

# DATA Engineering

---

Essential

By Hapro



인공지능

AI

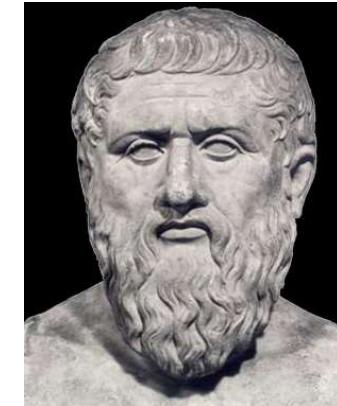
## 20도는 더운 온도다?

- 컴퓨터가 어제는 더운 온도라고 말했다가
- 오늘은 추운 온도라고 말한다면 인공지능인가?
- 인공지능이란 무엇인가?



# 인간처럼 생각한다

- 인간 처럼이란?
- 생각이란?
  - 생각하는 방법: 철학
  - 헤라클레이토스: BC535~
  - 플라톤: BC427~ 이데아에 대한 인식에 관한 고민 연역적 추론
  - 아리스토 텔레스: BC384~ 이데아에 대한 귀납적 추론
- 형이상학(Metaphysics)
  - 로도스의 안드로니쿠스가 아리스토 텔레스 저작물 정리(BC284)
  - 우리가 보고 느끼는 게 진짜인가?
  - 정말로 있다는 게 무엇인가?



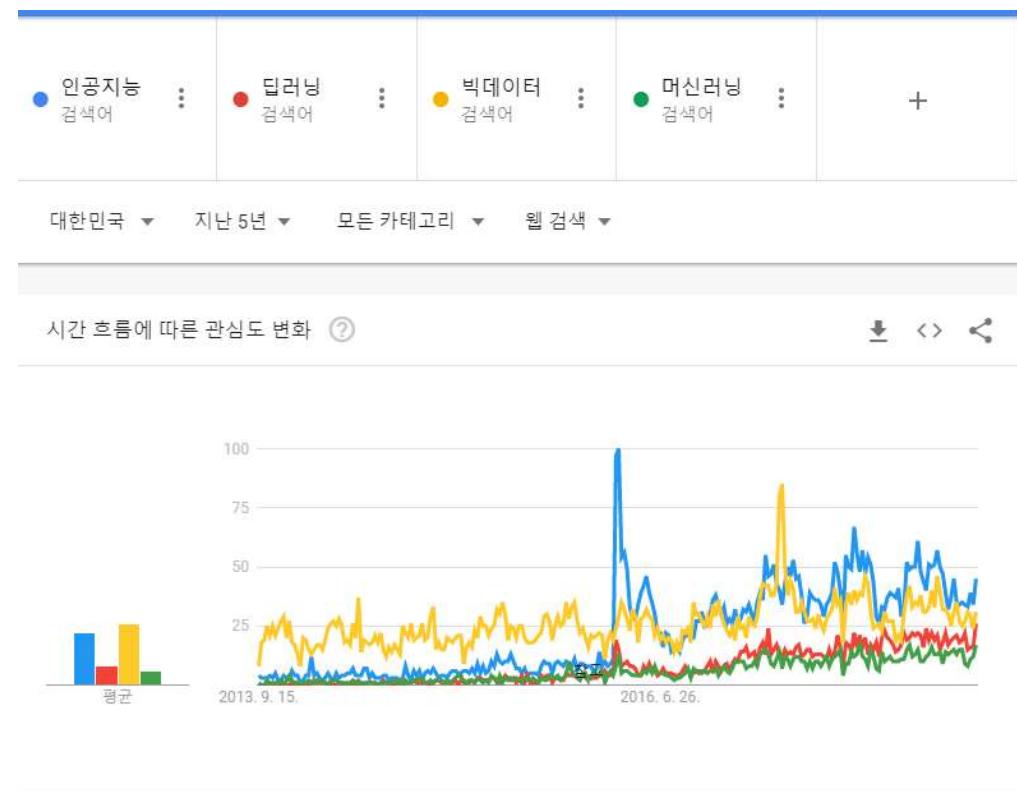
# 경험론

- 영국 경험론
  - 감각의 경험을 통해 얻은 증거들로부터 비롯된 지식을 강조하는 이론
  - 프랜시스 베이컨 1561~
  - 존 로크 1632~
- 영국 관념론
  - 실체 또는 우리가 알 수 있는 실체는 근본적으로 정신적이거나 정신적으로 구성되었거나 또는 비물질적
  - 인간의 생각, 특히 믿음과 가치가 사회를 어떻게 형성하는지
  - 토마스 힐 그린 1836~
  - 존 스튜어트 밀
- 일본
  - 1870 체계적 문명 개화 정책 시행
  - 독일 철학 및 영국 철학 번역서 등장
- 중국
  - 문화혁명



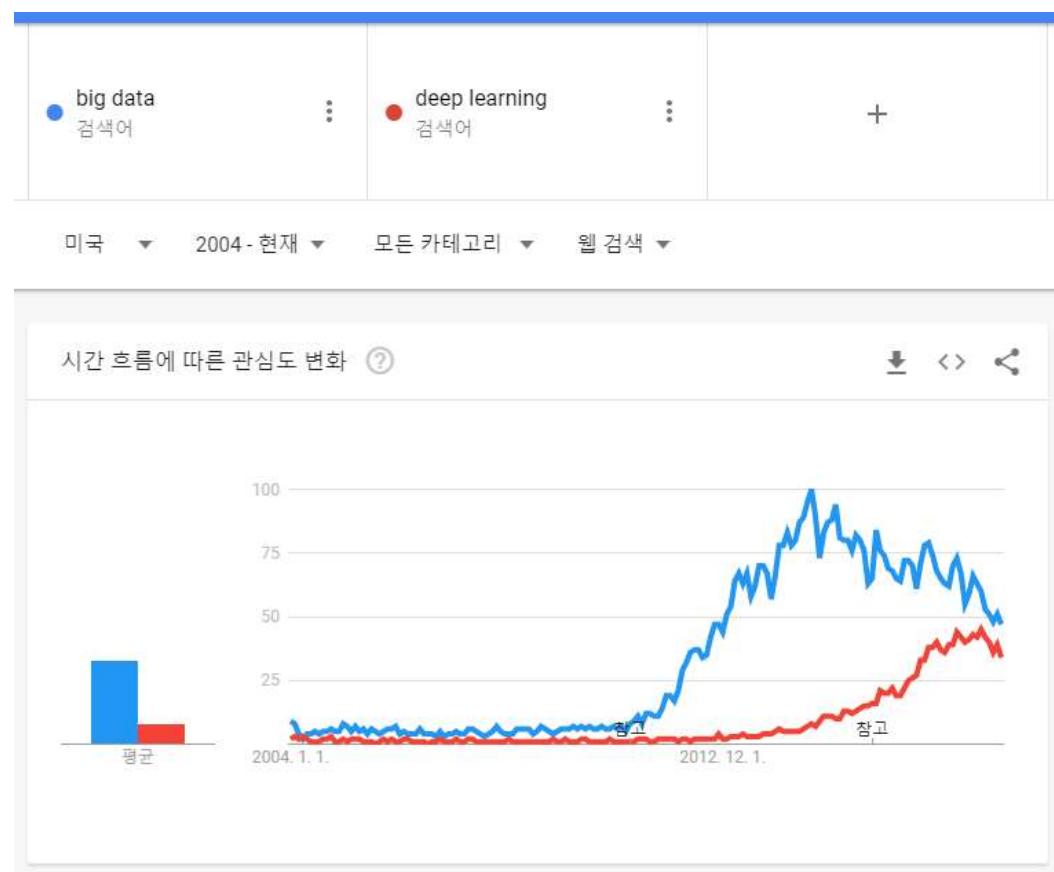
## 트렌드

- 데이터 산업
  - 인공지능>빅데이터>>딥러닝>머신러닝
  - 2016.3 이세돌-알파고 경기로 관심



# AI

- Artificial Intelligence vs Big Data



# AI의 주목

- 2014년 구글 딥마인드 인수
  - 데미스 허서비스와 12인의 AI전문가
  - 데미스 허서비스
  - 그리스계 아버지 싱가폴계 어머니 영국 태생
  - 8세때부터 프로그래밍 독학 체스천재
  - 15세때 게임회사 입사
  - 캠브리지 칼리지
  - 1993 유니버시티 칼리지 [인지과학](#) 박사과정에 진학하여 인간 기억의 메카니즘을 연구
  - 1995년 바둑에 입문 1년만에 4급
  - 1997년 딥블루가 체스 챔피언자리에 등극
  - 2011년 딥마인드 설립
  - 2014년 게임 인공지능으로 구글 경연진 감동시킴
  - 2016년 이세돌과 바둑매치
- "내성적인 소년이었습니다. 항상 뭔가를 생각하고 있던 것 같습니다". 그는 어린 시절을 이렇게 회상한다. "나의 뇌는 도대체 어떻게 말의 움직임을 생각해 낸 것일까. 자연스럽게 그런 의문을 갖기 시작했습니다. 그렇게 해서 나는, "생각"에 대해 생각하기 시작했습니다." (미국IT잡지 [Wired](#) 인터뷰中)

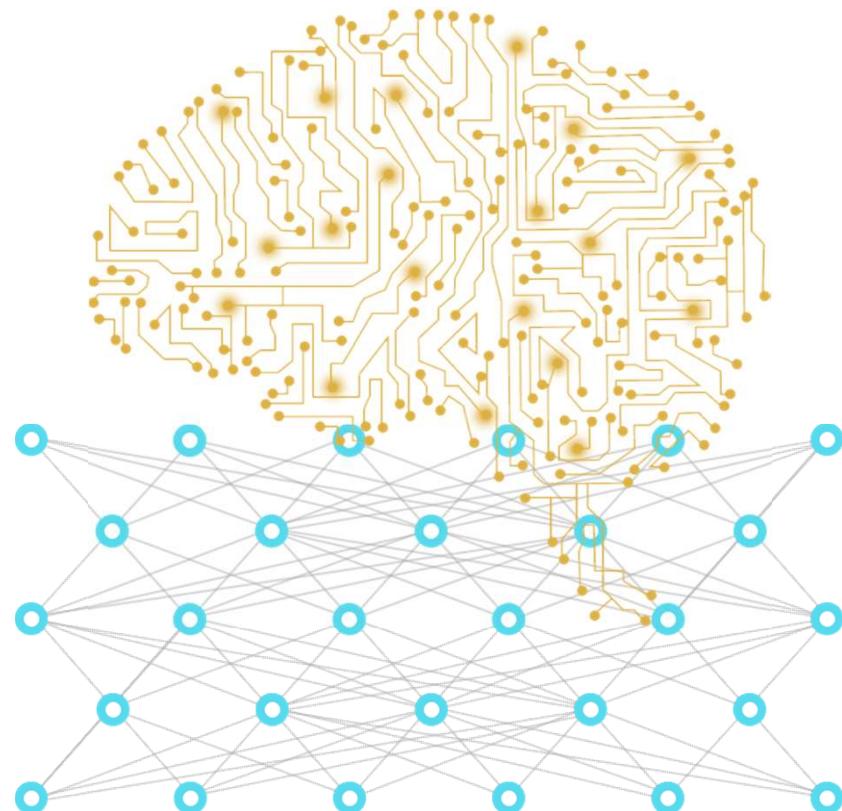


## 딥블루 vs 딥마인드

구분	딥블루(왓슨)	알파고
개발사	IBM	딥마인드
개발자	데이비드 페루치	데미스 허사비스
기록	체스 챔피언 1997 2013 제퍼디 퀴즈 우승	바둑 챔피언 2016
알고리즘	체스 전문 연산 엔진 선택가능 영역 셀프 러닝	범용 AI엔진 범용 학습 기능 활용
방식	전문가 학습	전문가 학습 + 자체 학습
알고리즘	게임트리	심층신경망

## 주목 이유

- 데이터 폭증의 시대
  - 데이터 분류 처리문제
  - 태깅 처리 필요량 증가
  - 컴퓨터로 인간 대체
  - 학습시켜 스스로 규칙을 형성





지능이란

Intelligence

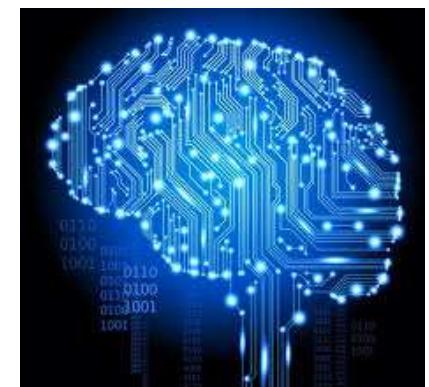
## 지능이론

- 지능 이론
  - 스피어만 요인설
    - 추론능력
    - 어휘능력
  - 카델의 유동지능/결정지능
    - -새로운 상황 적응하는 유동성적응
    - 학교,문화등 기술축적하는 결정성지능
  - 하워드가드너의 다중지능
    - 8가지 능력인 다중지능(신체,논리,운동,음악,언어,공간,자아,대인)
  - 스텐버그의 삼위일체 지능
    - 3가지 독립 지능(분석,창의,실제)



## 지능의 특징

- 적응적(Adaptive)
  - 지능이 높으면 다양한 상황과 문제에 융통성을 갖고 반응
- 학습능력(Learning Ability)
  - 지능이 높으면 신속하게 새로운 정보 처리.
- 선행지식(Use of Prior Knowledge)을 활용
  - 지능이 높으면 같은 실수를 반복 안한다.
- 상호작용(Interaction)과 조정
  - 지능이 높으면 여러가지 다른 정신 과정들의 융합하여 결과를 도출한다.
- 문화특수적(Cultural Specific)
  - 지능은 보편적이지 아니하다. 문화와 사회에 따라 다르다
- 지성은 감정과는 독립(Independent)하여 사고
  - 지능이 높으면 감정적 행동을 하지 않는다.
  - 지능이 높으면 주위에 선동되지 않는다.



# 알파고

- 개발 딥마인드(구글)
  - 발표 네이처 논문을 통해 발표(2016년 1월 )
- 학습
  - '전문가 지도학습'- 프로기사들의 기보 16만개를 입력
  - '자체경기를 통한 강화학습'
    - 스스로 학습하는 머신러닝 3천만개 추가 학습
  - 패턴이나 이미지분석 알고리즘
- 하드웨어
  - 판후이와의 대결
    - 1202개의 CPU와 176개의 GPU를 사용
  - 2016년 3월의 [이세돌 九단](#)과의 대결
    - GPU대신 48개의 [TPU](#)를 사용



# AI

- 정의
  - 인공 지능(Artificial Intelligence)
  - 외부 관찰자에게 인간과 비슷하게 보이는 '스마트한' 방법으로 소프트웨어를 작동시키는 폭넓은 방법, 알고리즘 및 기술
  - 인문학적 표현
- 유사어
  - 기계 지능(Machine Intelligence)
  - 컴퓨팅 지능(Computing Intelligence)

네 알파고도 실수를 하는군요~



## 지능의 핵심은 학습

- 알고 싶다?
- 무엇을?
- 완전한 답은 있는가?
- 판단하고 싶은가?



## 판단의 근거

- 추정
- 추정의 근거는 경험
- 모든 경험을 할 수 없으니 학습이 필요
- 학습으로 경험 강화
- 모든 학습량이 같을 수 없으니 객관화 필요
- 경험의 오염을 제거 하기 위한 통계
- 객관적 자료도 완전하지 않으니 확률
- ...
- 인간을 대신할 판단 머신 개발 AI



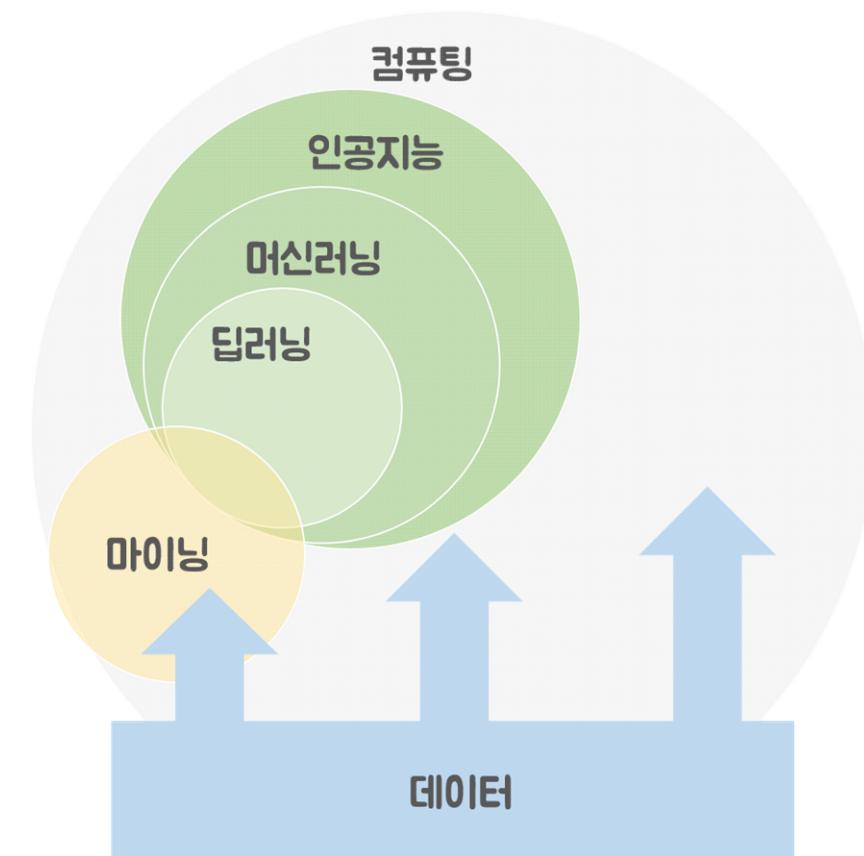


데이터 사이언스

Data Science

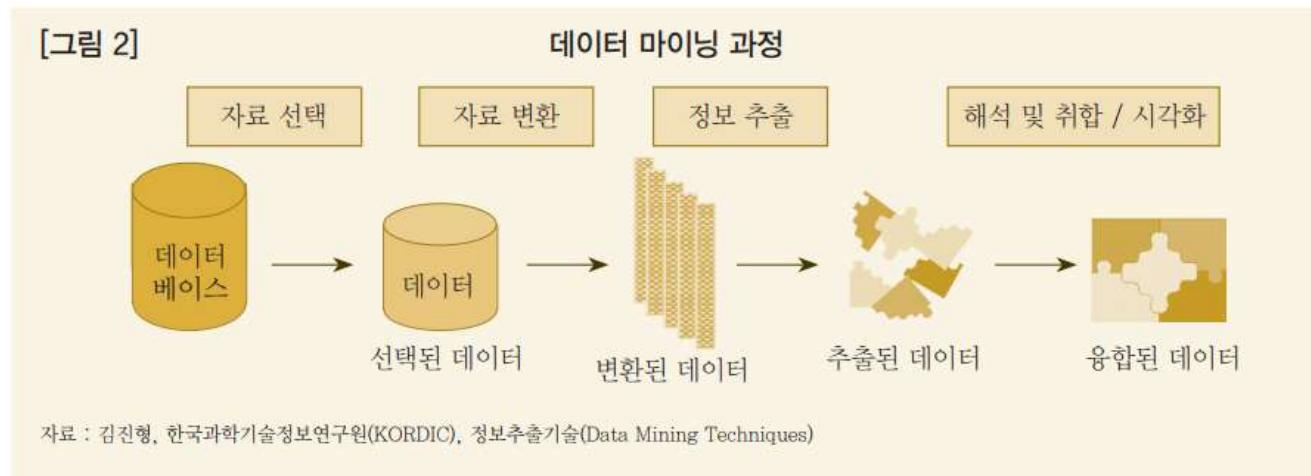
# 데이터사이언스

- 컴퓨팅(계산기반)
  - 단순 계산
- 인공지능(알고리즘 기반)
  - 인공적으로 만든 지능
- 머신러닝(기계학습 데이터기반)
  - 기계가 학습하다
- 딥러닝(심층학습)
  - 인간을 흉내 낸 기계학습
- 빅데이터
- 데이터 마이닝



# 데이터 마이닝(통계학)

- 필요 데이터로 정제
  - 발견할 지식의 종류에 따라서
  - 분류(Classification), 요약(Summarization), 군집화(Clustering) 등
  - 탐사할 데이터베이스의 종류에 따라 관계형(Relational) DB, 객체지향(Object-Oriented) DB 등
  - 탐사 기법에 따라서 기호처리식 인공지능적 방법론, 신경망적 방법



# 머신러닝(컴퓨팅)

- 목적

- 기계가 학습한다. 데이터에서 새로운 데이터가 제시 된 것에 대한 예측

- 특징

- 코딩에 의한 판단이 아닌 입출력 학습에 의한 판단

- 발생

- 1956 다트머스 컨퍼런스에서 인공지능 도입

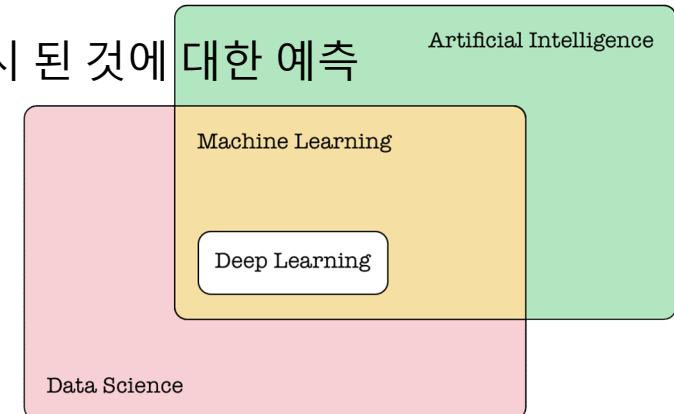
- 1959 아서 사무엘에의한 체커게임 논문에서 머신러닝 용어 사용

- 카네기 멜론의 톰 미첼 교수

- 머신 러닝의 정의

- 프로그램이 태스크  $T$ 를 수행함에 있어  $P$ 만큼 성능  $E$ 가 이뤄지면 태스크  $T$ 와 성능  $P$ 에 대해 경험  $E$ 를 학습했다고 할수 있다

- $\text{Learning} = P+E$



# 머신러닝의 발달

- **발달**

- 신경모형 패러다임: ,Deep Belief Network, Convolution Neural Network
  - 매컬록 피츠 :인공신경망이론->로센플래트의 퍼셉트론->연결주의론->1974풀워보스의 다층신경망학습이론->심층신경망(DBN),CNN등으로 발달
- 심볼릭 학습 패러다임
  - 1960 헌트의 인간처럼생각하는 프로그래밍->그래프 알고리즘->통계와 수학대신 논리와 그래프기반
- 지식집약 패러다임
  - 인지과학처럼 지식이 없는 상태에서 학습하는 신경모형 지향-> 한습된 지식 재활용
  - 1983 의사결정 트리 알고리즘:진행자가 의사결정 프로세스 확인가능
  - 1990실용적 머신러닝연구로 통계학과 논리학 데이터 마이닝도입
  - 1963러시아 블라디미르배프니크의 서포트벡터 머신등장으로 머신러닝의 핵심알고리즘으로 사용



학습

Learning

## 머신러닝 학습 종류

- **지도학습(Supervised Learning):**  $x \rightarrow y$  를 알려줌
  - 분류(Classification)  $y$  가 이산적
  - 회귀(Regression)  $y$  가 연속적(실수)
- **비지도학습(Unsupervised Learning):**  $Y$ 가 없는 경우
  - 군집화(Clustering):
  - 분포 추정(Underlying Probability Density Estimation):
- **반지도학습(Semisupervised learning):**  $Y$ 가 부분 없는 경우
  - 있는 경우를 보충해서 학습
- **강화학습(Reinforcement Learning)**
  - 지도가 없이 Action 에 대한 보상으로 가중치 설정
    - 자연보상 문제(당장 손해보더라도 왕을 잡아야하는데…)
    - 신뢰할당 문제(얼마나 보상해야 옳은 것인가?)
  - 관련학문
    - 인공지능, 제어이론, 신경과학, 운영과학, 행동심리학

