

KSBi-BIML 2021

Bioinformatics & Machine Learning (BIML)
Workshop for Life Scientists

생물정보학 & 머신러닝 워크샵(온라인)

Introduction to Artificial Intelligence, Machine Learning, and Deep Learning

정성원

Bioinformatics & Machine Learning for Life Scientists

BIML-2021

안녕하십니까?

한국생명정보학회의 동계 워크샵인 BIML-2021을 2월 15부터 2월 19일까지 개최합니다. 생명정보학 분야의 융합이론 보급과 실무역량 강화를 위해 도입한 전문 교육 프로그램인 BIML 워크샵은 2015년에 시작하였으며 올해로 7차를 맞이하게 되었습니다. 유례가 없는 코로나 대유행으로 인해 올해의 BIML 워크숍은 온라인으로 준비했습니다. 생생한 현장 강의에서만 느낄 수 있는 강의자와 수강생 사이의 상호교감을 가질 수 없다는 단점이 있지만, 온라인 강의의 여러 장점을 살려서 최근 생명정보학에서 주목받고 있는 거의 모든 분야를 망라한 강의를 준비했습니다. 또한 온라인 강의의 한계를 극복하기 위해서 실시간 Q&A 세션 또한 마련했습니다.

BIML 워크샵은 전통적으로 크게 생명정보학과 AI, 두 개의 분야로 구성되어오고 있으며 올해 역시 유사한 방식을 채택했습니다. AI 분야는 Probabilistic Modeling, Dimensionality Reduction, SVM 등과 같은 전통적인 Machine Learning부터 Deep Learning을 이용한 신약개발 및 유전체 연구까지 다양한 내용을 다루고 있습니다. 생명정보학 분야로는, Proteomics, Chemoinformatics, Single Cell Genomics, Cancer Genomics, Network Biology, 3D Epigenomics, RNA Biology, Microbiome 등 거의 모든 분야가 포함되어 있습니다. 연사들은 각 분야 최고의 전문가들이라 자부합니다.

이번 BIML-2021을 준비하기까지 너무나 많은 수고를 해주신 BIML-2021 운영위원회의 김태민 교수님, 류성호 교수님, 남진우 교수님, 백대현 교수님께 커다란 감사를 드립니다. 또한 재정적 도움을 주신, 김선 교수님 (AI-based Drug Discovery), 류성호 교수님, 남진우 교수님께 감사를 표시하고 싶습니다. 마지막으로 부족한 시간에도 불구하고 강의 부탁을 흔쾌히 허락하시고 훌륭한 강의자료를 만드는데 노력하셨을 뿐만 아니라 실시간 온라인 Q&A 세션까지 참여해 수고해 주시는 모든 연사분들께 깊이 감사드립니다.

2021년 2월

한국생명정보학회 회장 김동섭

강의개요

Introduction to artificial intelligence, machine learning, and deep learning

본 강의는 다양한 분야에서 이용되는 데이터 기반 인공지능 학습의 근간을 이루는 기계학습을 공부하는 입문 과정이다. 인공지능에 대한 개념 및 핵심 기초 이론을 다루어 널리 사용되는 여러 기계학습 알고리즘의 특성을 이해하고 향후 그에 맞는 응용을 할 수 있는 역량을 기르는 데 목표를 둔다. 이를 위하여 본 강의의 구성은 인공지능 기술의 핵심을 이루는 패턴인식/식별 관점에서의 기계학습에 대한 소개, 다양한 패턴인식 기법의 기초가 되는 regression 기법 및 그 연장선에 있는 deep learning 을 포함한 신경망 모델, 그리고 classification 문제의 특성 및 그에 대한 접근 방법으로 이루어진다.

강의는 다음의 내용을 포함한다:

- Machine learning 소개
- Regression analysis
- Classification: Methods and strategies
- Neural networks and deep learning

*참고강의교재:

패턴 인식에 대한 체계적인 학습에 도움이 되는 reference: "Pattern Classification, second edition", Duda, Hart, and Stork, Wiley-Interscience, 2000

*교육생준비물:

별도 준비물 없음

* 강의: 정성원 교수 (가천대학교 의예과 유전체의과학전공)

Curriculum Vitae

Speaker Name: Sungwon Jung, Ph.D.



► Personal Info

Name Sungwon Jung
Title Associate Professor
Affiliation Gachon University College of Medicine

► Contact Information

Address 38-13 Dokjeom-ro 3beon-gil, Namdong-gu, Incheon 21565
Email sjung@gachon.ac.kr
Phone Number 032-458-2740

Research interest : Pathway analysis, Systems biology, Machine learning

Educational Experience

1998 B.S. in Computer Science, KAIST, Republic of Korea
2000 M.S. in Computer Science, KAIST, Republic of Korea
2007 Ph.D. in Computer Science, KAIST, Republic of Korea

Professional Experience

2007-2008 Post Doctoral Research Associate, IBM-KAIST Bio-Computing Research Center, KAIST
2008-2013 Post Doctoral Fellow, Translational Genomics Research Institute, USA
2013-2015 Principal Scientist, Samsung Genome Institute, Samsung Medical Center
2015- Associate Professor, Department of Genome Medicine and Science, Gachon University College of Medicine

Selected Publications (5 maximum)

1. Jong min Lee, Hye Kyung Hong, Sheng-Bin Peng, Tae Won Kim, Woo Yong Lee, Seong Hyun Yun, Hee Cheol Kim, Jiangang Liu, Philip J. Ebert, Amit Aggarwal, Sungwon Jung and Yong Beom Cho, "Identifying metastasis-initiating miRNA-target regulations of colorectal cancer from expressional changes in primary tumors", Scientific Reports, 10:14919, 2020
2. Hyojung Kim, Sora Kim and Sungwon Jung, "Instruction of microbiome taxonomic profiling based on 16S rRNA sequencing", Journal of Microbiology, 58(3):193-205, 2020
3. Sungwon Jung, "KEDDY: a knowledge-based statistical gene set test method to detect differential functional protein-protein interactions", Bioinformatics, 35(4):619-629, 2019
4. Dongchan Kim, Sungwon Jung and HyunWook Park, "DRF-GRAPPA: A Parallel MRI Method with a Direct Reconstruction Filter", Journal of the Korean Physical Society, 73(1):130-137, 2018
5. Sungwon Jung, "Implications of publicly available genomic data resources in searching for therapeutic targets of obesity and type 2 diabetes", Experimental & Molecular Medicine, 50:43, 2018

KSBi-BIML 2021

Introduction to Artificial Intelligence, Machine Learning, and Deep Learning

정성원 (가천대학교 의예과 유전체의과학전공)

본 강의 자료는 한국생명정보학회가 주관하는 KSBi-BIML 2021 워크숍 온라인 수업을 목적으로 제작된것으로 해당 목적 이외의 다른 용도로 사용할 수 없음을 분명하게 알립니다. 수업 목적으로 배포 및 전송 받은 경우에도 이를 다른 사람과 공유하거나 복제, 배포, 전송할 수 없습니다.

만약 이러한 사항을 위반할 경우 발생하는 모든 법적 책임은 전적으로 불법 행위자 본인에게 있음을 경고합니다.

강의 개요

- ◆ 인공지능의 근간을 이루는 기계학습에 대한 소개
- ◆ 기계학습 기초 이론
 - Regression
 - Classification evaluation
 - Decision tree & Ensemble methods
 - Deep learning

Machine Learning 소개

Machine Learning 이란?

특정 작업을 잘 할 수
있도록 사례 데이터를
이용하여 컴퓨터를
학습시키는 것

넷플릭스의 개인화된 콘텐츠 추천

본인 취향의 콘텐츠 3개를 고르면,

해당 콘텐츠를 참고해서 취향에 맞는 콘텐츠
추천

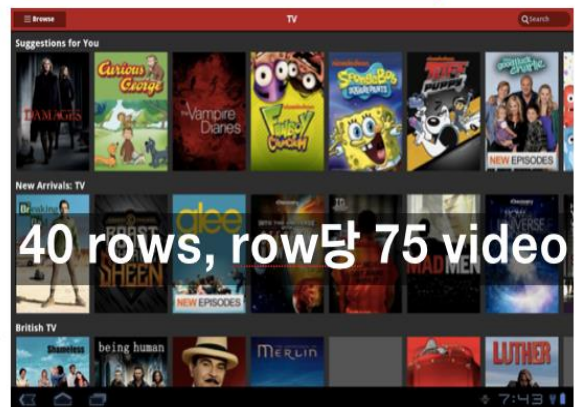
이후 사용 패턴에 따라 추천 업데이트

- 시청 기록, 콘텐츠 평가 결과 등
- 유사 취향 타 회원의 선호 정보 참고
- 장르, 카테고리, 배우, 출시연도 참고
- 하루 중 시청 시간대
- 시청시 사용하는 디바이스
- 시청 시간
- ...

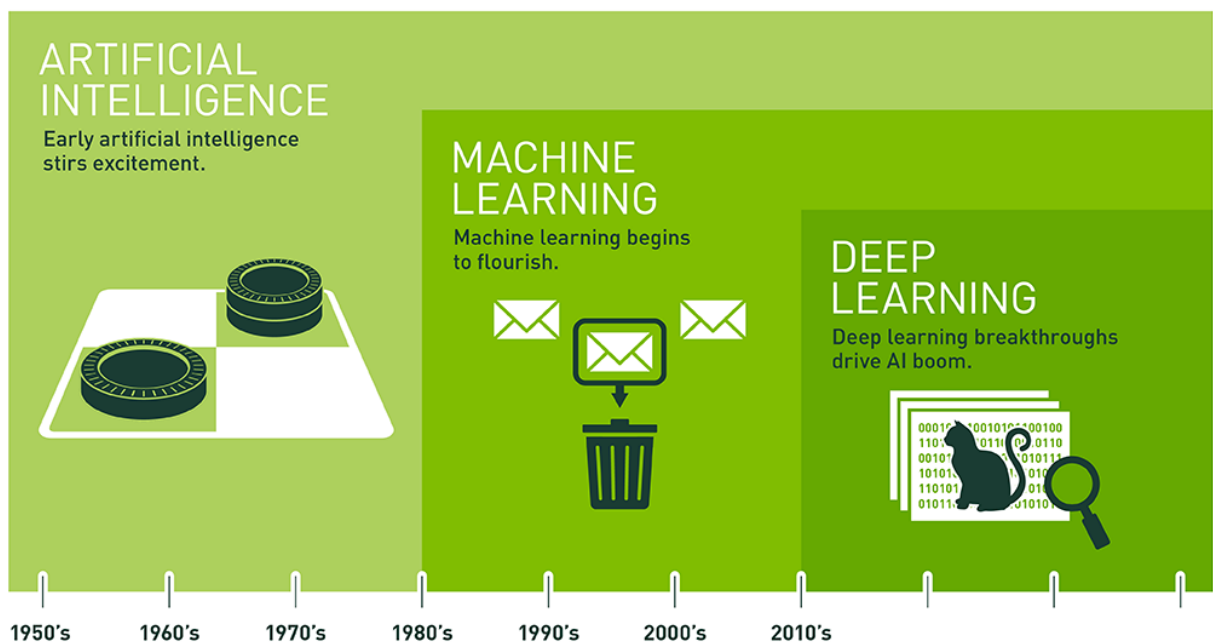


개인화 랭킹

각각 다른
미리보기



2017년 대통령선거 직전. 주요 후보들의 구글 트렌드 검색 결과

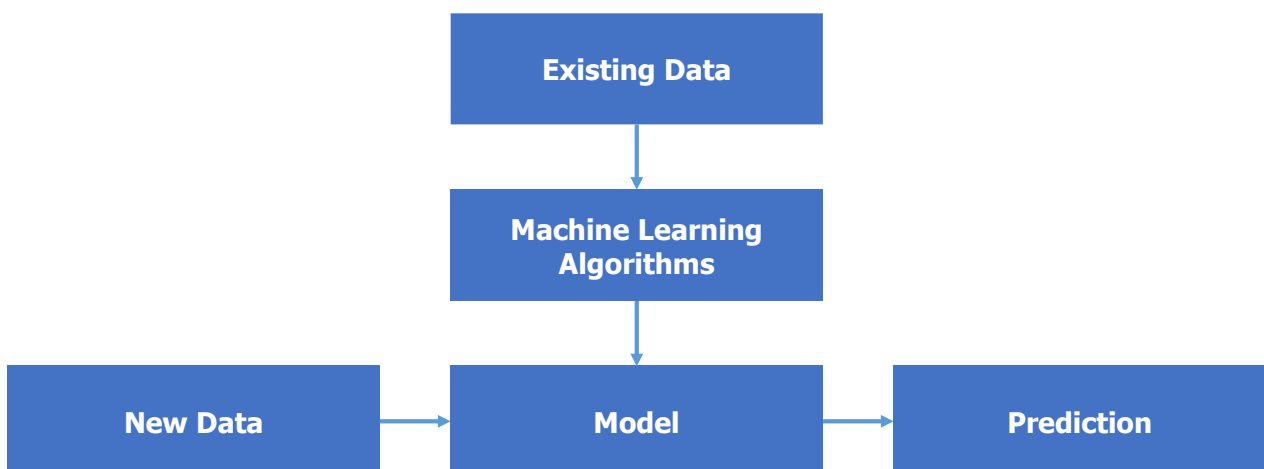


Source: <https://goo.gl/gz7EZT>

Machine Learning 기본 개념이 필요한 이유

- ◆ Deep learning 은 마법의 탄환이 아님
 - 여러 machine learning 기법 중 하나일 뿐
 - 장점과 단점이 있음
- ◆ Artificial intelligence - Machine learning 을 제대로 사용하는 길
 - I. 풀고자 하는 문제의 특성을 이해한다.
 - II. 풀고자 하는 문제에 입력이 되는 데이터에는 무엇이 있는지 파악한다.
 - III. 데이터의 특성과 양을 파악한다.
 - IV. 여기에 적합한 기법을 사용한다.
- ◆ Machine learning 에 있는 다양한 기법들의 이론적 배경, 특징을 이해하고 있을 때 효율적인 문제 해결이 가능

일반적인 Machine Learning 과정



Model, Algorithms

◆ Model

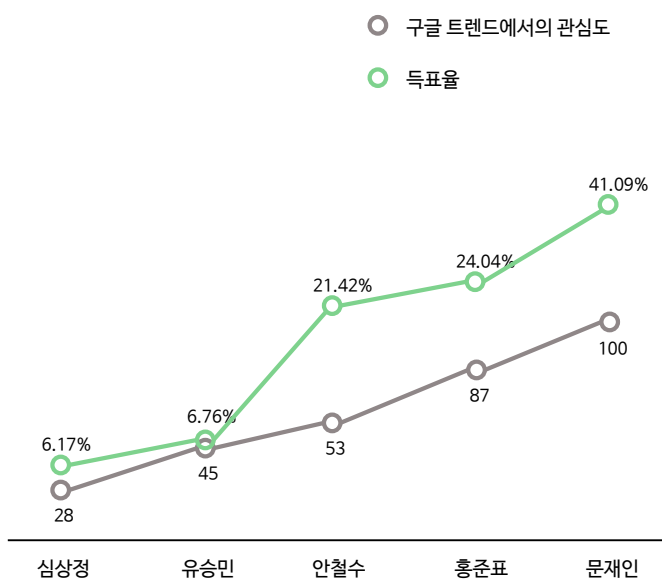
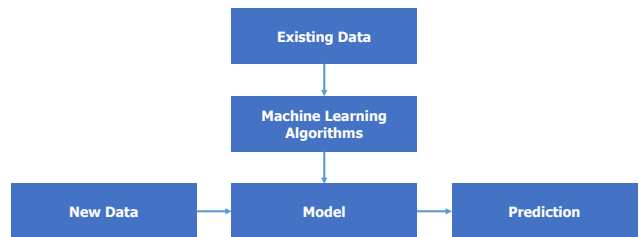
예측을 위한 수학 공식, 함수 등

1차 방정식, 확률분포, condition rule, ...

