Machine Learning (3)

2020.10

References:

- **KPC, DSAC(Data Scientist Academy & Certificate) Manual, 2019**
- **many internet sites**

Classification Performance
(분류 성능)

분류의 손실 함수

- 분류에서는 손실함수로 MSE를 사용하지 않는다.
- 대신, 분류에서 정확도(accuracy)를 손실함수로 사용할 수 있다
  - 예) 100명에 대해 남녀 분류 문제
    - 96명을 맞추고 4명을 오 분류 : 정확도 0.96
  - 그러나 정확도를 손실함수로 사용하는 데에는 다음과 같은 문제가 있다
- Category 분포 불균형시 문제
  - 예)
    - Group : 남자 95명, 여자 5명
    - 오 분류 케이스 – 남자 1명, 여자 3명
    - 정확도는 여전히 0.96 :
    - 문제 : 여자의 경우, 5명 중 3명을 오 분류 ➔ 결과 심각
  - 데이터 분포가 비대칭인 상황 : 질병 진단의 경우 자주 발생
  - 손실을 제대로 측정하지 못함
  - 이를 보완하기 위해서 크로스 엔트로피(cross entropy)를 사용
    - Category가 둘 이상인 경우에도 동일한 개념으로 적용 가능

크로스 엔트로피(Cross Entropy)

$$CE = \sum_i p_i \log\left(\frac{1}{p_i'}\right)$$

- p_i : 어떤 사건이 일어날 실제 확률, p_i' : 예측한 확률
- 남녀가 50명씩 같은 경우

$$CE = -0.5 \times \log\left(\frac{49}{50}\right) - 0.5 \times \log\left(\frac{47}{50}\right) = 0.02687$$

- 남자가 95명 여자가 5명인 경우

$$CE = -0.95 \times \log\left(\frac{94}{95}\right) - 0.05 \times \log\left(\frac{2}{5}\right) = 0.17609$$

- **크로스 엔트로피 값이 작을수록 분류가 잘 수행된 것**

대표적인 손실함수와 성능지표

	손실함수	성능 지표
정 의	손실함수를 줄이는 방향으로 학습	성능을 높이는 것이 머신러닝을 사용하는 최종 목적
회귀 모델	MSE (오차 자승의 평균)	R^2
분류 모델	크로스 엔트로피	정확도, 정밀도, 재현률, F1점수

혼돈 매트릭스 (Confusion matrix)

y_pred

y_true	예측 클래스 0	예측 클래스 1	예측 클래스 2
정답 클래스 0	정답 클래스가 0, 예측 클래스가 0인 표본의 수	정답 클래스가 0, 예측 클래스가 1인 표본의 수	정답 클래스가 0, 예측 클래스가 2인 표본의 수
정답 클래스 1	정답 클래스가 1, 예측 클래스가 0인 표본의 수	정답 클래스가 1, 예측 클래스가 1인 표본의 수	정답 클래스가 1, 예측 클래스가 2인 표본의 수
정답 클래스 2	정답 클래스가 2, 예측 클래스가 0인 표본의 수	정답 클래스가 2, 예측 클래스가 1인 표본의 수	정답 클래스가 2, 예측 클래스가 2인 표본의 수

혼돈 매트릭스 – 이진 분류

- 혼돈 매트릭스
  - 분류의 결과가 잘 맞았는지를 평가하는 **채점표**와 유사

- 결과 값이 P(Positive)또는 N(Negative) 둘 중 하나만 가질 수 있는 **binary 예측**의 경우를 설명하는 일반적인 용어

- Positive는 **찾고자 하는 현상**(ex. 암에 걸린 사실, 결함 등)이 나타난 것인지를 구분하는 것일 뿐, 긍정적인 결과를 찾았다는 뜻은 아님

실제 \ 예측	P로 예측	N로 예측
실제로 P	True positive (TP)	False negative (FN)
실제로 N	False positive (FP)	True negative (TN)

혼돈 매트릭스(Confusion Matrix)

- 용어의 의미 예시
  - True positive (TP)
    - 암/결함이라고 예측했는데 실제로 암에 걸린 경우
  - False positive (FP)
    - 암/결함이라고 예측했는데 실제는 암에 걸리지 않은 경우
  - False negative (FN)
    - 암/결함이 아니라고 예측했는데 실제는 암인 경우
  - True negative (TN)
    - 암/결함이 아니라고 예측했는데 실제로도 암이 아닌 경우

True: **예측이 맞음**	Positive: positive**로 예측**
False: **예측이 틀림**	Negative: negative**로 예측**

모델의 성능 지표 – 혼돈 매트릭스

- **정확도(accuracy): 정확하게 예측한 비율을 의미**
  - **accuracy = (TP+TN) / 전체 경우의 수(N)**

실제 / 예측	암(예측)	정상(예측)	합계
암환자(실제)	6 (TP)	4 (FN)	10
정상(실제)	2 (FP)	188 (TN)	190
합계	8	192	200

  - **암진단 정확도 = (6 + 188)/200 = 194/200 = 0.97 => 97%**
  - **오류율 = 1-accuracy = 0.03 => 오진율은 3%**
- **리콜(recall): 관심 대상을 얼마나 잘 찾아내는가**
  - **recall = TP / (TP+FN)**
  - **실제 암 환자 발견률 = 6 / (6+4) = 0.6 => 60%**
- **정밀도(precision): 예측의 정확도**
  - **precision = TP / (TP+FP) = 6 / (6+2) = 0.75 => 75%**

모델의 성능 지표

- recall과 precision의 두 가지 지표를 동시에 높이는 것은 어려움,
- F1은 이러한 두 요소를 동시에 반영한 새로운 지표임
- F1은 recall과 precision의 조화 평균을 구한 것

$$F1 = \frac{2 \times precision \times recall}{precision + recall}$$

조화 평균: $\frac{1}{c} = \frac{(\frac{1}{a} + \frac{1}{b})}{2}$

$$c = \frac{2ab}{a+b}$$

- 두 지표의 값이 각각 0.5와 0.7일 때
  - 산술 평균 c=(a+b)/2=(0.5)+(0.7)/2=0.6
  - 조화 평균 c=2ab/(a+b)=0.7/1.2=0.58
- 두 지표의 값이 각각 0.9와 0.3일 때
  - 산술 평균 c=(a+b)/2=(0.9)+(0.3)/2=0.6
  - 조화 평균 c=2ab/(a+b)=0.54/1.2=0.45