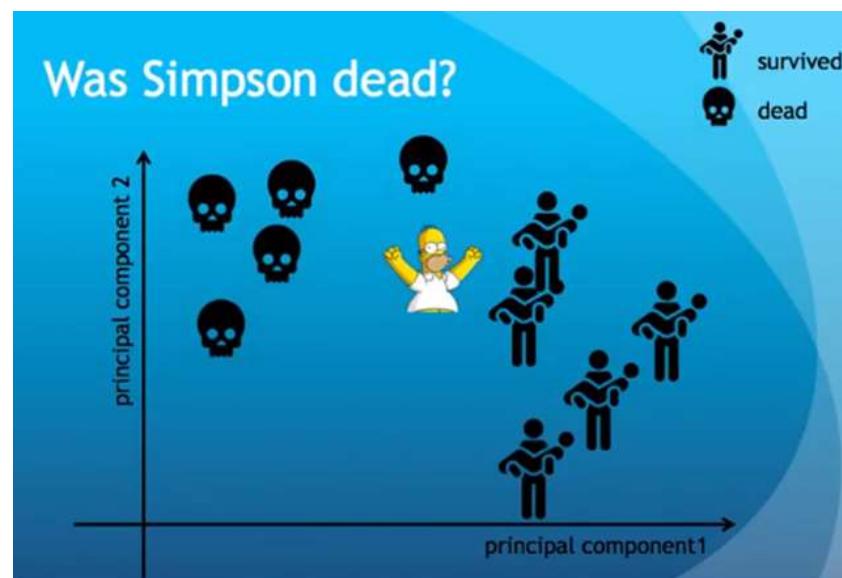
# 학습 내용

[표 1] 교사 학습, 비교사 학습, 반교사 학습의 차이

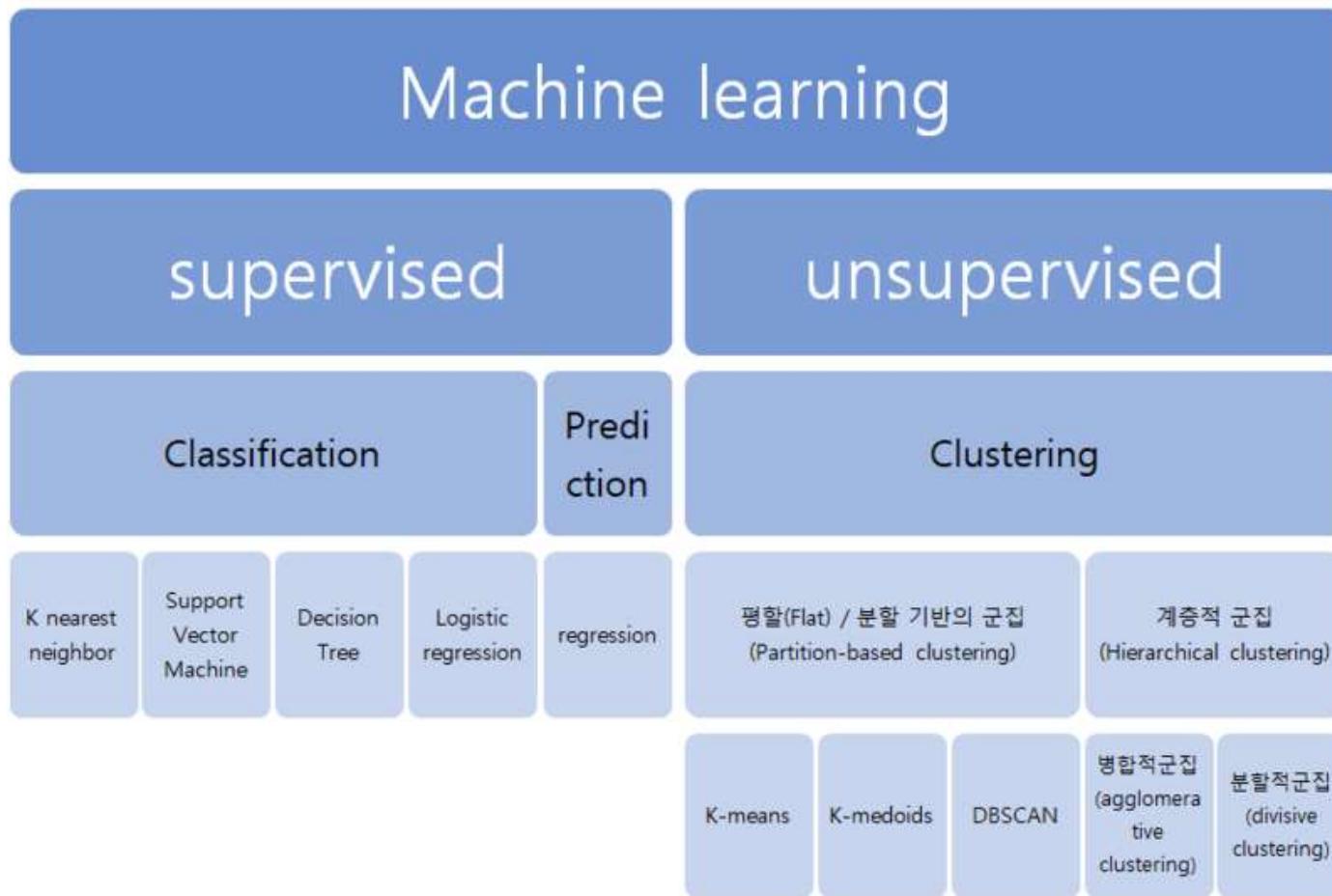
구분	내용
교사 학습	<pre> graph LR     subgraph Train [Training]         TD1[학습 데이터 Training Data] --&gt; FE1[특징 추출 Feature Extraction]         FE1 --&gt; ML1[머신러닝 알고리즘 Machine Learning Algorithm]         ML1 --&gt; EL1[예측 결과값 Expected Labels]         L1[목표 결과값 Labels] &lt;--&gt; FE1     end     subgraph Predict [예측 Predict]         TD2[새로운 데이터 Test Data] --&gt; FE2[특징 추출 Feature Extraction]         FE2 --&gt; PM1[예측 모델 Predictive Model]         PM1 --&gt; EL2[예측 결과값 Expected Labels]     end </pre>
비교사 학습	<pre> graph LR     subgraph Train [Training]         TD1[학습 데이터 Training Data] --&gt; FE1[특징 추출 Feature Extraction]         FE1 --&gt; ML1[머신러닝 알고리즘 Machine Learning Algorithm]     end     subgraph Analysis [분석 Analysis]         TD2[새로운 데이터 Test Data] --&gt; FE2[특징 추출 Feature Extraction]         FE2 --&gt; PM1[예측 모델 Predictive Model]         PM1 --&gt; LR1["가능성, 나은 표현 Likelihood or Better Representation"]     end </pre>
반교사 학습 <sup>18)</sup>	<pre> graph LR     D[데이터 Data] --&gt; TD[학습 데이터 Training Data]     TD --&gt; SL[교사 학습 Supervised Learning]     SL -.-&gt; UL[비교사 학습 Unsupervised Learning]     L[목표 결과값 Labels] --&gt; TD     TD --&gt; SL     SL --&gt; D     SL --&gt; UL </pre> <p style="text-align: right;">         → 교사 학습(Supervised Learning)          ...→ 비교사 학습을 교사 학습에 추가          ...→ 자가 학습(Self-Training)     </p> <p style="text-align: right;">         반교사 학습          Semi-Supervised Learning     </p>

# 머신러닝이란

- 기계학습 ML
  - 학습(Learning) = 표현(Representation) + 평가(Evaluation) + 최적화(Optimization)
- 진행
  - 주어진 방법을 가지고
  - 데이터를 처리하여
  - 최적값 도출을 프로그램이 인식



# 머신러닝 다이어그램



# 지도학습

- 지도학습
  - 예측
    - 선형 회귀(Linear Regression)
  - 분류
    - k-최근접 이웃(k-Nearest Neighbors)
    - 서포트 벡터 머신(SVM, Support Vector Machine)
    - 결정 트리(Decision Tree)와 랜덤 포레스트(Random Forest)
    - 로지스틱 회귀(Logistic Regression)
  - 신경망(Neural Network)
- 준지도 학습
  - 중첩데이터에서 주로 사용
  - 사진속 사람 환경에 따른 분류
-

## 비지도 학습

- 군집(Clustering)
  - K-평균(k-Means)
  - 계층 군집 분석(HCA, Hierarchical Cluster Analysis)
  - 기대값 최대화(Expectation Maximization)

시각화(Visualization)와 차원 축소(Dimensionality Reduction)

- 주성분 분석(PCA, Principal Component Analysis)
- 커널 PCA(Kernel PCA)
- 지역적 선형 임베딩(LLE, Locally-Linear Embedding)
- t-SNE(t-distributed Stochastic Neighbor Embedding)

- 연관 규칙 학습(Association Rule Learning)
  - 어프라이어리(Apriori)
  - 이클렛(Eclat)

## 강화학습

- 강화학습
  - **에이전트(Agent)**가
  - **환경(Environment)**을 관찰해서
  - **행동(Action)**을 실행하고
  - **보상(Reward)**을 받는 것을 통해
  - 정책(Policy)이라고 부르는 최상의 전략을 스스로 학습

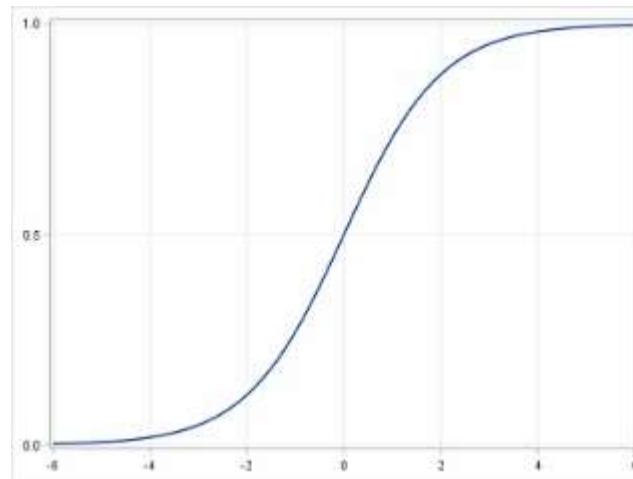
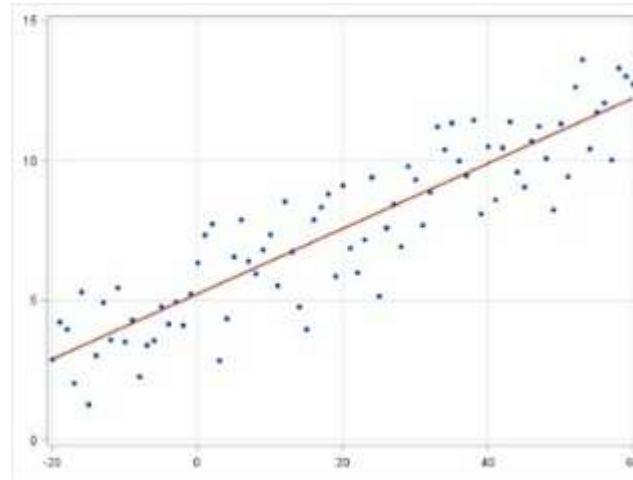
# M.L 알고리즘

---

By Hapro

## 선형회귀

- 선형회귀
  - 연속변수 추정
- 로지스틱 회귀
  - 범주형 변수 추정

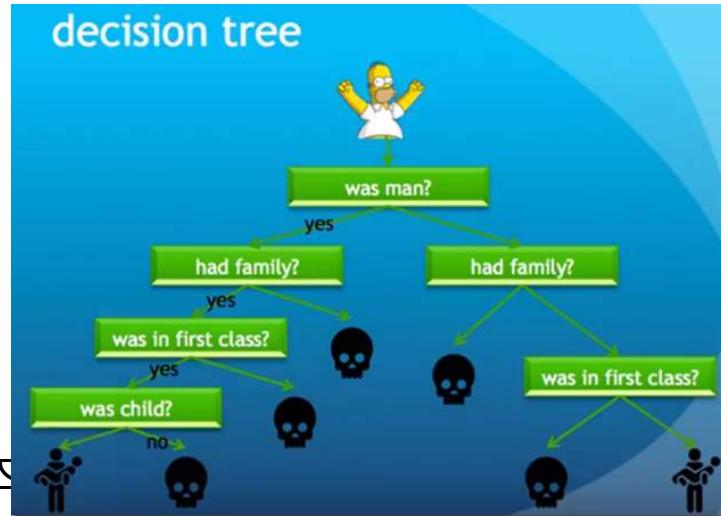


## DT 의사결정트리

- Decision Tree

- 데이터 분석을 통한 트리작성

- 트리탐색을 통한 판단



## Random Forest

- DT가 여러 개 모여서 Forest 를 이룬 형태
  - DT의 단점 보완
  - 여러 가지 트리중 많이 나오는 트리로 판단



## 나이브베이즈

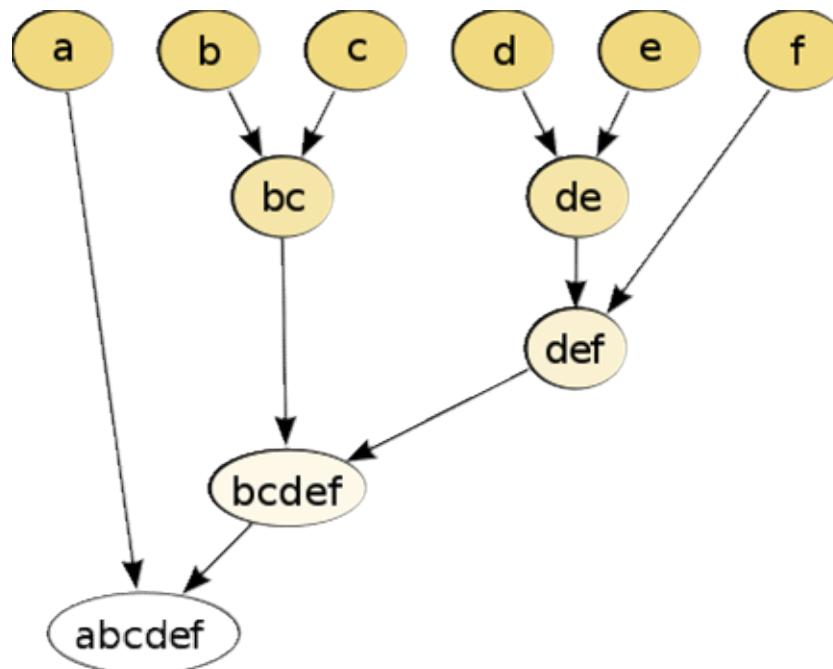
- 각 특성이 생긴 사전 확률을 통해 사후 확률 계산법
- 심슨의 성별 나이 등의 사전 확률이 주어지면 사후 확률을 통한 분류

### Naive Bayes


$$P(\text{survived} \mid \text{old, man}) = \frac{P(\text{old} \mid \text{survived} \cap \text{man}) * P(\text{man} \mid \text{survived}) * P(\text{survived})}{P(\text{old} \mid \text{man}) * P(\text{man})}$$

## 계층적 군집화

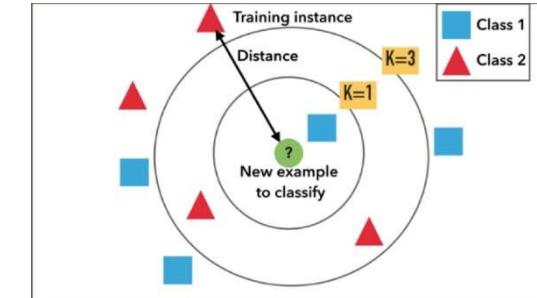
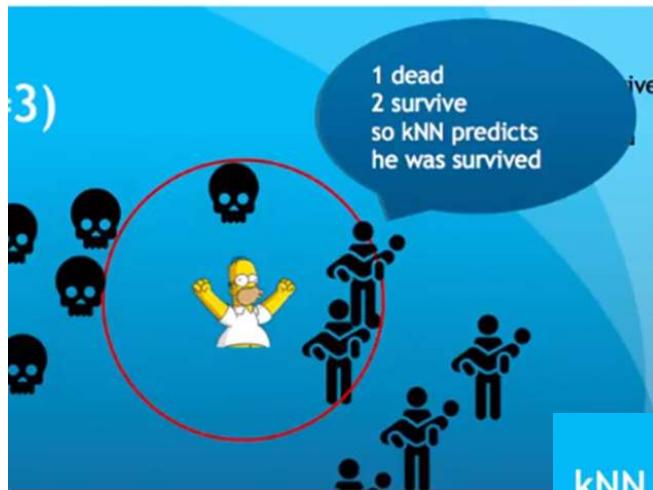
- 계층적 군집화(Hierarchical clustering)
  - 트리 구조 덴드로그램(dendrogram)를 이용해 시각화
  - 클러스터링을 정제하거나 최대화



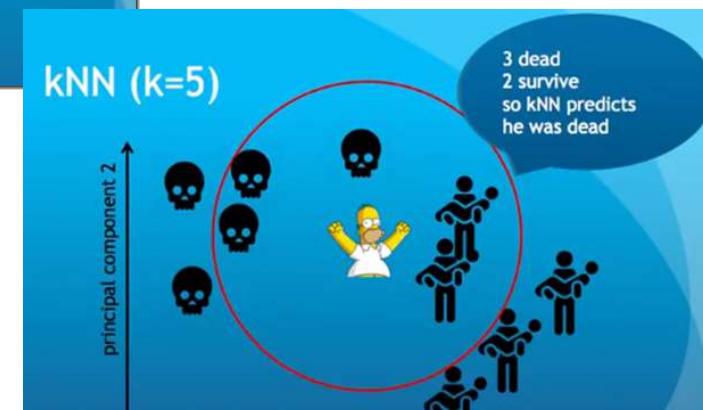
# 머신러닝 알고리즘

- KNN(최근접 이웃) K-Nearest Neighbor

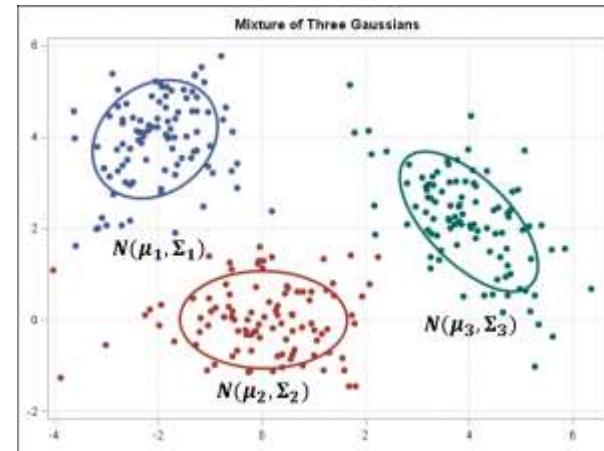
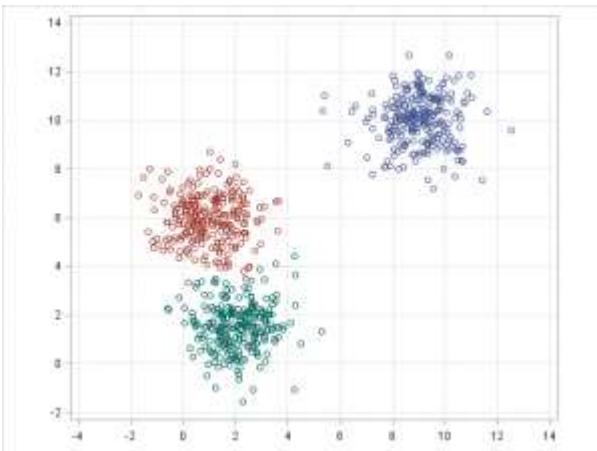
- K 기반 분류
  - K=3



- K=5

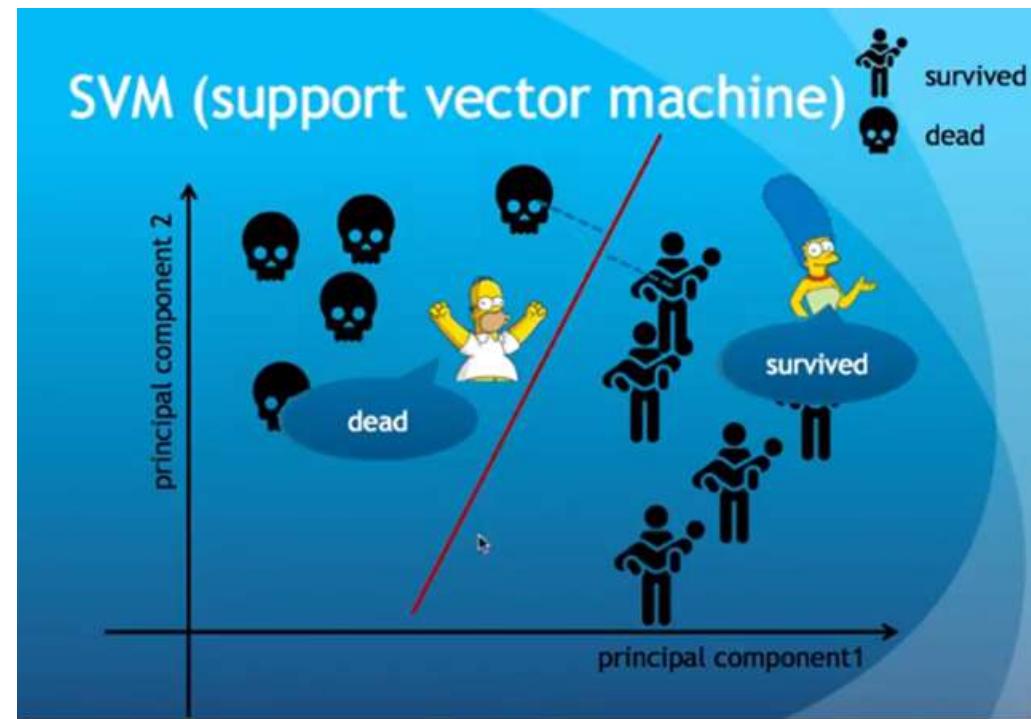
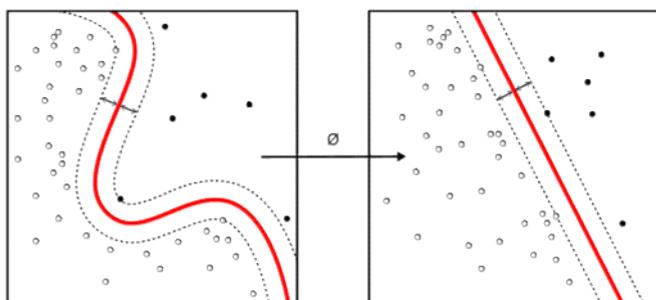


# K-mean clustering



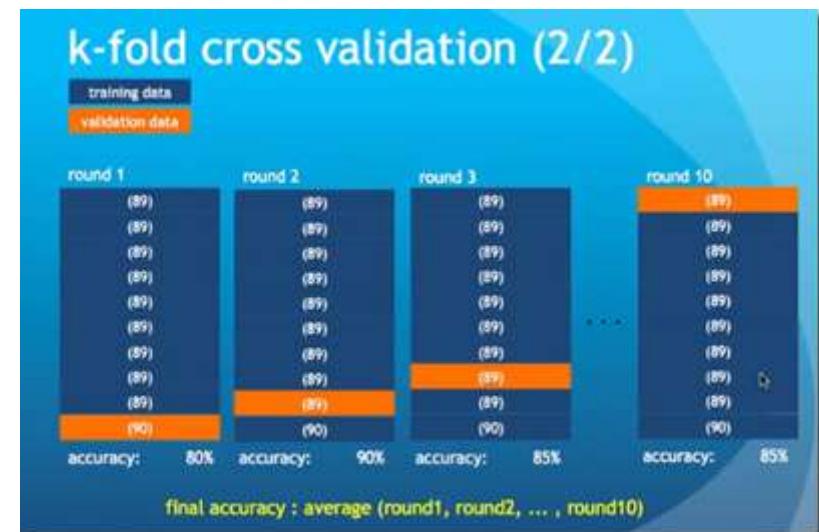
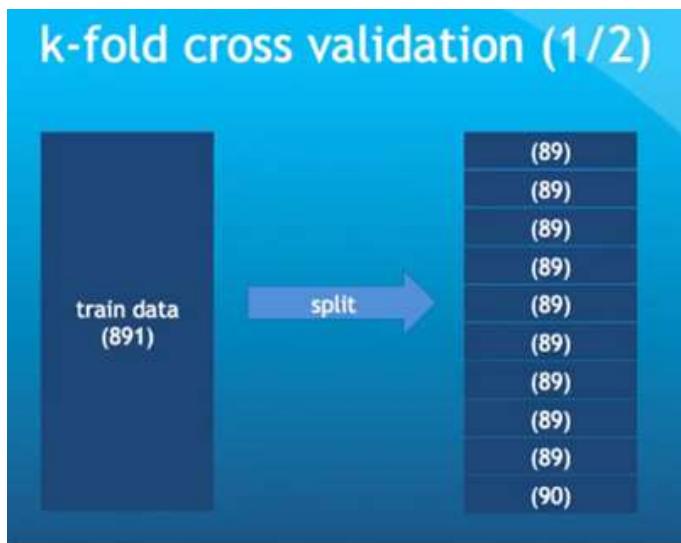
## SVM서포트벡터머신

- 분류가 될 수 있는 가장 먼 면을 계산
- 그 선을 기반으로 분류

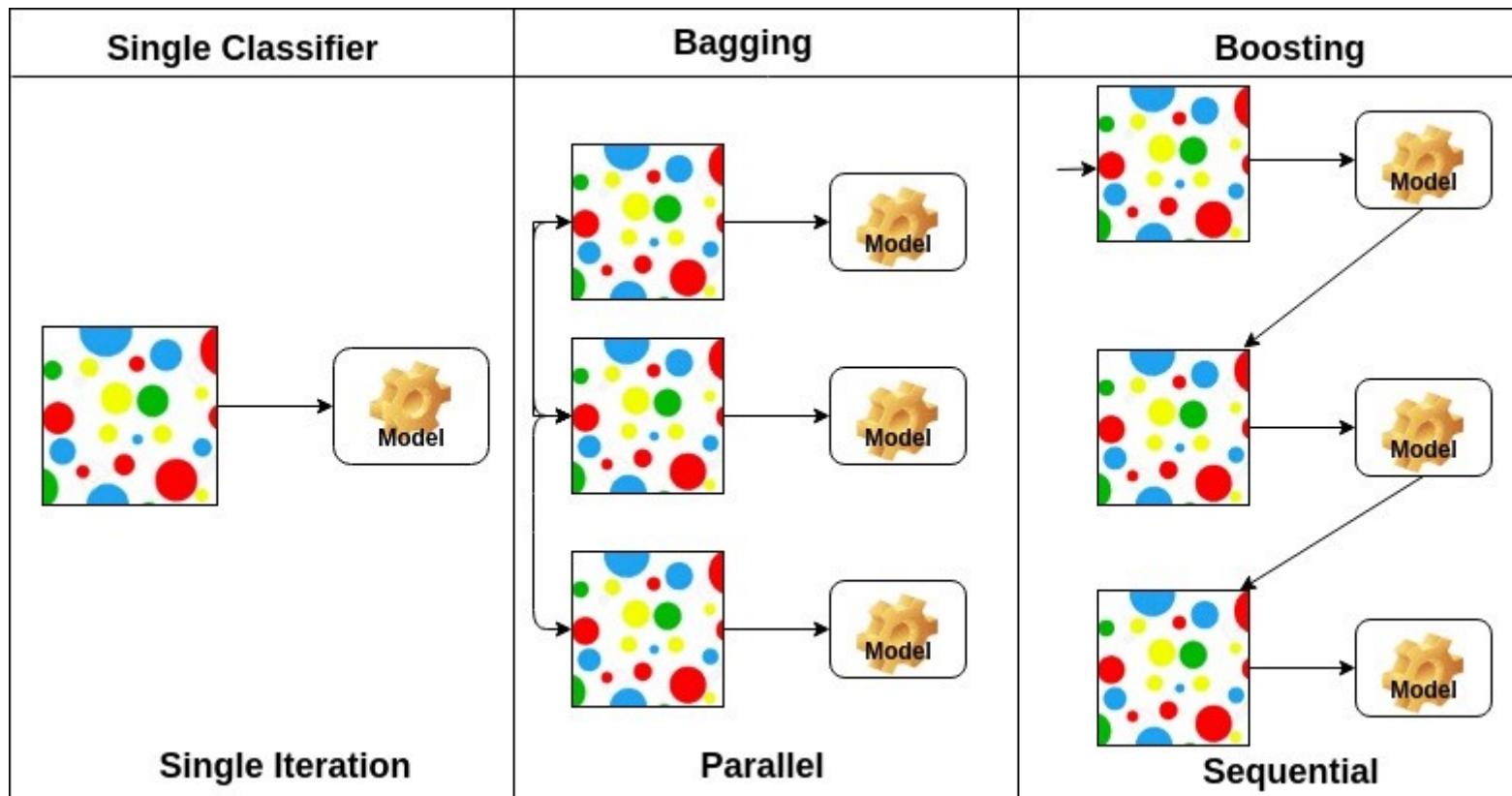


## 검증 K-fold cross Validation

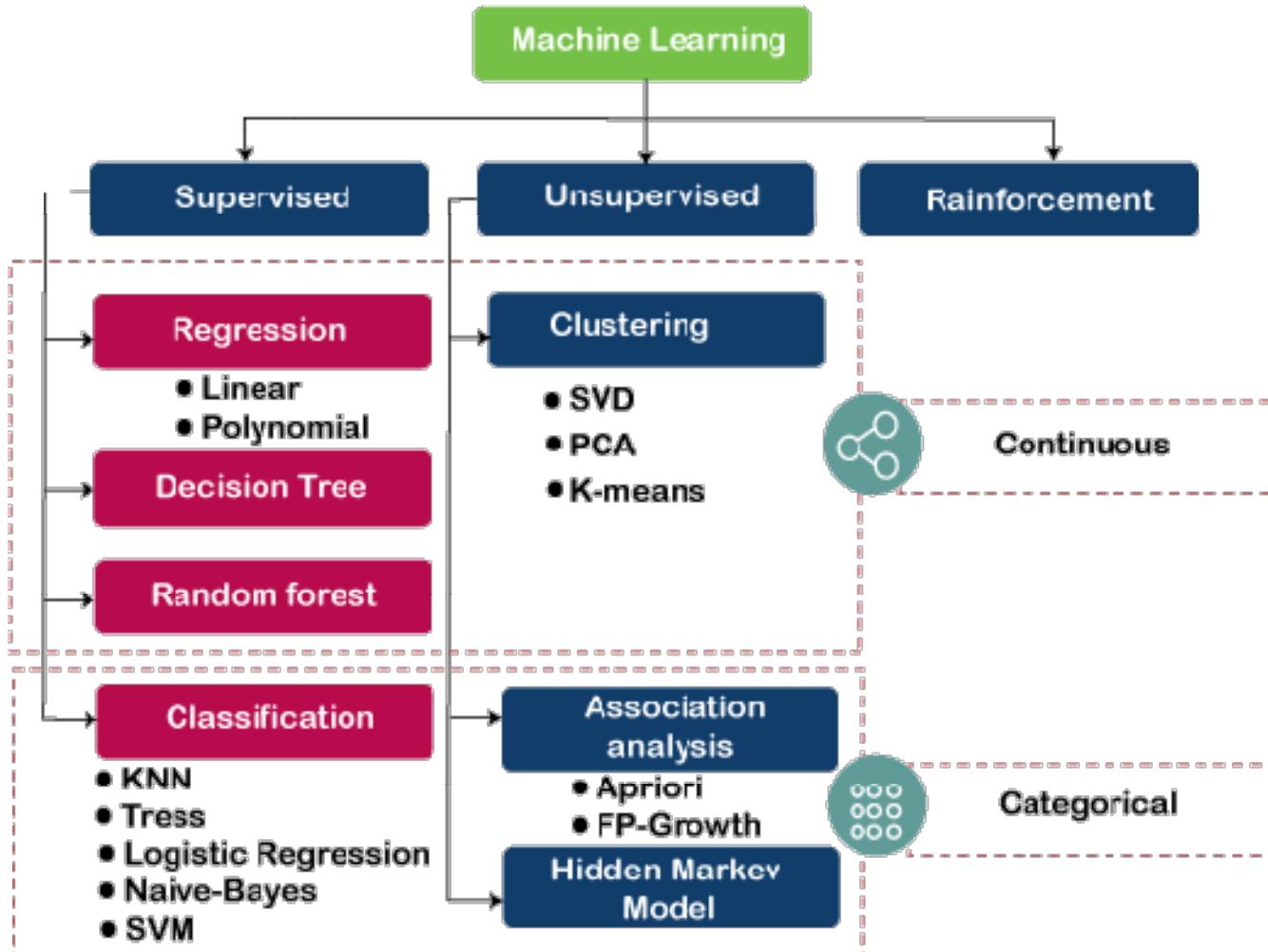
- K폴드 교차 검증
  - 검증자료 추출에 관한 문제
  - 기존데이터의 일부 추출시 Bias 가 있으면 테스트 오류발생
  - K개로 나눈다음 해당 조각에서 일부 추출



