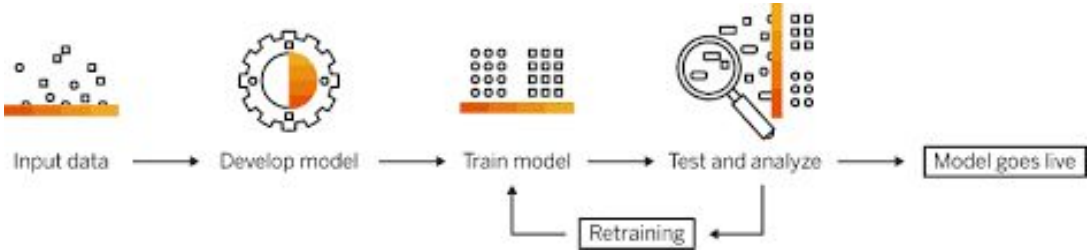
면접 준비

데이터 사이언티스트

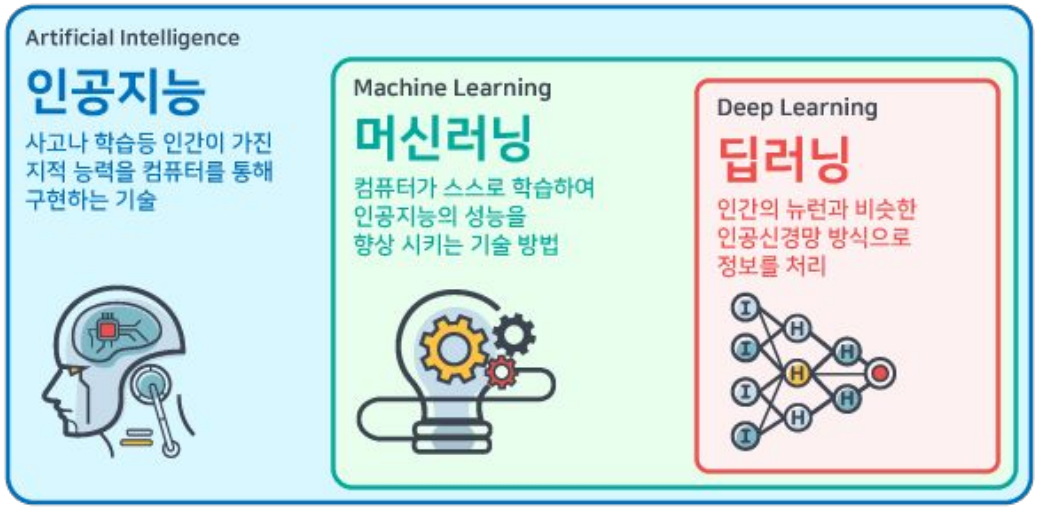
출처

- https://www.simplilearn.com/tutorials/data-science-tutorial/data-science-interview-questions?source=sl_frs_nav_playlist_video_clicked#basic_data_science_interview_questions
- https://www.ubuntupit.com/frequently-asked-machine-learning-interview-questions-and-answers/
-

0. Machine Learning

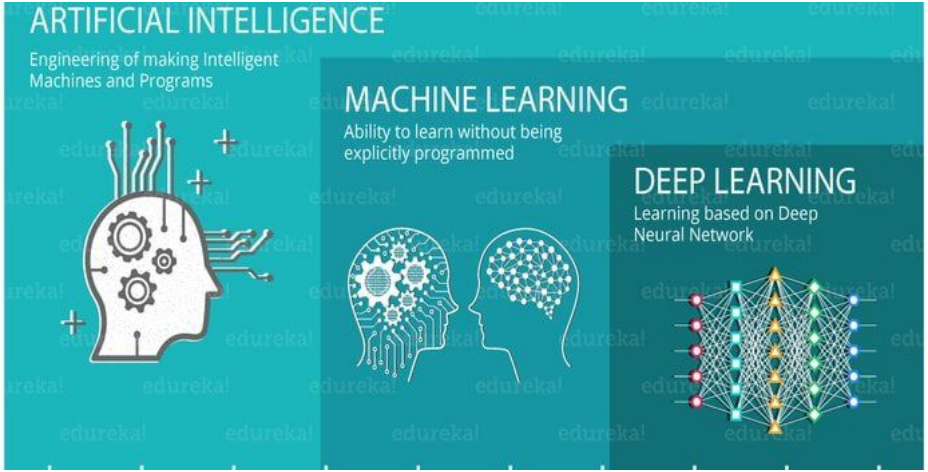


- 명시적으로 컴퓨터를 프로그래밍하는 대신, 데이터로 학습하고 개선하도록 훈련에 중점
- 대규모 데이터 세트에서 패턴과 상관관계를 찾고 분석을 토대로 최적의 의사결정과 예측을 수행하도록 훈련



0-1. AI vs Machine Learning vs Deep Learning

- Artificial intelligence : **인간의 뇌를 모방하는 컴퓨터로 구현**하는 것을 목표
- 딥 러닝 : 머신 러닝의 하위 개념, 머신 러닝에 해당하며 비슷한 방식으로 작동
- 차이점 : 머신 러닝 모델은 점진적으로 향상 중 약간의 안내가 필요 그러나 딥 러닝 모델은 **신경망**을 통해 예측의 정확성 여부를 스스로 판단



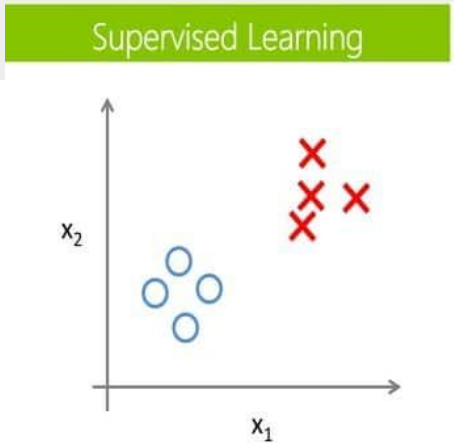
0-2. Machine Learning vs Data Mining

- 둘 모두 유사점이 굉장히 많다.
- 데이터 마이닝 : **데이터에서의 패턴을 추출하는 것**을 목적
- 머신러닝 : 데이터를 이용하여 **자동적으로 학습하는 기계를 만드는 것**을 목적

1. Supervised vs Unsupervised

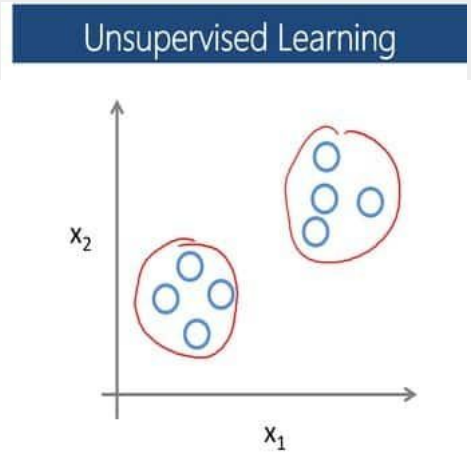
Supervised Learning

- 라벨링 데이터 사용
- decision tree
- logistic regression
- support vector machine



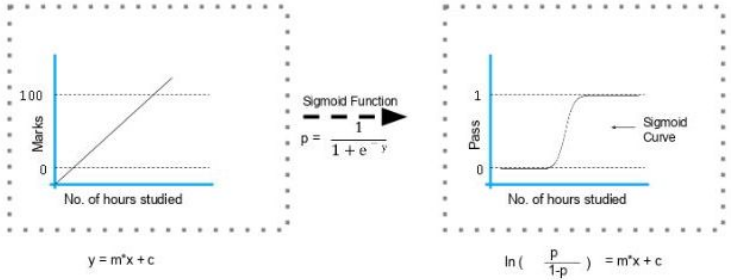
Unsupervised Learning

- 라벨링 데이터 사용 **X**
- k-means clustering
- hierarchical clustering
- apriori algorithm



2. Logistic regression

- 독립 변수의 선형 조합으로 종속 변수를 예측
  - 종속 변수 : dependent variable, label
  - 독립 변수 : independent variable, feature
- Logistic function ~ Sigmoid function
- Classification problem or 확률 예측
- Linear Regression의 목표는 범위가 정해지지 않은 종속 변수와 독립 변수 사이의 선형 관계를 측정하는 것이지만 Logistic Regression은 Linear Regression을 이용하여 확률을 예측



2-1. Linear regression vs Logistic regression

Linear Regression	Logistic Regression
- Regression problems - Continuous 데이터를 출력 - 종속 변수를 추정 - 직선 형태	- Classification problems 또는 확률값 예측 문제 - Categorical 데이터를 출력 - 종속 변수의 가능성을 계산 - Sigmoid curve

Q. 나이, 성별, 혈중 콜레스테롤 수치라는 3가지 위험 요인을 바탕으로 심장병으로 인한 사망 확률을 예측하고자 할때 적합한 모델은 무엇인가?

A. Logistic Regression

2-2. Sigmoid Function

- 시그모이드 함수는 S자형 곡선 또는 시그모이드 곡선을 갖는 수학 함수이다.