비용 최소화 – 병원 사례

- 1)의 경우 수익이 발생하며 이를 편의상 200만원 (E1)
- 2)의 경우 병원은 약간의 손실이 생기며 이를 -3만원 (E2)
- 3)의 경우는 병원에 큰 손실이 생기며 이를 -500만원 (E3)
- 4)의 경우 환자를 더 유치하므로 평균 이득이 2만원 (E4)
- 각 경우의 **발생 확률**과 **각 이득** 또는 비용을 곱하여 더하면 **총 기대치**를 구할 수 있다

비용 최소화 – 병원 사례

- **확률**

	암이라고 예측	암이 아니라고 예측
실제 암환자	p1 = 6/200=0.03	p3 = 4/200=0.02
실제로는 암 없음	p2 = 2/200=0.01	p4 = 188/200=0.94

- **예상 비용**

	암이라고 예측	암이 아니라고 예측
실제 암환자	E1 = 200만원	E3 = -500만원
실제로는 암 없음	E2 = -3만원	E4 = 2만원

비용 최소화 – 병원 사례

- **전체 기대치 : 위 두 테이블을 항목별로 곱한 후 더한다**

$$\text{전체 기대치} = p1E1 + p2E2 + p3E3 + p4E4$$
$$= (0.03*200) + (0.01*(-3)) + (0.02*(-500)) + (0.94*2) = -7.55\text{만원}$$

- **위의 의사는 암환자 진단 결과로 오히려 병원에 손실을 발생**
- **이렇게 손실이 발생한 원인은 무엇일까?**
  - **실제는 암인데 암이 아니라고 잘못 판정한 경우**

    (즉, FN의 댓가가 평균 -500만원으로 크기 때문)
  - **이 비용을 줄이려면 p3 확률을 줄이도록 노력해야 한다.**
  - **조금만 의심이 들어도 모두 "암 같은데요" 라고 판정해 주면 FN을 줄일 수 있다.**
  - **500만원 비용보다 -3만원 손실이 훨씬 적기 때문**

비용 최소화 – 병원 사례

- **보수적인 의사**

	암이라고 예측	암이 아니라고 예측
실제 암환자	10	0
실제로는 암 없음	90	100

실제 / 예측	암이라고 예측	암이 아니라고 예측
실제 암환자	p1 = 10/200=0.05	p3 = 0/200=0
실제로는 암 없음	p2 = 90/200=0.45	p4 = 100/200=0.5

$$\text{전체 기대치} = p1E1 + p2E2 + p3E3 + p4E4$$
$$= (0.05*200) + (0.45*(-3)) + 0 + (0.5*2) = 9.65\text{만원}$$

비용 최소화 – 병원 사례

- 이 의사의 진단 능력을 종합해 보면
  - 정확도(accuracy) : 55%로 나쁜 편 (110/200)
  - 재현률 : 10/10 = 100% (10명의 암환자를 모두 찾음)
- 앞의 의사의 경우
  - 정확도(accuracy) : 97%
  - 재현률(recall)이 60%
- 평가
  - 암 환자를 찾아내는 비율(재현률)이 높아졌다
    - 따라서 오진으로 인해 병원이 지출할 비용도 줄여주었다.
  - 그러나 정밀도(precesion)는 10 / (10+90) = 5%이며 이는 앞의 의사의 75%에서 크게 감소했다. 즉, 암이라고 진단해도 그 중에 5%만 암이고 나머지는 과잉 진단
  - 과잉 진단은 장기적으로 병원과 의사의 신뢰도를 떨어뜨릴 수 있다
- 판정 기준은 병원의 철학과 전체 비용에 따라 다르게 설정될 것이다.

동적 성능 평가

- 정적 성능 평가
  - 최종 분류의 결과만 본다

- 동적 성능 평가
  - 정적인 성능 평가 보다 알고리즘의 동작을 좀 더 세밀하게 평가
  - 최종 분류 결과만 보는 것이 아니라 분류한 순서를 평가하는 방법
  - ROC, AUC

동적 평가 – 분류 순서(rank) 평가

환자번호	성별	점수	순위	실제 값
7	F	0.98	1	N
125	M	0.96	2	C
4	F	0.95	3	N
199	M	0.86	4	C
2	F	0.84	5	N
200	M	0.82	6	C
176	M	0.81	7	C
73	M	0.80	8	N
82	M	0.79	9	C
3	F	0.77	10	N
123	F	0.76	11	N
		...		C
43	F	0.48	198	N
93	M	0.42	199	N
120	F	0.40	200	N

점수 : 예) 암일 확률

순서의 어디까지를 양성이라고 판정할지에 따라 혼돈 매트릭스가 달라진다

F T F T F T T F

F T F T F T T F T F F T

보수적

ROC (Receiver Operating Characteristic)