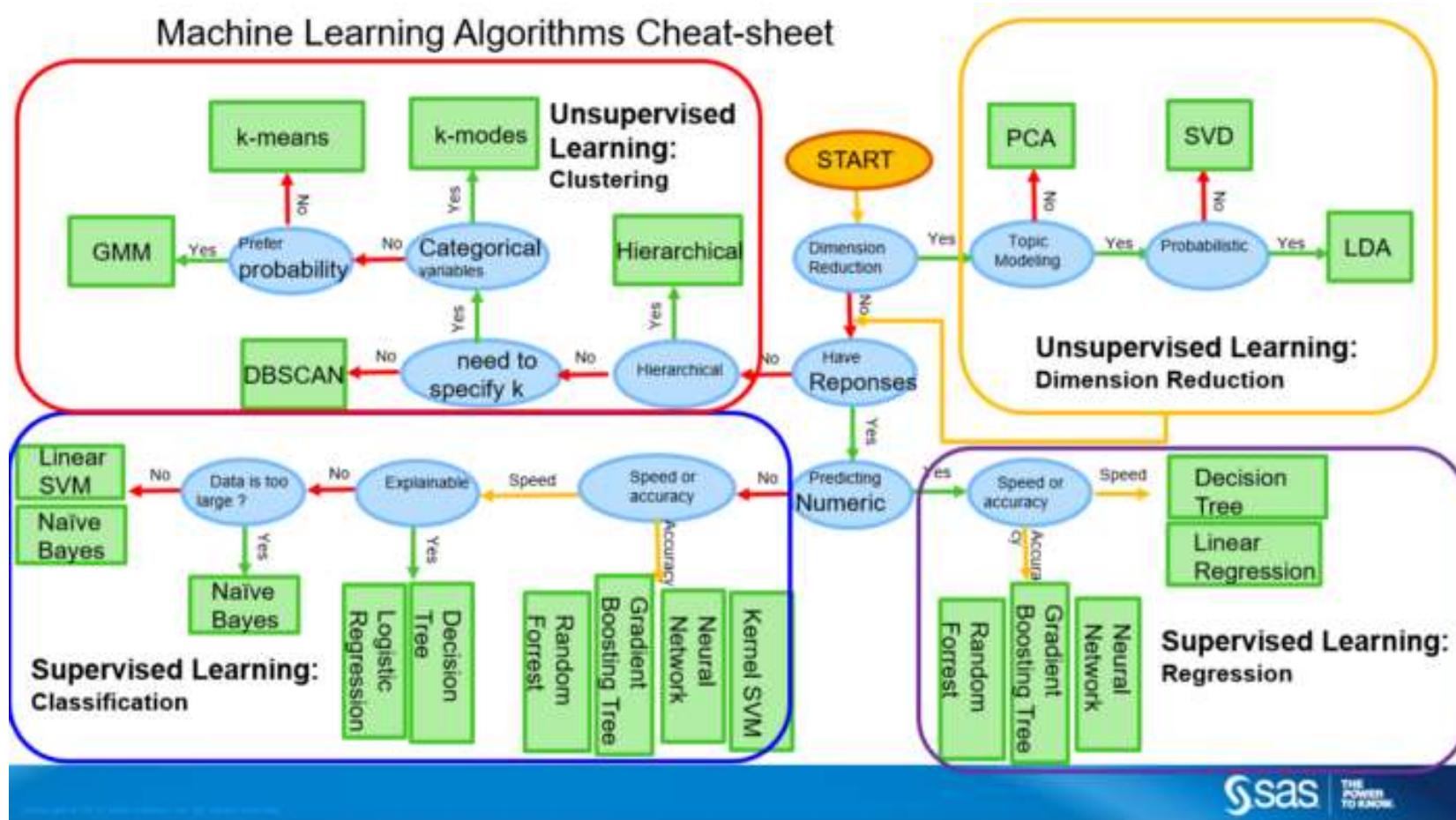
# 배깅/부스팅



# 머신러닝



# 머신러닝 순서도



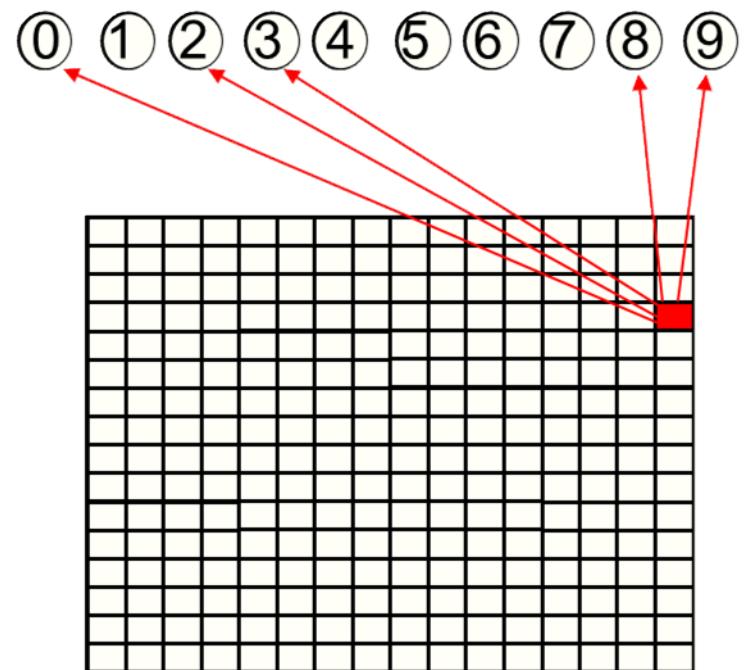


단순 학습모델

---

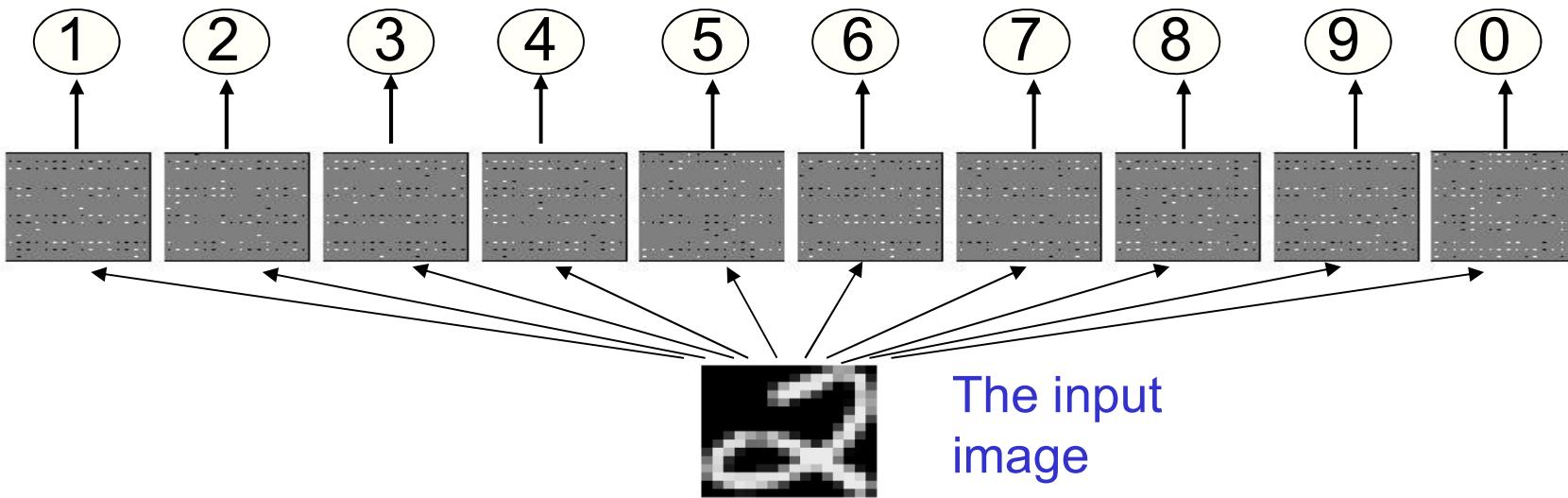
## 단순 학습 모델

- 2레이어 학습모델
  - top layer의 뉴런은 알려진 모양을 나타낸다.
  - bottom layer의 뉴런은 pixel intensities(픽셀의 강도)를 나타낸다.
- 픽셀은 픽셀강도를 높일 수 있음.
  - 각 학습마다 픽셀 강도 표시.
- 가장 많이 보이는 픽셀 표시.



## A simple example of learning

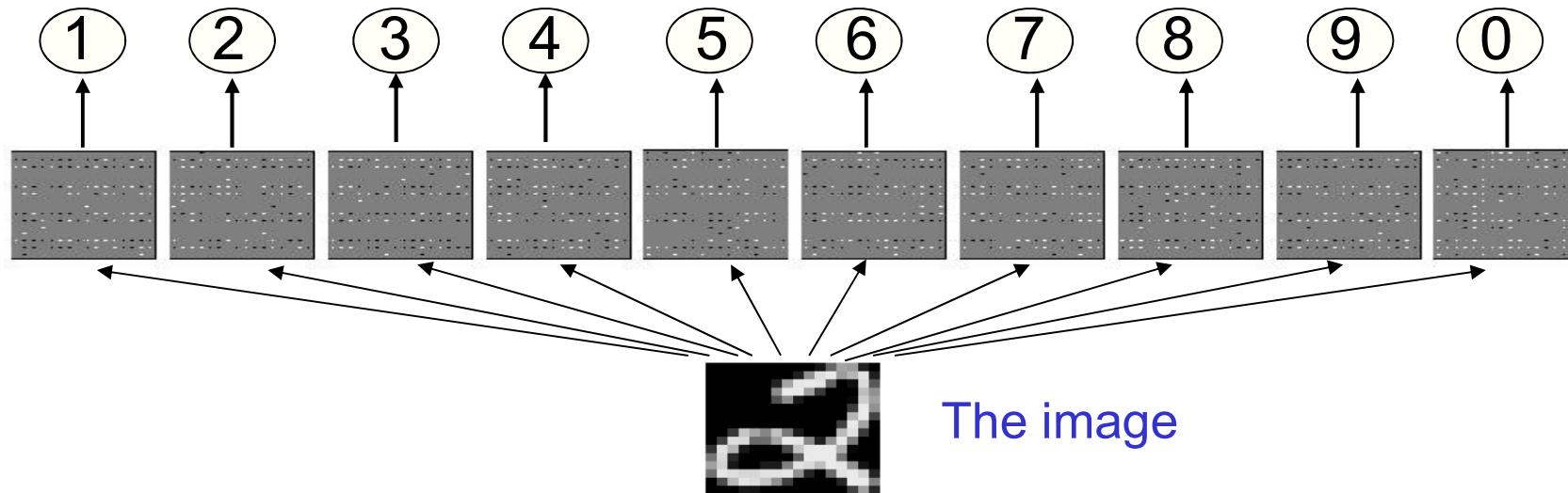
- How to display the weights



- 각 출력 유닛에 입력 이미지의 map을 지정하고, Map에 해당 픽셀에서 각 픽셀에서부터 전달되는 Weight를 표시
- Weight의 크기와 신호를 표시하는 색을 표현하는 영역에 검정과 흰색 칠(blob)을 사용.

## A simple example of learning

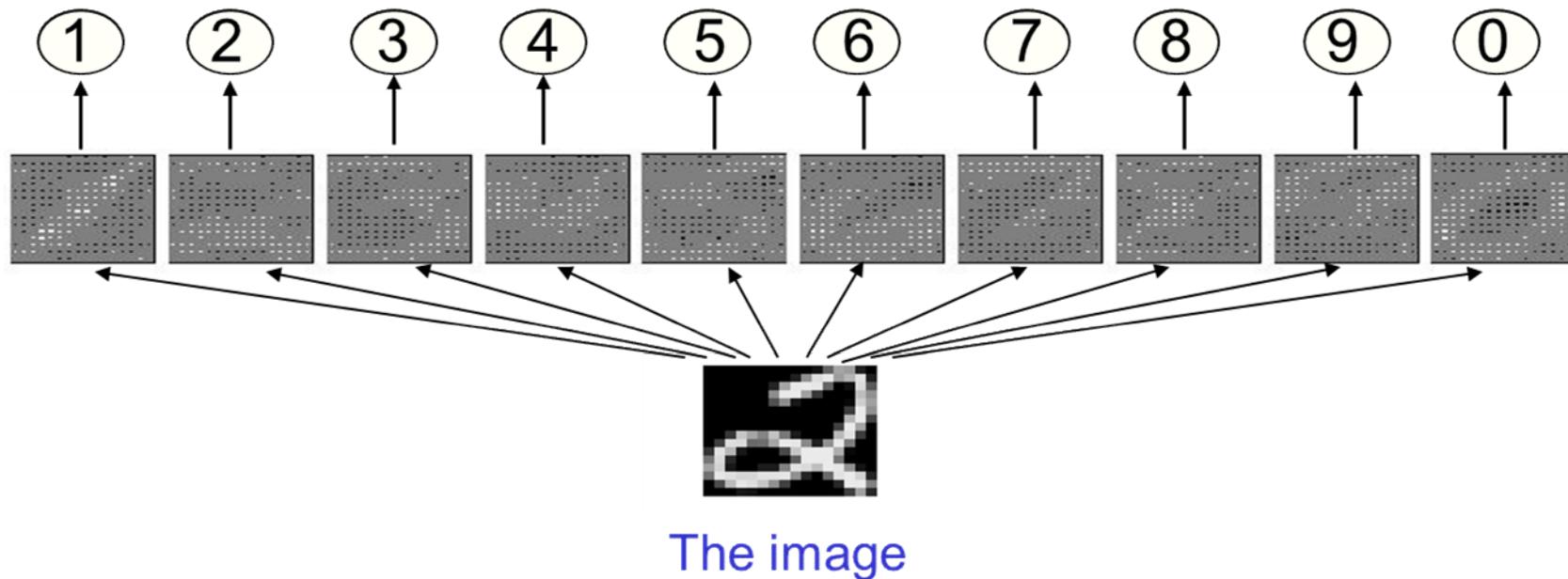
### How to display the weights



- 각 네트워크에 이미지를 보여주고 active 픽셀에서 올바른 클래스에 weight를 증가시키고 그렇지 않은 클래스는 active 픽셀에서 weight를 감소시킨다.