  - 로지스틱 함수

    $$f(x) = \frac{1}{1 + e^{-x}}$$

  - 쌍곡탄젠트 (위의 로지스틱 함수를 평행이동하고 상수를 곱한 것과 같음)

    $$f(x) = \tanh x = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

  - 아크탄젠트 함수

    $$f(x) = \arctan x$$

  - 오차 함수

    $$f(x) = \operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2}\, dt$$

  - 일부 대수함수, 예를 들어:

    $$f(x) = \frac{x}{\sqrt{1 + x^2}}$$

https://ko.wikipedia.org/wiki/%EC%8B%9C%EA%B7%B8%EB%AA%A8%EC%9D%B4%EB%93%9C_%ED%95%A8%EC%88%98

2-3. Logistic Function의 미분

$$
\begin{aligned}
\frac{d}{dx} sigmoid(x) &= \frac{d}{dx}(1+e^{-x})^{-1} \\
&= (-1)\frac{1}{(1+e^{-x})^2}\frac{d}{dx}(1+e^{-x}) \\
&= (-1)\frac{1}{(1+e^{-x})^2}(0+e^{-x})\frac{d}{dx}(-x) \\
&= (-1)\frac{1}{(1+e^{-x})^2}e^{-x}(-1) \\
&= \frac{e^{-x}}{(1+e^{-x})^2} \\
&= \frac{1+e^{-x}-1}{(1+e^{-x})^2} \\
&= \frac{(1+e^{-x})}{(1+e^{-x})^2} - \frac{1}{(1+e^{-x})^2} \\
&= \frac{1}{1+e^{-x}} - \frac{1}{(1+e^{-x})^2} \\
&= \frac{1}{1+e^{-x}}(1-\frac{1}{1+e^{-x}}) \\
&= sigmoid(x)(1-sigmoid(x))
\end{aligned}
$$

$$\frac{d}{dx} sigmoid(x) = sigmoid(x)(1-sigmoid(x))$$

http://taewan.kim/post/sigmoid_diff/

3. Decision Tree

- Algorithm
  1. 분기 전 데이터를 입력으로 사용
  2. 분기 전 데이터의 Entropy 계산, 분기 특징 후보들에 대한 Entropy 계산
  3. 분기 특징 후보들의 Information Gain 계산
  4. 가장 높은 Information Gain 값을 가지는 분기 특징을 선택
  5. 사전에 정의한 멈추는 조건이 될때까지 위 과정을 반복
- **Entropy : 데이터가 얼마나 균일하게 분류되었는지 알려주는 척도**, 즉 작을수록 잘 분류된 상태
- **Information Gain : 분기 이전의 Entropy에서 분기 이후의 Entropy를 뺀 수치**, 즉 높을수록 잘 분기했다고 판단
- 단점 : Overfitting 문제 >> **pre-pruning, post-pruning (가지치기)**, Random Forest

3-1. Decision Tree 장단점

장점		단점	
결과 해석 용이	- 직관적인 해석 가능 - 주요 변수와 분리기준 제시	비안정성	- 데이터 수가 적을 경우 특히 불안정 - 과대적합 발생률 높음(가지치기 필요)
비모수적 모델	- 통계모델에 요구되는 가정에 자유로움 (e.g., 정규성 독립성, 등분산성)	선형성 미흡	- 전체적인 선형관계 파악 미흡
변수 간 상호작용	- 변수 간의 상호작용을 고려하며 선형/비선형 관계 탐색 가능	비연속성	- 분리 시 연속형 변수를 구간화 처리(비연속화) - 분리 경계점 근처에 오류 발생 가능

https://heytech.tistory.com/145

3-2. Entropy and Information Gain (ID3)

$$E(S) = \sum_{i=1}^{c} - p_i \log_2 p_i$$

Play Golf	
Yes	No
9	5

Entropy(PlayGolf) = Entropy (5,9)
= Entropy (0.36, 0.64)
= - (0.36 $\log_2$ 0.36) - (0.64 $\log_2$ 0.64)
= 0.94

$$E(T, X) = \sum_{c \in X} P(c) E(c)$$

		Play Golf		
		Yes	No	
Outlook	Sunny	3	2	5
	Overcast	4	0	4
	Rainy	2	3	5
				14

E(PlayGolf, Outlook) = **P**(Sunny)***E**(3,2) + **P**(Overcast)***E**(4,0) + **P**(Rainy)***E**(2,3)
= (5/14)*0.971 + (4/14)*0.0 + (5/14)*0.971
= 0.693

$$Gain(T, X) = Entropy(T) - Entropy(T, X)$$

G(PlayGolf, Outlook) = **E**(PlayGolf) – **E**(PlayGolf, Outlook)
= 0.940 – 0.693 = 0.247

3-3. Entropy and Information Gain (C4.5)

Information gain ratio

3-4. Entropy and Information Gain (CART)

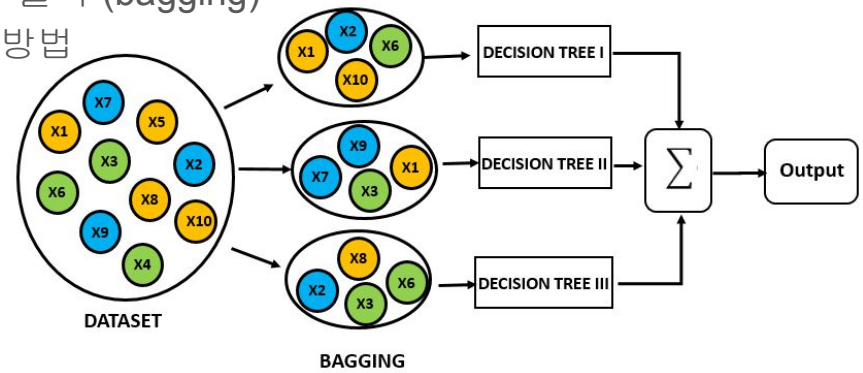
Gini index
Pre pruning - CCP

3-5. Pruning

- Decision Tree의 과적합을 줄이기 위해 수행
- Pre-pruning (사전 가지치기) : 트리 생성과정에서 가지치기
  - 트리가 일정 깊이에 도달하면 트리 생성 정지
  - 노드의 샘플 수가 임계값보다 작아지면 성장 정지
  - 분할이 검증 데이터의 정확도 향상 영향을 계산, 임계값보다 작아지면 성장 정지
  - **장점 : 계산이 간단, 효율적**
  - **단점 : 경험이 있어야 함, Underfitting 위험**
- Post-pruning (사후 가지치기) : 트리 생성 후 가지치기
  - CART의 CCP, … 종류 많음
  - **장점 : 사전 가지치기 보다 더 일반적인 성능 얻을 수 있음**
  - **단점 : 계산 시간이 오래걸림**

4. Random Forest

- 앙상블 머신러닝 모델
- Decision Tree로 생성된 Overfitting Tree에서 일반적인 결과 출력
- Algorithm
  1. 학습 데이터에서 n개 데이터 표본 선택 (bootstrap)
  2. k개 feature 중 $\sqrt{k}$ 개를 선택
  3. Decision Tree 생성
  4. 1~3 번의 과정을 m 번 반복
  5. 테스트 데이터에서 m개의 결과 중 다수결 결과 출력 (bagging)
  - Bagging : random forest에서 사용하는 앙상블 방법
  -



https://eunsukimme.github.io/ml/2019/11/26/Random-Forest/

앙상블

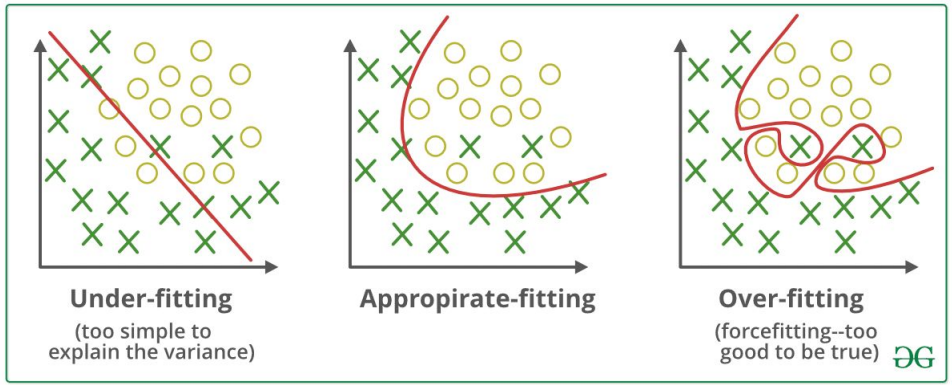
- Voting
  - 다른 알고리즘을 가진 분류기가 같은 데이터셋을 기반으로 학습되고 결합
- Bagging (bootstrap aggregating)
  - 같은 분류기가 같은 데이터셋을 기반으로 학습되고 결합 (부트스트랩)
  - 높은 bias의 underfitting 문제, 높은 variance로 인한 overfitting 문제 해결
  - categorical data는 투표, continuous data는 평균
  - 병렬적 구조
- Boosting
  - 직렬적 구조
- Stacking

https://libertegrace.tistory.com/entry/Classification-3-%EC%95%99%EC%83%81%EB%B8%94-%ED%95%99%EC%8A%B5Ensemble-Learning-Boosting

https://data-matzip.tistory.com/4

5. Overfitting

- 훈련 데이터에 과하게 맞추어져 훈련 데이터의 성능은 좋지만, 테스트 데이터에서는 성능이 저조
- Overfitting 방지
  1. 더 많은 데이터 확보
  2. 모델 복잡도 줄이기
  3. cross validation 방법 사용 : k-fold cross validation
  4. 정규화 사용 (LASSO, Ridge)
  5. 앙상블 학습 방법 사용
- Underfitting 방지
  1. 새로운 특성 추가
  2. 모델 복잡도 증가
  3. 정규화 계수 줄이기



6. Univariate, Bivariate and Multivariate analysis

- Univariate : 일변량 데이터
  - 1개의 feature
  - 평균, 중위수, 최빈값(mode), 산포도, 범위, 최대, 최소 등의 통계 분석 진행

- Bivariate : 이변량 데이터
  - 2개의 다른 feature
  - 원인과 영향을 두 변수 사이의 관계 비교를 통해 분석

- Multivariate : 다변량 데이터
  - 3개 이상의 feature

7. Feature Selection Method

- Filter Method
  - 각 변수들에 대해 통계적인 점수와 순위를 매기고 선택
  - Linear Discrimination Analysis
  - ANOVA
  - Chi-Square
- Wrapper Method
  - 변수의 일부만을 모델링에 사용 후, 평가 작업을 반복하여 변수 선택
  - Forward Selection
  - Backward Selection
  - Recursive Feature Elimination
- Embedded Method
  - 위의 두 방법을 결합하여 어떤 변수가 가장 크게 기여하는 지를 찾아내는 방법
  - LASSO
  - Ridge Regression
  - Elastic Net

7-1. Feature Selection vs Feature Extraction

- https://bioinformaticsandme.tistory.com/188

8. Python Print

- 3의 배수는 “fizz”
- 5의 배수는 “buzz”
- 3과 5의 배수는 “fizzbuzz”

```
for fizzbuzz in range(51):
    if fizzbuzz % 3 == 0 and fizzbuzz % 5 == 0:
        print("fizzbuzz")
        continue
    elif fizzbuzz % 3 == 0:
        print("fizz")
        continue
    elif fizzbuzz % 5 == 0:
        print("buzz")
        continue
    print(fizzbuzz)
```

