

DSAC Module2

(Data Process and Analysis)

2019.9

KPC(한국생산성본부)

목차

2

Title	Contents	Labs(gg-?)
I. 데이터분석 개요 (Data Analysis Basics)	<ul style="list-style-type: none">데이터분석 목적데이터분석 특징 <ul style="list-style-type: none">데이터분석 유형데이터분석 프로세스	
2. 탐색적분석 (EDA: Exploratory Data Analysis)	<ul style="list-style-type: none">탐색적 분석 정의통계적분석시각화 연습 <ul style="list-style-type: none">탐색적 분석 예Seaborn	17,18, 19,20,21,22,23,24
3. 데이터 전처리 (Data Preprocessing)	<ul style="list-style-type: none">전처리 개요스케일링 <ul style="list-style-type: none">데이터 변환전처리 실습	25,26
4. 클러스터링 (Clustering)	<ul style="list-style-type: none">유사도클러스터링 개요DBSCAN <ul style="list-style-type: none">공간거리클러스터링 예클러스터링 비교	27,28
5. 선형회귀 (Linear Regression)	<ul style="list-style-type: none">선형모델선형회귀 예제 <ul style="list-style-type: none">회귀 손실함수규제화 (Regularization) 연습	29,30,31
6. 선형분류 (Linear Classification)	<ul style="list-style-type: none">선형분류종류결정경계 <ul style="list-style-type: none">분류 손실함수교차검증	32
7. 로지스틱 회귀 (Logistic Regression)	<ul style="list-style-type: none">개념다항로지스틱 회귀 <ul style="list-style-type: none">성능 비교	33

데이터 분석

데이터 수집

- 전체 과정에서 70~80%의 시간을 소모함
- 핵심 데이터를 확보했는지 여부
- 데이터 품질
- 잘못된 데이터 사용은 잘못된 결과를 도출

데이터 수집 기술

5

- 수집 가능 여부 (보유 기관의 데이터 정책)
- 수집 주기 (일회성, 한시간/하루/한달에 한번 등)
- 비용 (무료 또는 유상, 통신 비용 등)
- 데이터 포맷 변경, 호환성, 처리 비용
- 정답 데이터 셋 확보 여부
- 수집 아이디어...
- (ex)구글 – 사람들의 이동 장소, 때 등 정보 수집



Data, Data, Data

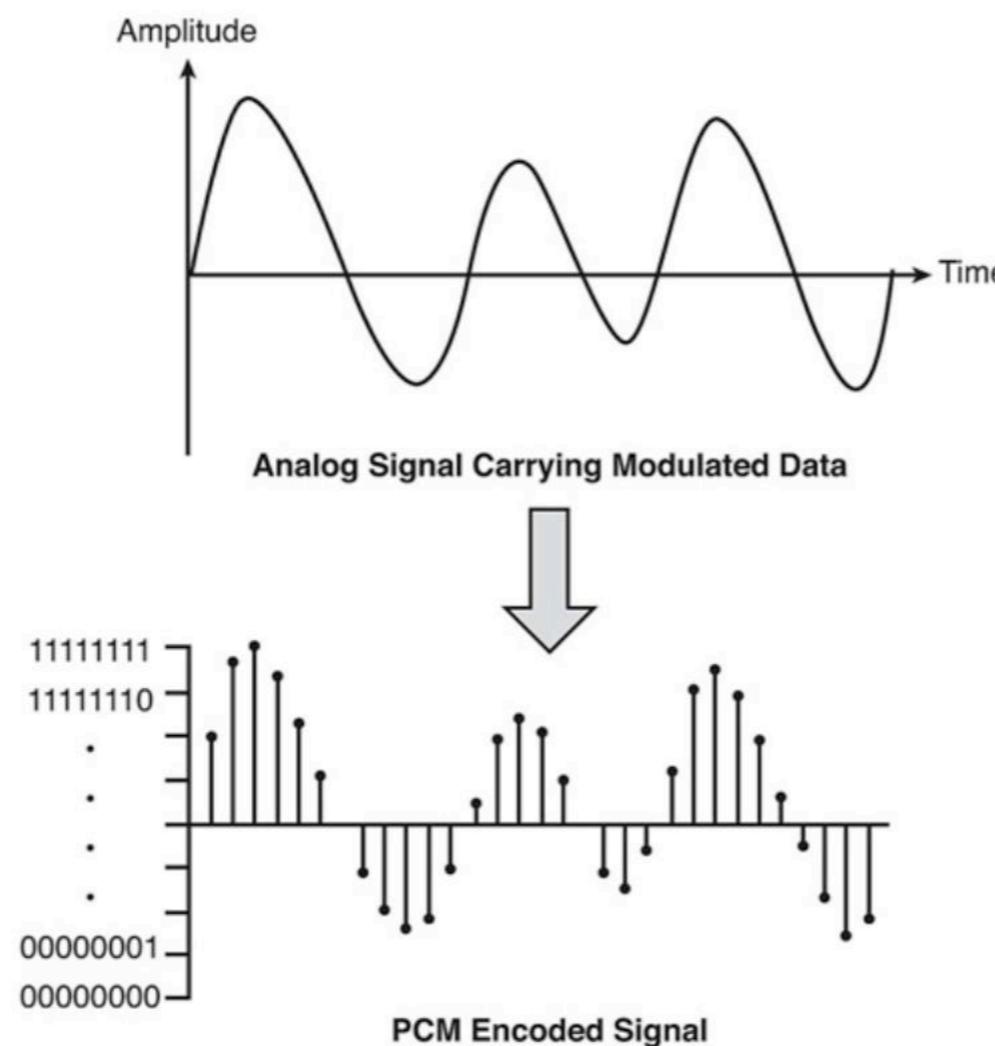
6



신호 샘플링

7

- 데이터 수집의 수준 결정
- 나이퀴스트의 샘플링 이론 (Nyquist sampling theorem)
 - 신호의 최고 주파수 시간의 두 배 속도로 샘플링



- 데이터분석 타입
 - descriptive – 설명적 분석
 - predictive – 예측 분석
 - recommendation - 추천

Descriptive 분석 (통계적 분석)

9

- 고객, 비즈니스 프로세스, 성과, 데이터를 이해하는 것
- 예
 - market research,
 - funnel analysis (고객의 행동을 분석해 마케팅 효과 극대화)
 - dashboard reporting (한 눈에 볼 수 있도록 모든 차트나 그래프를 한꺼번에 보여줌)
 - exploratory data analysis (EDA) : (ex) boxplot
- 결과는 인사이트를 얻는 것
 - 비즈니스 인텔리전스

Predictive 분석

10

- 새로운 샘플에 대한 미래 값을 예측하는 것
- 예
 - 날씨, 주가, 판매량 예측
 - 스팸 메일 예측, 질병 예측
 - 고객 세그멘테이션 예측
 - 사이버 공격의 위험 예측

- 최종적으로 의사결정을 돋는 것
 - 단순히 정보(인사이트)를 주는 것을 넘어서 최적의 추천
 - 다양한 설명 및 예측 모델을 종합적으로 활용
- 예
 - 약의 처방, 네비게이터, 검색엔진, 상품 추천
 - 자율차의 운행
 - 알파고와 같은 게임 플레이어,
 - 보험 사기청구 거절 등

데이터 타입

I2

형식	내용
정형 (structured)	<ul style="list-style-type: none">데이터의 포맷이 정해져 있는 데이터서식이 정해진 데이터(엑셀의 표 등)CSV(comma separated value) 파일과 같이 포맷이 일정
비정형 (unstructured)	<ul style="list-style-type: none">미리 정해진 포맷을 가지지 않는 데이터블로그, 트위터 데이터 등 임의의 문장 등으로 구성오디오나 비디오 데이터
반정형 (semi-structured)	<ul style="list-style-type: none">데이터 내부에는 논리적인 형식을 가지고 있으나 외형상으로는 데이터 포맷이 정형 데이터처럼 완전하게 정의되어 있지는 않은 데이터센서 데이터, 웹 사용 기록 등

- 데이터는 네가지 타입이 있다.
 - 문자형: "Hello World", "대한민국", ... (ex) string
 - 수치형: 1, 5, 10, 3.14, 0.9, ... (ex) int, float
 - 바이너리형: 0100100101010101... (ex) array or list
 - 논리형: True, True, False, True, ... (ex) boolean
- 수치형 데이터는 다시 범주형(categorical), 순서형(ordinal), 연속형(continuous)으로 나눌 수 있다.

- 범주형은 클래스를 구분하는 데이터이다.
 - 성별, 국가명, 요일, 사람 이름 등은 범주형 데이터이다.
 - 범주형은 대부분 문자로 표현되지만 편의상 숫자로 대체하여 표현하기도 한다. 예를 들어 월요일=1, 화요일=2, 수요일=3 등
- 순서가 의미를 가지는 데이터를 순서형 데이터라고 한다.
 - 여성의 옷 사이즈를 나타내는 44, 55, 66 같은 숫자, 달력의 1일, 2일, 3일 등이 순서형 데이터이다.
 - 순서형 데이터에서는 덧셈이나 뺄셈이 아무런 의미가 없다.
- 연속형 데이터는 숫자의 양이 어떤 의미를 가지는 데이터
 - 무게, 길이, 온도, 압력, 속도, 화폐 단위
 - 덧셈과 뺄셈의 결과가 계속 같은 연속형 데이터로서 의미를 갖는다.

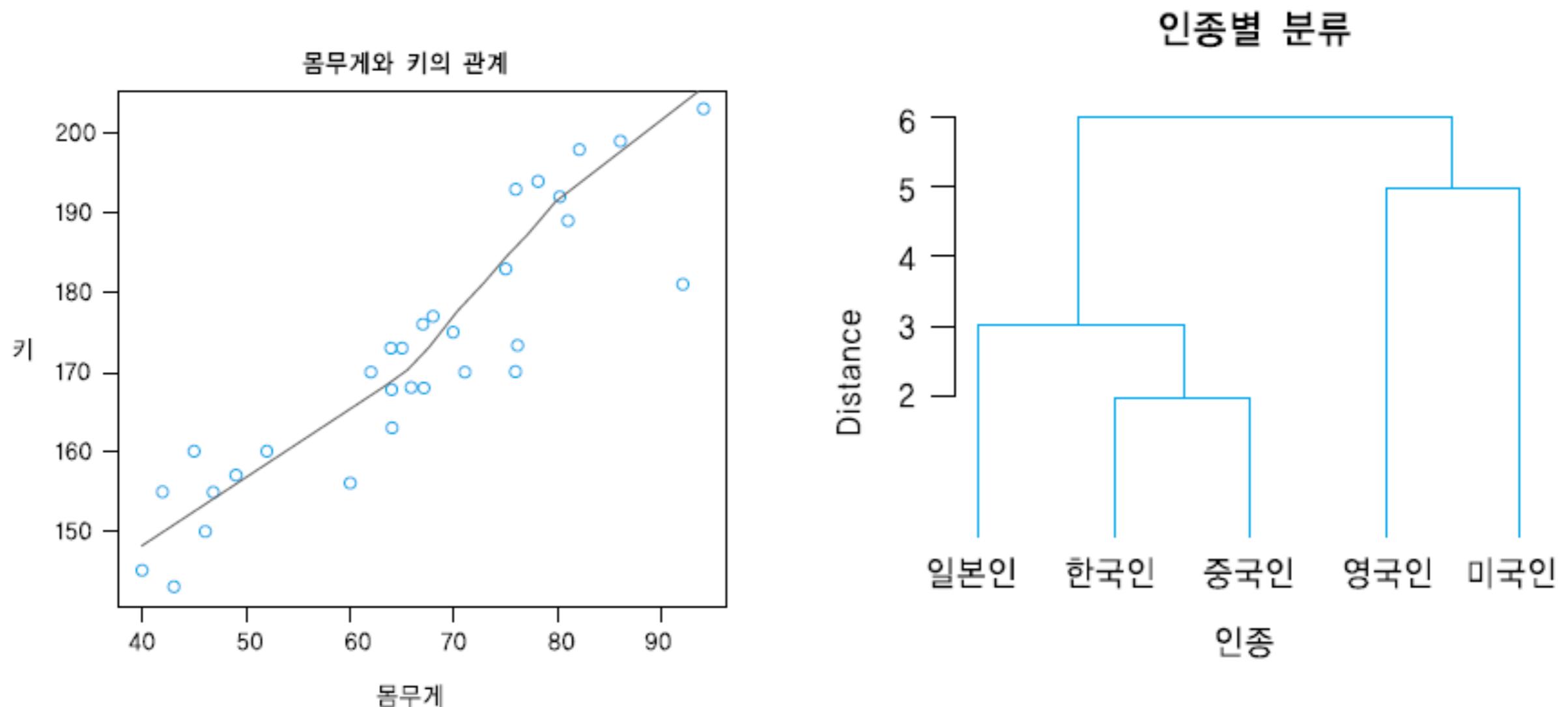
데이터 탐색과 시각화

- 수집한 데이터의 전체적인 특성을 분석
 - Exploratory Data Analysis: EDA
- 본격적인 데이터 분석에 앞서 수집한 데이터가 분석에 적절한지 알아보는 과정
- 기본적인 통계적 특성 파악
 - 숫자형 데이터의 평균, 최대값, 최소값, 표준편차, 분산 등
 - 시각화 도구 이용

- 데이터 시각화(visualization)란 그래프, 도표, 도형 등을 이용하여 데이터의 특징을 파악하게 하는 것
- 숨어 있던 새로운 의미를 찾아낼 수 있음
- 데이터 탐색 뿐만 아니라 분석 결과를 고객에게 설명할 때에 도 필수
- 위치, 길이, 각도, 방향, 형태, 면적, 부피, 명암, 색상 정보를 활용

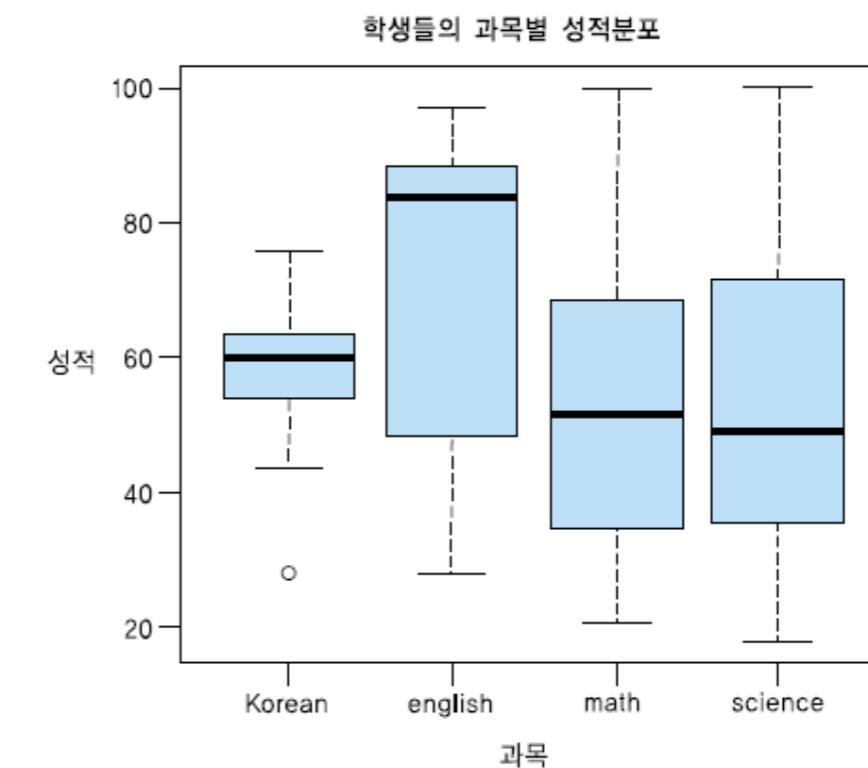
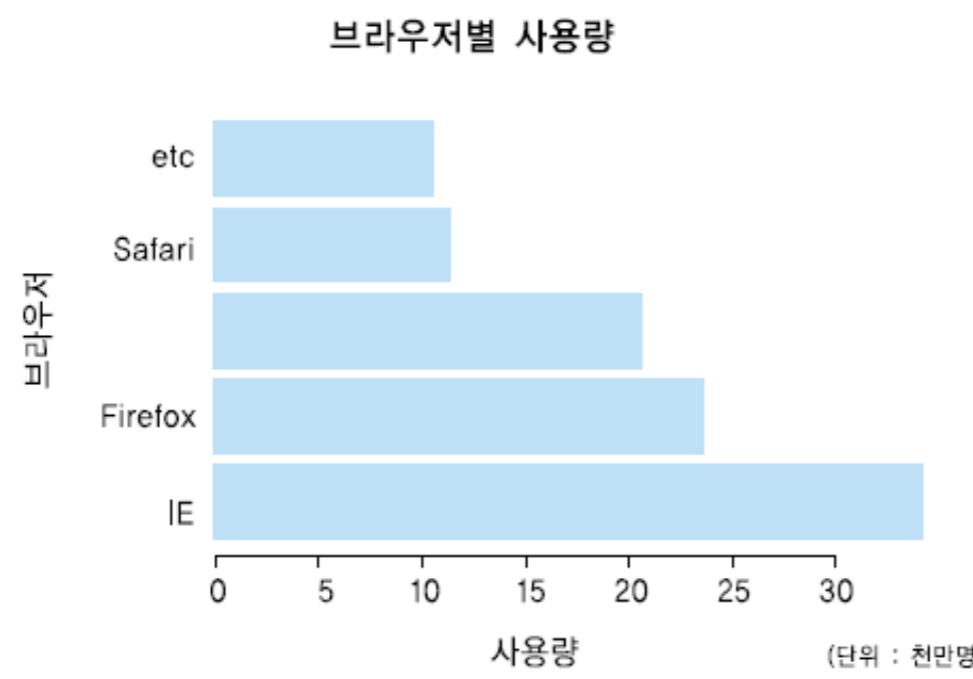
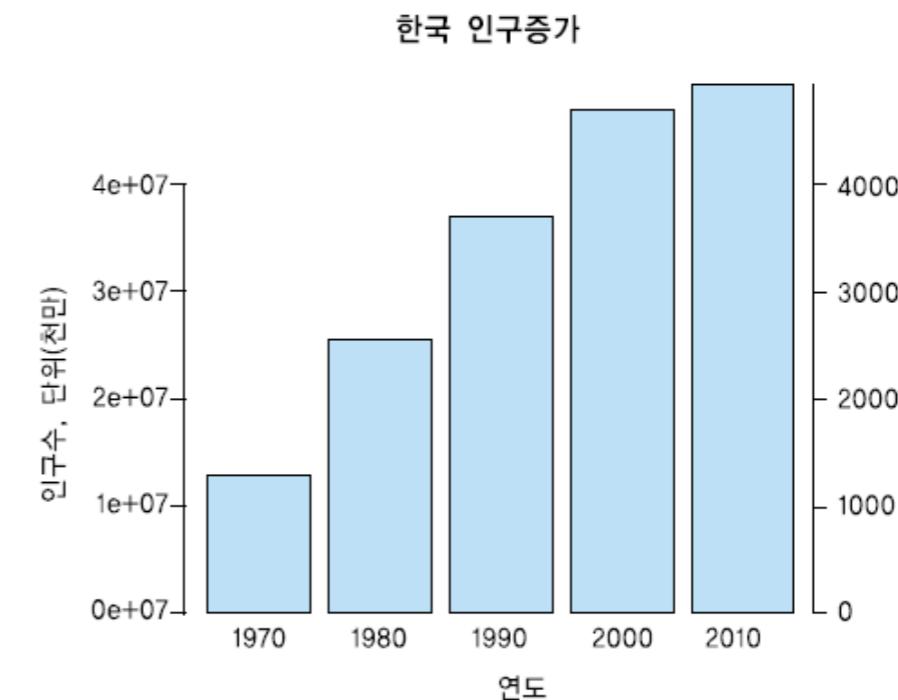
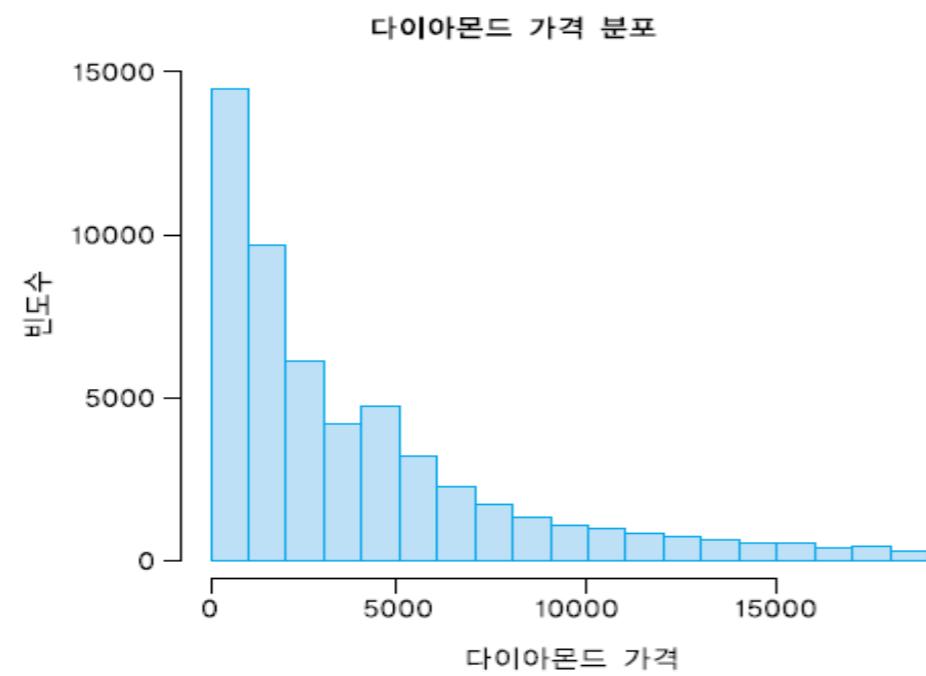
기본적 시각 모형 - 위치

- 산포도(scattering plot), 덴드로그램(dendrogram)

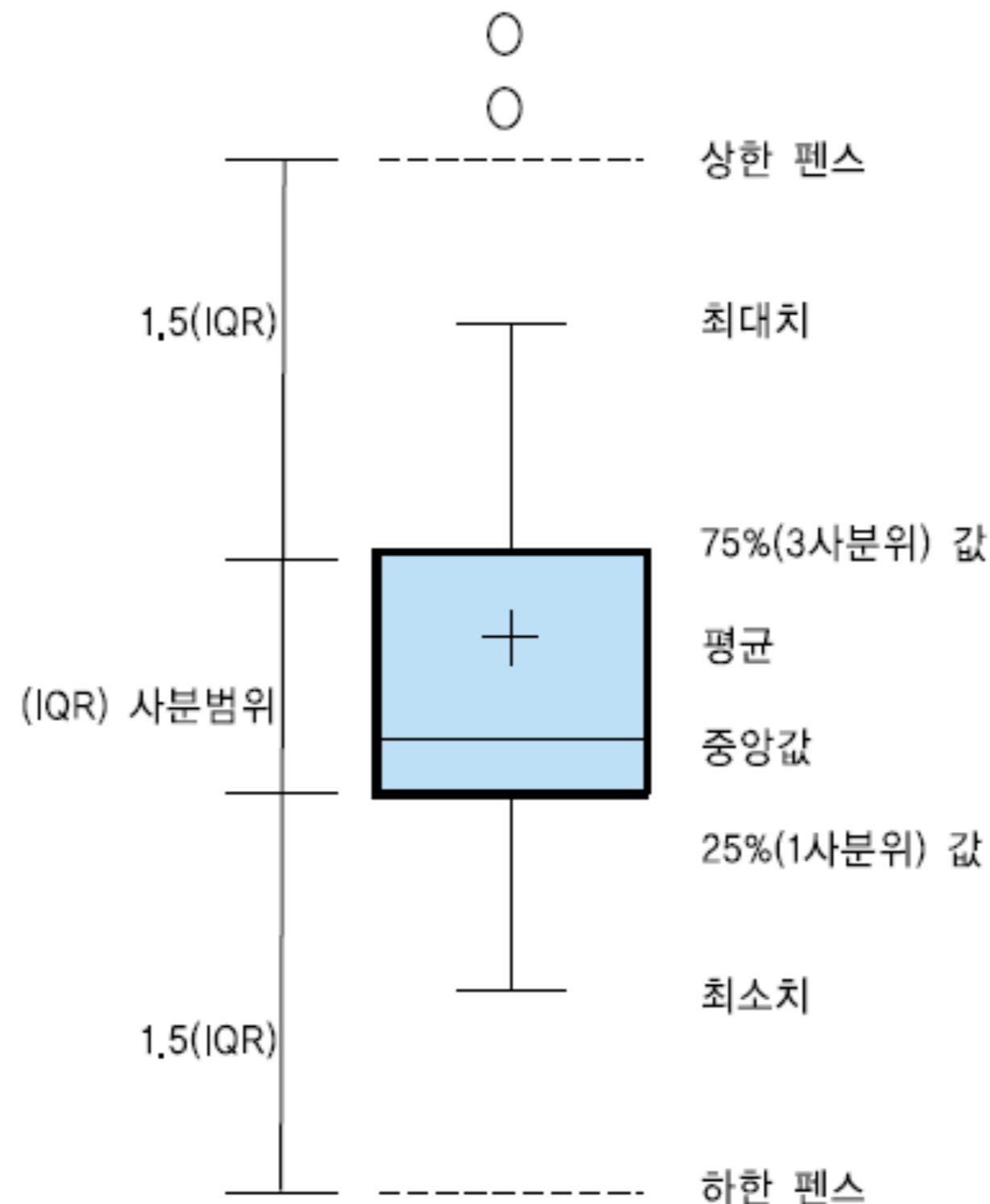


기본적 시각 모형 - 길이

- 히스토그램, 바플롯(막대그래프), 박스플롯

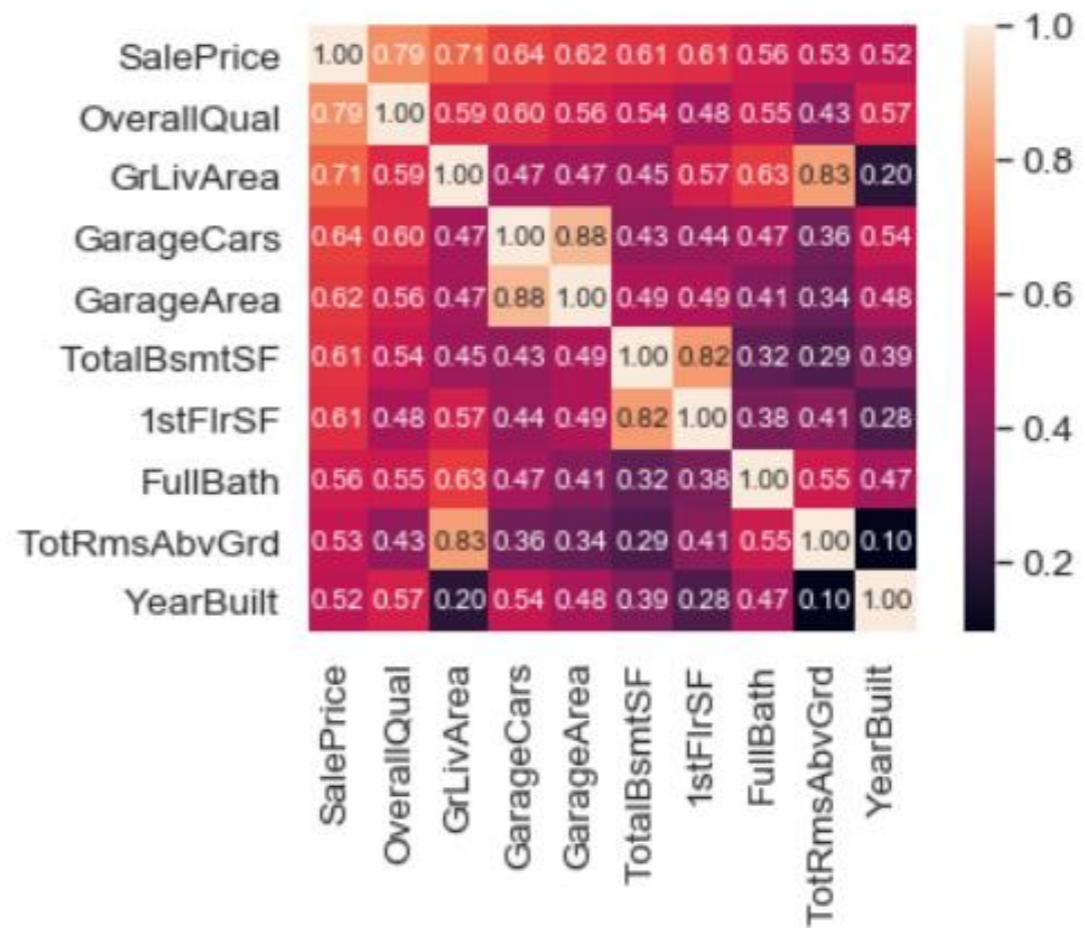
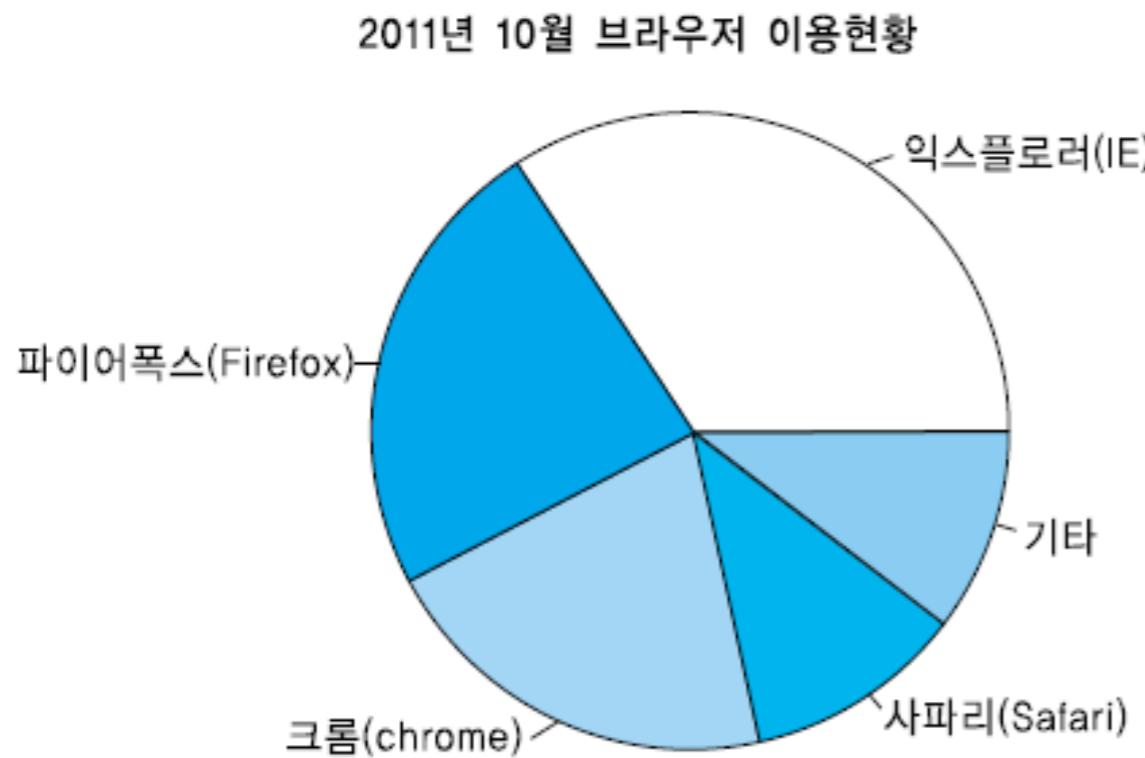


박스 플롯 boxplot()



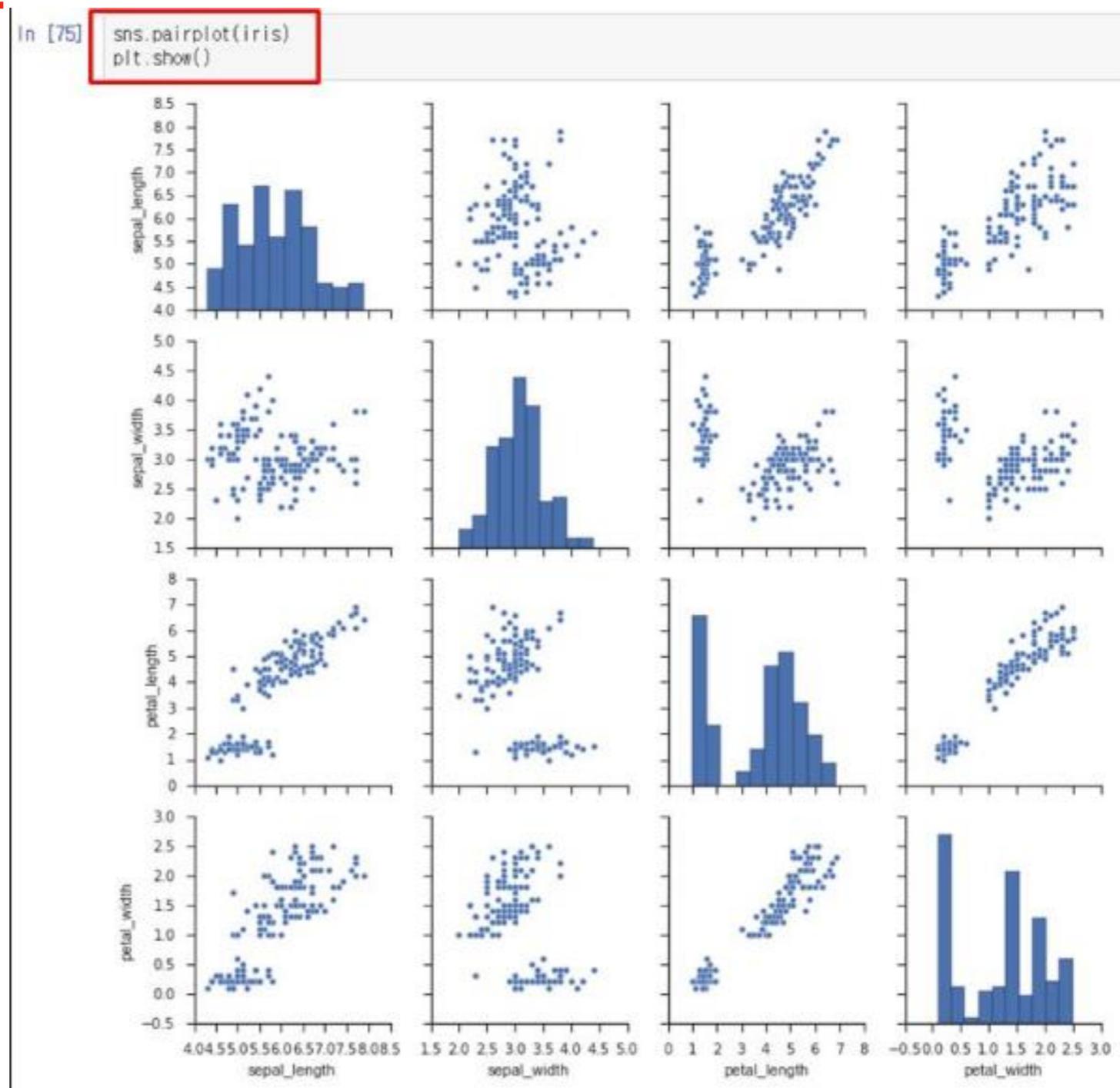
기본적 시각 모형 - 각도, 면적/부피

- Pie Chart(원그래프), Heatmap (히트맵)



기본적 시각 모형 – 각도, 면적/부피

- Pairplot (페어플롯)



Labs

- gg-17 데이터탐색 with 전력사용 데이터
- gg-18 통계적분석-주택가격예측
- gg-19 seaborn-데이터로드
- gg-20 seaborn-환경설정
- gg-21 seaborn-특성
- gg-22 seaborn-바플롯
- gg-23 seaborn-스트립
- gg-24 seaborn-PairGrid

데이터 전처리

데이터 전처리(preprocessing)

25

- 수집한 데이터를 분석하기 좋게 변환하는 모든 작업으로 데이터 정제(Data Cleaning)라고도 함
- 분석 목적에 맞는지 데이터의 품질을 확인하고 필요하면 품질을 높이는 작업
- 데이터 품질
 - 신뢰성
 - 정확성
 - 적시성 (최신성) 등

데이터 전처리 종류

26

구분	처리 방법
결측치 처리 (missing value) 처리	<ul style="list-style-type: none">• 결측치가 포함된 항목을 모두 버리는 방법 (버리는 항목의 비중이 크면 무시하기 어려움)• 결측치를 적절한 값으로 대체 (평균값, 인접 값으로 추정, 0, 최소값, 특정 상수 등)• 분석 단계로 결측치 처리를 넘김(NA로 표기)• 별도의 범주형 변수를 정의하여 추적 가능하게 관리• <code>dataframe.dropna()</code> <code>dataframe.fillna(0)</code> <code>dataframe.fillna(data.mean())</code>
틀린값 처리 (invalid value) 처리	<ul style="list-style-type: none">• 틀린 값이 포함된 항목을 모두 버리는 방법• 틀린 값을 다른 적절한 값으로 대체• 분석 단계로 틀린 값의 처리를 넘김• (예) 키 3.7 미터, 양수가 있어야 할 곳에 음수, 등

데이터 전처리 종류

27

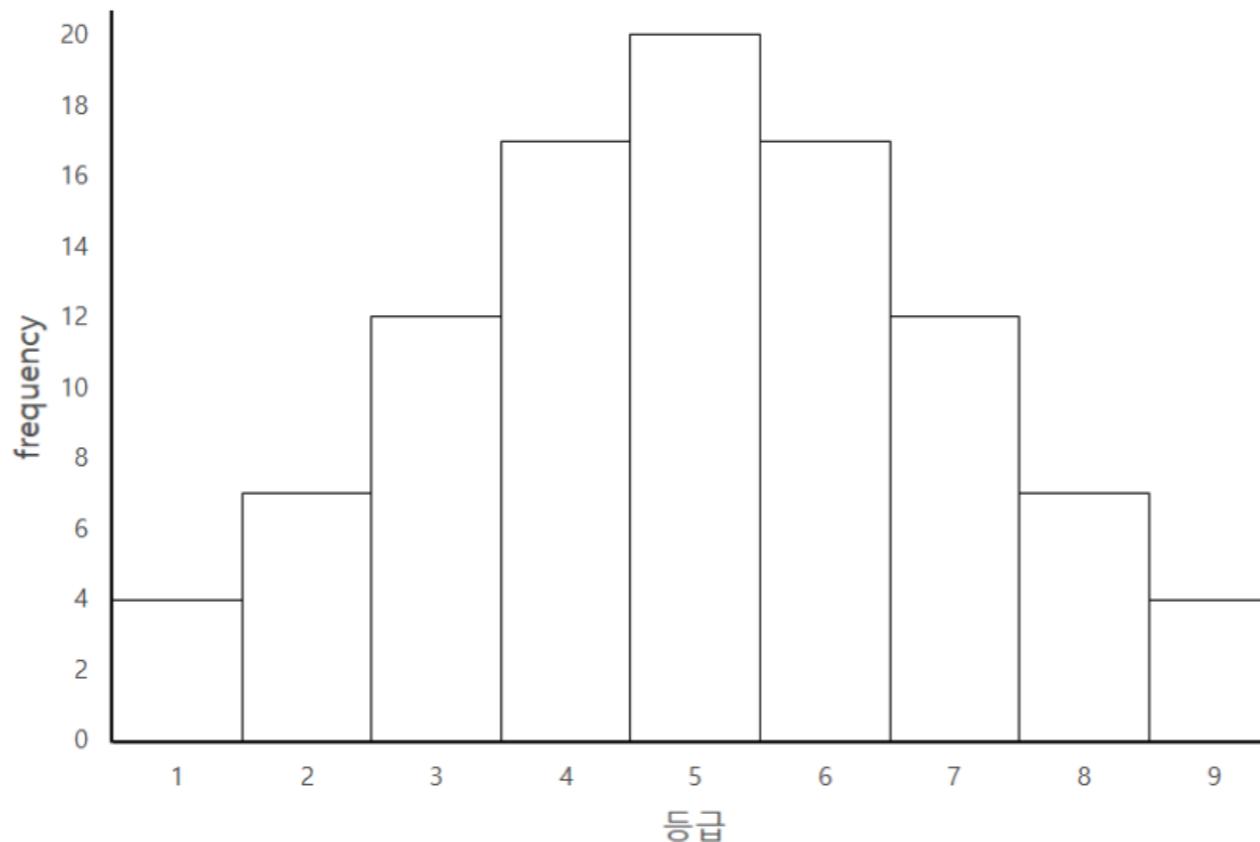
구분	처리 방법
이상치 처리 (outliers)	<ul style="list-style-type: none">값이 일반적인 범위를 벗어나 특별한 값을 갖는 경우데이터 분석 과정의 활동이므로 분석 단계로 넘김도난 카드의 사용, 불법 보험료 청구 등의 탐지(예) 키 2.0 미터 – 극히 드물지만 가능
데이터변환	<ul style="list-style-type: none">범주형 데이터 변환로그변환역수변환스케일링(min-Max Scaling, Standard Scaling, Robust Scaling)

- 데이터를 주어진 그대로 사용하지 않고 다른 형태로 바꾸어 사용하는 것이 필요한 경우가 많다.
 - 같은 성적을 나타내는데 A, B, C 등 학점으로 표현하거나 100점 만점으로 환산하기도 한다 (97, 94, 91 등).

