

# Chap 1. 한눈에 보는 머신러닝

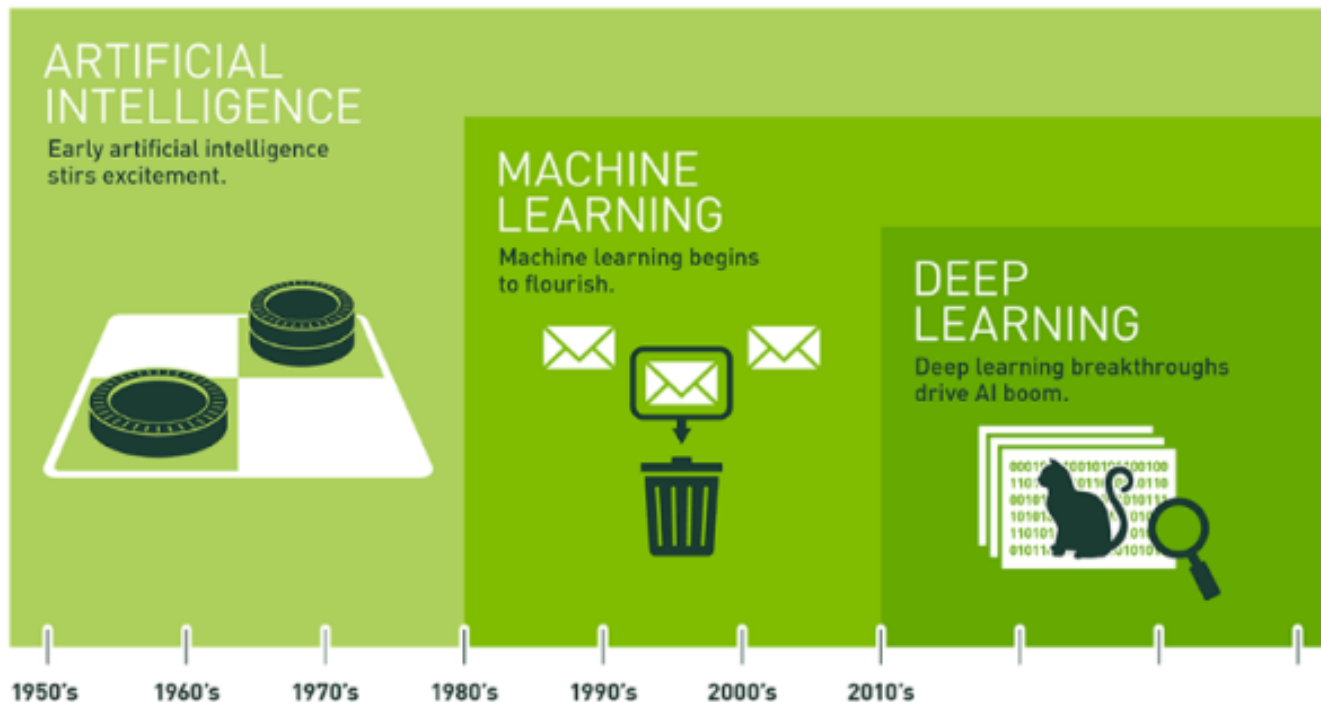
---

Seolyoung Jeong, Ph.D.

경북대학교 IT 대학

# 인공지능의 개념

- ◆ **인공지능** : 인간의 감각, 사고력을 지닌 채 인간처럼 생각하는 컴퓨터
- ◆ **머신 러닝** : 인공 지능을 구현하는 구체적 접근 방식.  
알고리즘을 이용해 데이터를 분석하고, 분석을 통해 학습하고, 학습한 내용을 기반으로 판단이나 예측 수행
- ◆ **딥 러닝** : 완전한 머신 러닝을 실현하는 기술.  
**인공신경망**에서 발전한 형태의 인공 지능.  
뇌의 뉴런과 유사한 정보 입출력 계층을 활용



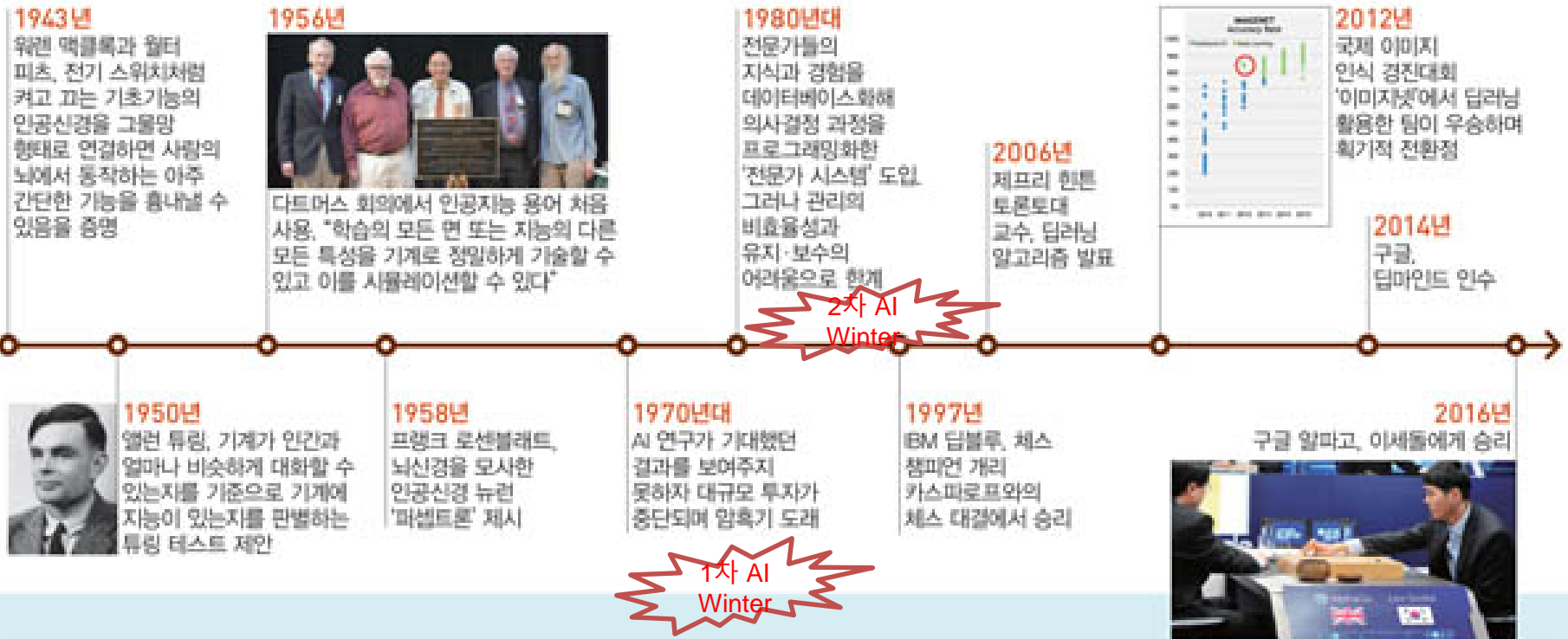
# 인공지능의 역사

## ◆ 2번의 AI Winter

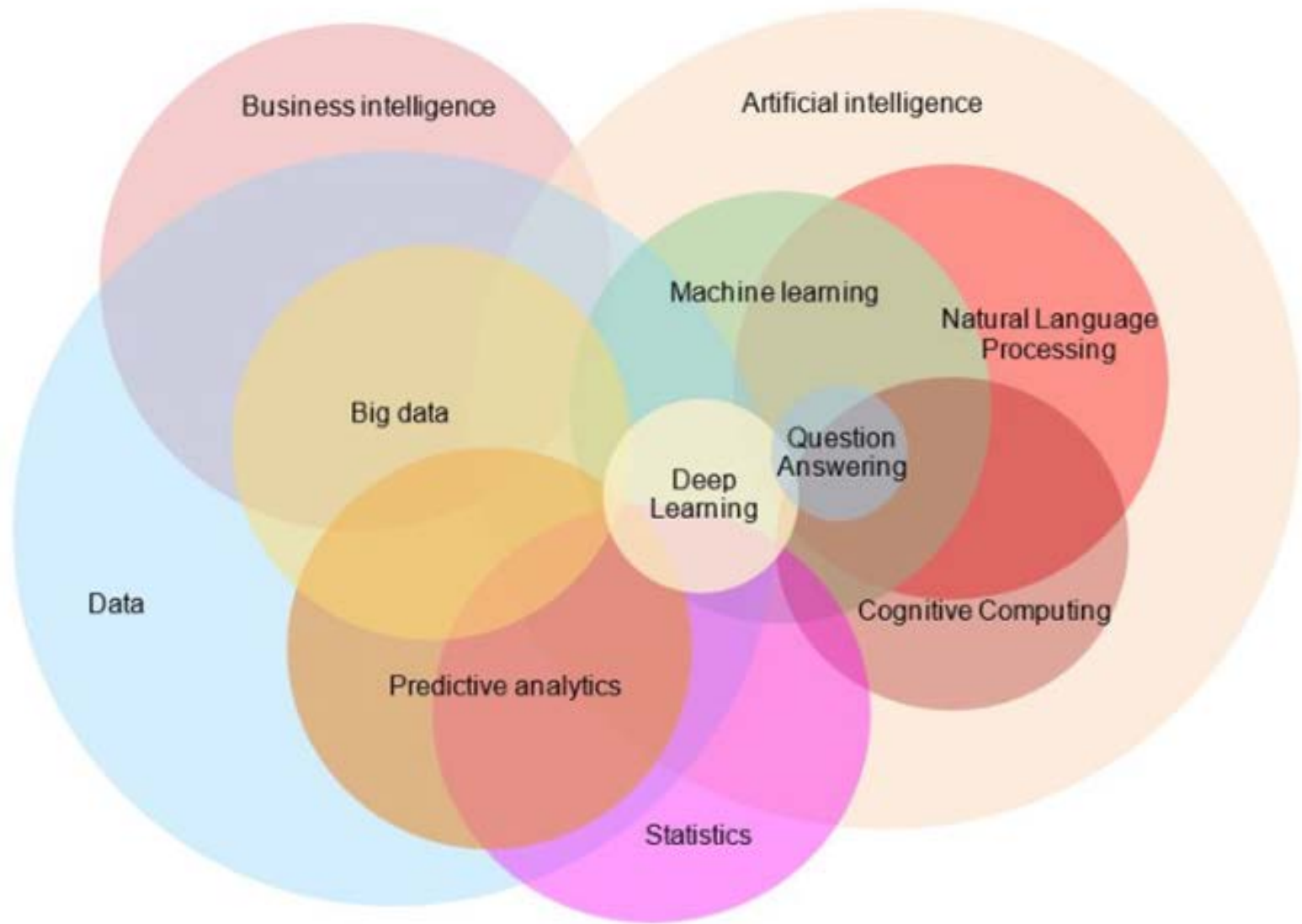
- 1960년대 말 ~ 1970년대 : 부족한 컴퓨터 기술, XOR 문제
- 1980년대 후반 ~ 1990년대 초 : 자연어 처리의 한계 (컴퓨터는 의미를 이해하지 못함)