9. Missing Value

- Missing data **삭제** (확보한 데이터가 충분히 클때)
- 특정 값으로 채우기
- 결측값의 앞 또는 뒷 방향의 값으로 채우기
- **mean, mode, medium**, trimmed mean
- **Knnimputer** 를 사용하여 결측치 채울 수 있음

- Pandas, scipy 등의 라이브러리를 사용하여 쉽게 채울 수 있음 (fillna)

https://rfriend.tistory.com/262

10. Euclidean Distance in Python

$$d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_i - q_i)^2 + \cdots + (p_n - q_n)^2}.$$

```
import math

plot1 = [1,3]
plot2 = [2,5]

euclidean_distance = math.sqrt((plot1[0]-plot2[0])**2 + (plot1[1]-plot2[1])**2)
```

```
euclidean_distance
```

```
2.23606797749979
```

11. Dimensionality Reduction

- 원래의 차원에서 작은 차원으로 변환
- 장점
  - 데이터 압축하여 저장 공간 감소
  - 계산 시간 감소
- 종류
  - **pca**
  - **auto-encoder**
  - **Linear Discriminant Analysis**
  - Independent Component Analysis
  - Isomap
  - Latent Semantic Analysis

11-1. PCA

-

11-2. AutoEncoder

-

11-3. Linear Discriminant Analysis

-

12. Eigenvalues and Eigenvectors

- Eigenvector : 어떤 벡터에 선형변환 결과가 방향은 변하지 않고 크기만 변환되는 벡터를 의미
- Eigenvalue : Eigenvector가 변환되는 크기를 의미

DEFINITION 1. 고윳값, 고유벡터

임의의 $n \times n$ 행렬 A 에 대하여, 0이 아닌 솔루션 벡터 $\vec{x}$ 가 존재한다면 숫자 λ 는 행렬 A 의 고윳값라고 할 수 있다.

$$A\vec{x} = \lambda\vec{x} \tag{2}$$

이 때, 솔루션 벡터 $\vec{x}$ 는 고윳값 λ 에 대응하는 고유벡터이다.

https://angeloyeo.github.io/2019/07/17/eigen_vector.html

12-1. Eigenvalues and Eigenvectors 계산 과정

선형 변환 A에 대해 Eigenvalue와 Eigenvector를 구하면 다음과 같다.

$$A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

$$det(A - \lambda I) = det\left(\begin{bmatrix} 2-\lambda & 1 \\ 1 & 2-\lambda \end{bmatrix}\right) = 0$$

$$\Rightarrow (2-\lambda)^2 - 1$$

$$= (4 - 4\lambda + \lambda^2) - 1$$

$$= \lambda^2 - 4\lambda + 3 = 0$$

그러므로, $\lambda_1 = 1$, $\lambda_2 = 3$이다.

$\lambda_1 = 1$인 경우에 대해,

$$A\vec{x} = \lambda_1 \vec{x}$$

$$\Rightarrow \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 1 \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$2x_1 + x_2 = x_1$$

$$x_1 + 2x_2 = x_2$$

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

$\lambda_2 = 3$인 경우의 고유벡터는

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

https://angeloyeo.github.io/2019/07/17/eigen_vector.html

12-2. PCA의 Eigenvalues and Eigenvectors

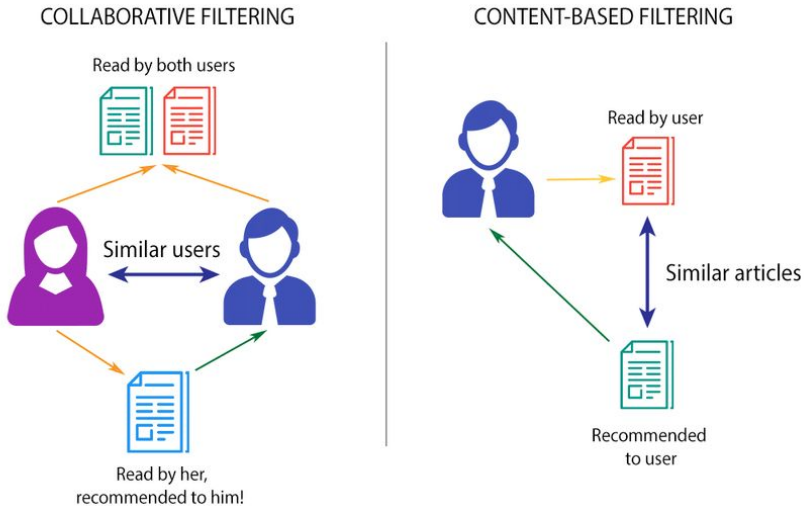
- pca

13. Model maintain

1. 모니터 : 제대로 작동하는지에 대한 지속적인 모니터링이 필요
2. 평가 : 새로운 알고리즘이 필요한지에 대한 여부를 판단
3. 비교 : 기존 모델과 새 모델을 비교
4. 재작성 : 성능이 더 좋은 모델로 변경

14. Recommender System

- 사용자가 자신의 선호도에 따라 특정 제품을 어떻게 생각할지 예측
- Collaborative Filtering
  - 다른 사용자와의 유사함에 기초
  - 비슷한 사용자가 좋아하는 아이템을 추천
  - ex) 아마존 추천 시스템..
- Content-based Filtering
  - 다른 아이템과의 유사함에 기초
  - 유사한 아이템을 추천
- 그 외
  - Hybrid Recommender System
  - Context-based Recommender System
  - …



15. RMSE and MSE

```
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error

# answer : y = 1 * x_0 + 2 * x_1 + 3
X = np.array([[1, 1], [1, 2], [2, 2], [2, 3]])
noise = np.array([0.00201, 0.00032, -0.0001, -0.071902])
y = np.dot(X, np.array([1, 4])) + 3 + noise

reg = LinearRegression().fit(X, y)
reg.score(X, y)

print('coefficients : ',reg.coef_)

print('intercept : ', reg.intercept_)
```

```
coefficients :  [0.99958  3.963254]
intercept :  3.056704
```

```
y_hat = reg.predict(np.array([[3, 5], [4, 5], [6, 7]]))
y_true = np.dot(np.array([[3, 5], [4, 5], [6, 7]]), np.array([1, 4])) + 3
```

```
print('y_hat : ', y_hat)
print('y_true : ', y_true)
```

```
y_hat :  [25.871714 26.871294 36.796962]
y_true :  [26 27 37]
```

```
print('rmse : ', mean_squared_error(y_true, y_hat, squared=False))
print('mse  : ', mean_squared_error(y_true, y_hat))
```

```
rmse :  0.1573181083834126
mse  :  0.024748987225335153
```

$$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \qquad RMSE = \sqrt{\sum_{i=1}^{n}\frac{(\hat{y}_i - y_i)^2}{n}}$$

15-1. Regression Metrics

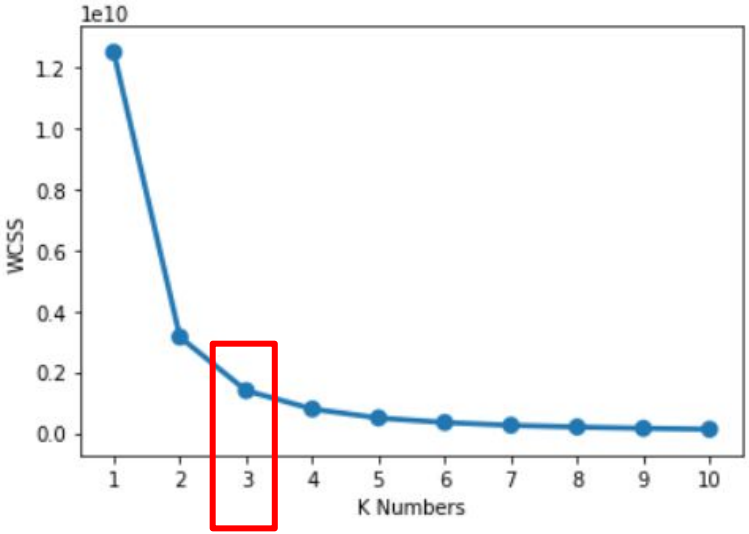
- MSE
- RMSE
- MAE
- R-Squared
- 등등.. 장단점

16. Select k for k-means?

- Elbow Method
  - 군집분석에서 군집수를 결정하는 방법
  - 군집내 총 제곱합 (WSS : Within cluster Sum of Squares) 을 계산하여 적절한 군집 수 설정
- WWS (Within Cluster Sum of Squares)

- **Within Cluster Sums of Squares :** $\mathrm{WSS} = \sum_{i=1}^{N_C} \sum_{x \in C_i} d(\mathbf{x}, \bar{\mathbf{x}}_{\mathbf{C_i}})^2$
- **Between Cluster Sums of Squares:** $\mathrm{BSS} = \sum_{i=1}^{N_C} |C_i| \cdot d(\bar{\mathbf{x}}_{\mathbf{C_i}}, \bar{\mathbf{x}})^2$

C_i = Cluster, N_c = # clusters, $\bar{x}_{c_i}$ = Cluster centroid, $\bar{x}$ = Sample Mean



16-1. WSS, BSS, TSS

- WSS
- BSS
- TSS

17. P-value

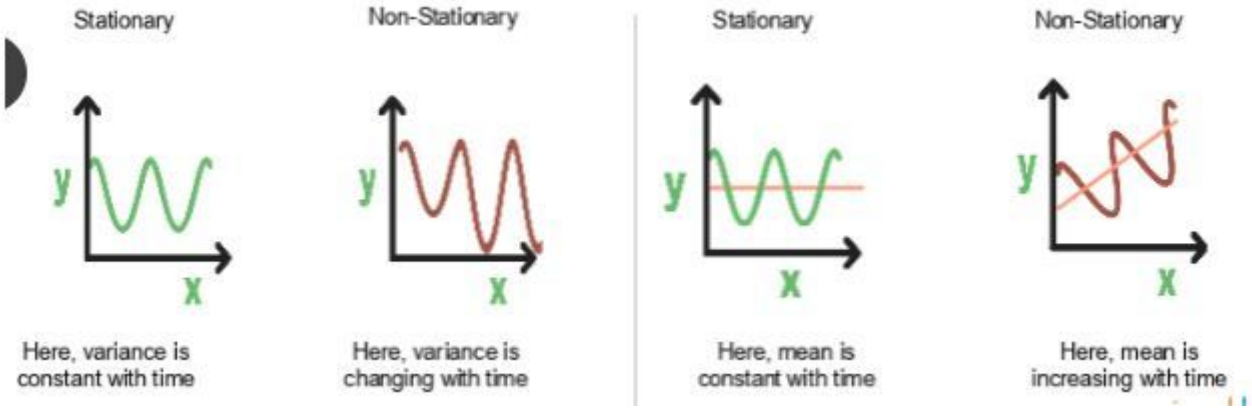
- 검정 통계량에 관한 확률로 크거나 같은 값을 얻을 수 있을 확률
- 귀무가설의 기각 여부를 결정
  - P-value < alpha : 귀무가설을 기각
  - P-value > alpha : 귀무가설을 수락
- 귀무가설 : 새로울게 없다는 가설, 똑같다는 가설
  - ex) 두 확률분포는 차이가 없다.
  - ex) 흡연 여부는 뇌혈관 질환의 발생에 영향을 미치지 않는다.