- 예측 결과를 순서대로 제시한 것이 실제 값과 얼마나 순서에 따라 잘 맞는지는 검증하는 2차원 그래프

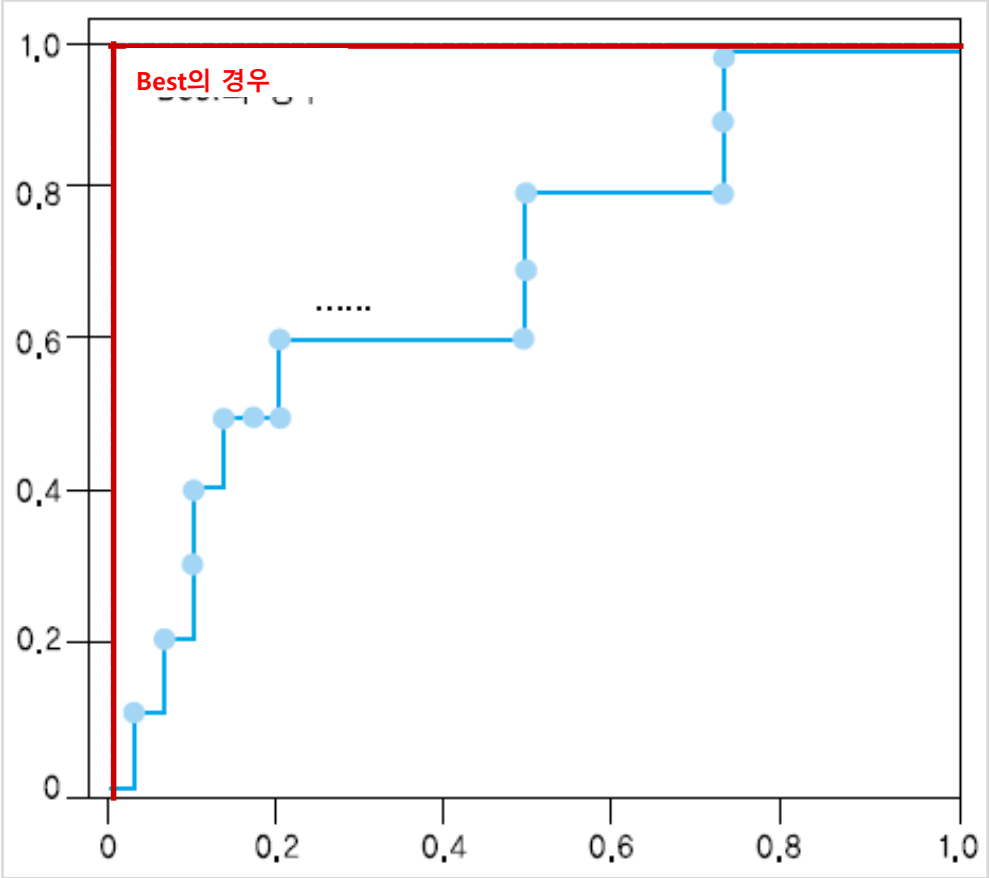
- ROC 커브는 (0,0)점에서 시작하여 한 행씩 진행하면서 정답을 맞추었으면 y축 위로 한 칸 이동, 정답을 맞추지 못했으면 x축 방향으로 한 칸 이동. 종점은 (1, 1) 지점

- 그래프의 x 축으로는 예측 오류가 날 때마다 이동하고, y축으로는 정답을 맞출 때마다 이동

- x축은 예측이 틀린 것을 나타내므로 false positive rate, y축은 예측이 맞은 것을 나타내므로 true positive rate를 나타냄

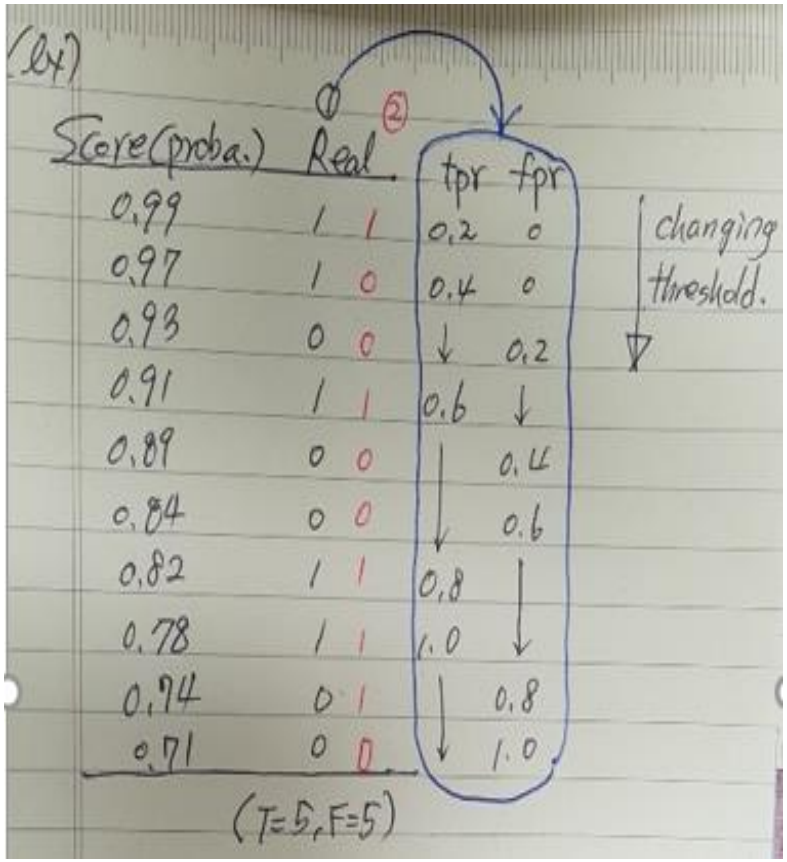
분류 순서 평가

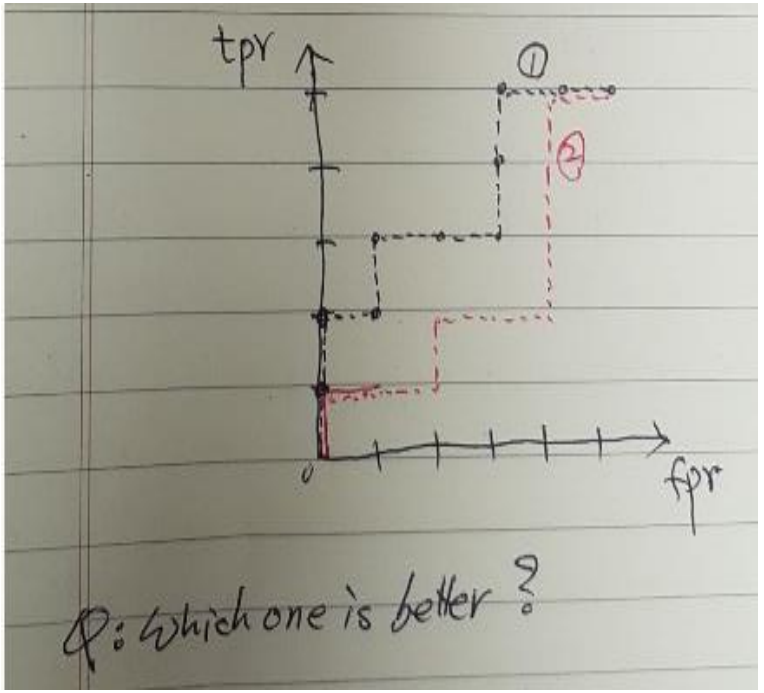
- 각 데이터 항목에 대해 계단형 그래프가 만들어짐
- 만일 의사의 예측 정확도가 높다면, 그래프가 초반에 위로 올라갈 것임



