Learning method

- Supervised learning : labled data, exit loss,
  - ex) classification, regression,
    - O,X / ABC
    - 실값 자리 어느정도

- Un Supervised learning : no labled data, no loss
  - ex) Clustering, GAN, K-mean,

⇒ labeled 된 data 부족해 등장하였다.

- Semi Supervised learning
  - Labled 100개로 학습, 그후 un labled 1000개 예측후 그 예측한 값 또다시 학습한다.

- Self Supervised learning
  - un Labled data 가공해 학습가능하게 만들어 (feature 등을 학습) Supervised learning 진행

Confusion Matrix

Pred \ Actual	T	F
T	TP	FP
F	FN	TN

100개중 dataset이 어제가 T, 1개가 F면 모델은 무조건 T라고만 => 하게된다. 이것을 하기 위해 precision, recall, ROC curve 나옴

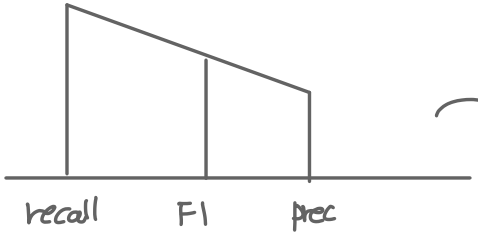
$\left(FPR = \frac{FP}{FP+TN} \right)$

- precision (정밀도) (모델입장) $= \frac{TP}{TP+FP}$ (날씨 맑다 예측했을때 실제 맑을 확률)

- recall (재현율) (hit rate) (정답입장) $= \frac{TP}{TP+FN}$ (실제 맑은 날씨 중 모델이 맑다고 한 확률)

=> 한쪽만 고려하는것은 일반적 모델 아님

- F1-Score $= 2 \times \frac{1}{\frac{1}{prec} + \frac{1}{recall}} = 2\frac{prec \times recall}{prec + recall}$ (조화평균)

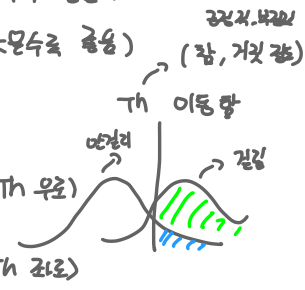


기하학적 의미 (각변좌표로 채워진 조화평균)

ROC Curve

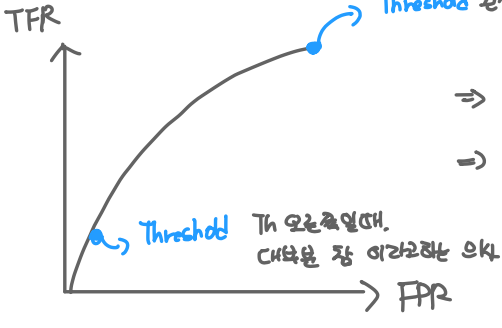
=> 성능평가 그래프로 오목한쪽을 볼록수록 좋음

(TPR (True Positive Rate) => 양을 양이라고 한 비율 (높을수록 좋음)
FPR (False 〃) => 양이아닌데 양이라고 한 비율 (낮을수록 좋음)



=> 하지만 둘은 비례 한다

흑시나 하여 양이라고 판단 양이해 의사 => TPR↑, FPR ↑ (Th 우로)
양이 아니라고 〃 => TPR↓, FPR↓ (Th 좌로)



=> FPR이 낮고 TFR이 높을수록 좋음 (좌,위)

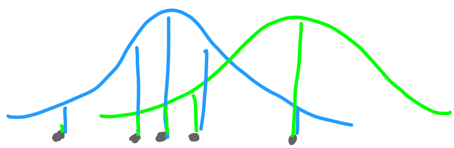
=> 모든 Threshold (모든 의사선생님) 를 고려하여 TFR, FPR 을 보겠다.

* 곡선의 확정도 => Classification 을 얼마나 더 잘하는지 나타내는 척도

MLE (Maximum likelihood estimation)

pdf = 확률밀도함수 $\int_{-\infty}^{\infty} f(x)\,dx = 1$ 어디서 확률 젤 높은지 알기쉬움

pdf를 모를때 data를 통해 pdf의 파라미터를 추정하는 방법



=> • 5개를 얻었다면 이 데이터는 파랑 or 초록중 어느 분포에서 왔을까를 추정

=> 직관적으로 보면 파랑이지만 이를 likelihood (가능도)의 곱으로 계산

높이 (가능도)를 $p(x_k|\theta)$라 하며 이들의 곱을 likelihood function 으로 정의

$p(x|\theta) = \prod_{k=1}^{n} p(x_k|\theta)$ 이때 $p(x|\theta)$의 값이 가장 커지게하는

θ값을 찾는것이 목적 => 미분하여 0이 되는것 찾음

그때 찾은 분포의 평균값 (확률 젤 높은부분을 측정값으로 쓰겠다.)