18. Outlier values treat

- 필요없는 데이터라면 삭제
- 다른 모델을 선택 (linear -> nonlinear)
- 데이터를 정규화
- 특이치에 강한 모델을 사용 (random forest)

19. Time series stationarity

- Stationarity : 시간이 변해도 일정한 분포를 따르는 경우
- 확인 방법
  - 그래프를 그려서 확인
  - 통계량의 변화를 확인
  - 통계적 검정



19-1. Time series stationarity

- 통계적 검정 방법

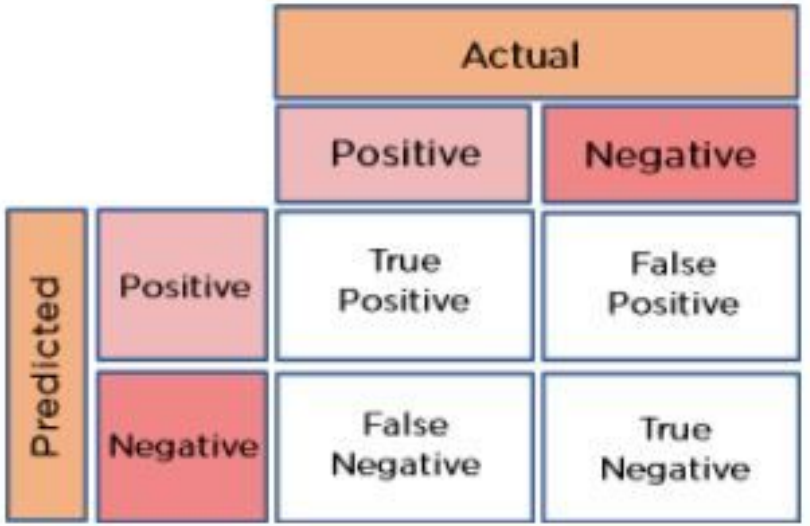
20. Confusion matrix

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

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$\text{F1-Score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$



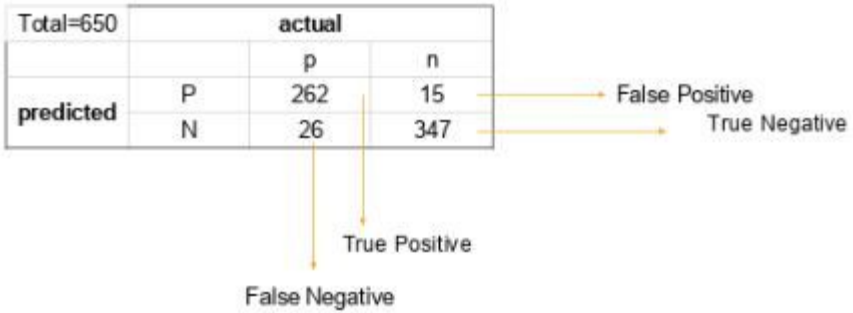
21. Precision and Recall Calculate

- precision = tp / (tp + fp)

  262 / 277 = 0.94

- recall = tp / (tp + fn)

  262 / 288 = 0.9



21-1. 평가지표의 한계

- Accuracy의 한계
- 정밀도와 재현율의 균형
- 평균제곱근오차의 예외

22. Basic SQL Query

- Order Table
  - OrderId
  - CustomerId
  - OrderNumber
  - TotalAmount
- Customer Table
  - Id
  - FirstName
  - LastName
  - City
  - Country

SQL query (모든 주문 리스트를 고객 정보와 같이 나열)

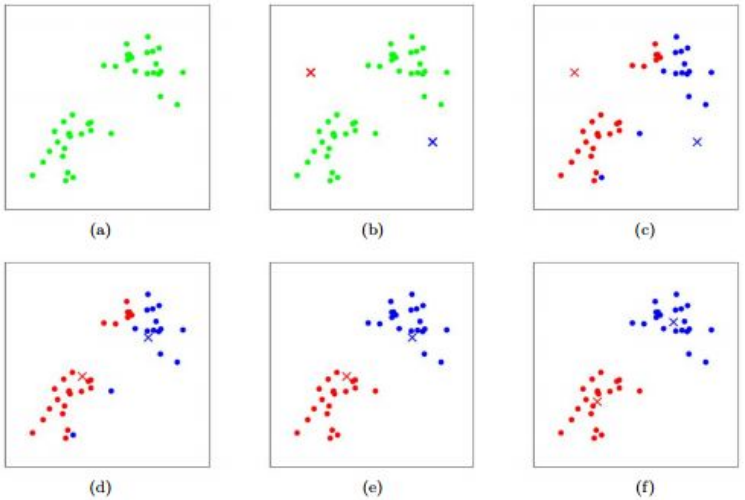
```
SELECT OrderNumber, TotalAmount, FirstName, LastName, City, Country

FROM Order

JOIN Customer

ON Order.CustomerId = Customer.Id
```

23. Data Imbalance Performance Matrix

- 라벨의 분포가 불균형한 경우
- Accuracy로 본다면 좋은 성능을 나타내지만 실제로 보면 좋지 못한 모델일 수 있음
  - 학습 데이터 : 99% 정상 데이터, 1% 이상 데이터
  - 모두 정상 데이터로 예측 시 Accuracy는 99% 이상일 수 있음
  - Positive를 이상 데이터로 할때
  - Precision은 낮게 나오고 Recall이 높게 나옴
  - fp(정상을 이상치로 예측) 가 높고
  - fn(이상치를 정상으로 예측) 가 낮음
- 위 경우 F1-score 를 이용

24. K-means clustering

- Clustering Algorithm
- Algorithm
  1. k 개의 중심점을 임의로 선택 (그전에 **정규화, outlier 제거** 필수)
  2. k 개의 중심점에 대해 가까운 데이터들을 이용하여 k개의 클러스터로 묶는다.
  3. 각 클러스터 데이터를 통해 새로운 중심점 k개를 계산한다.
  4. 중심점이 안바뀔때까지 2~3번 과정을 반복
- 장점
  - 쉽고, 계산이 빠르다
- 단점
  - Local minimum 으로 수렴
  - K 값을 임의로 정해야 함
  - 처음 중심점에 따라 결과가 크게 바뀜
  - Outlier에 민감
  - 원형의 클러스터로 구성

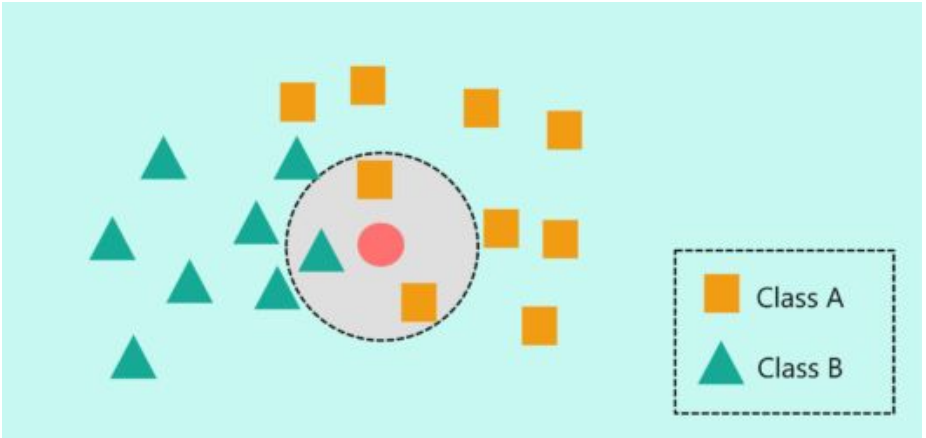


25. Linear Regression

- Algorithm
- 장점
- 단점
  - **회귀 모형의 4가지 가정 (선형성, 독립성, 등분산성, 정규성)**
  - Categorical, binary 문제 X
- 실제 사용 예시

26. KNN (k-nearest neighbors)

- Classification Algorithm
- Algorithm
  1. 테스트 데이터를 입력한다.
  2. 테스트 데이터와 다른 데이터들의 거리를 계산한다.
  3. 가장 가까운 K개의 데이터를 뽑는다.
  4. K개의 데이터 중 가장 많은 클래스를 최종 클래스로 선정
- 장점
  - 단순하고 효율적
- 단점
  - 적절한 k를 설정하기 어렵다.
  - 데이터가 많아지면 계산량이 많아진다.
- **데이터 정규화 과정이 필요함**