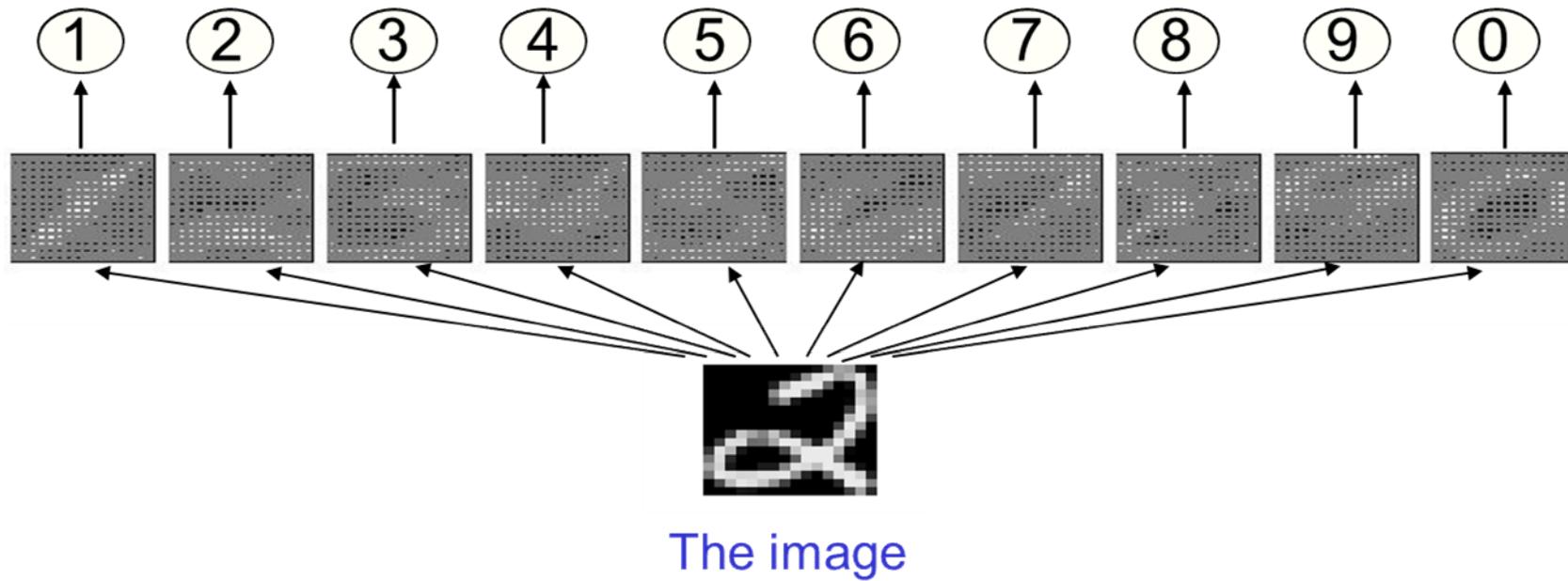
## A simple example of learning

### How to display the weights



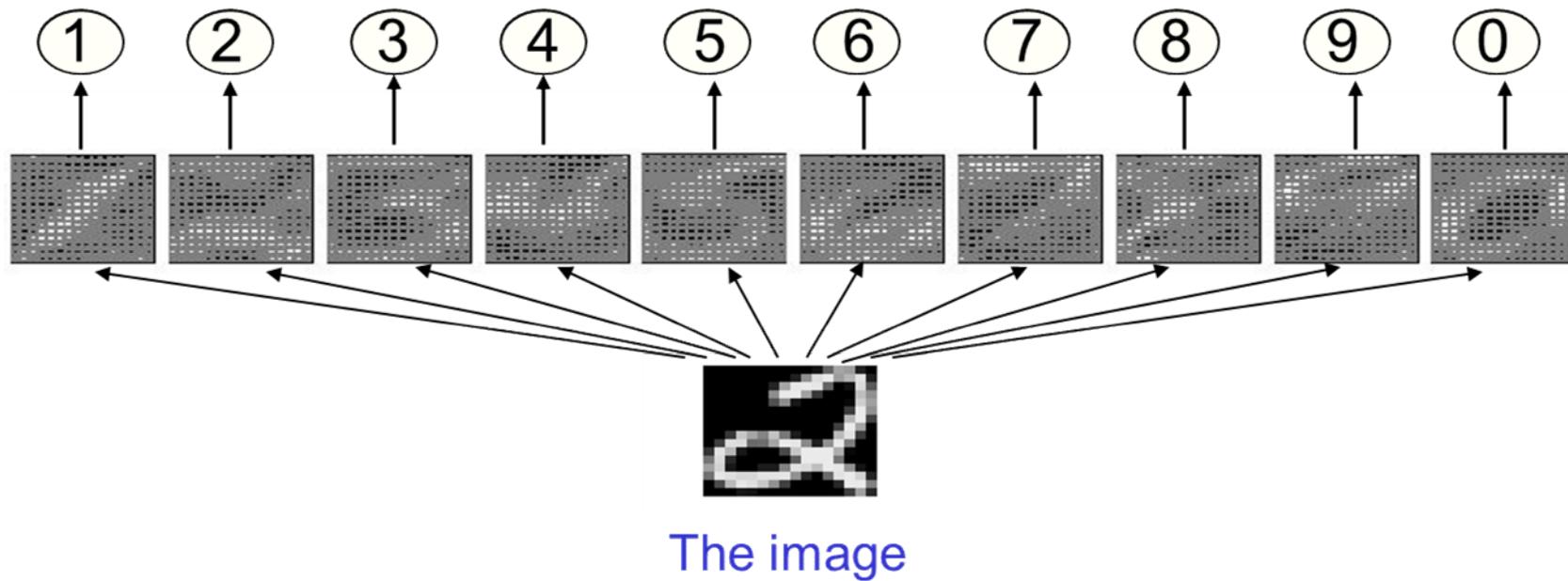
## A simple example of learning

### How to display the weights



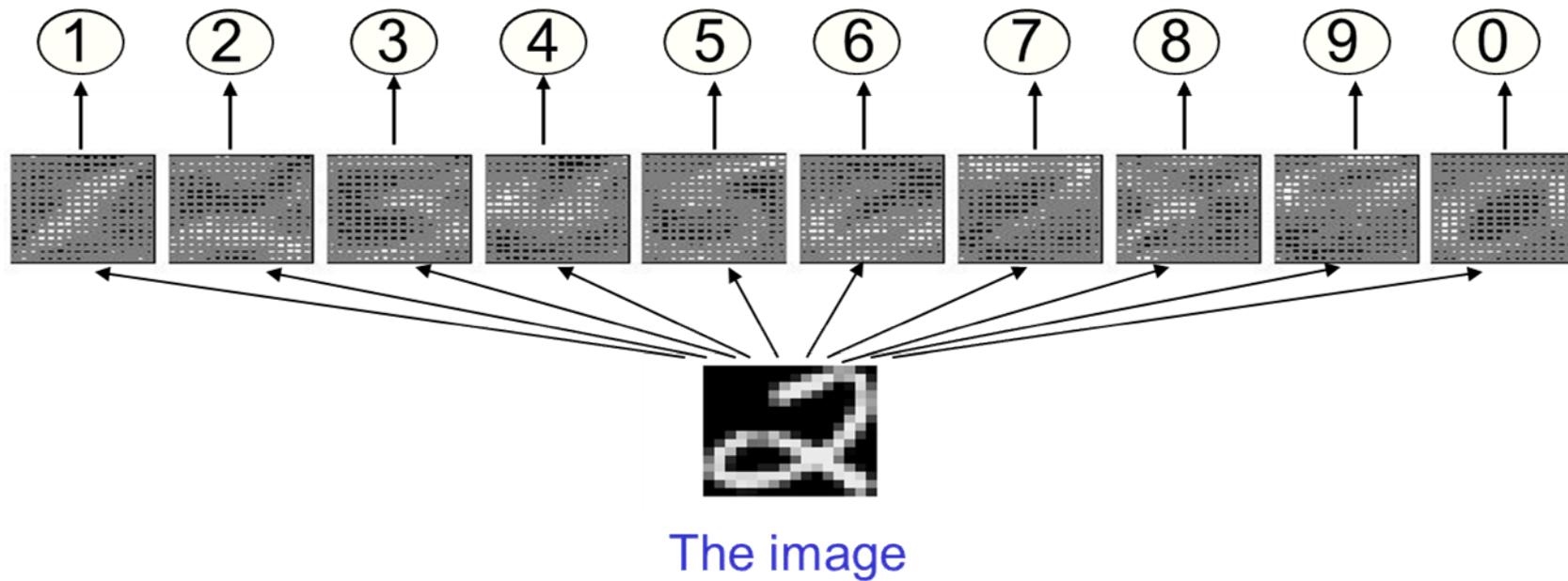
## A simple example of learning

### How to display the weights



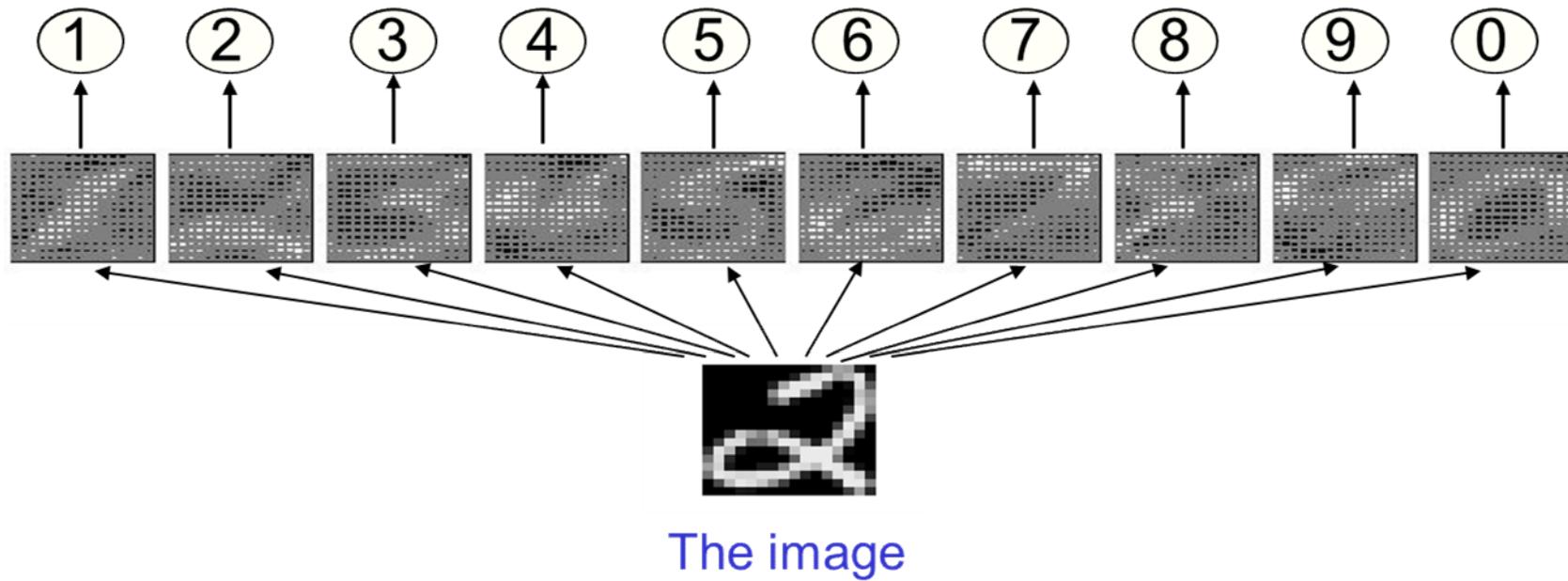
## A simple example of learning

### How to display the weights



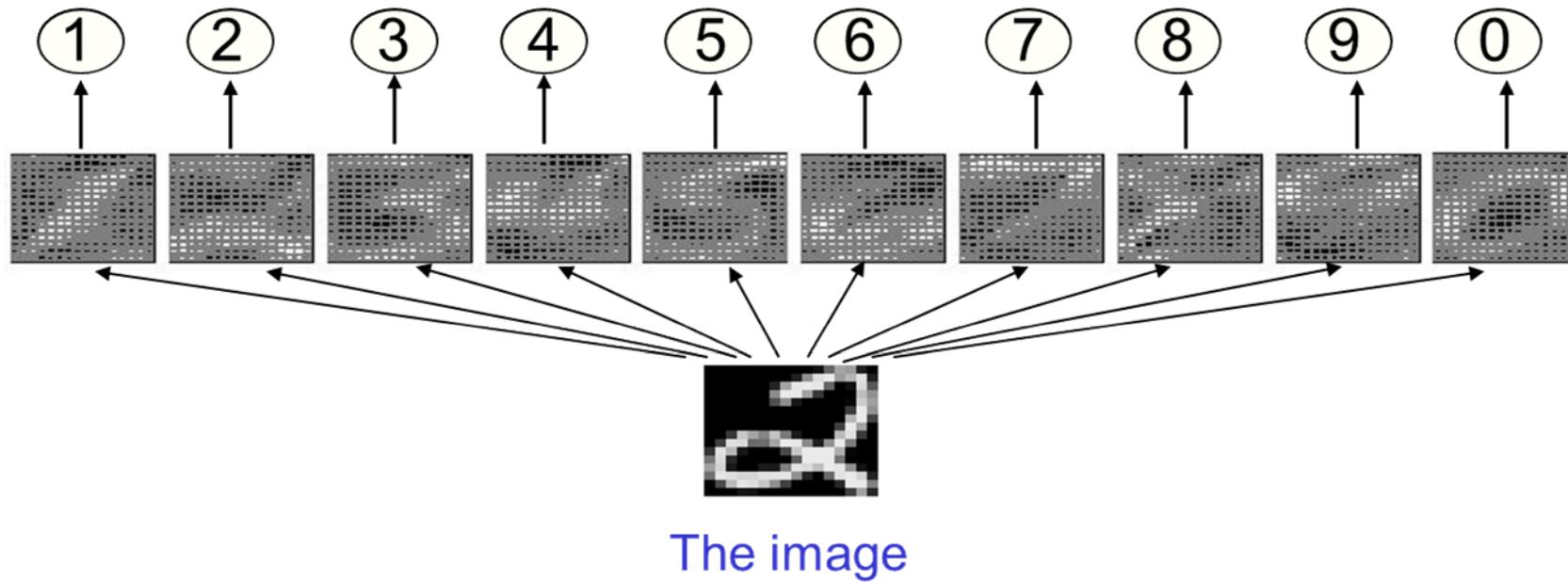
## A simple example of learning

### How to display the weights



## A simple example of learning

The learned weights



- 잉크와 가장 오버랩 많이 되는 템플릿이 출력값으로 표시.
- Hand-written digits(손으로 쓴 숫자)을 다루는 방법은 너무 복잡해서 전체 모양을 매칭하기에 간단한 템플릿에 의해 캡처되기 힘들다.
  - 숫자의 허용되는 모든 변화를 모두 캡쳐하기 위해서, 숫자를 구성하는 특징을 배울 필요가 있다.

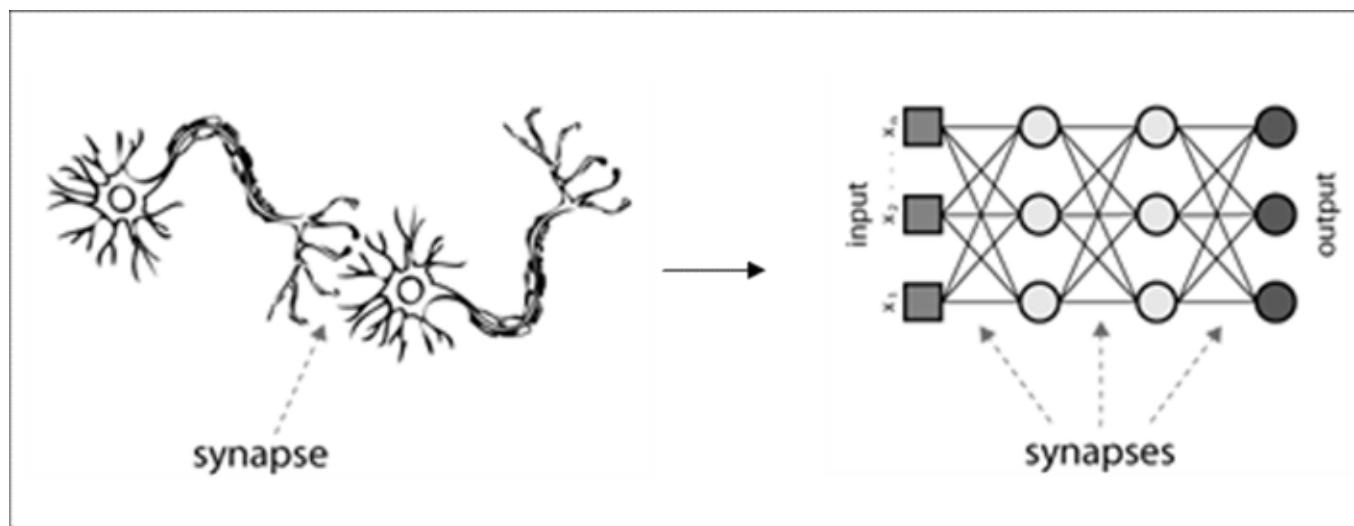


퍼셉트론

Perceptrons

# 딥러닝

- 신경망
  - 뇌의 신경 세포 작용을 느슨하게 본 딴 머신 러닝의 한 유형



# 신경망 연구

- 파블로프의 실험으로부터 보상이 뉴런 연결 강화를 통해 지능이 발달하는 것을 모방

## 헵의 이론

캐나다 신경과학자 도널드 헵은 1949년, 파블로프 실험으로부터 두뇌가 특정 뉴런 연결을 강화해 학습한다고 추정했다. 인공지능 연구자들은 이를 모방해 인공신경망을 개발했다.

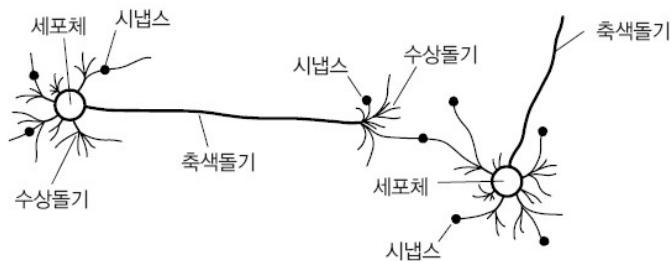
- 음식 볼 때 활성화
- 증소리 들을 때 활성화



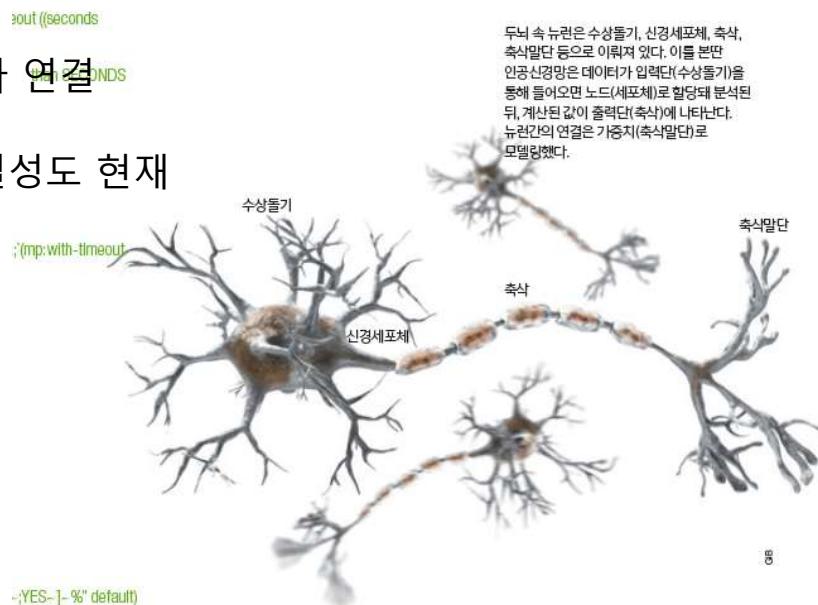
```
;; with-timeout & y-or-n-p
;; (defun with-timeout-f (timeout bodyf timeoutf)
  (let ((done nil) (process (current-process)))
    (make-process (format nil "Timeout monitor for ~
                                  ~S")
                  (lambda ()
                    (sit
                     (unless done
                       (process
                        (funcall timeoutf)))))))
    (unwind-protect (funcall bodyf)
      (funcall timeoutf)))))))
```

# 생물학적 신경망

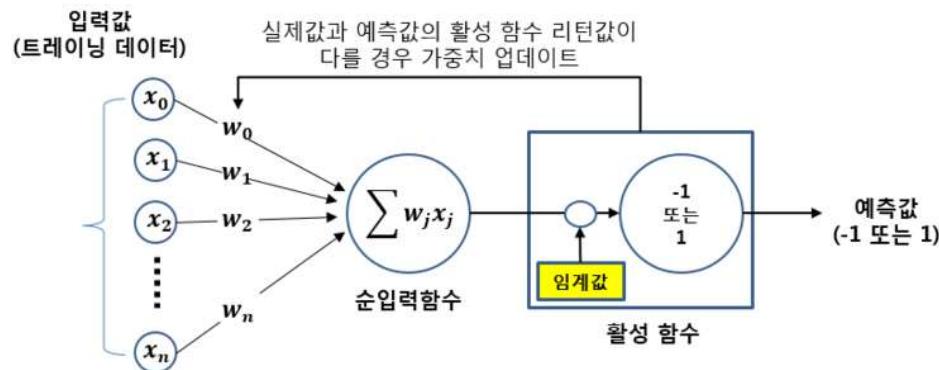
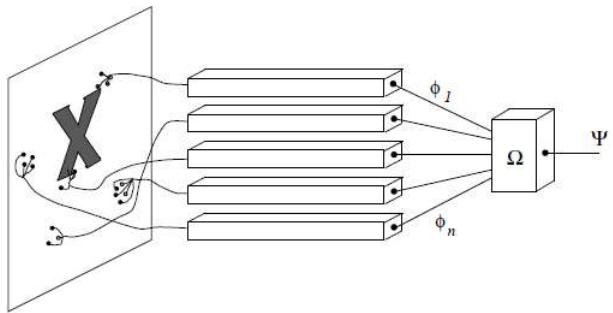
- **뉴런(신경세포):** 신경망에서 가장 기본이 되는 단위
- **뉴런의 기본적 기능:** 정보의 수용, 연산처리, 출력의 전송, 비선형성, 다입력 1출력, 흥분성과 억제성의 시냅스 결합, 순은, 적응, 피로 등
- 뉴런은 1000개에서 100,000개의 다른 뉴런들과 연결
- 인공신경망에서는 인간 두뇌의 1% 가량의 연결성도 현재 수준에서 원활히 처리하기 힘듦.



[그림 6-1] 생물학적인 신경망

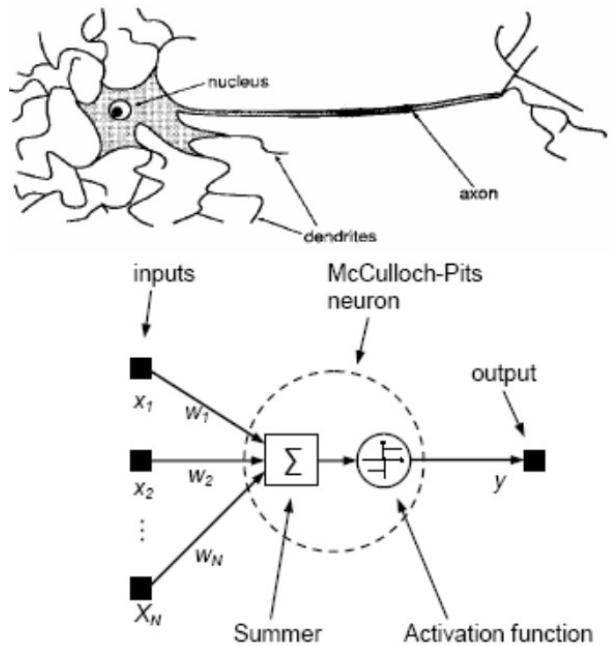


# Perceptrons 신경망

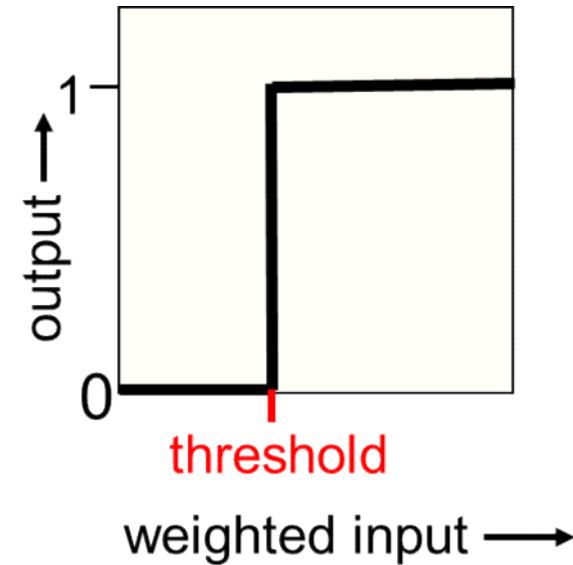


- 구조
  - 하나의 뉴런에 의해 생성되는 신경망 구조
  - $K$ 개의 입력, 1개의 bias,  $K+1$ 개의 weight로 이루어짐.
  - $s$ 는 특징벡터  $x$ 에 대한 선형함수형태
  - 선형 분류기
  - 활성함수는 계단함수(threshold activation functions).

# McCulloch-Pitts Neuron



- 1943년 McCulloch와 Pitts에 의해 제안.
- 초기 신경 시스템 모델 중 가장 잘 알려짐.



- 인간의 신경활동을 2진 단위(binary unit)들의 결합으로 설명
- 각 뉴런의 입력, 출력은 1 or 0
- 1은 뉴런이 흥분 상태
- 0은 뉴런이 정지상태

## 예제 AND 연산

- 예제
  - 단층 퍼셉트론을 이용하여 AND 연산을 학습시키는 예제
  - 학습률은 0.05
  - 임계치는 0
  - 허용오차는 0과 0.1 사이의 적당히 작은 값으로 설정
  - 단층 퍼셉트론에서는 허용오차 크기 계산 무의미
- 준비
  - 학습 데이터 정의
    - 학습시킬 데이터가 AND 로직에 의해 출력될 것을 알고 있음
  - 학습 데이터를 준비
  - 가중치 초기화

AND		
X1	X2	T
0	0	0
0	1	0
1	0	0
1	1	1

$$w_0 = 0.3$$

$$w_1 = 0.4$$

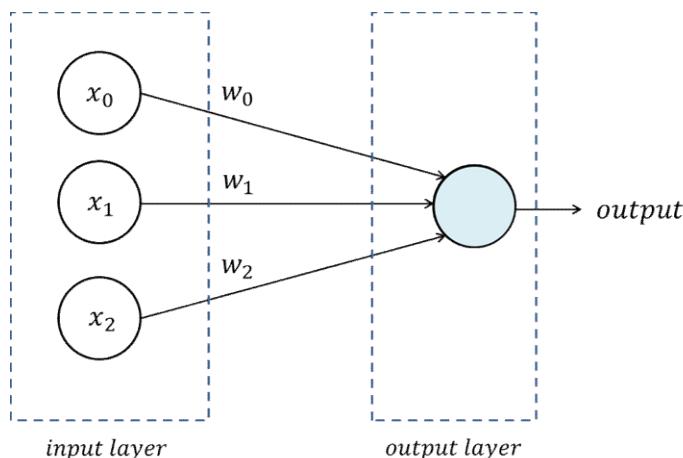
$$w_2 = 0.1$$

$$x_0 = -1$$

## 연산 1-1

- 학습데이터

- $F([x_1, x_2]) = f([0, 0]) = 0$
- $F([x_1, x_2]) = f([0, 1]) = 0$ 
  - 출력과 일치하여
  - 가중치 조정 없음



$$w_0 = 0.3, w_1 = 0.4, w_2 = 0.1$$

$$x_0 = -1, x_1 = 0, x_2 = 0$$

$$net = w_1x_1 + w_2x_2 + w_0x_0 = -0.3$$

$$f(net) = 0$$

$$w_0 = 0.3, w_1 = 0.4, w_2 = 0.1$$

$$x_0 = -1, x_1 = 0, x_2 = 1$$

$$net = w_1x_1 + w_2x_2 + w_0x_0 = -0.2$$

$$f(net) = 0$$

## 연산1-2

$$w_i = w_i + \eta x_i (t - f(\text{net}))$$

$\eta$  = learning rate  
 $t$  = target value

- $F([x_1, x_2]) = f([1, 0]) = 0$

$$\begin{aligned}w_0 &= 0.3, w_1 = 0.4, w_2 = 0.1 \\x_0 &= -1, x_1 = 1, x_2 = 0 \\net &= w_1 x_1 + w_2 x_2 + w_0 x_0 = 0.1 \\f(net) &= 1\end{aligned}$$

- 출력값이 다르니 가중치 조정

$$\begin{aligned}w_0 &= w_0 + \eta x_0 (T - f(\text{net})) = 0.3 + 0.05 \times (-1) \times (0 - 1) = 0.35 \\w_1 &= w_1 + \eta x_1 (T - f(\text{net})) = 0.4 + 0.05 \times 1 \times (0 - 1) = 0.35 \\w_2 &= w_2 + \eta x_2 (T - f(\text{net})) = 0.1 + 0.05 \times 0 \times (0 - 1) = 0.1\end{aligned}$$

- $F([x_1, x_2]) = f([1, 1]) = 1$

- 출력값 같아서 가중조정 없음

$$\begin{aligned}w_0 &= 0.35, w_1 = 0.35, w_2 = 0.1 \\x_0 &= -1, x_1 = 1, x_2 = 1 \\net &= w_1 x_1 + w_2 x_2 + w_0 x_0 = 0.1 \\f(net) &= 1\end{aligned}$$

## 연산2

- $F([x_1, x_2]) = f([0, 0]) = 0$ 
  - 보정 없음
- $F([x_1, x_2]) = f([0, 1]) = 0$ 
  - 보정 없음
- $F([x_1, x_2]) = f([1, 0]) = 0$ 
  - 보정 필요
    - $w_0 = 0.35 + 0.05 \times (-1) \times (0 - 1) = 0.4$
    - $w_1 = 0.35 + 0.05 \times 1 \times (0 - 1) = 0.3$
    - $w_2 = 0.1 + 0.05 \times 0 \times (0 - 1) = 0.1$
- 이후 목표값과 같아지면 반복 종료하고 놓바든 가중치들 멀노의 공간에 서상

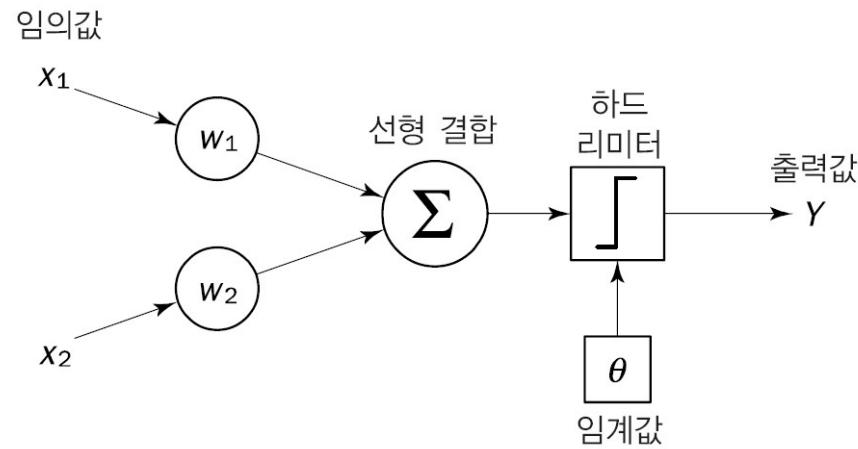
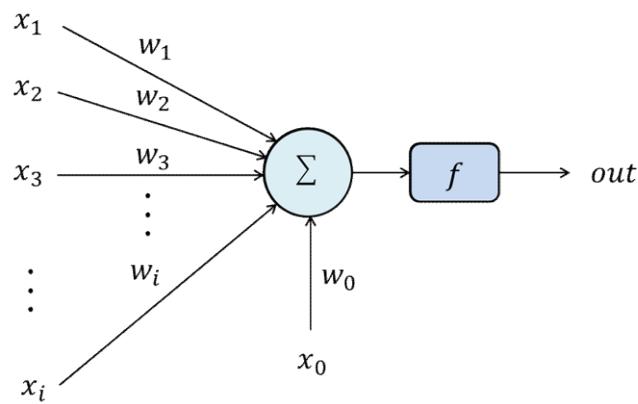


딥러닝

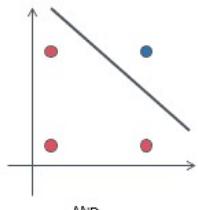
DeepLearining

## 단층 퍼셉트론(single-layer perceptron)

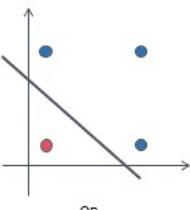
- 뉴런(neuron):
  - 인공신경망을 구성하는 가장 작은 요소 net값이 임계치보다 크면 활성화되면서 1을 출력하고, 반대의 경우에는 비활성화되면서 0을 출력
- 단층 퍼셉트론은
  - 입력층(input layer)
  - 출력층(output layer)으로 구성



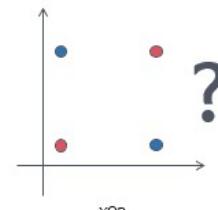
# 다층 퍼셉트론



AND



OR



XOR

AND		
Input_A	Input_B	Output
0	0	0
0	1	0
1	0	0
1	1	1

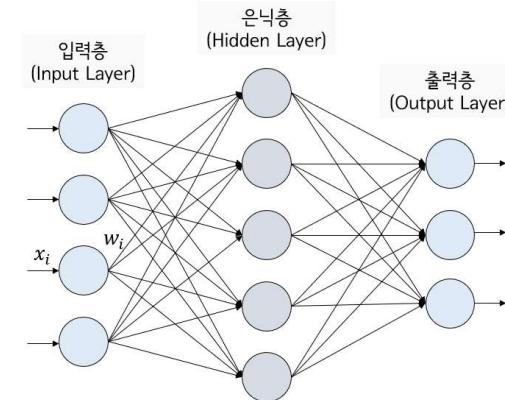
OR		
Input_A	Input_B	Output
0	0	0
0	1	1
1	0	1
1	1	1

XOR		
Input_A	Input_B	Output
0	0	0
0	1	1
1	0	1
1	1	0

- 단층 퍼셉트론은 XOR 문제를 해결하지 못함.
  - 이를 해결할 수 있는 대표적 모델기법이 다층 퍼셉트론
- 여러 신경망 모델 중 가장 일반적이며, 대표적인 모델.
- 입력층과 출력층 사이에 **1개 이상의 은닉층(hidden layer)**이 존재
- 목적:** 범용 머신러닝의 비선형 구분 능력

# Neural Network

- 사고를 생성하는 인간의 두뇌 메커니즘 규명
  - 생각하는 기계를 만들 수 있지 않을까 하는 아이디어에서 출발한 이론
  - 퍼셉트론 모델을 연결하여 문제 해결
- 가장 기본적인 단위는 뉴런이라는 세포.
  - 각각의 뉴런은 신경 시스템에서 여러 가지 기능적인 역할을 담당
  - connectionist models
  - parallel distributed processing
- 신경망 이론(차이)
  - 기준: 절차적인 순서에 의한 알고리즘을 통해 기호를 처리하여 문제를 해결
  - 인간의 두뇌 신경조직을 모델로 하여 단순한 기능을 하는 처리기(신경세포)들을 대규모로 상호 연결한 다음 연결 강도를 조절하여 문제를 해결