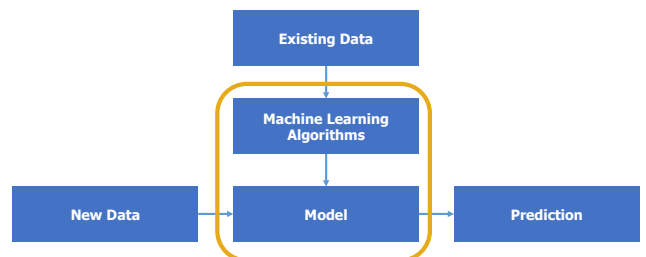
◆ Algorithms

어떠한 문제를 풀기 위한 체계

Model 을 생성하기 위한 (데이터 학습) 과정에 대한 description



$$y = ax + b$$



Machine Learning 의 가장 기본이 되는 것

- ◆ 데이터의 패턴을 반영하는 선(직선/곡선) 혹은 모델을 찾아내기

Pattern recognition

- ◆ 서로 다른 데이터 그룹을 구분할 수 있는 선(직선/곡선) 혹은 모델을 찾아내기

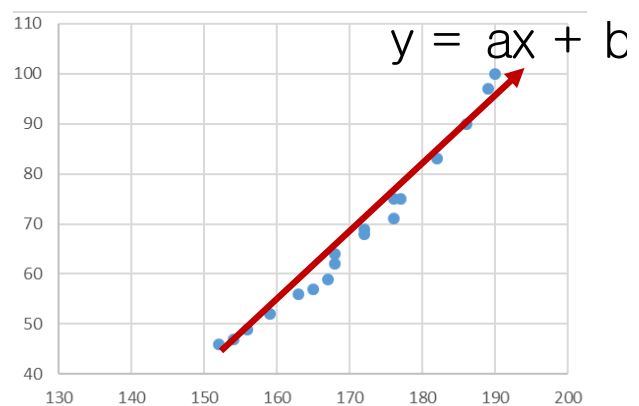
Pattern classification

그렇다면 어떤 선 혹은 모델을 데이터로부터 어떻게 찾아내야 하는가?

Regression - 회귀, 추세선을 찾는 것

Height	Weight	Height	Weight
152	46	172	69
154	47	172	68
156	49	176	71
159	52	176	75
163	56	177	75
165	57	182	83
167	59	186	90
168	64	189	97
168	62	190	100

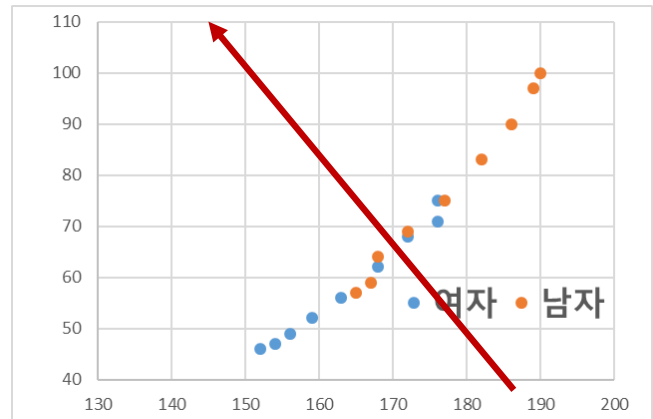
Data source: <http://goo.gl/gDscUQ>



Classification - 분류, 데이터를 패턴에 따라 나누는 것

- ◆ 기존 데이터를 바탕으로 데이터를 서로 다른 군으로 나누어 보기

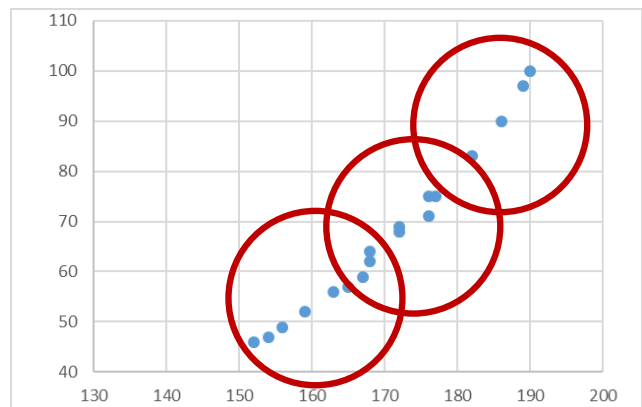
Sex	Height	Weight	Sex	Height	Weight
여자	152	46	남자	172	69
여자	154	47	여자	172	68
여자	156	49	여자	176	71
여자	159	52	여자	176	75
여자	163	56	남자	177	75
남자	165	57	남자	182	83
남자	167	59	남자	186	90
남자	168	64	남자	189	97
여자	168	62	남자	190	100



Clustering - 군집, 데이터를 모으는 것

- ◆ 데이터 사이의 유사도/거리에 기반하여 비슷한 데이터끼리 군집화

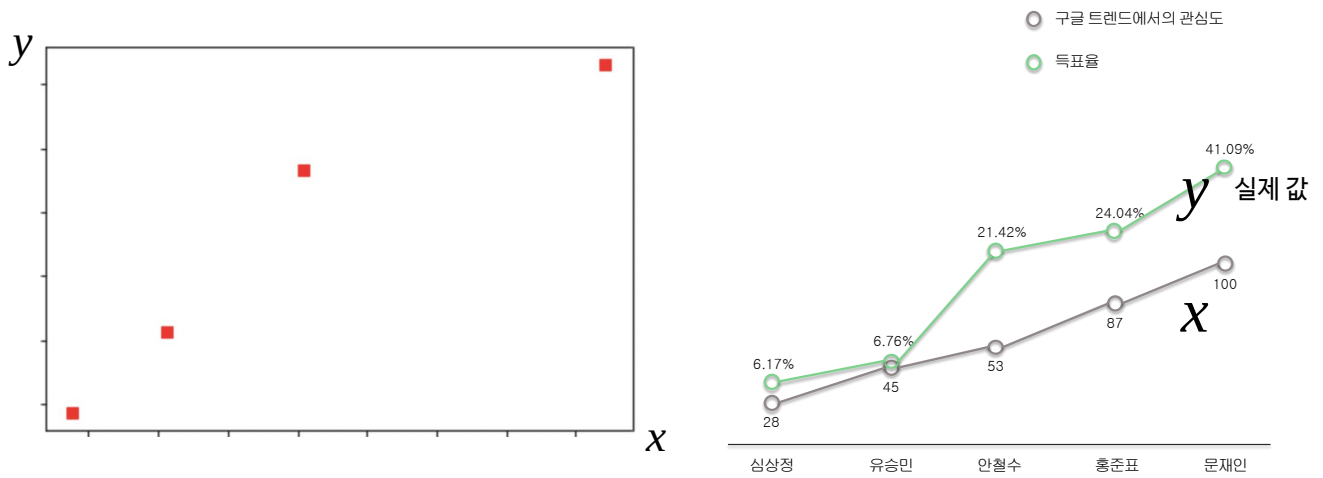
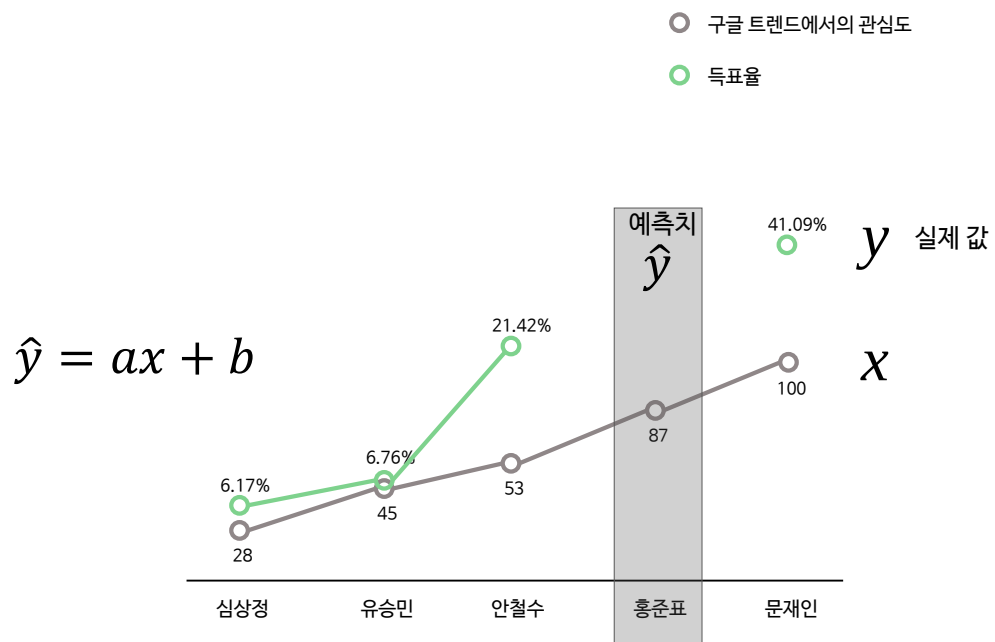
Sex	Height	Weight	Sex	Height	Weight
여자	152	46	남자	172	69
여자	154	47	여자	172	68
여자	156	49	여자	176	71
여자	159	52	여자	176	75
여자	163	56	남자	177	75
남자	165	57	남자	182	83
남자	167	59	남자	186	90
남자	168	64	남자	189	97
여자	168	62	남자	190	100



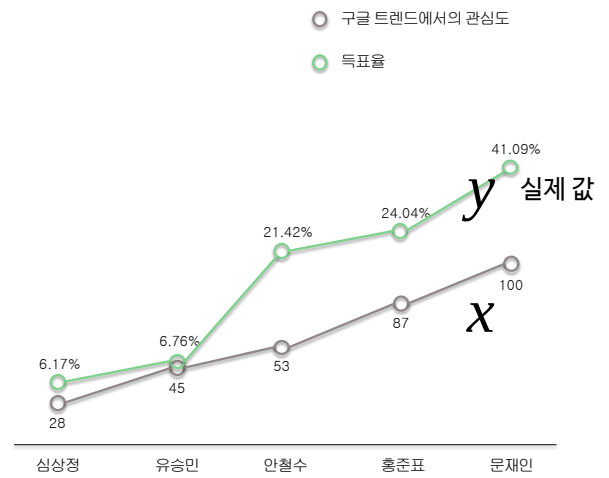
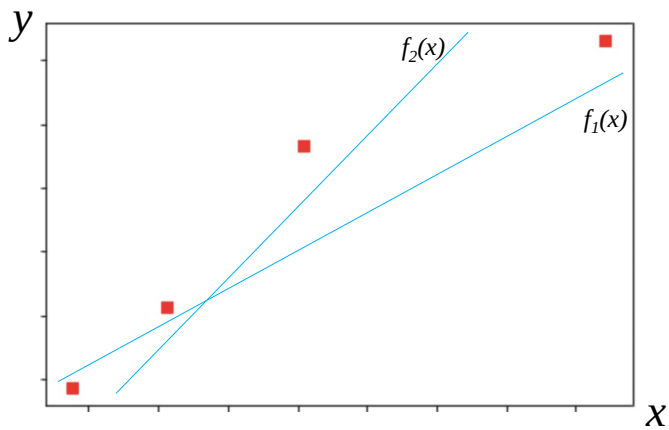
Machine Learning 기초 - Regression

Gradient Descent-based Learning

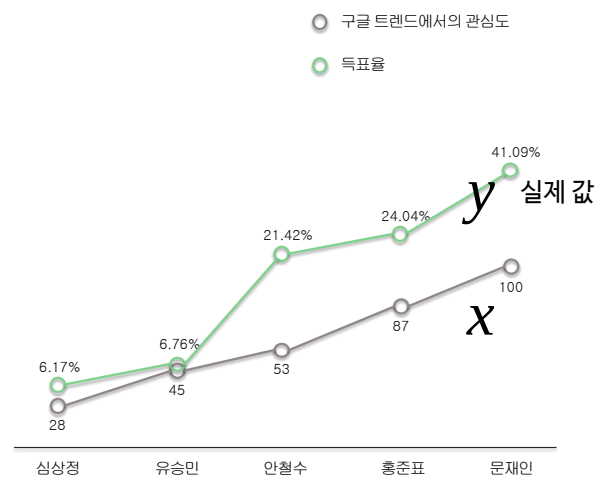
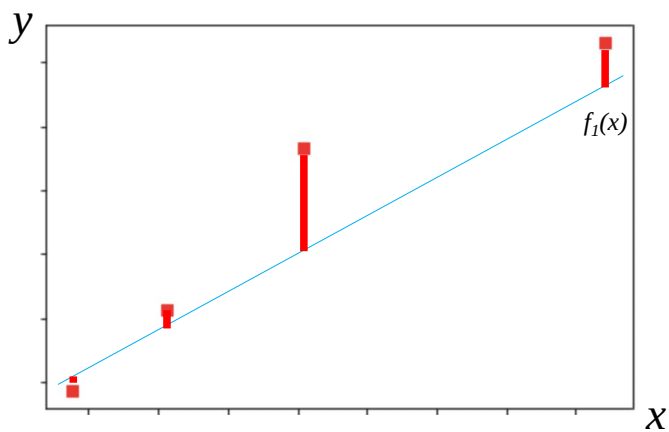
- ◆ 실제 값과 학습된 모델 예측치의 오차를 최소화
- ◆ 학습을 통해 모델의 최적 파라미터를 찾는 것이 목표



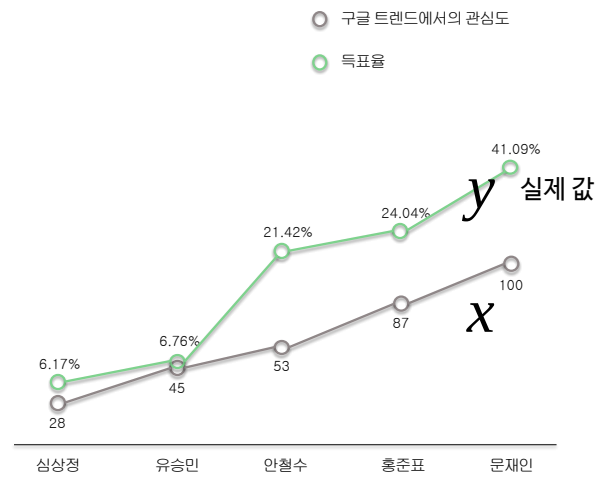
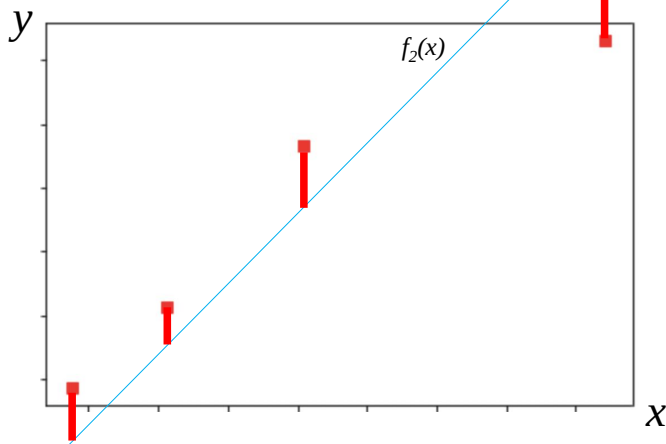
$$f(x) = \hat{y} = ax + b$$



$$f(x) = \hat{y} = ax + b$$



$$f(x) = \hat{y} = ax + b$$



Linear Regression

◆ (파라미터 a 와 b 의) 학습 원리

- 예측값과 실제값의 오차를 계산
- 오차를 최소화하도록 모델 파라미터(a, b)를 조정(학습)
 - 파라미터: 학습을 통해 최적화 해 주어야 하는 변수

$$\hat{y} = ax + b$$

y

Linear Regression

◆ (파라미터 a 와 b 의) 학습 원리

- 예측값과 실제값의 오차를 계산
- 오차를 최소화하도록 모델 파라미터(a, b)를 조정(학습)
 - 파라미터: 학습을 통해 최적화 해 주어야 하는 변수