범주형으로 변환

29

- 수치 데이터의 개별 값 구분이 오히려 혼란스러울 때
- 나이 => 10대, 20대, 30대, 40대
- 연간 소득 => 고소득층, 중간층, 저소득층
- 내신 등급 분포 (등급 차이에 대한 느낌이 같도록 정한다)



- 예를 들어 요일을 1, 2, 3, 4, 5, 6, 7 등으로 표시한 경우 이 변수를 컴퓨터가 연산(덧셈이나 곱셈)을 할 수 있는 숫자로 인식해서는 안 된다.
- 이 숫자를 범주형(카테고리형)으로 분명하게 처리되어야 한다. 컴퓨터가 범주형(카테고리형) 변수를 분명히 인식하게 하는 방법이 필요하다
- one hot encoding
 - 하나로 하나의 특성(컬럼)만 1이 될 수 있고 다른 특성은 모두 0으로 코딩하는 방법
 - 판다스가 제공하는 `get_dummies()`를 사용하면 카테고리형 변수들을 원핫인코딩으로 만들어준다

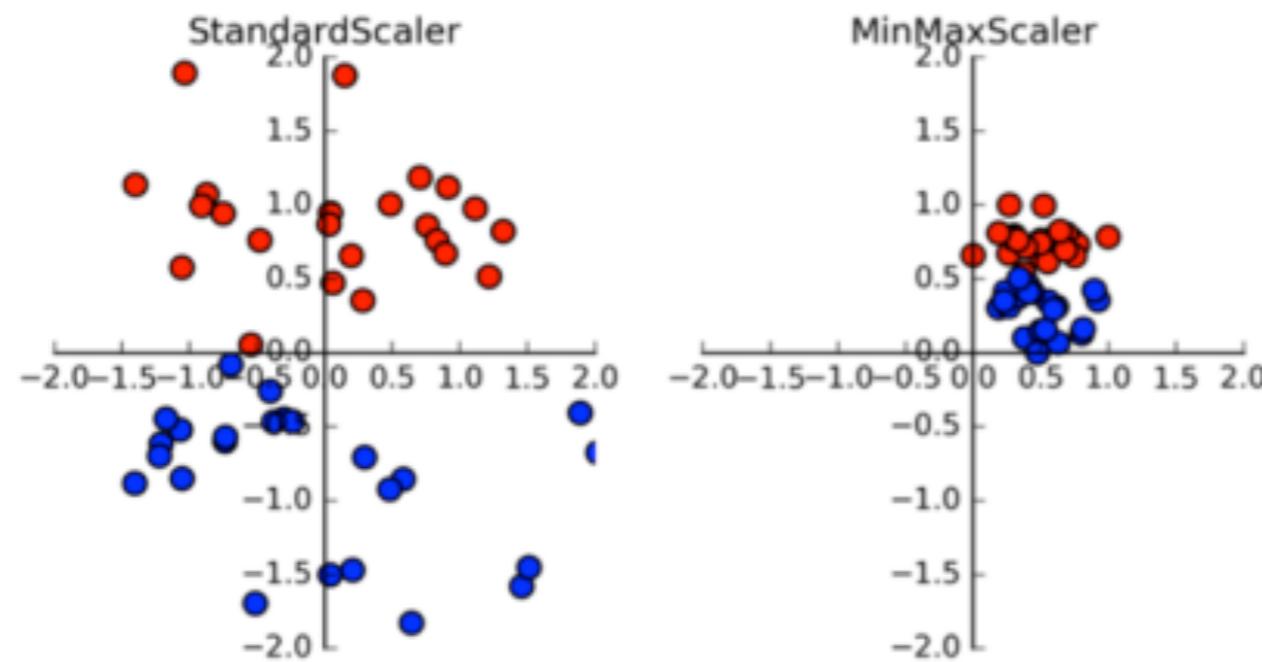
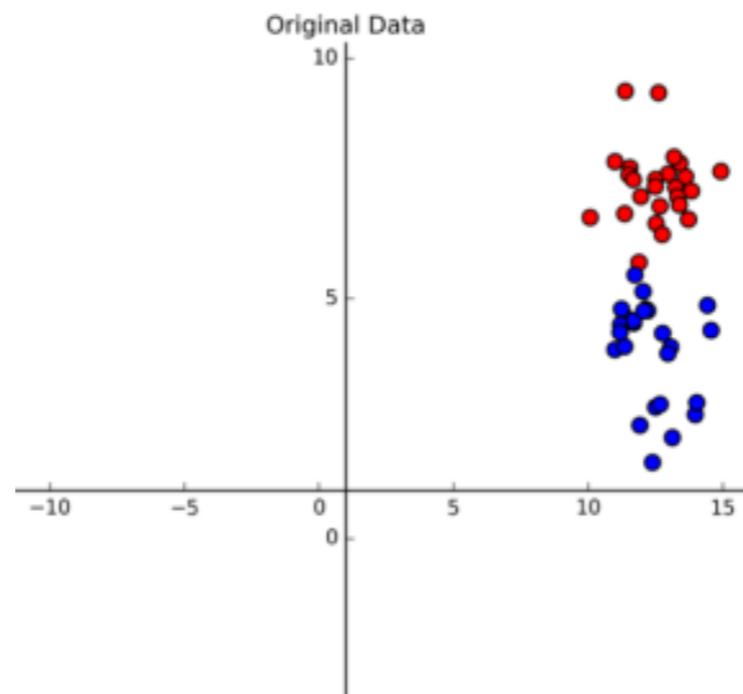
스케일링(Scaling)

31

- 원래 데이터가 갖는 값의 범위를 다르게 조정하는 작업
- 스케일링을 하는 이유는 여러 특성 변수의 중요도를 갖게 맞추기 위해서이다.
- **최소-최대 스케일링 (min-max scaling)**
 - 예를 들어 모든 시험은 100점 만점으로 환산해야 동일한 비중으로 취급되며, 어떤 과목은 50점 만점, 어떤 과목을 80점 만점이면 동일한 조건으로 특성이 반영되지 않는다.
 - 주어진 값의 최소값을 0으로 최대값을 1로 재조정하는 것
 - 파이썬에서는 MinMaxScaler() 함수를 사용
- 표준 스케일링 (standard scaling)
 - 데이터 분포를 평균은 0, 표준 편차는 1이 되도록 정규화 하는 방법
 - 파이썬에서 StandardScaler() 함수 사용
- Robust Scaler
 - IQR(inter-Quartile Range)에 대하여 스케일링 (Outliers에 robust)
 - 파이썬에서는 RobustScaler() 함수 사용

스케일링 비교

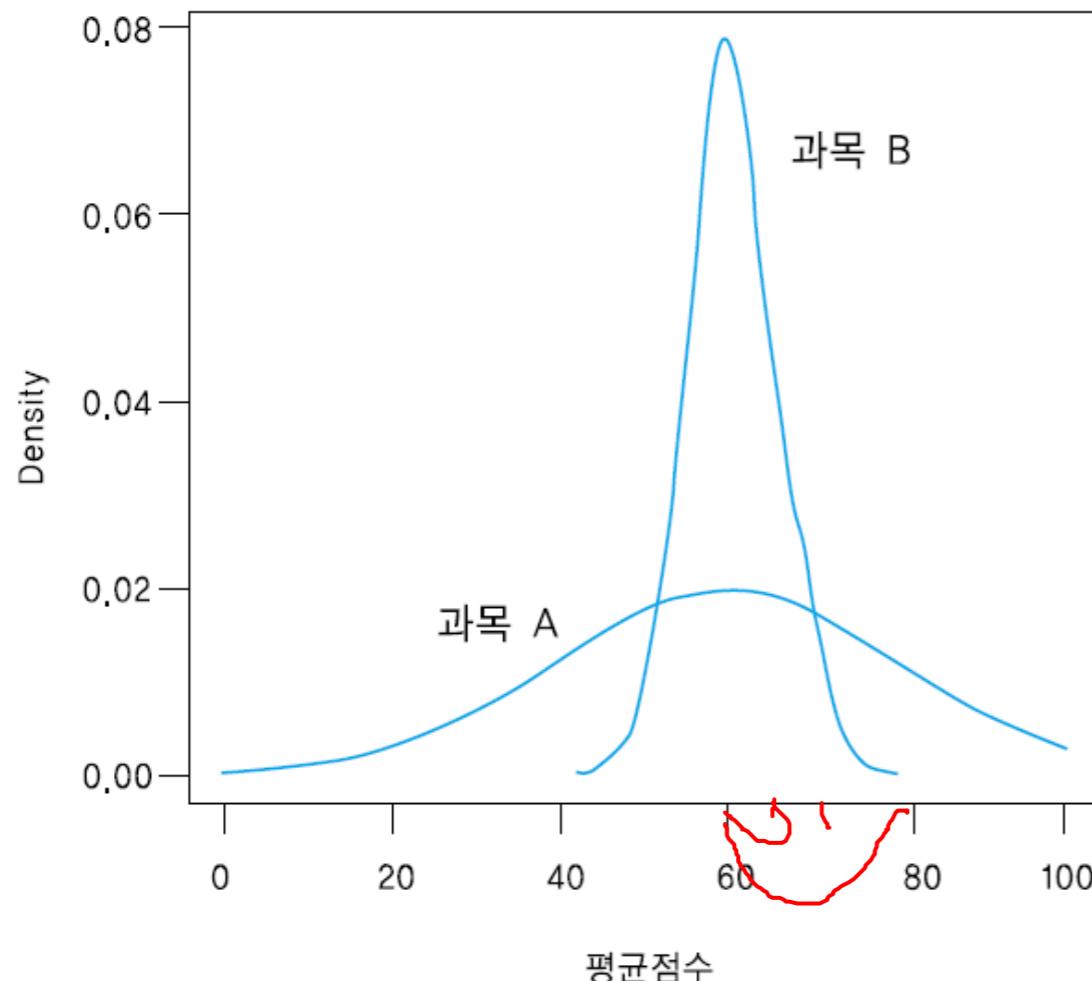
32



표준 스케일링 (표준 정규화)

33

- Who is better?



학생	과목 A	과목 B	평균
갑	90	80	85
을	80	90	85

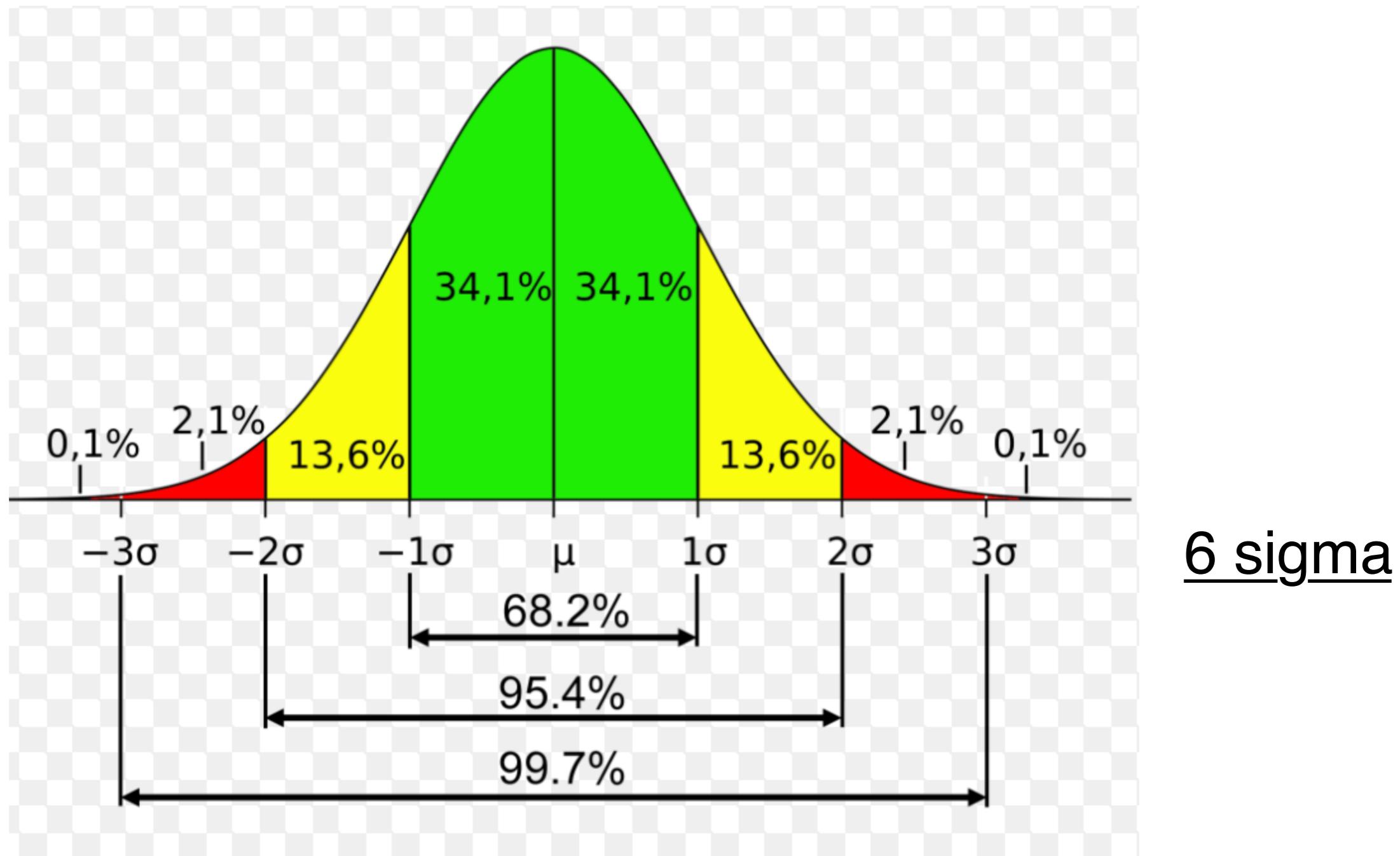
- 표준정규분포(Standard normalization, mean=0, sigma=1)로 변환
- Z-변환(Z-score Transform) 사용

$$z = \frac{x - u}{\sigma}$$

	학생	과목 A	과목 B	평균
변환 전	갑	90	80	85
	을	80	90	85
변환 후	갑	$(90 - 60) / 20$ $= 1.5$	$(80 - 60) / 5$ $= 4$	2.75
	을	$(80 - 60) / 20$ $= 1$	$(90 - 60) / 5$ $= 6$	3.50

정규 분포(Normal Distribution)

35

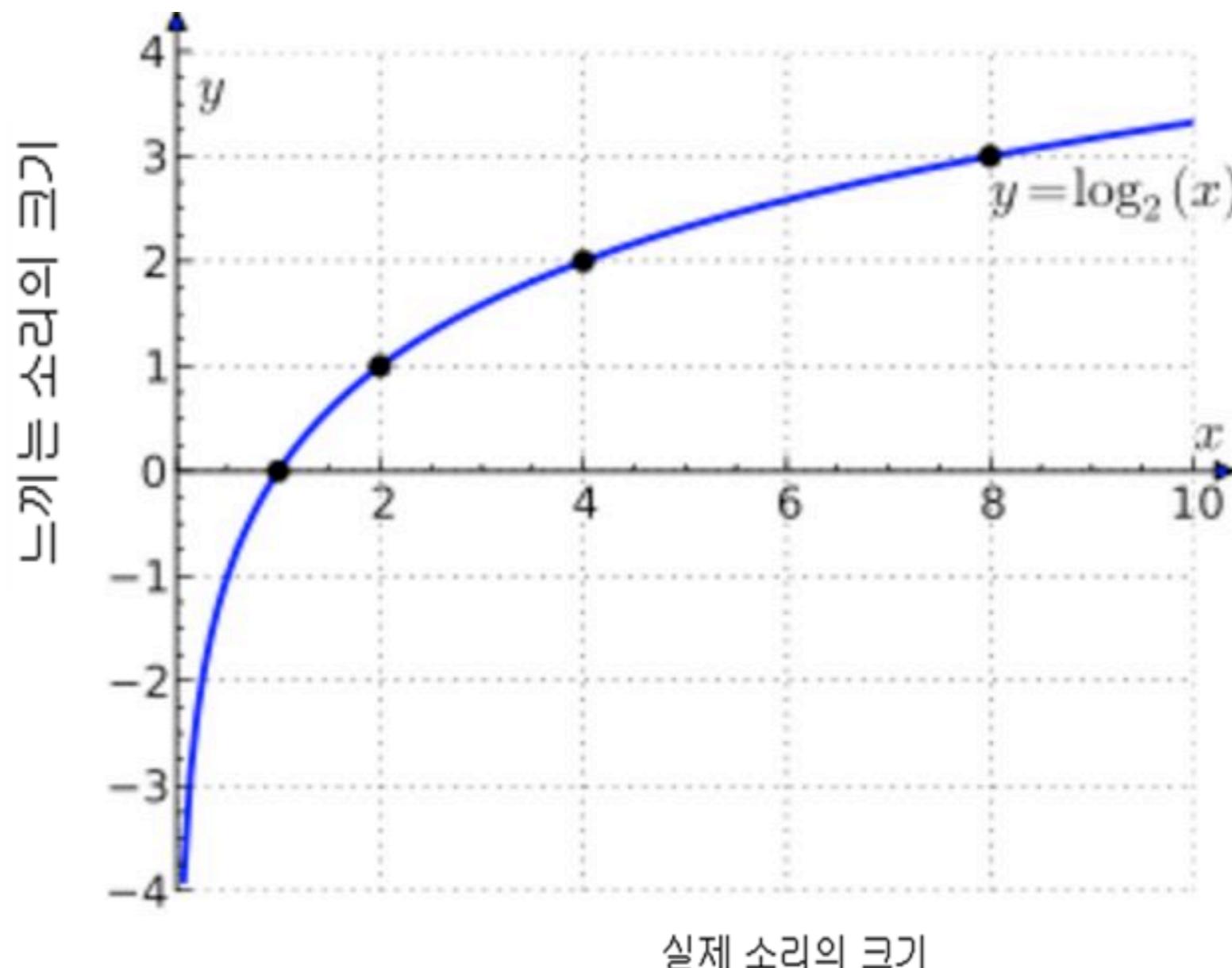


- 로그의 도입
 - 체감형 수치를 선형적으로 표현할 때 사용 - 사람이 자연적으로 느끼는 느낌의 양을 수학적 모델로 설명할 때 사용
 - 돈, 소리, 빛, 압력, 냄새 등 생물학적인 자극을 주는 경우
 - 같은 자극을 느끼려면 현재 보유한 양이 많을수록 이에 비례한 더 강한 자극이 필요하다는 것
 - 이를 수학적으로 표현하면 로그 함수가 됨
 - 현재 보유한 양이 x 이고 이의 변화량, 즉 미분값이 $1/x$ 이 되려면 로그 함수를 얻음
 - 로그를 취한 이후의 값에 대해서 사람들이 변화량을 느끼는 것이 선형적이라는 특성

로그 함수

37

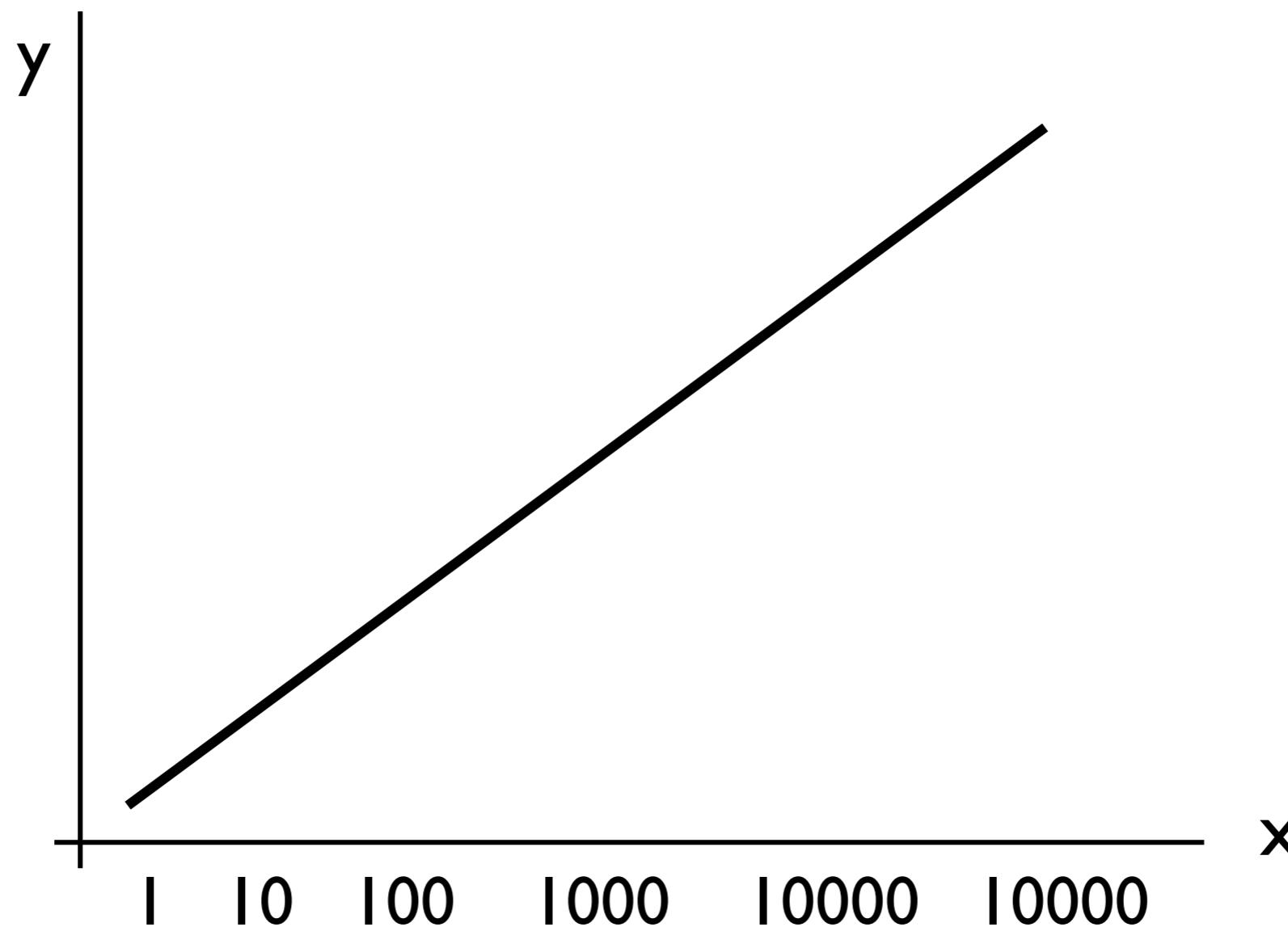
- $\log(x)$ 의 기울기(미분): $1/x$



로그 스케일

38

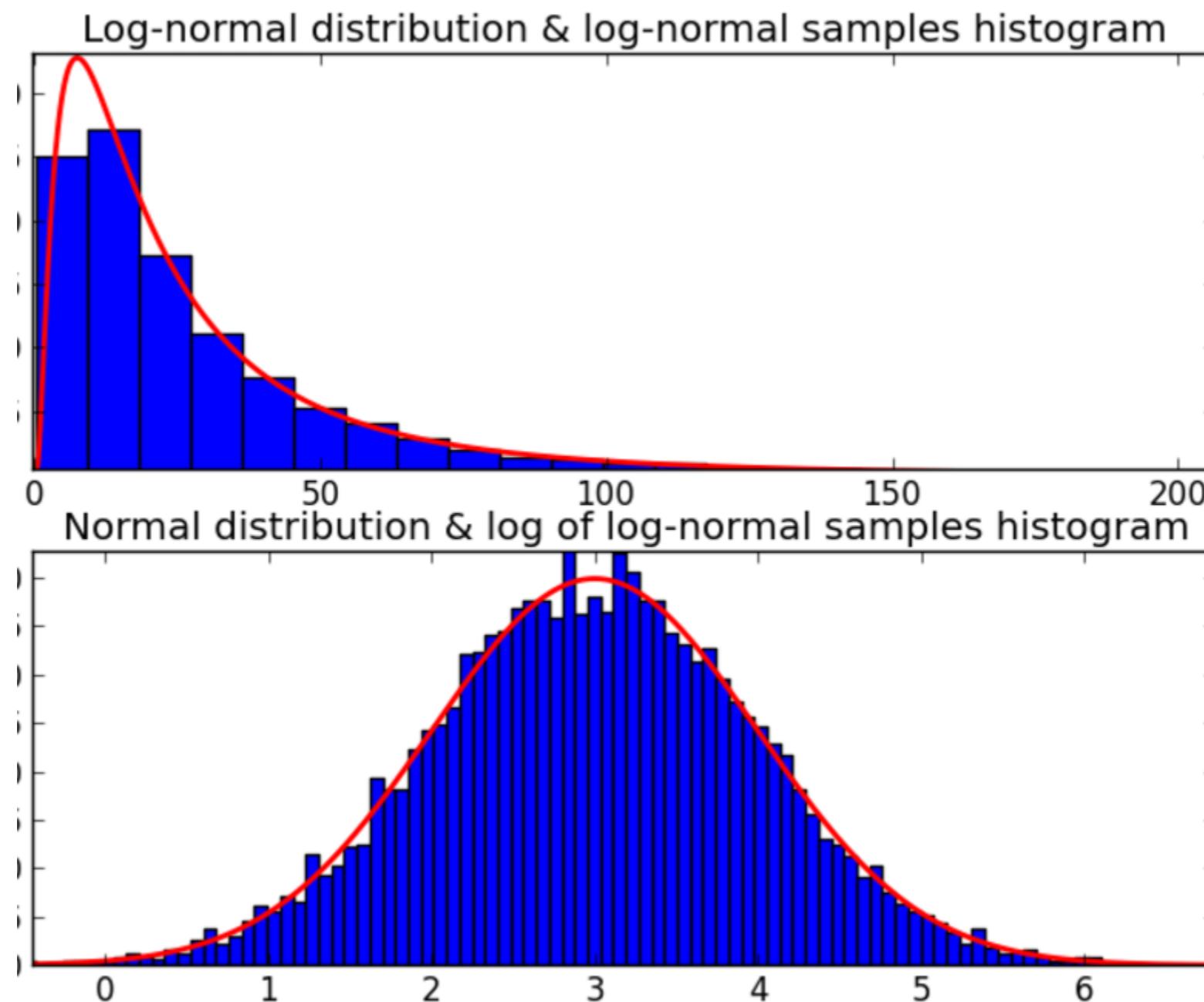
- 로그 스케일 입력에 대해 선형 특성을 갖는 경우



로그-노멀 분포

39

- (ex) 도시의 인구, 재산분포



- 역수를 사용하면 선형적인 특성을 가져 분석의 정확도가 높아지는 경우
- 자동차 마일리지(연료 1L로 가는 거리 Km)와 연비(100km 주행하는데 필요한 연료 L)는 모두 자동차의 성능을 나타내지만 서로 역수의 관계
- 측정 목적:
 - 같은 비용을 얼마나 멀리 갈 수 있는가?
 - 같은 거리를 여행하는데 비용이 얼마가 드는가?

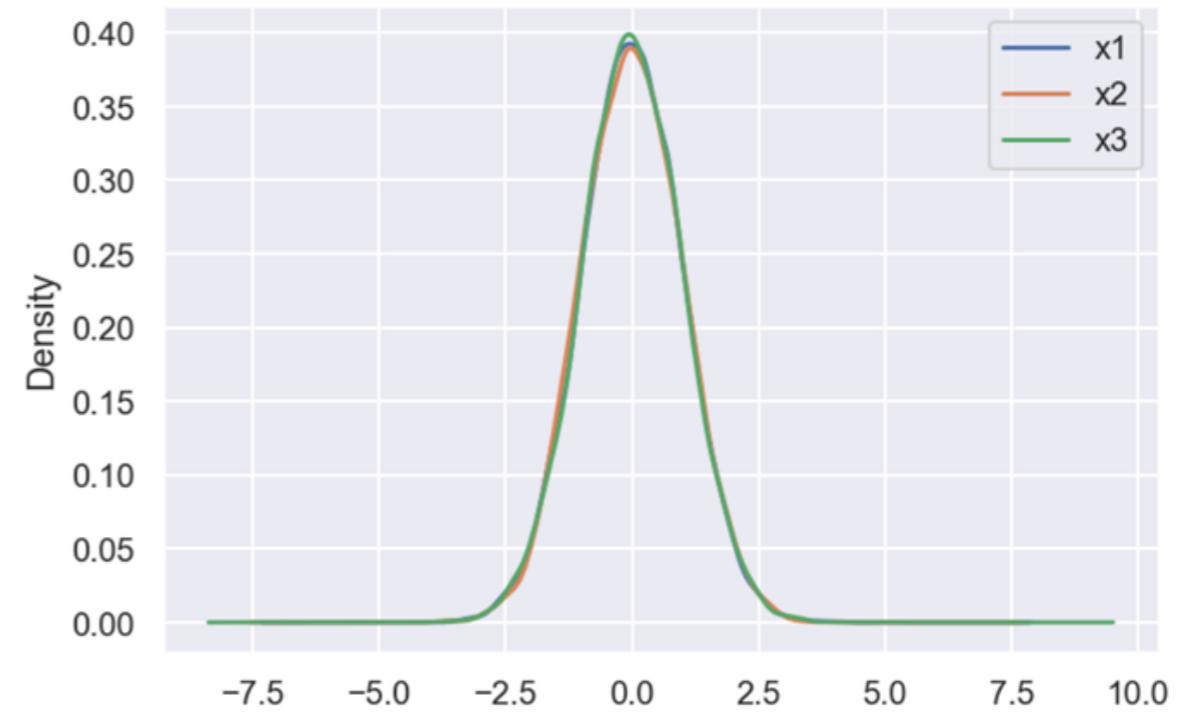
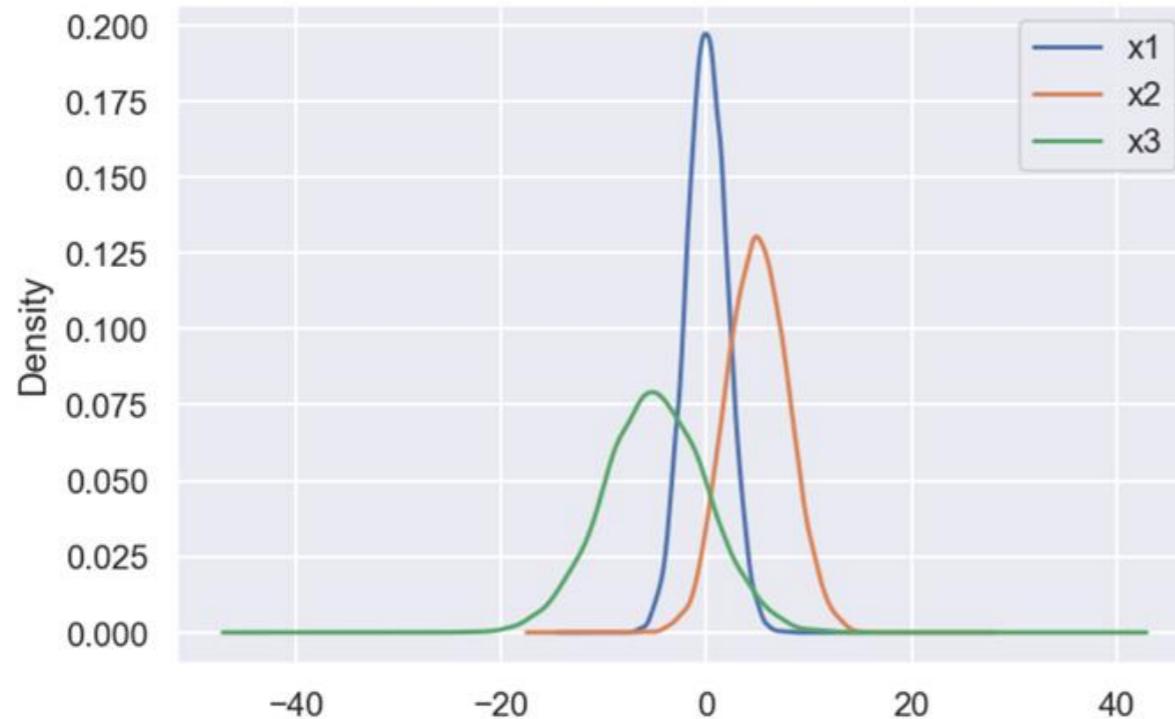
- 선택을 적절히 해야 한다

구분	내용
범주형으로 변환	<ul style="list-style-type: none">수치 데이터가 아닌 것을 명시
Min-Max Scaler	<ul style="list-style-type: none">수치 데이터의 범위가 다를 때
Standard Scaler (z-score 정규화)	<ul style="list-style-type: none">일반 정규화에 표준 편차를 고려한 변환
Robust scaler	<ul style="list-style-type: none">이상치에 강함
로그 변환	<ul style="list-style-type: none">로그를 취하면 선형 특성을 가질 때 (또는 로그 정규 분포를 가질 때)
역수 변환	<ul style="list-style-type: none">역수를 사용하면 선형적인 특성을 가질 때

정규화(Scaling)

42

- 표준정규화(Standard Scaler)