ROC

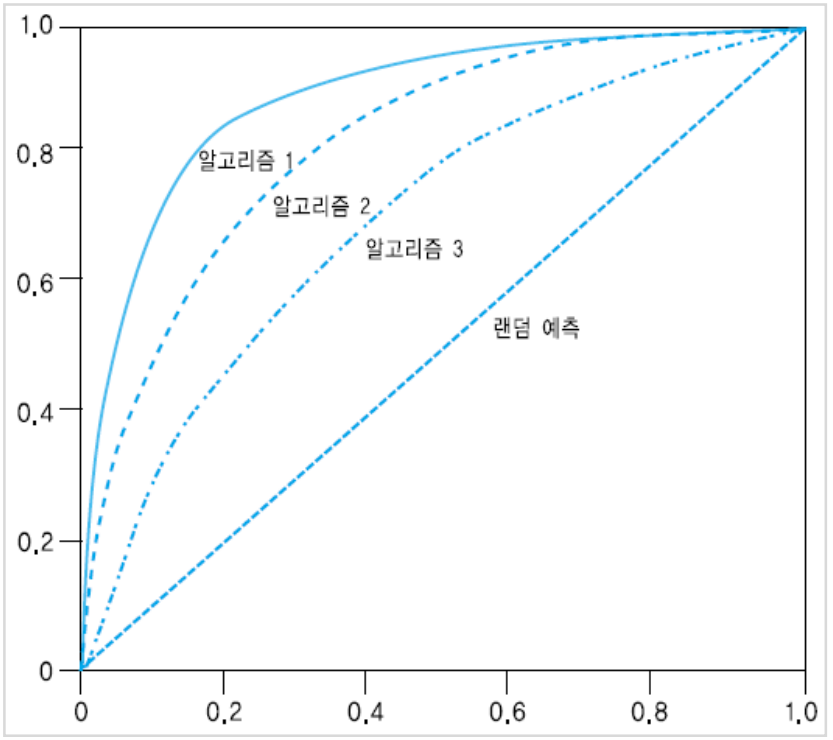
- **(ex)**


(ex)
Score(proba.)
Real
tpr
fpr
0.99 1 1 0.2 0
0.97 1 0 0.4 0
0.93 0 0 ↓ 0.2
0.91 1 1 0.6 ↓
0.89 0 0 | 0.4
0.84 0 0 ↓ 0.6
0.82 1 1 0.8 |
0.78 1 1 1.0 ↓
0.74 0 1 | 0.8
0.71 0 0 ↓ 1.0
(T=5, F=5)
changing threshold.


tpr
fpr
0
Q: Which one is better?

AUC (Area Under Curve)

- 예측 알고리즘의 성능을 간단히 수치로 나타내기 위해서 ROC 그래프의 면적을 계산하는 방법을 사용
- 우수한 알고리즘일수록 초반에 y축 상단 방향으로 이동하므로 ROC 커브의 면적이 넓어짐



다중 분류

- 다중 분류
  - 여러 개의 category를 가질 수 있는 입력 데이터를 분류
- sklearn의 이진분류 함수들은 다중 분류를 지원
  - 내부적으로 이진 분류를 확장해서 수행(해당 category인 것과 아닌 것)
  - 이러한 방식 : One-versus-Rest(OvR)방법
- 분류 결과만 알려면
  - predict()
- 다중 클래스 각각에 해당할 점수 또는 확률을 알려면
  - decision_function()
  - predict_proba()

다중 분류

- 3개 이상의 클래스 중에 하나를 예측해야 하는 경우
  - 로지스틱 회귀를 그대로 사용할 수 없다
  - 다항 로지스틱 회귀(multinomial logistic regression)를 이용
    - 다중 분류 (multiclass classification)
    - Softmax 함수 사용

- 이진 분류를 이용한 다중 분류 : OvR(One vs Rest) 방법
  - (예, A, B, C 분류)
    - A, {B, C}
    - B, {A, C}
    - C, {A, B}

- 한번에 다중 분류 가능 : 랜덤 포레스트, 나이브 베이즈 등
- 소프트맥스(softmax) 함수를 사용

$$\sigma(j) = \frac{\exp(\mathbf{w}_j^\top \mathbf{x})}{\sum_{k=1}^{K} \exp(\mathbf{w}_k^\top \mathbf{x})} = \frac{\exp(z_j)}{\sum_{k=1}^{K} \exp(z_k)}$$

소프트맥스 (Softmax)