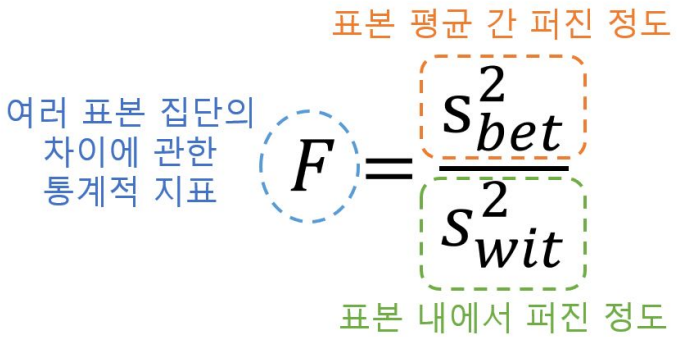


27. Association Rule

- Algorithm
- 장점
- 단점
- 실제 사용 예시

28. ANOVA

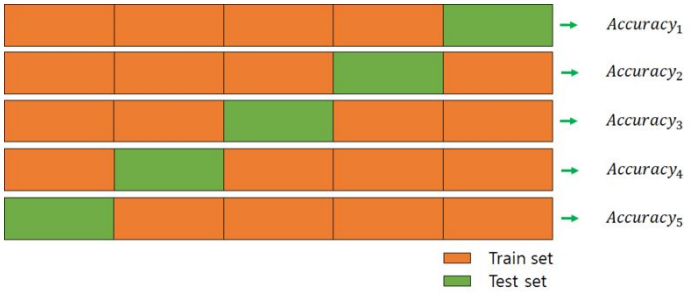
- 3개 이상 다수의 집단을 비교할 때 사용하는 가설검정 방법
- F 분포를 이용
- t-value ~ f-value 같은 의미를 지님



28-1. ANOVA 예제

-

29. 모델 평가 방법

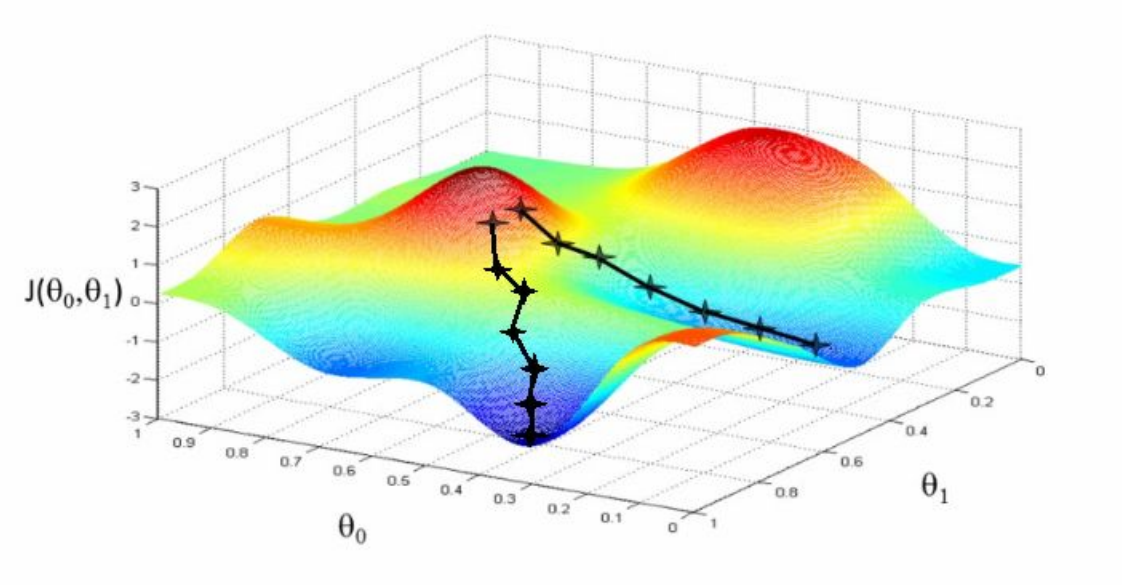


$$Accuracy = Average(Accuracy_1, \cdots, Accuracy_k)$$

- Holdout Validation
  - 초기 데이터를 일정 비율로 훈련, 검증 데이터로 구분
  - 초기 데이터를 어떻게 분리하는냐에 따라 모델 성능에 영향을 크게 끼침
- Cross Validation
  - **K-fold cross validation**
    - k개의 묶음으로 분리하고 k번 평가 후 평균값을 최종 평가 지표로 사용
  - LOOCV (Leave One Out Cross Validation)
    - 샘플 수 n개에 대해 모두 검정, 모든 샘플의 검증값을 평균하여 평가 지표로 사용
    - LpOCV의 한종류
    - 두 경우 모두 시간이 오래 걸림 (LOOCV < LpOCV)
- Bootstrapping
  - N개의 샘플에서 n번 복원 추출하여 n개의 훈련 데이터를 얻는다.
  - 원래 n개 샘플 데이터에서 추출되지 않은 데이터를 검정 데이터로 설정하고 평가한다.
  - 이 때의 검정 데이터를 **OOB (Out of Bag)**이라고 한다.

30. Gradient Descent method

- 최적화 알고리즘
- 매개변수를 업데이트하는데 사용
- 손실함수의 최적해를 찾기 위해 사용



31. Resampling

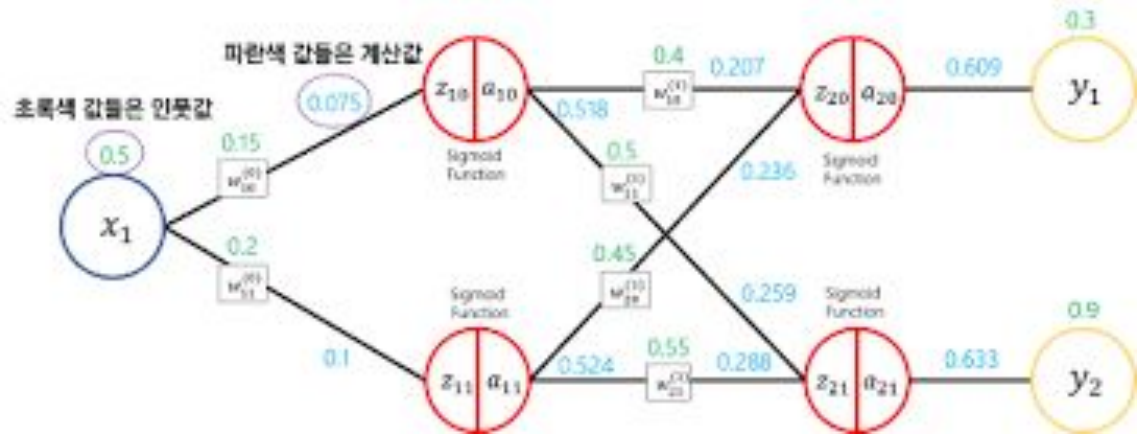
-

32. Bias

- Household bias
- Nonresponse bias
- Quota sampling bias
- Response bias
- **Selection bias**
- Size bias
- **Undercoverage bias**
- Voluntary response bias
- Word bias
- **Survivorship bias**

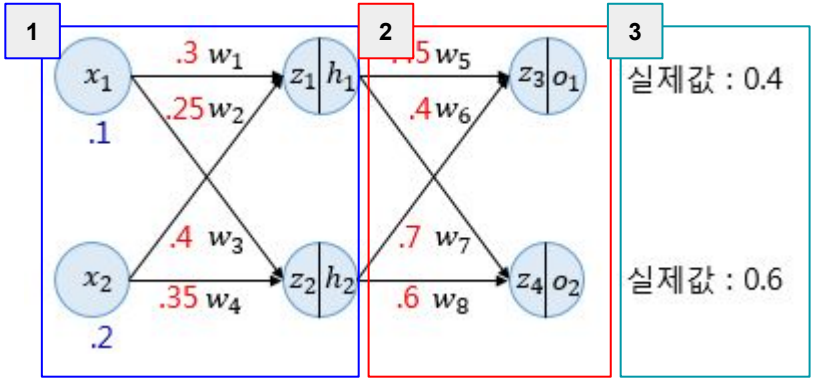
33. BackPropagation

- 신경망을 학습하기 위해 사용하는 방법
- gradient를 계산하며 신경망의 파라미터를 최적화



https://wikidocs.net/37406

33-1. BackPropagation 예제



손실 함수 : 평균 제곱 오차 사용

3

$$E_{o1} = \frac{1}{2}(target_{o1} - output_{o1})^2 = 0.02193381$$

$$E_{o2} = \frac{1}{2}(target_{o2} - output_{o2})^2 = 0.00203809$$

$$E_{total} = E_{o1} + E_{o2} = 0.02397190$$

1

$$z_1 = w_1x_1 + w_2x_2 = 0.3\times0.1 + 0.25\times0.2 = 0.08$$

$$z_2 = w_3x_1 + w_4x_2 = 0.4\times0.1 + 0.35\times0.2 = 0.11$$

$$h_1 = sigmoid(z_1) = 0.51998934$$

$$h_2 = sigmoid(z_2) = 0.52747230$$

2

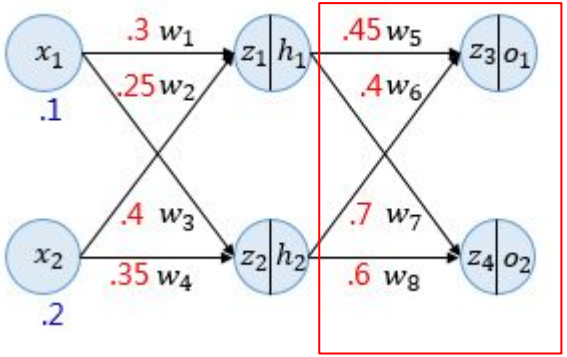
$$z_3 = w_5h_1 + w_6h_2 = 0.45\times h_1 + 0.4\times h_2 = 0.44498412$$

$$z_4 = w_7h_1 + w_8h_2 = 0.7\times h_1 + 0.6\times h_2 = 0.68047592$$

$$o_1 = sigmoid(z_3) = 0.60944600$$

$$o_2 = sigmoid(z_4) = 0.66384491$$

33-1. BackPropagation 예제



실제값 : 0.4

실제값 : 0.6

목표 : 손실함수에 대한 가중치 gradient 계산

$$w_5, w_6, w_7, w_8$$

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial o_1} \times \frac{\partial o_1}{\partial z_3} \times \frac{\partial z_3}{\partial w_5}$$

(By chain rule)

$$\frac{\partial E_{total}}{\partial o_1} = 2 \times \frac{1}{2}(target_{o1} - output_{o1})^{2-1} \times (-1) + 0$$

$$\frac{\partial E_{total}}{\partial o_1} = -(target_{o1} - output_{o1}) = -(0.4 - 0.60944600) = 0.20944600$$