## 역전파 Error back propagation Algorithm

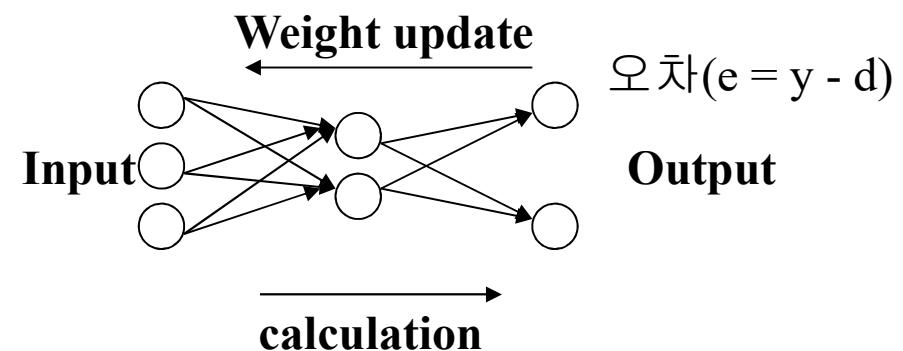
- 개념

- 은닉층의 학습을 위해 출력층에서 발생한 오류를 이용하여 은닉층 오차계산
  - 다시 이 값을 입력층으로 역전파시켜, 출력층의 오차가 원하는 수준이 될 때까지 반복
  - **다층 퍼셉트론 구조의 대표적인 학습방법.**
  - 최종 출력벡터와 목표 출력벡터(부류벡터)의 오차 제곱합을 줄이는 방향으로 각 퍼셉트론의 가중치를 조절.
  - **오차제곱합을 비용함수로 하고 이를 줄이는 방향으로 가중치를 조절**
  - 경사하강법을 이용.
- 신경망의 처리 : 입력층 → 은닉층 → 출력층
  - Weight 갱신의 학습방향: 출력층 → 은닉층

$$\text{Error}(e) = y - d$$

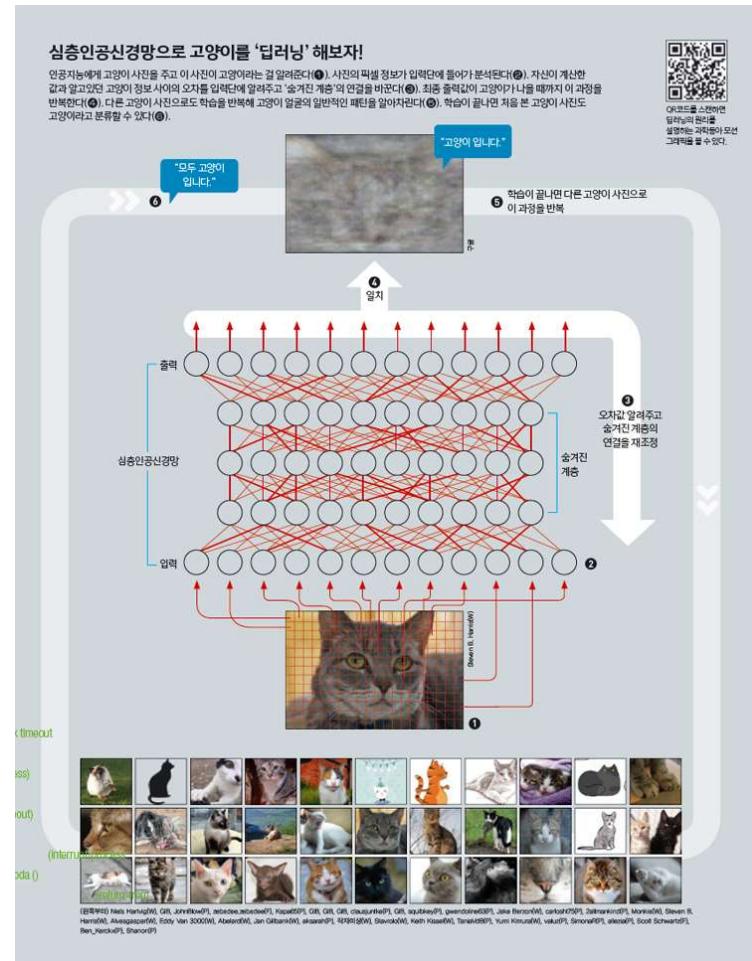
y: 데이터 입력에 대한 결과 값

d: 원 하는 값

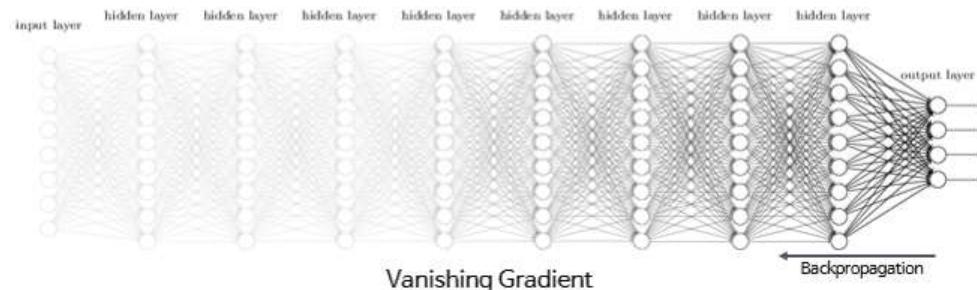
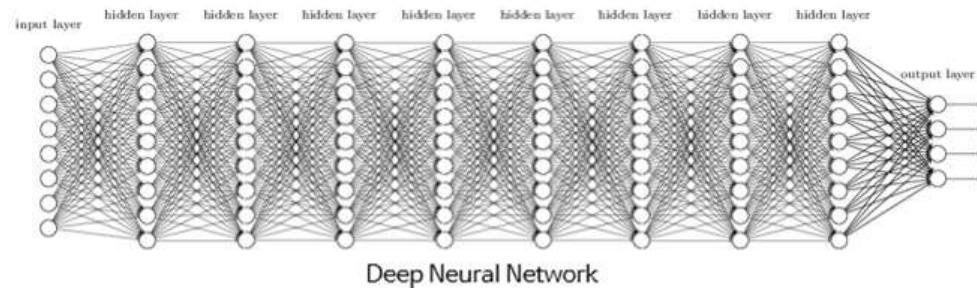


# 심층 인공신경망

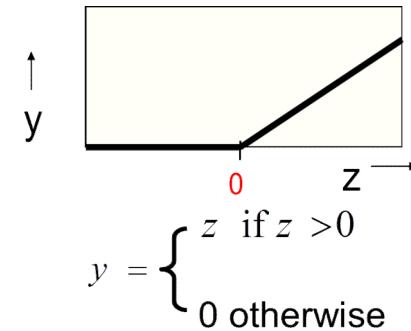
- 딥러닝
  - 모두 고양이인 점 인식
  - 픽셀 분석
  - 은닉단 오차 수정
  - 개별 픽셀 고양이 얼굴의 일반패턴 이해
  - 인간의 이해와 다름



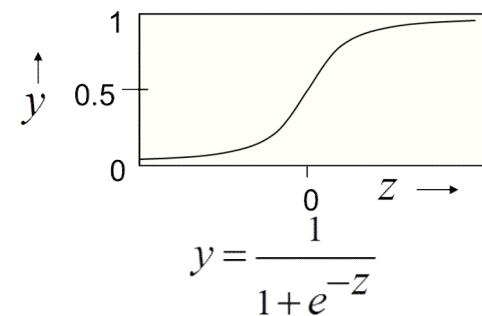
# 역전파 소실현상 Vanishing Gradient



Rectified Linear neurons  
(linear threshold neurons)

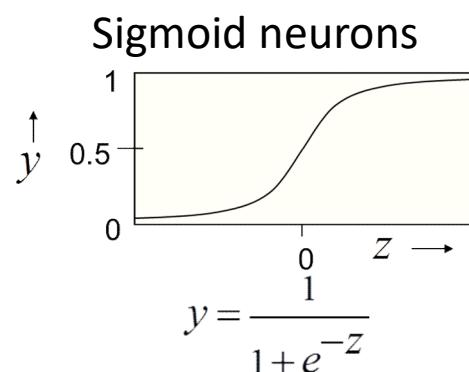
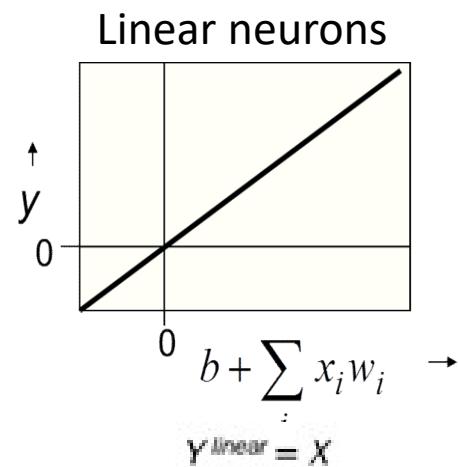


Sigmoid neurons

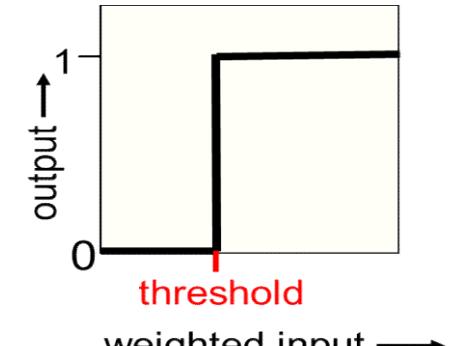


# 활성함수(Activation function)

선형 활성화 함수  
(linear activation function)



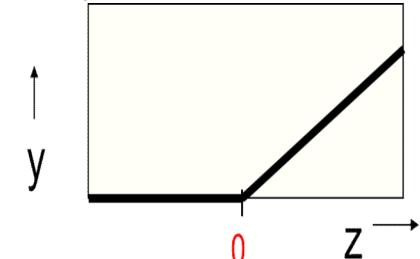
Binary threshold neurons



$$Y^{\text{step}} = \begin{cases} 1, & \text{if } X \geq 0 \\ 0, & \text{if } X < 0 \end{cases}$$

Rectified Linear neurons

(linear threshold neurons)



$$y = \begin{cases} z & \text{if } z > 0 \\ 0 & \text{otherwise} \end{cases}$$

## 크로스 엔트로피

- Information
  - 정보량
  - 브라질 vs 중국  $-\log P(x) = -\log(0.99) = 0.01$  vs  $-\log(0.01) = 4.6$
- Entropy
  - 의미
    - 정보량의 평균
    - 확률변수의 평균 정보량의 크기
    - 놀람의 평균적인 정도
    - 불확실성의 정도
    - 엔트로피는 정보량에 대한 기댓값이며 동시에 사건을 표현하기 위해 요구되는 평균 자원
    - 예측이 어려울수록 정보량이 많아지고 엔트로피는 커짐
  - $0.99 * -np.log(0.99) + 0.01 * -np.log(0.01) = 0.056$ 
    - 정보 의미가 극히 작다 – 늘 일어날 일이 일어난다. 정보의 가치가 적다
  - $0.5 * -np.log(0.5) + 0.5 * -np.log(0.5) = 0.693$ 
    - 정보 의미가 크다 – 어떤 일이 일어날지 모른다 정보의 가치가 크다.
  - $\text{entropy} = E(-\log P(x))$

## Entropy

- KL-divergence (KL 확연한 차이)
  - relative entropy, 즉 상대적인 entropy 를 말한다.  
※ Kullback-Leibler divergence (KLD) 의 줄임말이다.
  - 계산
    - $Q(x)$  : 예측  $P(x)$ :실제 확률
    - $KL\text{-divergence} = \text{relative entropy} = E(-\log(Q(x))) - E(-\log(P(x)))$
    - 예측정보량과 실제 정보량의 거리(차이) 0이면 예측과 실제가 같음
    - 극단적 정보의 차이를 배제한 정보 가치에 관한 척도로 Error 나 Cost 측정
- $KLD = E(-\log(Q(x))) - E(-\log(P(x)))$ 
  - 사전 예측시  $P(x)$ 를 알 수 없어 사전 예측시 Error 찾기 어려움

## cross entropy

- KLD한계
  - 실제 벌어진 일이  $P(x)$ 가 없으면 Error 를 최소화 하는 값을 모름
  - 모르는 상황에서 Error 최소화 방법은  $E(-\log(Q(x)))$ 를 최소화
  - Minimize  $E(-\log(Q(x)))$ 를 cross entropy 라 부름
- cross entropy
  - 확률분포인  $Q(x)$ 대신 실제확률분포인  $P(x)$  를 사용한 크로스 엔트로피
  - cross entropy = entropy  $P(x) = Q(x)$  일 때 최소화
  - $E(-\log(Q(x))) = -\sum P(x) * \log(Q(x))$
  - 그런데  $E(-\log(Q(x)))$  를 구할 때 각  $Q(x)$ 에 대한 가중치로  $P(x)$ 가 필요한데, 이때  $P(x)$ 는 우리가 가진 데이터의 비율을 사용. (empirical distribution)
  - 예
    - $E(-\log(Q(x))) = -\sum P(x) * \log(Q(x)) =$
    - $- (P(\text{브라질이 이길 확률}) * \log(Q(\text{브라질이 이길 확률})) + P(\text{아르헨티나가 이길 확률}) * \log(Q(\text{아르헨티나가 이길 확률})) )$
  - 만약 브라질이 아르헨티나를 이겼다는 데이터를 가지고 있는 경우
  - Cross entropy =  $- (1 * \log(Q(\text{브라질이 이길 확률})) + 0 * \log(Q(\text{아르헨티나가 이길 확률})) )$
  - $= - \log(Q(\text{브라질이 이길 확률}))$
  - 즉, log likelihood에 -1 을 곱해준 값 (negative log likelihood) 과 동일해짐
-

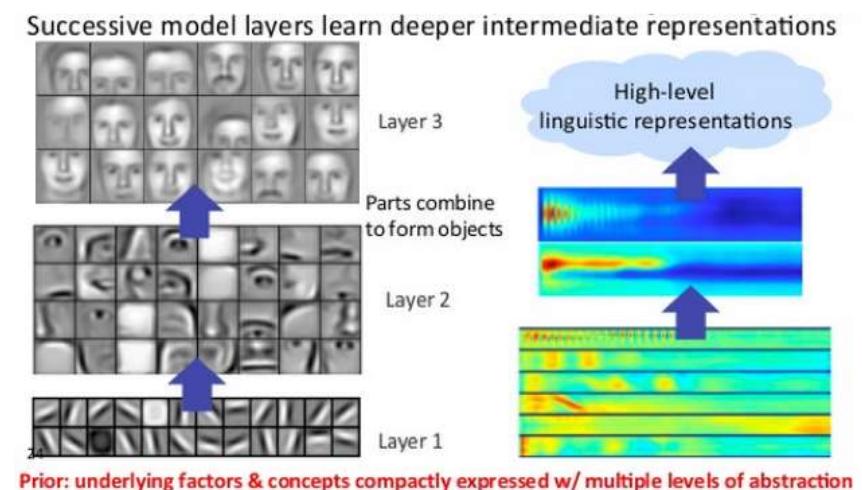
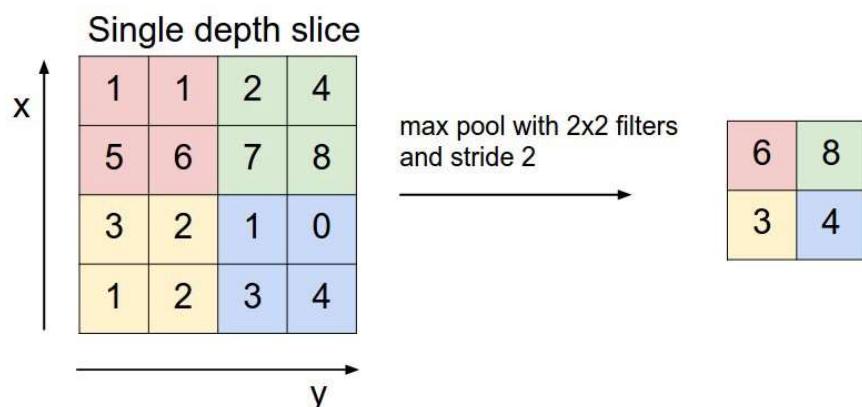


CNN

Convolution Neural Network

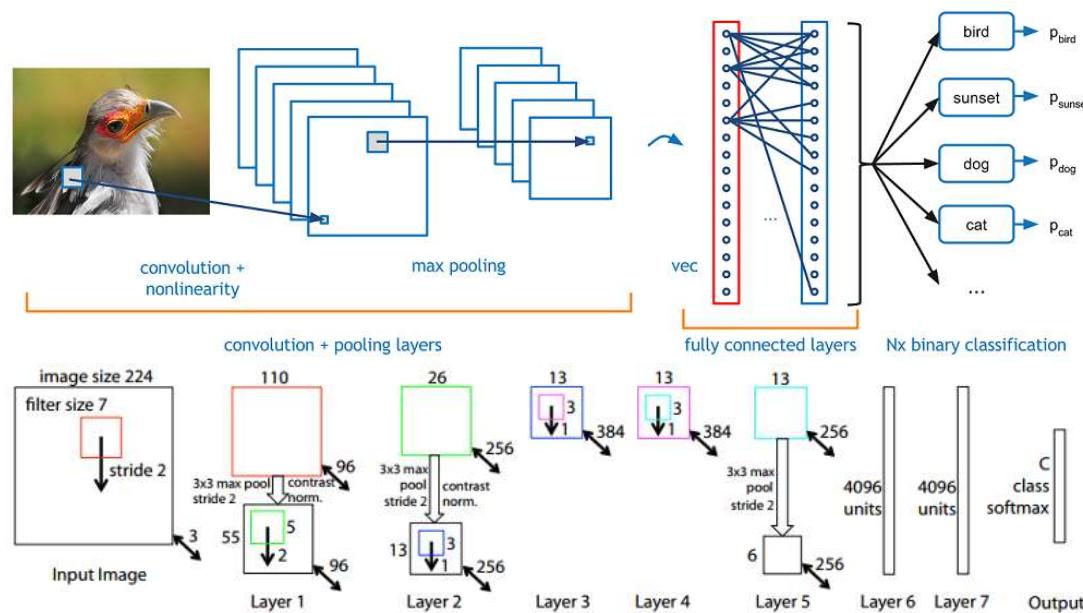
# CNN

- 목적
  - 주로 그래픽 처리 분류 목적
- 원리
  - 행렬 축소



# CNN

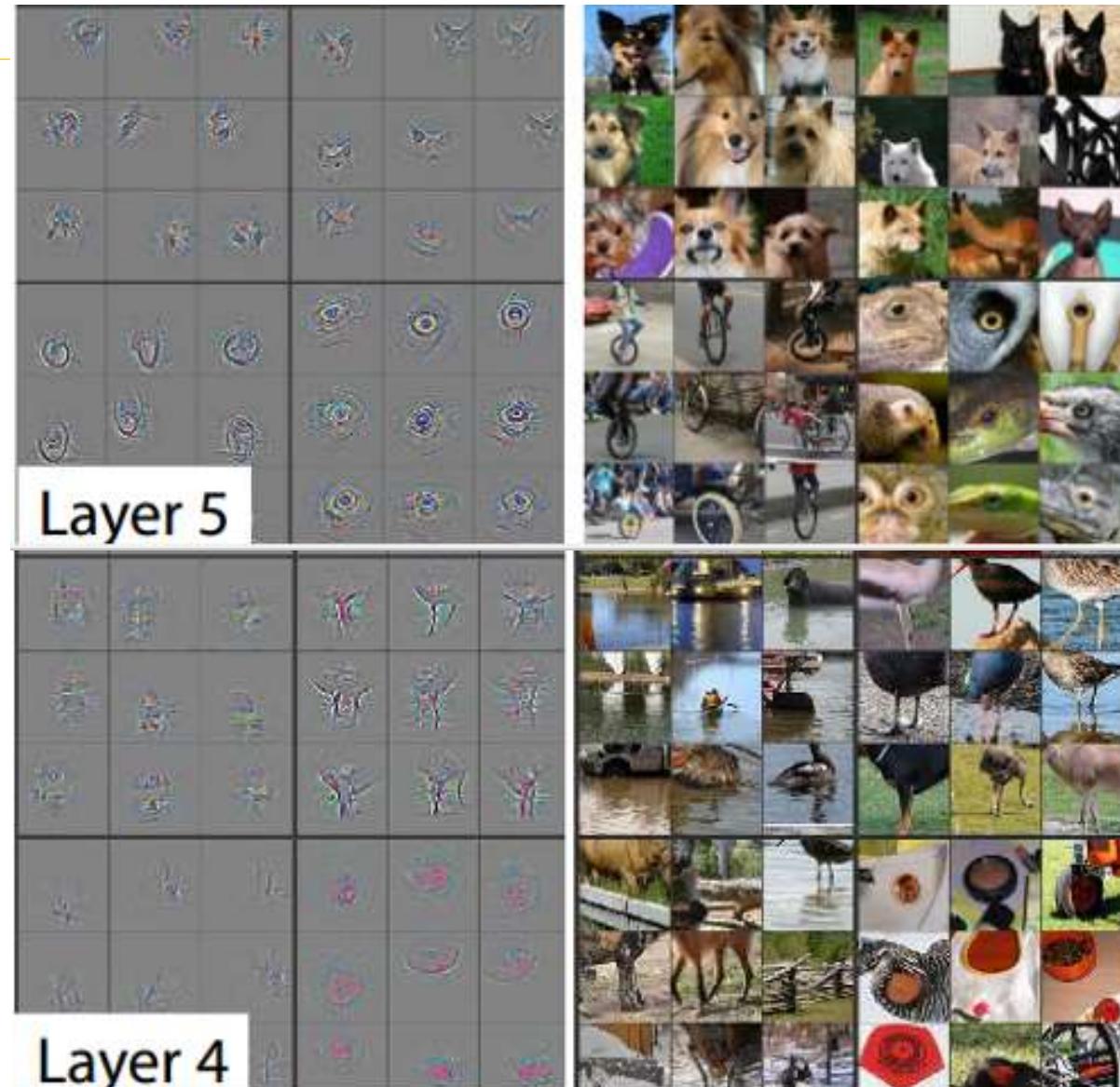
- VGG16모델
  - 창을 이용하여 특징 추출모델 pooling layer
  - 이미지 244\*244 필터 7\*7
  - 필터링 통해 축소



스탠퍼드 CNN 연구  
<http://cs231n.github.io/>

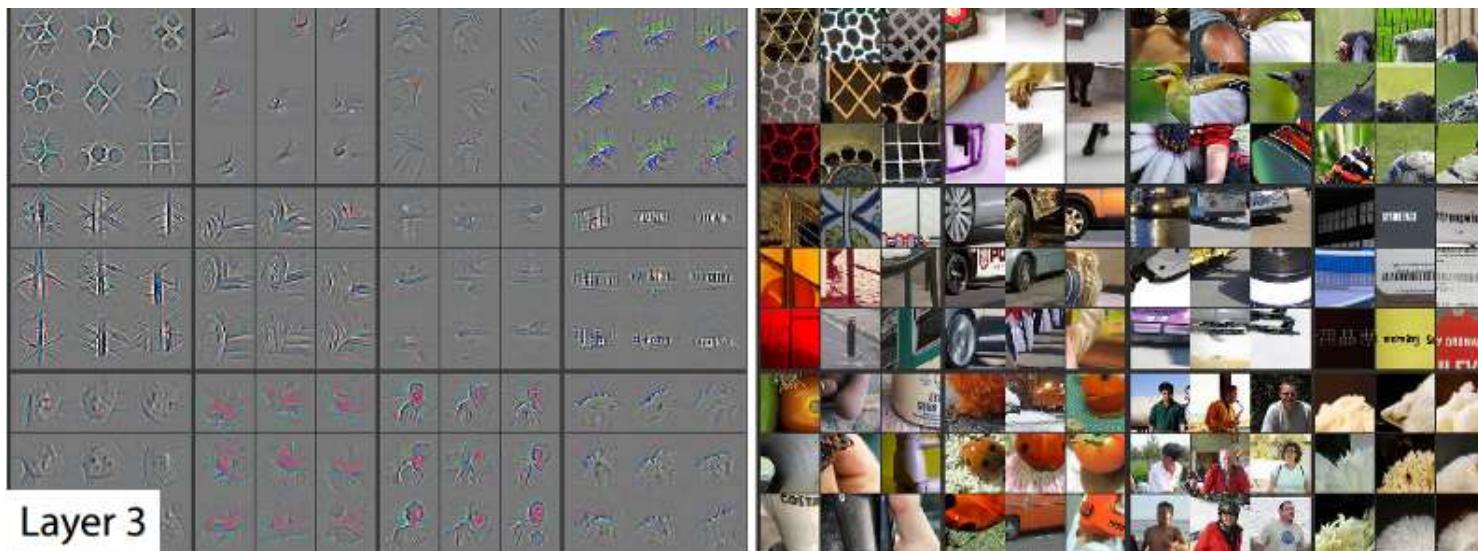
## CNN

- 다리인식
- 눈 원형인식



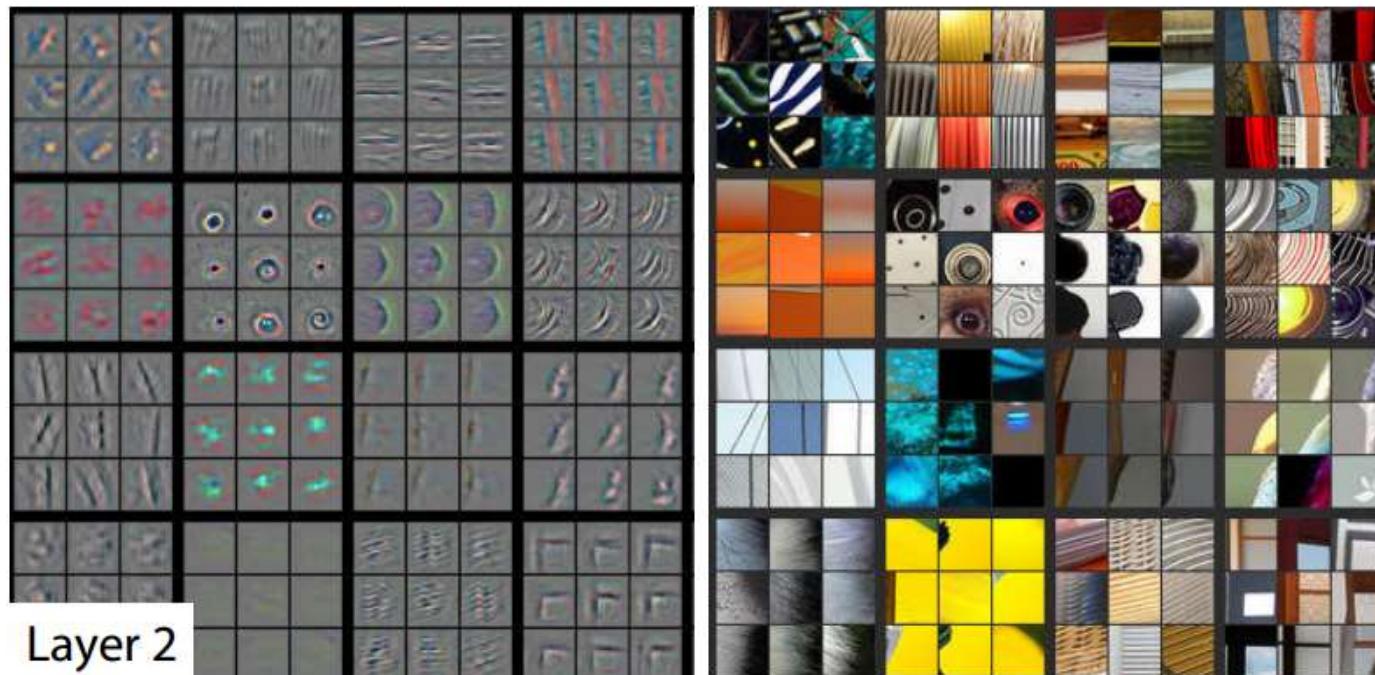
# CNN

- 패턴 타이어 등의 인식



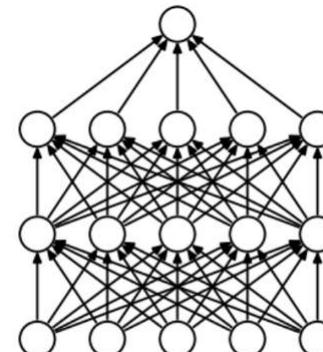
# CNN

- 원형 선 등의 특징 추출

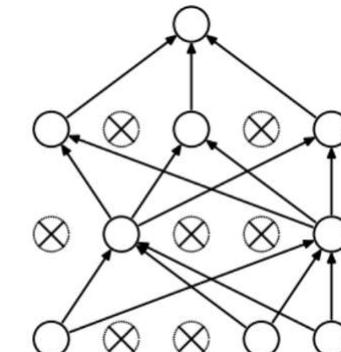


# CNN

- 언더피팅
  - 계산량 증가
- 오버피팅
  - 데이터추가
  - 데이터 증가사용
  - 정규화 추가 드롭아웃
  - 일반화



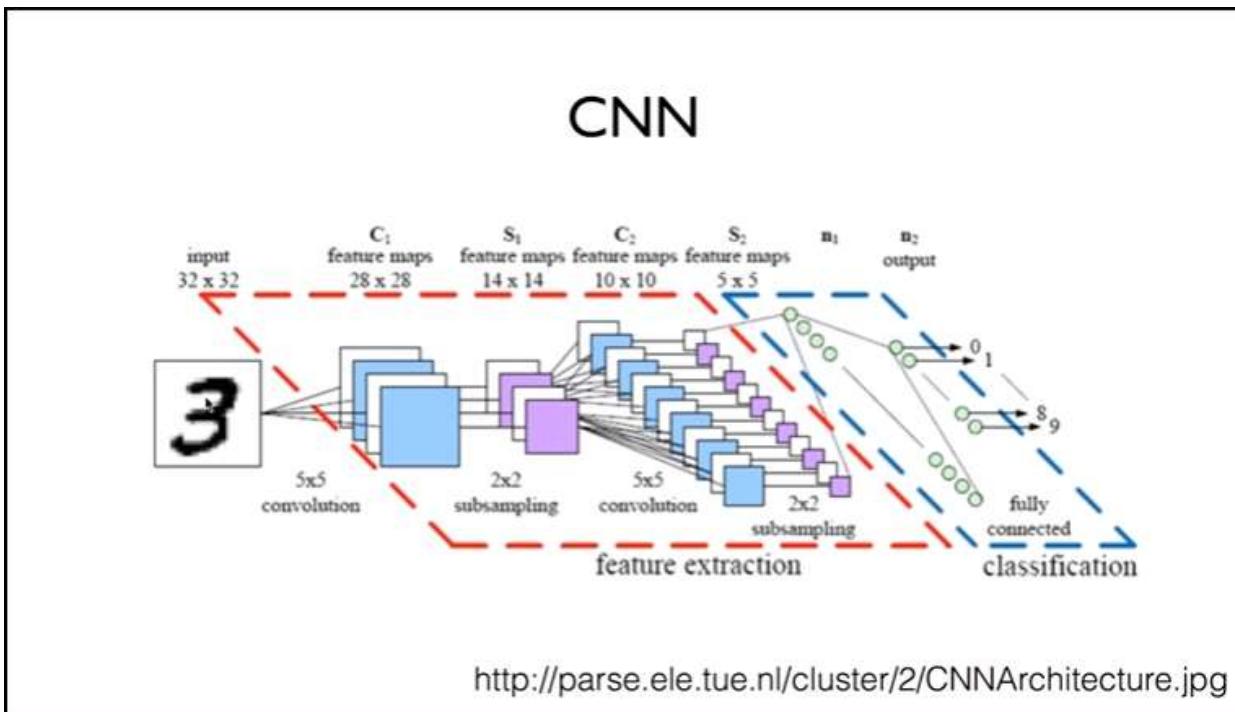
(a) Standard Neural Net



(b) After applying dropout.