## ◆ 3번의 AI 봄 (현재는 3차. 발전된 하드웨어 성능 및 이미지넷 등 딥러닝 개발)

### 인공지능(AI)의 역사

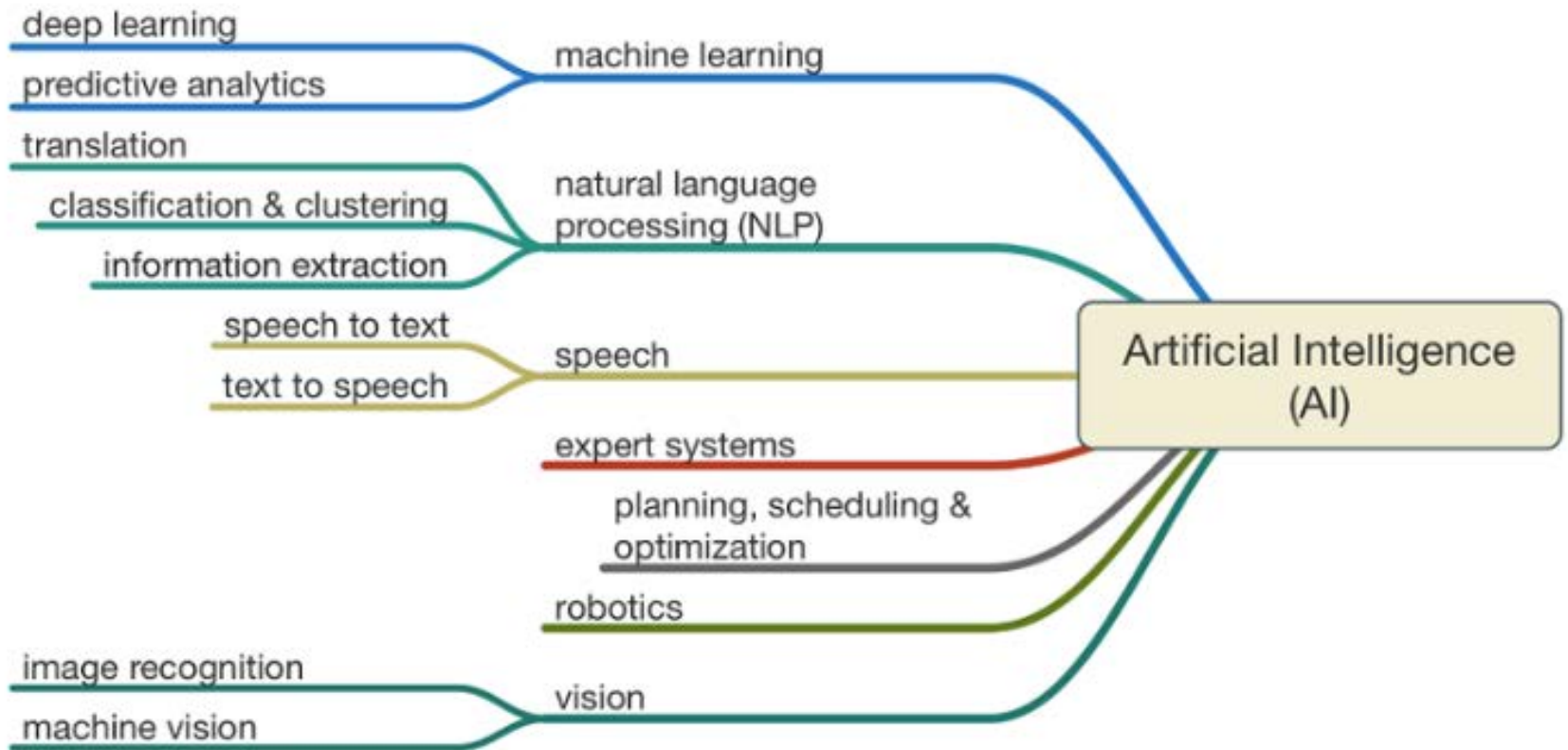


# 인공지능 연구 기술



# 인공지능 연구분야

- ◆ 자연어 처리, 음성인식, 전문가 시스템, 로봇, 컴퓨터 비전, 의료 등 다양
- ◆ 기계가 지능적으로 움직이기 위해 필요한 기술 연구



# Contents

---

## 1.1 머신러닝이란?

## 1.2 왜 머신러닝을 사용하는가?

## 1.3 머신러닝 시스템의 종류

### 1.3.1 지도 학습과 비지도 학습

### 1.3.2 배치 학습과 온라인 학습

### 1.3.3 사례 기반 학습과 모델 기반 학습

## 1.4 머신러닝의 주요 도전 과제

### 1.4.1 충분하지 않은 양의 훈련 데이터

### 1.4.2 대표성 없는 훈련 데이터

### 1.4.3 낮은 품질의 데이터

### 1.4.4 관련 없는 특성

### 1.4.5 훈련 데이터 과대적합

### 1.4.6 훈련 데이터 과소적합

## 1.5 테스트와 검증

# 1.1 머신러닝이란?

- ◆ 데이터로부터 학습하도록 컴퓨터를 프로그래밍하는 과학
- ◆ 기계 학습(機械學習) 또는 머신 러닝(영어: machine learning)
  - 인공지능의 한 분야로, 컴퓨터가 학습할 수 있도록 하는 알고리즘과 기술을 개발하는 분야
  - 예) 기계 학습을 통해서 수신한 이메일이 스팸인지 아닌지를 구분할 수 있도록 훈련 [출처: 위키백과 기계학습]
- ◆ 많은 양의 데이터를 분석하기 위해서는 먼저 시각화가 필요
- ◆ 데이터 시각화를 위한 라이브러리 :
  - R(ggplot), SQL, Zepplin(라이브러리 아님)
  - Python – Matplotlib (그래프/차트 도식화), Numpy (배열, 벡터 계산), Pandas (Numpy를 기반으로 개발. 데이터 정렬 가능한 자료구조 Dataframe)

# 1.1 머신러닝이란?

- ◆ 1959년, 아서 사무엘은 기계 학습을 "기계가 일일이 코드로 명시하지 않은 동작을 데이터로부터 학습하여 실행할 수 있도록 하는 알고리즘을 개발하는 연구 분야"라고 정의하였다. [위키백과:기계학습 정의]
- ◆ 어떤 **작업 T**에 대한 컴퓨터 프로그램의 **성능을 P**로 측정했을 때 **경험 E**로 인해 성능이 향상됐다면, 이 컴퓨터 프로그램은 작업 T와 성능 측정 P에 대해 E로 학습한 것이다. [토미첼, 1997]

구분	작업 T	경험 E	성능측정 P
스팸 메일 시스템	새로운 메일이 스팸인지 구분	훈련데이터	정확도

시스템이 학습하는데 사용하는 샘플 : 훈련세트 (training set)  
각 훈련데이터 : 훈련사례 (training instance, 샘플)



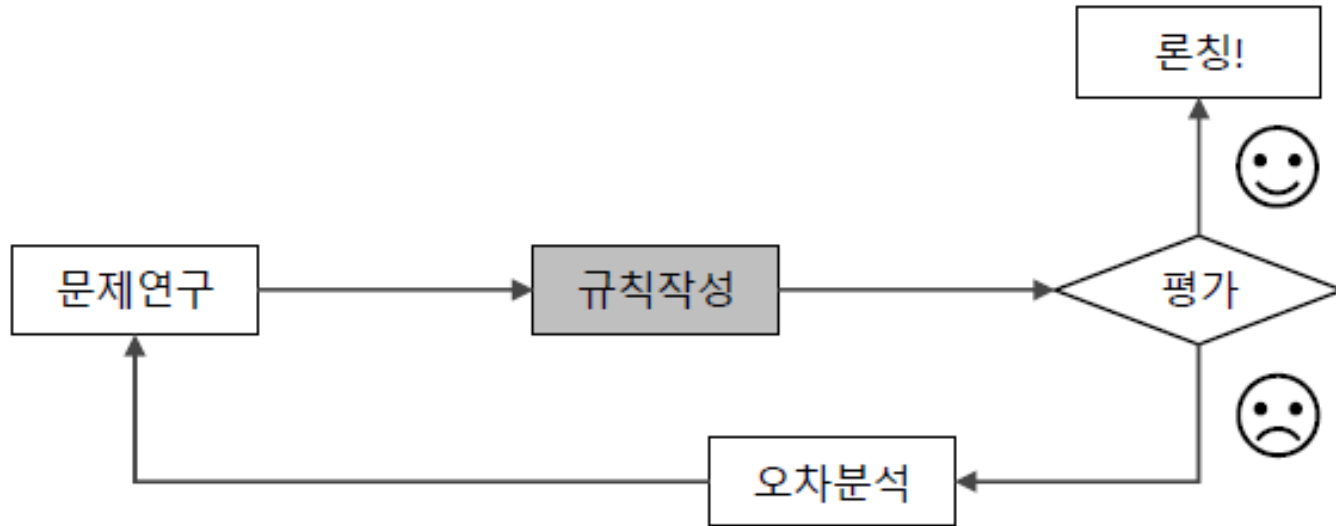
# 1.2 왜 머신러닝을 사용하는가?

## ◆ 머신러닝 기반 해결 필요한 문제들

- 기존 솔루션으로는 많은 수동 조정과 규칙이 필요한 문제
- 전통적인 방법으로는 전혀 해결 방법이 없는 복잡한 문제
- 유동적인 환경에 적응하기 어려운 문제
- 대량의 데이터와 복잡한 문제들로 해결하기 어려운 문제

# 전통적인 접근 방법

- ◆ 문제가 단순하지 않아 규칙이 점점 길고 복잡해지므로 유지보수가 매우 힘들

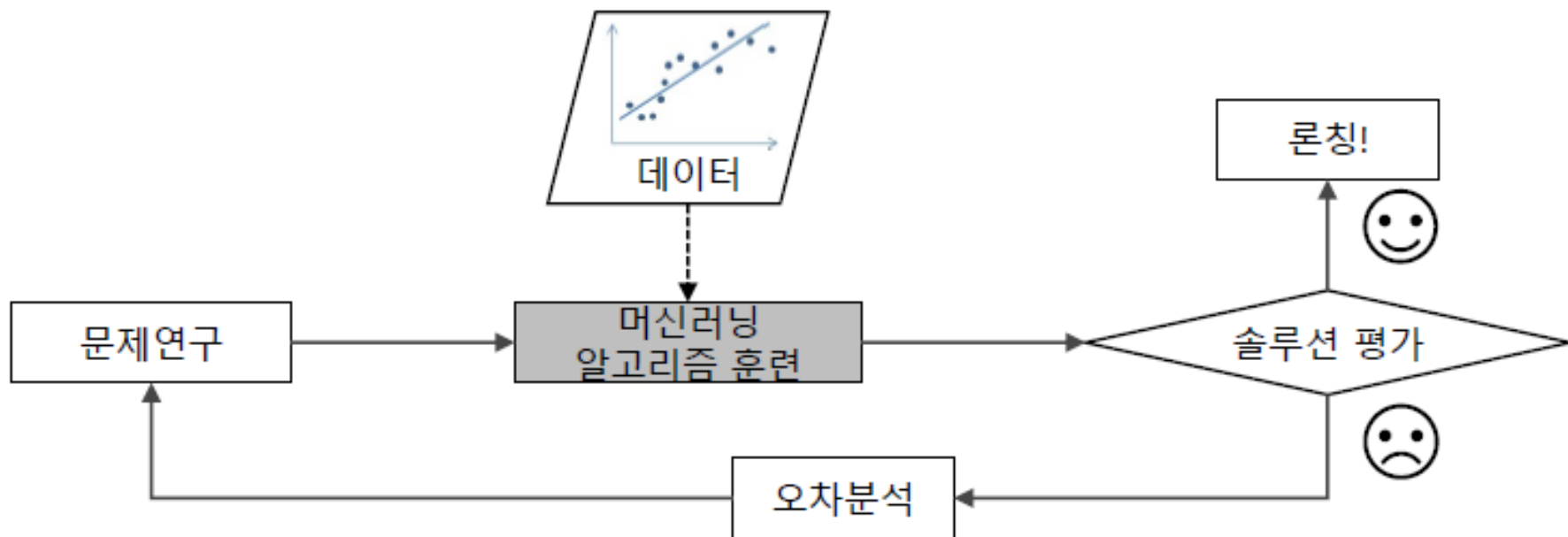


- 예) 스팸필터 : 규칙(4U, 신용카드, 무료, 대출, 광고, 대행 등)을 분석, 패턴을 감지하는 알고리즘 작성 후 테스트/적용
- 작성된 규칙 예)  
if “광고” in “이메일제목” :  
    스팸처리

# 머신러닝 기반 문제 해결 장점

## ◆ 머신러닝 접근 방법

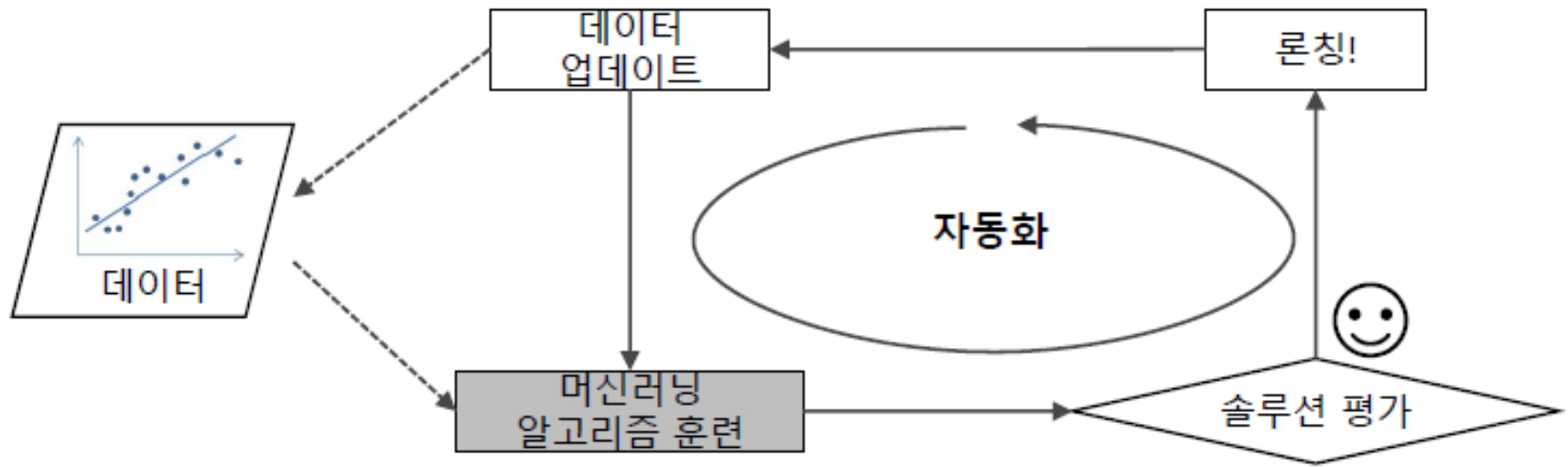
- 문제에 대한 패턴을 인지하고 학습
- 프로그램이 짧아지고 유지보수가 쉬우며 정확도를 높임