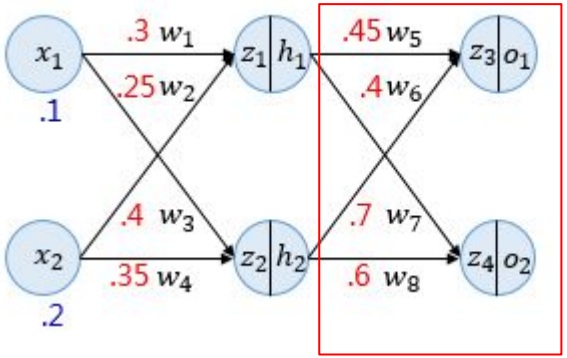
$$\frac{\partial o_1}{\partial z_3} = o_1 \times (1 - o_1) = 0.60944600(1 - 0.60944600) = 0.23802157$$

(sigmoid 미분, 2-3참고) $f(x) \times (1 - f(x))$

$$\frac{\partial z_3}{\partial w_5} = h_1 = 0.51998934$$

$$\frac{\partial E_{total}}{\partial w_5} = 0.20944600 \times 0.23802157 \times 0.51998934 = 0.02592286$$

33-1. BackPropagation 예제



실제값 : 0.4

실제값 : 0.6

목표 : 손실함수에 대한 가중치 gradient 계산

$$w_5, w_6, w_7, w_8$$

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial o_1} \times \frac{\partial o_1}{\partial z_3} \times \frac{\partial z_3}{\partial w_5}$$

(By chain rule)

가중치 업데이트

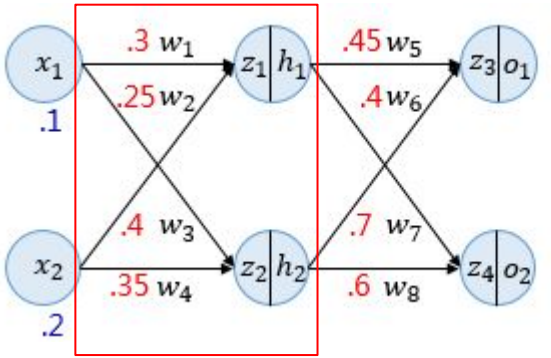
$$w_5^+ = w_5 - \alpha \frac{\partial E_{total}}{\partial w_5} = 0.45 - 0.5 \times 0.02592286 = 0.43703857$$

$$\frac{\partial E_{total}}{\partial w_6} = \frac{\partial E_{total}}{\partial o_1} \times \frac{\partial o_1}{\partial z_3} \times \frac{\partial z_3}{\partial w_6} \rightarrow w_6^+ = 0.38685205$$

$$\frac{\partial E_{total}}{\partial w_7} = \frac{\partial E_{total}}{\partial o_2} \times \frac{\partial o_2}{\partial z_4} \times \frac{\partial z_4}{\partial w_7} \rightarrow w_7^+ = 0.69629578$$

$$\frac{\partial E_{total}}{\partial w_8} = \frac{\partial E_{total}}{\partial o_2} \times \frac{\partial o_2}{\partial z_4} \times \frac{\partial z_4}{\partial w_8} \rightarrow w_8^+ = 0.59624247$$

33-1. BackPropagation 예제



실제값 : 0.4

실제값 : 0.6

목표 : 손실함수에 대한 가중치 gradient 계산

$$w_1, w_2, w_3, w_4$$

$$\frac{\partial E_{total}}{\partial w_1} = \frac{\partial E_{total}}{\partial h_1} \times \frac{\partial h_1}{\partial z_1} \times \frac{\partial z_1}{\partial w_1}$$

(By chain rule)

$$\frac{\partial E_{total}}{\partial h_1} = \frac{\partial E_{o1}}{\partial h_1} + \frac{\partial E_{o2}}{\partial h_1}$$

$$\frac{\partial E_{o1}}{\partial h_1} = \frac{\partial E_{o1}}{\partial z_3} \times \frac{\partial z_3}{\partial h_1} = \frac{\partial E_{o1}}{\partial o_1} \times \frac{\partial o_1}{\partial z_3} \times \frac{\partial z_3}{\partial h_1}$$

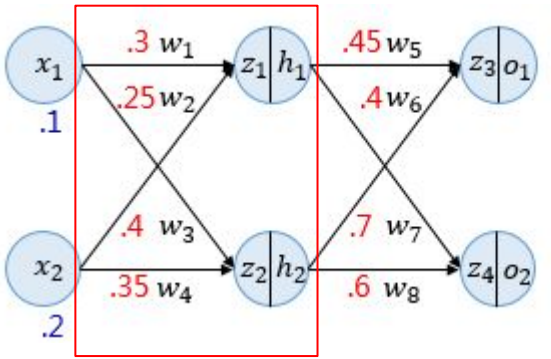
$$= -(target_{o1} - output_{o1}) \times o_1 \times (1 - o_1) \times w_5$$

$$= 0.20944600 \times 0.23802157 \times 0.45 = 0.02243370$$

$$\frac{\partial E_{o2}}{\partial h_1} = \frac{\partial E_{o2}}{\partial z_4} \times \frac{\partial z_4}{\partial h_1} = \frac{\partial E_{o2}}{\partial o_2} \times \frac{\partial o_2}{\partial z_4} \times \frac{\partial z_4}{\partial h_1} = 0.00997311$$

$$\frac{\partial E_{total}}{\partial h_1} = 0.02243370 + 0.00997311 = 0.03240681$$

33-1. BackPropagation 예제



실제값 : 0.4

실제값 : 0.6

목표 : 손실함수에 대한 가중치 gradient 계산

$$w_1, w_2, w_3, w_4$$

$$\frac{\partial E_{total}}{\partial w_1} = \frac{\partial E_{total}}{\partial h_1} \times \frac{\partial h_1}{\partial z_1} \times \frac{\partial z_1}{\partial w_1}$$

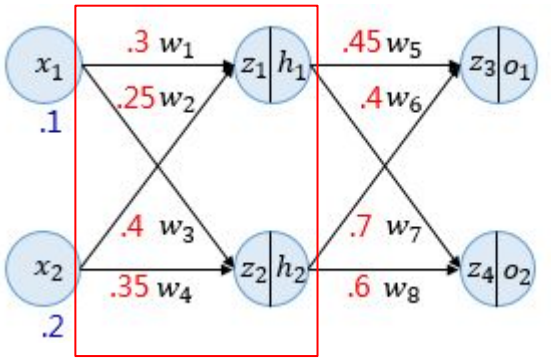
(By chain rule)

$$\frac{\partial h_1}{\partial z_1} = h_1 \times (1 - h_1) = 0.51998934(1 - 0.51998934) = 0.24960043$$

$$\frac{\partial z_1}{\partial w_1} = x_1 = 0.1$$

$$\frac{\partial E_{total}}{\partial w_1} = 0.03240681 \times 0.24960043 \times 0.1 = 0.00080888$$

33-1. BackPropagation 예제



실제값 : 0.4

실제값 : 0.6

목표 : 손실함수에 대한 가중치 gradient 계산

$$w_1, w_2, w_3, w_4$$

$$\frac{\partial E_{total}}{\partial w_1} = \frac{\partial E_{total}}{\partial h_1} \times \frac{\partial h_1}{\partial z_1} \times \frac{\partial z_1}{\partial w_1}$$

(By chain rule)

가중치 업데이트

$$w_1^+ = w_1 - \alpha \frac{\partial E_{total}}{\partial w_1} = 0.1 - 0.5 \times 0.00080888 = 0.29959556$$

$$\frac{\partial E_{total}}{\partial w_2} = \frac{\partial E_{total}}{\partial h_1} \times \frac{\partial h_1}{\partial z_1} \times \frac{\partial z_1}{\partial w_2} \rightarrow w_2^+ = 0.24919112$$

$$\frac{\partial E_{total}}{\partial w_3} = \frac{\partial E_{total}}{\partial h_2} \times \frac{\partial h_2}{\partial z_2} \times \frac{\partial z_2}{\partial w_3} \rightarrow w_3^+ = 0.39964496$$

$$\frac{\partial E_{total}}{\partial w_4} = \frac{\partial E_{total}}{\partial h_2} \times \frac{\partial h_2}{\partial z_2} \times \frac{\partial z_2}{\partial w_4} \rightarrow w_4^+ = 0.34928991$$

34. A/B Test

- 대조군과 실험군으로 나누어서 효과를 비교하는 방법
- 모델의 최종 효과를 검증하는 최종 수단
  - 오프라인 평가로는 모델의 과적합 위험을 모두 제거 힘듬
  - 오프라인 평가로는 지연, 데이터 손실, 레이블 손실 등과 같은 상황 반영 어려움
- 실험군 : 새로운 모델
- 대조군 : 기존 모델
- 사용자는 랜덤으로 정해야 샘플의 무편향성을 유지 가능

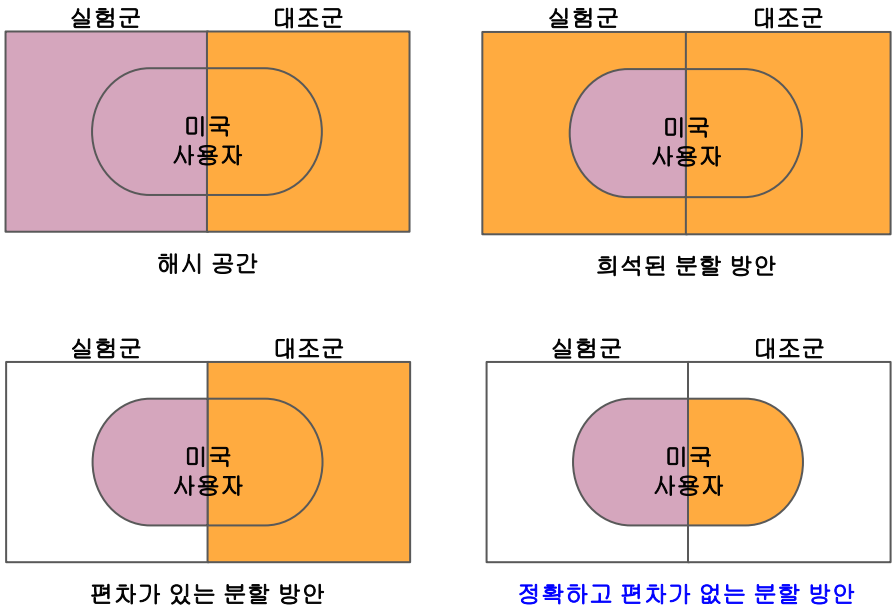
34-1. A/B Test 예제

- “미국 사용자”에 대해 새로운 콘텐츠 추천 모델 A를 적용시켜 보고자 한다. 현재 사용자에게 적용되고 있는 추천 알고리즘 모델은 B이다. 실험군/대조군 분류 방법중 정확한 방법을 골라라

