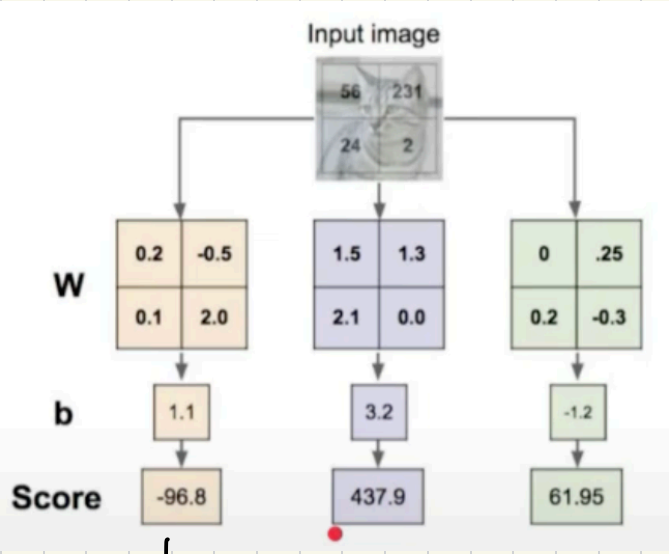
W가 학습된것이라고 가정하고 해석해보면.

이미지가 Cat이 되기 위해서는 내놓은 pixel이 중요하구나!

Ship " 고사용한 pixel이 "

bias는 이미지 자체에 Dog가 많고, Ship 같음을 알려줌.

Softmax classifier



↳ Linear classifier의 결과가 해석불가. $-\infty \sim \infty$ 여서 결과보면 엔빵크리몽 ⇒ 0~1 사이의 확률값으로 바꿔줘.

"Softmax function" 사용.

$$p(C_j|x) = \frac{e^{S_j}}{\sum_{i=1}^{n} e^{S_i}}$$

Loss function ⇒ 정답의 P와 예측 P를 유사하게!

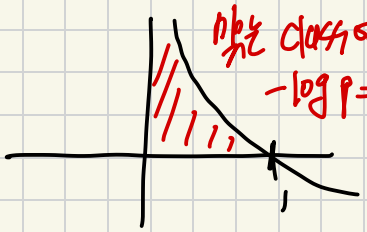
$$\text{cross entropy loss} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K} y_{ik}\log(\hat{y}_{ik})$$

→ 사실 class 1개만 살아남는다 나머지 $y_{ik}=0$

↳ 정답인 데이터에 대해서

↳ 정답 class에 대한 model의 확률값에 -log

맞는 class의 대한 p가 1일수록 -log p=0이되기에 Loss 0으로 -log를 붙인다.



KL Divergence (두 분포를 유사하게 만들수 있다)

$$D_{KL}(P||Q) = \sum_{i=1} P(i)\log\frac{P(i)}{Q(i)}$$