# 기본 개념



- conv 2개, pooling 2개, fully connnected
- <https://github.com/nlintz/TensorFlow-Tutorials>

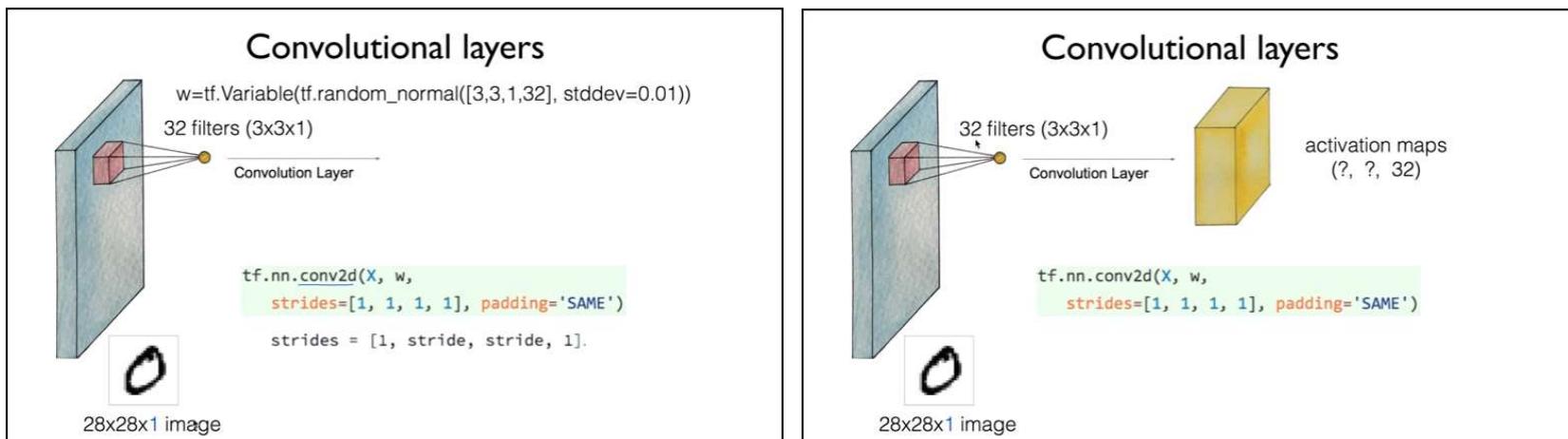
# Convolutional layers

- 수식

- `tf.nn.conv2d(input, filter, strides, padding, use_cudnn_on_gpu=None, data_format=None, name=None)`
- input : [batch, in\_height, in\_width, in\_channels] 형식. 28x28x1 형식의 손글씨 이미지.
- filter : [filter\_height, filter\_width, in\_channels, out\_channels] 형식. 3, 3, 1, 32의 w.
- strides : 크기 4인 1차원 리스트. [0], [3]은 반드시 1. 일반적으로 [1], [2]는 같은 값 사용.
- padding : 'SAME' 또는 'VALID'. 패딩을 추가하는 공식의 차이. SAME은 출력 크기를 입력과 같게 유지.

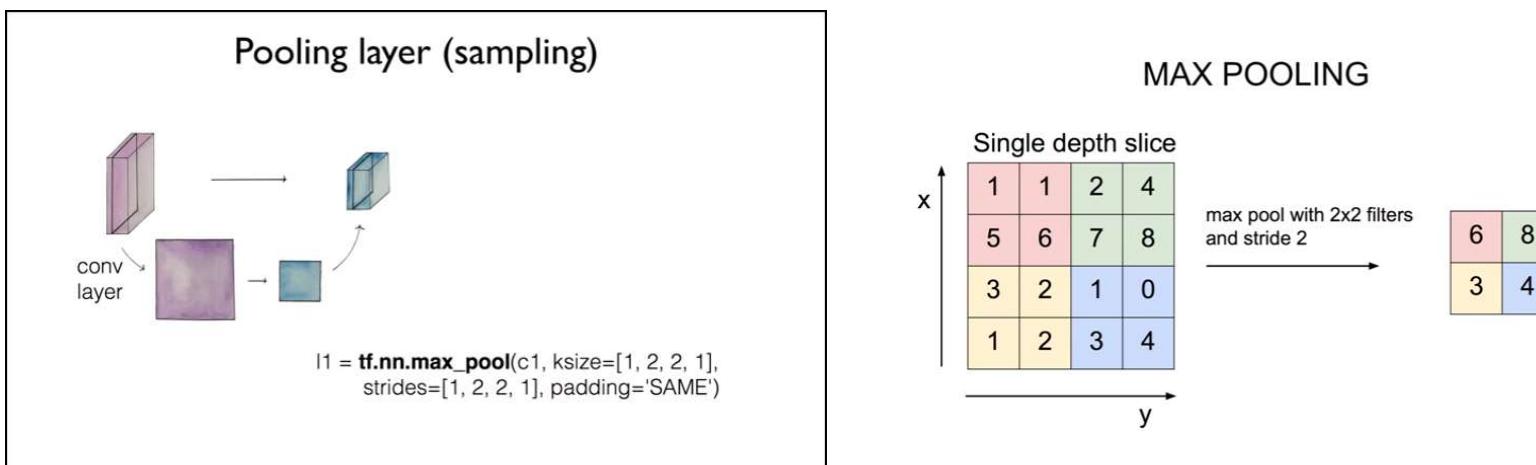
- 사용

- 3x3x1 필터를 32개 만드는 것을 코드로 표현
- [3, 3, 1, 32]가 된다. 순서대로 너비(3), 높이(3), 입력 채널(1), 출력 채널(32)을 뜻한다. 32개의 출력이 만들어진다.
- 만들어진 레이어를 RELU로 던지기만 하면 끝
- `Layer=tf.nn.conv2d(X, w, [1,1,1,1],"SAME")`
- `LaReLU=tf.nn.relu(Layer)`



# Pooling layer

- Pooling layer
  - 여러 개 중에서 하나를 선택(sampling)
  - `tf.nn.max_pool(value, ksize, strides, padding, data_format='NHWC', name=None)`
  - value : [batch, height, width, channels] 형식의 입력 데이터. ReLU를 통과한 출력 결과
  - ksize : 4개 이상의 크기를 갖는 리스트로 입력 데이터의 각 차원의 윈도우 크기
  - data\_format : NHWC 또는 NCHW. n-count, height, width, channel의 약자 사용.
- 내용
  - ksize가 [1,2,2,1]이라는 뜻은 2칸씩 이동하면서 출력 결과를 1개 생성
  - 4개의 데이터 중에서 가장 큰 1개를 반환하는 역할





RNN

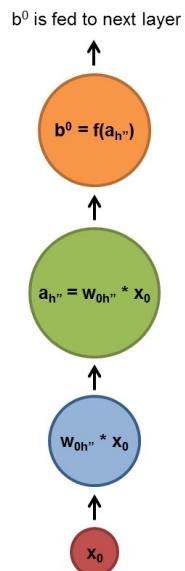
Recurrent Neural Network

## RNN

- RNN(Recurrent Neural Network)
  - 주로 시계열 데이터 분류 목적
  - 유닛 사이의 연결이 Directed Cycle 을 형성하며 자신을 가리키는 Recurrent Weight 을 포함
  - 글, 유전자, 손글씨, 음성 신호, 센서가 감지한 데이터, 주가 등 배열(sequence, 또는 시계열 데이터)의 형태를 갖는 데이터에서 패턴을 인식하는 인공 신경망
  - 매우 강력한 알고리즘-뇌와 유사- 패턴 기억 인식
  - 기억 공간의 한계가 있어 주로 과거 기억 삭제
    - 유사 교육 많이 받을수록 정확도 증가

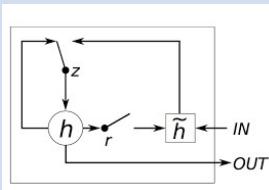
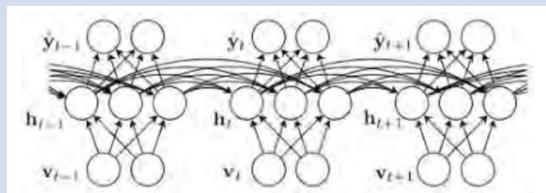
## FFNets vs RNN

- FFNets Feed-forward neural networks
  - 입력층->은닉층->출력 현재만 보고 가중치 판단
- RNN Recurrent neural networks
  - 입력층->은닉층->자기저장은닉층->출력



# RNN 구성요소

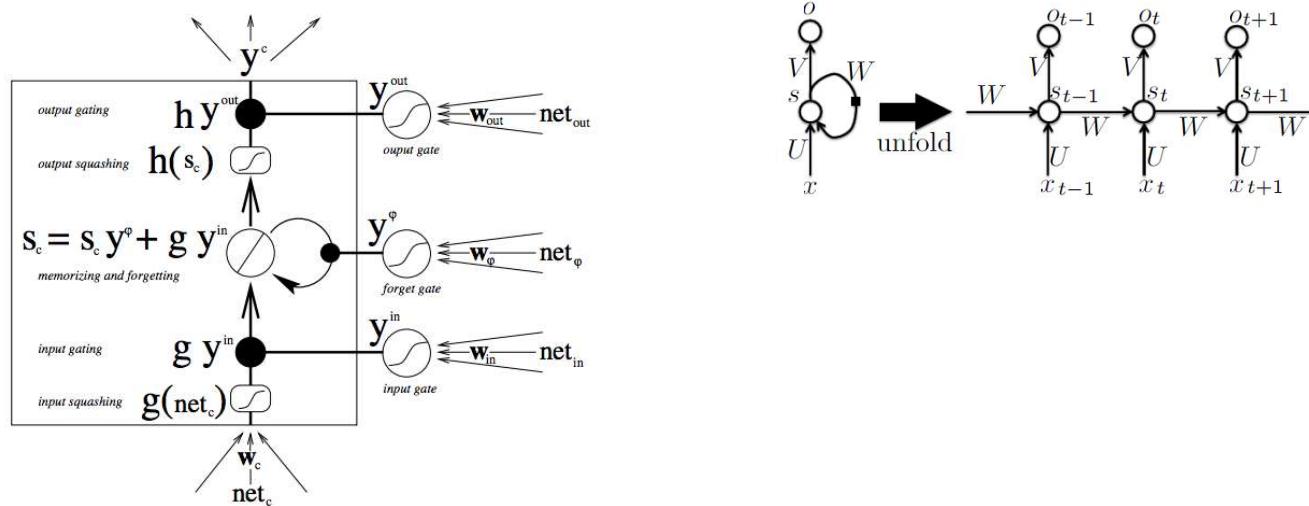
구분	기술	내용
레이어 구성	<b>Directed Cycle</b>	-방향성 있는 cycle 이용하여 하나의 입력 값에 여러 개의 값이 출력
	<b>Recurrent Weight</b>	-자기 자신을 가리 키는 반복 가중치 구조
	<b>BPTT 알고리즘</b>	-Back Propagation Through Time -오류역전파(Error Back Propagation)를 확장한 알고리즘으로 시간 방향 학습
성능개선	<b>LSTM</b> <b>Long Short TermMemory</b>	-input/output/forget 3가지 게이트를 이용하여 데이터의 입출력을 조절
	<b>GRU</b> <b>Gated Recurrent Unit</b>	-reset, update unit 을 이용하여 데이터의 입출력을 조절



BPTT 의 vanishing gradient 문제를 LSTM, GRU 이용하여 해결

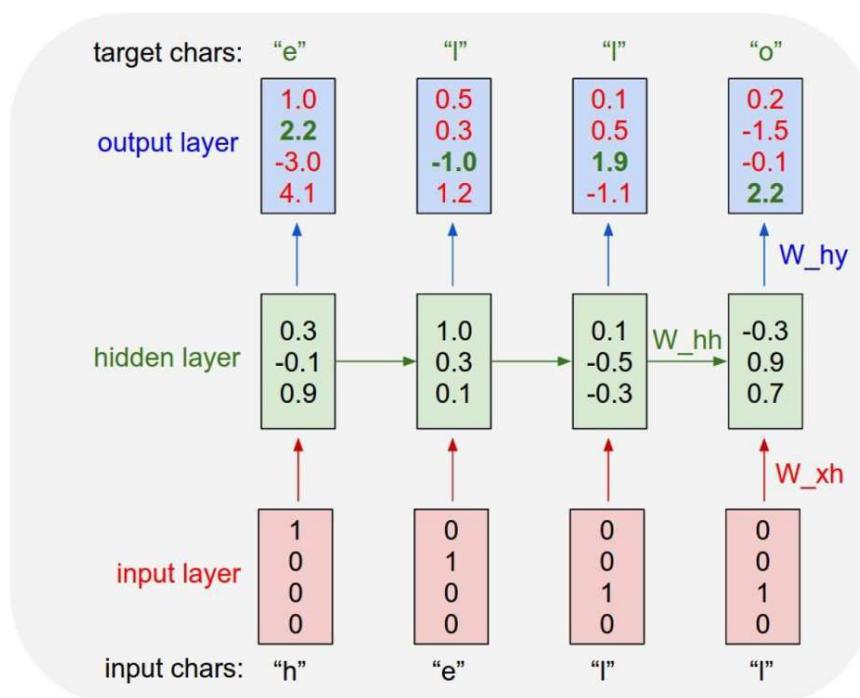
# Long Short-Term Memory Units

- Long Short-Term Memory Units (LSTM)
- RNN 의 한 종류
  - +로 과거 자료를 더함

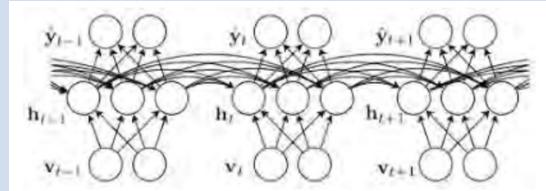
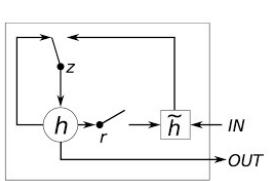


## FFNets vs RNN

- FFNets Feed-forward neural networks
  - 입력층->은닉층->출력 현재만 보고 가중치 판단
- RNN Recurrent neural networks
  - 입력층->은닉층->자기저장은닉층->출력



# RNN 구성요소

구분	기술	내용
레이어 구성	<b>Directed Cycle</b>	-방향성 있는 cycle 이용하여 하나의 입력 값에 여러 개의 값이 출력
	<b>Recurrent Weight</b>	-자기 자신을 가리 키는 반복 가중치 구조
	<b>BPTT 알고리즘</b>	-Back Propagation Through Time -오류역전파(Error Back Propagation)를 확장한 알고리즘으로 시간 방향 학습 
성능개선	<b>LSTM</b> <b>Long Short TermMemory</b>	-input/output/forget 3가지 게이트를 이용하여 데이터의 입출력을 조절
	<b>GRU</b> <b>Gated Recurrent Unit</b>	-reset, update unit 을 이용하여 데이터의 입출력을 조절 

BPTT 의 vanishing gradient 문제를 LSTM, GRU 이용하여 해결

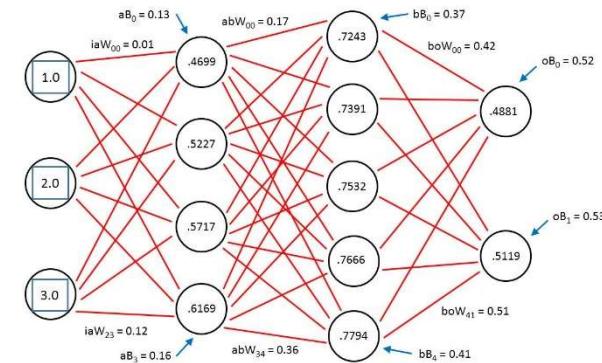
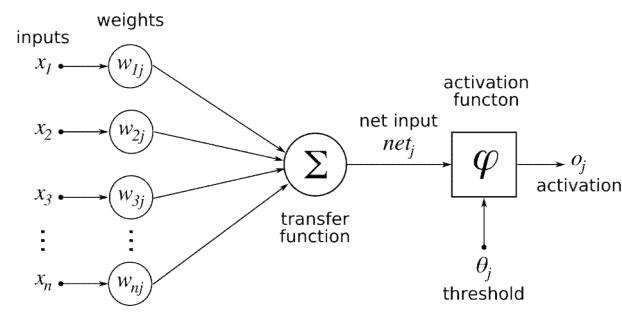


종합

인공지능

# 딥러닝

- 시대의 변화
  - 알고리즘이 귀찮아
  - 과거 규칙기반 모델->비정형 모델 자율학습
  - 인간을 베끼자!



## 문제해결

- 공부 못하는아이
  - 공부해도 모른다.(Under Fit)
  - 공부만 하고 있다. (Slow Fit)
  - 똑똑한데 시험만 보면 틀린다.(Over Fit)
- 해결책
  - 공부 시간을 늘린다.
  - 공부 요령을 가르친다.
  - 적당히 쉬게한다.

그래도 안되면

호랑이 선생님?

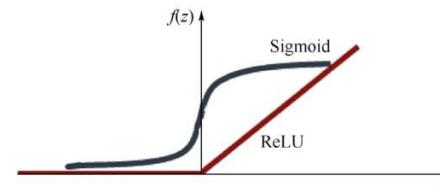


## 공부해도 모르는 아이(Under Fit)

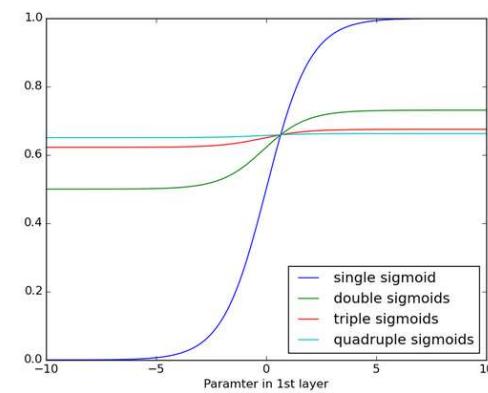
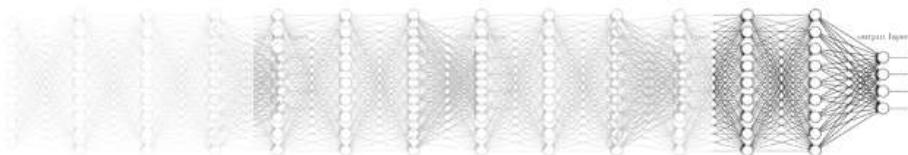


- 학습이 끝나면 시험결과 보고 복습을 하는데 뒤로 틀린 가중치 전달해서 곱함
- 시그모이드함수 곱할수록 작아져
- ReLU (Rectified Linear Unit) 사용

Sigmoid!



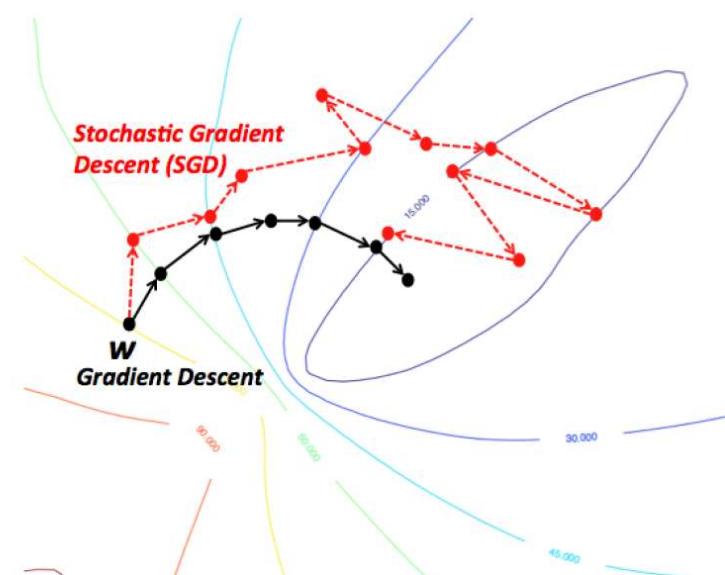
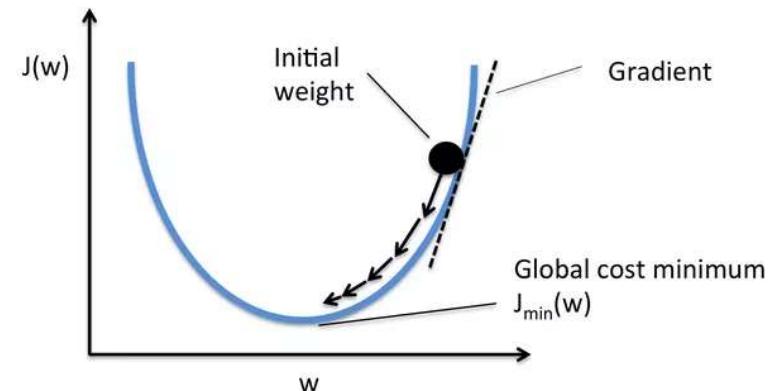
Vanishing gradient (NN winter2: 1986-2006)



# 공부만 하고 있는 아이(SLOW FIT)



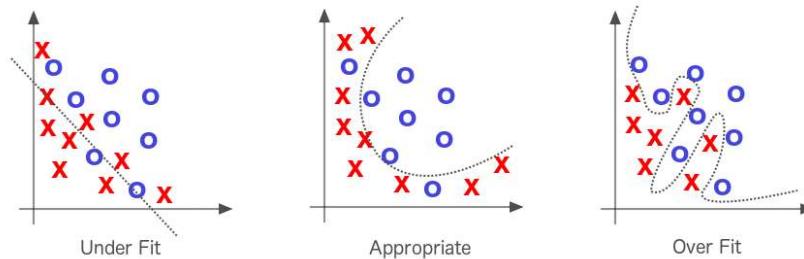
- Gradient Descent 방법
  - 훑어가며 공부(스텝증가)
  - LEARNING RATE
- SGD
  - stochastic gradient descent
  - 데이터의 gradient를 평균내어
  - gradient update를 하는 대신  
(이를 'full batch'라고 한다)
  - 일부의 데이터로 'mini batch'를 형성하여
  - 한 batch에 대한 gradient만을 계산
  - 배치 알고리즘 다양
- CNN
  - 창으로 부분부분 만들어 특징 추출
  - 이미지 분류 속도 증가



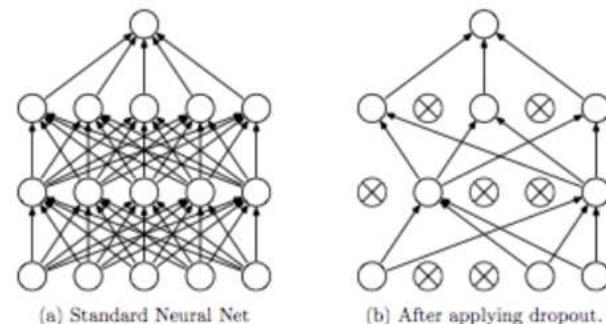
## 똑똑한데 시험 못보는아이(Over Fit)



- 시험만 보면 틀린다.
- 요령을 가르치자
  - Drop Out
- 정규화
- 학습지도
  - 다른애들 하는걸 보자
  - VGG16



Dropout: A Simple Way to Prevent Neural Networks from Overfitting [Srivastava et al. 2014]