오차의 합

$$(\hat{y}^{(1)} - y^{(1)}) + (\hat{y}^{(2)} - y^{(2)}) + (\hat{y}^{(3)} - y^{(3)}) + (\hat{y}^{(4)} - y^{(4)})$$

오차는 양수 또는 음수 가능 → 상쇄될 수 있음

$$(\hat{y}^{(1)} - y^{(1)})^2 + (\hat{y}^{(2)} - y^{(2)})^2 + (\hat{y}^{(3)} - y^{(3)})^2 + (\hat{y}^{(4)} - y^{(4)})^2$$

제곱의 합으로 변환

Linear Regression 모델

- ◆ Squared error 를 오차로 간주하고, 다음 식을 최소화하자.
- ◆ 어떤 함수의 최대 최소 문제의 해결은? 그 함수를 미분하여 시도
- ◆ 목표는 다음 식을 최소화하는 파라미터들 (a, b 혹은 w_0, w_1)

$$\sum_{i=1}^n (w_1 x^{(i)} + w_0 \times 1 - y^{(i)})^2$$

가설함수

- ◆ 우리가 현재 설정하는 예측함수를 가설함수로 부른다.

$$f(x) = h_{\theta}(x)$$

Loss Function

- ◆ 한 데이터 포인트에서, 실제 값과 예측값의 차이를 계산하는 함수
- ◆ 목적에 따라 여러 종류의 함수가 존재할 수 있음
 - 예: Squared error

$$(h_{\theta}(x^{(i)}) - y^{(i)})^2$$

Cost Function

- ◆ 주어진 데이터에 대한, 예측 값과 실제 값 차이의 평균

$$J(w_0, w_1) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$$

(Loss function = squared error 의 예)

- ◆ 우리의 목적 = Cost 를 줄이는 것
- ◆ 정의된 cost function 을 최소화 해 주는 파라미터 값을 찾는 것

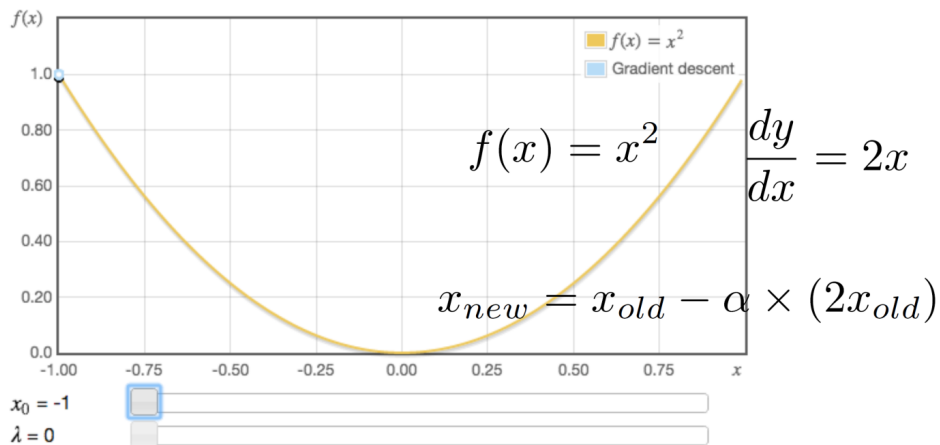
$$\arg \min_{\theta} \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$$

Cost Function 최소화를 위한 파라미터를 찾는 방법

- ◆ 다양한 최적화 기법을 사용 가능
- ◆ Practically 많이 사용되는 방법: Gradient descent

Gradient Descent

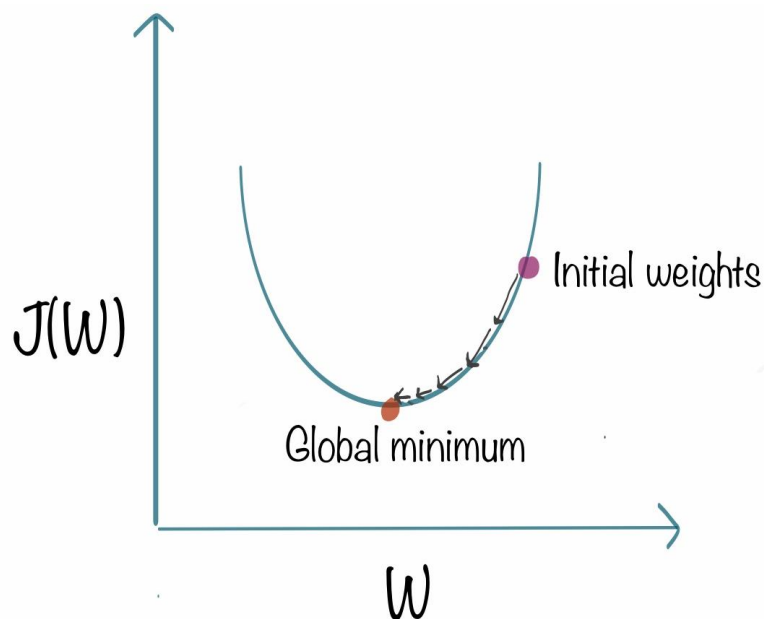
x 를 파라미터(w)로 갖는 다음과 같은 cost function (y)을 상상하면..



<http://www.onmyphd.com/?p=gradient.descent>

Gradient Descent 를 이용한 최소값 찾기

$$f(x) = x^2 \quad \frac{dy}{dx} = 2x$$
$$x_{new} = x_{old} - \alpha \times (2x_{old})$$
$$\begin{bmatrix} 1 \\ 1 - 0.1 * 2 * 1 = 0.8 \\ 0.8 - 0.1 * 2 * 0.8 = 0.64 \\ 0.64 - 0.1 * 2 * 0.64 = 0.512 \\ \vdots \end{bmatrix}$$



<https://www.kdnuggets.com/2018/06/intuitive-introduction-gradient-descent.html>

Learning Process

- Epoch
 - One processing round of the entire training set
 - One Epoch is when an **ENTIRE** dataset is passed through the model only **ONCE**
- Batch
 - Total number of training examples present in a single batch
- Iterations
 - Iterations is the number of batches needed to complete one epoch



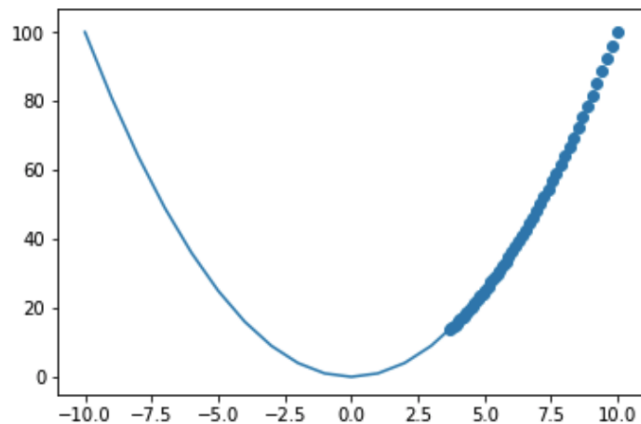
Gradient Descent 를 이용 할 때 정해야 할 것

- ◆ Learning rate (alpha) 를 어떤 값으로 정할 것인가?
- ◆ 파라미터를 업데이트하는 스텝을 얼마나 많이 반복할 것인가?

$$x_{new} = x_{old} - \alpha \times (2x_{old})$$

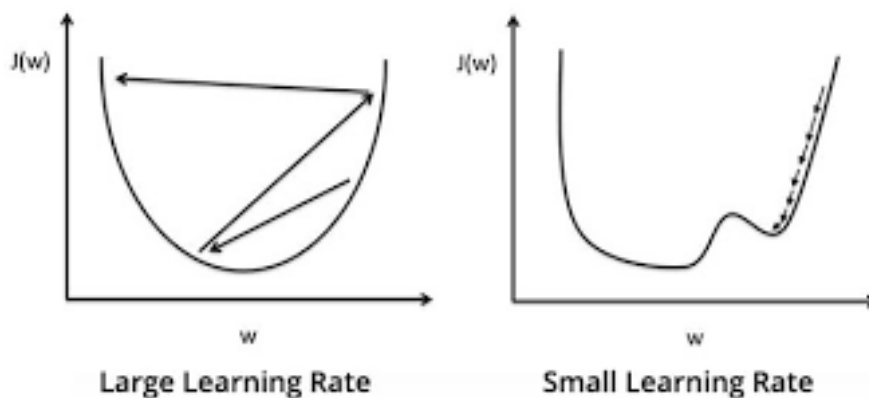
Gradient Descent 를 이용할 때 정해야 할 것

- ◆ Learning rate, Epoch 이 너무 작은 경우
- ◆ 최소 cost 에 수렴하기 전에 learning 프로세스가 끝나거나, 수렴하기까지 오랜 시간이 걸림



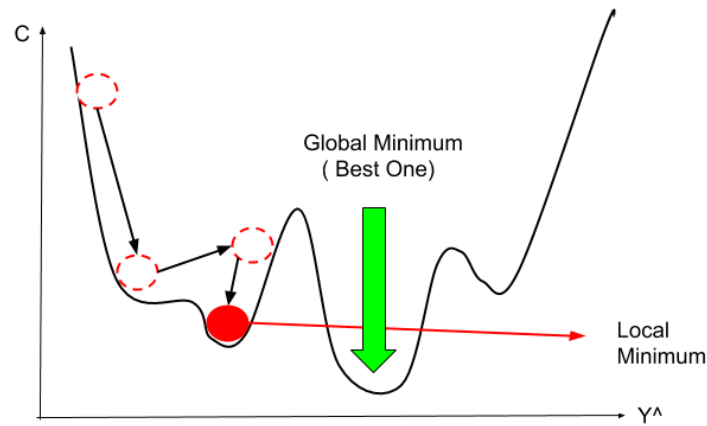
Gradient Descent 를 이용할 때 정해야 할 것

- ◆ Learning rate, epoch 이 너무 큰 경우
- ◆ 제대로 수렴하지 못 하거나 오히려 발산하는 경우가 생길 수 있음



Gradient Descent 를 이용할 때 고려해야 할 것

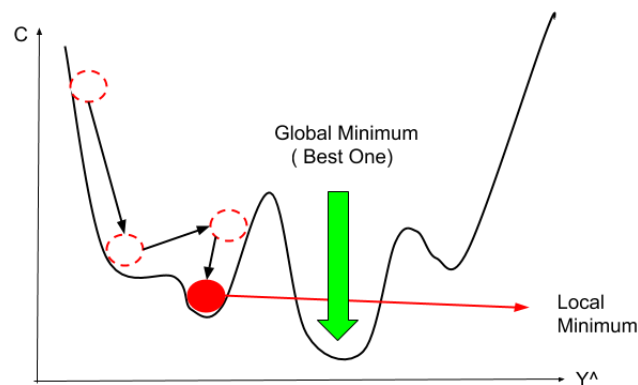
- ◆ Target(Cost) function 이 표현하는 공간의 형태가 복잡한 경우는 어려운 문제
 - 예) 2차원상의 그래프에 굴곡이 많은 경우



<https://www.mltut.com/stochastic-gradient-descent-a-super-easy-complete-guide/>

Gradient Descent 를 이용할 때 고려해야 할 것

- ◆ Target function 이 표현하는 공간의 형태가 복잡한 경우 벌어질 수 있는 일
 - 수렴에 실패
 - Local minimum 으로 수렴 (시작점에 따라 수렴하는 곳이 다를 수 있음)



<https://www.mltut.com/stochastic-gradient-descent-a-super-easy-complete-guide/>

Gradient Descent 를 이용한 Linear Regression

- ◆ 임의의 값으로 모델 파라미터를 초기화
- ◆ Cost function 이 최소화 될 때 까지 gradient 를 이용하여 학습
- ◆ 더 이상 cost function 이 줄어들지 않거나, 정해진 learning epochs 를 달성하였을 때 종료

Gradient Descent 를 이용한 Linear Regression

편미분을 이용한 파라미터별 gradient

$$\frac{\partial J}{\partial w_0} = \frac{1}{m} \sum_{i=1}^m (w_1 x^{(i)} + w_0 - y^{(i)})$$
$$\frac{\partial J}{\partial w_1} = \frac{1}{m} \sum_{i=1}^m (w_1 x^{(i)} + w_0 - y^{(i)}) x^{(i)}$$

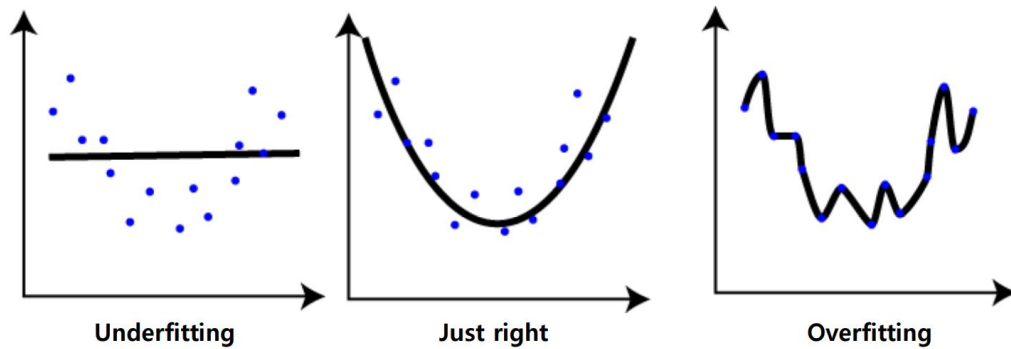
이렇게 했던 것 처럼, $x_{new} = x_{old} - \alpha \times (2x_{old})$

수렴 할 때 까지 or Loop 한계에 도달할 때 까지
파라미터들을 업데이트

$$\begin{aligned} & \text{loop until convergence} \{ \\ & \quad \text{do } \theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \\ & \} \end{aligned}$$

그 외 고려사항: Overfitting 회피

- ◆ Overfitting: 학습 데이터에 과다하게 최적화 → 새로운 데이터에 대한 예측 정확도 감소

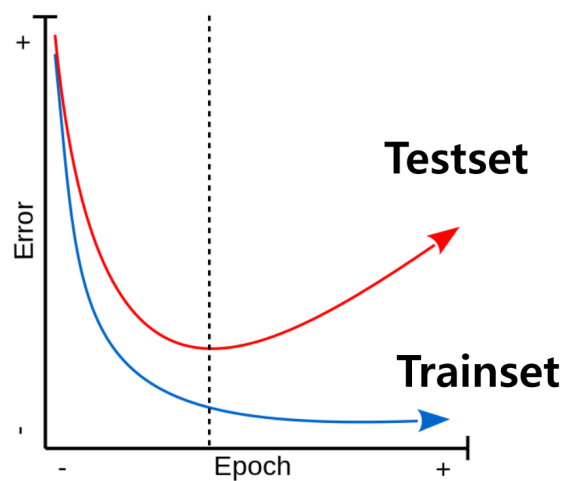


Occam's razor
보다 단순한 모델로 설명이 가능한 경우, 복잡한 모델을 사용하지 말라

<https://goo.gl/aP8iFa>

그 외 고려사항: Overfitting 회피

- ◆ Training error & Test error



그 외 고려사항: Overfitting 회피

- ◆ 더 많은 학습 데이터를 활용한다.
- ◆ 가능하면 단순한 모델을 사용한다. (파라미터의 수를 줄인다)
- ◆ 파라미터에 대하여 적절한 제약을 활용한다.
- ◆ Regularization 기법의 사용 (모델의 복잡도에 따른 페널티를 cost function 에 추가)