1. User_id에 기반하여 끝자리가 홀수인 사용자들에 대해 실험군과 대조군으로 나누고, 실험군에 대해 A를 적용. 그리고 대조군에는 B를 적용
2. User_id 끝자리가 홀수인 미국 사용자들을 실험군으로 나머지 사용자들은 대조군으로 분류
3. User_id 끝자리가 홀수인 미국 사용자들은 실험군으로 User_id 끝자리가 짝수인 사용자들을 대조군으로 분류
4. **User_id 끝자리가 홀수인 미국 사용자들은 실험군으로 User_id 끝자리가 짝수인 미국 사용자들을 대조군으로 분류**

34-1. A/B Test 예제

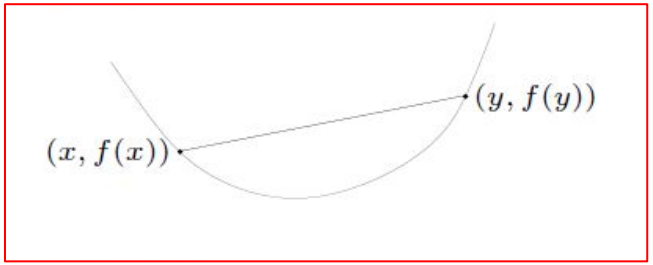
1. User_id에 기반하여 끝자리가 홀수인 사용자들에 대해 실험군과 대조군으로 나누고, 실험군에 대해 A를 적용. 그리고 대조군에는 B를 적용
2. User_id 끝자리가 홀수인 미국 사용자들을 실험군으로 나머지 사용자들은 대조군으로 분류
3. User_id 끝자리가 홀수인 미국 사용자들은 실험군으로 User_id 끝자리가 짝수인 사용자들을 대조군으로 분류
4. **User_id 끝자리가 홀수인 미국 사용자들은 실험군으로 User_id 끝자리가 짝수인 미국 사용자들을 대조군으로 분류**



35. Convex Function

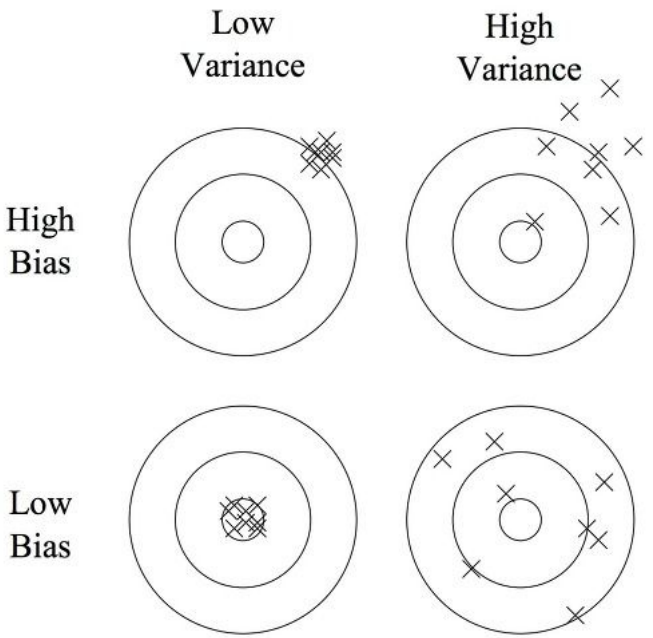
- 임의의 두 점을 직선으로 연결했을 때, 이 직선 위의 임의의 점은 해당 컨벡스 함수 아래에 위치하지 않는다.
- Logistic regression 문제가 convex 최적화 문제
- 컨벡스 최적화 문제는 모든 국소 최솟값이 전역 최솟값이므로 쉽게 풀수 있는 문제로 간주됨
- Non-convex 문제 : pca
  - 그러나 pca는 svd를 사용하여 전역 최솟값을 구할 수 있음

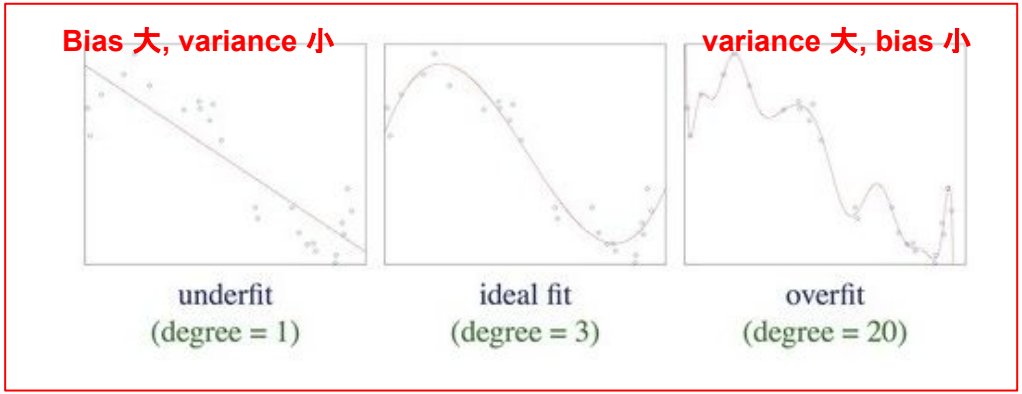
$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta) f(y)$$



36. Bias-Variance Tradeoff

- Error(X) = noise(X) + bias(X) + variance(X)
  - Noise : 데이터가 가지는 본질적인 한계치, irreducible error
  - Bias, Variance : 모델에 따라 변하는 것, reducible error
- Bias
  - 잘못된 것들을 학습하는 경향
  - Underfitting
  - 알고리즘의 평균 정확도가 얼마나 많이 변하는지
- Variance
  - random한 것들까지 학습하는 경향
  - Overfitting
  - 알고리즘이 얼마나 민감한지





36-1. Bias-Variance Tradeoff

- Target : $t = f(\mathbf{x}) + \epsilon$ ϵ은 평균이 0이고 분산이 σ^2 normal distribution
- Loss Function : MSE

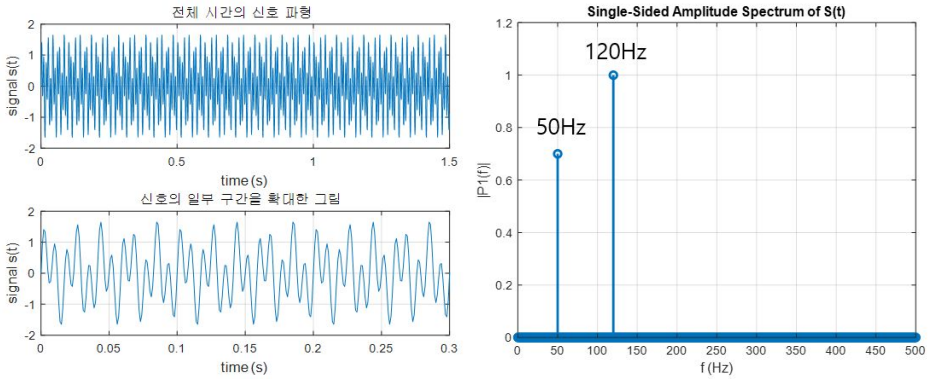
$$
\begin{aligned}
\mathbb{E}\left\{(t-y)^2\right\} &= \mathbb{E}\left\{(t-f+f-y)^2\right\} \\
&= \mathbb{E}\left\{(t-f)^2\right\} + \mathbb{E}\left\{(f-y)^2\right\} + 2\mathbb{E}\left\{(f-y)(t-f)\right\} \\
&= \mathbb{E}\left\{\epsilon^2\right\} + \mathbb{E}\left\{(f-y)^2\right\} + \underbrace{2\left[\mathbb{E}\left\{ft\right\} - \mathbb{E}\left\{f^2\right\} - \mathbb{E}\left\{yt\right\} + \mathbb{E}\left\{yf\right\}\right]}_{=0} \\
&= \mathbb{E}\left\{(f-\mathbb{E}\left\{y\right\} + \mathbb{E}\left\{y\right\} - y)^2\right\} + \mathbb{E}\left\{\epsilon^2\right\} \\
&= \mathbb{E}\left\{(f-\mathbb{E}\left\{y\right\})^2\right\} + \mathbb{E}\left\{(\mathbb{E}\left\{y\right\} - y)^2\right\} + \underbrace{2\mathbb{E}\left\{(\mathbb{E}\left\{y\right\} - y)(f-\mathbb{E}\left\{y\right\})\right\}}_{=0} + \mathbb{E}\left\{\epsilon^2\right\} \\
&= \underbrace{\mathbb{E}\left\{(f-\mathbb{E}\left\{y\right\})^2\right\}}_{\text{bias}^2} + \underbrace{\mathbb{E}\left\{(\mathbb{E}\left\{y\right\} - y)^2\right\}}_{\text{variance}} + \underbrace{\mathbb{E}\left\{\epsilon^2\right\}}_{\text{noise}}
\end{aligned}
$$

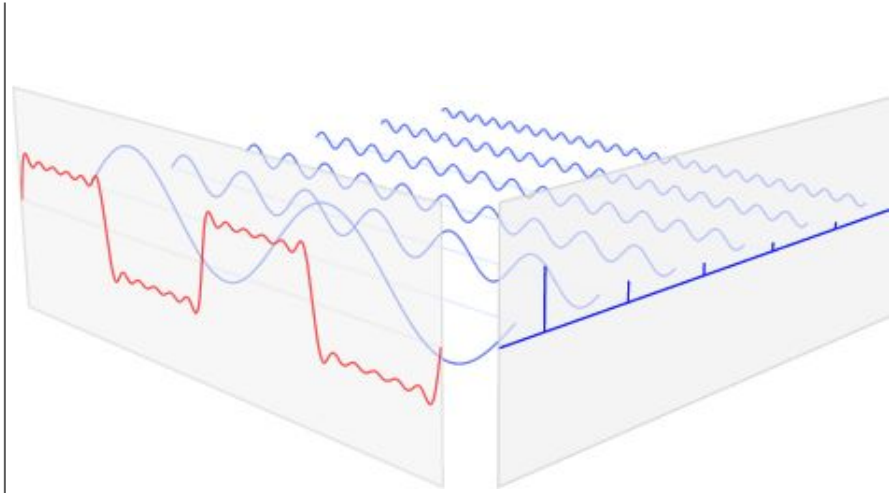
f, t 의 기댓값은 f

- **Bias 최소화 : E(y) = f 가 되도록 모델 학습, variance E(e^2) 이 된다. (Overfitting)**
- **Variance 최소화 : 특정 상수 a만을 반환 한다면, bias값 증가. (underfitting)**

37. Fourier Transform

- 임의의 주기함수는 삼각함수의 합으로 표현될 수 있다.