# 머신러닝 기반 문제 해결 장점

## ◆ 자동으로 변화에 적응

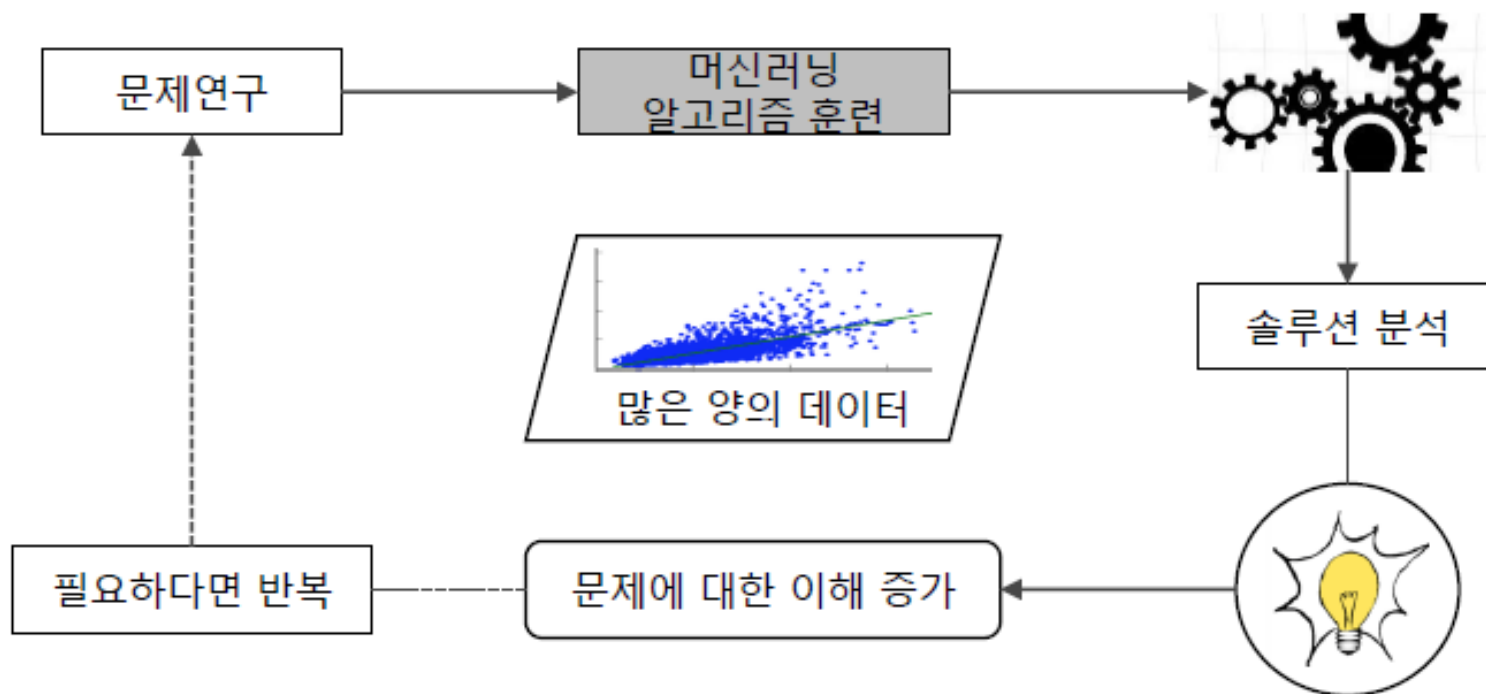
- 사용자가 지정한 데이터에서 특정 패턴을 자동으로 인식하고 별도의 작업이 없어도 분류



# 머신러닝 기반 문제 해결 장점

## ◆ 머신러닝을 통해 배우기

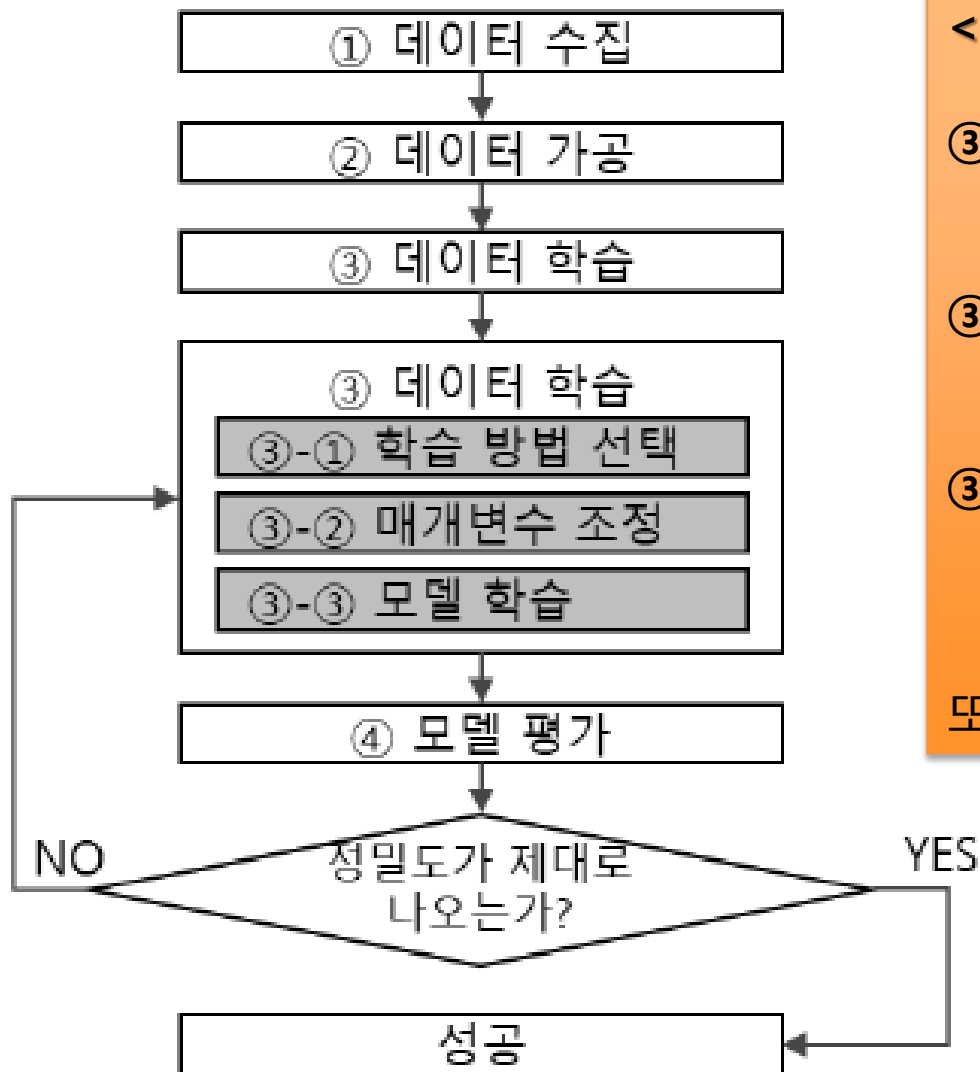
- 대용량의 데이터를 분석하면 겉으로는 보이지 않던 패턴을 발견  
→ 데이터 마이닝 (Data Mining)



## 1.3 머신러닝 시스템의 종류

- ◆ 머신러닝 시스템의 종류는 굉장히 많으며, 아래와 같이 크게 3가지로 분류할 수 있음
  - (지도학습 / 비지도학습) 사람의 감독 하에 훈련하는 것인지, 그렇지 않은 것인지
  - (온라인 학습 / 배치 학습) 실시간으로 점진적인 학습을 하는지 아닌지
  - (사례 기반 학습 / 모델 기반 학습)  
단순히 알고 있는 데이터 포인트와 새 데이터 포인트를 비교하는지,  
아니면 훈련 데이터셋에서 과학자들처럼 패턴을 예측하여 예측 모델을 만드는지

# 머신러닝의 과정



## <데이터학습>

### ③-① 학습방법선택 :

알고리즘 선택 (K-means, SVM 등)

### ③-② 매개변수 조정 :

데이터와 알고리즘에 맞게 변수 조정

### ③-③ 모델 학습 :

테스트 데이터를 활용해 정밀도 측정

정밀도가 나오지 않으면 매개변수 수정  
또는 알고리즘 변경 이후 반복 수행

## 1.3.1 지도 학습과 비지도 학습

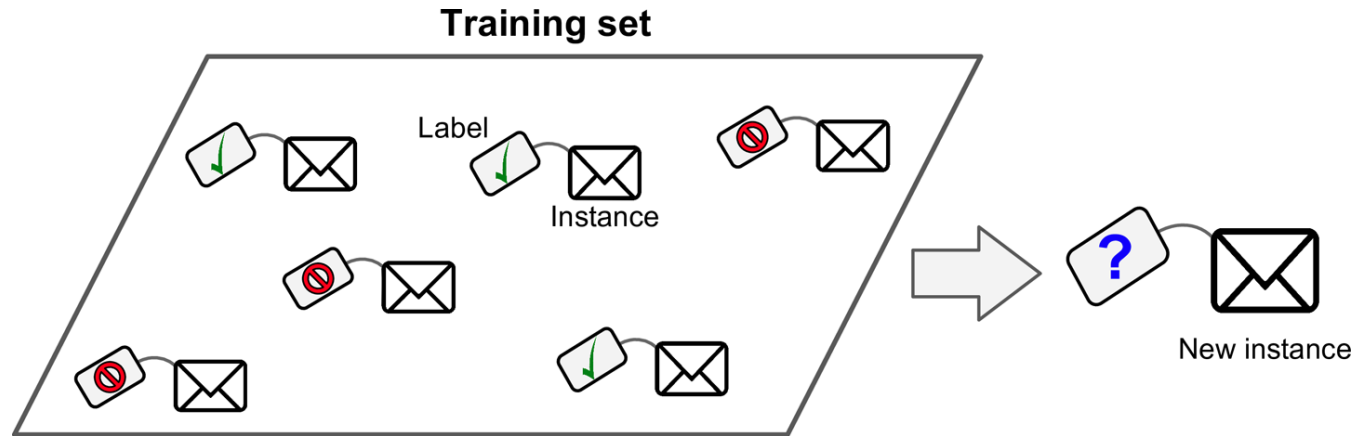
- ◆ 학습하는 동안의 감독 형태와 정보량에 따라  
지도학습 / 비지도학습 / 준지도학습 / 강화학습으로 구분

종류	내용	알고리즘
지도 학습	데이터와 정답을 함께 입력	<ul style="list-style-type: none"><li>• K-최근접 이웃</li><li>• 선형회귀</li><li>• 로지스틱 회귀</li><li>• 서포트 벡터 머신</li><li>• 결정트리와 랜덤 포레스트</li><li>• 신경망</li></ul>
	다른 데이터의 정답을 예측	
비지도 학습	데이터는 입력하지만 정답은 미입력	<ul style="list-style-type: none"><li>• 군집</li><li>• 시각화와 차원 축소</li><li>• 연관 규칙 학습</li></ul>
	다른 데이터의 규칙성 찾음	
강화 학습	부분적으로 정답을 입력	
	데이터를 기반으로 최적의 정답을 찾음	



# 지도 학습

- ◆ 알고리즘에 주입하는 훈련데이터에 레이블(LABEL)이라는 정답이 포함



- ◆ 가능한 작업1: 분류(Classfication)