- L1 regularization

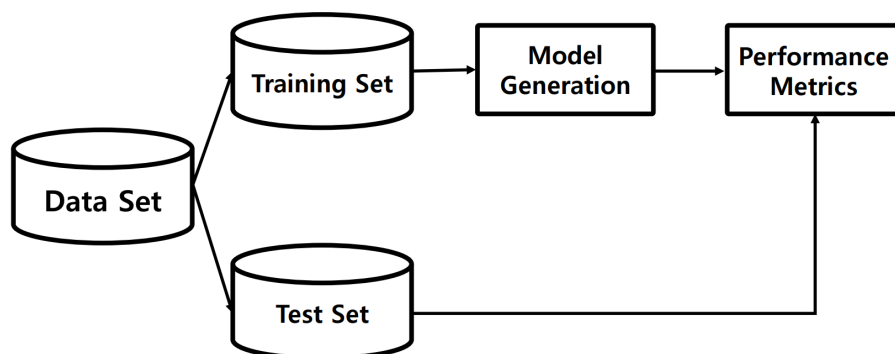
$$J(\theta) = \frac{1}{2m} \sum_{i=1}^m \left(h_{\theta}(x^{(i)}) - y^{(i)} \right)^2 + \frac{\lambda}{2} \sum_{j=1}^n |\theta_j|$$

- L2 regularization

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^m \left(h_{\theta}(x^{(i)}) - y^{(i)} \right)^2 + \frac{\lambda}{2} \sum_{j=1}^n \theta_j^2$$

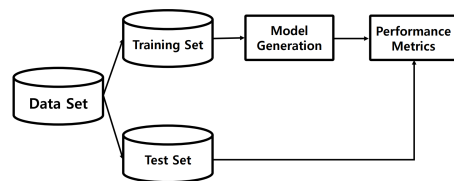
Training Data & Test Data

- ◆ 일반적인 machine learning 절차

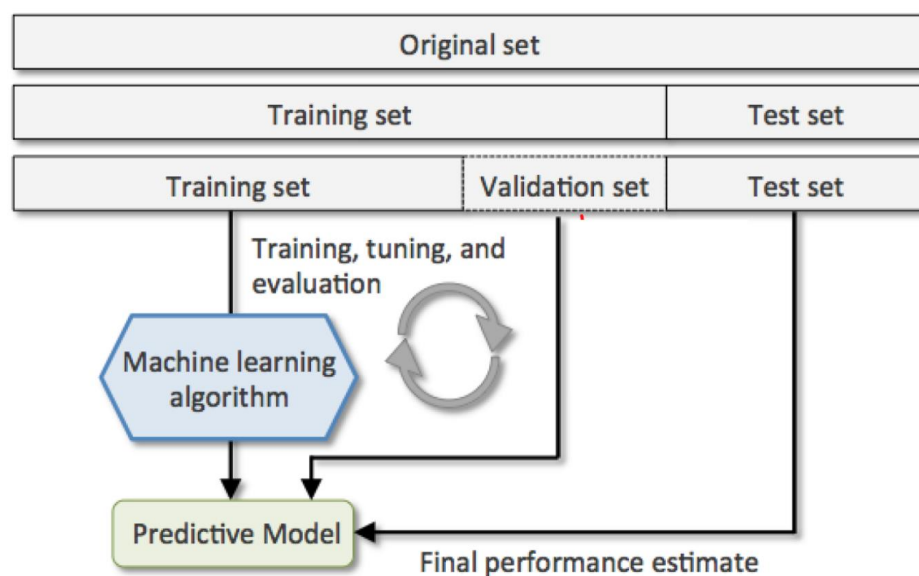


Training Data & Test Data

- ◆ 주어진 데이터를 training data 와 test data 로 나누어 모델을 학습하고 학습된 모델의 성능을 평가
- ◆ Machine learning 수행에 있어 가장 일반적인 데이터 활용 방법
- ◆ Training data 와 test data 를 나누는 비율, 방법 등은 데이터의 양 및 특성을 고려하여 결정
- ◆ Training data 의 세분화
 - Training data: 모델의 학습에 사용
 - Validation data: 모델 학습 과정에서 모델의 성능을 중간 평가하여 참고하기 위하여 활용



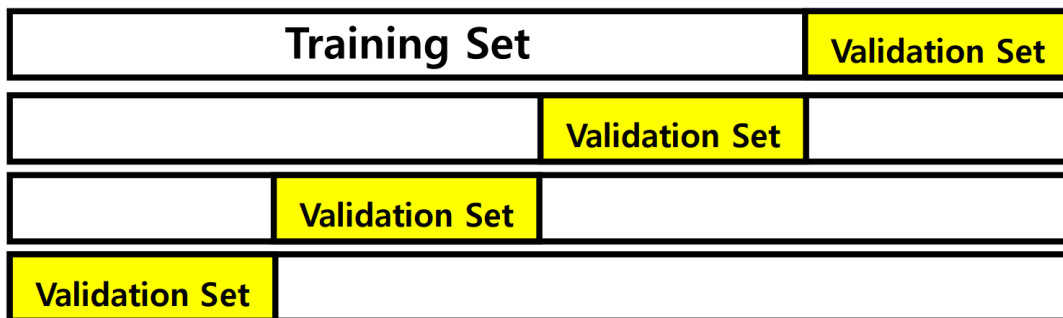
Training Data & Test Data



Training Data & Test Data

◆ K-fold cross validation

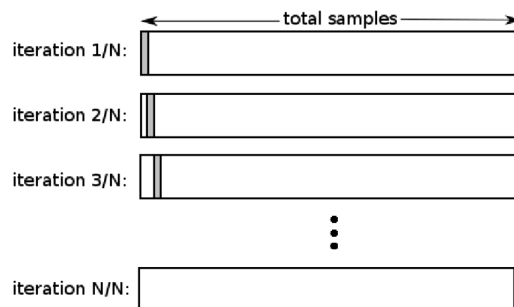
- 학습 데이터를 training data 와 validation(or test) data 로 K 번 나누어 사용
- K 번의 평균치를 성능 지표로 사용
- 학습 데이터의 일부를 고정적으로 training/test data 로 나눔에 따라 발생하는 bias 를 회피
- 모델의 파라미터 튜닝 등에 활용 가능



Training Data & Test Data

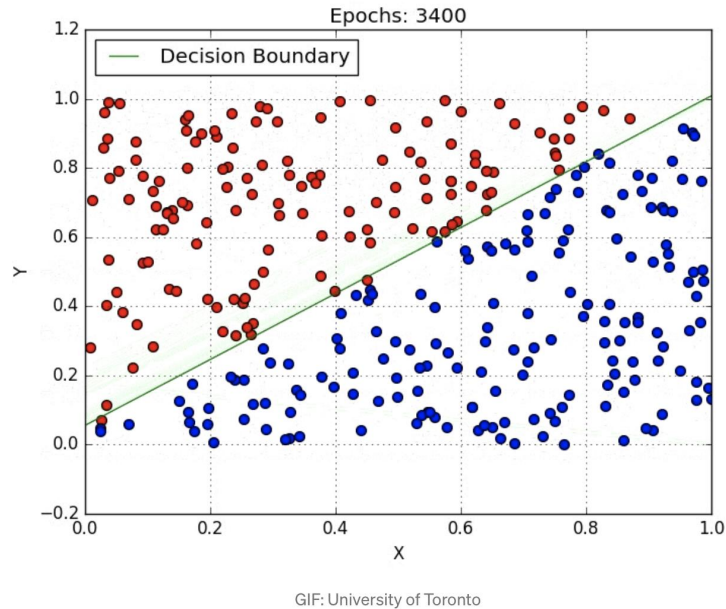
◆ Leave one out cross validation (LOOCV)

- Cross validation 에서 validation/test data 의 크기가 1인 경우 - 한 번에 한 개의 데이터만 validation/test data 로 사용



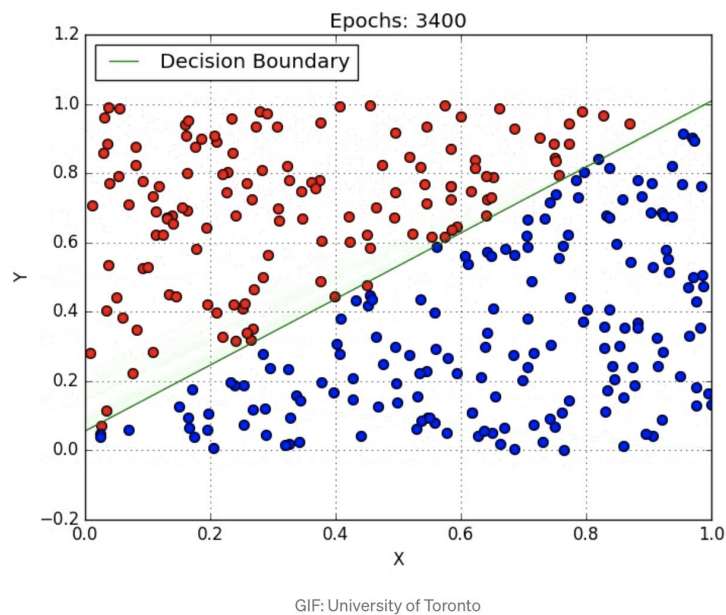
Logistic Regression for Classification

- ◆ 데이터를 구성하는 서로 다른 두 군집을 어떻게 구분 할 수 있을까?

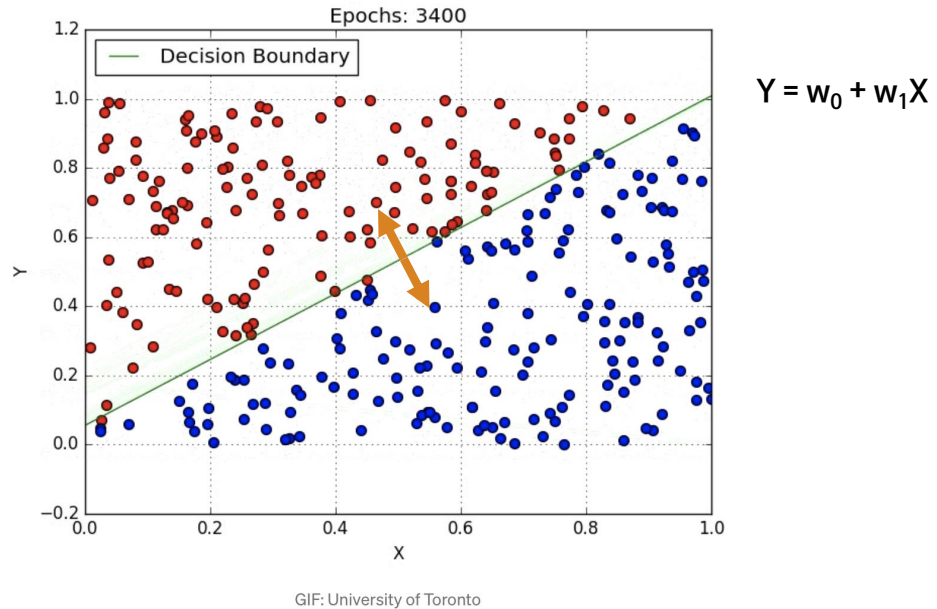


Logistic Regression for Classification

서로 다른 군집을 구분하는 선을 그어 보자



Logistic Regression for Classification



Logistic Regression for Classification

$$f(x) = Y = w_0 + w_1 X$$

$$Status = \begin{cases} 1 & \text{if } f(x) \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

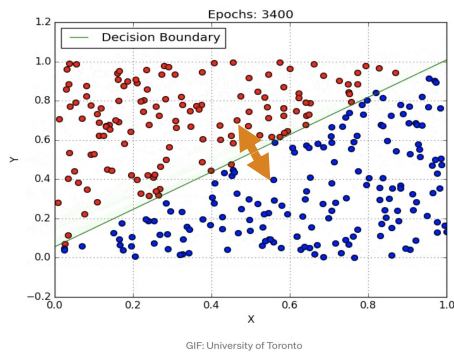
최적의 구분을 만드는 파라미터 w_0 와 w_1 을 어떻게 학습 할 것인가?

선을 긋는 문제니까, linear regression 으로 학습 가능하다!

Logistic Regression for Classification

◆ Linear regression 으로 한다 해도, 몇 가지 생각해 볼 점들

- $f(x) = 0$ 을 기준으로 분류한다 해도, 기준값(0)과의 차이를 어떻게 받아들여야 하는가?
- Target function 을 확률로 표현 할 수 있을까?



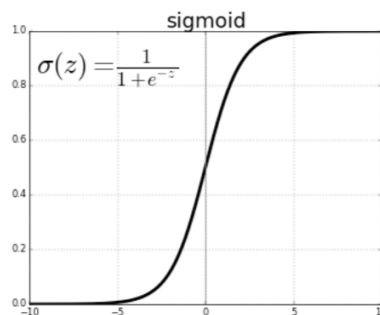
$$f(x) = Y = w_0 + w_1X$$

$$Status = \begin{cases} 1 & \text{if } f(x) \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

Logistic Regression for Classification

◆ Sigmoid(Logistic) function 을 사용하여 확률로 표현하자.