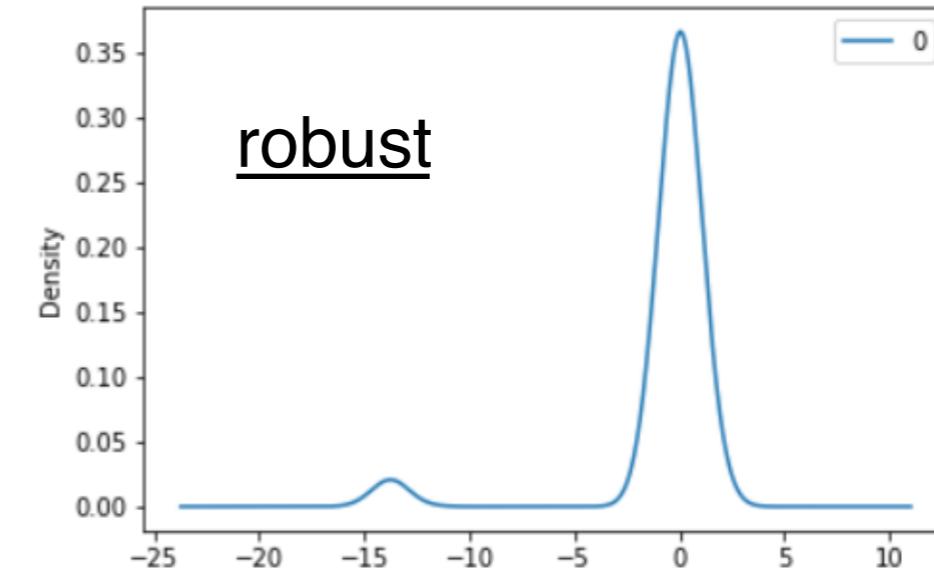
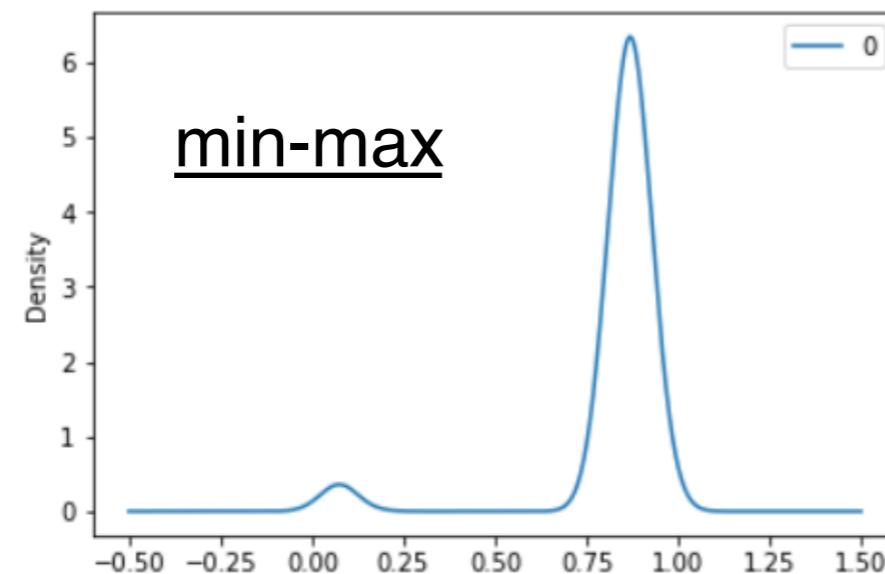
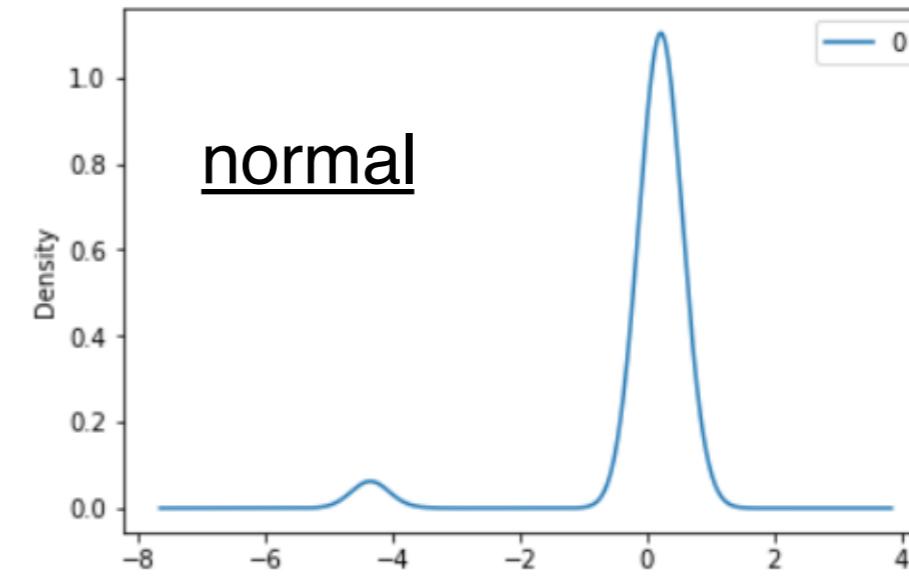
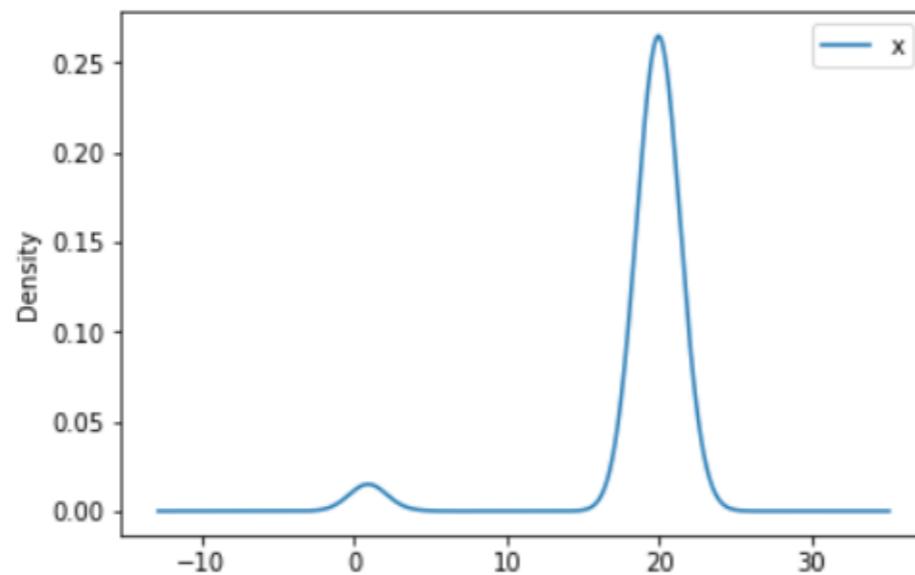


정규화(Scaling)

43

- (예제) Standard, Min-Max, Robust Scaler (data with outliers)

```
df = pd.DataFrame({ 'x' : np.concatenate([np.random.normal(20, 1, 1000),  
                                         np.random.normal(1, 1, 50)])  
})
```



텍스트 분석

- 텍스트 분석의 목적은 텍스트의 의미를 알아내는 것
 - 글의 목적
 - 글쓴이의 성향(찬성/반대)
 - 기분(기쁨/슬픔/우울함 등)
 - 제품 피드백 등
- 텍스트 자체는 대표적인 비정형 데이터
- 의미를 추출하려면 비정형 데이터에서 정형화된 정보를 먼저 얻어야 함
- 텍스트 구문을 분석하여 의미를 파악하고 이것을 정량적으로 측정함

- 트위터, 블로그, 페이스 북 등 SNS의 글을 분석하여 소비자들의 반응, 감성, 트렌드를 파악하거나 개인별 마케팅이나 상품 피드백을 분석하는데 사용된다.
- 이메일, 웹사이트 댓글, 신문기사, 콜센터 상담기록, 도서 등을 분석하여 글의 주요 내용을 파악하거나, 문서의 특징을 추출하거나, 유사한 글이나 저자를 찾는 작업 등을 수행한다.
- 참고문헌이나 본문 인용의 관계를 통해서 문서간의 연계성, 전문가들의 인적 네트워크 등을 파악하는데도 사용한다.
- 인공지능 스피커, 챗봇 등에서도 기본적으로 텍스트 분석이 필요하다

- 텍스트는 대표적인 비정형 데이터이다. 먼저 비정형 데이터인 글자로부터 정형화된 데이터인 수치 데이터를 얻어야 한다.
- 사람이 단어나 문장의 의미를 인식하듯이 컴퓨터가 단어 자체 의미를 직접 파악할 수는 없다
- 컴퓨터가 다루는 텍스트의 단위를 토큰이라고 하고 주어진 텍스트를 토큰으로 나누는 작업을 토큰화(tokenize)라고 한다.

- 데이터 분석에 주어진 전체 문서 집합을 말뭉치(corpus)라고 한다.
- 문서(document)란 코퍼스 내의 한 단위의 텍스트를 말한다. 예를 들어 하나의 블로그는 문서이고 분석할 대상 블로그가 1천개이면 이 1천개 블로그 집합이 말뭉치이다.
- 코퍼스에서 의미 있는 단어를 추출하는 작업을 파싱(parsing)이라고 한다.

- 먼저 토큰화의 단위를 단어(word)로 할지 아니면 글자(character) 단위로 할지를 정해야 한다.
- 단어를 어근(stem)으로 변환하면 어미 변화를 무시하거나 조사를 무시하게 되어 텍스트에 들어 있던 정보를 잃게 된다.
- 글자 단위로 토큰화를 하면 어근으로 변환할 때 정보를 잃는 문제를 피할 수 있다.
- 영어는 알파벳이 26글자이므로 음절단위의 토큰의 수가 매우 적다. 한글은 음절단위로 나누면 음절의 수가 수천 가지가 될 것이다.

- 토큰화 단위에는 크게 세 가지가 있다.
 - 단어(word) 단위
 - 글자(character) 단위
 - n-gram 단위
- n-gram이란 n개의 연속된 단어를 하나로 취급하는 방법이다.
- 예를 들어 “러시아 월드컵”이라는 표현을 “러시아”와 “월드컵” 두 개의 독립된 단어로만 취급하지 않고 두 단어로 구성된 하나의 토큰으로 취급한다.
 - $n=2$ 경우를 bi-gram이라고도 부른다
 - 단어의 갯수가 늘어난 효과를 얻는다

토큰화 – (n-gram)

51

텍스트: “어제 러시아에 갔다가 러시아 월드컵을 관람했다”

단어토큰: {"어제", "러시아", "갔다", "러시아", "월드컵", "관람"}

2-gram 토큰: {"어제 러시아", "러시아 갔다", "갔다 러시아", "러시아 월드컵", "월드컵 관람"}

- n-gram을 허용하면 토큰화 대상의 수가 매우 크게 증가한다. 이론적으로는 10만개의 단어를 두 개 붙여서 나올 수 있는 경우의 수는 10만의 자승이 된다.
- 실제로는 빈도수가 최소한 몇 개 이상인 것만을 다룬다.
- 토큰화 한 결과를 수치로 만드는 방법
 - 원핫(one-hot) 엔코딩
 - BOW(단어모음)
 - 단어벡터(Word Vector) 방법

원핫(one-hot) 엔코딩

53

- 토큰에 고유 번호를 배정하고 모든 고유번호 위치의 한 컬럼만 1, 나머지 컬럼은 0인 벡터로 표시하는 방법

텍스트: “어제 러시아에 갔다가 러시아 월드컵을 관람했다”

토큰 사전: {"어제":0, "러시아":1, "갔다":2, "월드컵":3, "관람":4}

원핫 코딩:

어제 = [1, 0, 0, 0, 0]

러시아 = [0, 1, 0, 0, 0]

갔다 = [0, 0, 1, 0, 0]

월드컵 = [0, 0, 0, 1, 0]

관람 = [0, 0, 0, 0, 1]

BOW(Bag of Word, 단어모음)

54

- 원핫 코딩 방식으로 단어(토큰)을 표현하면, 단어의 수가 적을 때에는 문제가 안되지만 예를 들어 단어가 모두 10만개이면 모든 단어가 항목이 10만개인 (0과 1로 구성된) 벡터로 표시된다.
- 만일 주어진 텍스트가 20개의 단어로 구성되어 있다면, $20 \times 100,000$ 개 크기의 벡터가 필요하다.
- 텍스트 분석은 “문장”을 단위로 하는 경우가 많으므로 한 문장을 하나의 벡터로 만드는 방법이 단어모음(BOW) 방식이다.
 - 한 문장을 단어 사전 크기의 벡터로 표현하고 그 문장에 들어 있는 단어의 컬럼만 1로, 단어가 없는 컬럼은 모두 0으로 표현한다.
- 먼저 단어 사전을 만들고 각 문장에 어떤 단어가 들어 있는지 조사하여 해당 컬럼만 1로, 나머지는 0으로 코딩한다

- 단어 사전: {"어제":0, "오늘":1, "미국":2, "러시아":3, "갔다":4, "축구":5, "월드컵":6, "올림픽":7, "관람":8, "나는":9, ..., "중국":4999 }
- Text_1: “어제 러시아에 갔다가 러시아 월드컵을 관람했다” 를 BOW로 표현하면

문장번호	0	1	2	3	4	5	6	7	8	9	10	...	4998	4999
Text_1	1	0	0	1	1	0	1	0	1	0	0	0	0	0
Text_2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Text_3	0	0	0	1	0	0	0	0	0	0	0	0	0	0
Text_4	0	0	0	0	0	0	0	1	0	0	0	0	1	0
...														
Text_50	0	0	0	0	0	0	1	0	0	0	0	0	0	0

문서-단어 행렬 (document-term matrix) 56

- 앞에서는 문장 단위로 어떤 단어들이 있는지를 BOW로 만드는 방법을 소개했다. 이를 확장하여 문서(document) 단위로 어떤 단어들이 있는지를 표현하는 것을 문서-단어 (document-term) 행렬이라고 한다.
 - 같은 단어가 여러 번 등장하면 1 이상의 값을 갖는다

문서번호	0	1	2	3	4	5	6	7	8	9	10	...	4998	4999
Doc_1	1	2	3	1	4	0	2	0	1	3	0	0	0	0
Doc_2	0	0	0	0	2	0	0	0	0	0	0	0	2	0
Doc_3	0	0	0	1	0	0	0	0	3	0	0	0	0	1
Doc_4	4	0	0	0	0	0	0	1	0	0	4	0	1	0
...														
Doc_100	0	2	0	0	0	0	0	1	4	0	1	0	0	0

- TfIdf(Term Frequency-Inverse Document Frequency)
- Tf(term frequency)란 단어가 각 문서에서 발생한 빈도이다
- 그 단어가 등장한 '문서'의 빈도를 document frequency (df)라고 한다
- 적은 문서에서 발견될수록 가치 있는 정보라고 할 수 있다
- 많은 문서에 등장하는 단어일수록 일반적인 단어이며 이러한 공통적인 단어는 tf가 크다고 하여도 비중을 낮추어야 분석이 제대로 이루어질 수 있다.
- 따라서 단어가 특정 문서에만 나타나는 희소성을 반영하기 위해서 idf(df의 역수)를 tf에 곱한 값을 tf 대신 사용한다

tf-idf (example)

58

- From http://www.datasciencecourse.org/notes/free_text/
 - Doc1 = “The goal of this lecture is to explain the basics of free text processing”
 - Doc2 = “The bag of words model is one such approach”
 - Doc3 = “Text processing via bag of words”

$$X = \begin{bmatrix} \text{the} & \text{is} & \text{of} & \text{goal} & \text{lecture} & \text{bag} & \text{words} & \text{via} & \text{text} & \text{approach} \\ 2 & 1 & 2 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & \dots \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \end{bmatrix} \begin{array}{l} \text{Document 1} \\ \text{Document 2} \\ \text{Document 3} \end{array}$$

tf-idf (example)

59

- Term frequency
 - Counts of each word in a document
 - $tf_{i,j}$ = frequency of word j in document i
- Inverse document frequency
 - Term frequencies tend to be “overloaded” with very common words (“the”, “is”, “of”, etc)
 - Idea if inverse document frequency weight words negatively in proportion to how often they occur in the entire set of documents

$$idf_j = \log \left(\frac{\# \text{ documents}}{\# \text{ documents with word } j} \right)$$

tf-idf (example)

60

$$X = \begin{bmatrix} \text{the} & \text{is} & \text{of} & \text{goal} & \text{lecture} & \text{bag} & \text{words} & \text{via} & \text{text} & \text{approach} \\ 2 & 1 & 2 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \end{bmatrix} \begin{array}{l} \text{Document 1} \\ \text{Document 2} \\ \text{Document 3} \end{array}$$

$$\text{idf}_{\text{of}} = \log \left(\frac{3}{3} \right) = 0$$

$$\text{idf}_{\text{is}} = \log \left(\frac{3}{2} \right) = 0.405$$

$$\text{idf}_{\text{goal}} = \log \left(\frac{3}{1} \right) = 1.098$$

tf-idf (example)

61

- Term frequency inverse document frequency = $\text{tf}_{ij} \cdot \text{idf}_j$
- Just replace the entries in the X matrix with their TFIDF score.

$$X = \begin{bmatrix} \text{the} & \text{is} & \text{of} & \text{goal} \\ 0.8 & 0.4 & 0 & 1.1 \\ 0.4 & 0.4 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Labs

- gg-25 범주형데이터코딩
- gg-26 데이터변환

머신러닝 개요

- 인공지능 구현하는 방법은 다양
- 머신 러닝 기반의 AI가 2000년대 이후 급속히 발전
- 딥러닝: 신경망을 기반으로 하는 머신러닝 기술
 - 마치 사람이 많은 정보에 접하면서 학습하듯이 컴퓨터도 데이터를 보고 학습하는 방법
 - 음성인식, 자동차 번호판 인식, 언어 번역, 채팅 대화, 글쓰기, 작곡 등 여러 분야에서 좋은 성과를 낸다

- 머신러닝:
 - 정답과 오답을 계속 가르쳐 주면 모델의 정확도가 높아짐
 - 많은 데이터를 사용하여 컴퓨터 성능이 점차 발전(학습)하는 것

- 예전에는 컴퓨터는 프로그래머가 코딩한 대로만 동작
 - 계산을 빨리 하든지,
 - 이미지를 처리하든지,
 - 정해진 알고리즘대로 빠르고 정확하게 동작하는 일
- 머신러닝에서는 컴퓨터가 데이터를 보면서 점차 성능을 향상시킨다.
- 컴퓨터가 데이터를 보고 스스로 기능을 향상시키는 방법을 찾아낸 것

- 머신러닝을 사용하는 목적 즉, 머신러닝으로 문제를 해결하는 유형
 - 설명(description)
 - ▶ 클러스터링 (Clustering, 군집)
 - ▶ 비지도 학습 (Unsupervised Learning) – based on input only
 - 예측
 - ▶ 회귀(regression)
 - ▶ 분류(classification)
 - ▶ 지도학습 (Supervised Learning) – based on input/output
 - 추천(recommendation)
 - 연관분석
 - 강화학습

- 지도학습은 정답을 예측하는데 사용된다.
- 정답은 목적(target) 변수, 레이블이라고도 한다
- 예측은 분류와 회귀로 나누어진다.
- 분류
 - **분류(classification)**란 어떤 항목(item)이 어느 그룹에 속하는지를 판별하는 기능을 말한다.
 - 두 가지 카테고리를 나누는 작업을 이진 분류(binary classification)라고 하고 세 개 이상의 클래스를 나누는 작업을 다중 분류(multiclass classification)라고 한다.
- 회귀
 - 수치를 예측하는 것을 **회귀(Regression)**라고 한다.

- 비지도 학습이란 정답이 없이 데이터로부터 중요한 의미를 찾아내는 머신러닝 기법이다.
 - **군집화**: 유사항 항목들을 같은 그룹으로 묶는다.
 - 데이터 변환: 데이터를 분석하기 좋게 다른 형태로 변환한다
 - **주성분분석(PCA)**: 머신러닝에 사용할 특성의 수를 줄인다.
 - **차원축소**: 데이터의 속성을 명확하게 시각화하기 위해서 고차원의 특성 값을 2차원이나 3차원으로 차원을 축소하는 작업

- 어떤 사건이 다른 사건과 얼마나 자주 동시에 발생하는지 파악
- 자주 발생하는 패턴 찾기(상품의 연관성, 취향의 연관성 등 분석)
- 같이 구매한 상품 분석(market basket analysis, 장바구니 분석)
- 상품의 진열 배치 및 상품 프로모션(쿠폰 발행 등)에 활용

- 강화학습(reinforcement learning)은 머신러닝 모델이 어느 방향으로 만들어져야 하는지 방향성만 알려주는 학습 방법
 - 입력 샘플마다 정답을 있어 답을 알려주는 것이 아니지만 시간이 흐르면서 모델이 바람직한 방향으로 가고 있는지를 알려줄 수 있고 이를 통해서 학습하는 방법
- 강화학습에서는 일정 기간 동안의 행동(action)에 대해 보상(reward)을 해줌으로써 잘 하고 있는지, 잘 못하고 있는지 를 알려주며 학습을 시킨다.
 - 예를 들어 로봇이 혼자 그네를 타는 방법, 전자 게임을 하는 방법, 바둑을 두는 방법의 학습에 사용된다.
 - 2017년에 우리나라 이세돌을 이긴 알파고(Alpha Go) 바둑 프로그램

머신러닝 알고리즘

72

	머신러닝 유형	알고리즘
지도학습	분류	kNN, 베이즈, 결정 트리, 랜덤포레스트, 로지스틱회귀, 그라디언트부스팅, 신경망
	회귀	선형회귀분석, SVM, 신경망
비지도학습	<u>군집화</u>	K-Means, DBSCAN
	데이터 변환	스케일링, 정규화, 로그변환
	<u>차원축소</u>	PCA, 시각화

- 해결할 문제에 적합한 머신러닝 모델 선택
 - 선형모델, 결정트리, 신경망, SVM, 랜덤포레스트 등
- 훈련 데이터로 모델을 학습
- 모델이 과대적합 (Over fitting)되었는지 또는 과소적합 (Under fitting) 인지를 검증
 - 과대적합이면 모델을 더 일반화 해야 하고, 과소적합이면 모델을 더 상세하게 설계해야 한다.
- 모델을 실제 테스트 데이터에 적용하고 성능을 평가

머신러닝 동작

- 머신러닝은 모델(model)을 사용한다
 - 스팸 메일을 찾아내는 모델,
 - 누가 게임에서 이길지 예측하는 모델,
 - 내일 날씨를 예측하는 모델
- 과학에서는 어떤 현상을 설명하는 모델로 수식을 주로 사용
 - 모든 질량을 가진 모든 물체는 서로 끌어당긴다는 만유인력 법칙은 두 물체의 질량에 각각 비례하고, 두 물체의 거리의 자승에 반비례하는 수식으로 표현된
- 머신러닝, AI 모델은 데이터 기반의 모델을 사용한다

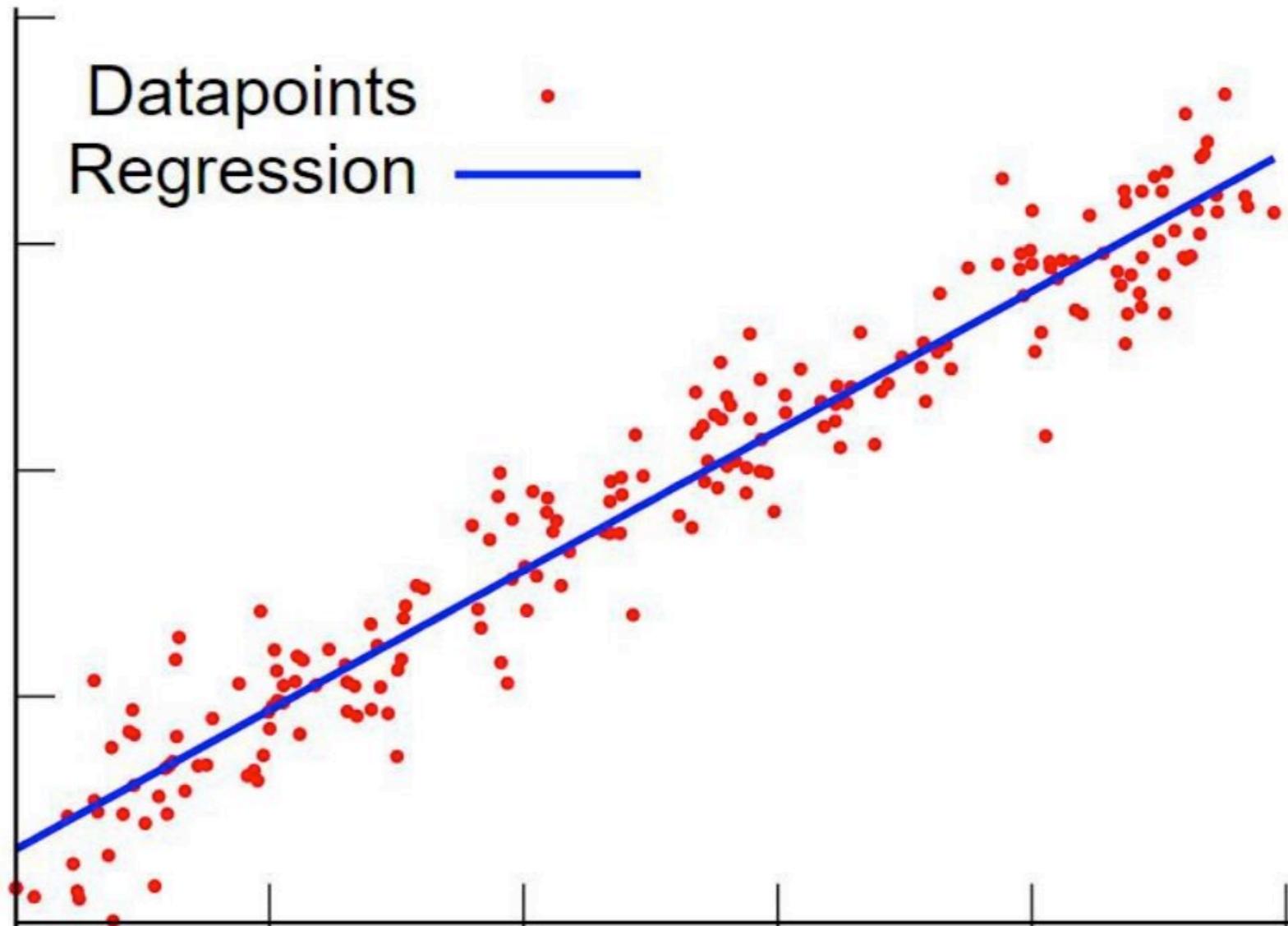
모델의 가치

76

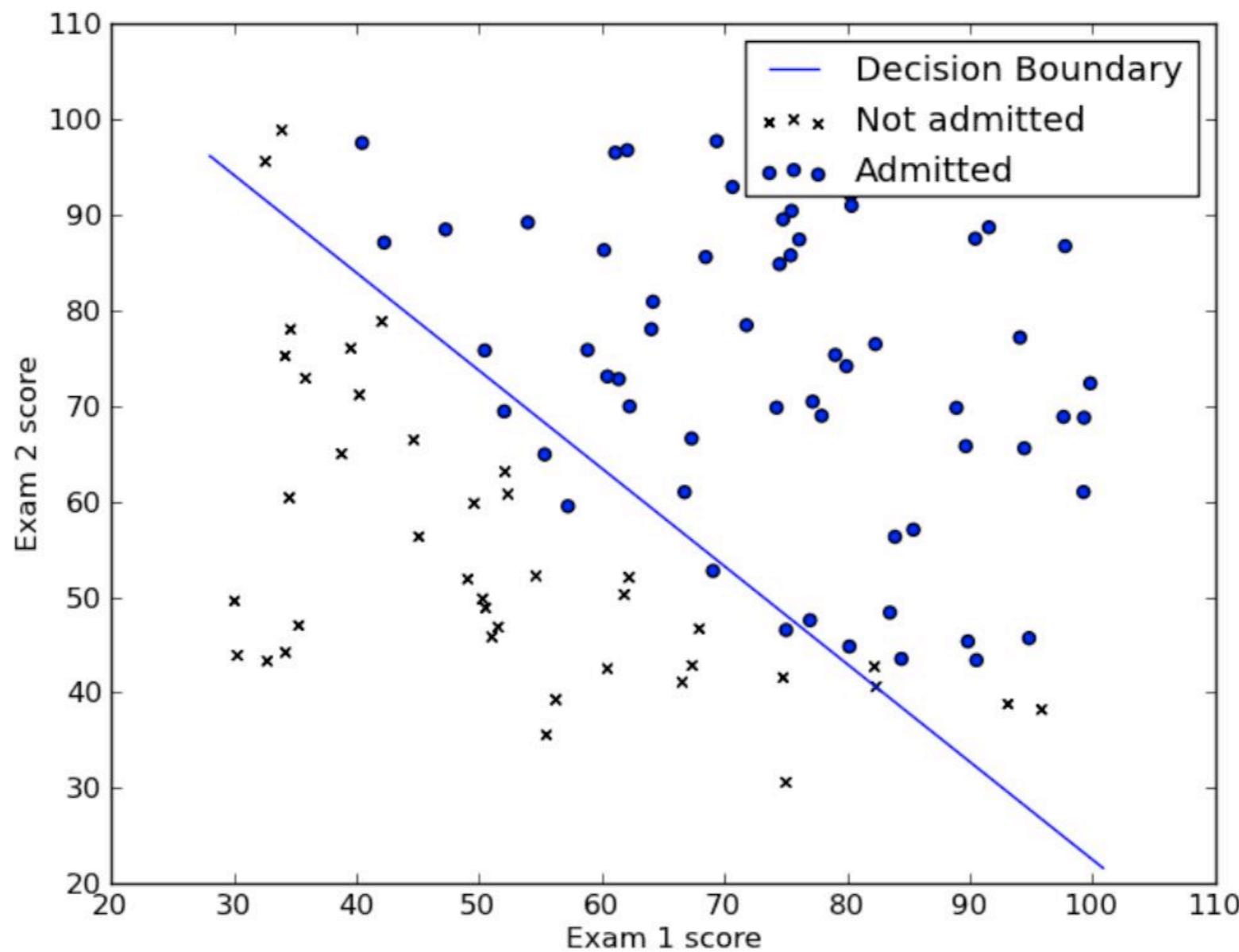
- 와인 품질 = $12.145 + (0.00117 \times \text{겨울철 강수량}) + (0.064 \times \text{재배철 평균기온}) - (0.00386 \times \text{수확기 강수량})$



- 선형 회귀(regression) $y = wX + b$



- 선형 분류(classification) $ay + bx > c$



모델의 특징

79

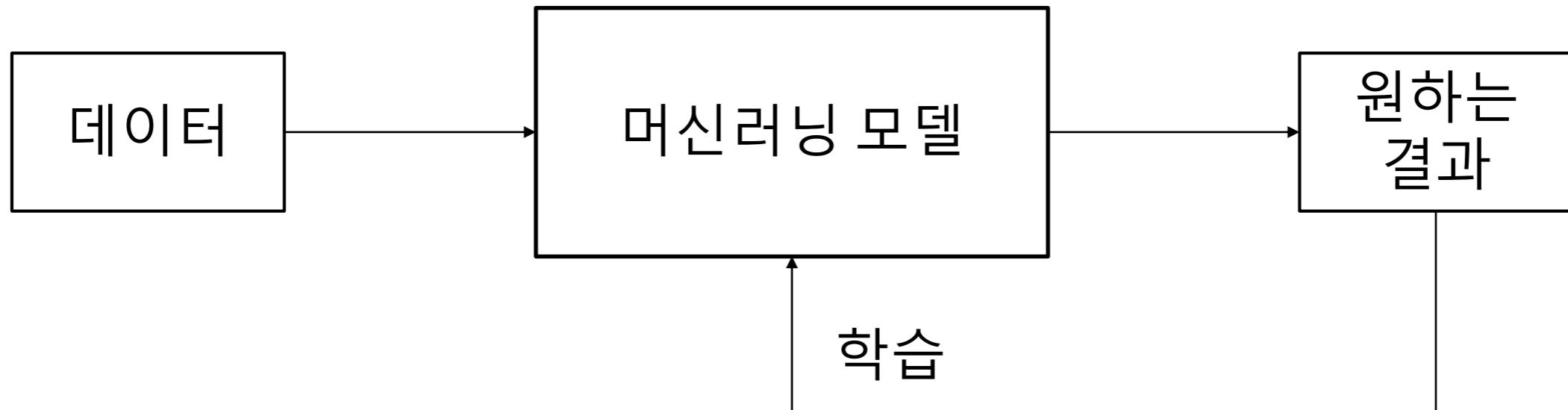
- 머신러닝에서는 데이터에 기반한 모델을 사용 (학습)
- 현실 세계의 많은 현상은 수식으로 간단히 모델링하기 어렵고 과학적으로 증명할 수는 없다.
- 그러나 성능이 꽤 유용



- 모델 구조: 모델의 동작을 규정하는 방법
- 모델 파라미터: 모델이 잘 동작하도록 정한 가중치 등 계수
 - 예: 머리카락 길이
 - 모델의 구조는 프로그래머가 선택
 - 적절한 파라미터를 찾는 것은 머신러닝 프로그램이 학습하여 찾는다

머신러닝 기본 동작

81



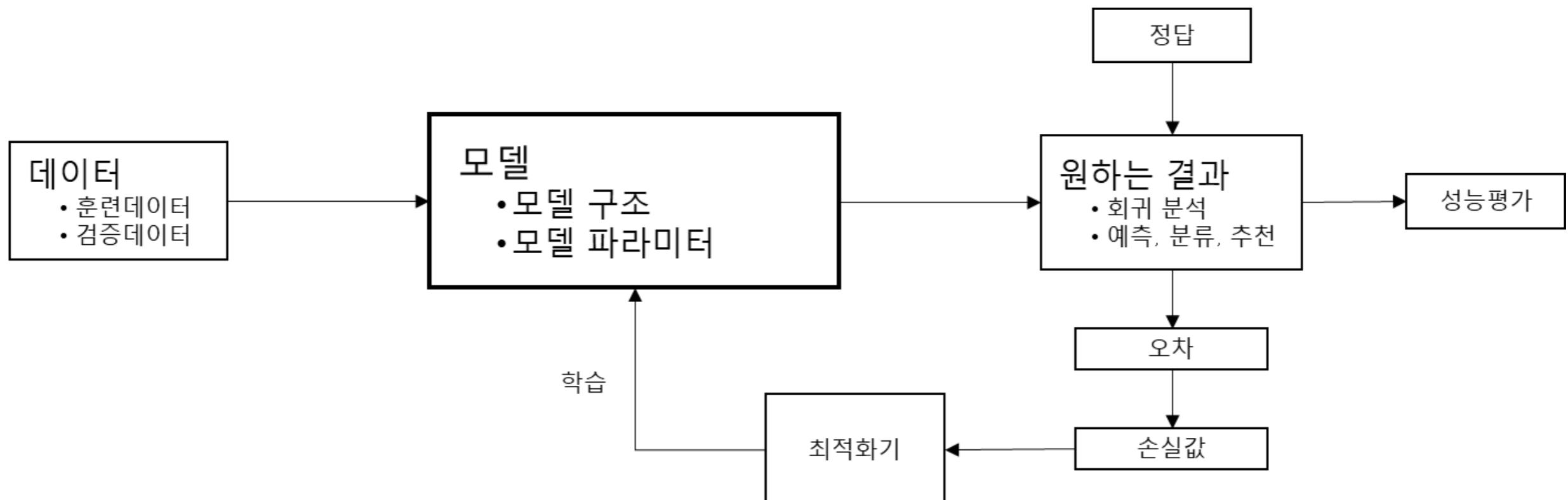
- 모델의 예측값과 실제 값과의 차이, 즉 오차로부터 손실함수 (loss function)을 계산한다
- 이 손실함수를 줄이는 방향으로 모델을 최적화 (학습) 한다
- 회귀분석에서 많이 사용하는 손실함수로는 오차 자승의 합의 평균치(**MSE: Mean Square Error**)

$$MSE = \sum_{k=1}^N (y - \hat{y})^2$$

- N: 배치 크기
- 배치 크기 같은 설정 환경 변수를 **하이퍼파라미터**라고 한다.
 - **하이퍼파라미터**는 사람이 선택하는 변수이며, 기계 학습으로 자동으로 갱신되는 변수는 “**파라미터**”라고 한다.

오차, 손실함수, 최적화, 파라미터

83



- 모델이 데이터를 이용하여 학습하는 과정을 훈련 (training)이라고 한다.
- 최적화 알고리즘에 의해서 파라미터(가중치 등)를 계속 갱신하여 모델의 예측값이 실제값에 수렴하도록 하는 것
- 엔트로피가 낮아지는 방향으로 (순도 혹은 데이터의 동질성이 높아지는 방향으로) 결정
- **검증(Validation)**
 - 훈련된 모델이 잘 동작하는지 확인하는 과정

모델의 성능 (Performance)

85

- 모델의 성능을 평가하는 척도 필요
- 분류에서는 정확도(accuracy)를 성능 척도로 주로 사용
 - (참고) 분류에서 손실함수로 크로스 엔트로피를 주로 사용한다
- **손실함수(Loss Function)**와 **성능지표(Performance)**의 차이점
 - **손실함수**를 정하는 목적은 모델을 훈련시킬 때의 기준으로 삼기 위해서이다. 모델은 손실함수를 최소화 하는 방향으로 학습한다.
 - **모델의 성능지표**는 이렇게 만든 모델이 궁극적으로 얼마나 잘 동작하는지를 평가하는 척도이다.