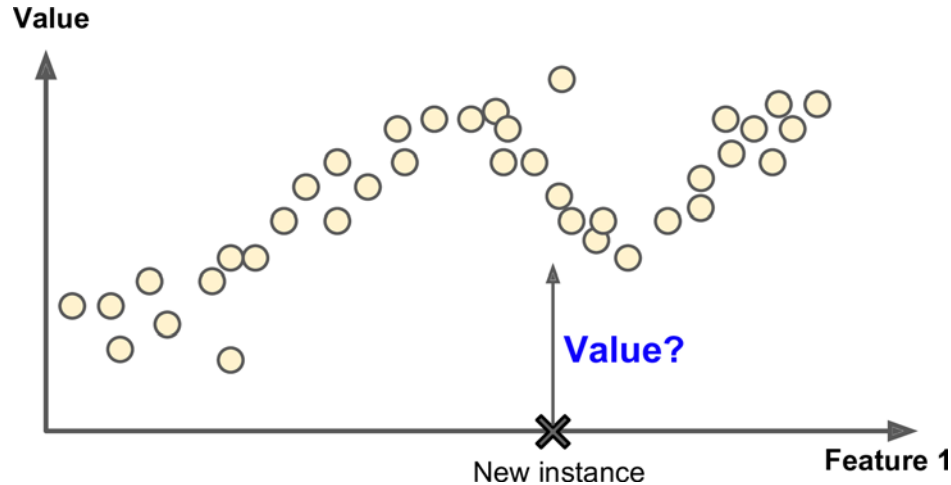
- 예) 스팸필터 작업 (많은 양의 메일, 스팸 유무)

- ◆ 가능한 작업2: 타겟 수치 예측

- 예) 중고차 가격
- 예측 변수(predictor variable)라 부르는 특성(feature)를 사용해 타겟 예측
- **Feature** : 주행거리, 연식, 브랜드, 사고유무 등
- Feature와 Label(중고차 가격)이 포함된 많은 데이터가 필요
- 이러한 작업을 회귀(regression)이라 함

# 지도 학습 알고리즘

## ◆ 회귀 (Regression)



## ◆ 예) 로지스틱 회귀 : 분류에 많이 사용됨. 클래스에 속할 확률 출력 (스팸일 가능성 20%)

## ◆ 대표적 지도 학습 알고리즘

- K-Nearest Neighbors (k-최근접 이웃)
- Linear Regression (선형 회귀)
- Logistic Regression (로지스틱 회귀)
- Support Vector Machines (SVM, 서포트 벡터 머신)
- Decision Tree(결정 트리)와 Random Forests (랜덤 포레스트)
- Neural networks (신경망) : 일부 신경망 구조는 비지도학습

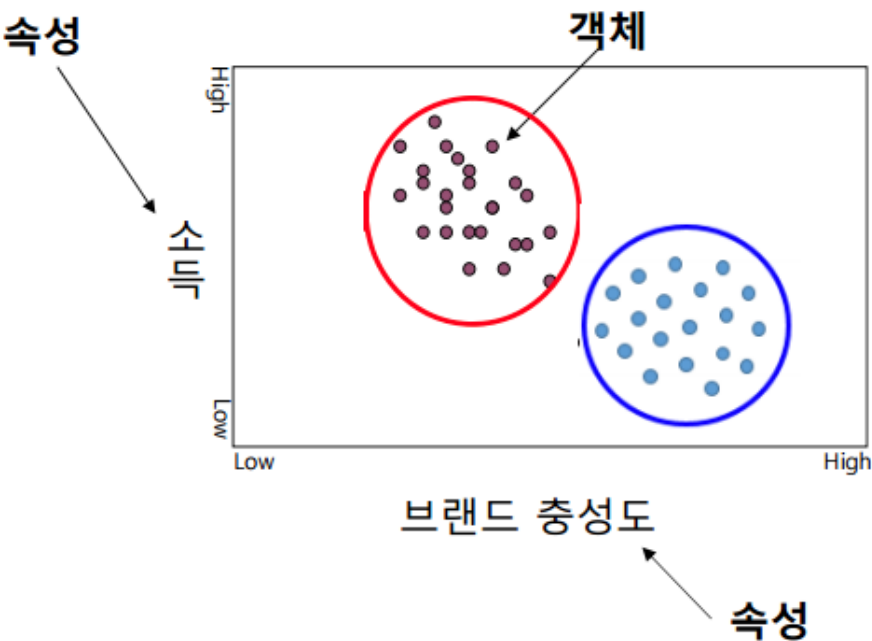
# 비지도 학습

- ◆ 훈련 데이터에 레이블이 없는 것
- ◆ 최종적으로 내야하는 답이 정해져 있지 않는 것이 특징
- ◆ 대표적 비지도 학습 알고리즘
  - **Clustering (군집)**
    - k-Means (k-평균)
    - Hierarchical Cluster Analysis (HCA, 계층 군집 분석)
    - Expectation Maximization (기댓값 최대화)
  - **Visualization(시각화)와 Dimensionality Reduction (차원 축소)**
    - Principal Component Analysis (PCA, 주성분(최적치) 분석)
    - Kernel PCA
    - Locally-Linear Embedding (LLE, 지역적 선형 임베딩)
    - t-distributed Stochastic Neighbor Embedding (t-SNE, t개의 분산 기반 확률적 근접 위상 배치법)
  - **Association Rule Learning (연관 규칙 학습)**
    - Apriori (어프라이어리) : breadth-first search strategy 사용
    - Eclat (이클렛) : depth-first search algorithm

# 비지도 학습

## ◆ 군집 (Clustering) :

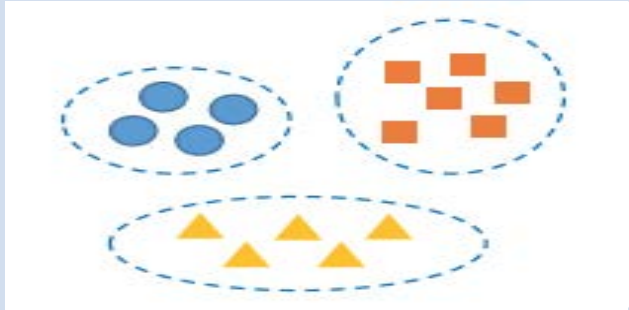
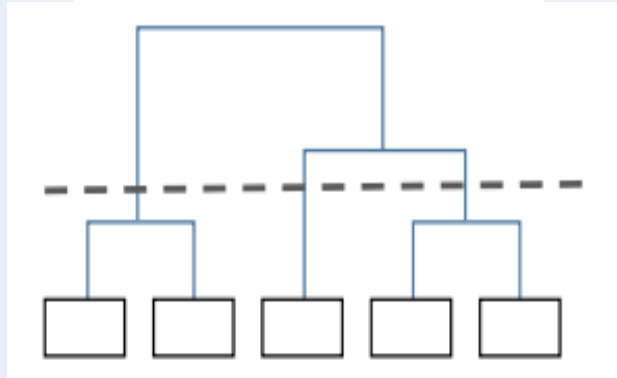
- 유사한 속성의 객체들을 군집(cluster)으로 나누거나 묶어주는 데이터마이닝 기법
- 예) 고객들의 구매 패턴을 반영하는 속성들에 관한 데이터 수집



군집 분석을 통해 유사한 구매패턴을 보이는  
고객들을 군집화하고 판매전략을 도출

# 비지도 학습

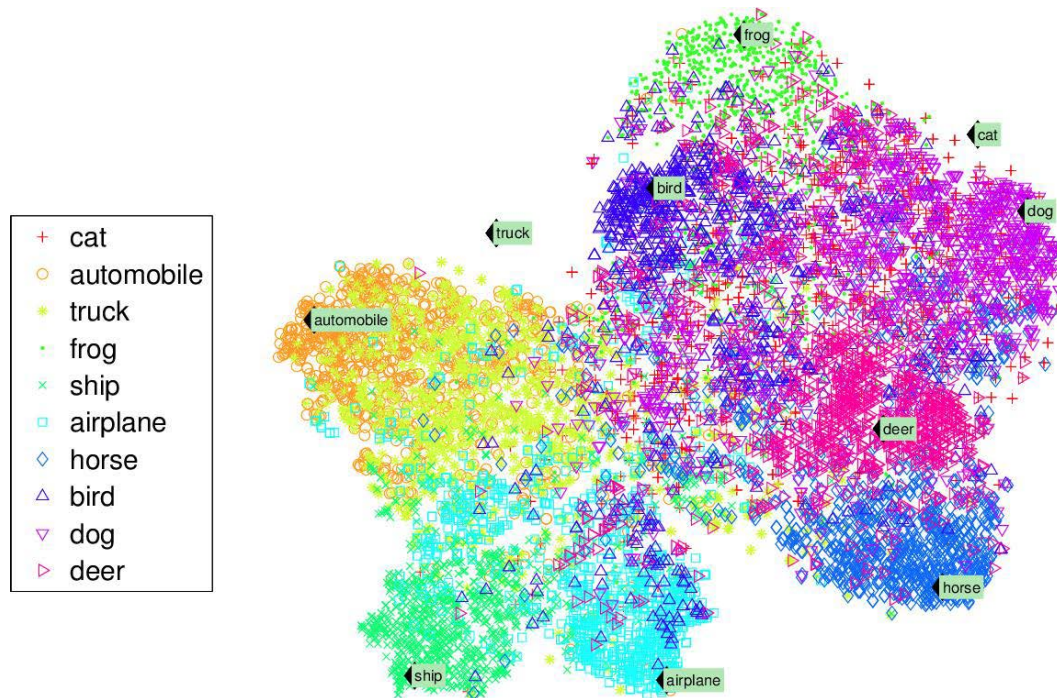
- ◆ 군집 분석의 방법은 ‘계층적 방법’ 과 ‘비계층적 방법’ 으로 구분

종류	내용	도식화
비계층적 군집 (Non-hierarchical Clustering)	사전에 군집 수 $k$ 를 정한 후 입력 데이터를 $k$ 개 중 하나의 군집에 배정	
계층적 군집 (Hierarchical Clustering)	사전에 군집 수 $k$ 를 정하지 않고 단계적으로 군집 트리를 제공	

# 비지도 학습

## ◆ 시각화와 차원 축소 (Visualization and Dimensionality Reduction)

- 레이블이 없는 대규모의 고차원 데이터를 2D나 3D로 표현함
- 데이터가 어떻게 조직되어 있는지 이해할 수 있고 예상치 못한 패턴 발견 가능



- 차원 축소 (Dimensionality Reduction)는 너무 많은 정보를 잃지 않으면서 데이터를 간소화하며, 하나의 특성으로 합침  
→ 특성 추출 (feature extraction)

# 비지도 학습

## ◆ 이상치 탐지 (Anomaly detection)

- 시스템은 정상 샘플로 훈련, 새로운 샘플이 정상 데이터인지 혹은 이상치인지 판단
- 예) 제조 결함 잡아내기, 학습 알고리즘 전 데이터셋에서 이상한 값 자동 제거



## ◆ 연관 규칙 학습 (Association rule learning)

- 동시 발생 규칙을 이용해 특성 간 관계 탐구로 데이터 패턴 분석
- 연관 규칙 분석 : 군집 분석 이후에 각 그룹의 특성을 분석하기 위해 사용
- 예) 바비큐 소스와 감자를 구매한 사람 → 스테이크 구매 경향 있음

# 준지도 학습

## ◆ 레이블이 일부만 있는 데이터

- 대부분 레이블이 없는 데이터가 많고, 레이블이 있는 데이터는 아주 조금

## ◆ 예) 구글 포토 호스팅 서비스

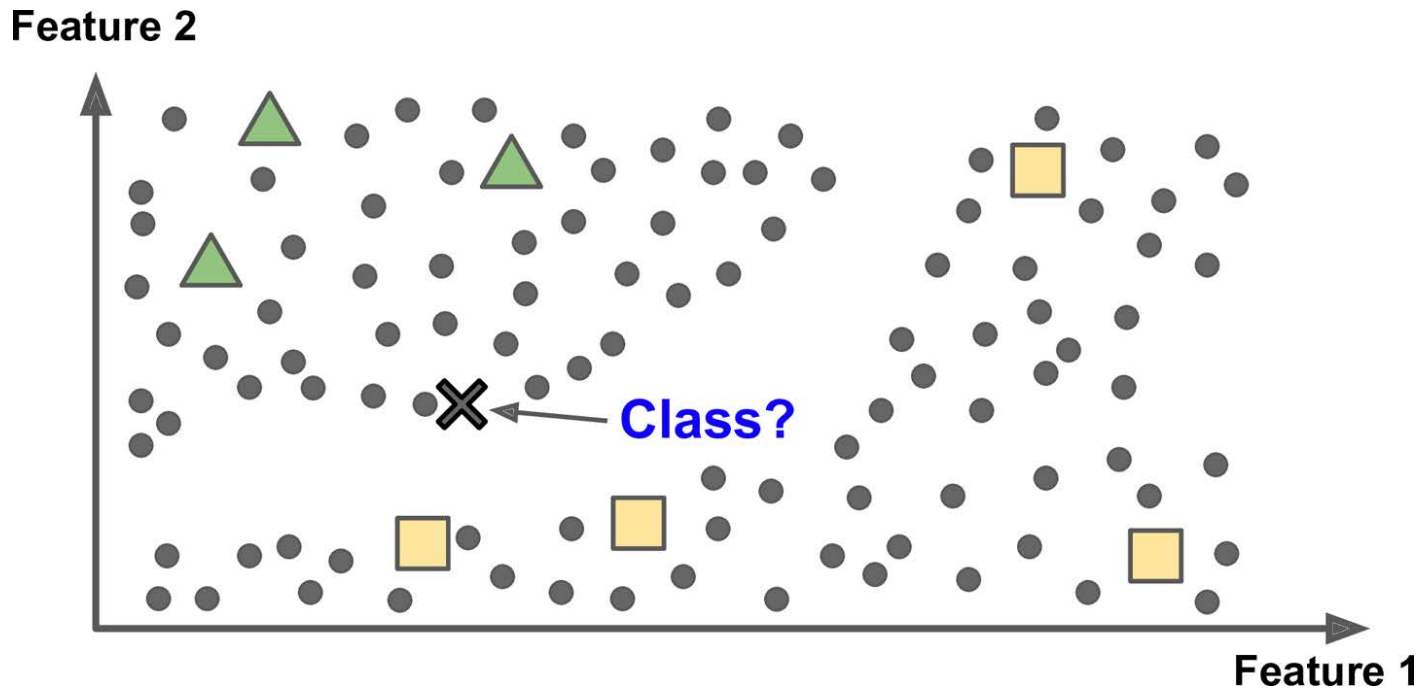
- 가족사진 업로드 서비스
- 사람 A : 사진 1,5,11 / 사람 B : 사진 2,5,7에 있음 자동 인식 : 비지도학습(군집)
- 문제: 이 사람들이 누구인가?
- 사람마다 레이블이 하나씩만 주어진다면 사진에 있는 모든 사람의 이름을 알 수 있음