미분가능한 연속구간으로 변환
S형태로 닮았다고 하여 **sigmoid function**으로 호칭



<https://goo.gl/38SsHw>

Logistic Regression for Classification

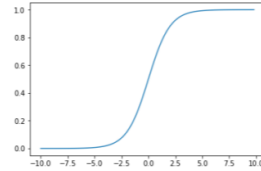
- ◆ 가설함수를 원래의 선형함수(z)에서 logistic function 을 이용한 함수(h)로 변환

$$h_{\theta}(x) = g(z) = \frac{1}{1 + e^{-z}}$$

where:

$$\begin{aligned} z &= w_0x_0 + w_1x_1 + \cdots + w_nx_n \\ &= \theta^T \mathbf{x} \end{aligned}$$

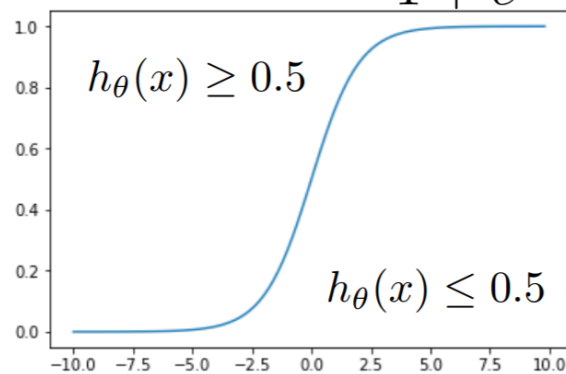
$$0 \leq h_{\theta}(x) \leq 1$$



Logistic Regression for Classification

- ◆ 확률값(h)에 의한 classification

$$h_{\theta}(x) = g(z) = \frac{1}{1 + e^{-z}}$$



Logistic Regression for Classification

◆ 파라미터 학습하기

$$h_{\theta}(x) = \frac{1}{1 + e^{-\theta^T \mathbf{x}}}$$

$$\theta^T \mathbf{x} = w_0 x_0 + w_1 x_1 + \cdots + w_n x_n$$

$$y = 0 \text{ or } 1$$

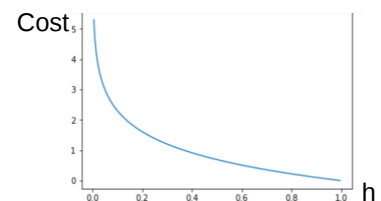
필요한 것:

Cost function 의 정의

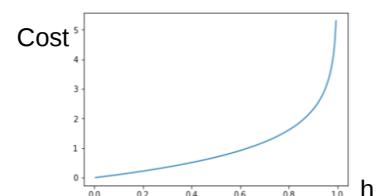
Cost function 의 gradient 계산

Logistic Regression for Classification

◆ 한 데이터 포인트에서의 cost 를 다음과 같이 정의



$$\text{Cost}(h_{\theta}(x), y) = \begin{cases} -\log(h_{\theta}(x)) & \text{if } y = 1 \\ -\log(1 - h_{\theta}(x)) & \text{if } y = 0 \end{cases}$$



Logistic Regression for Classification

◆ Cost function

$$\text{Cost}(h_{\theta}(x), y) = \begin{cases} -\log(h_{\theta}(x)) & \text{if } y = 1 \\ -\log(1 - h_{\theta}(x)) & \text{if } y = 0 \end{cases}$$

- $y = 1$ or 0 이므로, $\text{cost}(h_{\theta}(x), y) = -y\log(h_{\theta}(x)) - (1 - y)\log(1 - h_{\theta}(x))$

$$\begin{aligned} J(\theta) &= \frac{1}{m} \sum_{i=1}^m \text{Cost}(h_{\theta}(x^{(i)}), y^{(i)}) \\ &= -\frac{1}{m} \sum_{i=1}^m [y^{(i)} \log h_{\theta}(x^{(i)}) + (1 - y^{(i)}) \log(1 - h_{\theta}(x^{(i)}))] \\ \text{find } \theta, \text{ where } \min_{\theta} J(\theta) \quad h_{\theta}(x) &= \frac{1}{1 + e^{-\theta^T \mathbf{x}}} \end{aligned}$$

- ◆ 이 cost function J 를 최소화하는 파라미터 θ 를 찾기 위하여 편미분하면,

$$\frac{\partial}{\partial \theta_j} J(\theta) = \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)}) x_j^{(i)} \quad \text{이 gradient 를 이용하여 learning!}$$

2-2. Machine Learning 기초 - Classification Evaluation

Classifier Evaluation

◆ 기본 개념 - Confusion matrix (혼합 행렬)

- 실제 class 라벨과 예측 class 라벨의 일치 개수를 matrix 형태로 표현

		Prediction	
		Positive	Negative
Actual Class	Positive	True Positive	False Negative
	Negative	False Positive	True Negative

Metrics for Classification Performance

◆ Accuracy (정확도)

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

		Prediction	
		Positive	Negative
Actual Class	Positive	True Positive	False Negative
	Negative	False Positive	True Negative

◆ Error Rate (오차율)

$$Errorrate = \frac{FP + FN}{TP + TN + FP + FN} = (1 - Accuracy)$$

전체 데이터 대비 부정확한 예측의 비율

◆ Precision (정밀도)

$$Precision = \frac{TP}{TP + FP} \quad (PPV: Positive Predict Value)$$

참이라고 예측 한 것 중 진짜 참의 비율
(참이라는 예측이 얼마나 정확한지)

◆ Specificity (특이도)

$$Specificity = \frac{TN}{TN + FP} \quad (TNR: True Negative Rate)$$

실제 거짓 중 거짓으로 잘 예측한 비율

◆ Sensitivity (민감도)

(Recall)

$$Sensitivity = \frac{TP}{TP + FN} \quad (TPR: True Positive Rate)$$

실제 참인 것들 중 참으로 맞게 예측된 비율
(실제 참인 것들을 얼마나 많이 맞게 예측했는지)

Accuracy 외의 다른 Metric 존재 이유는?

- ◆ Class 분포가 bias 되어 있는 데이터가 있기 때문
- ◆ Class 별 정확도의 평가가 의미가 있는 경우 등

- ◆ 예시:

- 대학의 학사경고자 평균 비율 3%
- 하버드 입학 지원자의 합격률 2%
- 광고 이메일 수신자 중 2% 만 물건을 구매
- ...



통합 Metric

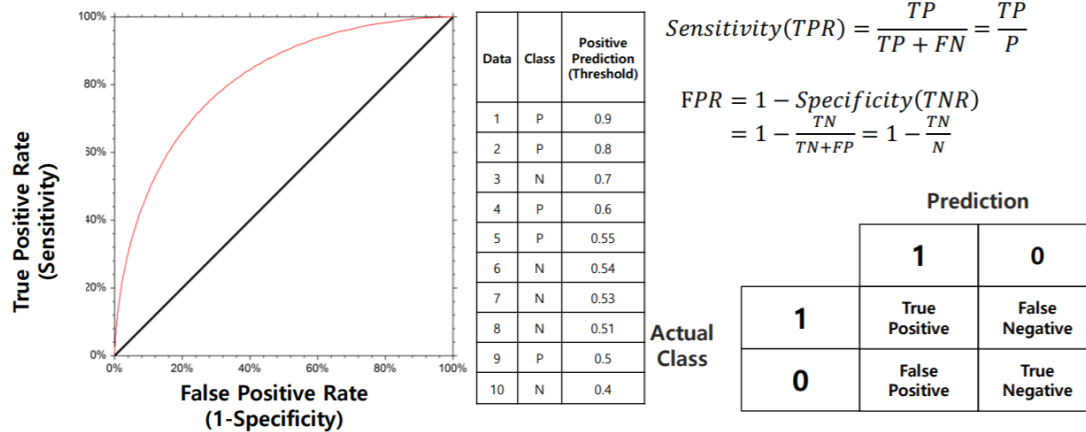
- ◆ F1 score (F-measure)
 - Precision 과 Recall 의 통합 지표 (harmonic mean)

$$F_1 = 2 \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$

		Prediction	
		1	0
Actual Class	1	True Positive	False Negative
	0	False Positive	True Negative

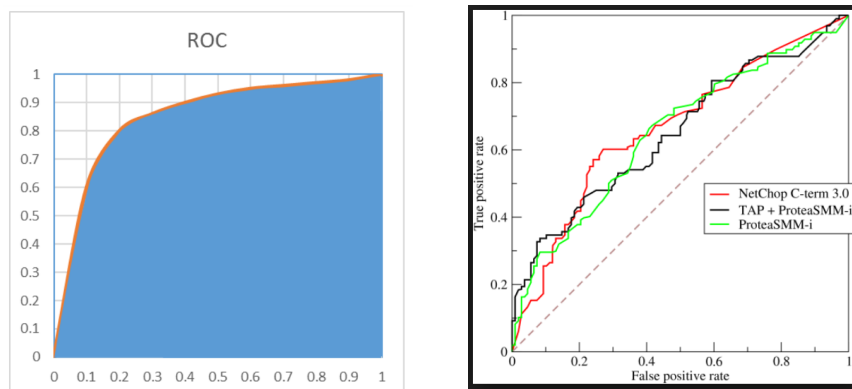
Receiver Operating Characteristics (ROC) Curve

- ◆ Classifier 의 threshold 를 조정하여, sensitivity ~ specificity 를 도식화
- ◆ Logistic regression, Naive Bayes 등과 같이 classification 기준을 조절 가능한 모델에 적용 가능



Summarizing ROC Curve

- ◆ Area under curve (AUC)
 - ROC curve 하단의 넓이
 - Curve 가 왼쪽 상단에 붙어 있을수록, 하단의 넓이가 넓을수록 높은 성능을 의미함

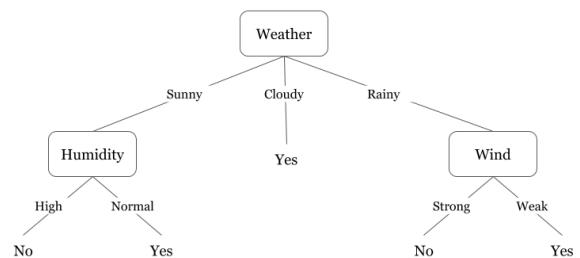


Machine Learning 기초 - Decision Tree & Ensemble Method

Decision Tree Classifier

- ◆ Data 를 잘 구분 할 수 있는 여러 decision 단계를 표현하는 tree 를 구성

Day	Weather	Temperature	Humidity	Wind	Play?
1	Sunny	Hot	High	Weak	No
2	Cloudy	Hot	High	Weak	Yes
3	Sunny	Mild	Normal	Strong	Yes
4	Cloudy	Mild	High	Strong	Yes
5	Rainy	Mild	High	Strong	No
6	Rainy	Cool	Normal	Strong	No
7	Rainy	Mild	High	Weak	Yes
8	Sunny	Hot	High	Strong	No
9	Cloudy	Hot	Normal	Weak	Yes
10	Rainy	Mild	High	Strong	No



Decision Tree 만들기

- ◆ 어떤 질문이 가장 효율적인 분류를 해 줄 것인가?
 - "효율적인 분류"의 정의에 따라 여러 가지 decision tree 구축 알고리즘이 존재
- ◆ 데이터를 이용하여 tree 의 splitting point 를 설정
- ◆ 많이 사용되는 기준: 어떤 질문이 답의 모호성을 줄여 줄 것인가?
 - 어떤 질문이 분류 이전 대비 분류 이후 데이터의 복잡도를 크게 낮추어 줄 것인가?
 - Information theory-based algorithms (ID3, C4.5 등)

Entropy

- ◆ Entropy
 - 무질서한 정도
 - 상태를 표현 할 수 있는 경우의 수
- ◆ Entropy 가 크다 =
 - 정보의 양이 많다.
 - 경우의 수가 많다.
 - 더 불확실하다.

Information and Information Entropy

◆ Information

$$I(X) = \log_2 \left(\frac{1}{P(X)} \right)$$

◆ Information entropy H(x)

$$\begin{aligned} H(X) &= E[I(X)] \\ &= E[-\log(P(X))] \\ &= -\sum P(x_i) \log(P(x_i)) \end{aligned}$$

Information Gain

$$Info(D) = -\sum_{i=1}^n p_i \log_2(p_i)$$

전체 데이터 D의 정보량

$$Info_A(D) = -\sum_{j=1}^v \frac{|D_j|}{D} * Info(D_j)$$

속성 A로 분류시 정보량

$$Gain(A) = Info(D) - Info_A(D)$$

A 속성의 정보 소득

데이터 전체의 정보량

Day	Weather	Temperature	Humidity	Wind	Play?
1	Sunny	Hot	High	Weak	No
2	Cloudy	Hot	High	Weak	Yes
3	Sunny	Mild	Normal	Strong	Yes
4	Cloudy	Mild	High	Strong	Yes
5	Rainy	Mild	High	Strong	No
6	Rainy	Cool	Normal	Strong	No
7	Rainy	Mild	High	Weak	Yes
8	Sunny	Hot	High	Strong	No
9	Cloudy	Hot	Normal	Weak	Yes
10	Rainy	Mild	High	Strong	No

$$\begin{aligned}
 h(D) &= - \sum_{i=1}^m p_i \log_2 p_i \\
 &= \left(-\frac{5}{10} \log_2 \frac{5}{10} \right) + \left(-\frac{5}{10} \log_2 \frac{5}{10} \right) \\
 &= 1
 \end{aligned}$$

“Yes” class 에 대한 부분
“No” class 에 대한 부분

Information Gain 기반 Decision Tree 의 구축

- ◆ 데이터 분류에 사용 가능한 각 변수들 중, 분류 이전 대비 분류 이후 정보량을 보다 많이 줄여 줄 수 있는 변수가 tree 의 상단에 오는 것이 유리
 - 최적의 decision tree 구축은 NP-complete problem
 - Greedy approach 등의 heuristic 을 일반적으로 사용

Day	Weather	Temperature	Humidity	Wind	Play?
1	Sunny	Hot	High	Weak	No
2	Cloudy	Hot	High	Weak	Yes
3	Sunny	Mild	Normal	Strong	Yes
4	Cloudy	Mild	High	Strong	Yes
5	Rainy	Mild	High	Strong	No
6	Rainy	Cool	Normal	Strong	No
7	Rainy	Mild	High	Weak	Yes
8	Sunny	Hot	High	Strong	No
9	Cloudy	Hot	Normal	Weak	Yes
10	Rainy	Mild	High	Strong	No

전체 데이터에서 Wind 로 분류하는 경우의 information gain

$$Gain(Wind) = Info(D) - Info_{Wind}(D)$$

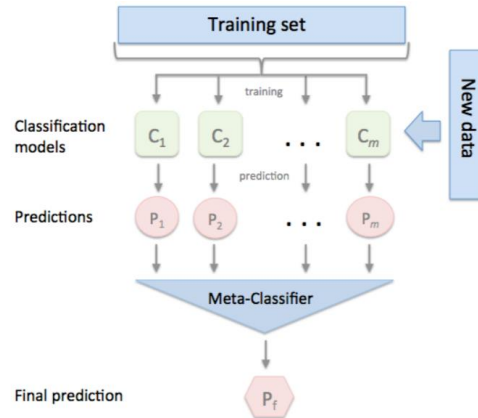
$$Info(D) = 1$$

$$Info_{Wind}(D)$$

$$= -\frac{4}{10} Info(D_{Wind=Weak}) - \frac{6}{10} Info(D_{Wind=Strong})$$

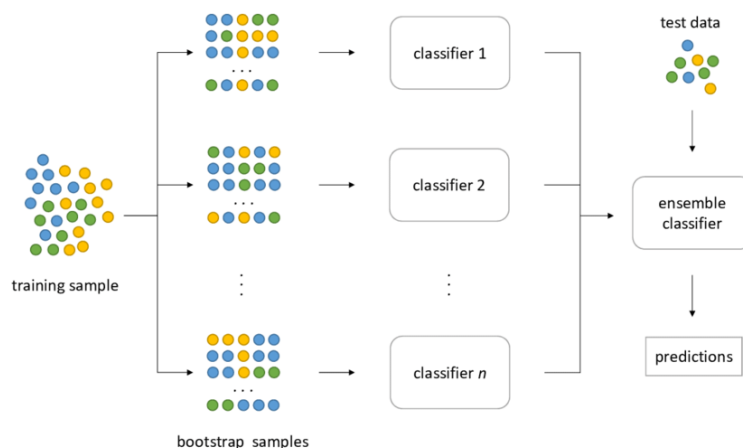
Ensemble Method

- ◆ 한 모델의 결과를 사용하지 않고, 여러 모델들의 결과들을 조합하여 class 예측
- ◆ 이유: 현실적으로 optimal classifier 를 찾기 힘들고, 대부분의 경우 local minima classifier 를 찾게 됨
 - 제한된 학습 데이터
 - 제한된 computing resource
- ◆ 여러 classifier 들의 결과를 조합하자.
 - 투표, averaging, ...