* 로그도 증가함수기 때문에 log를 붙일수 있으며 $\times \rightarrow +$ 되기때문에 log likelihood 자주사용한다

$$L(x|\theta) = \log P(x|\theta) = \log \prod p(x_k|\theta) = \sum \log p(x_k|\theta)$$

MAP (Maximum a Posterior estimation)

=> MLE는 prior 정보 잘 반영하지 못해, 이러한점 극복위해 사용함

목적 posterior 최대화 (bayes rule 식 그대로)

$$p(\theta|x) = \frac{p(x|\theta)\,p(\theta)}{p(x)}$$ 를 최대화하는 θ를 찾고 싶다

↳ $p(x)$도 알고 있으니 $\hat{\theta}_{map} = \arg\max_\theta p(x|\theta) = \arg\max_\theta p(x|\theta)\,p(\theta)$

Example (동장관고 보고 (x), 게임하는지 안하는지 판별 (θ))

- MLE => 게임한 사람중 그 동장관고 나올 likelihood
  Vs
  게임안 한사람중 그 〃 likelihood
) => 큰 높은것 선택
  prior (게임하는사람 영향 사람 비율) 은 결정에 반영 X

- MAP : 동장관고 (x) 주어졌을때 그게 게임을 하는 사람일 확률
  Vs
  게임 안하는 사람일 확률 (posterior)
) => 높은것 선택
  prior (게임하는사람 영향 사람 비율) 이 잘반영됨

- MAP 단점!

=> uncertainty 측정불가 (estimation 후 얼마나 신뢰할수 있는지 기준 x)

=> Overfitting 우려가 있음 (uncertainty 측정 불가능 때문)

Overfitting 막는법

=> dataset 늘리기

=> data augmentation

=> 모델 complex 줄이기

=> 가중치 규제 (regularization) L1 Norm, L2 Norm

=> Drop Out => 신경망 일부를 사용안하는것

Zeroshot Learning (ㅅ.말 알리고 얼룩말은 탑린거라 알려줄, 그럼 얼룩말 이라 맞춤)

=> online (실시간) offline (미리 학습시키고, 데이터 변한것도 계속 처리가능, Gradient 계속 20번 계산 가능)

↳ Gradient 1~2번 밖에 계산못함,

=> 데이터 한번보고 버림

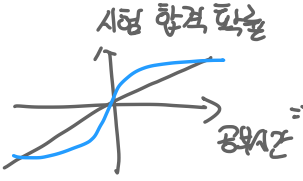
One shot => 한번이사진 1개 보여주고 다른것도 맞춤

few shot => 강아지 종여러종류, 알려주고, 새로운 종의 거 보여주면 맞춤

few shot
- meta learning => 여러개 task 동시 학습 + 각 task 파이도 학습 => few shot으로도 강한 범용적 모델 생성
- transfer learning => pre-trained 모델 중심으로 학습

Logestic Regression (지도학습)

=> data가 어떤 범주에 속할 확률을 0~1 값으로 예측하고 그 확률에 따라 가능성이 더 높은 범주에 속하는것으로 분류



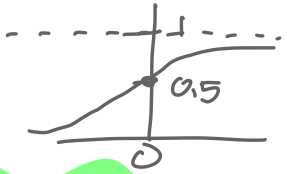
=> Linear regression은 $-\infty \sim \infty$ 으로 값이 벗어 나간다. 이는 말이안되는 상황들이 있어 확률로 모델의 범위 0~1로 만들어 표현

$$Odds = \frac{\text{일어날 확률}}{\text{안 일어날 확률}} = \frac{x}{1-x} \qquad log(odds) = 0 \sim 1 \text{ 가능}$$

여기서 (feature들 X Coeff)$_n$ + Intercept 으로 표현한다.

$$z = b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_n x_n$$

이를 Sigmod 로 표현 $h(z) = \frac{1}{1+e^{-z}}$ =>



Soft max regression

=> 로지스틱 회귀를 multi classification 이다.

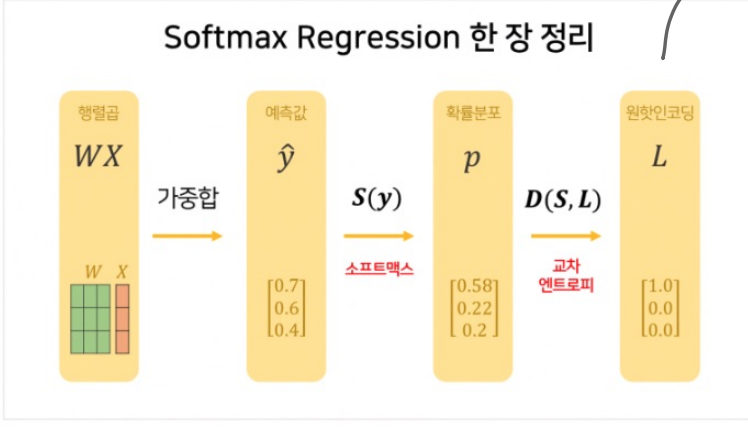
(로지스틱은 성공/실패 2가분류, soft max는 A, B, C 처럼 3개이상 classification)