- 회귀에서는 손실함수로 MSE를 주로 사용한다.
- 그런데 MSE 값은 성능 지표로 사용하기에는 부족하다.
 - 예를 들어 키를 예측하였는데 MSE 값이 5.7cm이 나왔다고 하면 이 것의 성능이 얼마나 우수한지 다른 경우와 비교하기 어렵다.
 - 몸무게를 예측하였는데 MSE값이 3.8kg이라면 얼마나 우수한 것인가?
- 회귀분석에서는 **Coefficient of Determination (R-squared)**를 주로 사용한다

$$R^2 \equiv 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}}$$

$$SS_{\text{res}} = \sum_i (y_i - f_i)^2 = \sum_i e_i^2$$
$$SS_{\text{tot}} = \sum_i (y_i - \bar{y})^2$$

- 평균으로 예측 경우 $\rightarrow 0$
- 모든 예측이 정확하면 $\rightarrow SS_{\text{res}}=0, R^2=1$
- 모든 예측을 평균으로 하면 $\rightarrow R^2=1-1=0$
- $R^2 < 0 \rightarrow$ 예측을 평균보다도 못한 경우

대표적인 손실함수와 성능지표

87

	손실함수	성능지표
정의	손실함수를 줄이는 방향으로 모델이 학습을 함	성능을 높이는 것이 머신러닝을 사용하는 목적임
회귀 모델	MSE (오차 자승의 평균)	R^2
분류 모델	크로스 엔트로피	정확도, 정밀도, 재현률, F1점수

- 일반적으로 모델이 정교하고 복잡할수록 성능은 좋아지지만 모델을 만들거나 적용하는데 시간이 오래 걸린다.
 - 학습 시간 – 모델을 만드는데 걸리는 시간
 - 동작 속도 – 모델을 적용하는데 걸리는 시간

클러스터링

- ◆ 유사도 (Similarity)
- ◆ need “**Scaling**” as a preprocessing step
- ◆ K-Means
- ◆ 병합군집(agglomerative clustering)
- ◆ DBSCAN

유사도(Similarity)

90

- 항목간의 유사한 정도를 수치로 나타낸 것
- 분류나 예측에서 필요
- 메일이 스팸에 가까운지 아니면 정상 메일에 가까운지
- 추천에서 두 아이템 또는 사람이 서로 얼마나 가까운지

유사도 측정

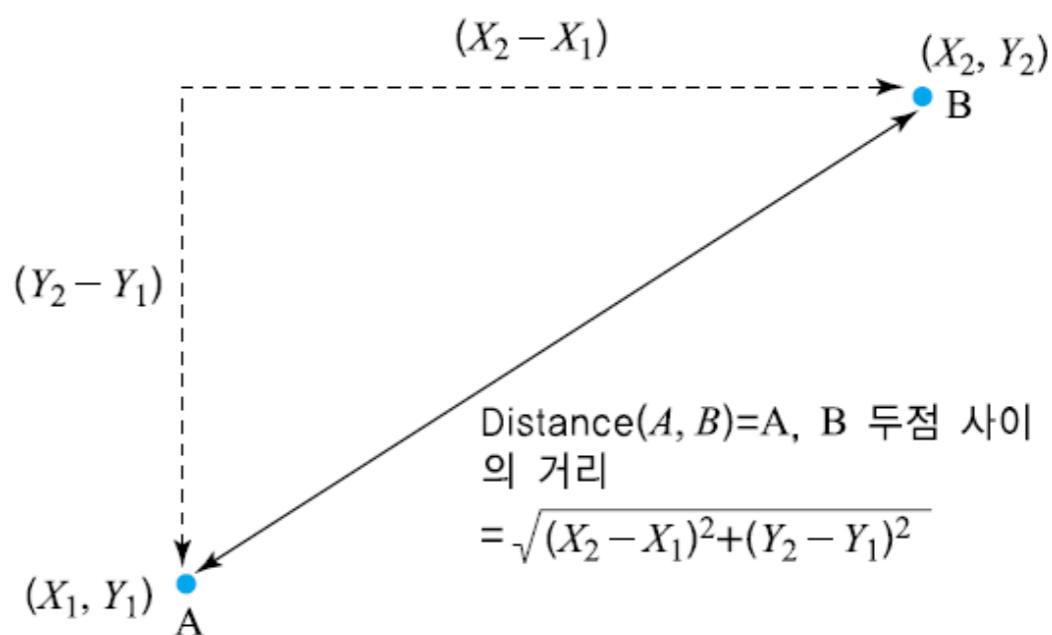
91

- A, B, C 중 누가 서로 가까울까?
 - 상대적인 차이를 보통 사용 (z 변환(표준 스케일링))

구분	키	몸무게	나이
A	174cm	70kg	21세
B	170cm	61kg	27세
C	162cm	73kg	29세

- 유사도 결과에 따라 데이터 분석 결과가 달라짐
- 분석 경험과 도메인에 대한 이해 필요함
- 최적의 분석 결과가 나오도록 유사도를 변경해 가면서 반복 수행 필요함
- 유사도 $s(\text{similarity})$ 는 $0 \leq s \leq 1$ (1에 가까울수록 유사도 높음)
- 유사도의 상대 개념으로 거리(distance) 사용
 - 유사도와 거리의 관계: $d = 1 - s$

- 기하학의 공간(space) 상의 거리 - 유클리디언 (Euclidian) 거리

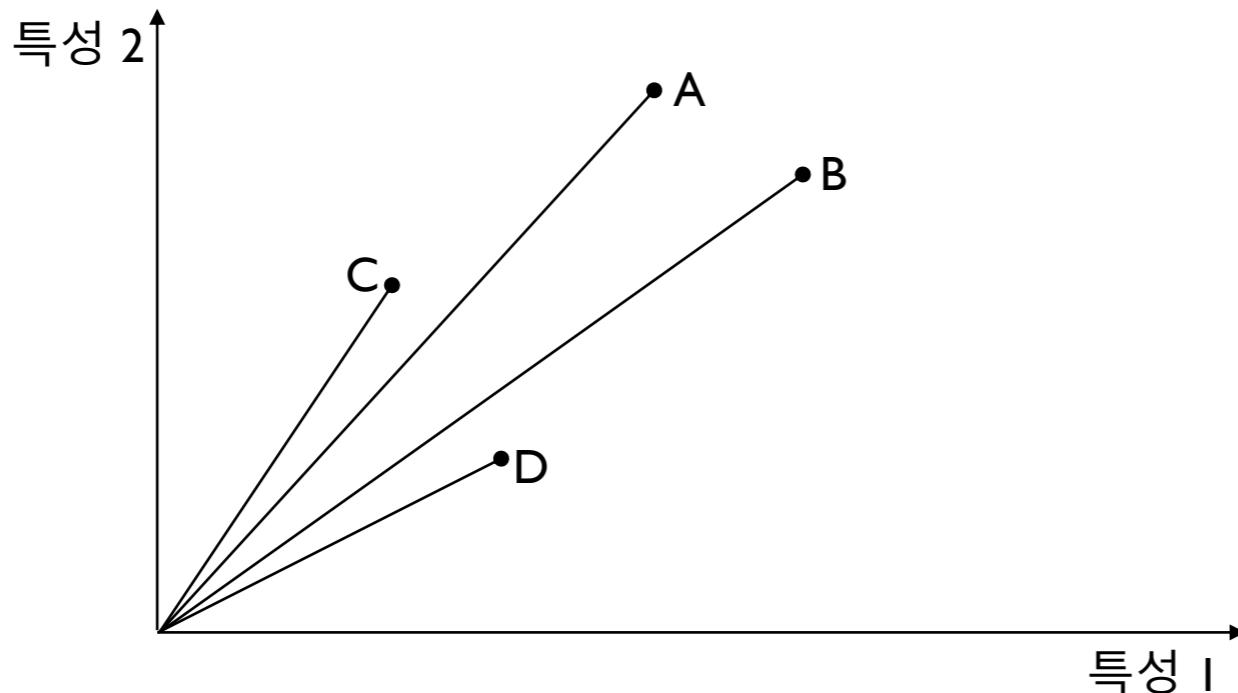


- n 차원 : $\sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \dots + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^n (p_i - q_i)^2}$

코사인(cosine) 유사도 – 방향성, 취향

94

- 공간상의 두 점이 만드는 각도를 기준으로 유사도를 측정하는 방법 (-1 ~ +1 사이의 값을 갖는다)
- 공간상 거리가 멀어도 두 점이 가리키는 방향이 같으면 서로 비슷하다고 보는 것



$$s_{\cos}(x, y) = \frac{X \cdot Y}{|X||Y|}$$

- A와 C가 가깝고 B와 D가 가깝다고 정의

자카드(Jaccard) 유사도

95

- 비슷한 취향의 사람을 찾을 때 사용 - 영화, 도서, 음악 추천 등
- 영화 보는 취향에 따른 유사도 측정
- 지난 1년 동안 국내에 개봉된 영화가 500편
 - A와 B가 본 영화 중 겹치는 영화가 5편, $5/500 = 0.01$
 - A와 C가 본 영화 중 겹치는 영화가 10편, $10/500 = 0.02$
 - 즉, $0.01 < 0.02$ 이므로 A와 C가 더 가깝다고 할 수 있음
 - 위와 같은 계산 방법이 적절한가?

- 어떤 두 항목이 겹치는 부분의 절대량만을 보지 않고, 두 항목의 공통 부분이 얼마나 많은지를 고려하여 이에 대한 상대적인 값을 유사도로 사용해야 함

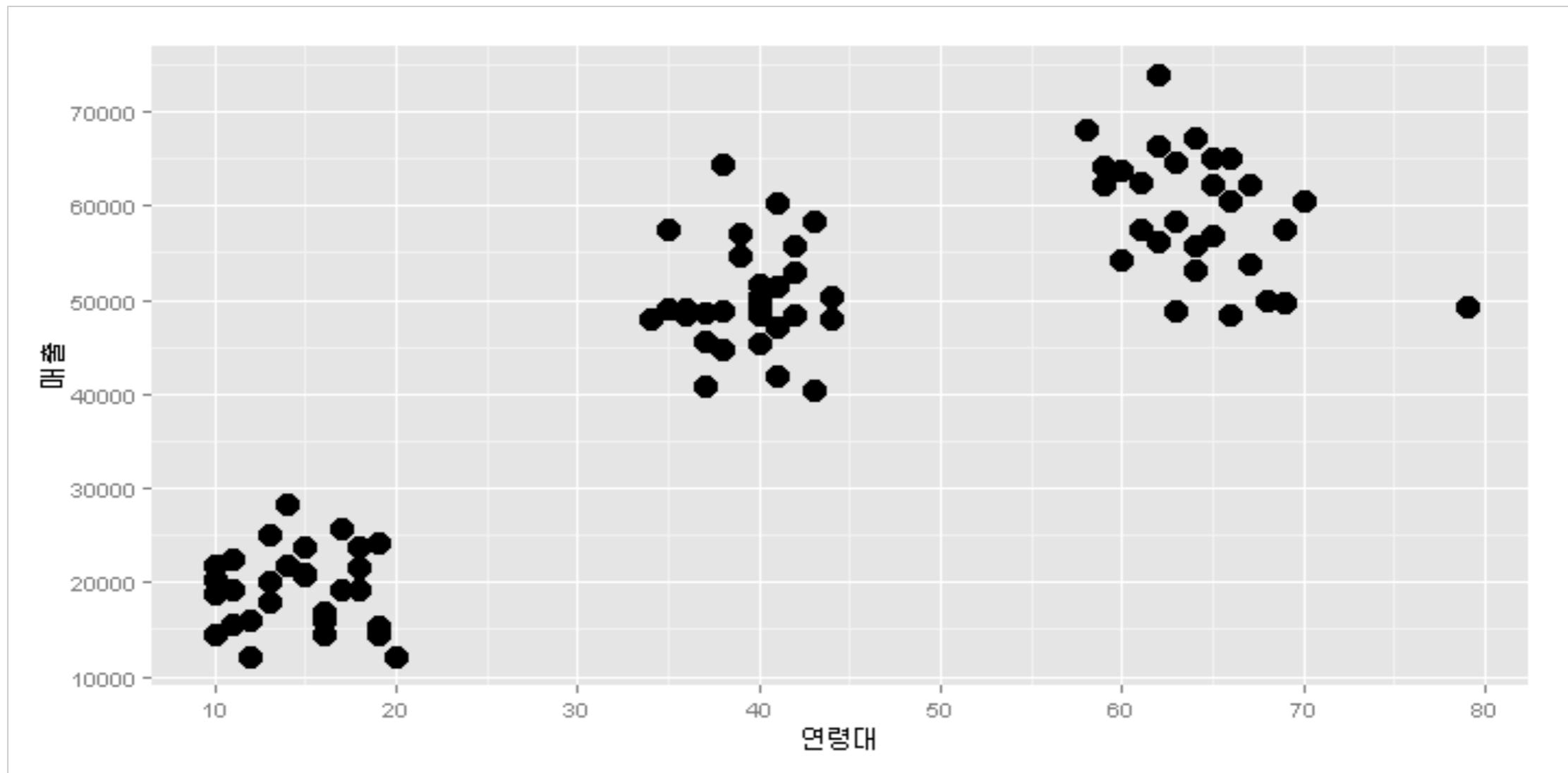
$$S_{Jaccard}(X, Y) = \frac{|X \cap Y|}{|X \cup Y|}$$

- A, B, C가 각각 지난해 본 영화의 총 개수가 20편, 50편, 200편
- $J(A,B) = 5 / (20+50-5) = 0.076$
- $J(A,C) = 10 / (20+200-10) = 0.047$
- 즉, $0.076 > 0.047$ 이므로 A와 B가 더 가깝다고 할 수 있음

클러스터링 (clustering)

97

- 성격이 비슷한 항목들을 그룹으로 묶는 작업
(예: 아래의 경우 고객을 몇 개의 그룹으로 나누는게 좋을까?)

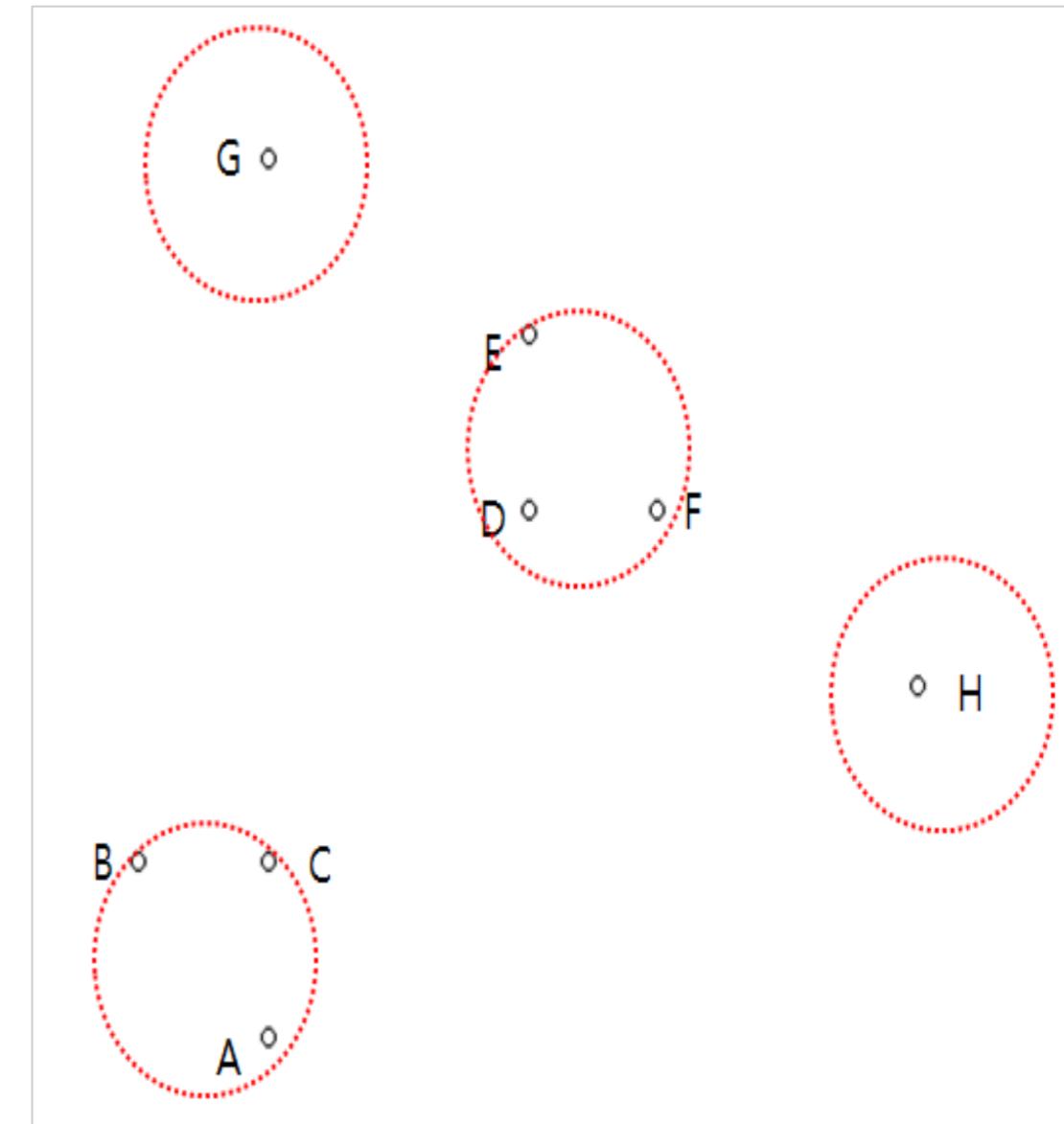
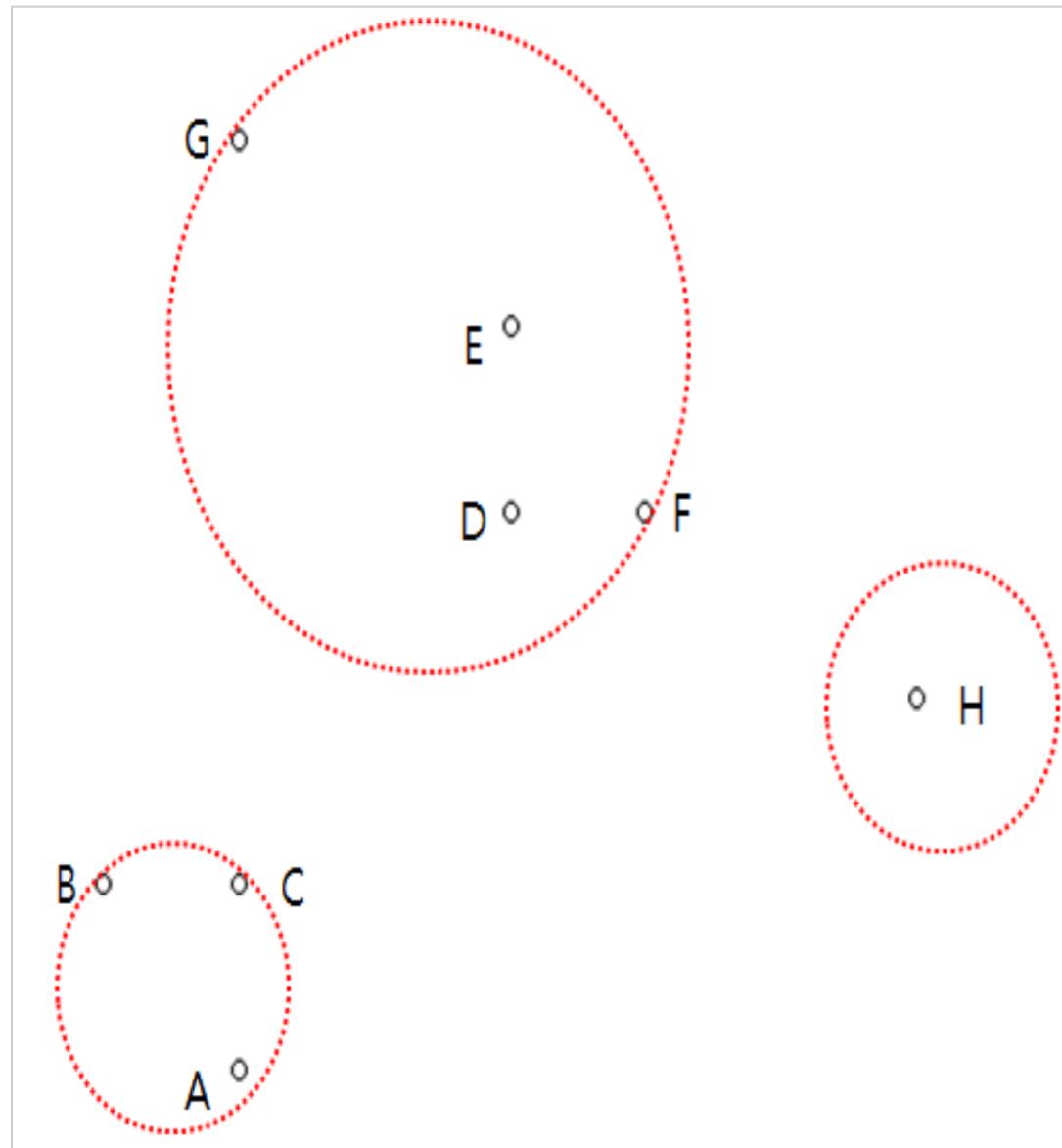


- 조건
 - 같은 그룹 내의 항목들은 서로 속성이 비슷함 (**유사도**가 큼)
 - 다른 그룹에 속한 항목과는 속성이 서로 다름 (유사도가 작음)
- 비정상 패턴 (이상치) 식별에도 사용된다
 - (ex) 컴퓨터 시스템에 침입한 해커의 행동

클러스터 수, k

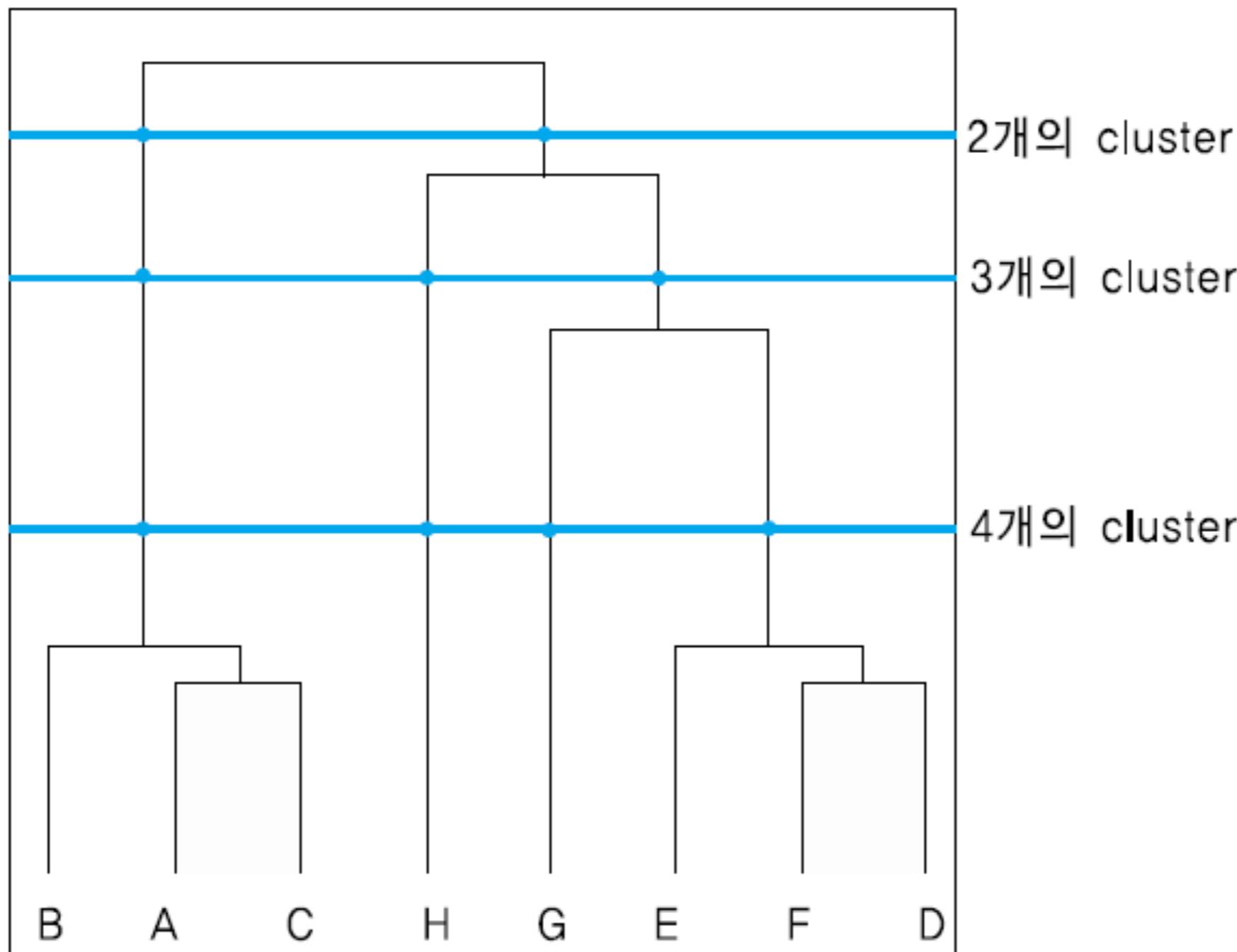
99

- 군집화에서 가장 중요한 작업으로 적정한 군집의 수(k)를 먼저 찾아야 함.



덴드로그램(dendrogram)

100



K-Means 알고리즘

101

- 공간상에 임의의 k 개의 임의의 초기 지점을 클러스터 중점으로(cluster center) 정함
- 클러스터 중점을 중심으로 거리가 가까운 항목을 선택하여 클러스터 공간을 나눔
- 각 클러스터에 포함된 항목들의 평균 위치를 구해 이를 새로운 클러스터 중점(centroid)으로 변경
- 새로 설정된 **센트로이드**를 중심으로 경계를 다시 그림
 - 각 항목들이 소속된 클러스터가 바뀔 수 있음
- 변경된 항목들을 가지고 클러스터 중심을 다시 계산
- 더 이상 클러스터의 모양이 바뀌지 않을 때까지 반복 수행함
 - KMeans() 사용

Mean-Shift 알고리즘

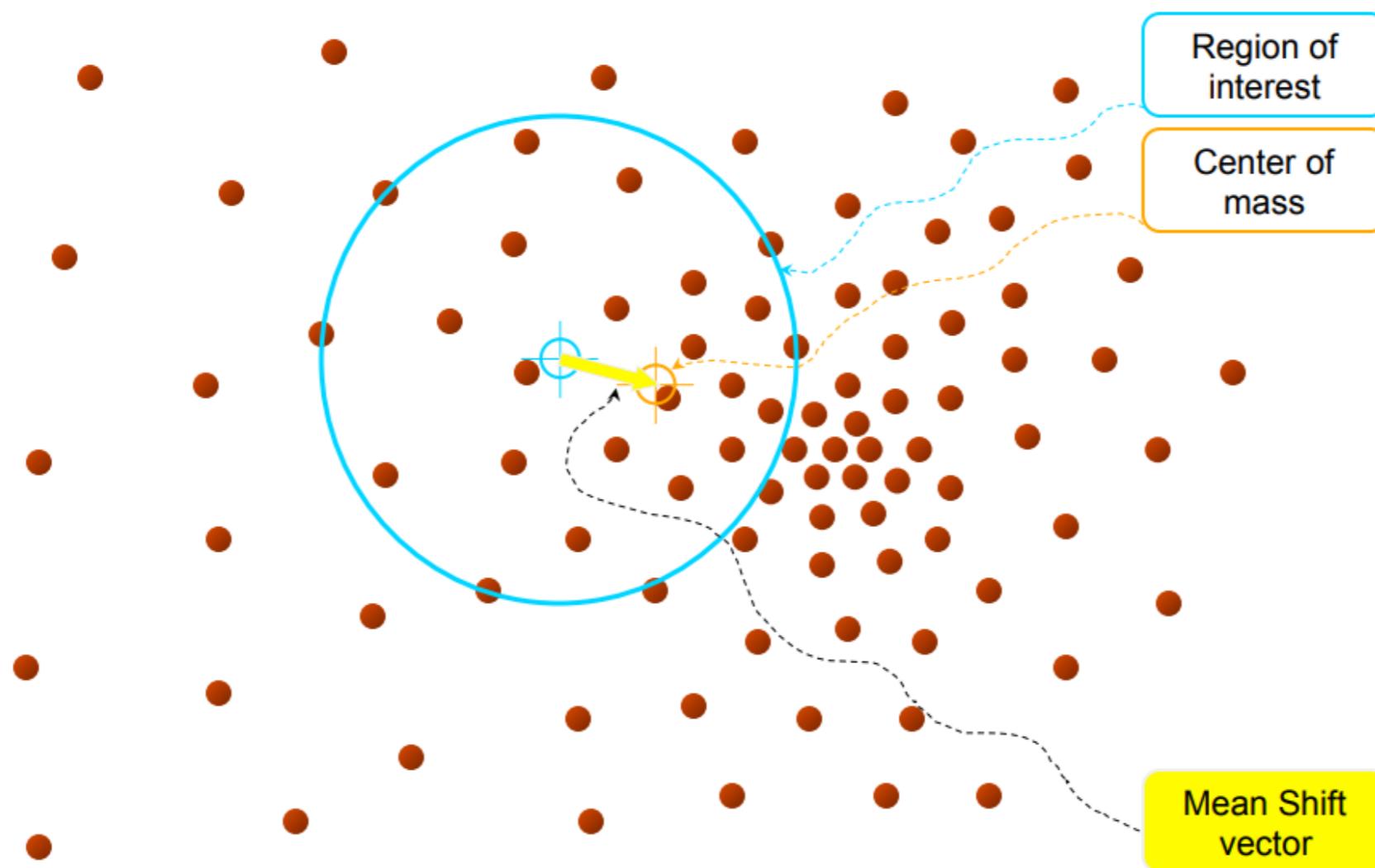
102

- 데이터의 분포에 가정을 하지 않는 non-parametric 클러스터링 기법 (clustering 수(k)를 주지 않는다)
- 어떤 데이터 분포의 peak 또는 무게중심을 찾는 방법으로, 현재 자신의 주변에서 가장 데이터가 밀집된 방향으로 이동.
- 즉, 중심점(centroid, local maxima)를 찾아 군집의 개수(k)가 정해짐.
- 모든 학습 데이터에 개별적으로 윈도우를 정하고, 각 윈도우의 중심점을 계산하여 중심점의 위치를 계속 갱신함.
- 계산량이 많고, local minima 에 빠질 위험이 있음.
- MeanShift() 사용
- Mode-seeking 알고리즘
 1. 현재위치에서 반경 r 이내에 들어오는 데이터를 구한다.
 2. 이들의 무게중심을 구해 현재 위치를 이동시킨다.
 3. 1-2 과정을 위치변화가 없을 때까지 (수렴할 때까지) 반복한다.

Mean-Shift 알고리즘

103

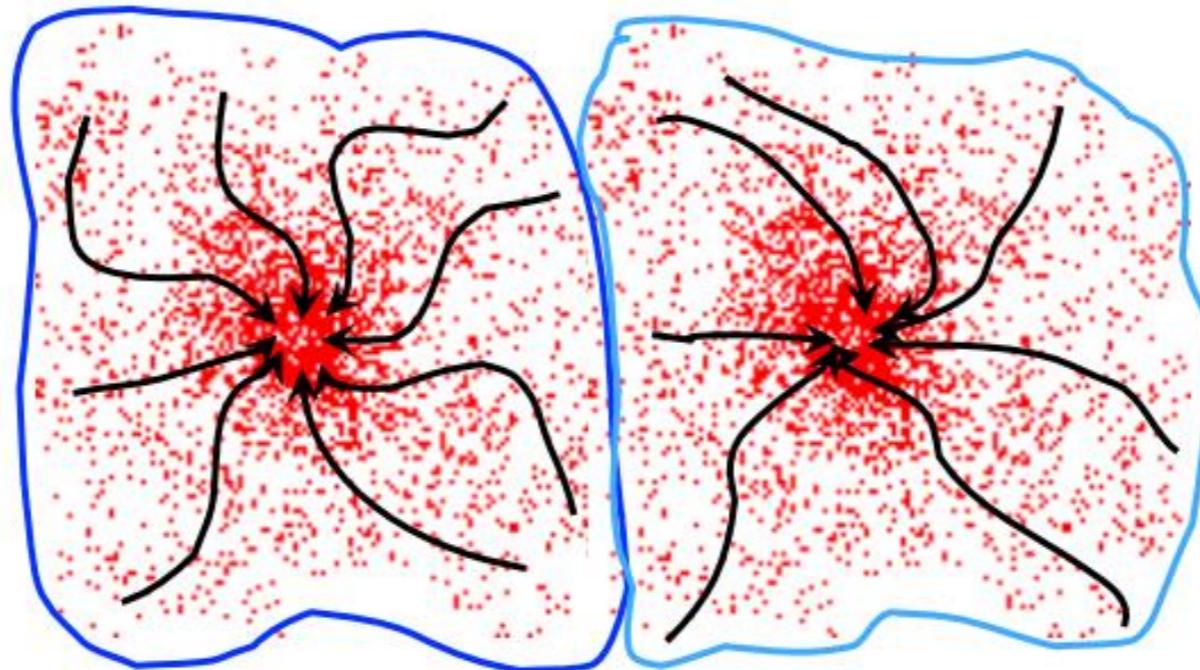
- (ex) from the slide by Y.Ukrainitz & B.Sarel at Stanford University



Mean-Shift 알고리즘

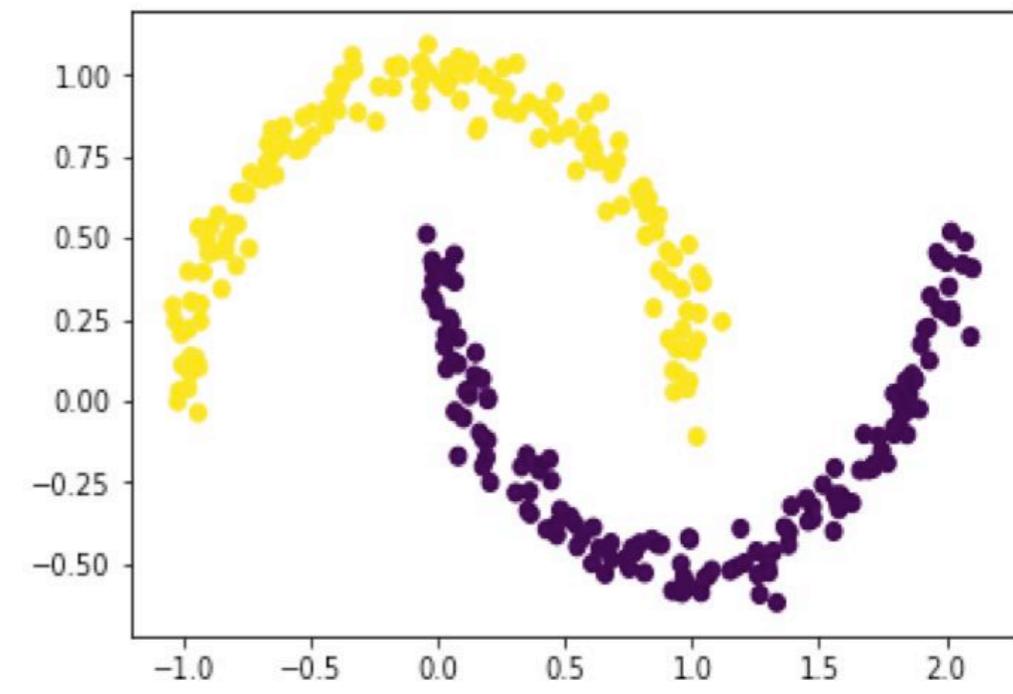
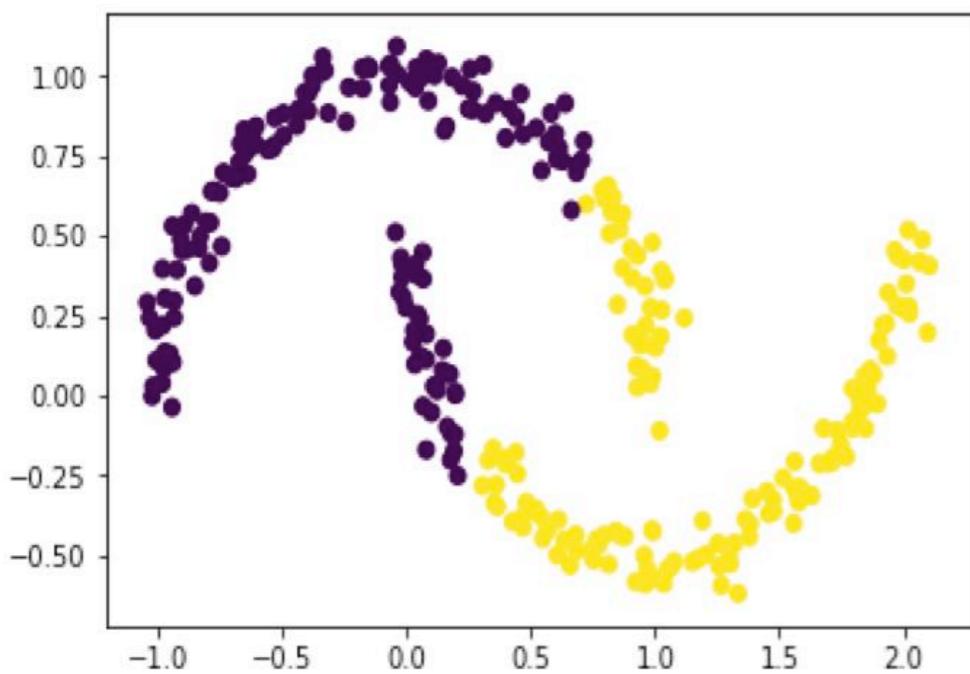
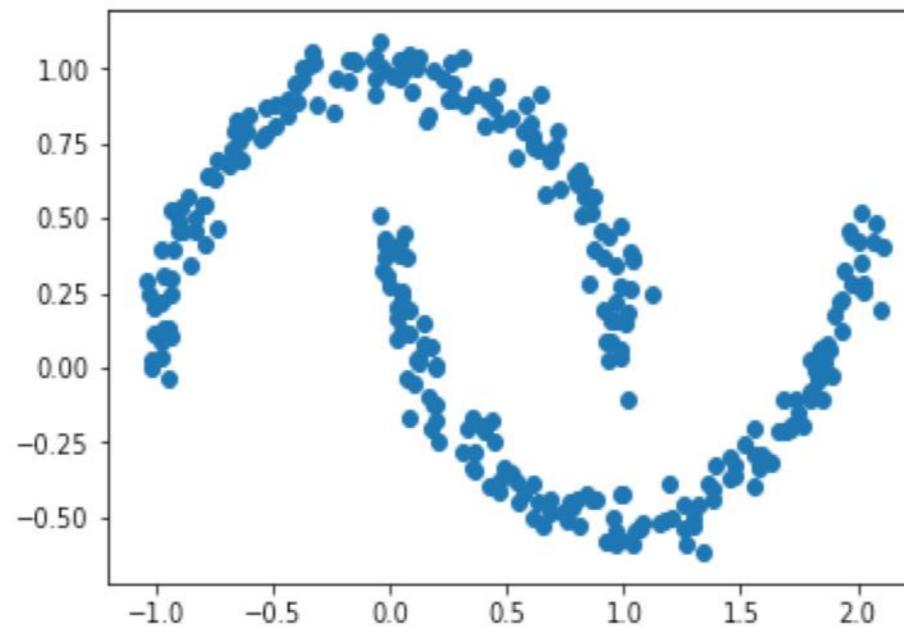
104