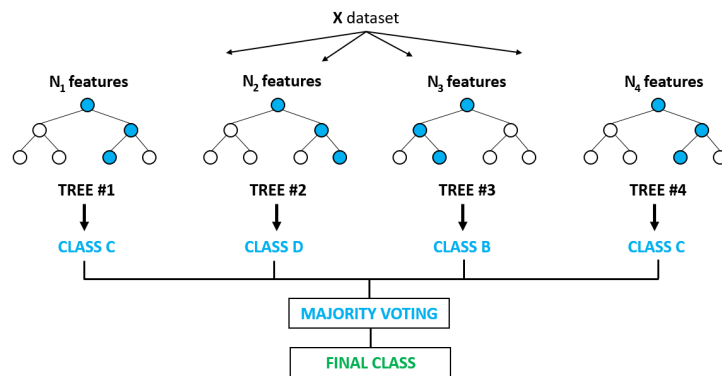
여러 Classifier Instance 학습을 위한 Bootstrapping & Bagging

- ◆ Bootstrapping: 학습 데이터로부터 임의의 subset 데이터를 N 개 추출
- ◆ Bagging: Bootstrap 에서 나온 subset 데이터로 N 개의 모델을 학습



Random Forest

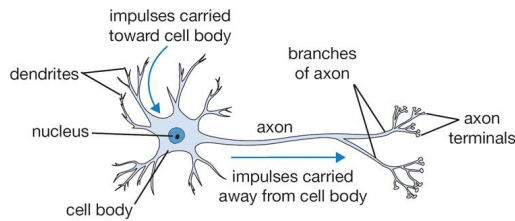
- ◆ Random forest: Decision tree + Bagging approach
- ◆ 여러 개의 tree (forest)
- ◆ 간단하면서도 높은 성능을 보여주는 대표적 ensemble 모델



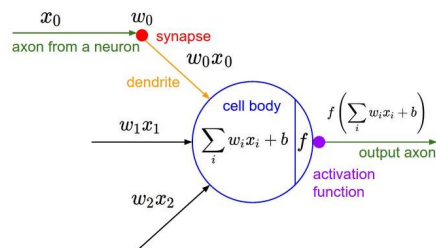
Machine Learning 기초 - Deep Learning

THE Building Unit of Deep Learning (and Artificial Neural Networks): “Neuron”

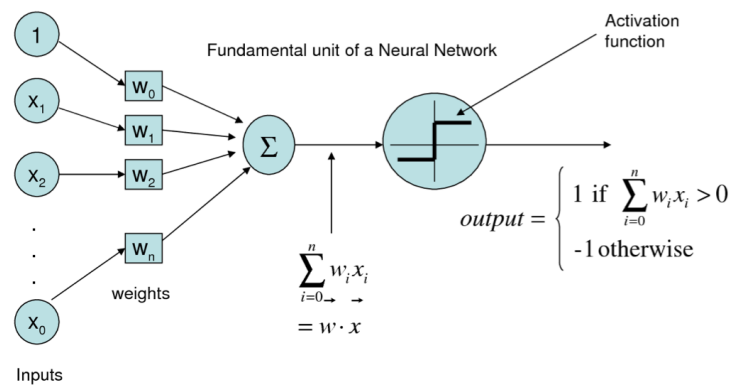
Neuron



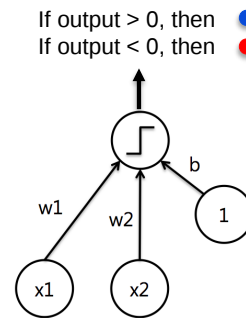
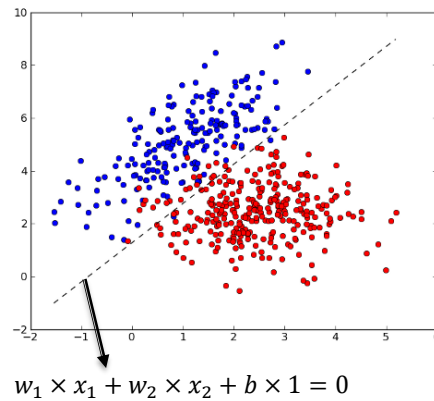
“Artificial Neuron”



“Artificial Neuron” – Perceptron

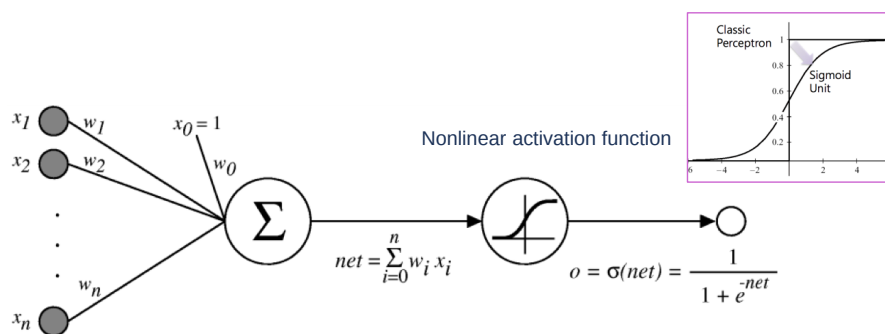


How Perceptron Works

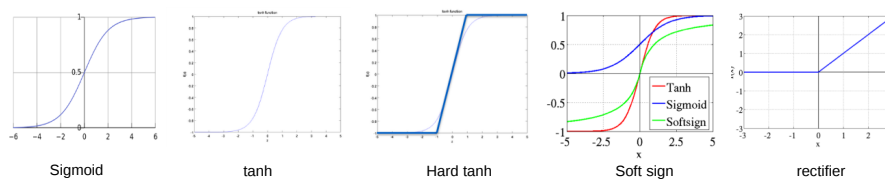


A linear classifier

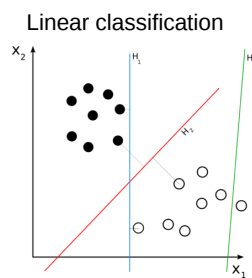
Going Nonlinear



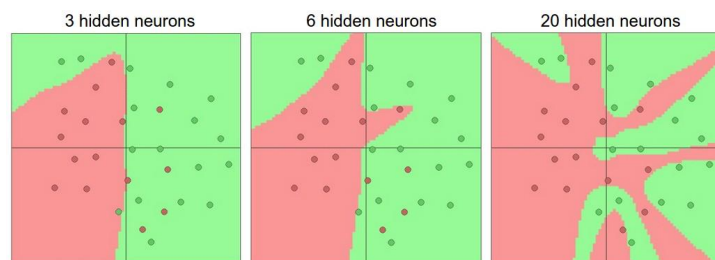
Various choices



Combination of Neurons for More Complex Space Separation



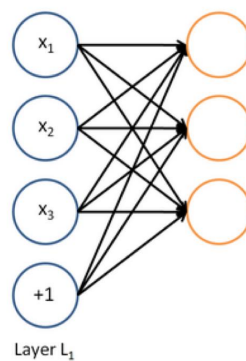
We need more than linear classification here.



Figures from Stanford CS231n github

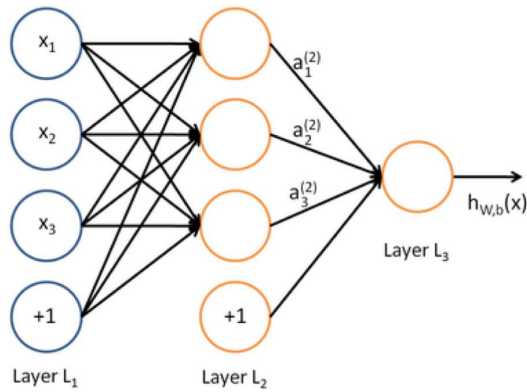
Neural Network

- ◆ Combination of multiple logistic regression



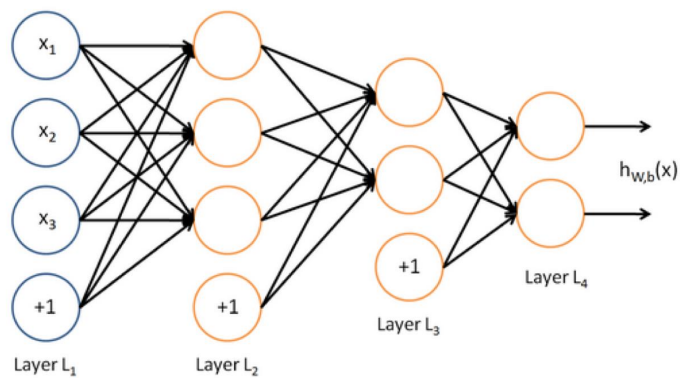
Neural Network

- ◆ Output can be summarized into another logistic regression.

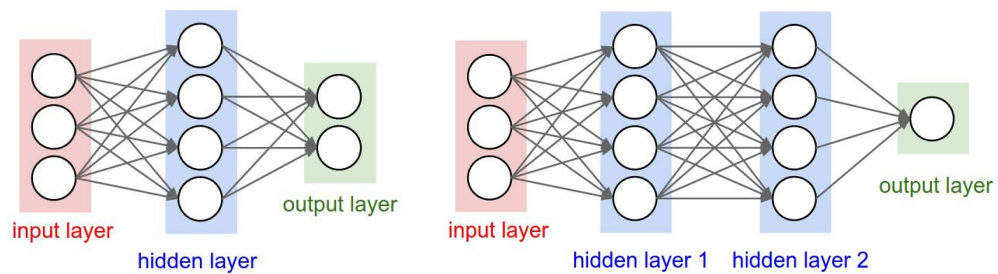


Multilayer Neural Network

- ◆ Why not going further for even more complex problems?

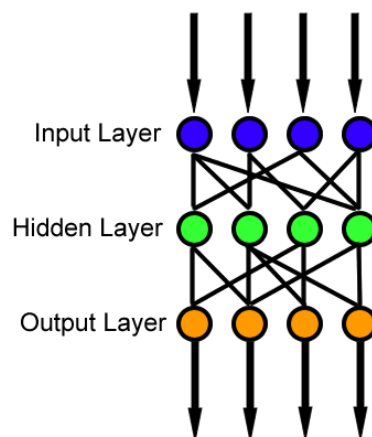


Multilayer Neural Network

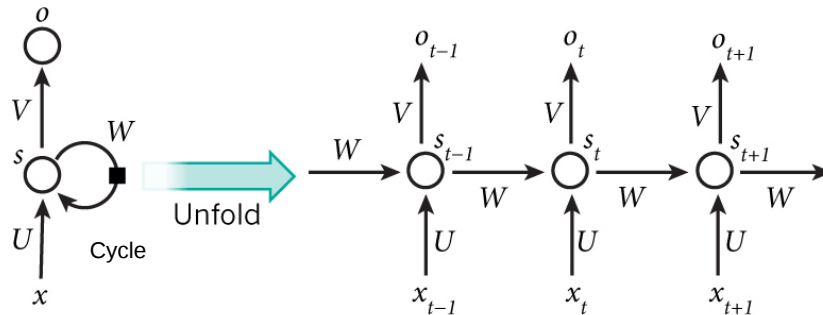


Feedforward Neural Network

- ◆ Connections between units do not form a cycle.
 - Information always moves one direction.



Recurrent Neural Network



By unfolding, it can utilize sequential information.

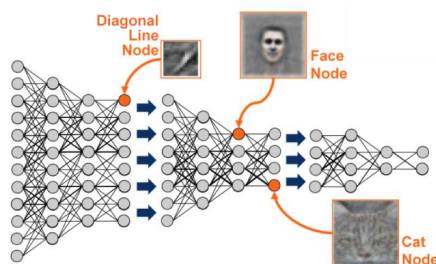
- Sentence
- Voice
- Video
- Other continuous signals (where a current input depends on previous status of input)

Figure from wildml.com

Deep Learning?

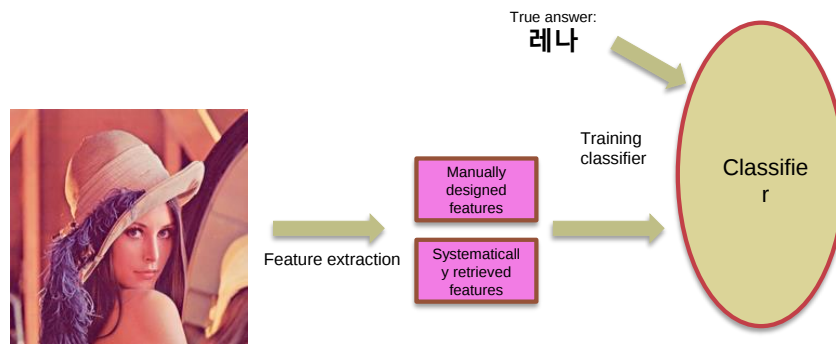
♦ From Wikipedia:

“Deep learning is a branch of machine learning based on a set of algorithms that attempt to model high-level abstractions in data by using multiple processing layers, with complex structures or otherwise, composed of multiple non-linear transformations.”



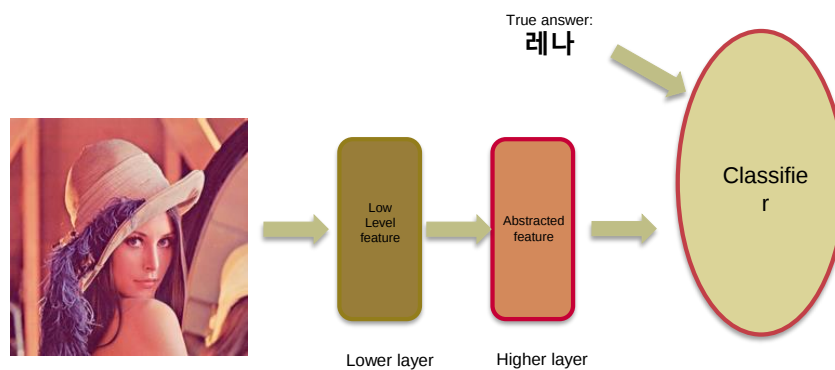
Conventional Pattern Recognition

- ◆ I. Feature extraction
- ◆ II. Classifier training



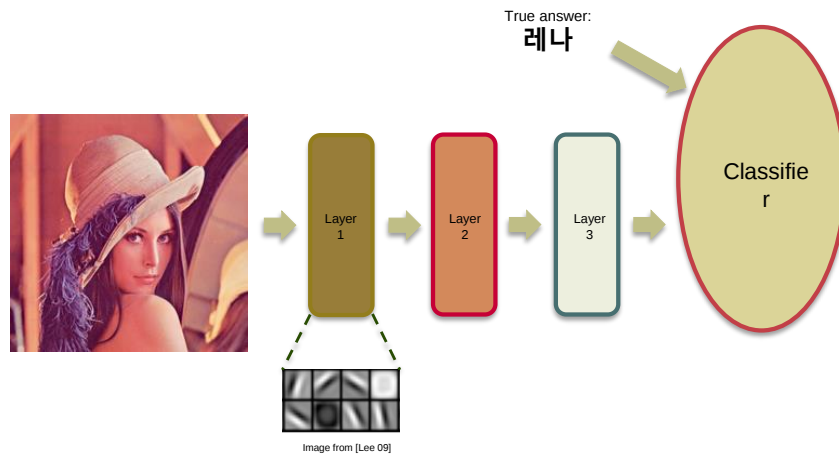
Deep Architecture

- ◆ Features (including abstracted features) are learned during the model training process.
 - Various levels of abstracted features by multi-layer representation