# 준지도 학습

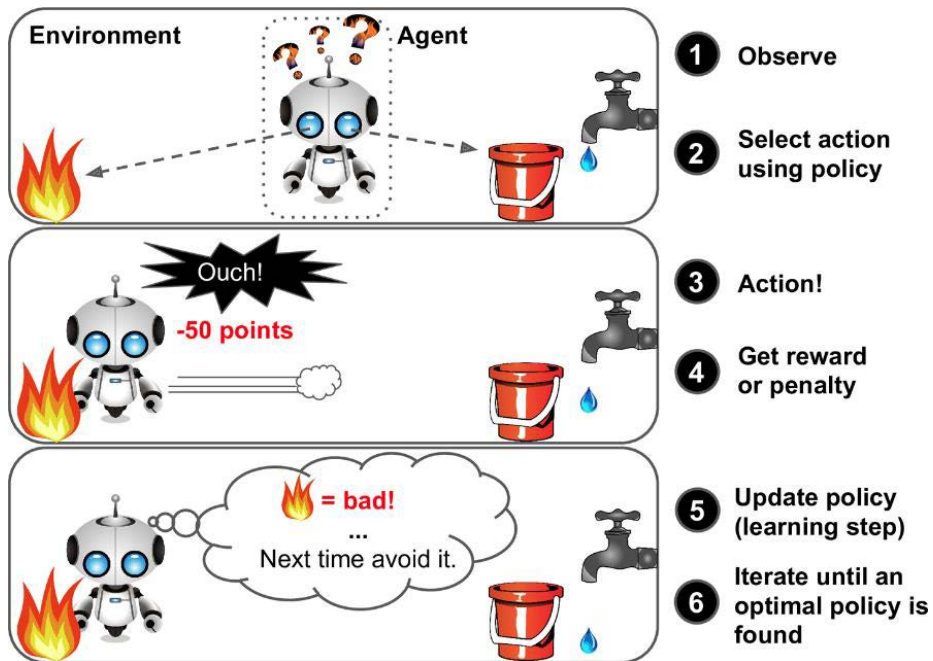
## ◆ 대부분의 준지도 학습 : 지도 학습 + 비지도 학습의 조합

- 비지도 학습 방식으로 순차적으로 훈련된 다음, 전체 시스템이 지도 학습 방식으로 세밀하게 조정됨



# 강화 학습

- ◆ 현재 상태를 관찰해서 어떻게 대응해야 할지와 관련된 문제를 다룬다
- ◆ 행동의 주체, 환경(상황 또는 상태), 보상/벌점 등으로 구성
  - 학습하는 시스템: 에이전트
  - 환경을 관찰해서, 행동을 실행하고, 그 결과로 **보상(reward)** 또는 **벌점(penalty)**을 받음
  - 큰 보상을 얻기 위한 최상의 전략(policy) : 시간이 지나면서 스스로 학습

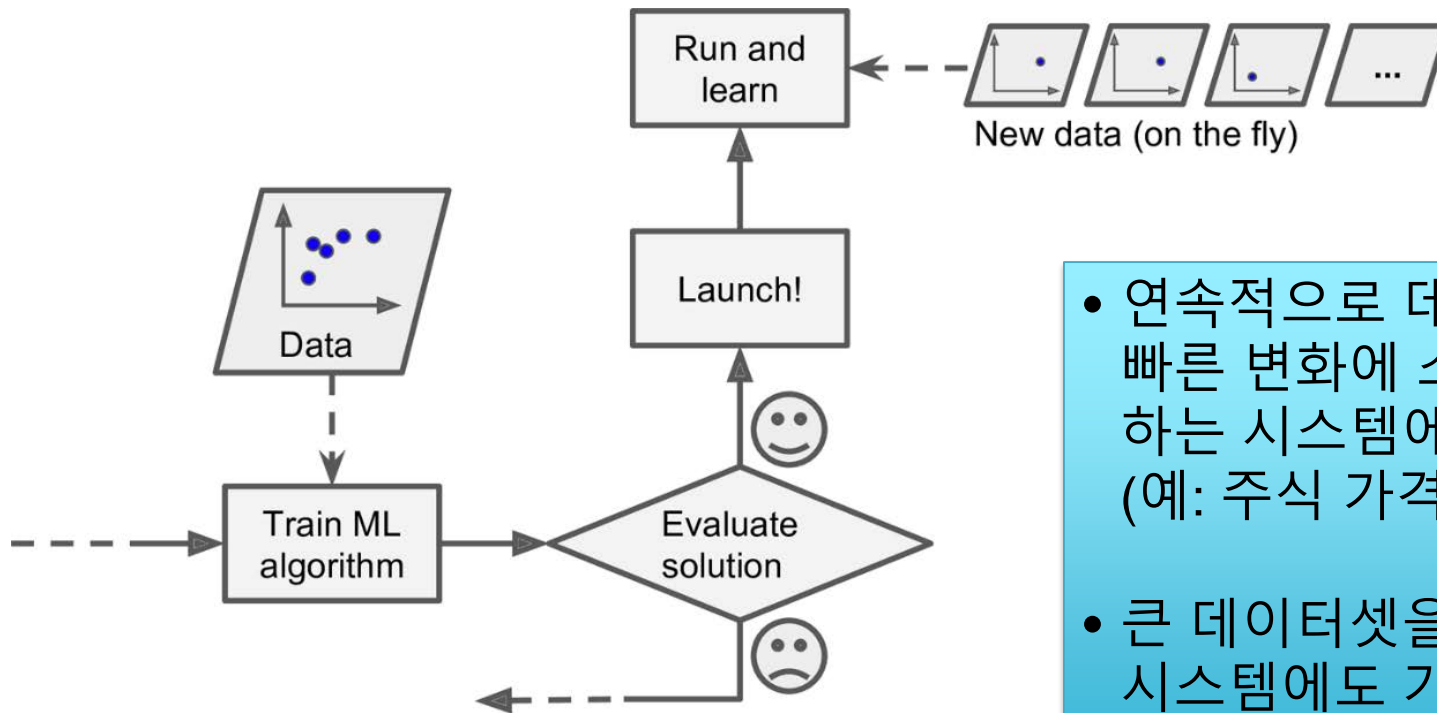


## 1.3.2 배치 학습과 온라인 학습

- ◆ 머신러닝 시스템을 분류하는데 사용하는 또 다른 기준
- ◆ 입력 데이터의 스트림으로부터 점진적으로 학습 가능 여부
- ◆ 배치학습 (batch learning)
  - 가용한 데이터를 모두 사용하여 훈련 → 시간과 자원 많이 소모
  - 먼저 시스템을 훈련시키고, 제품에 적용하여 더 이상의 학습 없이 실행됨
  - 즉, 학습한 것을 단지 적용만 함 : 오프라인 학습 (offline learning)
- 머신러닝 시스템 훈련, 평가, 론칭 과정 자동화
- 주기적으로 데이터 업데이트 후 시스템 새버전 훈련
- 문제점
  - 새로운 데이터를 학습하려면 전체 데이터를 사용하여 처음부터 다시 훈련해야 함
  - 전체 데이터셋 사용 훈련에 많은 컴퓨팅 자원 필요 (CPU, 메모리 공간 등)
  - 자원이 제한된 시스템(스마트폰, 로봇 등)에서 많은 양의 훈련 데이터를 이동, 학습을 위해 몇 시간씩 많은 자원 사용은 심각한 문제 일으킴

# 온라인 학습

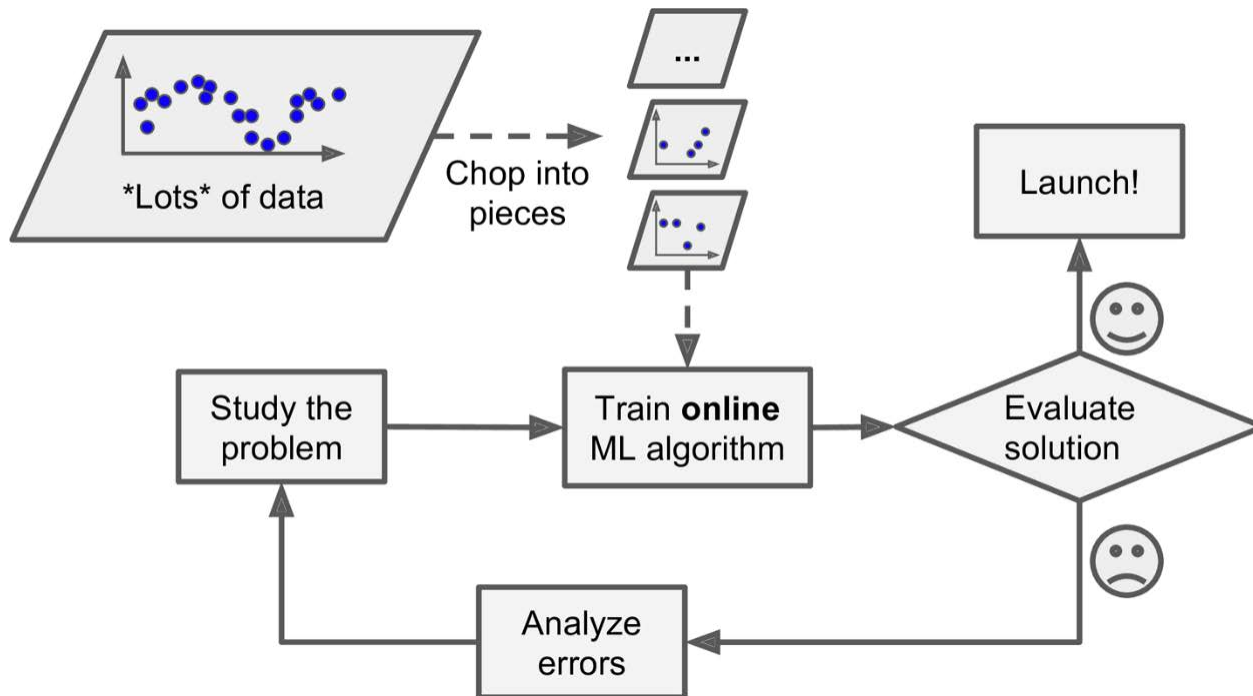
- ◆ 데이터를 순차적으로 한개씩 또는 **미니배치(mini-batch)**라 부르는 작은 묶음 단위로 주입하여 시스템 훈련
- ◆ 비용이 적게 들어 시스템은 데이터가 도착하는 대로 즉시 학습 가능



- 연속적으로 데이터를 받고 빠른 변화에 스스로 적응해야 하는 시스템에 적합 (예: 주식 가격)
- 큰 데이터셋을 학습하는 시스템에도 가능 : 외부 메모리 (out-of-core) 학습

# 온라인 학습

- ◆ **학습률 (learning rate)** : 변화하는 데이터에 얼마나 빠르게 적응할 것인가
- ◆ 학습률을 높게 하면 데이터에 빠르게 적응하지만, 예전 데이터를 금방 잊음
- ◆ 학습률이 낮으면 시스템의 관성으로 더 느리게 학습, 데이터 잡음에 덜 민감