- (ex) from the slide by Y.Ukrainitz & B.Sarel at Stanford University
 - Cluster: all data points in the attraction basin of a mode
 - Attraction basin: the region for which all trajectories lead to the same mode



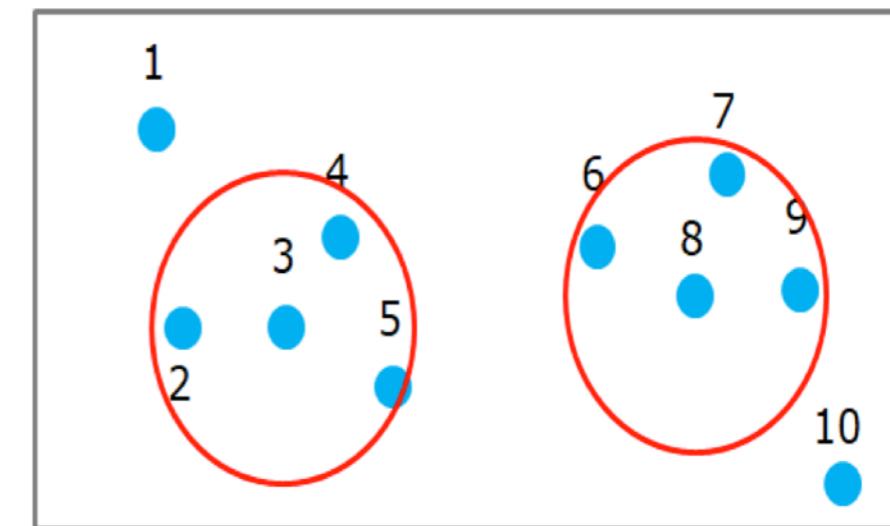
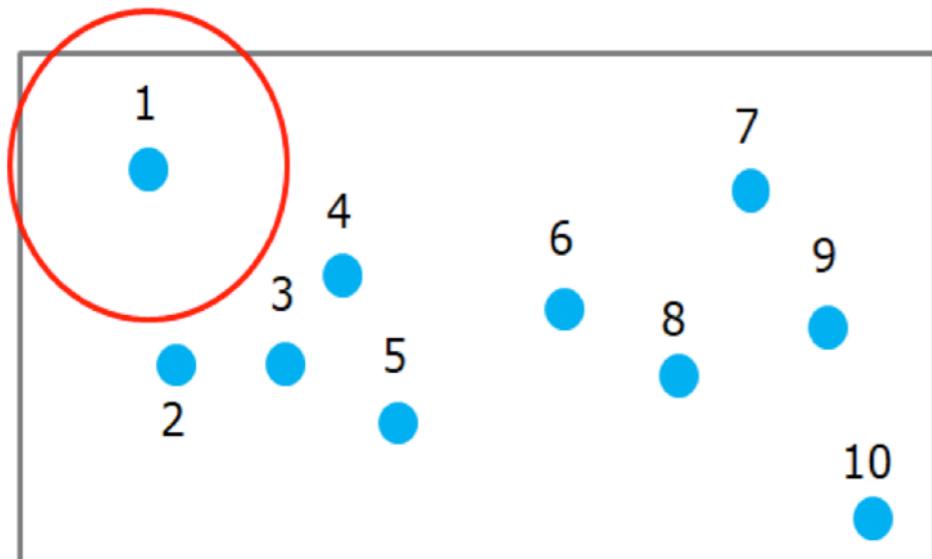
Two Moons 데이터

105

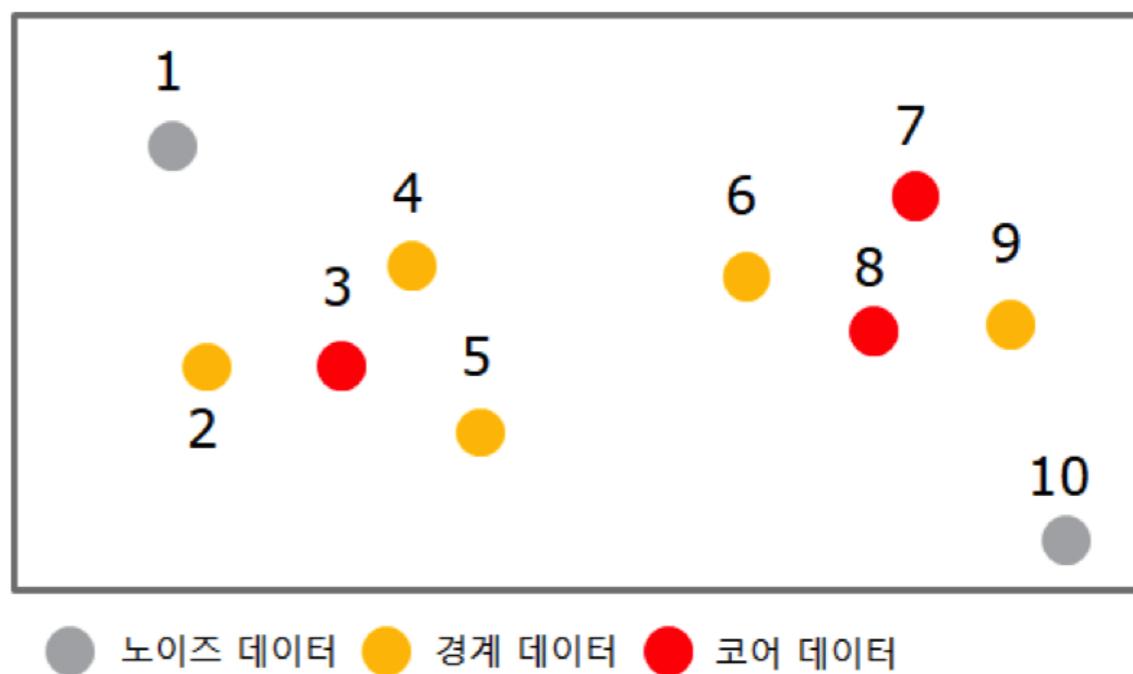
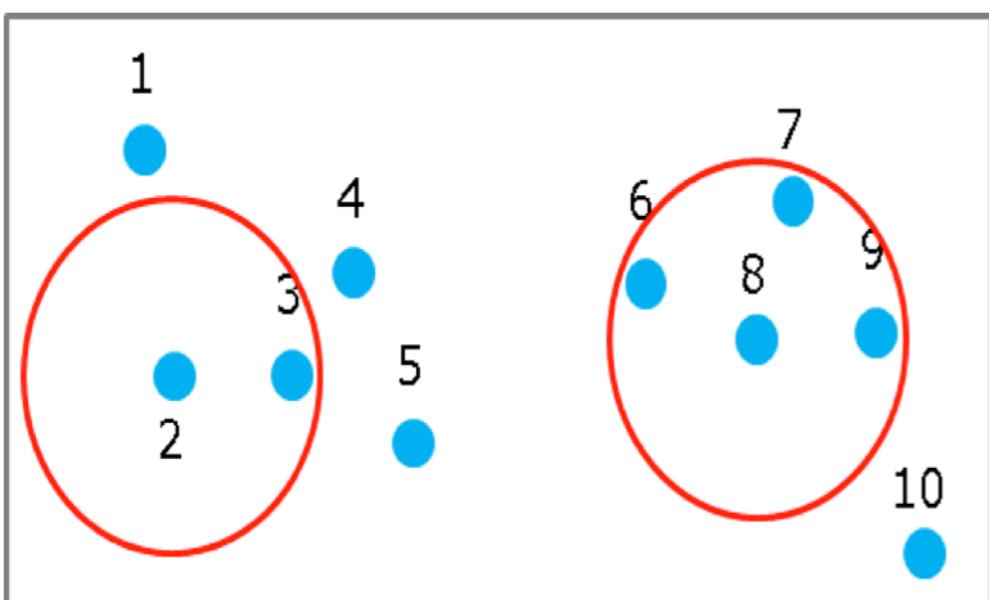


- Density based Spatial Clustering of Applications with Noise
(one of the most common clustering algorithms)
- **밀도기반** 클러스터링 알고리즘이다.
- k-means처럼 단순히 거리만을 기준으로 군집화를 하는 것이 아니라 “가까이 있는 샘플들은 같은 군집에 속한다”는 원칙으로 군집을 차례로 넓혀가는 방식이다.
- 샘플들의 몰려 있는 정도 즉, 밀도가 높은 부분을 중심으로 인접한 샘플들을 포함시켜 나간다.
- 한 점을 기준점으로 반경 r 내에 점이 n 개 이상 있으면 하나의 군집으로 인식하는 방식이다.

- 1번 데이터를 중심으로 보면 반지름 r 인 원 안에 군집이 되기 위한 최소기준인 (예를 들어 $n=4$ 라면) 샘플이 없다.
- 이 데이터는 노이즈 데이터(noise point)가 되며 클러스터에서 제외한다.
- 3번과 8번 데이터를 중심으로 보면 원 안에 4개의 점이 있으며 이러한 데이터를 코어 데이터(core point)라고 한다.
 - 코어 데이터들은 스스로 클러스터를 형성할 수 있다.



- 2번 데이터는 최소 기준인 4개의 데이터를 포함하지는 못하지만 코어 데이터인 3번을 포함한다. 이런 데이터를 경계 데이터(border point)라고 하며 인접한 군집에 포함시킨다.
- 정해진 반지름 r 인 원을 이용해 코어 데이터, 경계 데이터, 노이즈 데이터들을 분류하면 아래와 같다.
- 두 개의 클러스터와 두 개의 노이즈를 구분했다.
- 코어데이터가 다른 코어의 일부가 되면 하나의 군집으로 연결.



Labs

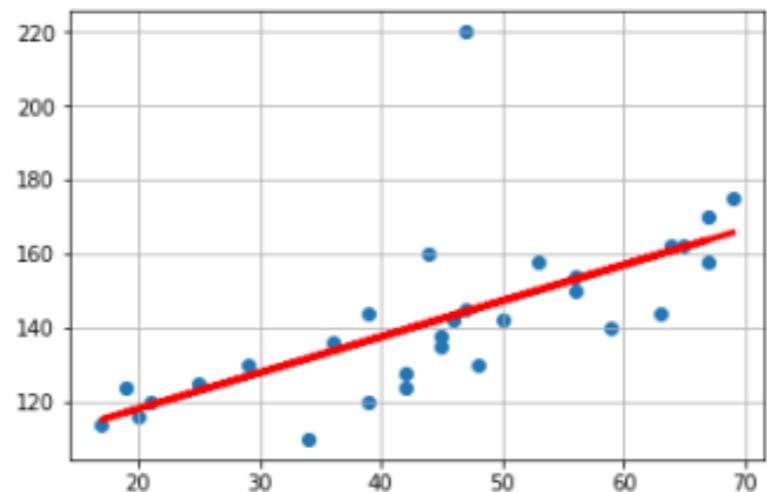
- gg-27 전력클러스터링
- gg-28 클러스터링비교

회귀

선형 회귀

III

- (예) 나이와 혈압의 관계



$$y = ax + b$$

- 다중 선형회귀(multiple linear regression)

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_p x_p$$

- 손실함수로 RMSE 또는 MSE를 주로 사용

$$\text{RMSE}(X, h) = \sqrt{\frac{1}{m} \sum_{i=1}^m (h(X^{(i)}) - y^{(i)})^2}$$

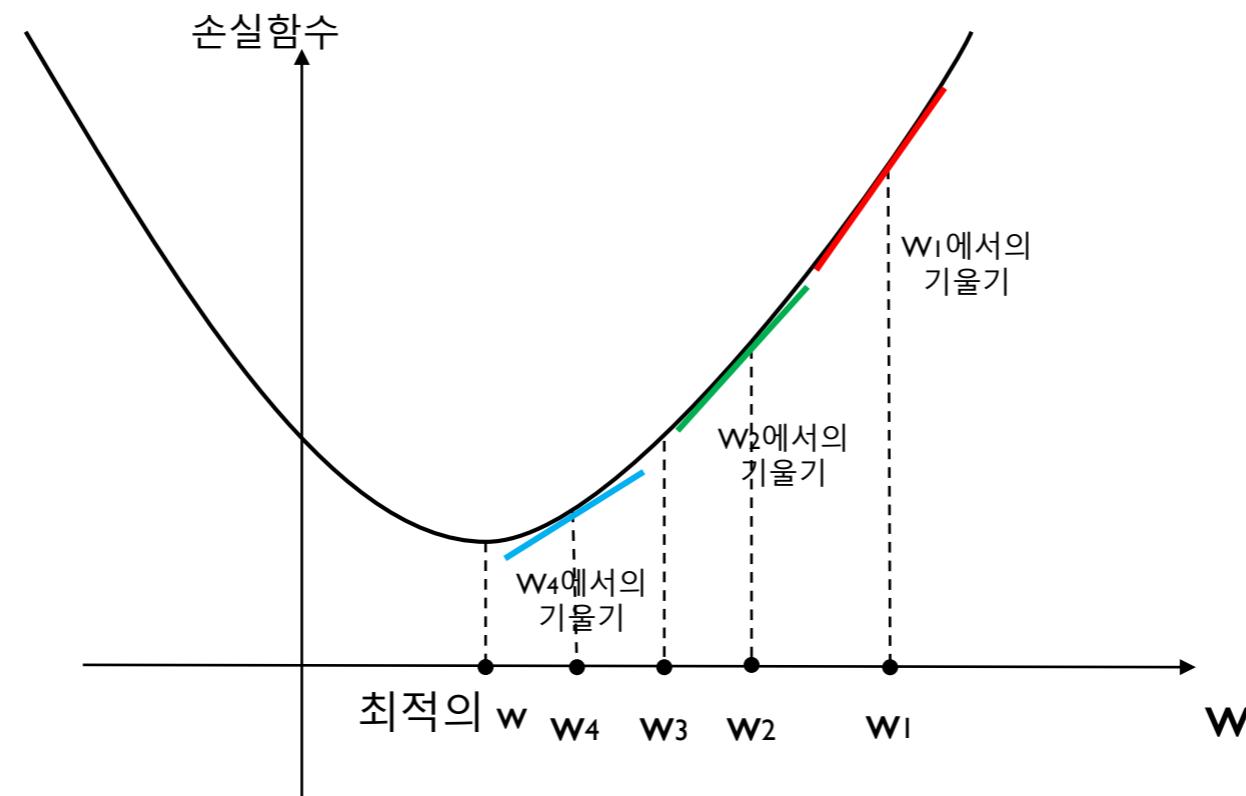
변수: Feature, Target(Label)

112

- 독립변수: $x_1, x_2, x_3, \dots, x_p$
 - Predictor, **Feature**, 설명변수(explanatory variable) 이라고 함.
- 종속변수: y
 - 목적변수(**target**), outcome 변수, response 변수, 또는 **Label** 이라고 함

- 가장 일반적인 최적화 알고리즘: 경사하강법 (Gradient Descent)
- 손실함수를 계수에 관한 그래프로 그렸을 때 최소값으로 빨리 도달하기 위해서는 현재 위치에서의 기울기(미분값)에 비례하여 반대방향으로 이동하는 방식

$$W_i = W_{i-1} - \eta \text{Grad}(i)$$



학습률 (Learning Rate)

114

- 학습률: 계수를 업데이트 하는 속도를 조정하는 변수
 - 학습률이 너무 작으면 수렴하는데 시간이 오래 걸리지만 최저점에 도달했을 때 흔들림 없이 안정적인 값을 얻게 되고,
 - 학습률을 너무 크게 정하면 학습하는 속도는 빠르나 자칫하면 최저점으로 수렴하지 못하고 발산하거나 수렴하더라도 흔들리는 오차가 남아있을 수 있다.
- 학습 스케줄(learning schedule) 기법
 - 초기에는 학습률을 크게 정하고 (학습률을 빠르게 학고) 오차가 줄어들면 학습률을 줄여서 안정상태(steady state)의 오차를 줄이는 방법

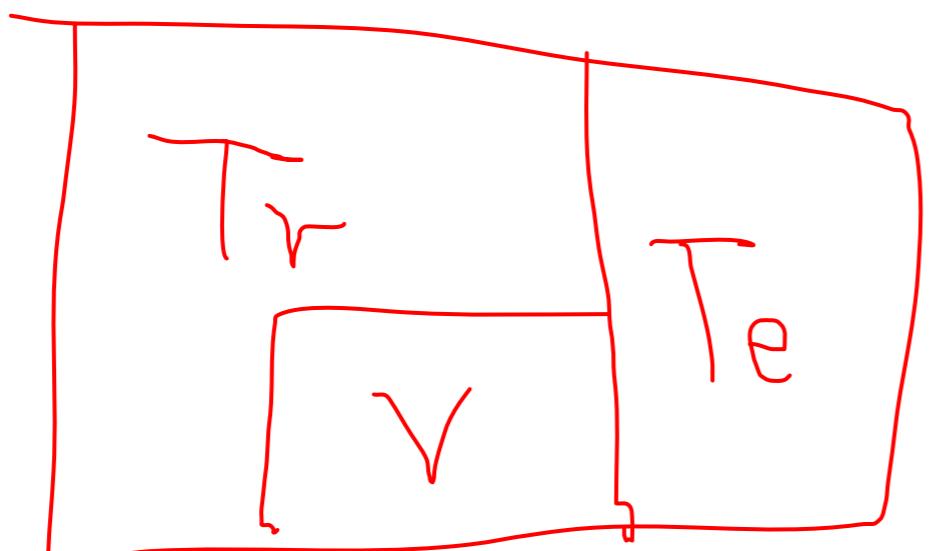
- 경사하강법을 적용하려면 특성 변수들을 모두 동일한 방식으로 **스케일링**해야 한다.
- 특성 값마다 크기의 편차가 크면 특정 변수에 너무 종속되어 동작할 수 있고 이로 인해 수렴속도가 직선이 되지 않고 오래 걸릴 수가 있다.
- Local minimum 에 머무를 수 있음.
- 배치 사이즈가 커지면 시간이 오래 걸림.

- 배치(Batch) GD (or Mini Batch GD)
 - 일반적으로 배치 GD방식을 많이 사용하는데, 적절한 크기 (10~ 1,000)의 배치단위로 입력 신호를 나누어 경사하강법을 적용하는 방식이다.
- SGD (확률적 경사하강법: Stochastic GD)
 - 한 번에 한 샘플씩 랜덤하게 골라서 훈련에 사용하는 방법이다.
 - 즉 샘플을 하나만 보고 계수를 조정한다. 계산량이 적어 동작속도가 빠르고, 랜덤한 방향으로 학습을 하므로 전역 최소치 (global minimum)를 찾을 가능성이 높아진다.
 - 매 샘플이 너무 랜덤하여 방향성을 잃고 수렴하는데 시간이 오래 걸릴 가능성도 있다. 노이즈가 심함.
 - In Python, **use different Loss function and penalty**
 - ▶ SGDClassifier() for classification
 - ▶ SGDRegressor() for regression

데이터의 대표성

117

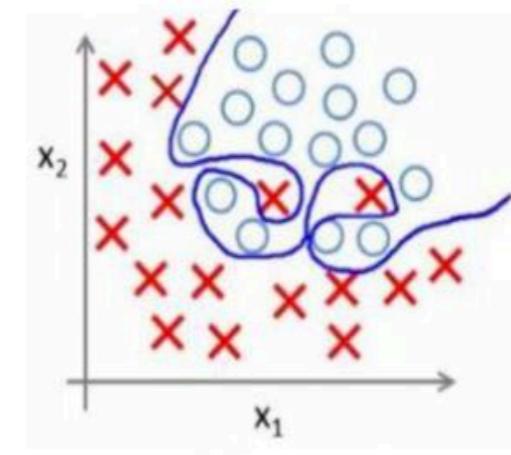
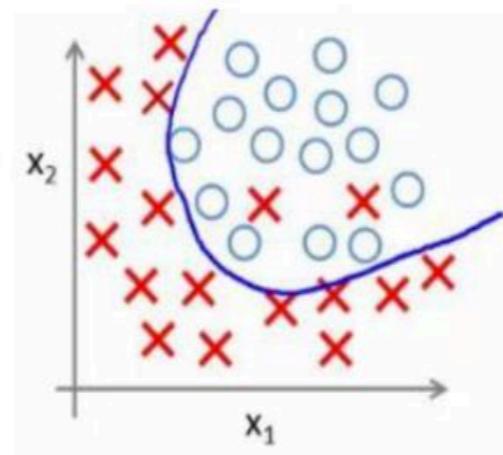
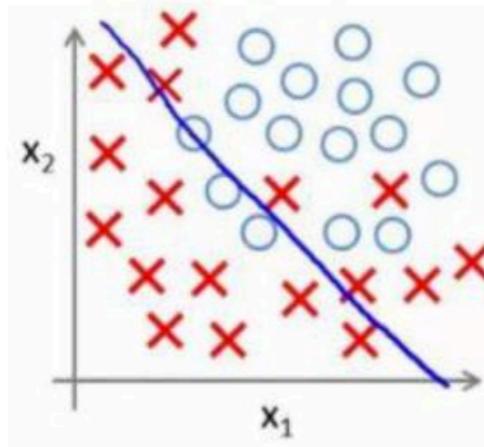
- 훈련 데이터가 미래에 나타날 가능성이 있는 모든 데이터의 특징을 반영하도록 구성해야 한다.
 - 예: 지리적, 인종적, 나이별, 소득별, 성별 등 균일성 유지
- **훈련, 검증, 테스트** 샘플 데이터가 전체 데이터의 특징을 계속 유지할 수 있어야 함



과대적합 (over fitting)

118

- 모델이 훈련 데이터에 대해서만 잘 동작하도록 훈련되어 새로운 데이터에 대해서는 오히려 잘 동작하지 못하는 것
- 과대적합된 모델은 훈련 데이터에 대해서는 매우 우수한 성능을 보이지만 일반성이 떨어진다.
- 머신러닝에서는 과대적합을 피해서 일반적으로 잘 동작하게 모델을 만드는 것이 매우 중요하다.
 - 이를 모델의 일반화(generalization)라고 한다.



규제화 (Regularization)

119

- 과대적합이 발생하는 원인은 훈련 데이터가 너무 적어서 모델이 학습을 충분히 할 수 없는 경우에 발생한다.
- 과대적합을 줄이려면 훈련 데이터를 많이 확보하여 다양한 경우를 대비하여 일반적인 모델을 만들어야 한다.
- 만일 학습할 데이터가 부족하다면 모델 구조를 단순하게 만들어야 한다.
- 이렇게 모델이 일반성을 갖도록 모델의 기능을 제한하는 것을 모델에 제약을 가한다고 하여 규제화(regularization)라고 한다.
 - Data augmentation
 - L1, L2 Regularization
 - Early stopping
 - Dropout: neural net

과소적합 (under fitting)

|20

- 모델이 너무 간단하여 성능이 미흡한 경우
- 과소적합을 피하려면 좀 더 상세한 모델 구조를 사용해야 한다.
- 머신러닝에서는 과대적합과 과소적합을 모두 피해야 하며 최적의 예측을 수행하는 모델을 만드는 것이 중요하다.

회귀 성능 평가 지표

|2|

- 회귀 분석에서 학습에 사용하는 손실함수로 RMSE (또는 MSE)를 사용한다고 했다.
- MSE 값을 보면 오차의 절대적인 크기를 알 수는 있으나 성능 평가에는 사용하기가 어렵다.
- 회귀분석에서는 성능 평가 지표로 R^2 를 주로 사용한다

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y}_i)^2}$$

y_i 는 실제 값, \hat{y}_i 는 예측한 값, \bar{y}_i 는 샘플의 평균 값

- 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 추가하였다.
 - 계수 자체가 가능하면 작은 값이 되기를 원한다는 것이다.
 - 알파는 이 축 항목의 비중을 얼마나 크게 할지를 정하는 하이퍼파라미터이다.
- 이는 여러 계수들을 가능한 골고루 반영하라는 의미를 갖는다. 왜냐하면 계수의 자승의 합을 줄이려면 각 계수의 크기를 줄여야 전체 자승의 합이 최소화되기 때문이다.

- 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

- 라소 규제에서는 모든 계수의 절대치들의 합을 추가로 더하는 방법이다 (자승을 취하지 않음).

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

- 릿지 규제와 유사한 것처럼 보이지만 사실은 반대의 효과를 얻는다.
- 릿지 규제에서는 특별히 비중이 큰 계수를 지양하고, 가능한 여러 가중치를 골고루 반영하는 효과를 얻었다.
- 라쏘 규제를 적용하면 절대값이 작은 (영향력이 적은) 계수가 먼저 사라지는 효과를 얻는다.
- 라쏘 규제는 L1 규제라고도 부른다

- 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 알파는 일반화의 정도를 조정하는 하이퍼파라미터이고 감마는 L2와 L1 규제의 반영 비중을 조절하는 하이퍼파라미터이다.

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

Labs

- gg-29 선형회귀
- gg-30 회귀직선
- gg-31 선형회귀특성

분류

분류의 손실함수(Loss Function)

|28

- 분류에서는 손실함수로 MSE를 사용할 수 없다.
- 대신, 분류에서 정확도(accuracy)를 손실함수로 사용할 수 있다
 - 예를 들어 남녀 각각 50명씩 100명에 대해 남녀 분류를 시도하였으나 96명을 맞추고 4명을 틀렸다면 정확도는 0.96이다.
 - 그러나 정확도를 손실함수로 사용하는데에는 다음과 같은 문제가 있다.
- **카테고리 분포 불균형** 시 문제
 - 남자가 95명, 여자가 5명이 있는 그룹에서 남자는 1명을 잘 못 분류하고 여자는 3명을 잘 못 분류했다고 하면, 정확도는 여전히 0.96이다
 - 손실을 제대로 측정하지 못한다
 - 이를 보완하기 위해서 **크로스 엔트로피(cross entropy)**를 사용한다.



크로스 엔트로피(Cross Entropy)

|29

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

정답의 불확실성

- p_i 는 어떤 사건이 일어날 실제 확률이고, p_i' 는 예측한 확률이다
- 남녀가 50명씩 같은 경우

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

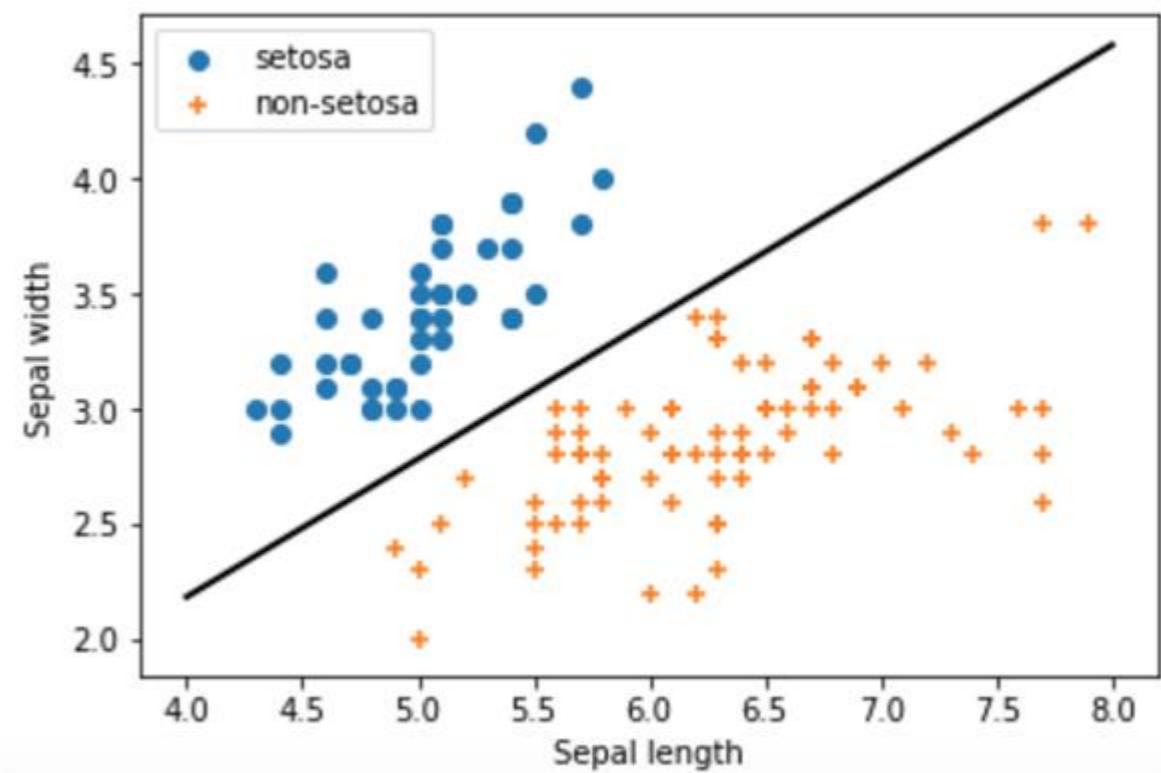
- 남자가 95명 여자가 5명인 경우

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

- 선형분류에서의 결정경계

$$ax + by + c > 0$$

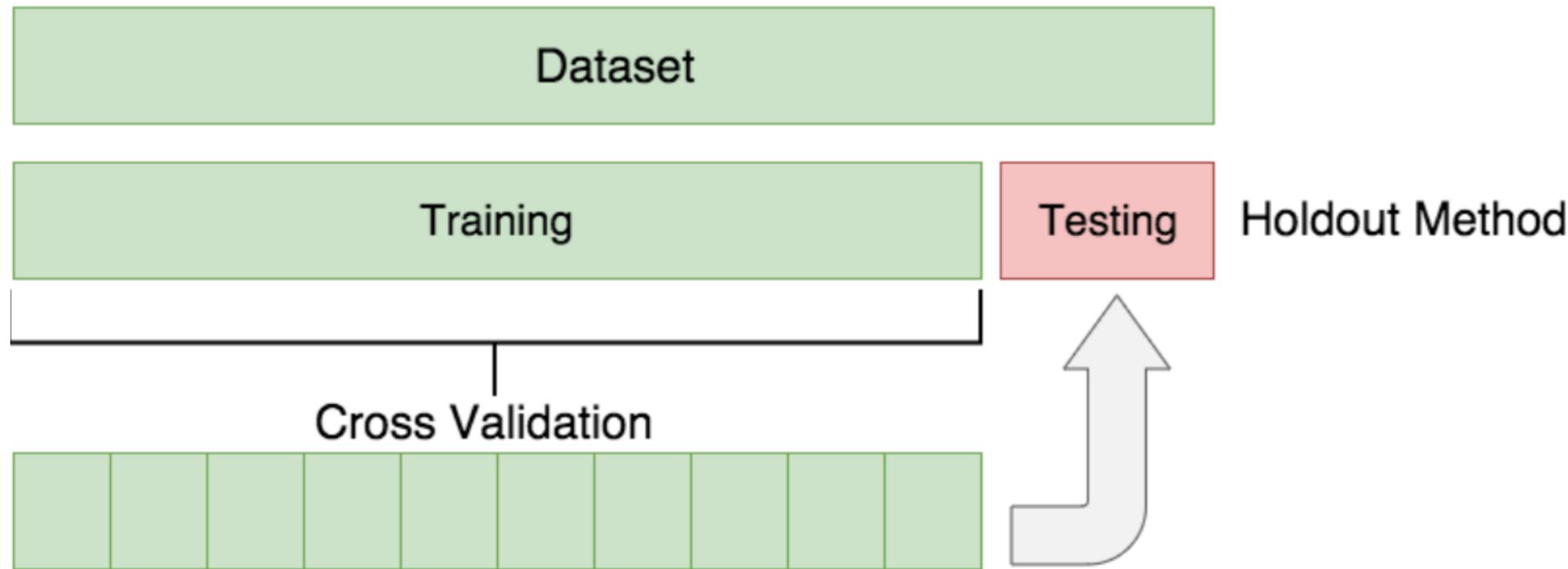
$$y > (-a/b)x - c/b$$



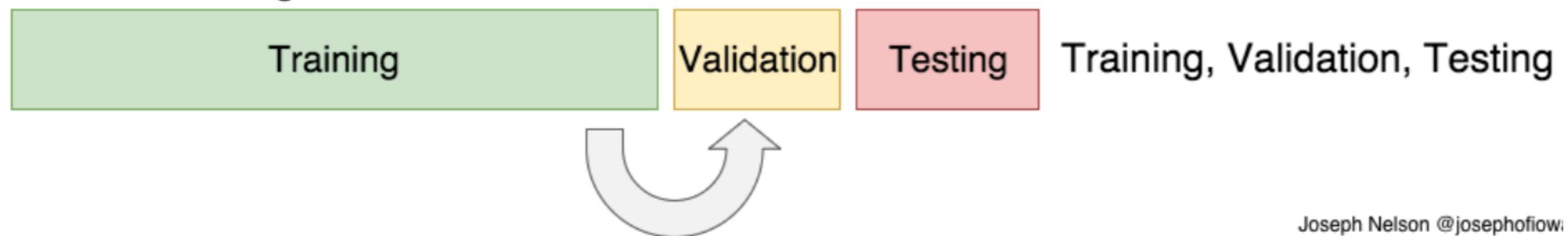
- 모델 동작이 얼마나 우수한지를 검증할 때는 훈련 데이터로 해서는 안되며 훈련에 사용하지 않은, 새로운 검증 데이터 (validation data)를 사용해야 한다.
- 보통 검증 데이터를 따로 제공하지 않으므로 훈련에 사용할 데이터의 일부를 검증용으로 미리 확보해야 한다.
- 훈련에 사용하지 않고 남겨 두었다가 모델이 제대로 동작하는지 테스트할 때 사용하는 데이터를 hold-out 데이터라고 한다.