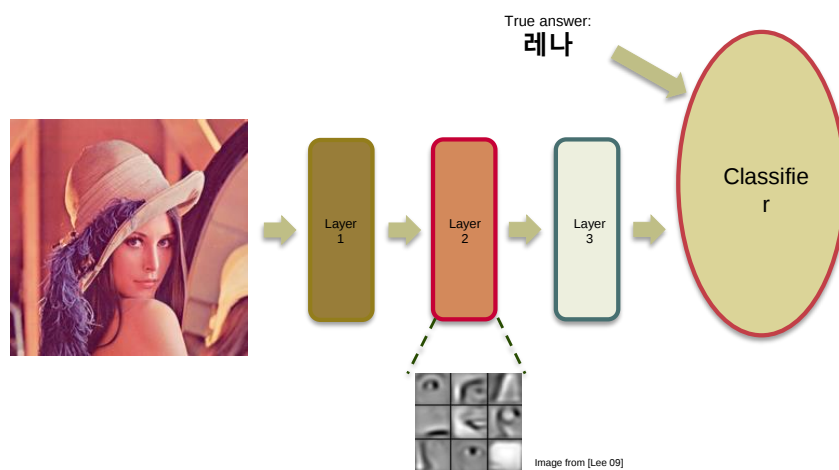
Deep Architecture

- ◆ Learning multiple levels of features through layerwise models



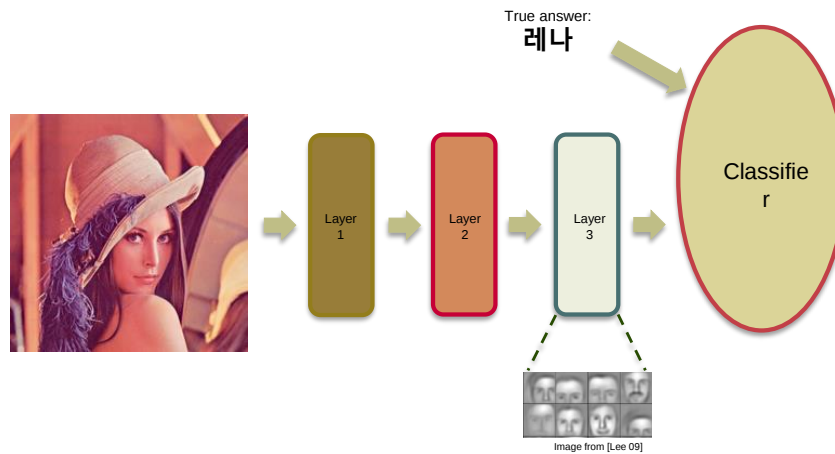
Deep Architecture

- ◆ Learning multiple levels of features through layerwise models



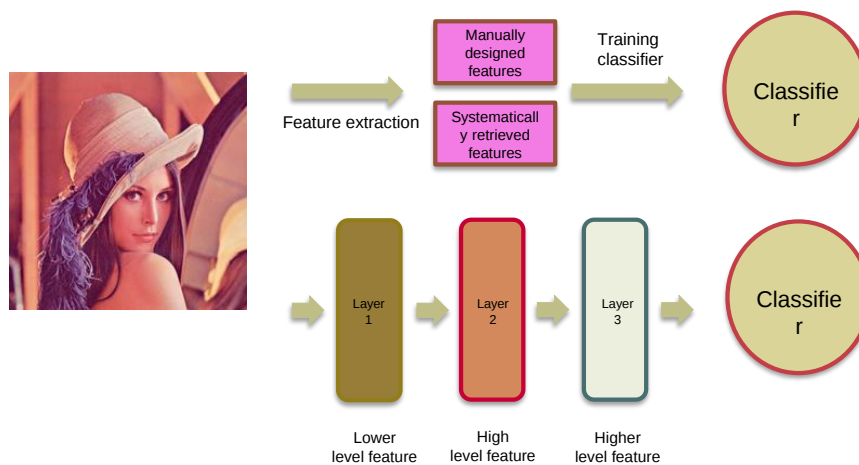
Deep Architecture

- ◆ Learning multiple levels of features through layerwise models



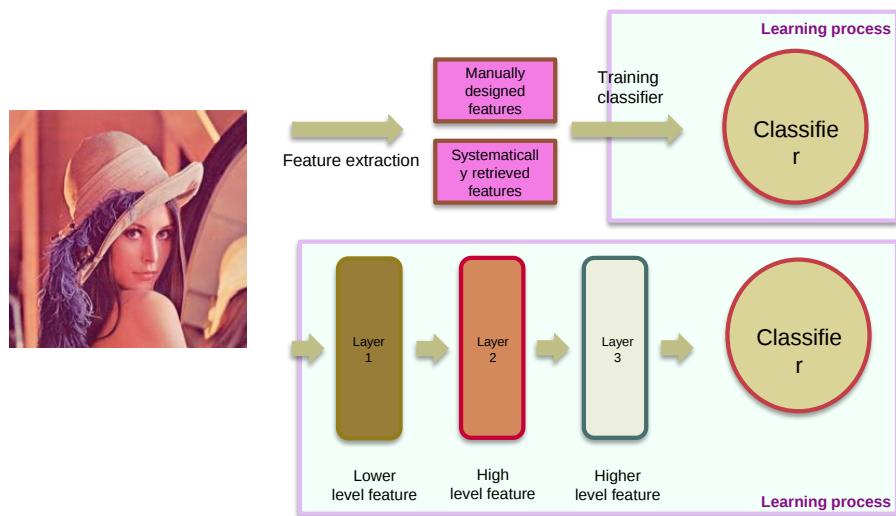
Show It Be Deep?

Can be more efficient for “simple”, well-structured problems.



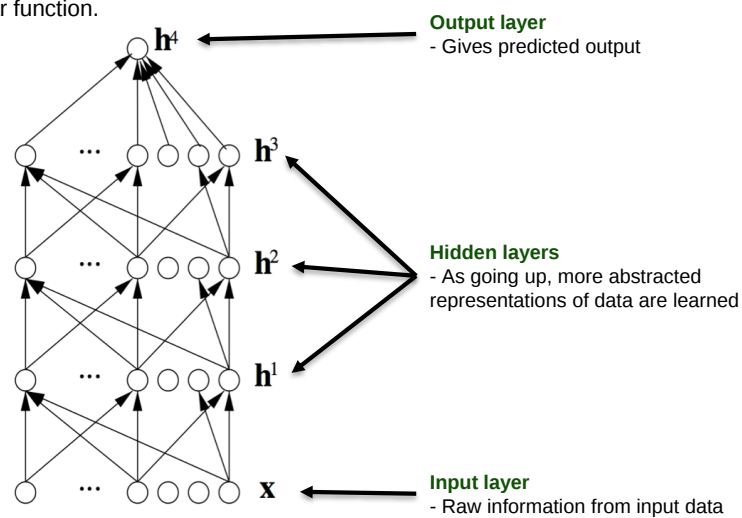
With more representational power, it can be appropriate for more complex problems accompanying complex, hierarchical concepts.

Difference



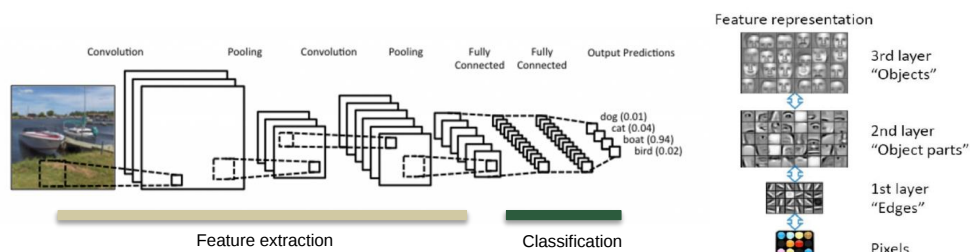
A Deep Architecture

With multilayer structure of nonlinear modules, it can approximate highly complex nonlinear function.



Convolutional Neural Network

- ◆ Handwritten digit recognition (LeCun 98)
- ◆ A neural network architecture that utilizes the characteristics of images - locality.
- ◆ Convolution: Retrieving features from local regions
- ◆ Pooling: Dimension reduction & combination of local features

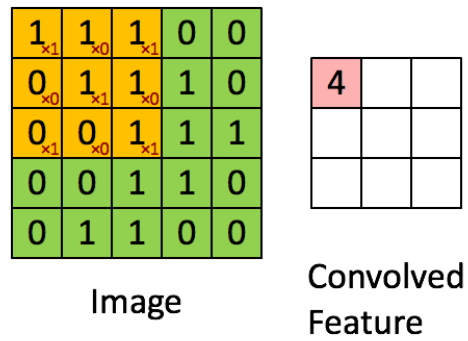


Convolution Layer



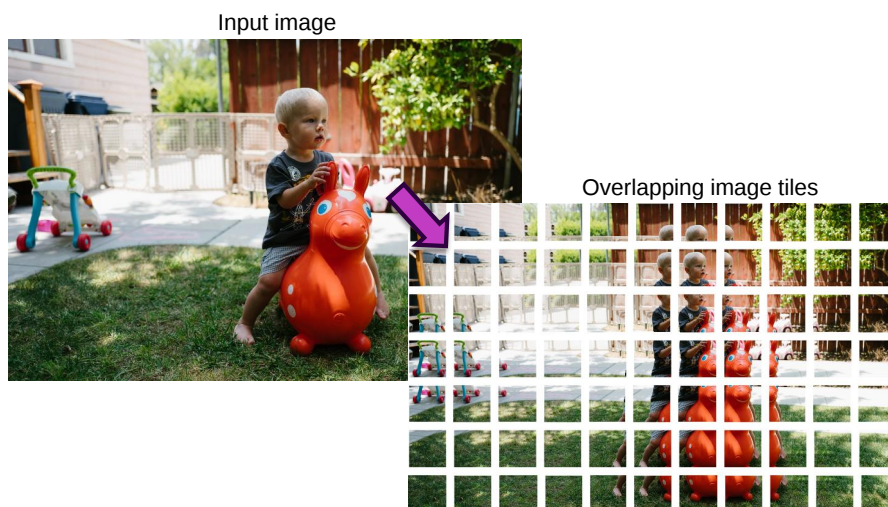
- ◆ Locality of image
 - Each pixel is related to only relatively small neighborhood region.
 - Local connection
- ◆ Stationary characteristics of image
 - Certain characteristics are consistent across regions.
 - Shared weights

Convolution



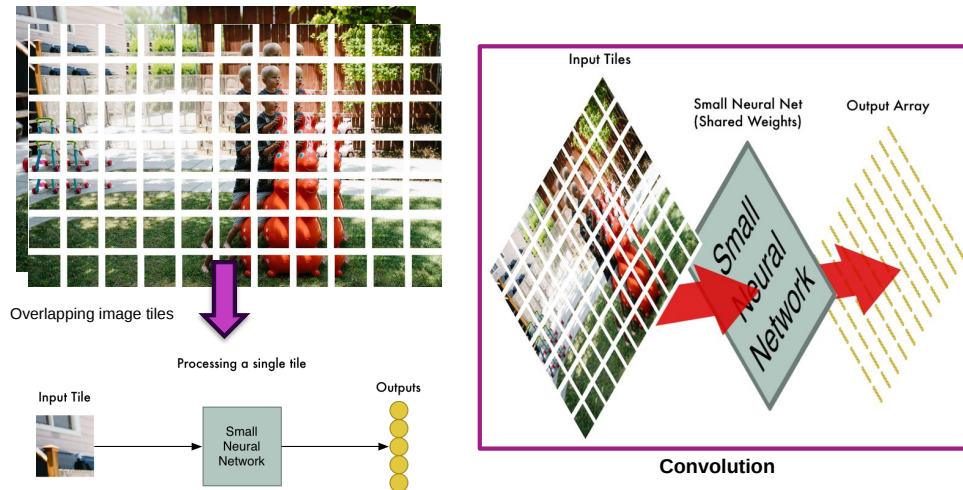
http://deeplearning.stanford.edu/wiki/index.php/Feature_extraction_using_convolution

Convolutional Neural Network



Images from <https://medium.com/@ageitgey/>

Convolutional Neural Network



Images from <https://medium.com/@ageitgey/>

CNN - Activation Layer

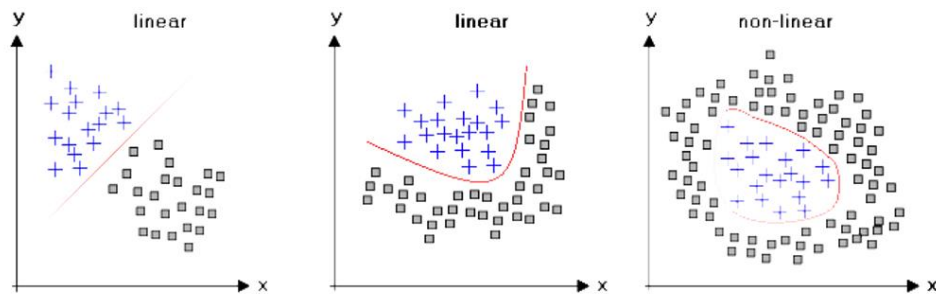
Input -> Conv -> ReLU -> Conv -> ReLU -> Pool -> ReLU -> Conv -> ReLU -> Pool -> Fully Connected

Activation Layer

- Conv Layer 바로 다음에 사용하는 nonlinear layer
(Conv layer의 output을 activation map이라고 부르는 이유!)

CNN - Activation Layer

Activation layer 에 nonlinear function 을 사용함으로써 데이터 공간상의 복잡한 패턴을 보다 용이하게 표현 가능



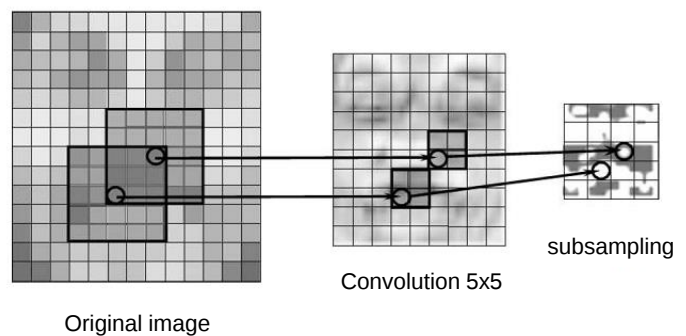
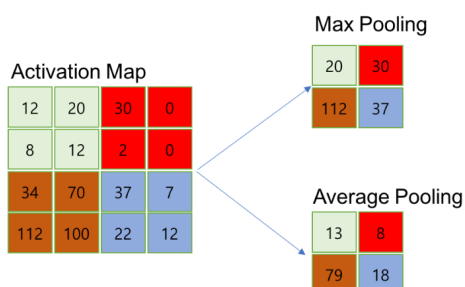
universal function approximator.

http://www.statistics4u.com/fundstat_eng/cc_linvsnonlin.html

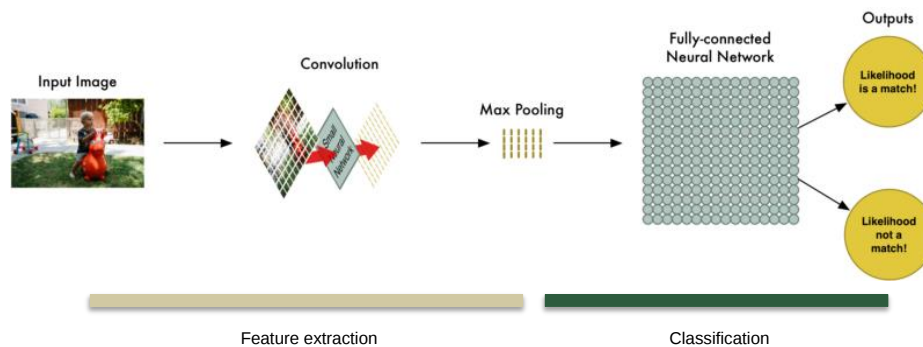
CNN - Pooling Layer

Pooling layer

- Too many pixels and features
- Reduce dimension by subsampling
- Prevent overfitting



Convolutional Neural Network



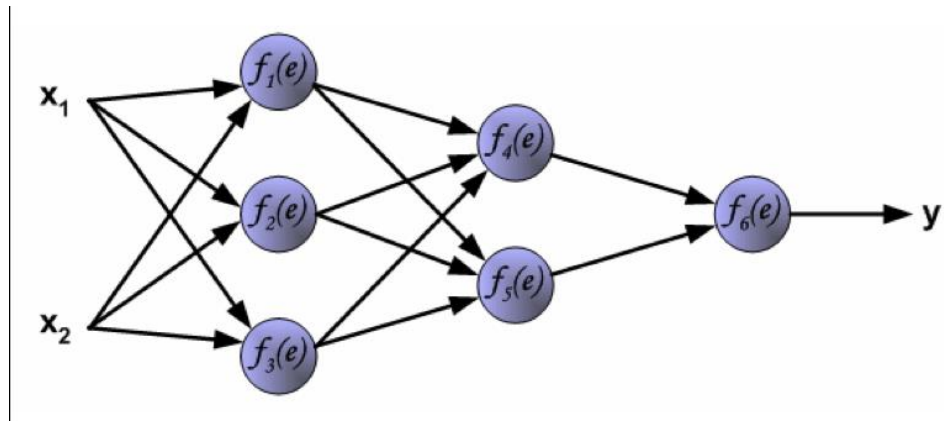
Images from <https://medium.com/@ageitgey/>

Today's Popularity of Deep Learning

- ◆ Recall that neural networks and their base theories have been around since 60's.
- ◆ Difficulty of using deep neural networks
 - Lack of dataset large enough to train such complex models
 - Lack of computing power to train complex models with large dataset
 - Lack of efficient learning algorithms
- ◆ Since 2000's
 - Large datasets become available
 - Internet + mobile) Facebook, Google, Instagram, Twitter, ...
 - IoT) Huge data collection sources
 - Etc
 - Huge computing resources become available
 - HPC, multicore architecture, GPUs, ...
 - DBN and many new efficient algorithms

Training Deep Neural Networks - Backpropagation Algorithm

- ◆ Example) 3 layers, 2 inputs, 1 output

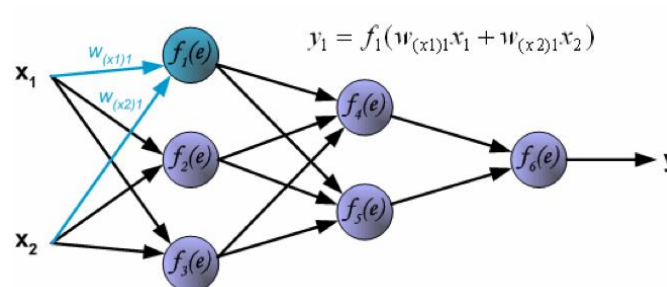


Training a neural network model:

Given a training data $\{(D_i, z_i)\}$, finding the weight values in the network that minimize the difference between the desired output and the output from the network.