문제점: 나쁜  
데이터가 주입되면  
시스템 성능이  
점진적으로 감소

# 1.3.3 사례 기반 학습과 모델 기반 학습

- ◆ 머신러닝 시스템이 어떻게 일반화 되는가에 따라 분류
- ◆ 대부분의 머신러닝 작업: 예측을 만드는 것
- ◆ 주어진 훈련 데이터로 학습하지만, 훈련데이터에서는 본 적 없는 새로운 데이터로 일반화되어야 함
- ◆ 훈련 데이터에서 높은 성능을 내는 것이 좋지만 그게 최종 목표는 아님
- ◆ 진짜 목표는 새로운 샘플에 잘 작동하는 모델
- ◆ 일반화를 위한 두 가지 접근법 :
  - 사례 기반 학습 / 모델 기반 학습

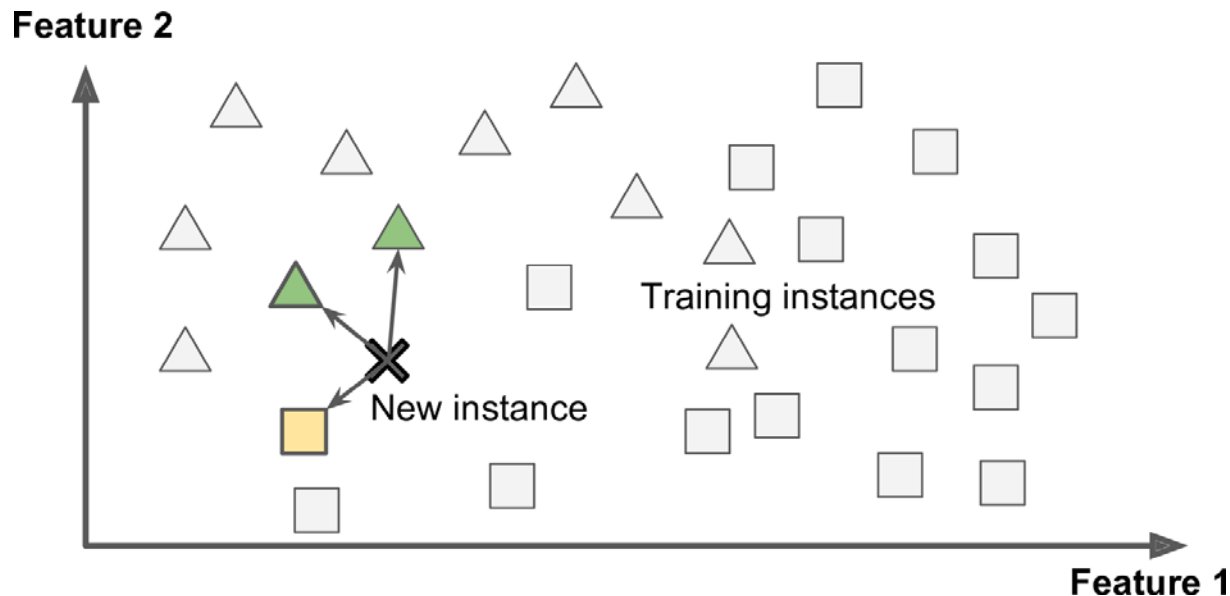
# 사례 기반 학습

## ◆ 유사도(similarity) 측정

- 스팸 메일과 동일한 메일을 스팸이라고 지정하는 대신  
스팸 메일과 매우 유사한 메일을 구분하도록 스팸 필터 프로그램 개발
- 예) 공통으로 포함한 단어의 수를 세는 것  
스팸 메일과 공통으로 가지고 있는 단어가 많으면 스팸으로 분류

## ◆ 사례 기반 학습 (instance-based learning)

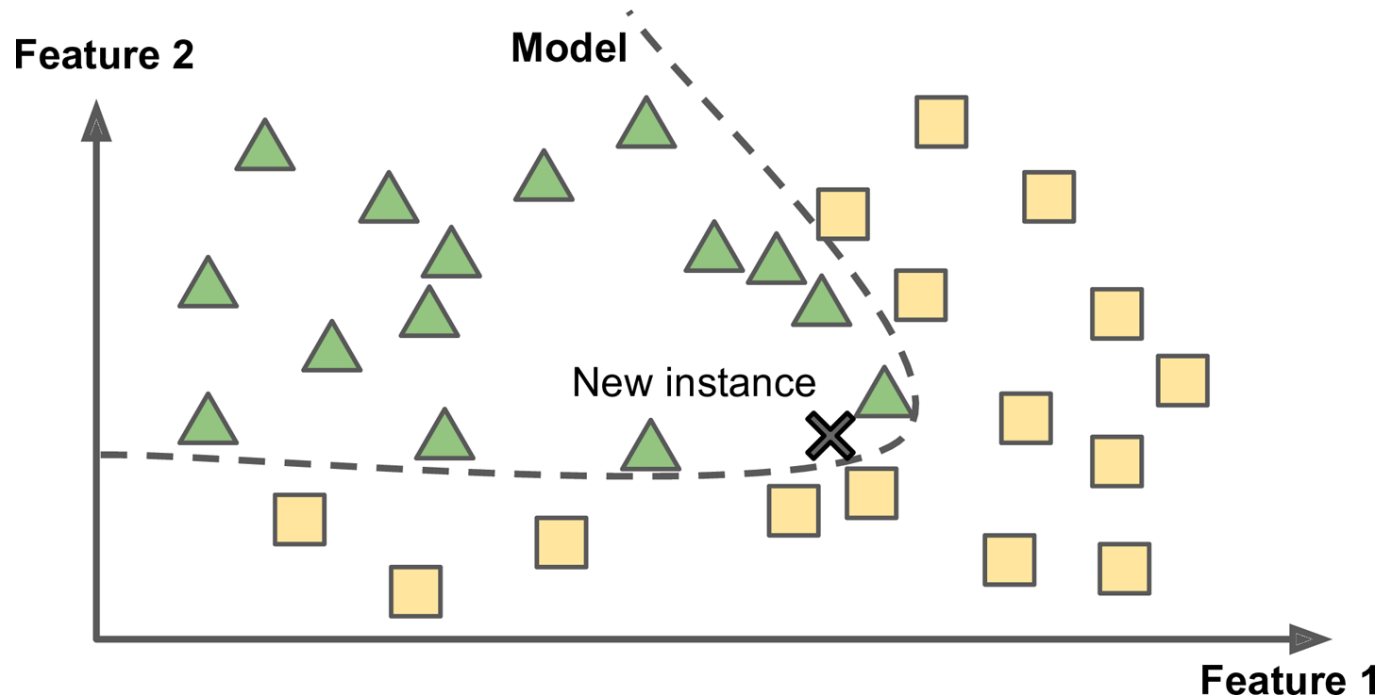
- 시스템이 사례를 기억함으로써 학습  
유사도 측정을 통해 새로운 샘플을 일반화



# 모델 기반 학습

## ◆ 모델 기반 학습 (model-based learning)

- 샘플들의 모델을 만들어 예측에 사용



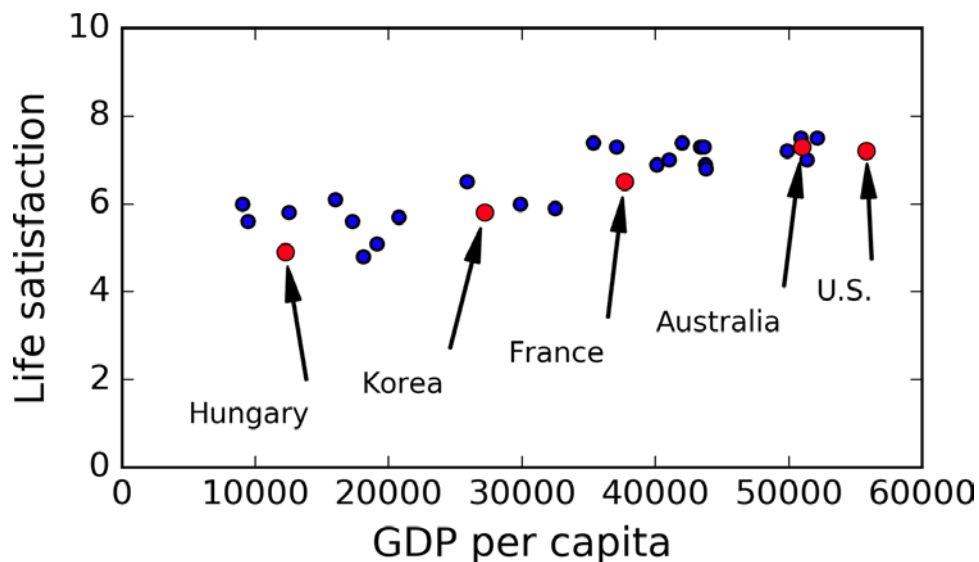


# 모델 기반 학습

## ◆ 예) 1인당 GDP에 대한 삶의 만족도

- 국가별 '1인당 GDP' 와 '삶의 만족도' (표 / 그래프)

Country	GDP per capita (USD)	Life satisfaction
Hungary	12,240	4.9
Korea	27,195	5.8
France	37,675	6.5
Australia	50,962	7.3
United States	55,805	7.2