훈련, 검증, 테스트 데이터

132



Data Permitting:



Joseph Nelson @josephoflow

- **훈련(Training)** 데이터 – 모델 파라미터를 훈련하는데 사용
- **검증(Validation)** 데이터 – 과대적합이나 과소적합을 검사하고 최적 모델 구조(하이퍼파라미터 등)를 찾는데 사용
- **테스트(Test)** 데이터 – 모델의 성능을 최종적으로 테스트하는데 사용

K-fold 교차검증

| 34



K-fold 교차검증

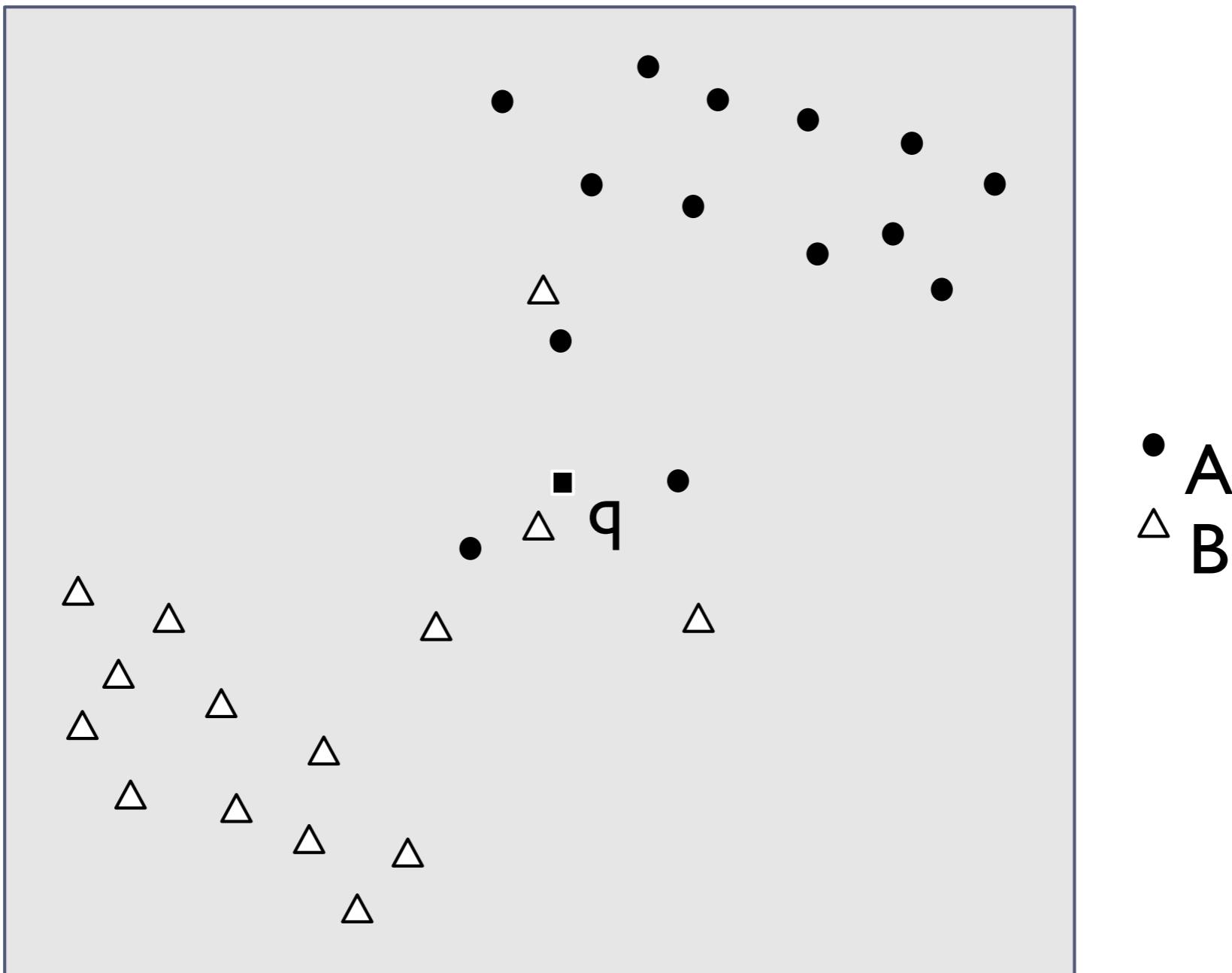
135

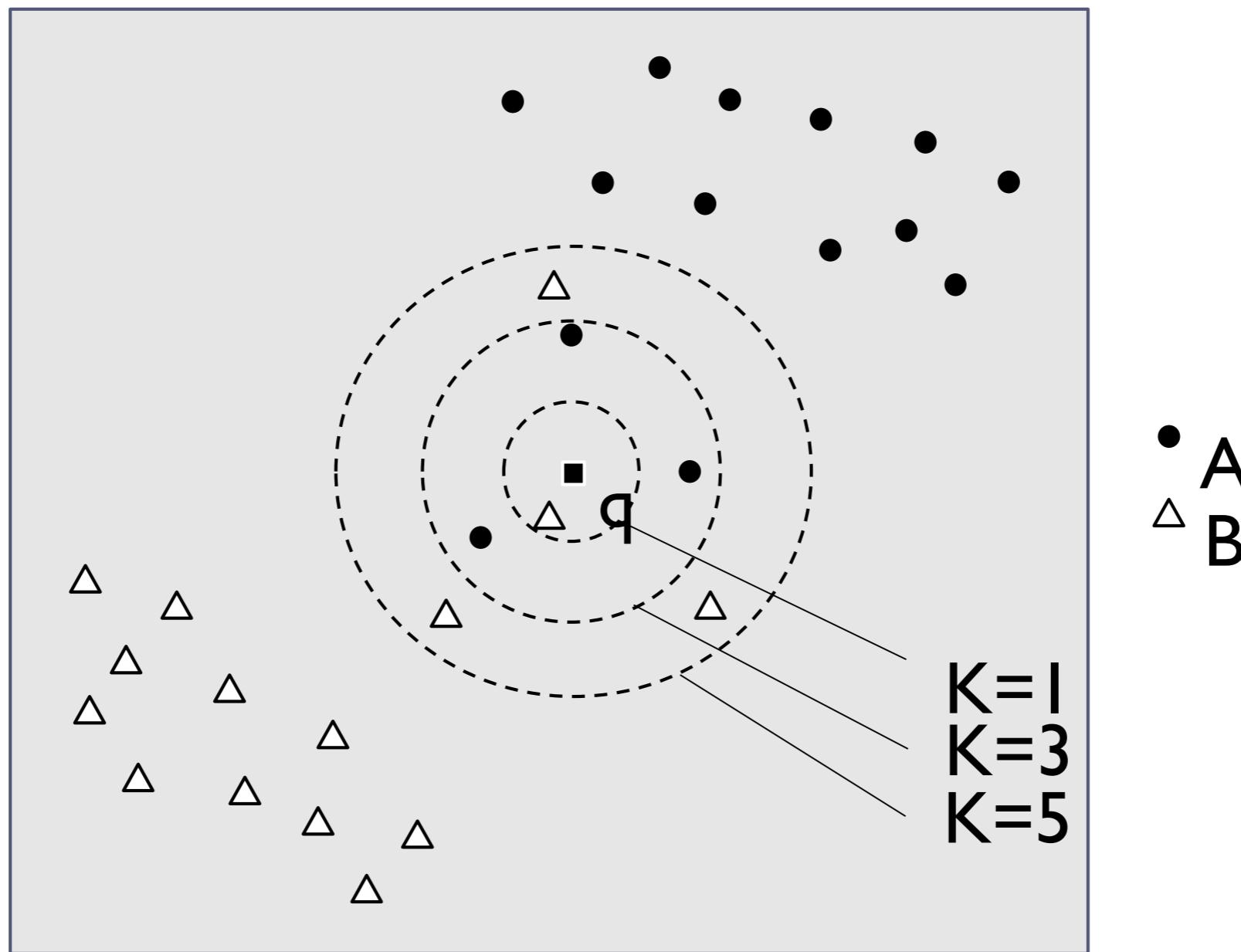
- 주어진 데이터 전체를 골고루 검증용으로 사용하여 모델의 동작을 보다 정교하게 확인하기 위함. 보통 K는 5~10을 주로 사용
- 사이킷런에서는 `cross_val_score()` 함수가 교차검증을 자동으로 수행하고 성능도 평가한다.
- 교차검증에서 주의할 것은 K개의 점수가 골고루 나오면 안정적인 모델이지만 만일 K개의 점수의 편차가 크면 모델이 불안정하다고 볼 수 있음. 검증 데이터를 어떤 부분을 택하는가에 따라서 모델의 성능 차이가 크게 나는 것은 모델이 불안정하다는 뜻.
- 교차검증을 수행하는 목적은 만든 모델이 잘 동작하는지 **성능을 검증하는 것이 목적**이지 모델 자체를 잘 학습시키는 것은 아니라 는 것을 주의해야 한다.
- 즉, 모델을 학습시키는 것은 모델선택, 알고리즘 선택, `fit` 등 별도의 작업에서 이루어지고 선택된 모델이 여러 가지 데이터 조건에도 잘 동작하는지 일반성을 갖는지를 확인하기 위해서 교차 검증을 하는 것이다.

k-NN(k-nearest neighbor) 알고리즘

136

- 주어진 샘플의 특성 값을 보고 가장 가까운 특성을 가지는 이웃(neighbor)을 k개 선택하고 이들 레이블의 평균치로 이 샘플이 속할 **분류**를 예측하는 방식
- kNN은 직관적으로 이해하기 쉬운 분류 알고리즘으로서 추천 시스템에서 많이 사용된다
 - 적절한 추천을 하기 위해서 추천을 요청한 사람의 성향을 특성들로 파악하고 그 사람과 가장 성향이 유사한 k명의 사람들이 좋아하는 품목을 추천하는 방식을 사용한다.
- kNN알고리즘을 협업 필터링(collaborative filtering)이라고도 부른다.

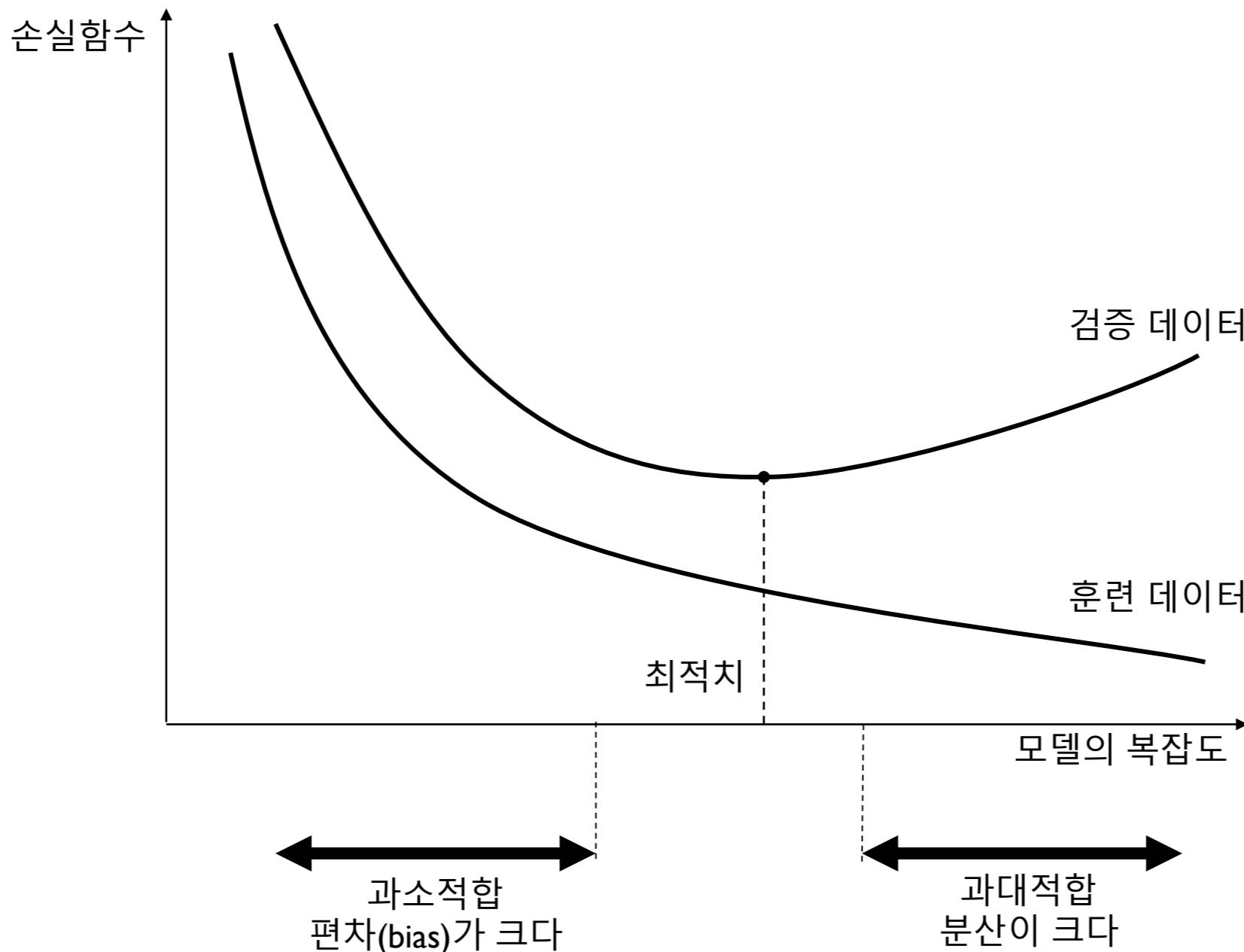




- k 값을 너무 작게 잡으면 주변 데이터에 너무 예민하게 반응하고 k 값을 너무 크게 잡으면 주변에 너무 많은 데이터의 평균치를 사용하므로 분류가 무뎌진다.
- 극단적으로 $k=N$ (전체 샘플 수)로 잡으면 항상 전체 데이터의 평균치 값을 예측하게 된다.
 - 영화 추천에서 $k=N$ 으로 한다면 이는 평균적으로 가장 많은 사람들 이 본 영화 즉, 종합 베스트셀러를 추천하는 것과 같다.
- k값을 작게 잡으면 노이즈에 민감하나 정확도는 올라가고 k를 크게 잡을수록 노이즈에 강하나 정밀한 예측이 어렵다.
- kNN의 단점은 훈련시간이 거의 없는 것에 비해 분류를 처리하는 시간, 즉 알고리즘을 수행하는 시간이 길다는 것이다. (확보한 샘플들을 모두 비교해서 어떤 그룹에 가까운지 새롭게 계산- 즉, 샘플이 추가될 때마다 이웃이 달라짐)
- 어떤 변수가 더 중요한지 파악이 어려움.

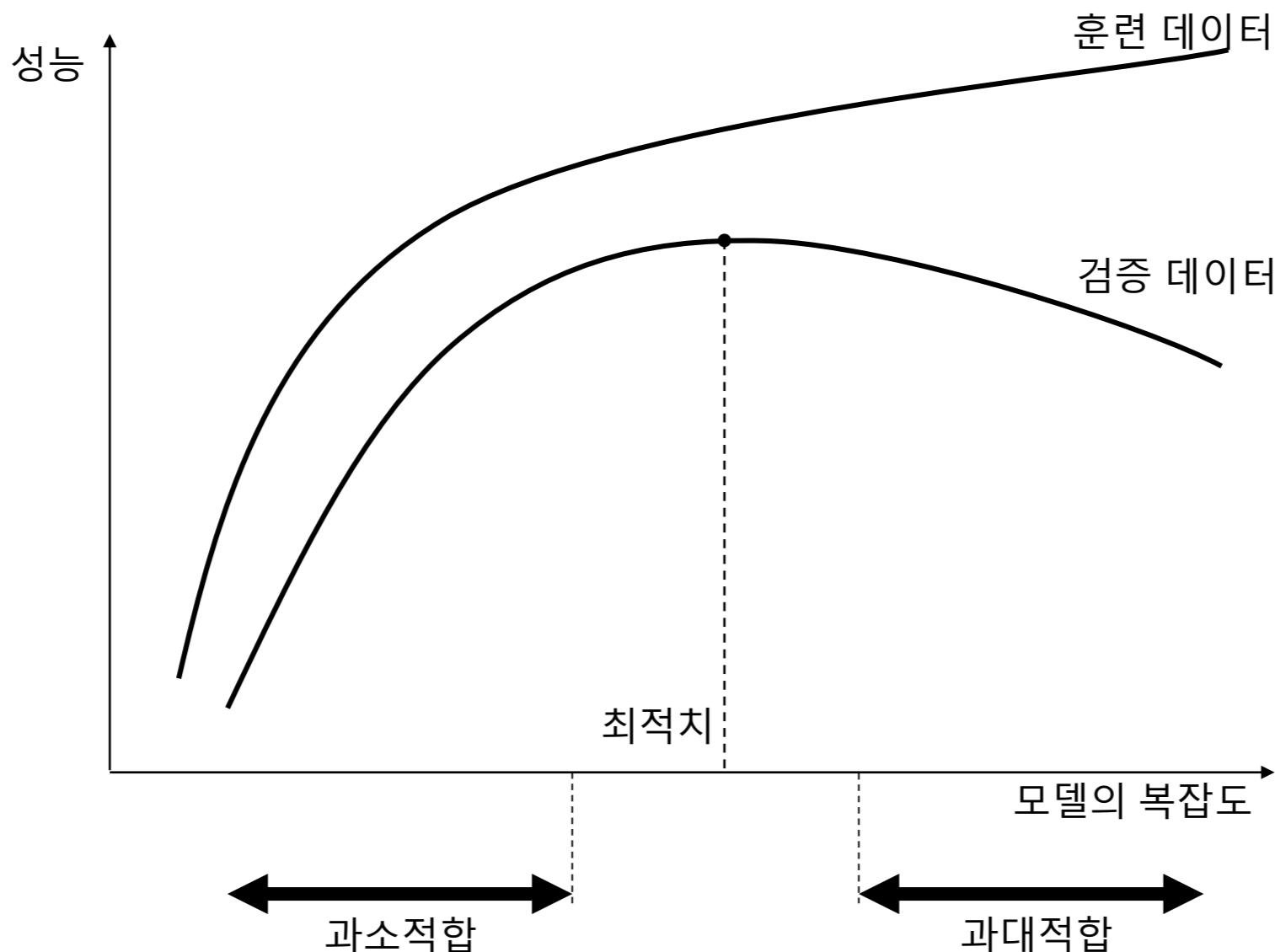
과대적합 과소적합 판단 - 손실함수

140



과대적합 과소적합 판단 - 성능

|4|



다중 분류 (Multinomial Classification)

| 42

- 분류 알고리즘은 기본적으로 이진 분류를 수행한다.
- 사이킷 런의 이진 분류 함수들을 다중 분류에 사용할 수 있는데 이는 내부적으로 이진 분류를 확장해서 수행한다.
 - One-versus-Rest (OvR) - 이진 분류 알고리즘을 여러번 적용하면 다중 분류를 수행할 수 있다.
- 분류 결과만 알려면 `predict()`를 사용하면 된다. 그러나 다중 클래스 각각에 해당할 점수 또는 확률을 알려면 `decision_function()` 또는 `predict_proba()`를 사용한다
 - `decision_function()`: using distance to the separating hyperplane
 - `pred_prob()`: (soft) classifier outputting probability of instance being in each class

결정 트리

결정 트리(Decision Tree)

| 44

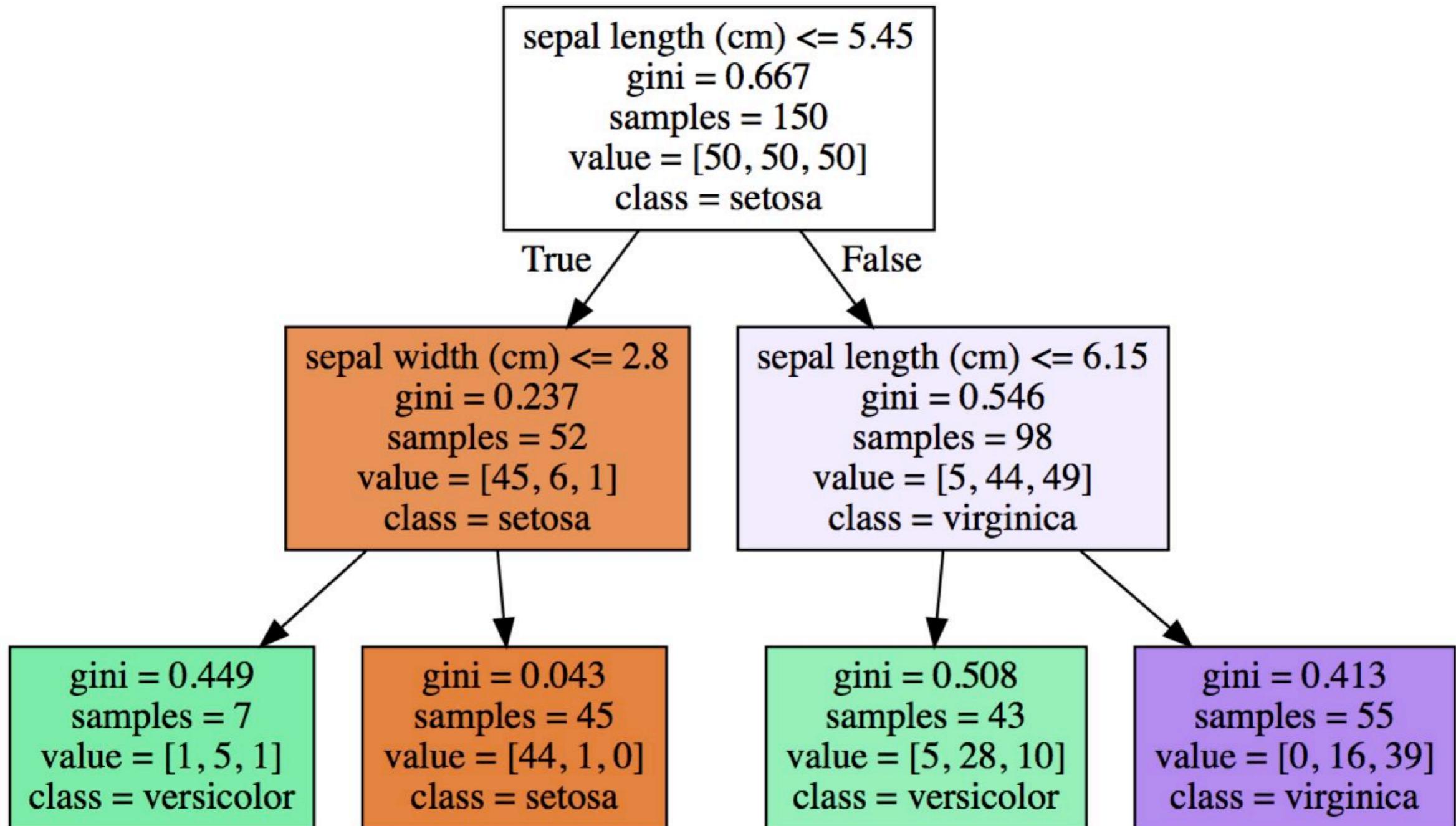
- 분류 기법 적용에서 가장 많이 사용하는 방법
- 분류 작업을 수행하기 위해 한번에 한 특성 변수씩 해석
- 결정 트리 모델을 사용하면 동작을 설명하기 수월함
 - 대출 거부 사유
 - 신용도가 낮은 이유
 - 불합격 사유 등

- 선형회귀 모델은 특성들을 대상으로 곱셈과 덧셈과 같은 연산을 하고 그 값을 기준으로 회귀나 분류를 예측했다.
- 결정 트리(decision tree)는 이와 달리 각 특성을 독립적으로 하나씩 검토하여 분류 작업을 수행한다.
 - 마치 스무고개 하여 예측을 하듯이 동작 한 번에 한 특성을 따져보는 방법이다.
- 결정 트리는 주로 분류와 회귀에 모두 사용된다.
 - 예를 들어 분류용 모델은 `DecisionTreeClassifier()`가 있고 회귀분석 모델로는 `DecisionTreeRegressor()`가 제공된다.

- 결정 트리에서 핵심이 되는 부분은 가장 효과적인 분류를 위해서 먼저 어떤 변수를 가지고 판별을 할지 결정하는 것이다.
- 이 판별은 트리를 내려가면서 계속되어야 하는데 매 단계마다 어떤 변수를 기준으로 분류를 하는 것이 가장 효과적인지를 찾아야 한다.
- 여기서 그룹을 효과적으로 “잘 나누는 것”의 기준은 그룹을 나눈 후에 생성되는 하위 그룹들에 가능하면 같은 종류의 아이템들이 모이는지를 기준으로 삼는다.
- 한 그룹에 같은 종류의 아이템이 많이 모일수록 순수(pure)하다고 하는데, 만일 나누어진 하위 그룹이 100% 같은 항목들로만 구성되면, 순도(purity)가 100%라고 한다.
- 최적 결정트리 학습은 NP-Complete

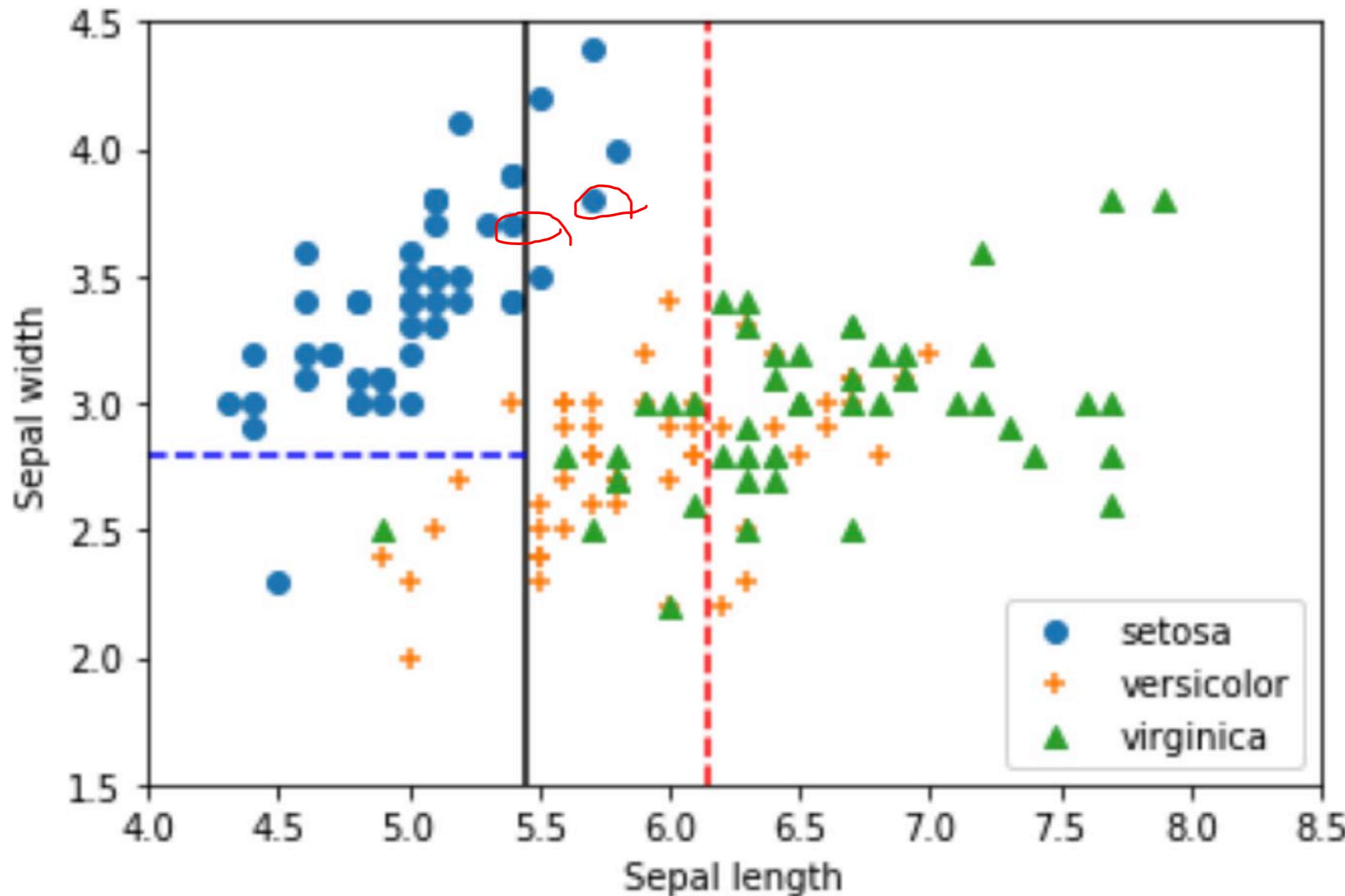
결정 트리 예 – iris dataset

147



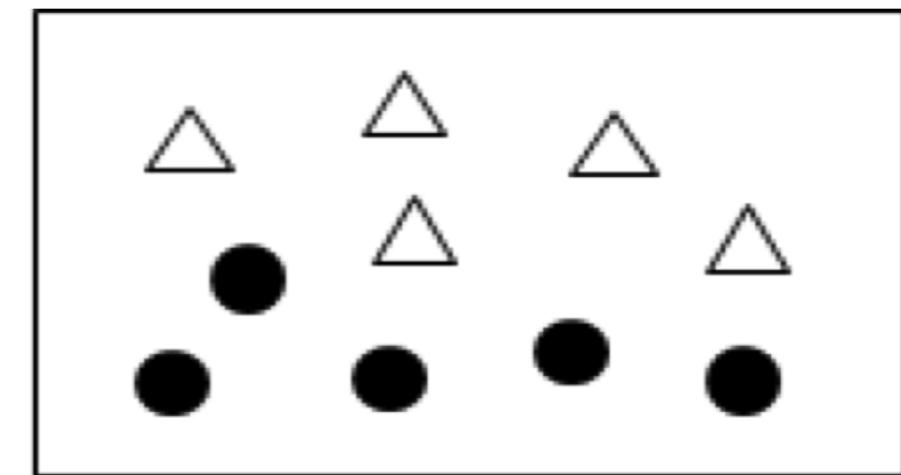
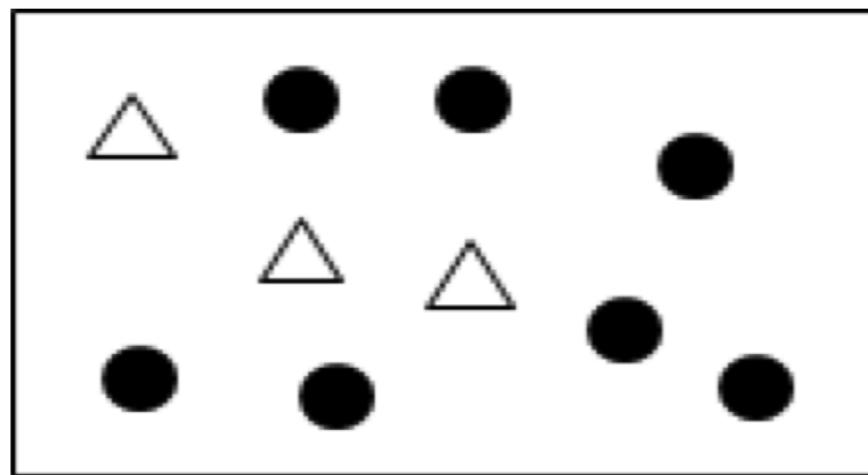
결정 트리 예

148



- 결정 트리는 나누어지는 그룹의 순도가 가장 높아지도록 그룹을 나누어야 한다.
- 그룹의 순도를 표현하는 데 **지니(Gini)** 계수 또는 **엔트로피(entropy)**가 주로 사용된다.
- Gini** 계수는 다음과 같이 정의된다.
 - 엔트로피(Entropy) 처럼 Log 를 씌우지 않고 확률값 그 자체를 사용 (불확실성이 가장 클때 0.5, 예측가능성이 가장 클때 0)

$$Gini = 1 - \sum_{k=1}^m p_k^2$$



$$\text{좌측 박스: 지니}(7:3) = 1 - \left[\left(\frac{7}{10} \right)^2 + \left(\frac{3}{10} \right)^2 \right] = 1 - (0.49 + 0.09) = 0.42$$

$$\text{우측 박스: 지니}(5:5) = 1 - \left[\left(\frac{5}{10} \right)^2 + \left(\frac{5}{10} \right)^2 \right] = 1 - (0.25 + 0.25) = 0.5$$

(*) Worst: 0.5
Best: Gini=0 (all in one class)

- 결정 트리를 계속 만들어 상세하게 분류를 하면 언젠가는 훈련 데이터에 대해서 100% 순도의 분류가 가능하다
- 이는 과대적합된 것이므로 테스트 데이터에 대해서는 성능이 오히려 떨어지게 된다.
- 결정 트리 모델은 트리를 만드는 깊이를 제한하지 않으면 과대적합할 위험이 높으므로 주의해야 한다.
- 한편 트리의 깊이를 적절한 값보다 너무 작게 제한하면 과소적합이 된다.

트리 종료 조건- 하이퍼 파라미터

152

- **max_depth**: 트리의 최대 깊이 (이보다 깊은 트리를 만들지 않는다)
- **max_leaf_nodes**: 리프 노드의 최대 수 (리프 노드를 이보다 많이 만들지 않는다)
- **min_samples_split**: 분할하기 위한 최소 샘플수 (이보다 작으면 분할하지 않는다)
- **min_samples_leaf**: 리프 노드에 포함될 최소 샘플수 (이보다 작은 노드는 만들지 않는다)
- **max_features**: 최대 특성수 (분할할 때 이보다 적은 수의 특성만 사용한다)

- 엔트로피(entropy)도 지니와 유사하게 순도를 나타내는데 사용된다.
- 한 노드(그룹)에 여러 클래스가 골고루 균일하게 섞여 있을 때는 엔트로피가 가장 높고, 동종의 클래스로 모여 있을수록 엔트로피가 낮다.
- 엔트로피의 정의는 아래와 같으며 앞과 같은 분류시의 엔트로피를 구하면 아래와 같다.

$$\text{Entropy} = - \sum_{k=1}^m p_k \log_2 (p_k)$$

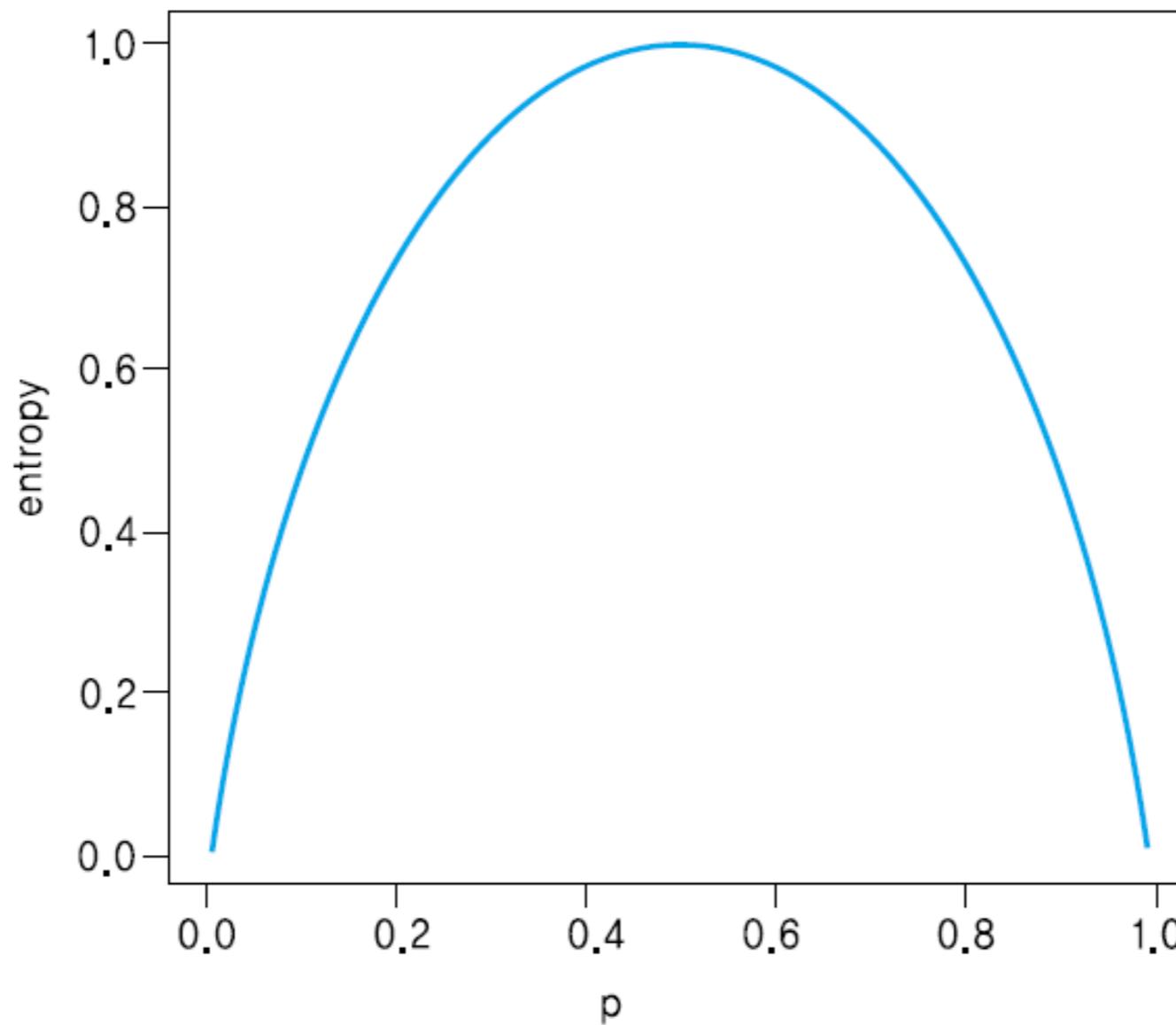
$$\text{엔트로피}(7:3) = -[0.7 * \log_2(0.7) + 0.3 * \log_2(0.3)] = -[(-0.36) + (-0.52)] = -(-0.88) = 0.88$$

$$\text{엔트로피}(5:5) = -[0.5 * \log_2(0.5) + 0.5 * \log_2(0.5)] = -[(-0.5) + (-0.5)] = -(-1) = 1$$

- 데이터(이벤트)가 포함하고 있는 정보의 총 기대치, 정보의 가치
- 정보량을 표현하기 위해 해당 사건이 발생할 확률 (probability)을 사용
- 사건이 발생 확률이 1이라면 정보가 주는 가치가 없고, 사건 발생 확률이 낮을수록 정보가 주는 가치가 높음
- 정보량은 일어날 확률의 역수에 비례
- 정보량 = $\log\left(\frac{1}{p}\right)$

- 정보량의 기대치란 어떤 사건이 갖는 가치와 그 사건이 발생 할 확률의 곱 - 이를 **엔트로피(entropy)**라고 함
- 엔트로피(정보량의 기대치)
 $= p \log(1/p) = -p \log(p)$

- 바이너리 사건의 경우 엔트로피는 p 가 0.5 일 때 가장 높음
- 즉, 불확실성이 가장 높을 때 엔트로피가 가장 높음



- 결정 트리 모델을 만든 후에, 어떤 특성이 결정 트리를 생성할 때 중요한 역할을 했는지 비중을 파악할 수 있다.
- 이 결과를 보고 중요하지 않은 특성은 향후에 제외하기도 한다.
- 내부 변수 *feature_importances_*에서 확인할 수 있다.

- 테스트 샘플이 속할 가장 확률이 높은 클래스 하나만 알려주는 것이 아니라, 이 샘플이 각 클래스에 속할 확률을 각각 알려주는 것이 가능하다.
- 예를 들어 소프트 투표(soft voting)를 도입하면 보다 정확한 다중 분류를 수행할 수 있다.
- 각 클래스에 속할 확률을 구하려면 *predict_proba()* 함수를 사용한다.

- 결정 트리는 거의 모든 종류의 분류에서 사용할 수 있는 범용 모델이다. 결정 트리의 가장 큰 장점은 알고리즘의 동작을 쉽게 남에게 설명할 수 있다는 것이다.
- 훈련데이터가 바뀌면 모델의 구조가 달라지는 단점이 있다.
- 결정 트리의 또 다른 장점은 특성 변수의 **스케일링이 필요 없다는 것이다**. 트리 모델에서는 변수간의 연산이 없기 때문이다.
 - 각 노드에서는 한 번에 한 특성을 검토하여 어떤 기준으로 트리를 나누면 순도가 올라 가는지만 점검하면 되므로 다른 특성 값과의 관계를 계산할 필요가 없다.

랜덤 포레스트

랜덤 포레스트(Random Forest)

|61

- 비교적 간단한 구조의 결정 트리들을 수십~수백개를 랜덤하게 만들고 각 결정 트리의 동작 결과의 평균치를 구하는 방법
- 이렇게 여러 개의 모델을 만들고 평균을 구하는 방식을 **앙상블(ensemble) 방법**이라고 하며 하나의 모델만 만드는 것보다 좋은 성능을 보인다.
- 주어진 훈련 데이터를 모두 한 번에 사용해서 하나의 최상의 트리 모델을 만드는 방식이 아니라, 데이터의 일부 또는 속성의 일부만 랜덤하게 채택하여 결정 트리를 다양하게 만들고 그 결과의 평균치를 취하는 방식이다.
- 나무(tree)가 많이 모였다는 의미로 숲(forest)라는 용어를 사용했다.