## 인공지능의 역사

---

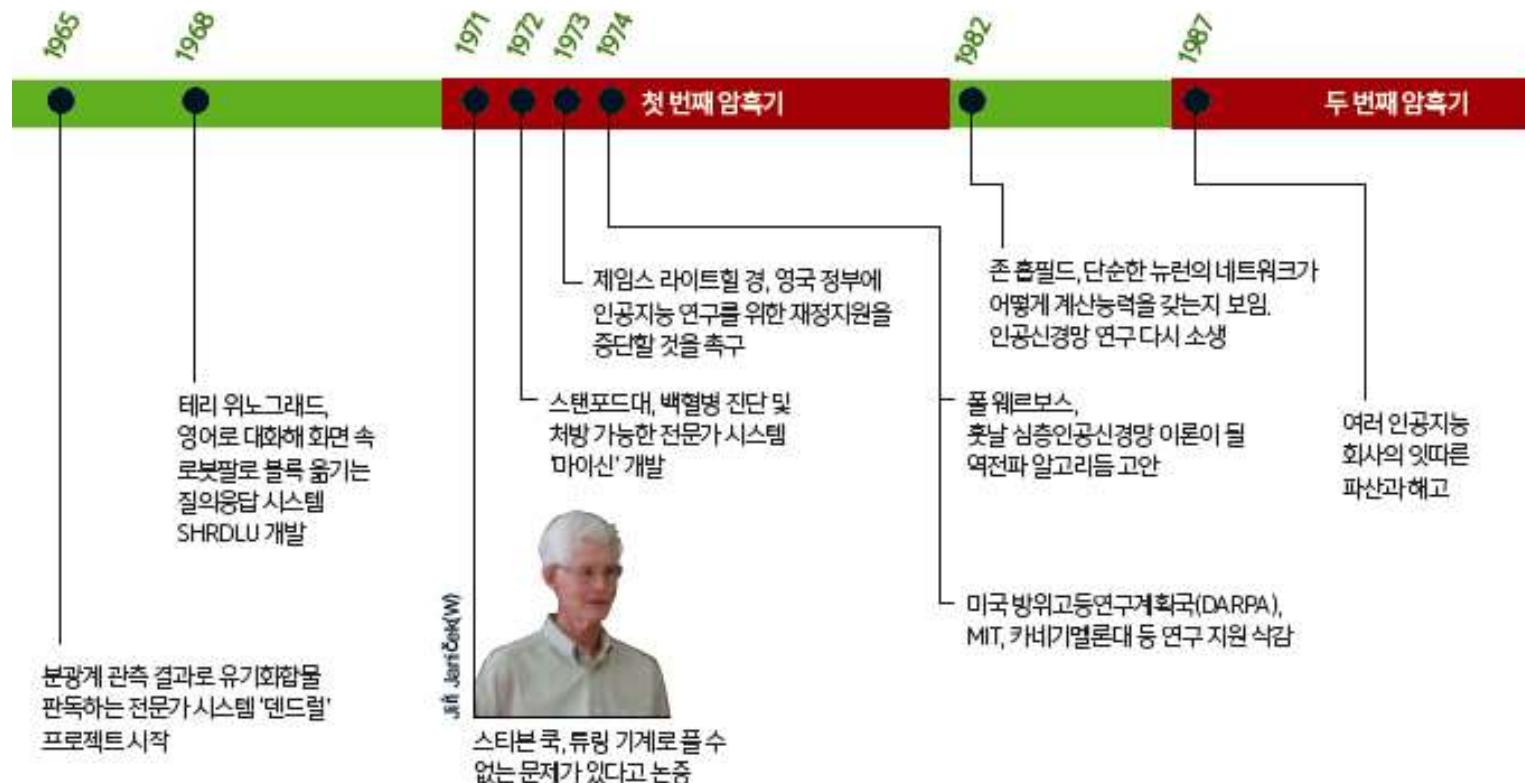
# 역사



# 역사1

- 태동기 60년대
  - 1956 인공지능 artificial intelligence 연구제안
  - 존 매카시(다트머스대), 마빈 민스키(하버드대), 너대니얼 로체스터(IBM), 클로드 쇄넌(벨 연구소) 드림등 10명
  - 1958 MIT와 카네기멜론대에 인공지능 연구소 설립,
  - 사이먼과 앤더슨 뉴웰이 1959년에 만든 범용 문제 해결 알고리즘은 '하노이의 탑 퍼즐' 해결
  - 1959 머신러닝 등장 아서 사무엘
- 1차암흑기 70년대
  - 1969 마빈스키의 동료 세이무어 페퍼트는 퍼셉트론이론의수학적 한계 증명
  - 1971 영국 맨체스터 대 보고서:인공지능의 데이터 증가에 따른 계산 시간이 지수적 증가
  - 1971년 스티븐 쿡, 1972년 리처드 카프, 1973년 레오니드 레빈이 컴퓨터로 얼마나 빨리 풀 수 있는지를 다루는 'P-NP' 문제 해결 난제
- 재건기 80년대
  - 전문가 시스템 도입: 정부주도적 전략투자
    - 전문가의 지식과 규칙 데이터베이스
    - 자문과 의사결정 추론엔진:인공지능
    - 사용자와 시스템간의 I/O
  - 인공지능추론
    - 베이지안 정리
    - 퍼지이론 0/1 이아닌 다중값 논리 수학의 원리
  - 화학기업 듀퐁이나 컴퓨터기업 DEC등에서 도입하여 원가절감
  - 비즈니스 인텔리전스로 발달

# 역사



# 역사2

# 재건기

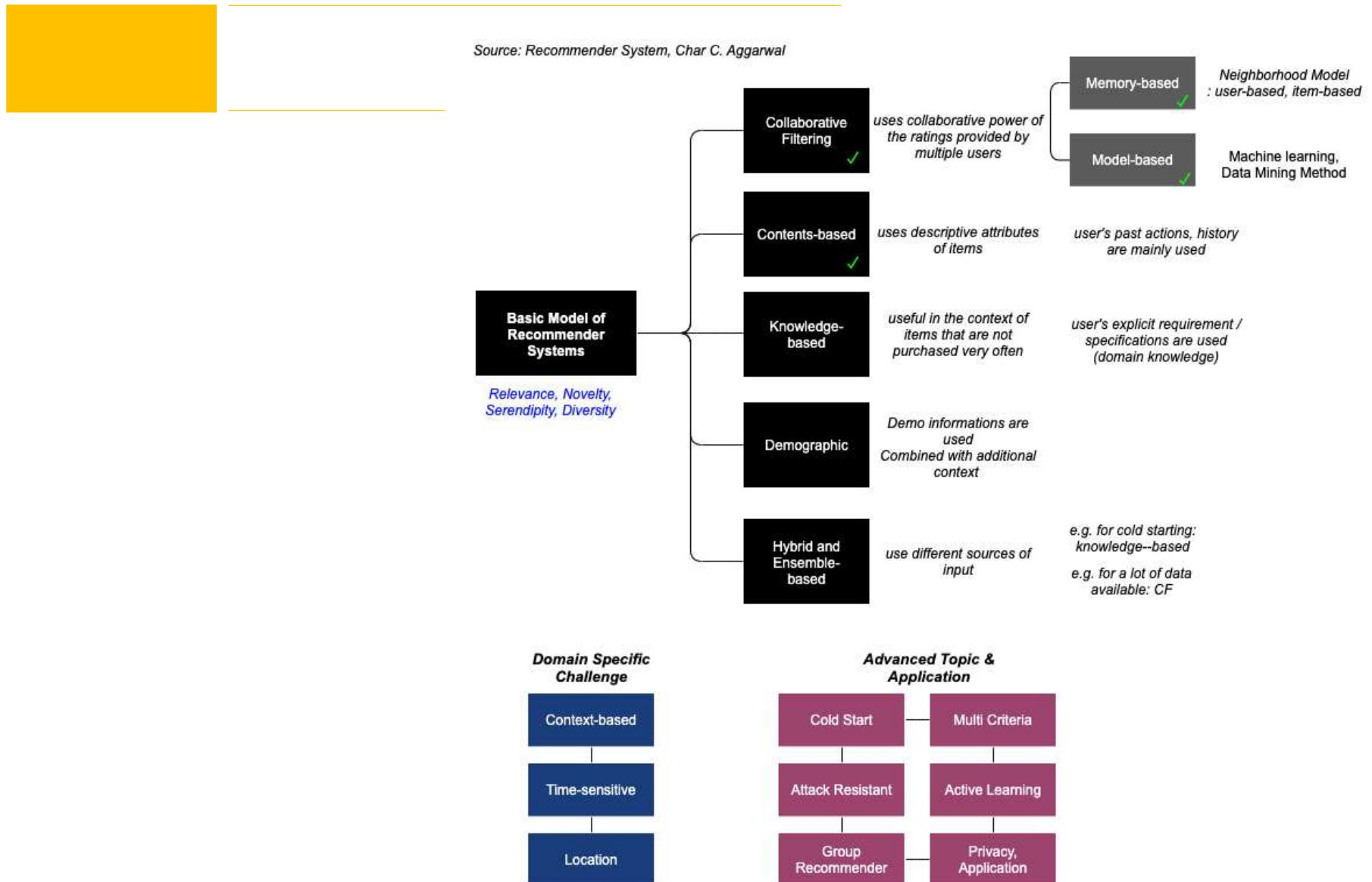




추천시스템

# 추천시스템 종류

- 정의
  - 정보 필터링 (IF) 기술의 일종으로, 특정 사용자가 관심을 가질 만한 정보 (영화, 음악, 책, 뉴스, 이미지, 웹 페이지 등)를 추천
- 종류
  - Collaborative Filtering 협업 필터링
    - 사용자가 주로 들어온 밴드와 각각의 음원이 무엇인지 관찰
    - 다른 사용자와의 행동(behavior)을 비교하면서 추천된 음악의 "상태"를 생성 비교추천
      - 잠재요인 협업 필터링
      - 음악의 경우 장르 작곡가 템포 등을 잠재 요인으로 분석
    - [장점] 컨텐츠 정보 몰라도 행동기반 추천 가능
    - [단점] 대량 데이터 필요
    - 거리기반 추천 : KNN, Corr
  - Content-based filtering (personality-based approach) 컨텐츠 기반 추천시스템
    - 음악자체의 특성을 활용
    - 유사 사용자의 특성과 비교 추천
    - [단점] 컨텐츠에 대한 특성 필요
    - [장점] 소량 데이터로 추천 가능
    - 필터 기반 추천 : TF-IDF, Info filter
  - Knowledge-based systems : 지식 기반 추천 시스템.
  - 추천을 위한 행렬 분해 (Matrix decomposition for recommendations)
- Hybrid recommender systems
  - Cold start: 충분한 정보가 없어서 필요한 정보를 얻지 못하는 것 해결
  - Sparsity(희소의 문제) 해결



# 추천시스템 제작

- 제작 요건

- 도메인지식
  - 목적분류
    - 유사상품 추천
    - 판매 증진
    - UX 만족도 증가
  - 경쟁사 포지션
  - 시장규모등
- 데이터
  - 다양한 추천에 따른 선택 시스템
- 피드백 시스템
  - 추천 선택이 잘되고 있나 단기분석
  - 장기적 추적 분석

## 추천 목적

목적	설명	예시	측정
Best 최고 추천	잘 팔리는 상품 추천	베스트 셀러	인기 추종 순위 비교
Related 연관추천	관심상품과 유사도 측정	해당상품 구매고객 정보	아이템 유사도
Personalized 개인화 추천	개인화 적합 추천	님만을 위한 음악	사용자별 아이템 유사도
Context Aware 문맥추천	상황에 적합한 상품추천	날씨 계절 기반 음악	상황별 아이템 유사도

### • 극복 과제

- 콜드 스타트 극복 : 자료가 없으면 추천 어려움
- 추천의 다양성 확보 : 뻣한 자료의 반복은 고객의 불만을 가중(공영방송)
- 리스트간 유사성 분석 : 동일 패턴의 반복(공영 조사기관)
- 끊임없는 피드백을 통한 변화



기업별 AI 전략

---

## AI 기업별 기술 IBM

- 왓슨

- 카네기 멜론대 딥블루(체스 인간 승 1997.5)후속
- 자연어 처리 전문 인공지능
- 2011 제퍼디 출연 74연승 완승
- 분야
  - 캠브리지대 항암유전자에 영향을 미치는 단백질 6개 발견
  - 헬스케어 인공지능 암센터 운영
  - 법률미 로펌 판례검색 시스템
  - 감성 분석
  - 이미지인식 기상예측등
  - 애플의 시리와 합작



- 진화

- 클라우드기반 머신 인텔리전스 선두주자



## AI 기업별 기술 Google

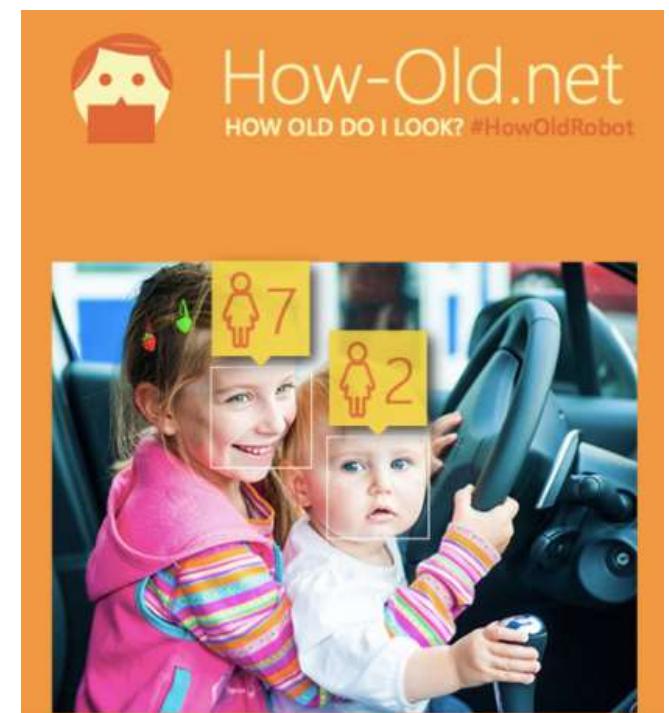
- 딥마인드 인수(영국,2014)
  - AI 후발주자
  - 대용량 데이터 처리방법 융합
  - 빅데이터 누적으로 양질의 데이터 확보
- 텐서플로
  - AI 툴 오픈소스화



## AI 기업별 기술MS

- 옥스포드(Project Oxford)

- 특징
  - 뛰어나지는 않지만 개발 효율성을 높여줄 많은 API 제공
- 분야
  - 머신 비전(machine vision)
  - 음성 인식
  - 언어 분석을 포괄
  - 고수준의 큐레이트 API 제공
- 발전
  - Ajure 클라우드 AI 개발 플랫폼 확보 목적
  - DMLT(Distributed Machine Learning Toolkit)
  - 자체 머신러닝 프로젝트를 오픈소스로 공개



## AI 기업별 기술 Intel

- 업체인수로 AI 가속화
  - 인지컴퓨팅 업체 샤프론 인수 2015
  - 비전컴퓨팅 업체 모비디우스 인수 2016
  - AI 트레이닝 업체 너바나 인수 2016
  - AI 후발주자
    - 이종 데이터 연결 인지 지능화 연구
  - AI 개발의 중심을 속도로 전이
    - 고대역폭 병렬처리 기술 확보
- 방향
  - 하드웨어와 유기적 결합: AI 전용칩 생산



## AI 기업별 기술 기타

- 애플
  - 시리:음성인식및 자연어 처리에 노력
  - 래티스 인수:비정형 데이터에 특화 범죄 예방 지원
  - 혁신 압박
- 페이스북
  - 이미지 처리 태깅 처리에 노력 안면인식
  - 자연어 처리 챗봇 오픈
- 야후
  - 데이터 개방 연구 알고리즘 확보
  - 하드웨어와 유기적 결합:AI전용칩 생산
- 텐센트/알리바바/아마존
  - 인재확보
  - 판매목적 AI 확충

# Ai 비서

표 1 국내외 주요 기업들의 AI 음성비서 개요

업체	플랫폼	출시	지원 언어	특징, 적용분야, 확산전략 등
Apple	Siri	'11.10.	英·佛·獨·스페인어·日·中·韓 등 17개	<ul style="list-style-type: none"> <li>iOS, Mac OS 등 자사 운영체제에서 이용. 문맥 파악, 대화 가능</li> <li>iPhone·iPad 등에 탑재. 향후 스마트홈 솔루션에 탑재 예상</li> <li>iPhone 출시 10주년 맞아 업그레이드 예정</li> <li>개발자들에 문호 확대 시작</li> </ul>
amazon.com	Alexa	'14.11.	英·獨	<ul style="list-style-type: none"> <li>자연어처리, 빠른 반응 속도, 다중계정 지원(수동), 클라우드 기반</li> <li>스피커(Echo 등) 외 다양한 스마트홈 디바이스에 탑재</li> <li>서드파티에 개발자 키트 공개</li> </ul>
Google	Google Assistant	'16.10.	英·獨	<ul style="list-style-type: none"> <li>Google Now('12) 업그레이드 버전, 자사 검색엔진과 연동, 다중 계정 지원 개발 중</li> <li>모바일메신저(Allo), 스마트폰(Pixel, G6 등), 스마트워치(LG Watch Sport 등), 스피커(Google Home), 자동차 등으로 탑재 확대</li> <li>언어 확대, 스마트폰 점유율 기반 보급 확산. 스마트워치 등 여러 기기나 서비스에 통합될 수 있도록 API 공개</li> </ul>
Microsoft	Cortana	'14.4.	英·佛·獨·스페인어·日·中 등 15개 (韓 '17.4월 추가 예정)	<ul style="list-style-type: none"> <li>Window 10, Xbox one 등 MS의 운영체제와 기기에서 이용 가능</li> <li>API를 공개하여 Window 10 기기 외에 스마트TV 등 각종 스마트 기기 등에서도 이용할 수 있도록 준비 중</li> </ul>
Baidu 百度	度祕 (Duer)	'15.9.	中	<ul style="list-style-type: none"> <li>'2015 바이두 세계대회('15.9.8.)에서 공개</li> <li>CES 2017에서 공개된 음성명령 지원 가정용 로봇 사오위(小魚·Little Fish)에 탑재</li> <li>교육, 헬스케어, 가사 등으로 서비스 영역 확대</li> </ul>
SK telecom	NUGU	'16.9.	韓	<ul style="list-style-type: none"> <li>'16년 9월 국내 최초 AI 스피커 NUGU에 탑재</li> </ul>
KT	GiGA Genie	'17.1.	韓	<ul style="list-style-type: none"> <li>'17년 1월 AI 스피커 GiGA Genie에 탑재</li> </ul>
SAMSUNG	Bixby	'17.1H. (예정)	英·韓 등 7~8개	<ul style="list-style-type: none"> <li>'16년 11월에 공개되었으며 갤럭시S8('17.4. 출시 예정) 탑재 전망</li> <li>개방형 플랫폼 구축 후 서드파티에 개방하고, 자사 가전제품과 연동하여 스마트홈 기능 제공 계획</li> </ul>
NAVER	Clova	'17 (예정)	-	<ul style="list-style-type: none"> <li>오감(五感) AI 개발에 초점. 충분한 콘텐츠를 확보하고 있는 한·일 양국을 토대로 점차 아시아, 세계로 확대</li> <li>AI 비서로 개발하던 기존 'Amica'의 업그레이드 버전으로 네이버·라인의 AI 기술이 총집결된 AI 플랫폼</li> </ul>

자료 : 각종 자료

## IBM 웃슨 서비스(API)

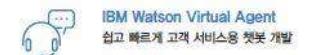
IBM 웃슨 API는 총 15개로 크게 6가지 영역으로 나눌 수 있다.

### 대화



#### IBM Watson Conversation

모바일 기기, 메시징 플랫폼, 로봇 등에  
손쉽게 헷볼과 가상 애이전트를 구축하여  
자연어 대화를 지원



#### IBM Watson Virtual Agent

쉽고 빠르게 고객 서비스용 헷볼 개발

### 이미지



#### IBM Watson Visual Recognition

이미지를 식별 및 분류하고, 사용자가  
직접 추가 훈련을 더하여 다양한 산업과  
애플리케이션에 활용 가능

### 감정



#### IBM Watson Personality Insights

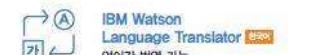
텍스트 분석을 통해 글쓴이의  
5가지 성격 특성, 가치관 등을 추론



#### IBM Watson Tone Analyzer

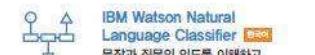
글의 전반적 느낌과 문제를 감지하여  
글쓴이의 행복함, 슬픔, 자신감 등의  
감정을 파악

### 언어



#### IBM Watson Language Translator

언어간 번역 가능



#### IBM Watson Natural Language Classifier

문장과 질문의 의도를 이해하고  
정보를 의도에 맞게 분류



#### IBM Watson Retrieve & Rank

질문에 가장 관련성이 높은  
정보를 찾아 답을 제공

### 디스커버리



#### IBM Watson Discovery

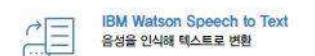
데이터 분석을 통해  
적합한 정보를 빠르게 검색



#### IBM Watson Discovery News

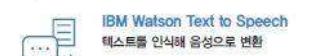
뉴스 및 블로그 컨텐츠에서 동향 분석

### 음성



#### IBM Watson Speech to Text

음성을 인식해 텍스트로 변환



#### IBM Watson Text to Speech

텍스트를 인식해 음성으로 변환



#### IBM Watson Knowledge Studio

언어의 가장 적은 단위인  
형태소로 구분하여 분석 및 분류



#### IBM Watson Document Conversion

다양한 형식의 문서를 웃슨 서비스가  
활용할 수 있는 형태로 변환

# 100대 ai



## 100 STARTUPS USING ARTIFICIAL INTELLIGENCE TO TRANSFORM INDUSTRIES

### CONVERSATIONAL AI/ BOTS



### VISION



### AUTO



### ROBOTICS



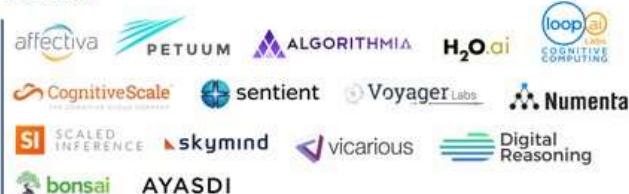
### CYBERSECURITY



### BUSINESS INTELLIGENCE & ANALYTICS



### CORE AI



### AD, SALES, CRM



### HEALTHCARE



### TEXT ANALYSIS/ GENERATION



### IOT/IOT



### COMMERCE



### FINTECH & INSURANCE



### OTHER





인공지능의 미래

---

# 인공지능의 미래

- 미래학자 레이커즈 웨일
  - 2045년 인공지능 특이점 도래
- 긍정적
  - 머신러닝 활용회사 증가
  - 매우 정확해졌다.
  - 못 찾는 것을 찾는다.
- 부정적
  - 금융 온라인 회사외 사용 없음
  - 빅데이터 대부분 없음
  - 클라우드 플랫폼 판매용
  - 통계보다 비싸다
  - 대안이 많다
  - 창의가 필요한 시대

인공지능 분야별 시장규모 (2016~2025 누적)

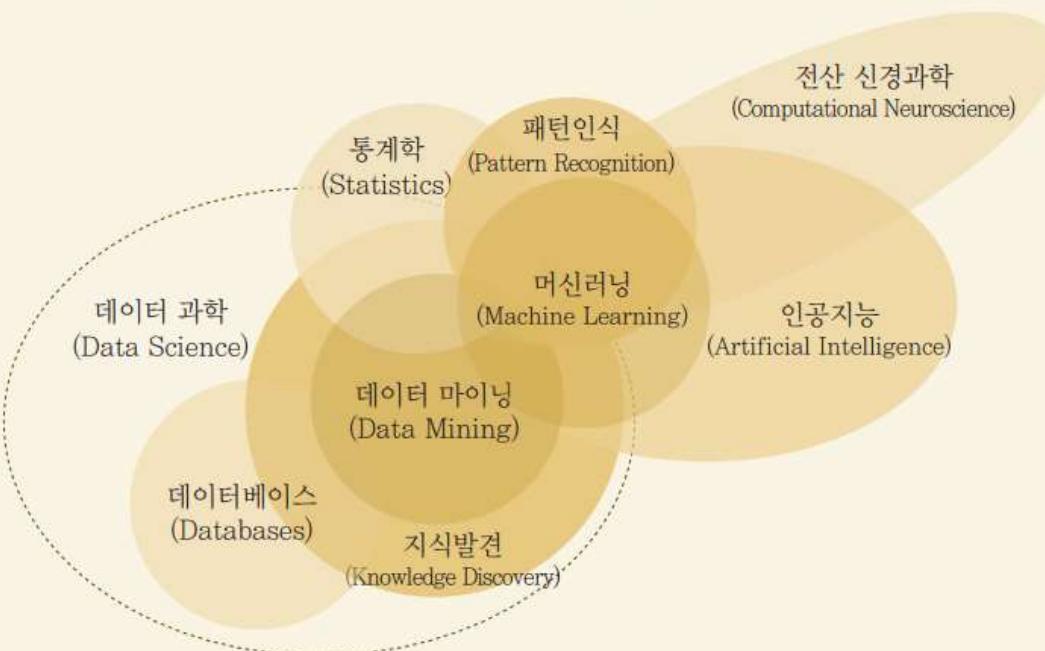


출처: Tractica(2016)

# 학문과 연계

[그림 1]

머신러닝의 여러 학문 분야와의 연계성



자료 : SAS Institute Inc., An Overview of Machine Learning with SAS® Enterprise Miner™

# 인지 컴퓨팅

- 개념
  - 인간의 인지와 비슷하거나 인지시스템의 영감을 받은 고차원적인 추론과 이해에 초점을 둔 컴퓨팅
  - 순수한 데이터 또는 센서 스트림보다는 상징적이고 개념적인 정보 대상
  - 복잡한 상황에서 고수준의 결정을 내리는 것을 목표
  - 자동차 충돌 예측 vs 좋아 한다 정도 예측
- 형태
  - 연계 작동하는 여러 AI 서브시스템으로 구성된 하나의 온전한 아키텍처
- 반론
  - “‘인지적’이라는 말은 마케팅을 위한 헛소리” 기계는 생각할 수 없다.- 가트너 부사장 톰 오스틴-



## 10대 인공지능

Forbes

- 포브스(Forbes) 선정 10대 인공지능 기술
  - 자연어 생성
  - 음성인식
  - 가상 에이전트
  - 머신러닝(ML) 플랫폼
  - AI 최적화 하드웨어
  - 의사결정
  - 딥러닝(DL) 플랫폼
  - 생체인식
  - 로봇 자동화 프로세스
  - 텍스트 분석 및 자연어 처리

# 부정적 시각

- 물리적 한계
  - 두뇌를 모방할 컴퓨터 칩 없음
- 자본적 한계
  - 2013년 구글이 유튜브에 있는 고양이 얼굴을 구분하는 데 중앙처리장치(CPU) 1만6000개가 필요
  - 4세 지능 정도 수준은 워딩으로 아는 수준
- 기술적 한계
  - 전문가들은 반도체 칩 용량이 18개월에 2배씩 향상된다는 '무어의 법칙'이 수십 년 안에 끝날 것
  - 인텔은 수년째 제자리
  - 인위적 학습 없이는 아직 바퀴벌레 반응도 못 따라감
- 정치적 한계
  - 조금만 불안해도 다 없애버릴 것이다.
- 윤리적 한계
  - 초지능은 인간을 볼 이유가 없다.
- 현실적 한계
  - 2013년 한맥투자증권 자본 260억원
  - 직원 실수로 로봇 트레이더들이 2분만에 460억원 손실 입힘
  - 폐업



참고

---

# Ai 개발 플랫폼

- **Python**

- [Tensorflow](#) - 구글 딥러닝 플랫폼
- [Theano](#) - 딥러닝 알고리즘을 파이썬으로 쉽게
  - [Keras](#) - Theano 기반 모듈화
  - [Pylearn2](#) - Theano를 유지, 보수하고 있는 Montreal 대학의 Yoshua Bengio 그룹에서 개발 ML용
  - [Lasagne](#) - 가볍고 모듈화가 잘 되어 있어서 사용하기 편리함
  - [Blocks](#) - Theano 기반으로 신경망
- [Chainer](#) - 자유도가 매우 높음
- [nolearn](#) - scikit-learn과 연동 기계학습에 유용
- [Gensim](#) - 큰 스케일의 텍스트 데이터를 효율적으로 다루는 것을 목표
- [deepnet](#) - cudamat과 cuda-convnet 기반의 딥러닝
- [CXXNET](#) - MShadow 라이브러리 기반으로 멀티 GPU까지 지원
- [DeepPy](#) - NumPy 기반의 라이브러리
- [Neon](#) - Nervana에서 사용하는 딥러닝 프레임워크

## Ai 개발 플랫폼

- C++
  - [Caffe](#) - Berkeley
  - [DIGITS](#) - NVIDIA
  - [cuda-convnet](#) - Alex Krizhevsky와 Geoff Hinton이 ImageNet 2012 딥러닝 챌린지 우승
  - [eblearn](#) - NYU의 Yann LeCun 그룹에서 ImageNet 2013 딥러닝 챌린지를 우승
  - [SINGA](#) - Apache Software Foundation

# Ai 개발 플랫폼

- **JAVA**

- [ND4J](#) - N-Dimensional Arrays for Java. JVM
- [Deeplearning4j](#) - Java와 Scala로 작성된 비지니스용 오픈소스 분산처리 딥러닝 라이브러리
- [Encog](#) -머신러닝 프레임워크로 SVM, ANN, Genetic Programming, Genetic Algorithm, Bayesian Network, Hidden Markov Model 등을 지원

- **JavaScript**

- [ConvnetJS](#) - 스크립트형 딥러닝 모델의 학습.
- [RecurrentJS](#) - RNN/LSTM을 구현한

- **Julia**

- MIT에서 새로 개발한 언어
- [Mocha.jl](#) - C++ 프레임워크인 Caffe에 영감을 받아 만들어진 Julia 기반의 딥러닝 프레임워크
- [Strada.jl](#) - Caffe 프레임워크를 기반. CNN과 RNN을 CPU/GPU로 학습
- [KUnet.jl](#) - 최대한 적은 양의 코드로 작성하고자 하는 시도에서 만들어진 딥러닝 패키지

- **R**

- [darch](#) - 레이어가 많은 neural network(deep architecture)
- [deepnet](#) - BP, RBM, DBN, Deep autoencoder 등의 딥러닝 아키텍쳐, NN 알고리즘을 구현한 패키지.