Backpropagation Concept Illustration

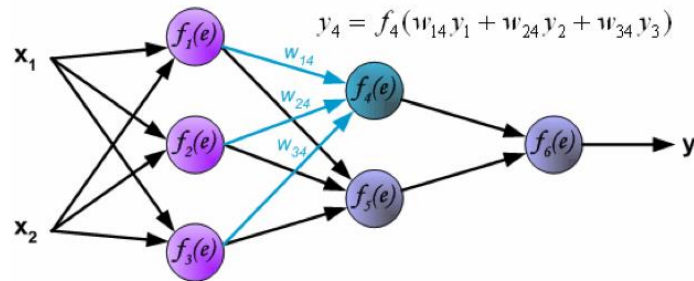
- ◆ Training starts through the input layer:



- ◆ The same happens for y_2 and y_3 .

Backpropagation Concept Illustration (cont'd)

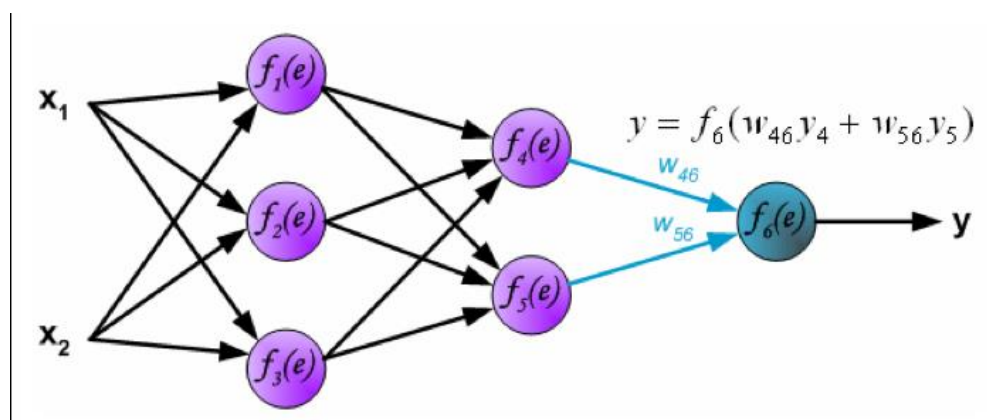
- ♦ Propagation of signals go forward through the hidden layer:



- ♦ The same happens for y_5 .

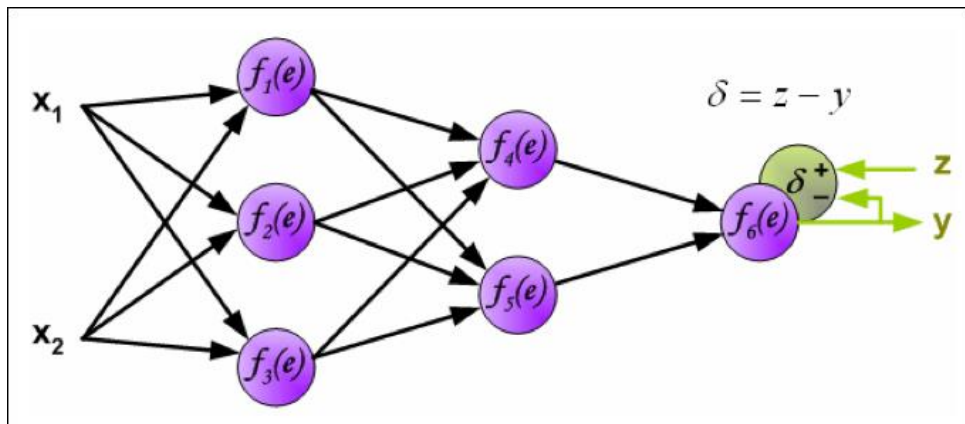
Backpropagation Concept Illustration (cont'd)

- ♦ Propagation of signals through the output layer:



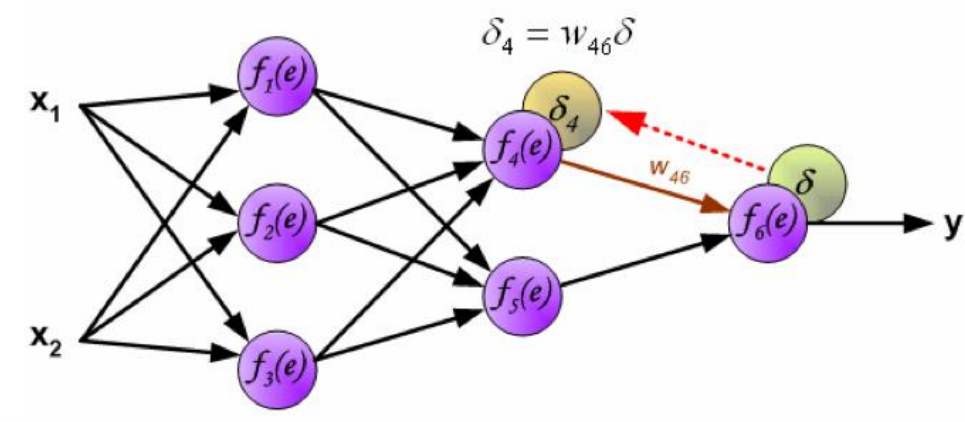
Backpropagation Concept Illustration (cont'd)

- ◆ Error from the output layer neuron:



Backpropagation Concept Illustration (cont'd)

- ◆ Propagate error back to all neurons.

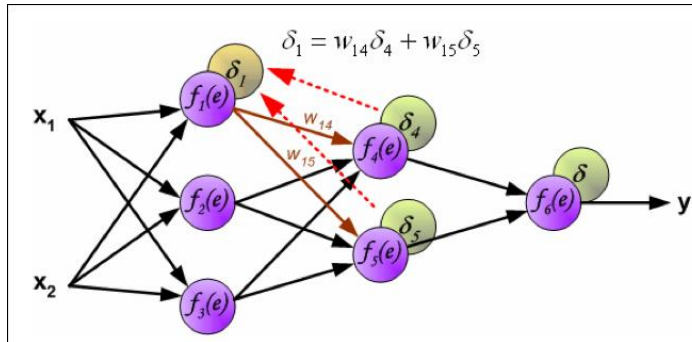


Backpropagation:

Propagating error derivatives backwards, updating weights.

Backpropagation Concept Illustration (cont'd)

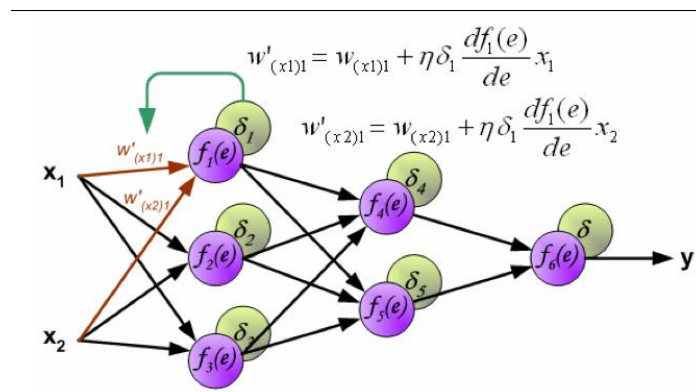
- ◆ If propagated errors come from multiple neurons, they are added up:



- ◆ The same happens for neuron-2 and neuron-3.

Backpropagation Concept Illustration (cont'd)

- ◆ Weight updating starts:

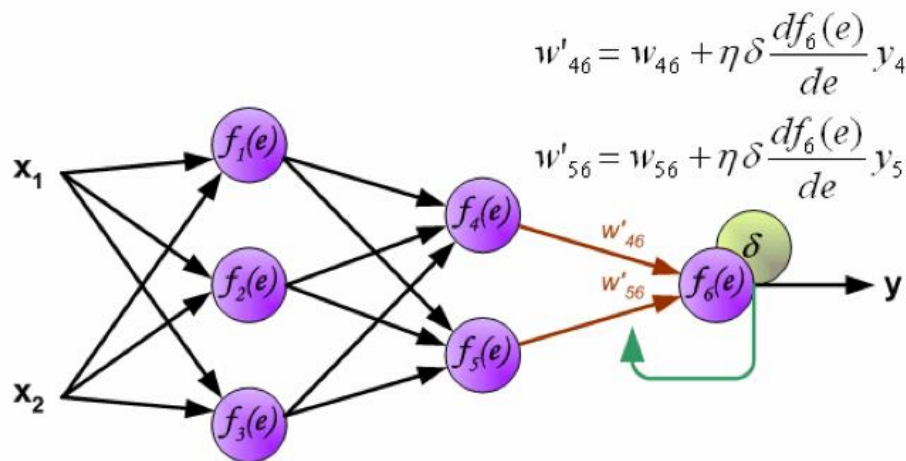


- ◆ The same happens for all neurons.

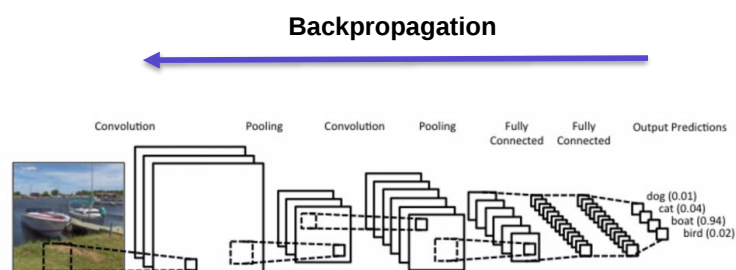
각 layer 의 parameter 에 대한 gradient 는 chain rule derivative 를 이용하여 효과적으로 계산 가능

Backpropagation Illustration (cont'd)

- ◆ Weight updating all the way to the output neuron:

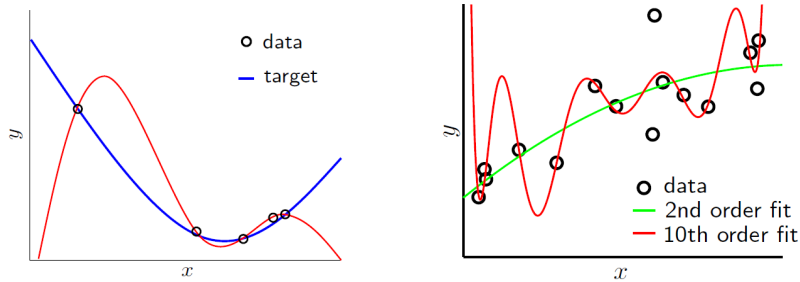


CNN: Backpropagation



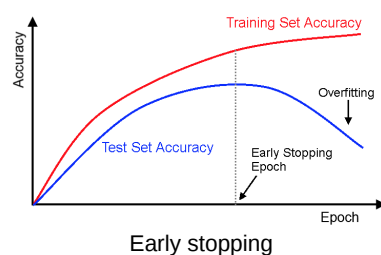
A Key Point in Deep Learning

- ◆ Deep neural network is a very complex model (with many parameters to be optimized).
- ◆ Can easily lead to **overfitting**.



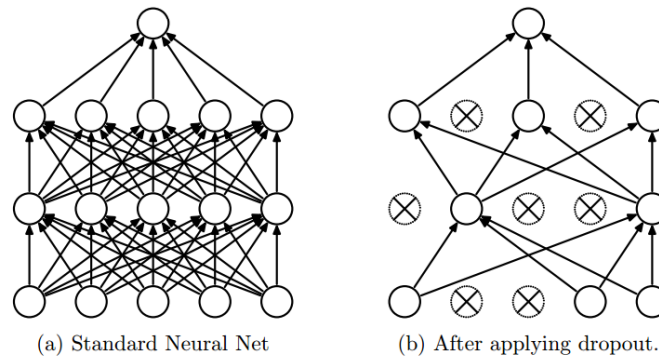
Restricting Model Capacity

- ◆ Restricting the model complexity
 - Limiting the number of layers
 - Limiting the number of neurons in each layer
 - Sharing weights
- ◆ Other learning techniques to avoid overfitting



Shrinking Model on the Fly

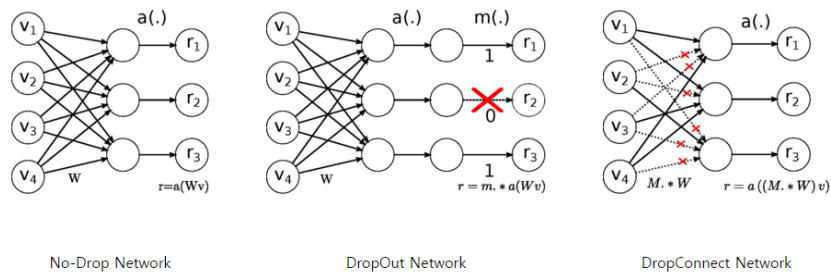
♦ Dropout



Srivastava, Nitish, et al. "Dropout: A simple way to prevent neural networks from overfitting." *The Journal of Machine Learning Research* 15.1 (2014): 1929-1958.

Shrinking Model on the Fly

♦ DropConnect



Li Wan, Matthew Zeiler, Sixin Zhang, Yann LeCun, Rob Fergus
"Regularization of neural networks using DropConnect." *ICML*, 2013.

Summary on Deep Learning

- ◆ Deep learning works better than simple systems because of its hierarchical abstraction of data information.
 - Can be appropriate for problems implying various concepts that can be hierarchically abstracted.
- ◆ Essentials of deep learning
 - Large-scale training data
 - Enough computing resources
 - Efficient training algorithms

Considerations in Deep Learning for Biology/Biomedicine

- ◆ Correct understanding of a target problem
 - What is a data instance?
 - How many classes?
- ◆ Selecting a proper learning model
 - Do not think the deep learning is always the best solution! (Because it is not)
- ◆ Checking data availability / feasibility of data collection
 - Positive and negative control
 - Class imbalance
- ◆ Designing proper learning architecture and strategies
 - Selecting appropriate models
 - Appropriate architecture design
 - Effective learning strategies
 - (These are usually THE KNOW-HOW / SPECIALTY of deep learning specialists.)

Thank you!