- 샘플도 랜덤하게 선택하고, 속성도 랜덤하게 선택하여 다수의 결정 트리를 만들고 이의 평균을 구하면 단일 결정 트리를 사용하는 것보다 안정적이고 우수한 성능을 낸다.
- 성능이 우수한 하나의 모델을 사용하는 것보다, 각각의 성능이 최상이 아니지만 다수의 모델을 사용하고 평균치를 구하는 방식이 더 우수하다
 - 이를 대중의 지혜, 큰 수의 법칙 등으로 설명하기도 한다.
- 랜덤 포레스트의 단점은 모델의 동작을 한가지 트리를 선택하여 설명하기가 어렵다는 것이다.
- 또한 계산량이 많아진다.

- 양상블 기법에서는 여러 개의 작은 모델의 평균 값을 구하거나 투표를 통하여 최적의 값을 찾는 절차가 필요하다. 이는 랜덤 포레스트 뿐 아니라, 다양한 양상블 기법에서 공통으로 필요하다.
- 먼저 여러 모델의 결과를 가지고 최적의 타겟 변수를 찾아내기 위해서 투표가 필요한 경우가 많은데 투표에는 직접 투표와 간접 투표가 있다.

간접 투표(soft voting)

| 64

- Hard Voting: 여러 개 분류기의 결과를 집계하여 가장 많은 표를 얻는 쪽으로 예측: Q (1 : 2)
- Soft Voting: 각 분류기의 예측 결과를 평균하여 확률이 가장 높은 쪽으로 예측: P (0.533 : 0.456)

	P일 확률	Q일 확률	판정결과 (직접투표)
세부 모델 A	0.9	0.1	P
세부 모델 B	0.4	0.6	Q
세부 모델 C	0.3	0.7	Q
확률의 평균 (간접 투표)	$(1.6)/3 = \textbf{0.533}$	$(1.4)/3 = 0.456$	P or Q

- 배깅이란 bootstrap aggregation의 줄인 말이며 전체 훈련 데이터에서 “중복을 허용”하여 데이터를 샘플링을 하는 방법이다.
 - 중복을 허용하므로 같은 데이터가 중복되어 선택될 수 있다.
 - **bootstrap resampling**의 줄임말로 부트스트래핑이라고도 부른다.
- 배깅과 달리 주어진 원래 데이터에서 중복을 허용하지 않고, 즉, 한 번 샘플링 된 것은 다음 샘플링에서 제외하는 방식은 **페이스팅(pasting)**이라고 한다.
- 배깅을 수행하면 학습에 선택되지 않는 샘플은 평균 37%가 되는데 이 샘플을 oob(out of bag) 샘플이라고 한다. 이 oob 데이터는 훈련에 사용되지 않았으므로 검증에 사용하기에 좋다
 - 결정 트리 구조에 배깅을 적용한 방식이 랜덤 포레스트 모델이다.

- 양상블 방법 중에 부스팅 알고리즘이 있다.
- 랜덤 포레스트와 달리, 간단한 결정 트리를 다수 만들어 각각 독립적으로 실행한 후에 이들을 평균하는 것이 아니라,
 - 앞의 모델을 보고 성능을 점차 개선하는 방식으로 동작한다.
 - 부스팅에는 아다부스트와 그라디언트 부스트가 널리 사용된다.
- 아다부스트(adaptive boosting)에서는 앞에서 사용한 세부 모델에서 과소적합했던 샘플, 즉 분류에 실패한 샘플의 가중치를 높여주는 것이다.
 - 즉, 소외되었던 샘플을 주목하여 학습을 다시 시키는 방식이다.

Labs

- gg-32 선형분류

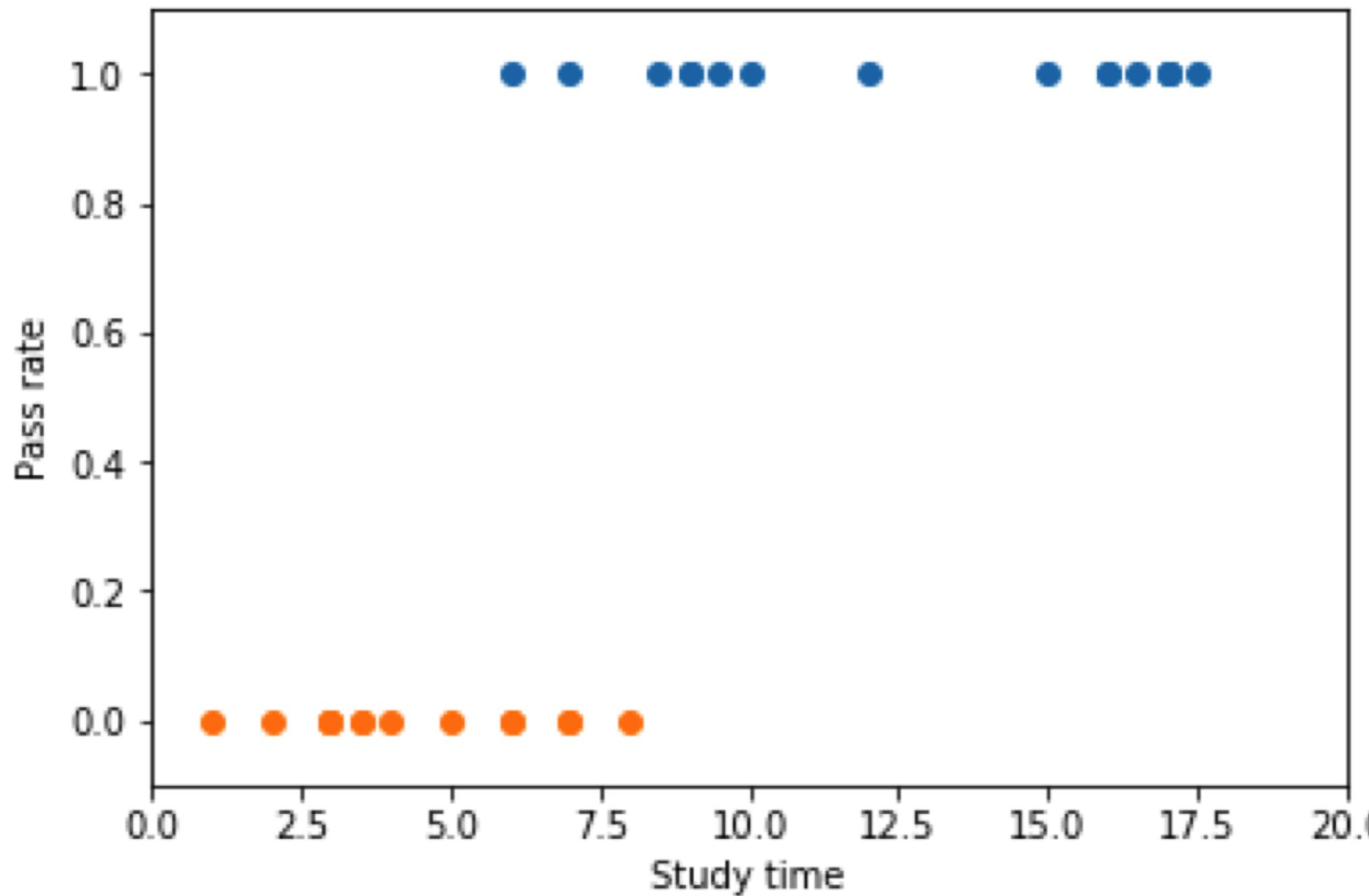
로지스틱 회귀

로지스틱 회귀분석(logistic regression)

- 임의의 범위를 갖는 값으로부터 0과 1사이의 값을 예측하거나 이진 분류에 사용하는 알고리즘이다.
- 로지스틱 회귀분석은 보통 독립 변수와 종속 변수의 관계를 S형 커브로 매핑함(선형 회귀분석 사용이 불가한 경우)
- 신용도 판단, 연간 구매량 기준 우수 고객 여부 판단, 평가 지표 기준 합격 여부 판단, 건강 지표에 따른 건강 여부, 팀의 승리/패배 여부 예측 등 여러 경우에 사용함

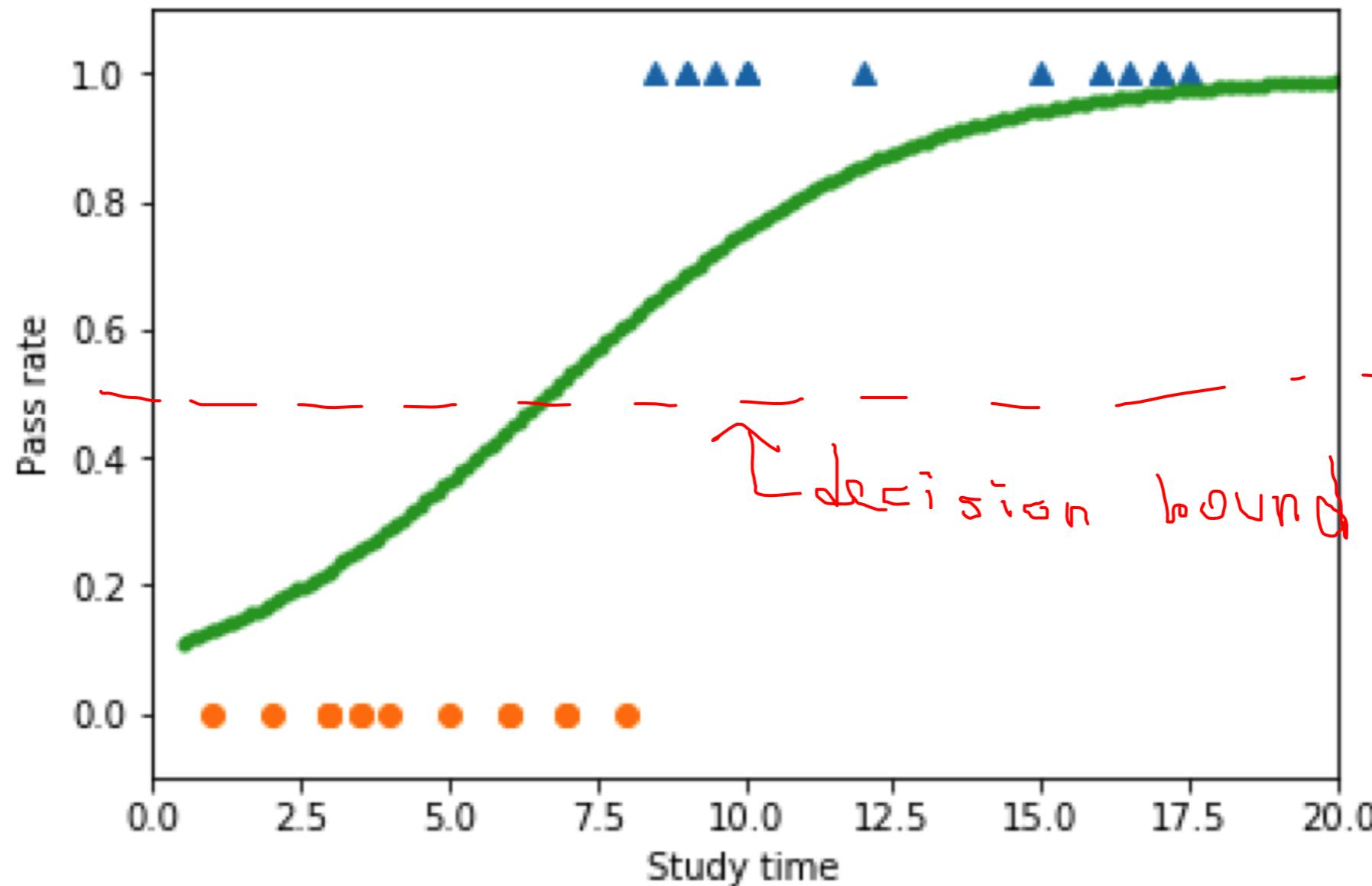
공부시간과 합격 여부

170



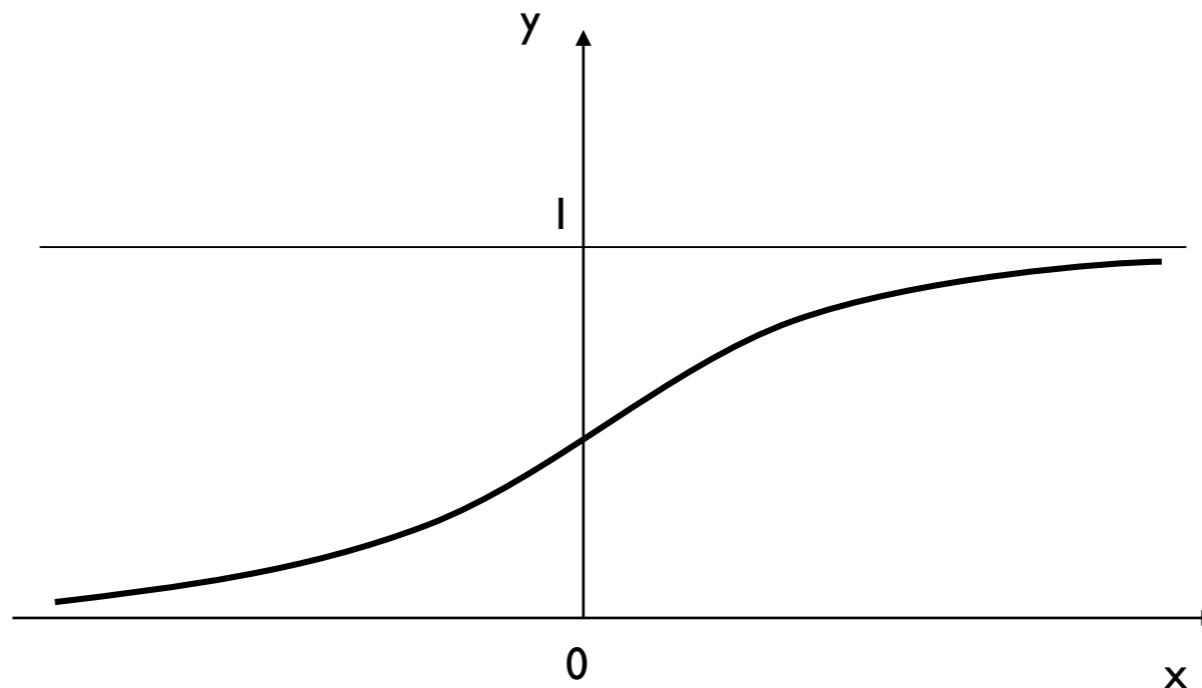
로지스틱 회귀 모델링

171



시그모이드 함수

172



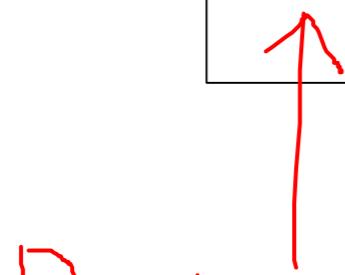
$$p = \frac{1}{1 + e^{-y}}$$

$$p = \frac{1}{1 + e^{-(ax + b)}}$$

a: 기울기 결정
b: 중간축의 위치

- 앞에서는 이진 분류, 즉 합격/불합격 등 두 개의 레이블을 가진 경우에 로지스틱 회귀를 사용하는 예를 소개했다.
- 그런데 2개가 아니라 3개 이상의 클래스 중에 하나를 예측해야 하는 경우는 **다항 로지스틱 회귀**(multinomial logistic regression)를 이용한다.
- 또는, **소프트맥스 (softmax)** 함수를 사용한다

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

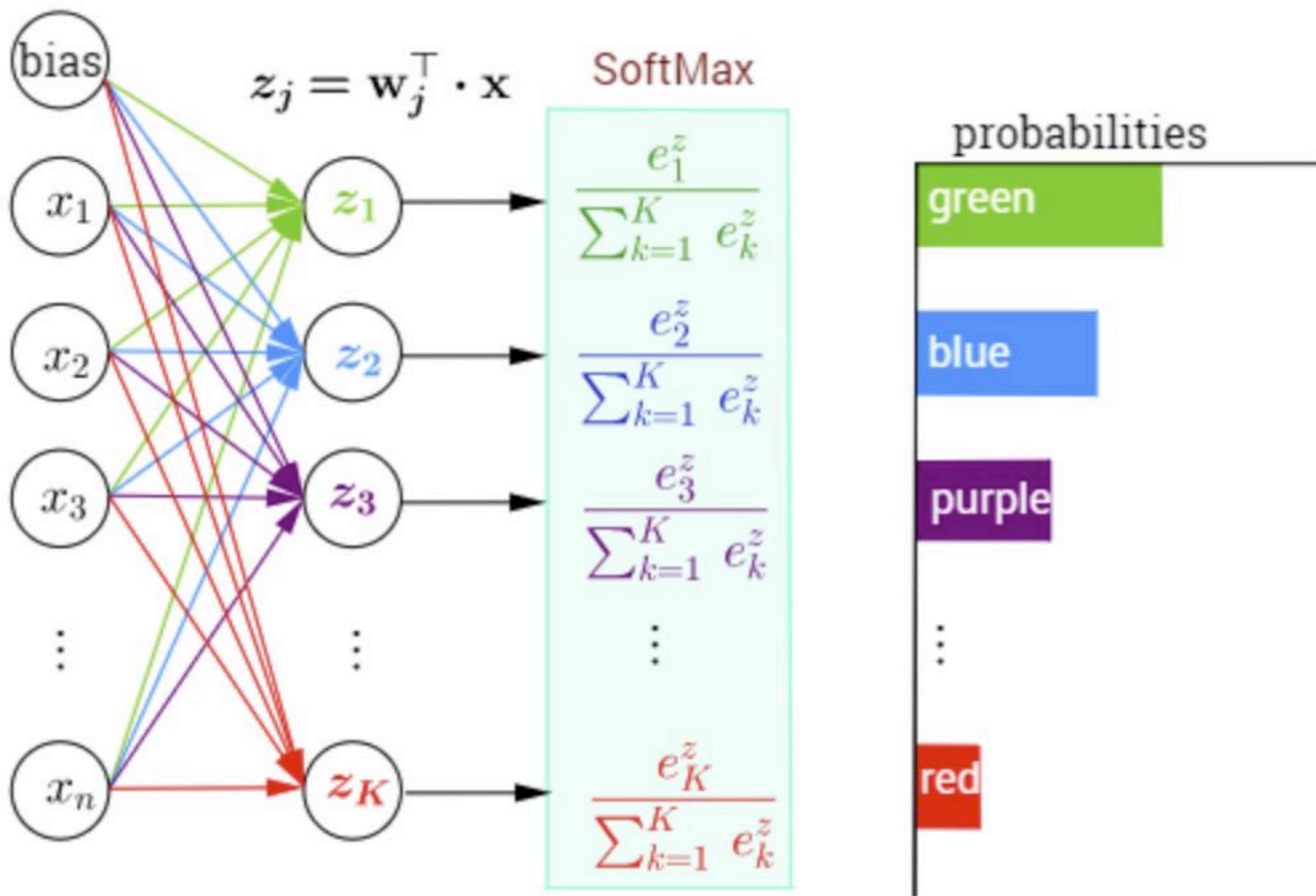


P(class = j)

소프트 맥스

174

- 상대적인 점수 비교를 확률처럼 0~1 사이 값으로 매핑



Labs

- gg-33 로지스틱회귀

Module2는 여기까지입니다.

모델 성능

- 분석 모델이 얼마나 잘 동작하는지의 성능 평가 기준 필요
 - 예측의 정확도
 - 동작 속도
- 데이터 분석 프로젝트의 종료 기준으로 사전 정의

컨퓨전 매트릭스(confusion matrix)

179

- 컨퓨전 매트릭스란 분류의 결과가 잘 맞았는지를 평가하는 채점표와 유사
- 결과 값이 P(Positive)또는 N(Negative) 둘 중 하나만 가질 수 있는 binary 예측의 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)

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

첫 번째 단어: 예측 평가	두 번째 단어: 추정 내용
True: 예측이 맞음	Positive: positive로 예측
False: 예측이 틀림	Negative: negative로 예측

- 정확도(accuracy): 정확하게 예측한 비율을 의미
 - $\text{accuracy} = (\text{TP} + \text{TN}) / \text{전체 경우의 수}(N)$

실제 \ 예측	암이라고 예측	암이 아니라고 예측	합계
실제 암환자	6 (TP)	4 (FN)	10
실제로 암환자 아님	2 (FP)	188 (TN)	190
합계	8	192	200

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

- recall과 precision의 두 가지 지표를 동시에 높이는 것은 어려움, F1은 이러한 두 요소를 동시에 반영한 새로운 지표임
- **F1은 recall과 precision의 조화평균(harmonic Mean).**
- $F1 = 2 \times \text{precision} \times \text{recall} / (\text{precision} + \text{recall})$
- 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 $c=2ab/(a+b)=0.54/1.2=0.45$

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

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

분류 순서 평가

184

- 알고리즘을 좀 더 세밀하게 평가하기 위해 분류결과뿐만 아니라 **분류한 순서를** 평가하는 방법.

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

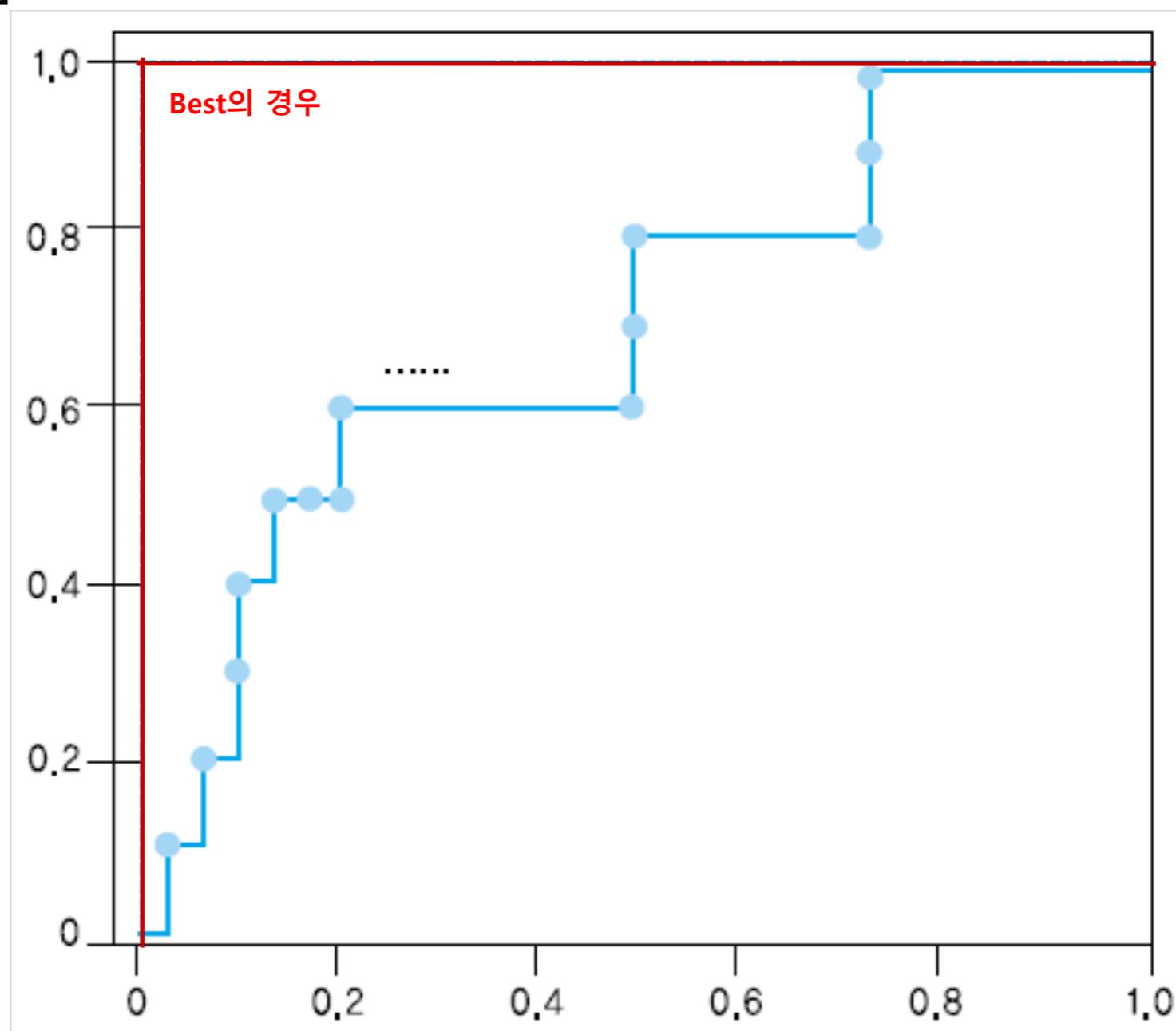
ROC(Receiver Operating Characteristic) 185

- 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 (120명 모두를 암환자라고 진단했다고 보고 생각하는 것이 편함)

ROC 그래프 예시

| 186

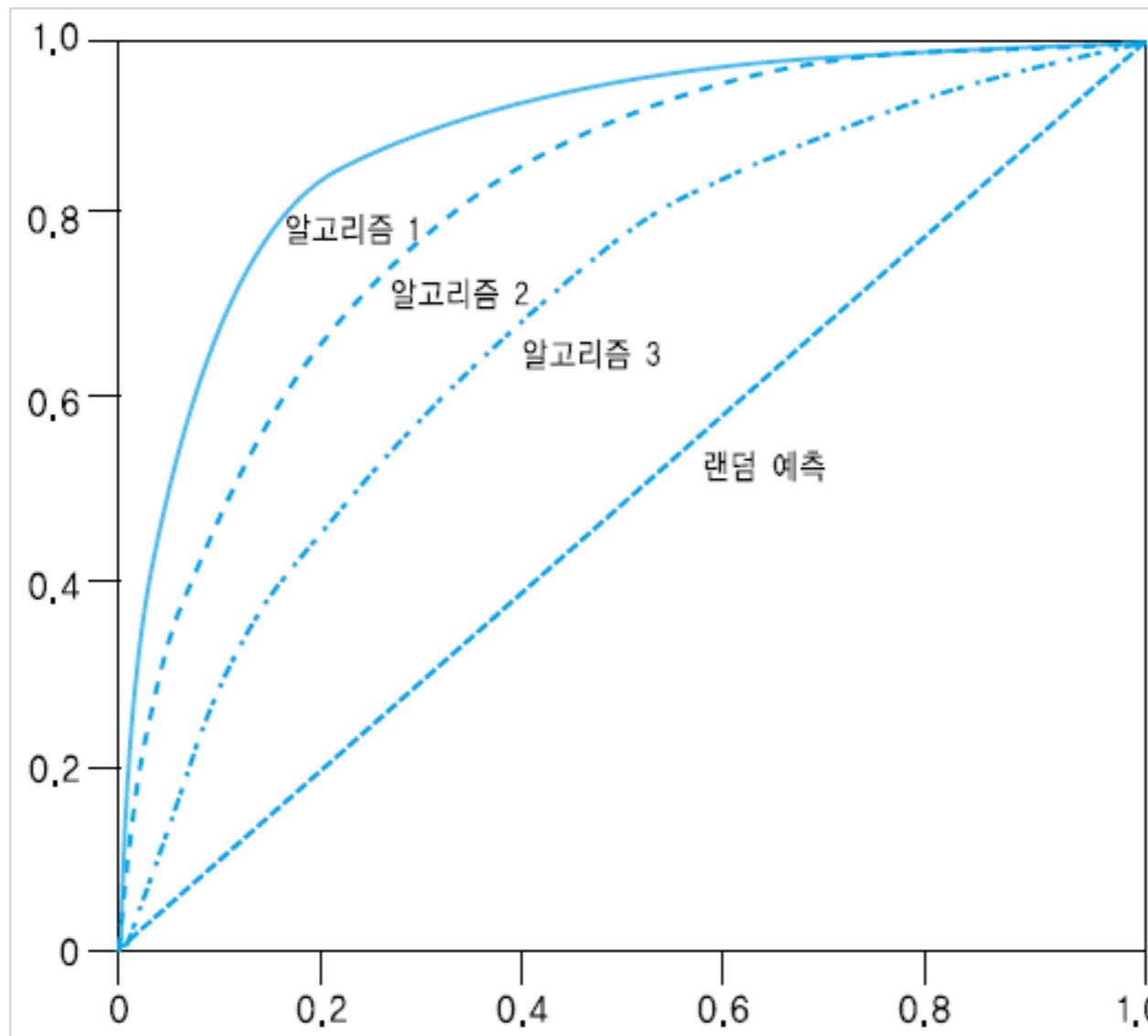
- 각 데이터 항목에 대해 계단형 그래프가 만들어짐
- 만일 의사의 예측 정확도가 높다면, 그래프가 초반에 위로 올라갈 것임



AUC(Area Under Curve)

187

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



- 분류의 결과가 어떤 이득(benefit)과 비용(cost)을 발생하는지를 따져보고 의사결정을 돋는 것이 데이터 분석의 목적이다.
- 분류 확률을 높이는 것 자체가 중요한 것이 아니라 분류 결과의 기대치를 구해봐야 한다. 암 진단의 경우 분류 결과에 따라 발생 할 수 있는 이득과 비용은 다음과 같이 생각해볼 수 있다
 - 1) 암이라고 예측했는데 실제로 암에 걸린 경우 - 환자는 치료를 계속 받고 병원은 수익을 낸다
 - 2) 암이라고 예측했는데 실제는 암에 걸리지 않은 경우 - 환자는 정밀검사를 받겠지만 암이 아닌 것에 안도한다
 - 3) 암이 아니라고 예측했는데 실제는 암인 경우 - 환자가 항의를 할 것이고 손해배상을 청구할 수도 있다
 - 4) 암이 아니라고 예측했는데 실제로도 암이 아닌 경우 - 환자는 진료가 정확하다고 믿고 다음에도 이 병원을 찾는다

- 1)의 경우 수익이 발생하며 이를 편의상 200만원이라고 하자 (E1),
- 2)의 경우 병원은 약간의 손실이 생기며 이를 -3만원이라고 하자(E2),
- 3)의 경우는 병원에 큰 손실이 생기며 이를 -500만원이라고 하자(E3),
- 4)의 경우 환자를 더 유치하므로 평균 이득이 2만원이라고 하자(E4)
- 이제 각 경우의 발생 확률과 각 이득 또는 비용을 곱하여 더하면 총 기대치를 구할 수 있으며 이를 계산하면 다음과 같다.

비용 최소화 (병원 사례)

190

■ 확률

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

■ 기대치

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

비용 최소화 (병원 사례)

191

- 전체 기대치는 위 두 테이블을 항목별로 곱한 후 더하면 된다.

$$\begin{aligned}\text{전체 기대치} &= p_1E_1 + p_2E_2 + p_3E_3 + p_4E_4 \\ &= (0.03 * 200) + (0.01 * (-3)) + (0.02 * (-500)) + (0.94 * 2) = -7.55 \text{만원}\end{aligned}$$

- 위의 의사는 암환자 진단 결과로 추가 수익을 내는 것이 아니라 오히려 병원에 손실을 발생시키고 있다.
- 이렇게 손실이 발생한 이유는 무엇일까? 실제는 암인데 암이 아니라고 잘 못 판정한 경우(즉, FN)의 댓가가 평균 -500만원으로 크기 때문이다.
 - 이 비용을 줄이려면 p_3 확률을 줄이도록 노력해야 한다. 조금만 의심이 들어도 모두 “암 같은데요”라고 판정해 주면 FN을 줄일 수 있다.
 - 500만원 비용보다 -3만원 손실이 훨씬 적기 때문이다.

비용 최소화 (병원 사례)

192

■ 보수적인 의사

실제 \ 예측	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	$p2 = 90/200=0.45$	$p4 = 100/200=0.5$

- 이 의사는 병원의 손실을 줄이고 이익이 나게 했다. 그러나 암 진단을 남발해서 실제로는 암이 없는 환자 90명이 재검사를 받게 되었다.

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

비용 최소화 (병원 사례)

193

- 이 의사의 진단 능력을 종합해 보면, 정확도(accuracy)는 200명 중 110명($10+100$)을 맞추었으므로 55%로 나쁜 편이다.
- 그러나 10명의 암환자를 모두 찾아냈으므로 재현률은 $10/10 = 100\%$ 가 된다. 앞의 의사의 경우 재현률(recall)이 60%였으므로 암환자를 찾아내는 비율이 매우 높아졌다
 - 오진으로 인해 병원이 지출할 비용도 줄여주었다.
- 그러나 정밀도(precision)는 $10 / (10+90) = 5\%$ 이며 이는 앞의 의사의 75%에서 크게 감소했다. 즉, 암이라고 진단해도 그 중에 5%만 암이고 나머지는 과잉 진단이다.
 - 과잉 진단은 장기적으로 병원과 의사의 신뢰도를 떨어뜨리게 된다.
- 판정 기준은 병원의 철학과 전체 비용에 따라 다르게 설정될 것이다.

다중 분류 (Multinomial Classification)

194

- 분류 알고리즘은 기본적으로 이진 분류를 수행한다.
- 사이킷 런의 이진 분류 함수들을 다중 분류에 사용할 수 있는데 이는 내부적으로 이진 분류를 확장해서 수행한다.
 - One-versus-Rest (OvR) - 이진 분류 알고리즘을 여러번 적용하면 다중 분류를 수행할 수 있다.
- 분류 결과만 알려면 `predict()`를 사용하면 된다. 그러나 다중 클래스 각각에 해당할 점수 또는 확률을 알려면 `decision_function()` 또는 `predict_proba()`를 사용한다
 - `decision_function()`: using distance to the separating hyperplane
 - `pred_prob()`: (soft) classifier outputting probability of instance being in each class

하이퍼파라미터 최적화

- 최적의 머신러닝 모델을 만드는 과정은 모델을 설계하고, 학습하고, 하이퍼파라미터 최적화 세 단계로 구성된다.
 - 모델 구조(알고리즘) 선택
 - 파라미터 학습
 - 모델 최적화 - 하이퍼 파라미터 선택
- 그런데 최적의 성능을 내는 모델을 완성하려면 모델의 구조를 구성하는 하이퍼파라미터의 최적값을 찾아야 한다.
- 이 과정에서 과대적합과 과소적합을 피하면서 성능평가지표를 최대로 하는 하이퍼 파라미터를 찾는다.

- 모델을 구성하는 환경 변수들을 하이퍼 파라미터라고 하는데, kNN 알고리즘에서 k값, 선형회귀에서 계수의 개수, 결정트리에서 트리의 깊이, 랜덤포레스트에서 특성 선택비율, SVC에서 규제화 변수 등을 말한다.
- 과대적합 등을 피하면서 즉, 모델을 일반화하면서 높은 성능을 내는지를 검증하는데 사용하는 데이터가 검증 데이터이다.
- 테스트 데이터는 모델을 최종적으로 만든 다음에 오로지 성능을 “평가”하기 위해서 사용되는 데이터인 반면, 검증 데이터는 모델을 만드는 “과정”에서 하이퍼 파라미터를 최적화하는데 사용된다.
- 검증에서는 대부분 교차검증을 수행한다.

- 하이퍼 파라미터가 가질 수 있는 전체 범위를 몇 개의 구간으로 나누어 일일이 하나씩 점검해보는 방식이다.
 - 예를 들어, SVC 모델을 사용할 때 gamma 변수와 C 변수의 값을 각각 5 가지, 4가지로 나누고 총 $4 \times 5 = 20$ 가지 경우를 시도한다.
 - 최고의 score를 얻는 gamma와 C 값을 찾는다.
- 하이퍼 파라미터 예시
 - kNN: k값
 - 결정 트리: 트리의 깊이, 분류 조건, 분류를 위한 최소 샘플수
 - SVM: 커널 타입, 커널 계수, 규제화 파라미터(감마, C)
 - 랜덤포레스트: 트리수, 사용할 특성수, 분리 조건, 분류할 최소 샘플수
 - 그라디언트 부스팅: 트리수, 학습률, 트리깊이, 분할할 조건, 분류할 최소 샘플 수

- 과대적합을 피하기 위해서 수행하는 규제화용 파라미터도 하이퍼 파라미터이다.
 - 커널 SVM에서 감마 파라미터를 조절했었다. 신경망에서는 드롭아웃 등이다.
- GridSearchCV() 함수를 사용하면 그리드 탐색을 하며 동시에 교차검증을 수행하여 예상되는 성능을 측정하기에 편리하다.
 - GridSearchCV()로 생성한 모델 객체는 fit, predict, score 함수를 제공하며 fit을 호출할 때, 여러 파라미터 조합에 대해서 교차검증을 수행한다.