- 소프트맥스(softmax) 함수를 사용

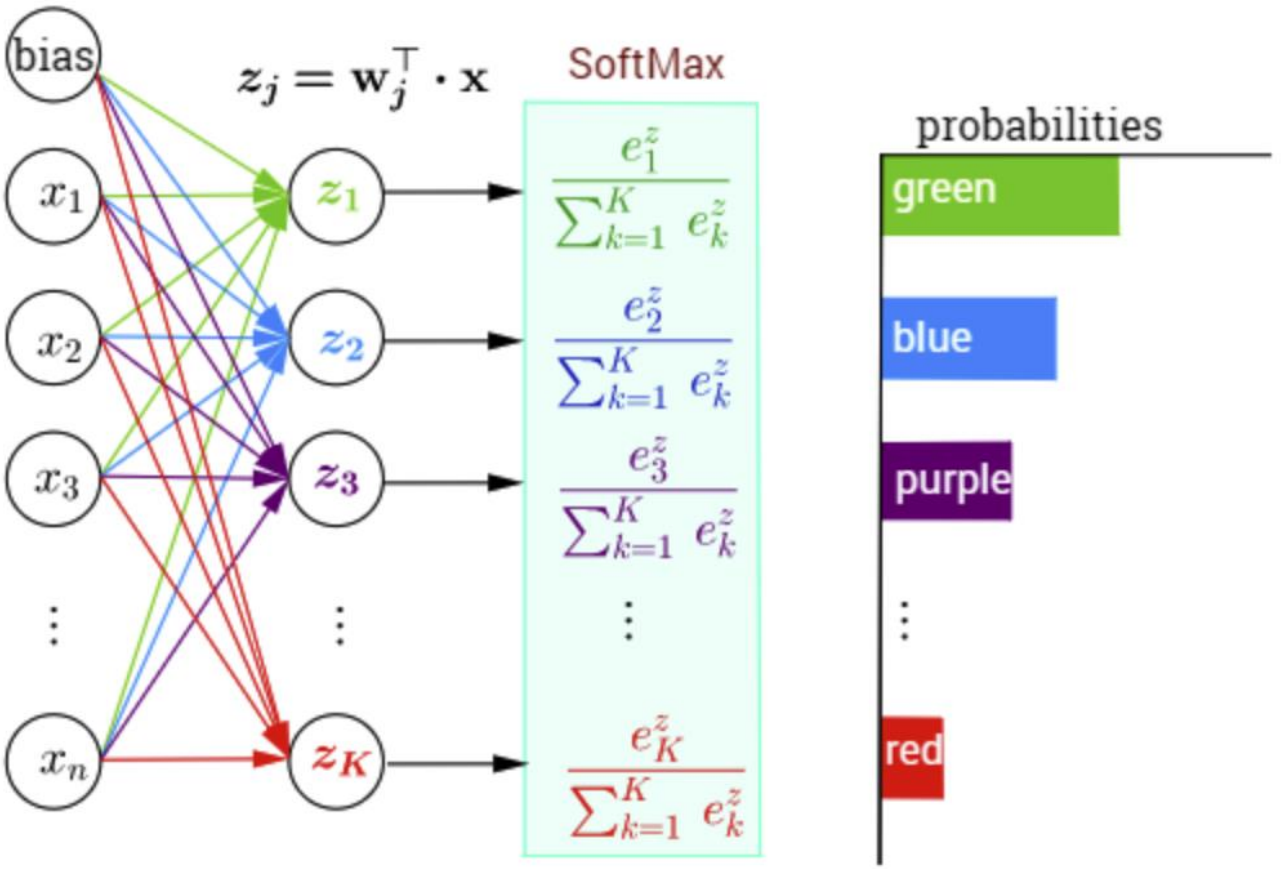
$$\hat{p}_k = \sigma(s(x))_k = \frac{\exp(s_k(x))}{\sum_{j=i}^{K} \exp(s_j(x))}$$

  - $\widehat{p_k}$: 클래스 k에 속할 확률
  - x : 주어진 샘플
  - $S_k(x)$: 소프트맥스 회귀 모델이 각 클래스k에 대한 점수

- 예) 얼굴을 보고 한국인, 중국인, 일본인 3 class로 구분
  - 어떤 샘플에 대해서 모델의 예측 값이 1.5, 2.0, 1.8 로 가정
  - 소프트맥스 적용 : 0.23, 0.43, 0.34 (합 = 1.0)
    - 각각의 점수를 확률처럼 사용 가능
    - 모델 예측 값이 음수(-)이어도, 소프트맥스 출력 값은 0~1 사이 값

소프트맥스

- 상대적인 점수 비교 : 확률처럼 0~1 사이 값으로 매핑



특성공학

차원 축소

- 기존 여러 특성 중 분석에 가장 영향력이 있는 특성을 선택하는 것이 필요한 때가 있다
  - 학생들의 학업 능력을 평가
    - 모든 과목의 성적을 다 사용하지 않고 국어, 수학 성적이 대표성이 있다면 이 두 과목의 성적만 평가 점수로 사용

- 특성 공학
  - 머신 러닝에서 사용할 특성을 잘 선택하는 것
  - 유효한 특성을 잘 선택하면 학습 속도가 빨라지고 성능도 좋아진다.

- 차원 축소
  - 머신 러닝의 성능을 떨어뜨리지 않으면서 특성의 수를 줄이는 기술

- 차원 축소가 필요한 이유
    - 계산량, 메모리 사용량을 줄이기 위해
    - 샘플의 특징을 보기 좋게 시각화하기 위해

차원 축소

- 자동으로 영향력이 큰 특성 선택 방법
  - 목적 변수와 상관관계가 높은 변수를 선택
  - SelectPercentile()
    - 상관관계가 높은 순서대로 특성들을 나열하고
    - 상위 몇 %까지의 특성을 선택해 줌

- 머신 러닝에 별로 도움이 되지 않는 특성들을 제거함으로써 모델 개발 시간을 줄일 수 있다

주성분 분석 (PCA)

- 주성분 분석 (Principal Component Analysis)
  - 여러 속성들을 조합하여 이들을 대표할 수 있는 적은 수의 특성을 찾아내는 작업
  - 기존 속성들의 선형 조합 : 가장 많이 사용
    - 예 : 학생의 능력 – 주성분 (수업 능력, 활동 지수)
    - 수업 능력 : 국어, 영어, 수학 성적(가중치 0.4, 0.3, 0.3)을 각각 곱해 더한다
    - 활동 지수 : 독서량, 운동량, 친구(가중치 0.4, 0.3, 0.3)을 각각 곱해 더한다
- 가장 적절한 주성분 찾기
  - 기존의 속성 값들을 어떤 비율로 가중 합산해야 할 지는 컴퓨터가 자동으로 여러 조합을 만들어 보고 최적의 조합을 찾아준다
- 최종적으로 필요한 주성분의 개수는 알려 주어야 한다.