주가 예측 프로젝트

Kospi prediction

## 주가 예측 모형 프로젝트

- 팩스넷의 데이터 웹크롤링을 통한 코스피 일별 거래정보 획득
- 주가정보 획득
- 주가정보 정제
- DB 저장
- 추후 데이터 융합

```
import sqlite3  
import requests  
from bs4 import BeautifulSoup  
dbname='paxstock.db'
```

# 주가 예측 모형 프로젝트

- DB 설계

```
#['20190710', '2058.78', '2061.55', '2068.20', '2056.48', 1, '6.75', '+0.33', '355677']

def maketable():

    with sqlite3.connect(dbname) as conn:

        cur=conn.cursor()

        sql="""create table kosp(      IDX integer primary key,      EPRICE REAL,      SPRICE
REAL,      HPRICE REAL,      LPRICE REAL,
UDCODE integer ,      UDOWN REAL,      UDRATE REAL,      TRQTY integer )"""

        cur.execute(sql)

        conn.commit()

        cur.close()
```

## 주가 예측 모형 프로젝트

- DB 정보 획득
- DB정보 표시

```
def checkidx(t,cur):  
    sql_title='select IDX from kospo where IDX='+t  
    resultid=cur.execute(sql_title).fetchall()  
    if(resultid):  
        print('already inserted:',resultid[0][0])  
        return False  
    else:  
        return True
```

# 주가 예측 모형 프로젝트

- DB 입력
- DB 생성 후 입력

```
def insertdb(data):  
    with sqlite3.connect(dbname) as conn:  
        cur=conn.cursor()  
        t=data[0]  
        if checkIdx(t,cur):  
            #종가 시가 고가 저가 전일비 등락률 거래량(천주)  
            sql="""insert into kospo  
            values(?,?,?,?,?,?,?,?,?) """  
            idx=cur.execute(sql,data)  
            sql_title='select last_insert_rowid()'  
            resultid=cur.execute(sql_title).fetchall()  
            #print('lastid in:',resultid[0])  
            return resultid[0][0]  
        else:  
            return False
```

# 주가 예측 모형 프로젝트

- 주가 정보 획득 1
- 주가 정보 위치 파악
- 크롤링 정보 획득

```
def Mycrawl(soup):  
    tgt=soup.select('div.table-scroll table.table-data tbody > tr ')  
    #, row_factory=sqlite3.Row  
    with sqlite3.connect(dbname) as conn:  
        for t in tgt:  
            data=[]  
            idx=t.select('td > strong')[0].get_text() #print(idx)[0].get_text()  
            idx=idx.replace('.','')# 날짜를 키값으로 사용하기  
            #print("row:",idx,end="\n")  
            #종가 시가 고가 저가 전일비 등락률 거래량(천주)  
            eprice=t.select('td.a-right')[0].get_text().replace(',','')  
            sprice=t.select('td.a-right')[1].get_text().replace(',','')  
            hprice=t.select('td.a-right')[2].get_text().replace(',','')  
            lprice=t.select('td.a-right')[3].get_text().replace(',','')
```

# 주가 예측 모형 프로젝트

- 주가 정보 획득 2
- 주가 정보 위치 파악 주가 상승 하락에 관한 차이인 경우도 추적

```
udrate=t.select('td.a-right')[5].get_text().strip().replace('%','')

trqty=t.select('td.a-right')[6].get_text().replace(',','')

updown=t.select('td.a-right > span.rise')

upfall=t.select('td.a-right > span.fall')

if(updown):

    udown=updown[0].get_text()

    udcode=1

    udown=udown.replace('상향','').strip()

elif(upfall):

    udcode=-1

    udown=t.select('td.a-right > span.fall')[0].get_text()

    udown=udown.replace('하향','').strip()

else:

    udcode=0

    udown=0

data=[idx,eprice,sprice,hprice,lprice,udcode,udown,udrate,trqty]

#print(data,end="\n")

insertdb(data)
```

## 주가 예측 모형 프로젝트

- 프로세스 전반 결정
- 주가 정보 위치 분석
- 주가 정보 획득 - 일별 분석을 통한 빅데이터 수집

```
Def MyPrs(fr=0,to=10):
    for i in range(fr,to):
        url=stockurl
        url+=str(i)
        url+='&searchSort=&searchExchange=KSE&searchTerm=&searchUpDown='
        result=requests.get(url)
        src=result.content
        #print(result.content)
        soup=BeautifulSoup(src,'html.parser')
        #print(url,end="\n")
        Mycrawl(soup)
```

# 주가 예측 모형 프로젝트

- 주가 기초 분석
- 데이터셋 준비

```
def seldb():  
    with sqlite3.connect(dbname) as conn:  
        cur=conn.cursor()  
        sql="""      select  
IDX,EPRICE,SPRICE,HPRICE,LPRICE,UDCODE,UDO  
WN,UDRATE,TRQTY  
        from kospi      order by IDX      """  
        data=cur.execute(sql).fetchall()  
        xdata=[]  
        Ey=[]  
        Sy=[]  
        Hy=[]  
        Ly=[]  
        #UDCODE,UDOWN,  
        URy=[]  
        QTy=[]  
        i=0
```

```
for d in data:  
    xdata.append(i)  
    Ey.append(d[1])  
    Sy.append(d[2])  
    Hy.append(d[3])  
    Ly.append(d[4])  
    URy.append(d[7])  
    QTy.append(d[8])  
    i+=1  
    #print(d,end="\n")  
    conn.commit()  
    cur.close()  
    print('total',i)  
return (xdata,Ey,Sy,Hy,Ly,URy,QTy)
```

## 주가 예측 모형 프로젝트

- 베이스 라인 잡기
- 이평선 확인을 위해 기초 평균용 함수 제작

```
def baseline(data,seed,day):  
    if (seed-day)<0:  
        st=0  
    else:  
        st=seed-day  
    da=data[st:seed]  
    res=np.average(da)  
    #print(res)  
    return res
```

# 주가 예측 모형 프로젝트

- 이평선 크로스 지점 확인
- 이평선 크로스 지점 중심으로 매수 매도 타이밍 발생

```
def checkcross(mdata,tgt):  
  
    lm=len(mdata)  
  
    presign=1  
  
    result=[]  
  
    x=[]  
  
    y=[]  
  
    for i in range(lm-1):  
        #print('mdata',mdata[i+1])  
  
        if(i>0):  
            presign=mdata[i-1]/abs(mdata[i-1])  
  
        else:  
            presign=1  
  
        if(i<lm):  
            nextsign=mdata[i+1]/abs(mdata[i+1])  
  
        else:
```

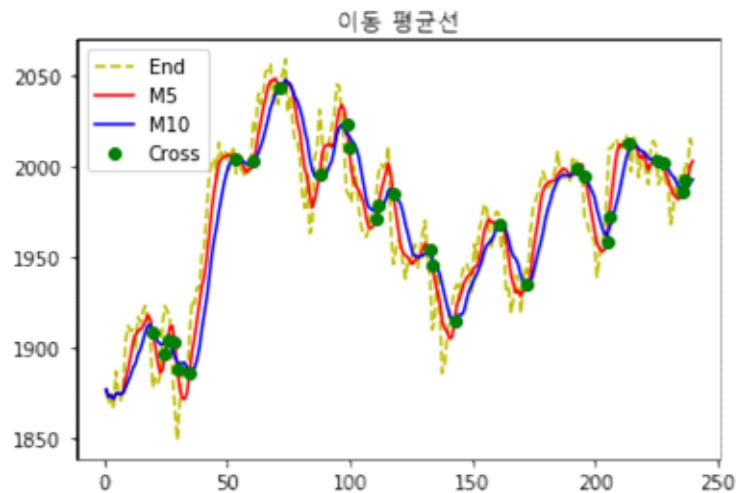
```
crossresult=presign-nextsign  
  
    #print('c result:',crossresult)  
  
    if(crossresult==2):  
        #그래프용  
        x.append(i)  
        y.append(tgt[i])  
        result.append((i,-1))  
  
    elif(crossresult== -2):  
        #그래프용  
        x.append(i)  
        y.append(tgt[i])  
        result.append((i,1))  
  
    #print('not 0')  
  
return (result,x,y)
```

# 주가 예측 모형 프로젝트

- 주식 매매 매수 실행을 통한 이익 실현
- 이익실현 정보를 통한 수익률 분석

```
def perform(ey,cp):  
    myprop=0  
    mypos=1  
    for p in cp:  
        x=p[0]  
        pos=p[1]  
        if pos==mypos:  
            myprop+=ey[x]*pos*-1  
        if pos==1:  
            print('time:',x,'매수 :',ey[x],'재산 :',myprop)  
            mypos=-1  
        else:  
            print('time:',x,'매도 :',ey[x],'재산 :',myprop)  
            mypos=1  
    print("****50)  
    print("잔액:",myprop)
```

# 주가 기본 분석을 통한 수익률 분석



time: 24 매수 : 1923 재산 : -1923  
time: 28 매도 : 1867 재산 : -56  
time: 34 매수 : 1907 재산 : -1963  
time: 53 매도 : 1996 재산 : 33  
time: 60 매수 : 2024 재산 : -1991  
time: 71 매도 : 2048 재산 : 57  
time: 87 매수 : 2031 재산 : -1974  
time: 98 매도 : 1986 재산 : 12  
time: 110 매수 : 1983 재산 : -1971  
time: 117 매도 : 1946 재산 : -25  
time: 142 매수 : 1932 재산 : -1957  
time: 160 매도 : 1974 재산 : 17  
time: 171 매수 : 1945 재산 : -1928  
time: 195 매도 : 1971 재산 : 43  
time: 204 매수 : 1982 재산 : -1939  
time: 213 매도 : 2010 재산 : 71  
time: 225 매수 : 1990 재산 : -1919  
time: 227 매도 : 2001 재산 : 82

## 주가 예측 모형 프로젝트

- 머신 러닝을 통한 주가 분석
- RNN 활용 시계열 분석
- 주가 노멀라이즈 함수

```
def normalise_windows(window_data):  
    normalised_data = []  
  
    for window in window_data:  
  
        normalised_window = [((float(p) / float(window[0])) - 1) for p in window]  
  
        normalised_data.append(normalised_window)  
  
    return normalised_data
```

# 주가 예측 모형 프로젝트

- 모델 생성
- RNN 활용 시계열 분석

```
model = Sequential()

model.add(LSTM(
    input_dim=1,
    output_dim=50,
    return_sequences=True))

model.add(Dropout(0.2))

model.add(LSTM(
    100,
    return_sequences=False))

model.add(Dropout(0.2))
```

```
model.add(Dense(
    output_dim=1))
model.add(Activation('linear'))

start = time.time()

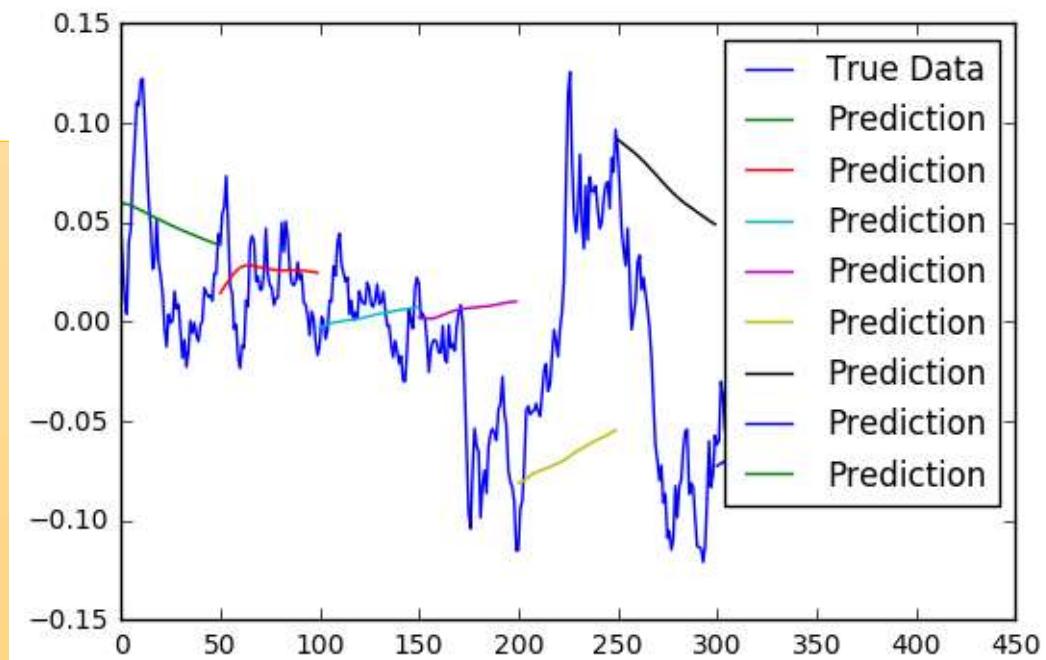
model.compile(loss='mse',
optimizer='rmsprop')

print 'compilation time : ',
time.time() - start
```

# 주가 예측 모형 프로젝트

- 머신 러닝을 통한 주가 분석
- 모델 학습

```
model.fit(  
    X_train,  
    y_train,  
    batch_size=512,  
    nb_epoch=1,  
    validation_split=0.05)
```





가상코인 예측 프로젝트

비트코인 prediction

# 비트코인 가격 예측 프로젝트

- 고 위험군 자산에 대한 가격 예측 및 투자 전략 프로젝트
- ML을 통한 매수 매도 타이밍 분석

```
get_ipython().run_line_magic('matplotlib', 'inline')

import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import plotly as py
import plotly.graph_objs as go
import requests
from keras.layers import *
from keras.models import *
from keras.callbacks import *
from datetime import datetime
from sklearn.preprocessing import MinMaxScaler
plt.style.use('bmh')
```

## 비트코인 가격 예측 프로젝트

- 고 위험군 자산에 대한 가격 예측 및 투자 전략 프로젝트
  - API로 받은 자료에 대한 분석
  - 데이터 소스 및 구조

```
https://poloniex.com/public?  
command=returnChartData&c  
urrencyPair=USDT_BTC&  
  
start=1405699200&end=9999  
999999&period=14400  
  
ret =  
requests.get('https://poloniex.  
com/public?command=return  
ChartData&  
  
currencyPair=USDT_BTC&start  
=1405699200&end=99999999  
99&period=86400')  
  
print(ret)
```

# 비트코인 가격 예측 프로젝트

- 판다스 활용 기초 데이터 분석

```
df = pd.DataFrame(js)
scaler = MinMaxScaler()
df[['close']] = scaler.fit_transform(df[['close']])
print(df)

# close date high low open \
# 0 0.003428 1424304000 244.000000 225.000000 225.000000
# 1 0.003231 1424390400 245.000000 240.250000 240.250118
# 2 0.003480 1424476800 245.000000 245.000000 245.000000
# 3 0.002955 1424563200 249.000000 235.000000 245.000000
# 4 0.002955 1424649600 235.001000 235.000000 235.000002
# 5 0.003205 1424736000 239.750000 235.000000 235.000000
```

# 비트코인 가격 예측 프로젝트

- 판다스 활용 기초 데이터 분석

```
w_size = 5  
x = []  
y = []  
  
for i in range(len(price) - w_size):  
    x.append([price[i+j] for j in range(window_size)])  
    y.append(price[window_size + i])  
  
print(x)  
print(y)  
  
# [[0.0034278368698297933, 0.0032309273586996947, 0.0034803460727978203,  
# 0.0029552540431175573, 0.0029552540431175573], [0.0032309273586996947,  
# 0.0034803460727978203, 0.0029552540431175573, 0.0029552540431175573,  
# 0.003204672757215681], ...]
```

# 비트코인 가격 예측 프로젝트

- 트레이닝 데이터 준비

```
train_test_split = 1000

x_train = x[:train_test_split, :]
y_train = y[:train_test_split]

x_test = x[train_test_split:, :]
y_test = y[train_test_split :]

print(x_test.shape)
# (566, 5)

print(x_train[0])
# [0.00342784 0.00323093 0.00348035 0.00295525 0.00295525]
```

# 비트코인 가격 예측 프로젝트

- 모델 작성
- Keras 활용
- LSTM 활용 시계열 요인 적극분석

```
model = Sequential()  
  
model.add(LSTM(128, input_shape=(5,1,)))  
model.add(Dropout(0.2))  
model.add(Dense(1, activation='linear'))  
model.compile(loss='mse', optimizer='adam')  
model.summary()  
model.fit(x_train, y_train, epochs=100, batch_size=1)  
  
# Epoch 1/100  
# 1000/1000 [=====] - 6s 6ms/step - loss: 4.0569e-04  
# Epoch 2/100  
# 1000/1000 [=====] - 6s 6ms/step - loss: 1.5022e-04  
# Epoch 3/100 ...
```

# 비트코인 가격 예측 프로젝트

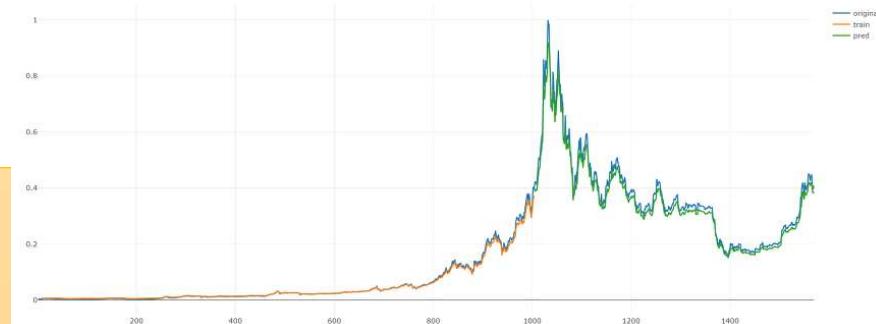
- 트레이닝 분석

```
train_predict = model.predict(x_train)  
test_predict = model.predict(x_test)
```

```
plt.figure(figsize=(10,10))  
plt.plot(price)
```

```
split_pt = train_test_split + window_size  
plt.plot(np.arange(window_size, split_pt, 1), train_predict, color='g')
```

```
plt.plot(np.arange(split_pt, split_pt + len(test_predict), 1), test_predict, color='r')
```





## 신용 위험 분석 프로젝트

Credit Risk Analyze

## 신용 위험 평가

- 함부르크 대학 신용 데이터 분석 자료 활용
- 익명의 은행 대출 자료를 분석하여 신용등급 예측
- 고유값 처리

```
def uniqueItems(column):  
    list = np.unique(column)  
    fixedList = []  
    for item in list:  
        fixeListItem = [item]  
        fixedList.append(fixeListItem)  
    return fixedList
```

```
def newWidth(column):  
    return len(np.unique(column))  
def encodeColumn(oldCol, encoder):  
    newCol = []  
    for c in oldCol:  
        newCol.append(encoder.transform(c)  
                      .toarray())  
    return np.array(newCol)
```

# 신용 위험 평가

- 함부르크 대학 신용 데이터 분석 자료 활용
- 기초 데이터 분류 및 재정의
- 기본 데이터를 분류에 맞게 정의

```
with open('german.data.txt') as rawDataFile:  
  
    csvReader = csv.reader(rawDataFile, delimiter=' ', quotechar='|')  
  
    rows = []  
  
    for row in csvReader:  
  
        cols = []  
  
        for col in row:  
  
            # Change the value here into a floating point number  
  
            if col[0] == 'A':  
  
                value = float(col[1:])  
  
            else:  
  
                value = float(col)  
  
            cols.append(value)  
  
        rows.append(cols)  
  
  
    rowCount = len(rows)  
  
    colCount = len(rows[0])
```

#컬럼을 이진화처리

```
def binarizeColumn(oldCol, trueVal):  
  
    return [1 if x==trueVal else -1 for x in oldCol]  
  
# 원핫 인코딩 처리  
  
def makeOnehotencoding():  
  
    ...
```

## 신용 위험 평가

- 회귀 분석 모형
- 회귀 모델 작성

```
dataset = genfromtxt('german.data-numeric.csv', delimiter=',')
print
rows, cols = dataset.shape
x_width = cols-2
x = tf.placeholder(tf.float32, [None, x_width])

W = tf.Variable(tf.ones([x_width, num_classes]))
b = tf.Variable(tf.zeros([num_classes]))
y_ = tf.nn.softmax(tf.matmul(x, W) + b)
y = tf.placeholder(tf.float32, [None, num_classes])
```

# 신용 위험 평가

- 세션초기화 및 변수설정

```
square_diff = tf.square(y - y_)

good, bad = tf.split(square_diff, 2, 1)

costwise_loss = false_neg_cost*tf.reduce_sum(good) + false_pos_cost*tf.reduce_sum(bad)

train_step = tf.train.GradientDescentOptimizer(learning_rate).minimize(costwise_loss)

init = tf.initialize_all_variables()

sess = tf.InteractiveSession()

sess.run(init)

kf = KFold( n_splits=num_k_folds ) #####>12.0

fold_counter = 1

val_cmats = []

val_precisions = []

val_recalls = []

val_f_scores = []
```

## 신용 위험 평가

- Kfold 분석
- 회귀분할
- 머신러닝 활용

```
for i in range(3000):
    batch_xs, batch_ys = random_batch(train_dataset, batch_size)
    sess.run(train_step, feed_dict={x: batch_xs, y: batch_ys})
    yT = np.argmax(val_dataset[:, 59:61], axis=1)
    y_p = tf.argmax(y_,1)
    yP = sess.run(y_p, feed_dict={x: val_dataset[:, 0:59], y: val_dataset[:, 59:61]})
    [precision, recall, f_score, _] = precision_recall_fscore_support(yT, yP, average='macro')
    print("Validation k-fold #{} - precision: {}, recall: {}, f-score: {}" .format(
        fold_counter, precision, recall, f_score))
    conmat = confusion_matrix(yT, yP)
```



자연어처리 금리 분석

NLP IR Analysis

## NLP 금리 예측 프로젝트

- 금통위 자료 수집
- NLP할 자료 수집

```
def get_minutes_list(from_date='20050101'):  
    prefix_addr = "https://www.bok.or.kr"  
    from_date = datetime.strptime(from_date, '%Y%m%d')  
  
    for pageIndex in range(1, 31):  
        url =  
            'https://www.bok.or.kr/portal/bbs/B0000245/list.do?menuNo=200761&pageIndex={}'.format(pageIndex)  
        user_agent = 'Mozilla/5.0'  
        headers ={'User-Agent' : user_agent}  
        page = requests.get(url, headers=headers)
```

## 비정형 자연어 처리

- 자연어 처리 프로젝트 프로젝트
- '인공지능 비정형 텍스트 분석'과 '인공지능 비정형 텍스트 분석 실습'
- - 자료논문
  - <https://www.bok.or.kr/portal/singl/pblictn/view.do?nttId=10049321&searchOptn10=TALK&menuNo=200633&pageIndex=>
- - 논문작성자 설명 자료:
  - <https://github.com/entelecheia/eKoNLPy/blob/master/docs/eKoNLPy.pdf>
    - 작성 예제 코드 :
  - <https://github.com/fininsight/text-mining-tutorial/tree/master/%ED%85%8D%EC%8A%A4%ED%8A%B8%EB%A7%88%EC%9D%B4%EB%8B%9D%20%ED%99%9C%EC%9A%A9%20%EA%B8%B0%EC%A4%80%EA%B8%88%EB%A6%AC%20%EC%98%88%EC%B8%A1>
    - 한국은행 조사연구자료 게시물 중 통화에 관련된 주제의 게시물을 모아놓은 게시판 : <https://www.bok.or.kr/portal/singl/pblictn/view.do?nttId=10049321&searchOptn10=TALK&menuNo=200633&pageIndex=>

**THANK YOU**