- 확률 총합인 1의 확률 분포로 바꾸기 위해 이용 $S(y)$로 표현 되고 이를거쳐 확률분포 p로 바뀐다

$$S(y_i) = \frac{e^{y_i}}{\sum_j^k e^{y_i}}$$

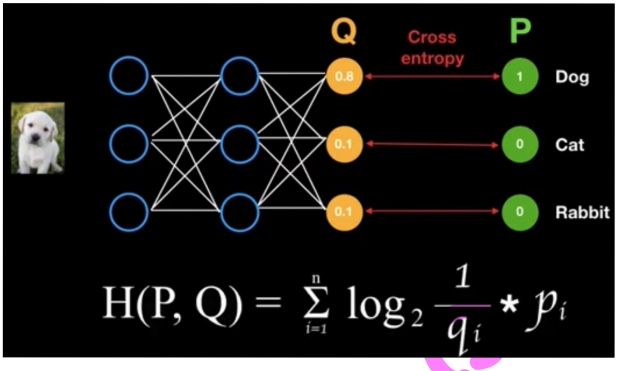
전체 그림

1 2 3. 가중치 다르니 0과 1로 가중치 없애고 공평하게 효율적임!



Cross Entropy

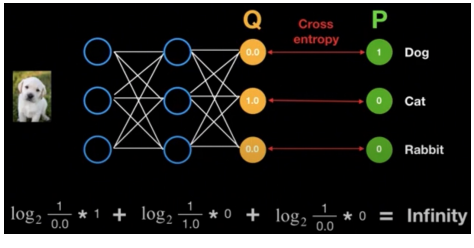
↳ 틀릴수 있는 정보 (모델의 결과) 를 구한 엔트로피



→ 그냥 엔트로피 식은

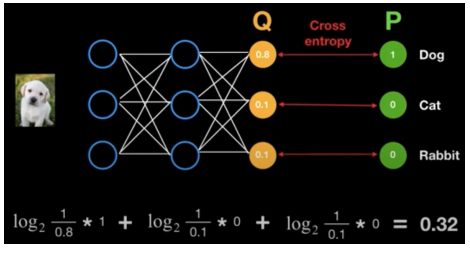
$\Sigma \log_2 \frac{1}{p_i} \times p_i$

예측한 확률 Q와 실제 결과인 P를 (Cross) 둘을 얼마나 근사하게 표현하고 싶은것



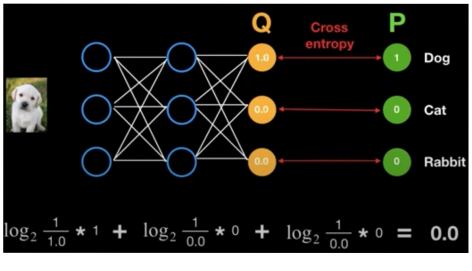
→ 학습이 안된 상태 (완전 틀림)

Cross Entropy 는 큰값 ∞



→ 어느정도 학습 되어 (꽤나 맞춤)

0.32 나옴



→ 그냥 100% 다맞출 값 = 0

→ loss Function 으로 사용해 이를 줄이도록 업데이트 한다.

Cross Entropy function

$$\text{Cost}(W) = -\sum_{j=1}^{k} y_j \log(p_j)$$

↳ k 개의 Class가 있을때 y_j는 one-hot-vector의 값

p_j는 j번째 Class일 확률 (추정치)

실제로 c가 one-hot-vec 이고, $p_c = 1$ 이면 $-\log 1 = 0$ 이됨.

Cost(W)를 최소화 하는 방향으로 해야하고, data n개 class k개일

$$\text{Cost}(W) = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{k} y_j^{(i)} \log(p_j^{(i)})$$ 가 되어야 한다.

Binary Cross Entropy (이건이거 변한것 뿐!)