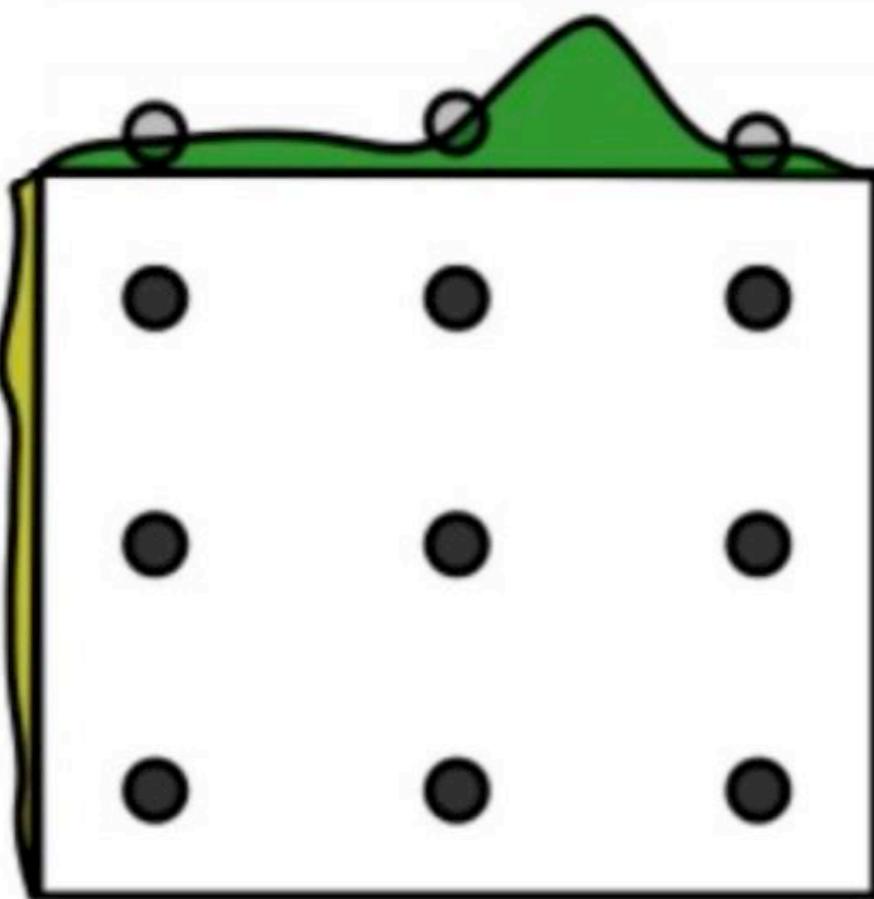
- 그리드 탐색은 단순하나 여러 경우의 수를 모두 탐색하는데 시간도 오래 걸리고 최적의 값을 놓치는 경우가 있다.
 - 격자 모양의 파라미터 조합의 값 사이에 실제로 최적의 하이퍼 파라미터 값이 있었다면 이를 찾아낼 방법이 없다.
- 랜덤 탐색은 두 단계로 이루어진다.
 - 일정한 범위 내에서 랜덤하게 하이퍼 파라미터를 선택하여 성능을 실험하여 대체로 어떤 영역에서 성능이 좋은지를 찾는다.
 - 다음에는 이 영역을 중심으로 세밀하게 탐색을 한다.
 - RandomizedSearchCV()함수를 사용한다.

랜덤 탐색의 특징

201

Grid Layout

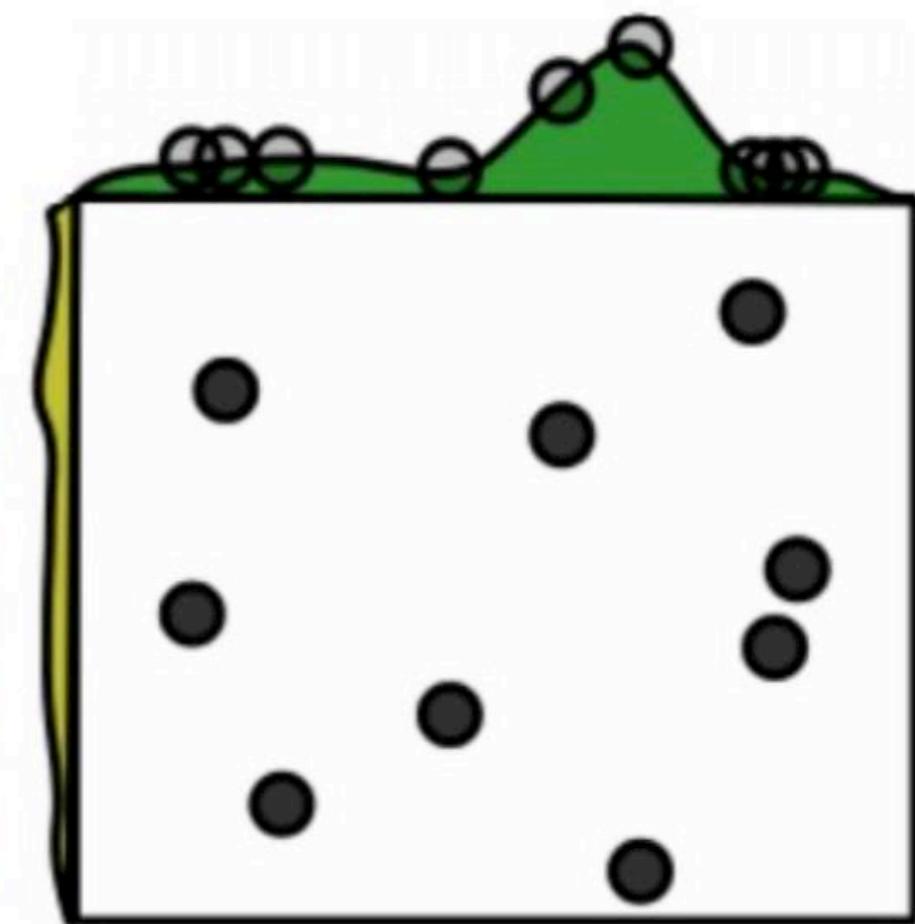
Unimportant parameter



Important parameter

Random Layout

Unimportant parameter



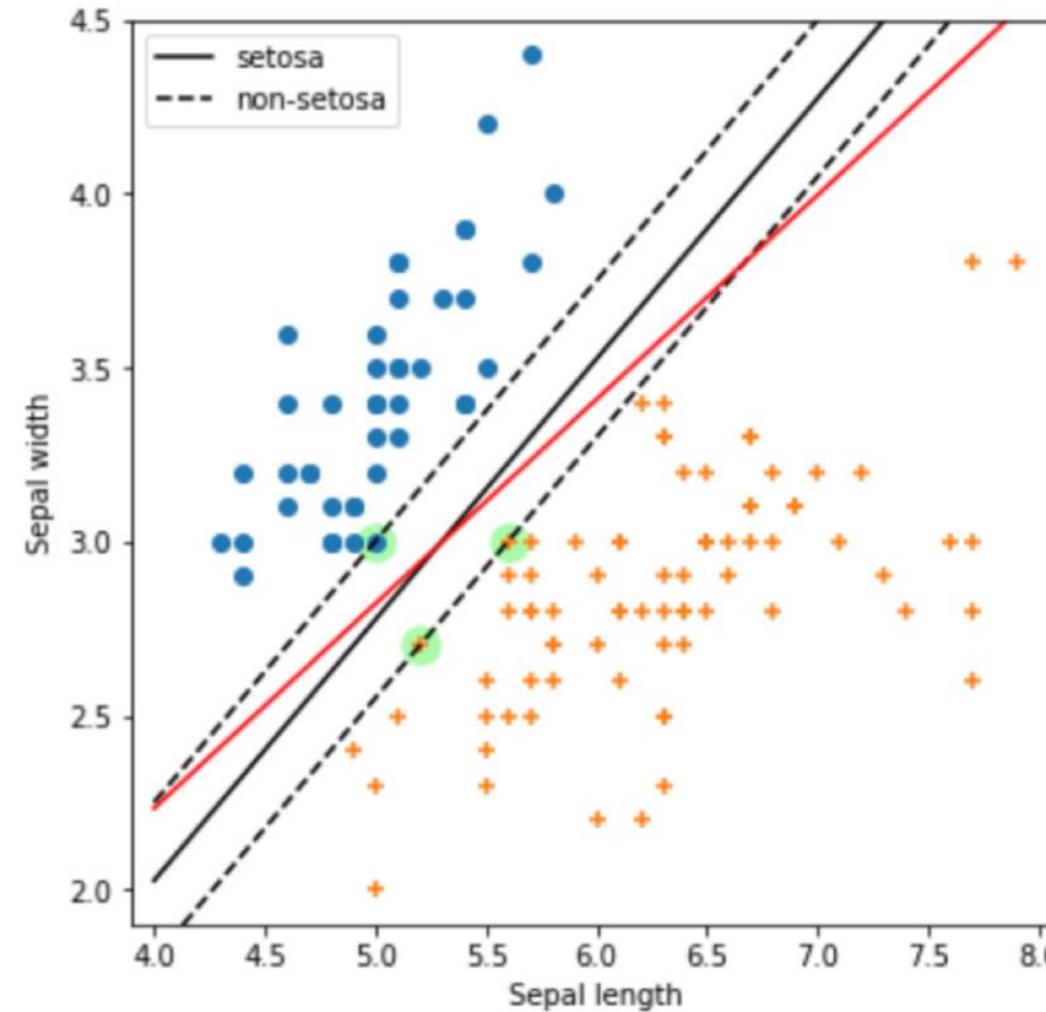
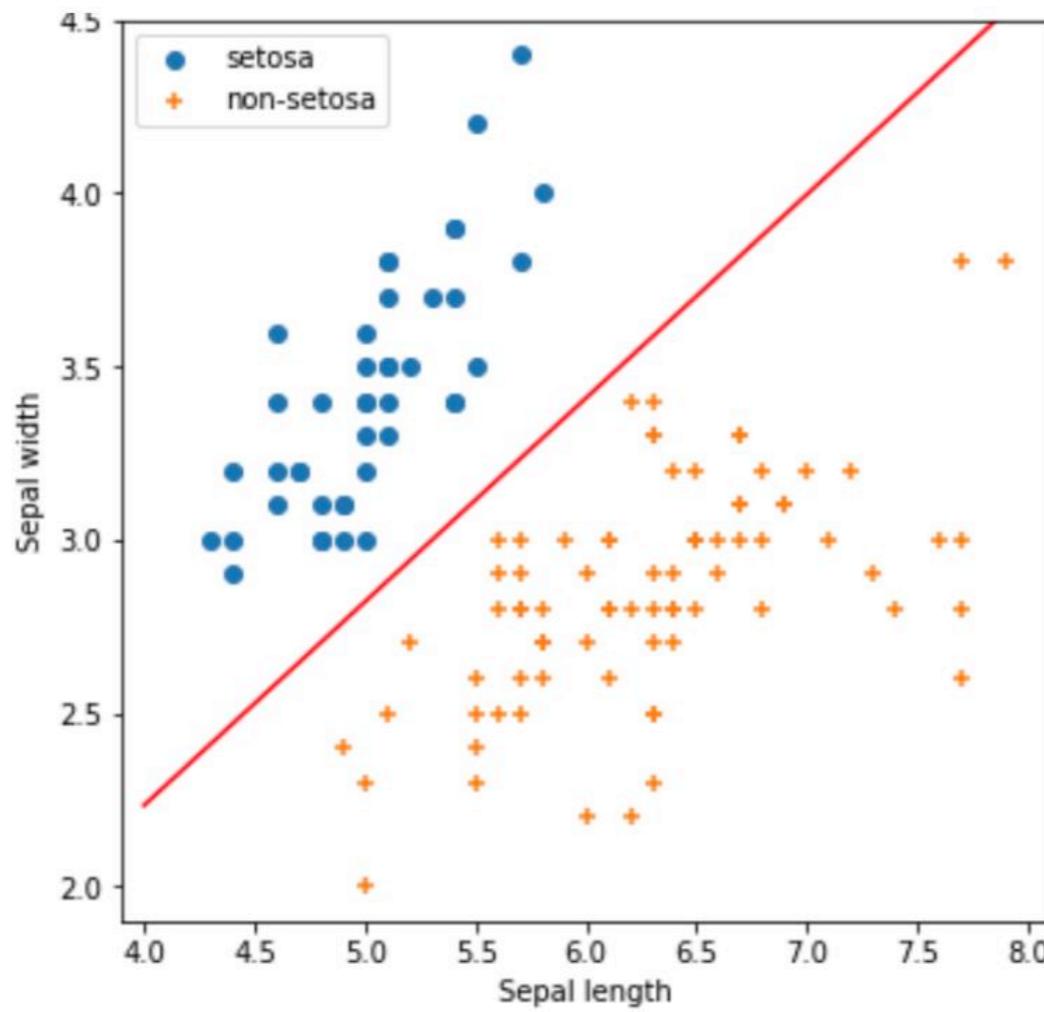
Important parameter

SVM

SVM (Support Vector Machine)

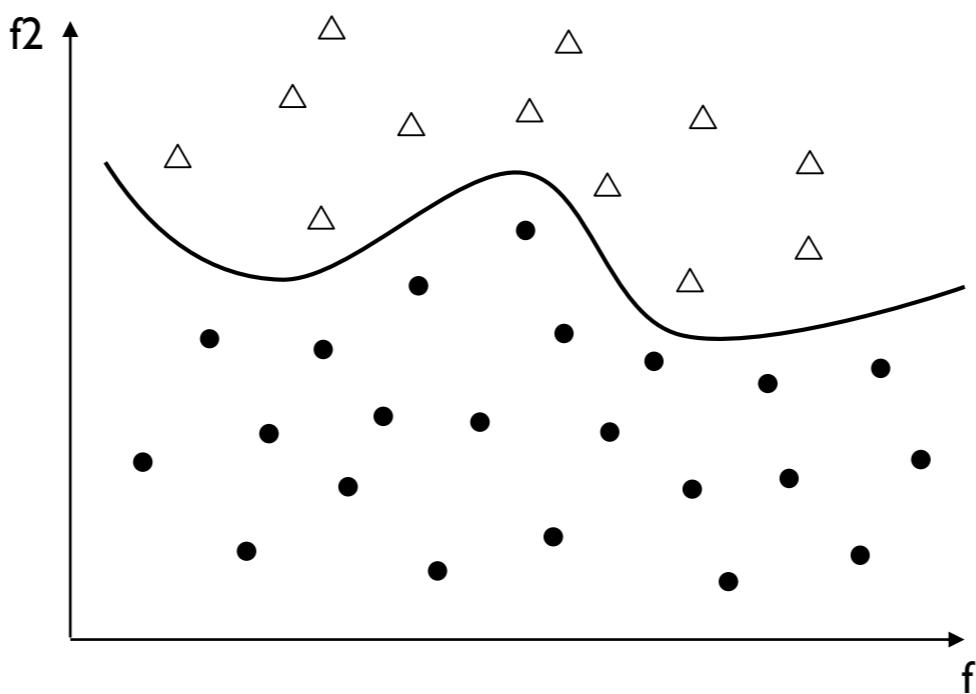
203

- 선형 모델의 성능을 개선하였다.
- 분류시에 경계면을 가능한 일반화 하는 것이다.
- 샘플들을 단순히 나누기만 하는 것이 아니라 가능한 거리를 멀리 나눌 수 있는 경계면을 찾는 작업을 한다.



- 선형 분류 직선이 분류를 수행하는데 문제가 없어 보인다. 그러나 이러한 경계면은 새로운 샘플 데이터에 대해서는 잘 동작하지 않을 것이다.
- 두께가 없는 선으로 경계를 만드는 것이 아니라 우측 그림의 점선으로 이루어진 두께가 있는 굽은 경계면을 만들고 그 두꺼운 경계면의 중앙을 지나는 선을 선택
- 더 일반적인, 안정적인 경계선을 얻을 수 있으며 과대적합을 피하게 된다.
- (주의) SVM은 선형 모델과 마찬가지로 속성에 계수를 곱하고 덧셈을 하는 연산에 기반하므로 여러 속성을 함께 사용하려면 반드시 스케일링을 해야 한다.

- 특성을 그대로 사용하지 않고 이의 2승, 3승, 4승 등 고차원의 속성을 내부적으로 만들어서 사용하는 방식



베이즈 알고리즘

베이스(Bayes) 이론

- 독립적인 두 사건 A와 B가 있을 때
- 사건 A가 발생했다는 조건에서 사건 B가 발생할 확률

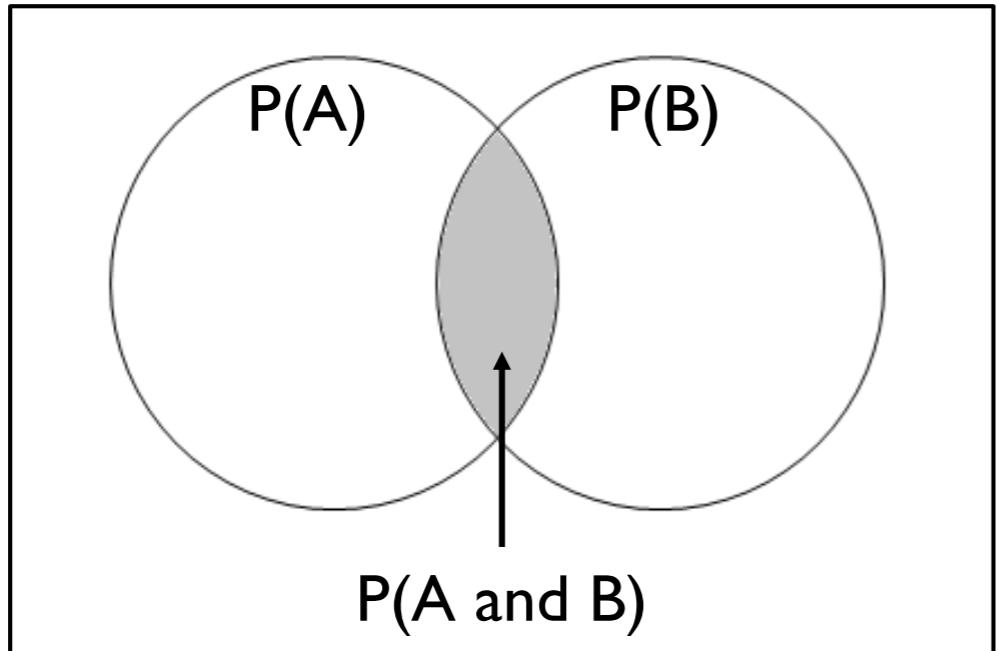
$$P(B|A) = \frac{P(A|B)P(B)}{P(A)} = \frac{P(A,B)}{P(A)}$$

- 사건 B가 발생했다는 조건에서 사건 A가 발생할 확률

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

- $P(A,B)$ 로 정리하면 $P(B|A)P(A) = P(A|B)P(B)$
 - 수식에서 3개의 확률을 알면 나머지 하나는 구할 수 있음

- 은조건부 확률 이론을 이용하여 새로운 사건의 조건부 확률을 예측하는 방법
 - 베이즈 이론은 단순하지만 매우 강력한 이론이다.



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

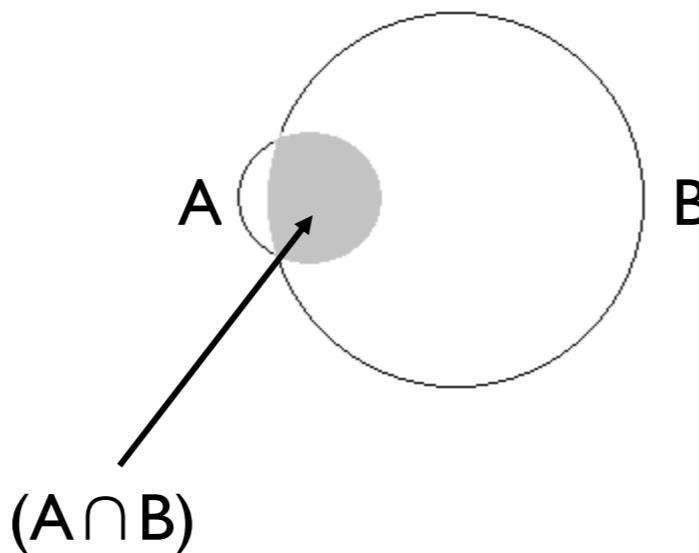
$$P(A|B) = \frac{P(A,B)}{P(B)}$$

$$P(B|A)P(A) = P(A|B)P(B)$$

- 세 개의 확률을 알면 나머지 한 가지 확률을 구할 수 있다.

베이스 이론의 예

- 사건 A는 피부암에 걸릴 확률을 나타내고 사건 B는 붉은 반점이 생길 확률이라고 하자. 그리고 피부암에 걸리면 붉은 반점이 생길 확률이 0.99라고 하자.
 - 피부암 발생 확률 $P(A) = 0.0001$ (1만명 중에 1명)
 - 붉은 반점이 나타난 확률 $P(B) = 0.01$ (1만명 중에 100명)
- 붉은 반점이 나타났다는 조건하에 내가 피부암일 확률
$$P(A|B) = P(B|A)P(A)/P(B) = (0.99)*(0.0001)/0.01 = 0.01$$



- 베이즈 알고리즘은 이와 같이 미리 파악한 사전 확률들을 기초로 어떤 새로운 사건의 조건부 확률을 예측하게 해준다
 - 주는 간결하지만 명확한 동작을 한다
- 베이즈 알고리즘은 의학분야에서 널리 사용되며 (여러 증상을 보고 병을 진단할 때 등),
- 스팸메일 검출에도 사용된다

- 수신한 메일이 스팸인 사건을 A라고 하자. 메일이 스팸일 확률이 0.01이라고 하면 즉 100통의 메일 중에 하나가 스팸이면,
 - $P(A) = 0.01$ 이다
- “쿠폰”이라는 단어 하나만 보고 스팸인지 아닌지를 판단하는 간단한 알고리즘을 생각해 보자. 메일에 “쿠폰”이 들어 있을 확률이 과거 통계로부터 평균 0.003이라고 하자.
- 전체 메일중에 ‘쿠폰’ 단어가 들어있는 사건을 B라고 하면
 - $P(B) = 0.003$ 이다.
- 전체 스팸 중에 쿠폰을 포함한 메일이 10%였다고 하면 이는 다음과 같이 표현된다.

$$P(B|A) = 0.1$$

- 이제 새로운 메일이 하나 도착했는데 여기에 “쿠폰” 단어가 포함되어 있다면 이 메일이 스팸일 확률은 얼마일까?
- 베이즈 이론에 따라 이 값을 다음과 같이 구할 수 있다.

$$\begin{aligned} P(A|B) &= P(B|A)P(A)/P(B) \\ &= (0.1)(0.01)/(0.003) \\ &= 0.333 \end{aligned}$$

스팸 메일 필터링

213

		“쿠폰”		
		있음	없음	합
스팸	1	9	10	
정상 메일	2	988	990	
합	3	997	1000	

- 문석 대상인 1000 개의 메일 중에 스팸의 송 수가 10개 이고 따라서 스팸 메일이 발생할 확률은 $P(A)=0.01$ 이다.
- 모든 메일 중에 “쿠폰” 단어가 들어간 메일은 총 3개이므로 $P(B)=0.003$ 이다.
- 스팸 메일 10개만을 놓고 볼 때 쿠폰 단어가 들어간 것은 1개이고 9개에는 쿠폰 단어가 없었다. 즉, 10%의 스팸 메일에 쿠폰이 들어 있었다.
- 새로운 메일에 쿠폰이라는 단어가 들어 있었다. 이 메일이 스팸일 확률은
$$P(A|B) = (1)/(1+2) = 1/3 = 0.3333$$

- 과거에 스팸에 자주 들어 있었던 단어들, 예를 들면 할인, 급매, 비아그라, 판매, 당첨 등과 같은 단어들을 동시에 고려한다면 스팸을 찾아낼 확률이 높아질 것이다.
- 그런데 여러 개의 사건이 결합된 경우 조건부 확률 식은 매우 복잡해진다. 왜냐하면 각 단어의 발생들 간에도 서로 조건부 확률이 있을 것이고 이들을 고려하는 알고리즘은 구현하기가 매우 복잡해진다.
- 예를 들어 '쿠폰'과 '할인'이라는 단어는 서로 독립적인 사건이 아니며 같이 발생할 확률이 높다.
- 이렇게 복잡해지는 문제를 단순화 한 알고리즘으로 베이즈 이론을 단순화하며 확장한 나이브 베이즈 (Naive Bayes, NB) 알고리즘이 널리 사용된다.

- 나이브(naive)라는 단어를 사용한 이유는, 분류에 고려한 여러 특성변수들의 서로 독립적이라는 “순진한” 가정을 하기 때문이다.
- 이들 단어의 발생이 독립적이지 않지만 서로 독립이라고 가정하여도 상대적인 확률을 구하는 것은 가능하다.
- 나이브 베이즈로 추정한 값은 0~1 사이의 값을 갖지만 이는 수학적으로 정확한 확률값을 구한 것이 아니라 상대적인 점수(score)를 구한 것이다.
- 일정한 점수 이상의 메일을 스팸으로 처리하고 오차가 발생하면 조정하는 방식으로 학습하면 되기 때문이다.
- NB 알고리즘은 동작이 매우 단순하고 처리속도도 빠르며 평소의 통계를 기반으로 확률 값만 구해두면 된다.
- NB 알고리즘은 특정 단어가 들어 있는지를 파악해야 하는 블로그 분석이나 트위터 분석 등 텍스트 분석에서 널리 사용되며 거의 모든 데이터 분석에서 사용할 수 있다.

단어 임베딩

- 앞에서 소개한 세 가지 텍스트 코딩 방식인 원핫 인코딩, BOW(단어모음), 문서-단어 행렬방식은 단어마다 고유번호를 배정하여 사용하였다.
- 그러나 이 고유 번호 숫자에는 아무런 의미가 들어 있지 못하며 단지 인덱스의 성격만 갖는다.
- 단어를 인덱싱이 아니라, 의미 있는 숫자들의 집합, 즉, 벡터로 표현하는 방법이 단어 임베딩 (Word Embedding)이다.

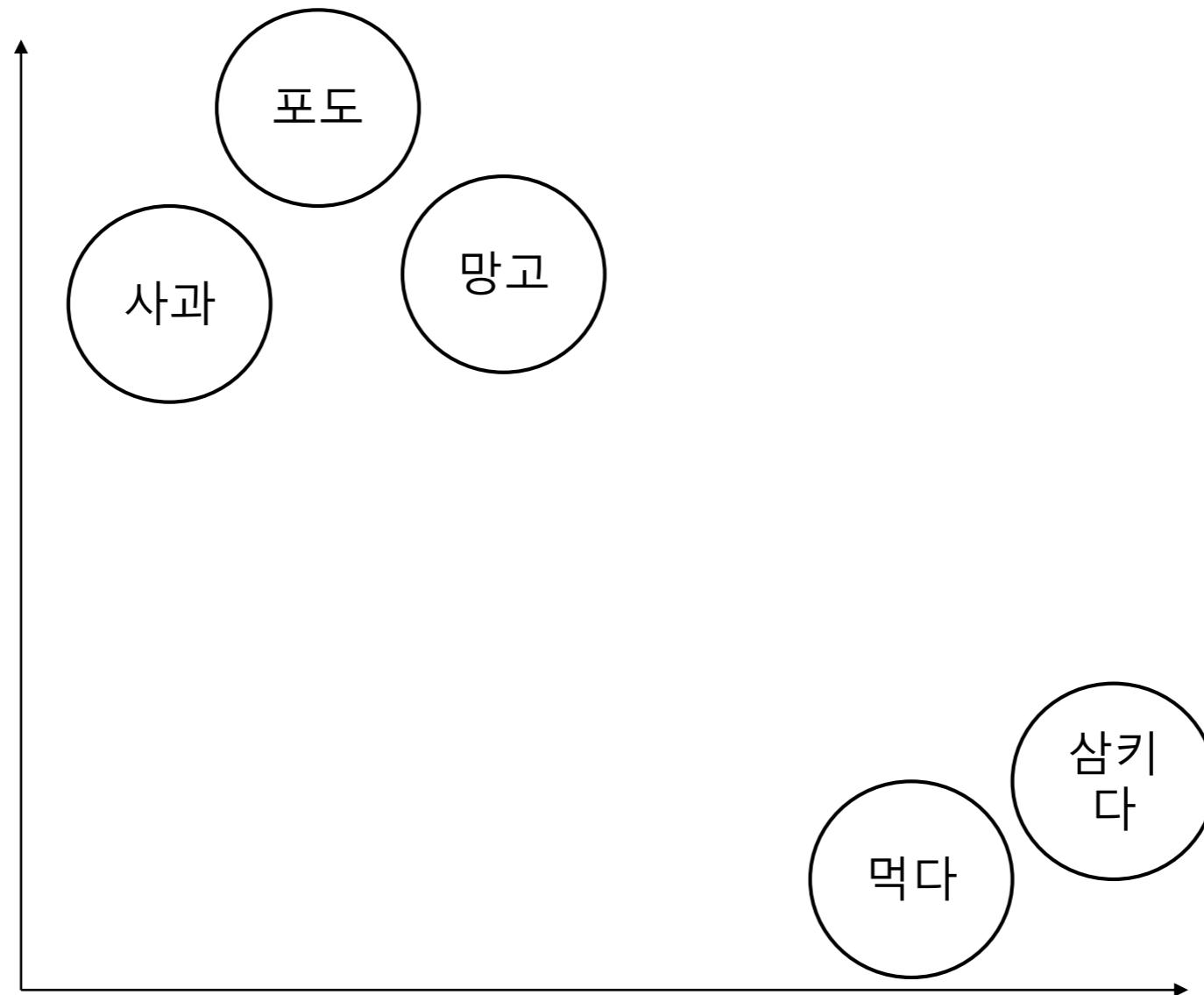
- 단어 벡터란, 각 단어를 50~300개 정도의 차원으로 구성된 벡터로 표현하는 방법이다.

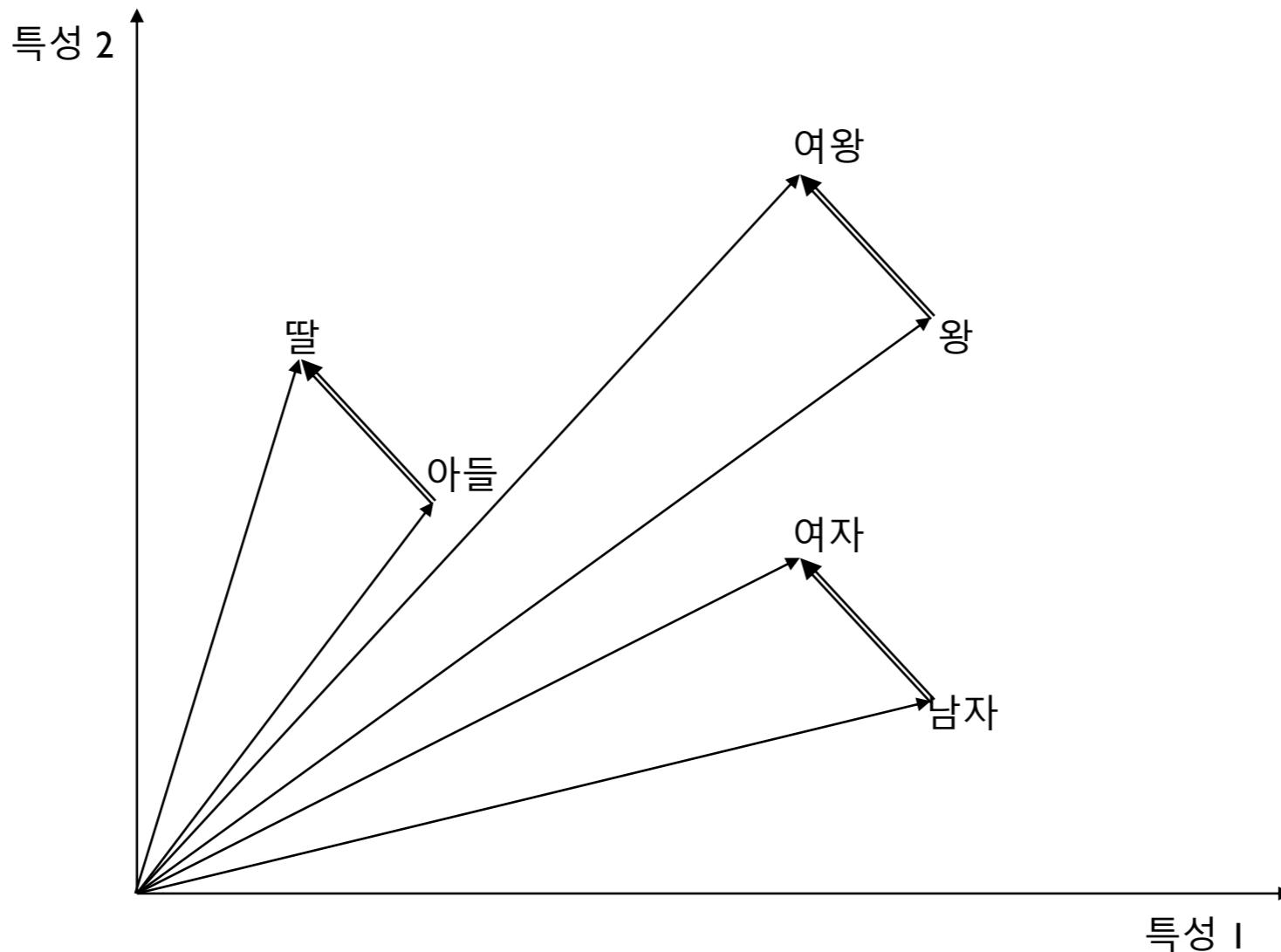
학교 = [0.23, 0.58, 0.97, ... , 0.87, 0.95]

바다 = [0.45, 0.37, 0.81, ... , 0.22, 0.64]

- 단어 벡터를 사용하면 각 단어들 사이의 “거리”를 계산할 수가 있고 이 고차원 가상 공간에서의 거리를 기반으로 유의어 반대어 등을 찾아낼 수가 있다.
 - 특정 벡터 성분은 동물의 성별을 구분하거나, 단수와 복수를 구분하거나, 동사와 명사를 구분할 수도 있다.
 - 그러나 각 벡터 값의 의미는 알 수 없다

- 단어 벡터는 대형 말뭉치로부터 학습을 시킨다. 말뭉치의 문장들을 계속 입력하여 학습을 시키면 단어 벡터를 얻는다.





- 이미 만들어져 있는 단어 벡터를 가져다 사용할 수도 있다.
- 2014년 스탠포드에서 만든 Global Vectors for Word Representations
 - 위키피디아 데이터로부터 학습을 하여 40만개 단어를 100차원으로 임베딩했다.
 - nlp.stanford.edu/projects/glove에서 다운로드받을 수 있다.

- 문서-단어 행렬을 이용하여 네이버 영화 평점 데이터를 이용한 감성분석 방법을 소개하겠다.
- 데이터는 Naver sentiment movie corpus v1.0 (github.com/e9t/nsmc/)를 사용한다.
 - 이 데이터에는 영화 리뷰 20만 건이 저장되어 있다. 각 평가 데이터는 0과 1로 레이블링 되어 있는데 0은 부정, 1은 긍정 리뷰를 나타낸다.
- 한글 자연처 처리를 위해서 konlpy 패키지에서 제공하는 Twitter 문서 분석 라이브러리를 사용하겠다.

- 단어벡터 만드는 과정을 소개하겠다.
- 가장 널리 사용되는 라이브러리는 Gensim이다.
pip install gensim

```
from gensim.models.word2vec import Word2Vec  
model = Word2Vec(sentense_list, min_count=1)  
model.most_similar(positive="조선")  
##  
[('일본', 0.9953970909118652),  
 ('관련', 0.9941188097000122),  
 ('인물', 0.9938454031944275),  
 ('러시아', 0.9931197166442871),  
 ('주요', 0.9918481111526489),  
 ('대원군', 0.9915156960487366),  
 ...]
```

- 두 개의 문자열이 얼마나 다른지를 나타내는 편집거리를 이용하여 단어의 유사도를 나타낼 수 있다.
- 한 단어에서 글자를 추가, 제거, 변경함으로써 다른 단어로 바꿀 때 필요한 최소한의 편집 행동의 횟수로 결정한다.
- NLTK 라이브러리를 활용하여 두 문장의 편집거리를 계산한다.

- 영어에서는 단어들이 대부분 스페이스로 구분되므로 단어를 구분하는 것이 어렵지 않다.
- 그러나 한글은 스페이스로 나눠진 단어가 조사를 포함하거나 복합명사인 경우 등이 있어 품사를 구분하는 작업이 영어처럼 간단하지 않다.
- 한글 문장을 처리하려면 단어를 다시 더 작은 단위인 형태소로 나누는 절차가 필요하며 이를 형태소 분석(morphological analysis)이라고 한다.
- KoNLPy 라이브러리의 Twitter 형태소분석기를 사용한다

- 문서의 주제(카테고리)를 구분하는 것으로 미리 카테고리가 정해져 있지 않으므로 비지도 학습에 해당된다.
- 관련된 단어나 문서의 집합을 찾는 방법이 필요한데 잠재 디리클레 할당, LDA(Latent Dirichlet Allocation) 방법을 주로 사용
 - 관련성이 높은 단어들이 발생하면 같은 토픽으로 분류한다.
- 한 문서에는 여러 토픽이 복합적으로 존재할 수 있다. 각 토픽의 비중은 다를 수 있다.
- DLA에서는 문서의 각 토픽들이 디리클레 분포를 따른다고 가정하고 각 문서를 각 토픽에 “할당”하는 방식으로 동작한다.
 - 문서마다 토픽이 어떻게 분포되어 있는지, 그리고 토픽마다 단어의 분포가 어떤지 파악.
 - 토픽에 따라 단어의 분포를 결정하고 그중 가장 높은 확률의 단어를 선택한다.

- LDA는 말뭉치로부터 대표적인 토픽을 먼저 선정하고 해당 토픽으로부터 단어들을 뽑아서 문서를 생성한다.
- 반대로 보면 주어진 문서에 등장한 단어들이 어떤 토픽에서 뽑혔고 그 토픽의 확률이 어떻게 분포하였는지를 추론해 내는 것이 LDA의 학습 과정이다.
- 말뭉치의 단어가 어떤 토픽에 해당하는지 명시적으로 표시되어 있지 않기 때문에 학습을 통해 '잠재적'인 정보를 추출해야 한다.

- 여기서는 LDA를 이용하는 예로서, 뉴스기사를 사용하여 토픽 모델링을 수행하는 예를 설명하겠다.
- 먼저 여러 문서에서 자주 나타나는 공통 단어를 제거하고 상위 10000개의 단어를 선택하여 BOW 모델을 생성한다.
- LDA 분석으로 얻은 결과는 주제를 구별하는데 도움을 주지만 비지도 학습이기 때문에 완벽한 정답은 아니다.
 - 주제에 할당된 문서를 확인하여 평가하고 검증하는 과정은 사람이 해주어야 한다.