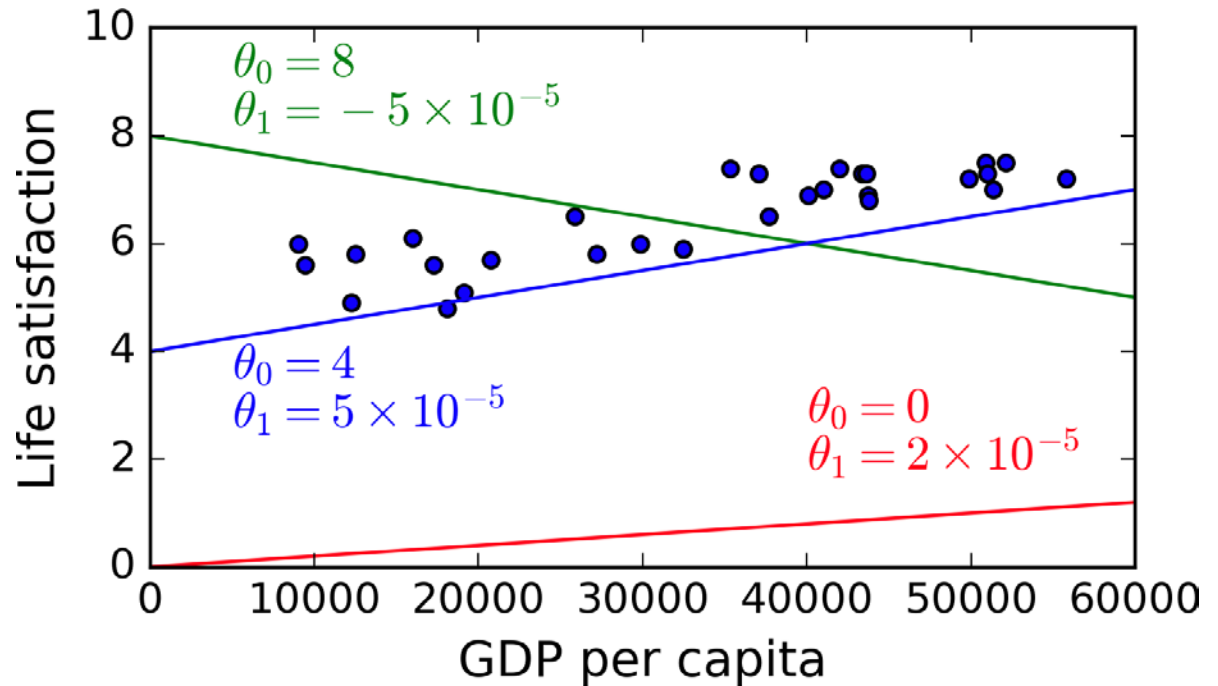


- 1인당 GDP가 증가할수록 선형으로 같이 올라감
- 1인당 GDP의 선형 함수로 **삶의 만족도 선형 모델**을 얻음

$$\text{삶의 만족도} = \theta_0 + \theta_1 \times \text{1인당 GDP}$$

# 모델 기반 학습

- ◆ 모델 파라미터  $\theta_0, \theta_1$  조정
- ◆ 가능한 몇 개의 선형 모델

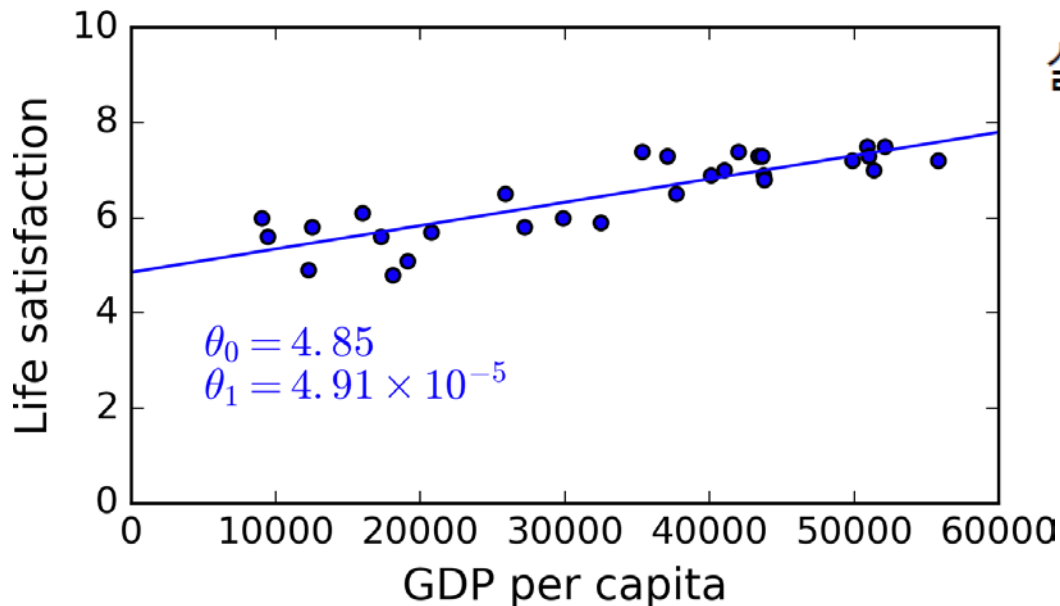


## ◆ 모델의 최상 성능 측정지표

- 모델이 얼마나 좋은지 측정 : 효용 함수, 적합도 함수
- 모델이 얼마나 나쁜지 측정 : 비용 함수

# 모델 기반 학습

- ◆ 예) 선형 회귀 → 훈련과 예측 데이터 사이의 거리를 재는 비용함수 최소화가 목표
- ◆ 선형 회귀 알고리즘
  - 알고리즘에 훈련 데이터를 공급하면 데이터에 가장 잘 맞는 선형 모델 파라미터 찾음 : **모델을 훈련(training) 시킨다.**
- ◆ 훈련 데이터에 최적인 선형 모델 찾음



$$\text{삶의 만족도} = \theta_0 + \theta_1 \times \text{1인당 GDP}$$

예) 국가별 '1인당 GDP'와 '삶의 만족도' 표에 없는 키프로스 국가의 1인당 GDP는 22,587달러  
→ 삶의 만족도 계산 : 5.96

# 머신 러닝 시스템 작업

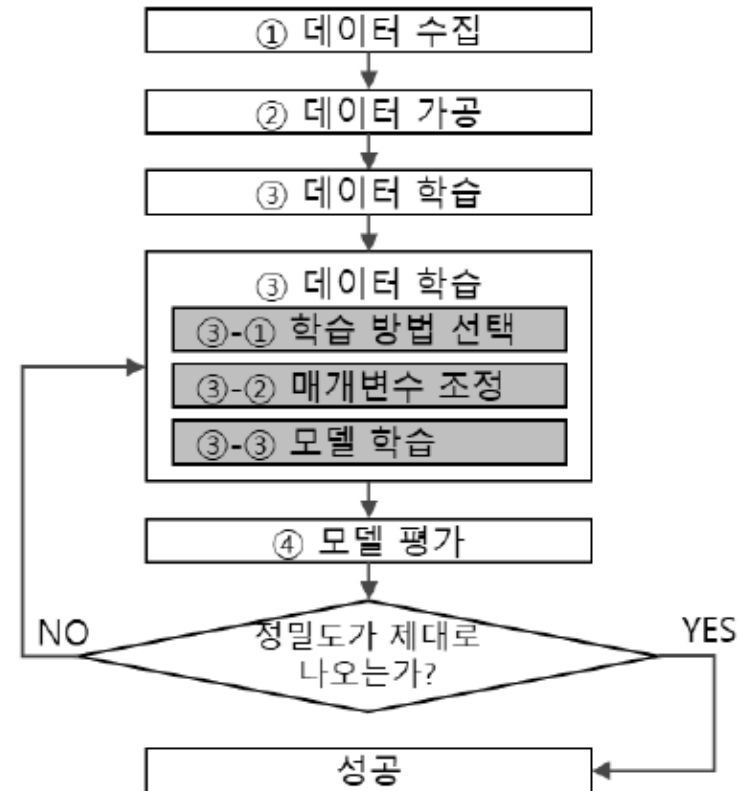
◆ 모든게 다 잘 되면 모델은 좋은 예측 내놓음

◆ 그렇지 않은 경우 추가 처리 필요

- 더 많은 특성 (고용률, 건강, 대기 오염) 을 사용하거나,
- 좋은 훈련 데이터를 더 많이 모으거나,
- 더 강력한 모델(예: 다항 회귀 모델)을 선택해야 할지도...

◆ 머신러닝 시스템 작업 요약

1. 데이터를 분석
2. 모델을 선택
3. 훈련데이터로 모델을 훈련시킴  
(비용함수가 최소인 모델 파라미터를 찾음)
4. 새로운 데이터에 모델 적용하여 예측 수행  
모델의 일반화 기대...



# 1.4 머신러닝의 주요 도전 과제

## ◆ 우리의 주요 작업

- 학습 알고리즘을 선택해서, 어떤 데이터를 훈련시키는 것
- 문제될 수 있는 두 가지 : 나쁜 알고리즘, 나쁜 데이터

## ◆ 나쁜 데이터 사례

- 충분하지 않은 양의 훈련 데이터
- 대표성 없는 훈련 데이터
- 낮은 품질의 데이터
- 관련 없는 특성

## ◆ 나쁜 알고리즘 사례

- 훈련 데이터 과대적합
- 훈련 데이터 과소적합

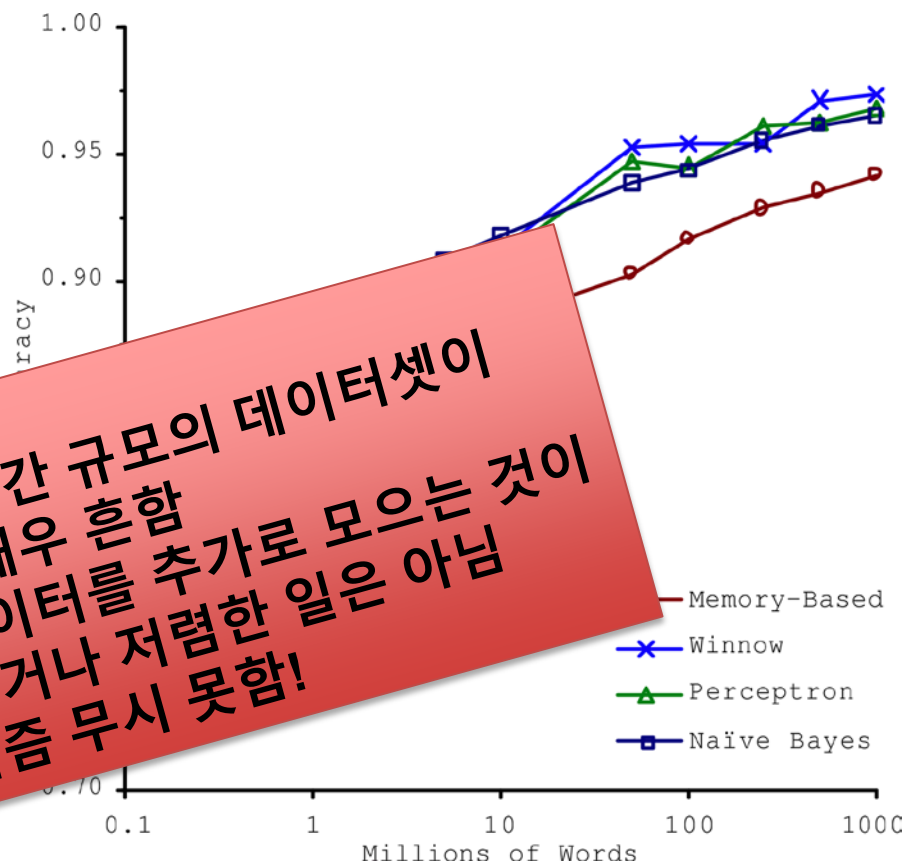
## 1.4.1 충분하지 않은 양의 훈련 데이터

- ◆ 어린 아이는 ‘사과’에 대해 알려주면 ‘모든 종류의 사과’를 쉽게 일반화 함
- ◆ 예) 충분한 데이터가 주어지면 여러 다른 머신러닝 알고리즘과 관계없이 거의 비슷하게 잘 처리함

시간과 돈을  
알고리즘 개발에 쓰는 것  
vs.  
말뭉치 개발에 쓰는 것  
trade-off 관계

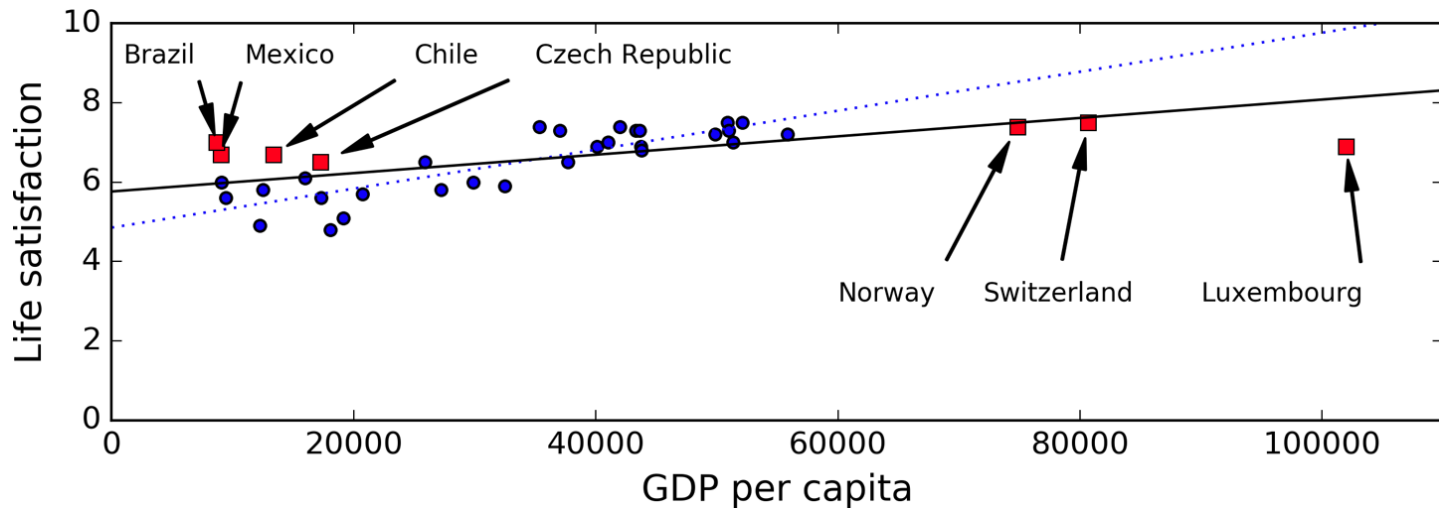
[Michele Banko and Eric Brill, “Scaling to very very large corpora for natural language disambiguation”, 2001]

하지만,  
✓ 작거나 중간 규모의 데이터셋이  
여전히 매우 흔함  
✓ 훈련 데이터를 추가로 모으는 것이  
항상 쉽거나 저렴한 일은 아님  
✓ 알고리즘 무시 못함!