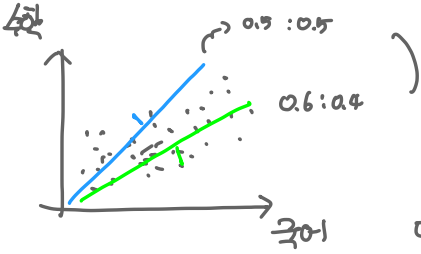
$$\sum_{j=1}^{2} y_j \log p_j = y_1 \log p_1 + y_2 \log p_2$$

$$= y_1 \log p_1 + (1-y_1) \log(1-p_1)$$

즉 $$\text{Cost}(W) = -\frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)} \log p^{(i)} + (1-y^{(i)}) \log(1-p^{(i)})\right)$$

PCA (Principal Component Analysis)

=> 데이터 분포에서 어떤 선분으로 정사영 시켰을때 데이터 분포 잘 표현 되었나? (국어 영어 중요성을 반영 비율 (벡터) 찾기



=> 이렇게 변형 비율 다르

5:5로 변형한거 6:4로 반영한거

데이터 분포따라 무엇이 잘 반영한건지 다르다! 최적의 projection 할 line 찾기

↳ 2차원 영역은 2개의 벡터를 찾아야 한다

principle component (주성분) = data의 분산이 가장 큰 방향 벡터 (잘 퍼짐)

ex) 얼굴인식, 1800개 사진 변화 1800개 주성분 벡터중 분산이 큰 20개 뽑아서 사용 (얼굴 수직 얼굴 전반적 형태, 뒤얼수록 세부적)

K-mean clustering

=> 키와 몸무게 에 따라 옷을 생산 할때, 모든 사이즈 고려할수 없어 S.M.L 로 사이즈 나눌때 이용가능 하다.

* Centroid · Cluster 의 중심

데이터 분포, k 정함. Centroid 설정

while (변함 없을)

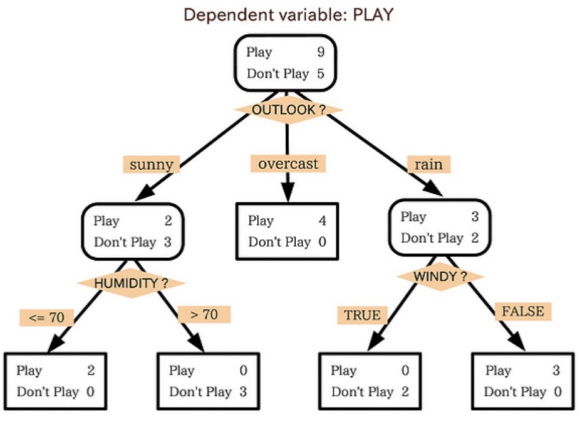
모든 데이터 Centroid 와 유클리드 거리로 분류

Centroid 를 Cluster 된 데이터의 중간으로 다시 설정

다시 새로운 Centroid 로 모든 데이터 유클리드 거리로 분류

=> K 최적 값을구 단점

Decison Tree



=> 질문을 던져 분류하고 input이 무엇인지 예측함

(불확실성)

-> 질문 어떤것을 먼저 하느냐에 따라 다른데 이때 사용되는 개념 entropy

$$H(X) = -\sum_{i=1}^{n} p_i \log_2 p_i$$

확률 정보의 양

			entropy	
일반적 동전	50%	50%	1	=> 값두 맞추기 어렵, 불확실성 높다
앞면만 있는 동전	100%	0%	0	=> 확실함, 불확실성 0
구부러진 동전	90%	10%	0.47	

=> 경우의 수가 2^n 개일때 entropy 최댓값 n

=> Decision tree 만들때 entropy 낮은 (확실한) 질문 우선으로해 초반에 잘 갈라 낸다

Random forest

→ 여러개의 의사결정 트리를 만들고, 투표시켜 다수결로 결과 진행

부트스트랩 => decison tree 생성

Bagging, Boosting => Random forest 생성

Naive Bayes Classify

$$P(B|A) = \frac{P(A|B) \times P(B)}{P(A)}$$

NBR 을 사용해 Classify 한다

↳ ex) SPAM 메일 분류

$$P(\text{Spam}|\text{free}) = \frac{P(\text{free}/\text{spam}) \times P(\text{spam})}{P(\text{free})}$$

(메일에 free 란 단어가 있을 때 Spam일 확률)

=> 하지만 단어가 train 할때 없었던 확률 0 이되는데
각 확률에 대해 분모 분자에 +1 을해주는 Laplace Smoothing 사용하기도 한다.

SVM (Support vector machine)

=> margin 을 최대화 (class 간 분류) 하는 선을 긋는것

* Support Vector = 선과 가장 가까운 point
* margin = Support vector 와 선의 distance
* decision boundary = 두 data 구분하는 선
* robust 하다 = outlier의 영향을 덜 받는다.

e.g.

	1	2	3	4	5
	1	2	3	4	100

	mean	median
	3	3
	22	3
	↓	↓
	not robust	robust

(outlier에 취약. 정확)

-> 무조건 margin이 큰순이 분류 잘하지 않음

그래서 원 데이터를 분류하는 범위 원리 찾고

그 범위 안에서 margin 최대화 하는선 찾음

but 목표가 outlier 있는 경우 어느정도 outlier 무시하고 선찾음

-> Gamma 와 decision boundary 휘어짐. 작다. 직선에 가깝다.