주성분 분석 (PCA) 알고리즘

- **Data Preprocessing**
  - **Mean normalize and feature scaling (opt**

$$x_j^{(i)} \leftarrow \frac{x_j^{(i)} - \mu_j}{s_j}$$

- **Compute Covariance matrix, Σ**

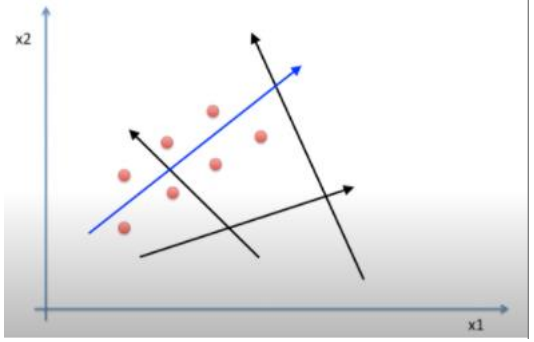
$$\Sigma = \frac{1}{m} \sum_{i=1}^{n} (x^{(i)})(x^{(i)})^T$$

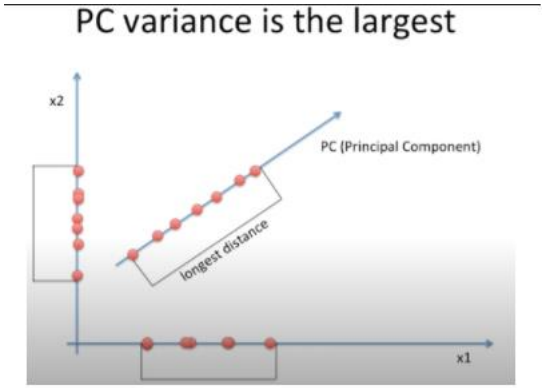
$n \times 1$ $1 \times n$ $n \times n$

- **Compute eigenvectors of matrix, Σ**
  - **Use Eigen decomposition or SVD**
- **Choose k (number of principal components)**

주성분 분석 (PCA) 예제

- (ex) 2차원 -> 1차원 변환





- 무수히 많은 화살표 중 , 데이터를 화살표네 projection 했을 때 최대한 겹치지 않게, 그리고 멀리 퍼지게 하는 길이가 긴 화살표를 찾아야 함.
- 거기에 데이터 투영
- (차원이 높은 경우) 또 하나의 축이 필요하다면 서로 직각이 되어야 함. 최대한 데이터가 겹치지 않게.

➔ (linear algebra)

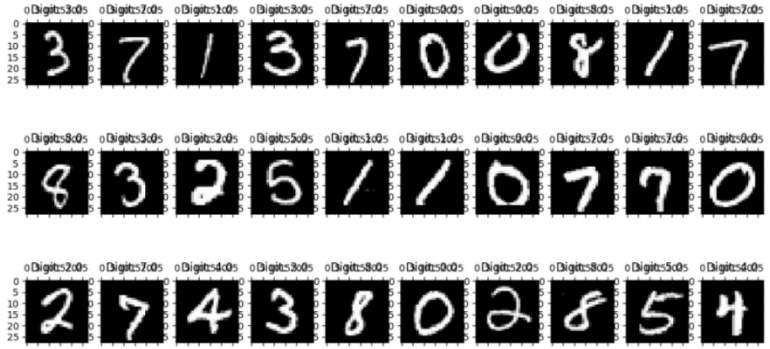
- 공분산 행렬에서 고유벡터/고유값을 찾고,
- 가장 분산이 큰 방향으로의 고유벡터 e1에 입력데이터 선형 변환
- e1 에 직교하며, 다음으로 분산이 큰 e2 고유벡터로 또 선형변환

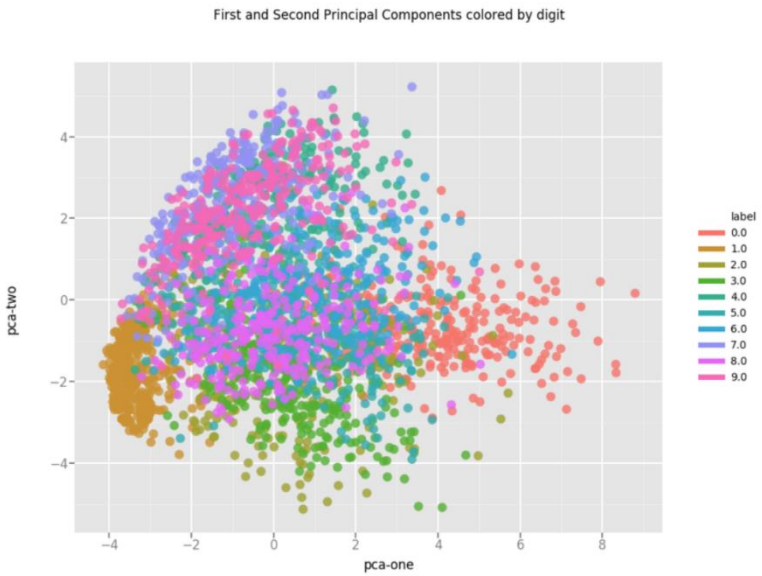
t-SNE (Stochastic Neighbor Embedding)