## 1.4.2 대표성 없는 훈련 데이터

- ◆ 일반화가 잘되려면, 원하는 새로운 사례를 훈련 데이터가 잘 대표하는 것이 중요
- ◆ 선형 모델 훈련 예) 사용한 나라의 집합에 일부 나라 빠져 있어 대표성이 완벽하지 못함
  - 누락된 나라 추가했을 때 (대표성이 더 큰 훈련 샘플)



- ◆ 누락된 나라 추가 전 모델 : 점선 → 추가 후 모델 : 실선
  - 간단한 선형 모델은 잘 동작하지 않음!
- ◆ 샘플링 잡음 (noise) : 샘플이 작거나 대표성 없는 데이터
- ◆ 샘플링 편향 (bias) : 표본 추출 방법이 잘못된 경우 → 대표성 없음

## 1.4.3 낮은 품질의 데이터

- ◆ 훈련 데이터가 에러, 이상치, 잡음으로 가득한 경우
- ◆ 패턴을 찾기 어려워 잘 동작 안함
- ◆ → 훈련 데이터 정제 필요
  
- ◆ 사실, 대부분의 데이터 과학자가 데이터 정제에 많은 시간 할애
- ◆ 이상치 샘플 무시하거나 수동으로 고침
- ◆ 일부 샘플에만 특성 몇개가 누락될 시,  
특성 모두 무시 / 샘플 무시 / 빠진 값 채움

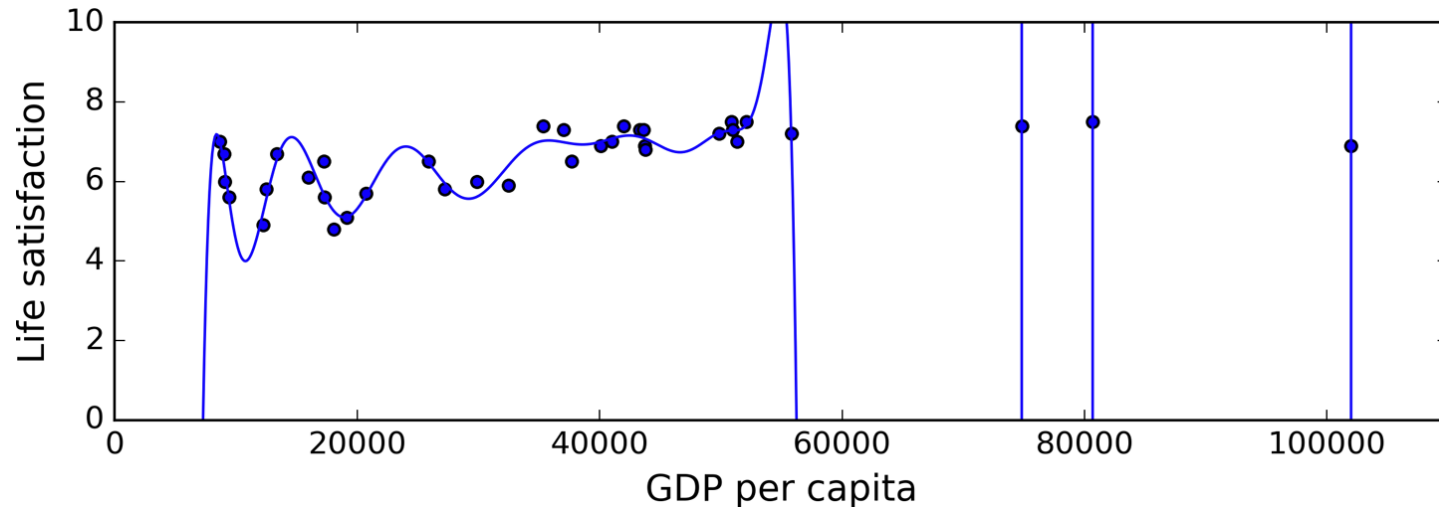


## 1.4.4 관련 없는 특성

- ◆ 훈련에 사용할 좋은 특성을 찾는 것 : 특성 공학
- ◆ 성공적인 머신러닝 프로젝트의 핵심 요소
- ◆ **특성 선택 (feature selection)** : 가지고 있는 특성 중에서 훈련에 가장 유용한 특성 선택
- ◆ **특성 추출 (feature extraction)** : 특성을 결합하여 더 유용한 특성을 만듦 (예: 차원 축소 알고리즘 사용)
- ◆ 새로운 데이터를 수집해 새 특성 만듦

## 1.4.5 훈련 데이터 과대적합

- ◆ 해외여행 중 택시 운전사가 내 물건을 훔쳤다고 가정.  
그 나라 모든 택시 운전사는 도둑이라고 생각
- ◆ 일반화의 오류. 머신러닝에서는 과대적합(overfitting)이라고 함
- ◆ 모델이 훈련데이터에만 너무 잘 맞는 경우



- ◆ 훈련 세트에 잡음이 많거나 데이터셋이 너무 작으면 (샘플링 잡음이 발생하므로) 잡음이 섞인 패턴을 감지하게 됨.
- ◆ 이런 패턴은 새로운 샘플에 일반화되지 못함

# 훈련 데이터 과대적합

## ◆ 고차원의 다항회귀모델의 경우

### ◆ 예) 삶의 만족도 모델 : ‘나라 이름’ 특성 추가

- ‘w’가 들어간 나라들의 삶의 만족도는 ‘7’보다 크다는 패턴 감지
- 신뢰할 수 없음 (우연히 훈련 데이터에서 찾은 것)
- 이 패턴이 진짜인지, 잡음 데이터로 인한 것인지 모델이 구분해낼 방법 없음

### ◆ 과대적합 ← 훈련 데이터에 있는 잡음의 양에 비해 모델이 너무 복잡할 때 자주 발생

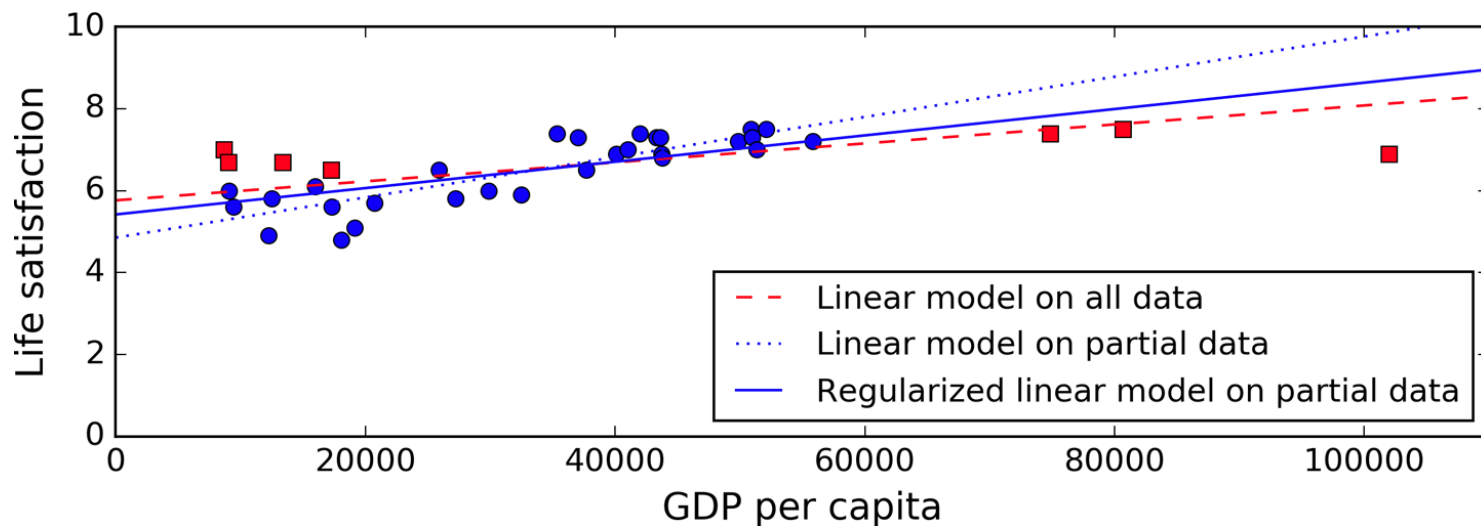
#### ◆ 해결방법

- 파라미터 수가 적은 모델 선택 (고차원 다항 모델보다는 선형 모델)
- 훈련 데이터 특성 수를 줄이거나, 모델에 제약을 가해 단순화시킴
- 훈련 데이터를 더 많이 모음
- 훈련 데이터의 잡음을 줄임 (예: 오류 데이터 수정과 이상치 제거)

# 훈련 데이터 과대적합

## ◆ 규제 (regularization)

- 모델을 단순하게 하고, 과대적합의 위험을 감소시키기 위해 모델에 제약을 가하는 것
- 규제로 과대적합될 위험 감소



- 규제가 모델의 기울기를 더 작게 만들어 훈련데이터에는 덜 맞지만, 새로운 샘플에는 더 잘 일반화됨

# 훈련 데이터 과대적합

## ◆ 하이퍼파라미터

- (모델이 아니라) 학습 알고리즘의 파라미터
- 훈련 전에 미리 저장되고, 훈련하는 동안에는 상수로 남아 있음
- 매우 큰 값의 규제 하이퍼파라미터 지정 (기울기=0에 가까운)  
→ 평편한 모델
- 과대 적합될 가능성 없지만 좋은 모델 찾지 못함
- 머신러닝 시스템 구축 시 하이퍼파라미터 튜닝은 매우 중요한 과정임

## 1.4.6 훈련 데이터 과소적합

- ◆ 과대적합의 반대
- ◆ 모델이 너무 단순해서 데이터의 내재된 구조를 학습하지 못할 때 발생
- ◆ 예) 삶의 만족도 선형 모델 : 현실은 이 모델보다 더 복잡. 훈련 샘플에서조차도 부정확한 예측
- ◆ 해결 방법
  - 파라미터가 더 많은 강력한 모델 선택
  - 학습 알고리즘에 더 좋은 특성 제공 (특성 공학)
  - 모델의 제약을 줄임 (예: 규제 하이퍼파라미터 감소)

## 1.5 테스트와 검증

- ◆ **모델이 새로운 샘플에 얼마나 잘 일반화될까?**
  - 새로운 샘플에 실제로 적용해 봄
  - 실제 서비스에 모델을 넣고 잘 동작하는지 모니터링
  - 모델이 아주 나쁠 때 고객 불만 토로
- ◆ **훈련 데이터 : 훈련 세트 + 테스트 세트로 나눔**
  - 훈련 세트를 사용해 모델을 훈련시킴 (데이터의 80%)
  - 테스트 세트를 사용해 모델을 테스트함 (나머지 20%)
- ◆ **새로운 샘플에 대한 오류 비율 : 일반화 오차 (외부 샘플 오차)**
  - 테스트 세트에서 모델을 평가하여 오차에 대한 추정값 획득
  - 이전에 본 적이 없는 새로운 샘플에 모델이 얼마나 잘 작동하는가?
  - 훈련 오차가 낮지만 (훈련 세트에서 모델의 오차 적음), 일반화 오차가 높다면 : 훈련 데이터에 과대 적합됨

# 모델 평가

## ◆ 모델 평가 : 두 모델 중 선택 갈등 (선형 모델 vs. 다항 모델)

- 두 모델 모두 훈련 세트로 훈련
- 테스트 세트를 사용해 얼마나 잘 일반화되는지 비교
- 선형 모델이 더 잘 일반화되었다고 가정
- 과대적합을 피하기 위해 규제 적용
- 하이퍼파라미터 값 선택 (100개의 하이퍼파라미터 값으로 100개의 다른 모델 훈련)
- 모델을 실제 서비스에 투입
- 성능이 예상보다 좋지 않음 (오차 15% 발생)
- 테스트 세트에 최적화된 모델
- **검증 세트** : 두번째 홀드아웃(holdout) 세트 생성
  - 최상의 성능을 내는 모델과 하이퍼파라미터 선택
- 만족스러운 모델을 찾으면 일반화 오차의 추정값을 얻기 위해 테스트 세트로 단 한번의 최종 테스트 수행
- **교차검증** : 훈련 데이터에서 검증 세트로 너무 많은 양의 데이터를 뺏기지 않기 위해 훈련 세트를 여러 서브셋으로 나누고 각 모델을 서브셋의 조합으로 훈련시키고 나머지 부분으로 검증
- 모델과 하이퍼파라미터 선택되면 전체 훈련 데이터를 사용하여 최종 모델 훈련
- 테스트 세트에서 일반화 오차 측정



# 한눈에 보는 머신러닝

- ◆ 머신러닝은 명시적인 규칙을 코딩하지 않고, 기계가 데이터로부터 학습하여 어떤 작업을 더 잘하도록 만드는 것
- ◆ 여러 종류의 머신러닝 시스템
  - 지도 학습 / 비지도 학습
  - 배치 학습 / 온라인 학습
  - 사례 기반 학습 / 모델 기반 학습
- ◆ 훈련 세트에 데이터를 모아 학습 알고리즘에 주입
  - 학습 알고리즘이 모델 기반인 경우 훈련 세트에 모델 맞추기 위해 파라미터 조정
  - 사례 기반인 경우 샘플을 기억하는 것이 학습, 새로운 샘플 일반화 위해 유사도 측정 사용
- ◆ 훈련 세트가 너무 작거나 대표성이 없는 데이터이거나, 잡음이 많고 관련 없는 특성으로 오염되어 있다면 시스템 잘 작동하지 않음
- ◆ 모델이 너무 단순하거나(과소적합), 너무 복잡(과대적합) 하지 않아야 함

**Any Questions...**  
**Just Ask!**