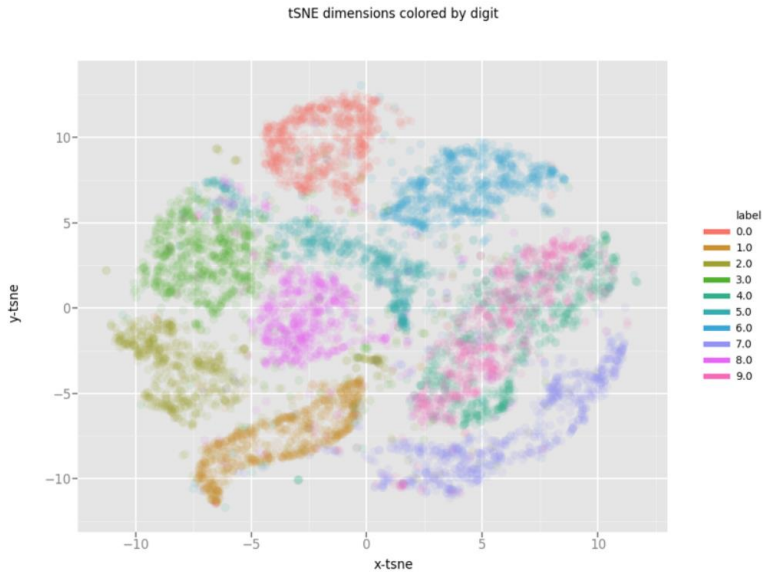
- 데이터의 특성을 한 눈에 파악하는데는 시각화가 매우 유용
  - 그러나 실제 세계의 데이터는 속성의 차원이 높으므로 이러한 다차원 공간에 속성들의 관계를 그리는 것은 불가능
  - 명확한 시각화를 위해서는 데이터를 2, 3차원 이하로 변환 필요
    - 비지도 학습
- t-SNE
  - 시각화를 위해 고차원의 특성을 가진 데이터를 저차원으로 축소하는 기술
  - 보통 word2vec으로 임베딩한 단어벡터를 시각화하는 데 많이 사용

t-SNE

- MNIST 데이터 30개
  - 28x28의 차원(즉, 784 차원)의 데이터
  - 2차원으로 축소 : PCA, t-SNE







모델 최적화

릿지 규제

- 모델을 일반화 하는 방법으로, 손실함수로 MSE 항목과 함께 계수의 크기 자체도 줄이도록 하는 방법을 도입한 것

$$J(W) = MSE(W) + \alpha \frac{1}{2} \sum_{i=1}^{n} W_i^2$$

- 계수의 자승의 합을 손실함수에 추가
  - 계수 자체가 가능하면 작은 값이 되기를 원한다
  - α : 이 축 항목의 비중을 얼마나 크게 할지를 정하는 하이퍼파라미터
- 이는 여러 계수들을 가능한 골고루 반영하라는 의미
  - 왜냐하면 계수의 자승의 합을 줄이려면 각 계수의 크기를 줄여야 전체 자승의 합이 최소화되기 때문

릿지 규제

- 그러나 릿지 규제를 너무 강하게 하면 MSE 항은 무시되고, 모든 계수의 값이 동일하게 된다.
- 릿지 규제는 선형회귀에서만 사용되는 것이 아니라 SVM, 신경망 등 다른 머신러닝 모델에서도 사용될 수 있다.
- 릿지 규제는 L2 규제라고도 부른다

라쏘 규제

- 라쏘 규제에서는 모든 계수의 절대치들의 합을 추가로 더하는 방법 (자승을 취하지 않음)

$$J(W) = MSE(W) + \alpha \sum_{i=1}^{n} |W_i|$$

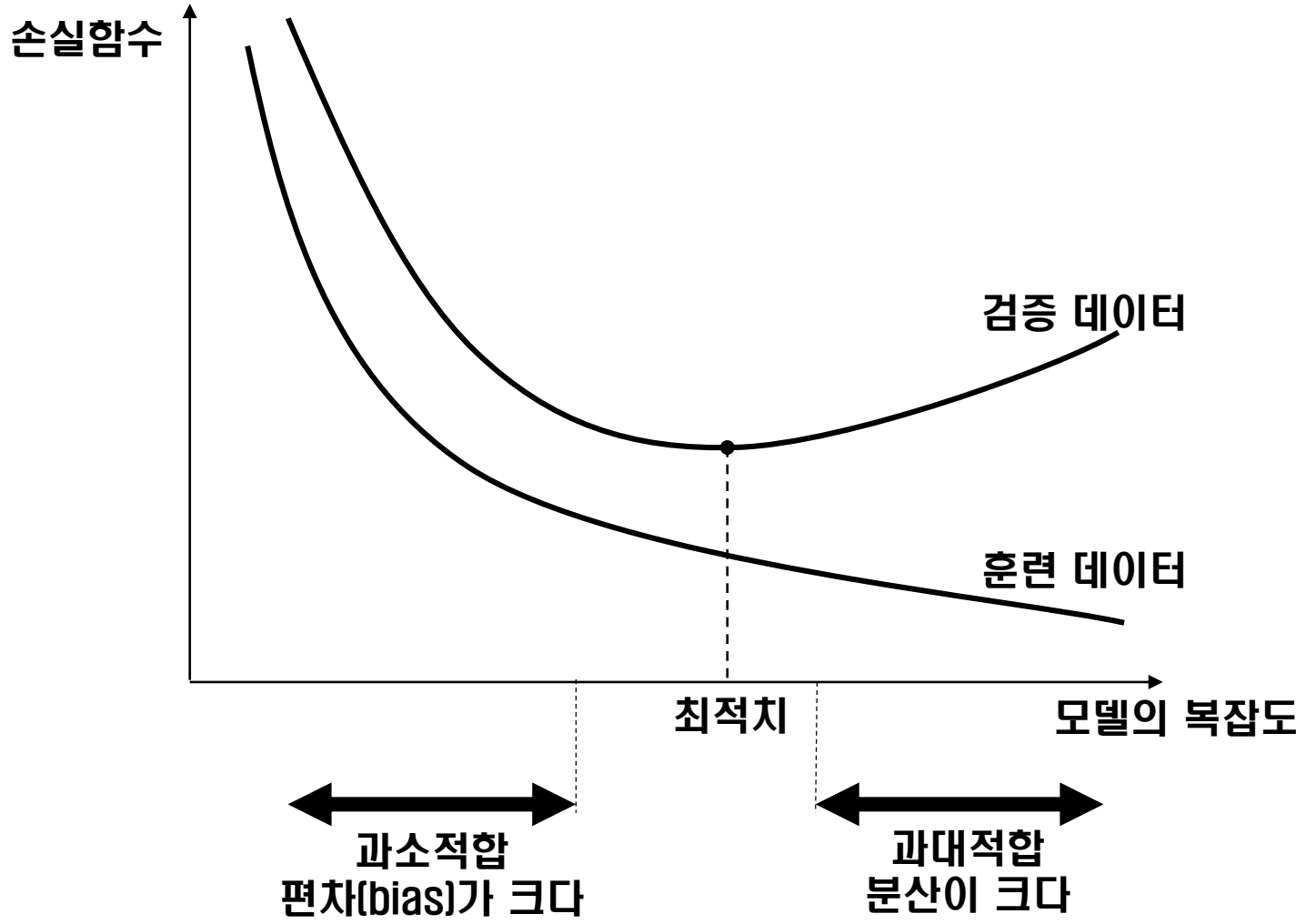
- 릿지 규제와 유사한 것처럼 보이지만 사실은 반대의 효과
- 릿지 규제에서는 특별히 비중이 큰 계수를 지양하고, 가능한 여러 가중치를 골고루 반영하는 효과
- 라쏘 규제를 적용하면 절대값이 작은 계수가 먼저 사라지는 효과
  - 라쏘 규제는 L1 규제라고도 함

엘라스틱 넷

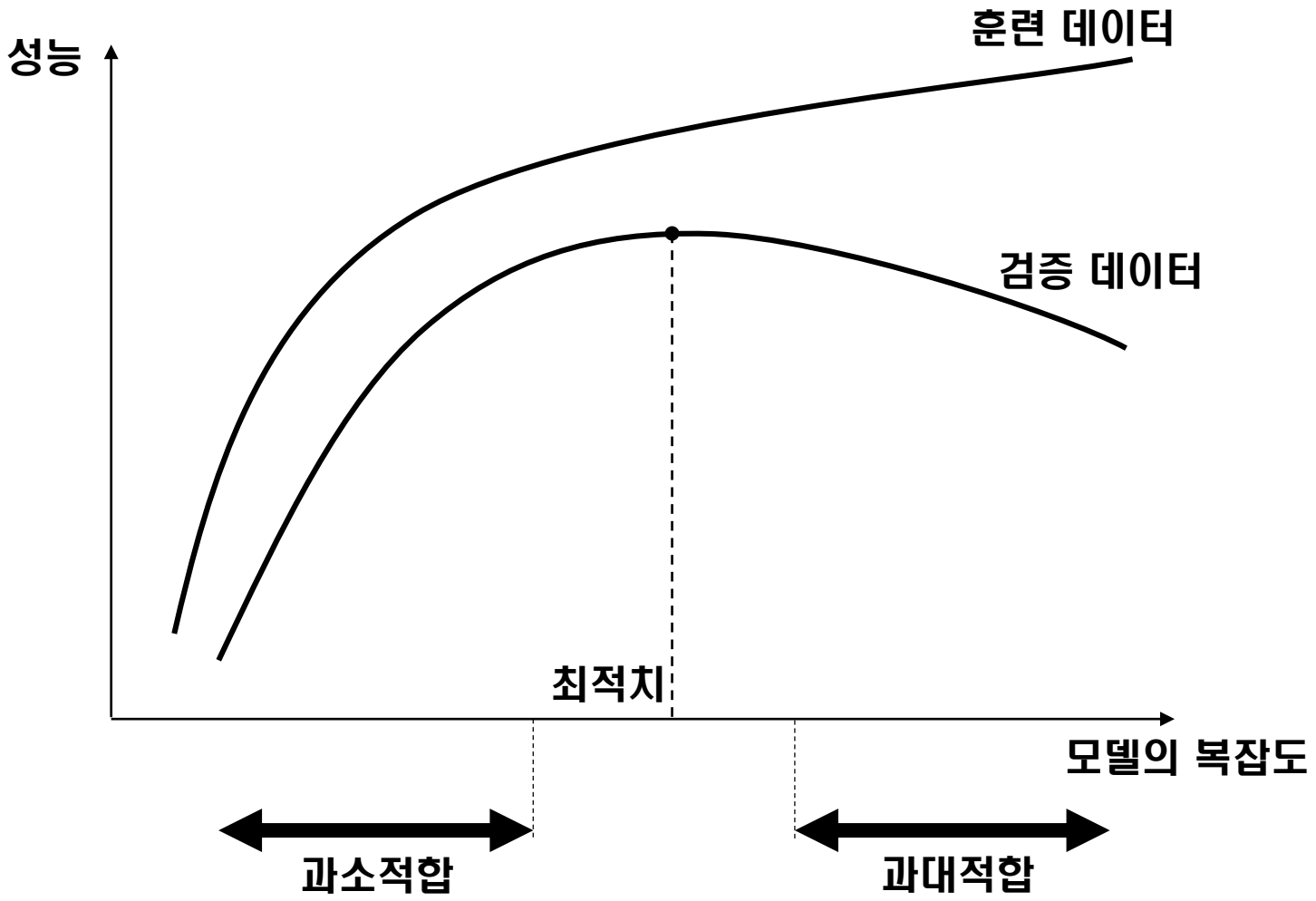
- 릿지 규제와 라쏘 규제가 동시에 필요한 경우가 있다.
  - 특정 계수가 크게 영향을 주는 것도 피하고 싶고(L2 규제)
  - 동시에 영향력이 적은 계수의 수를 줄이는 것이 필요할 수가 있다(L1 규제)
- 아래에서
  - α : 일반화의 정도를 조정하는 하이퍼파라미터
  - γ : L2와 L1 규제의 반영 비중을 조절하는 하이퍼파라미터

$$J(\theta) = MSE(\theta) + \gamma\alpha\sum_{i=1}^{n}|\theta_i| + \frac{1-\gamma}{2}\alpha\sum_{i=1}^{n}\theta_i^2$$

과소 적합과 과대적합 판단 – 손실함수



과소 적합과 과대 적합 판단 – 성능



편향과 분산

- 예측 모델에서 발생하는 오차는 분산(variance)과 편향(bias) 두 가지 성분으로 설명할 수 있다.
- 분산이란 모델이 너무 복잡하거나 학습데이터 민감하게 반응하여 예측 값이 산발적으로 나타나는 것이다.
- 편향이란 모델 자체가 부정확하여 피할 수 없이 발생하는 오차를 말한다.

편향과 분산