38. Bayes' Theorem

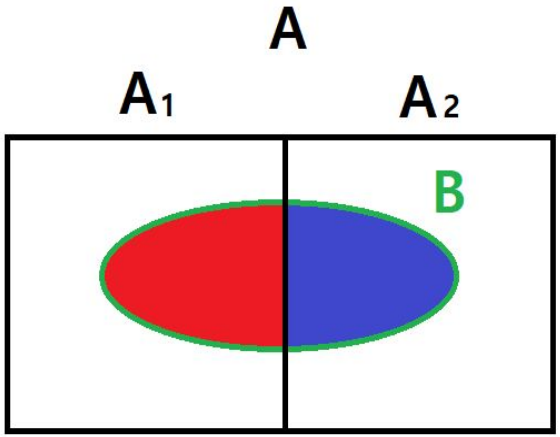
- 수식

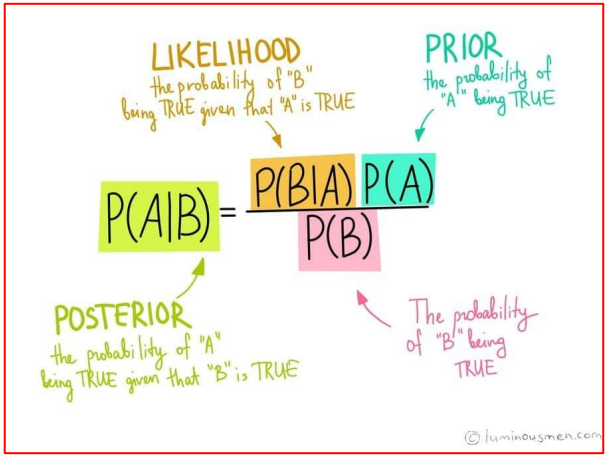
• 베이즈 정리

$$P(A_1|B) = \frac{P(B \cap A_1)}{P(B)}$$

$$= \frac{P(B|A_1)P(A_1)}{P(B)}$$

$$= \frac{P(B|A_1)P(A_1)}{P(B|A_1)P(A_1) + P(B|A_2)P(A_2)}$$





- 위 식에서 $p(A_1)$는 A_1의 사전 확률(**Prior Probability**), $p(A_1|B)$는 A_1의 사후 확률(**Posterior Probability**), $p(B|A_1)$는 우도(**Likelihood**), $p(B)$는 B의 사전 확률

38-1. Bayes' Theorem Example

- 유방암의 발병률 : 0.1%, 실제 유방암에 걸린 사람의 양성반응 : 99%, 건강한 사람의 양성반응 : 2%, 이때 어떤 사람의 검사 결과가 양성 반응을 보였다면 이 사람이 실제로 유방암에 걸렸을 확률은 얼마일까?
- B : 검사결과가 양성, A : 실제로 유방암, p(A|B) 구하기
- p(A) = 0.001, p(B|A) = 0.99

$$
\begin{aligned}
p(B) &= p(B|A)p(A) + p(B|A^c)p(A^c) \\
&= 0.99 * 0.001 + 0.02 * 0.999 \\
&= 0.020079
\end{aligned}
$$

$$
p(A|B) = \frac{p(B|A)p(A)}{p(B)} = \frac{0.99 * 0.001}{0.020079} = 0.049
$$

39. Covariance vs Correlation

- Cov(X, Y) > 0 : X가 증가 할 때 Y도 증가한다.
- Cov(X, Y) < 0 : X가 증가 할 때 Y는 감소한다.
- Cov(X, Y) = 0 : 두 변수간에는 아무런 선형관계가 없으며 두 변수는 서로 독립적인 관계에 있음을 알 수 있다. 그러나 두 변수가 독립적이라면 공분산은 0이 되지만, **공분산이 0이라고 해서 항상 독립적이라고 할 수 없다.**

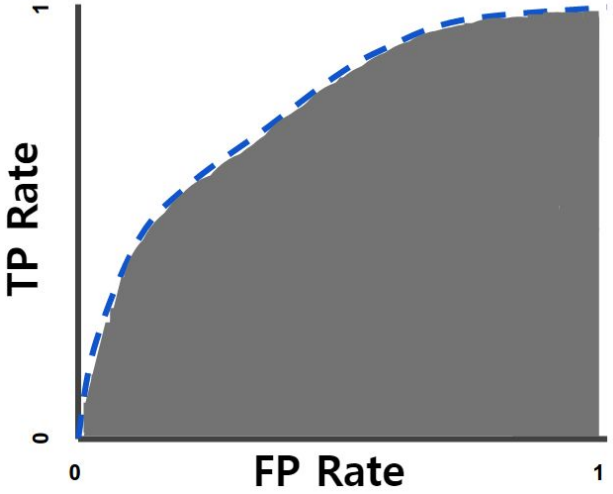
$$Cov(X,\ Y)\ =\ E((X\ -\ \mu)(Y\ -\ \nu))$$

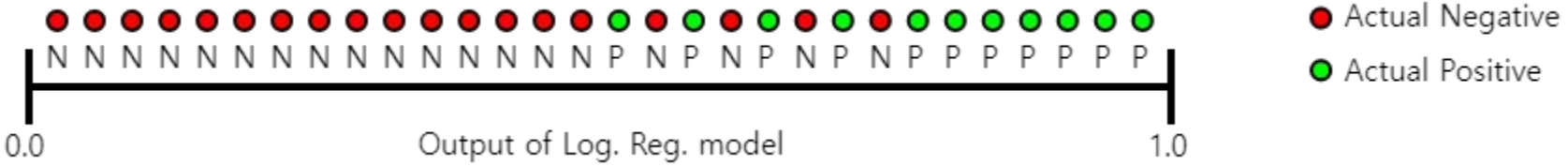
- 공분산은 X와 Y의 단위의 크기에 영향을 받는다. 그래서 절대적 크기에 영향을 받지 않도록 **단위화**

$$\rho\ =\ \frac{Cov(X,\ Y)}{\sqrt{Var(X)\ \ Var(Y)}},\quad -1\leq\ \rho\ \leq 1$$

40. ROC Curve

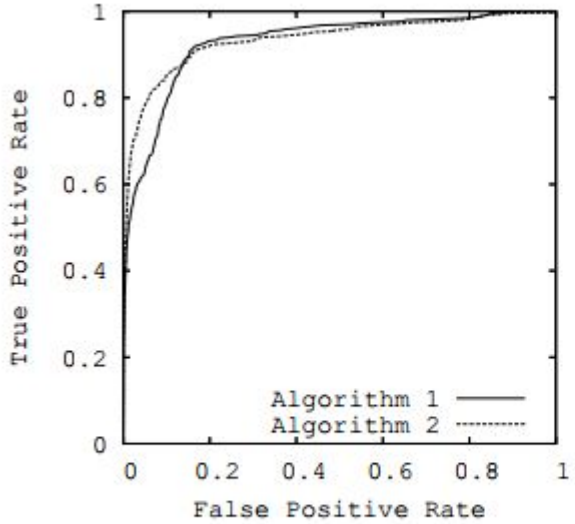
- TPR = TP / (TP + FN) == Recall
- FPR = FP / (FP + TN)
- Threshold를 움직여가며 TPR, FPR를 계산
- **FPR를 x축, TPR를 y축**으로 하여 그래프를 그린다.
- **AUC는 ROC Curve의 아래 면적**



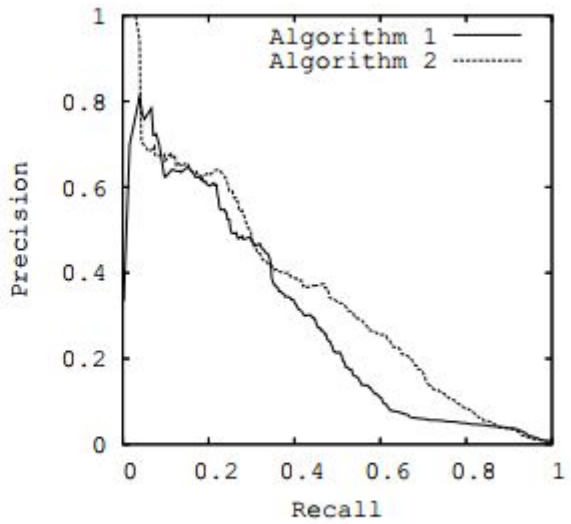


40-1. ROC Curve vs P-R Curve

- 일반적으로 성능지표의 차이만 날뿐
- **그러나 클래스 불균형 문제에서는 P-R 곡선이 크게 변화한다.**



(a) Comparison in ROC space



(b) Comparison in PR space

41. Naive Bayes Classification

- **목표 : 밑의 Bayes 정리를 사용하여 데이터x가 클래스 k에 속할 확률을 계산한다.**

$$p(C_k|\mathbf{x}) = \frac{p(C_k)\,p(\mathbf{x}|C_k)}{p(\mathbf{x})}.$$

- 데이터의 특징들이 서로 독립이라 가정하면 다음과 같이 전개 할 수 있습니다.
- 따라서 각 클래스의 사후 확률을 계산하여 가장 큰 값을 가지는 클래스로 데이터를 예측 할 수 있다.

$$\begin{aligned} p(C_k|x_1,\ldots,x_n) &\propto p(C_k,x_1,\ldots,x_n) \\ &\propto p(C_k)\,p(x_1|C_k)\,p(x_2|C_k)\,p(x_3|C_k)\,\cdots \\ &\propto p(C_k)\prod_{i=1}^{n} p(x_i|C_k)\,. \end{aligned}$$

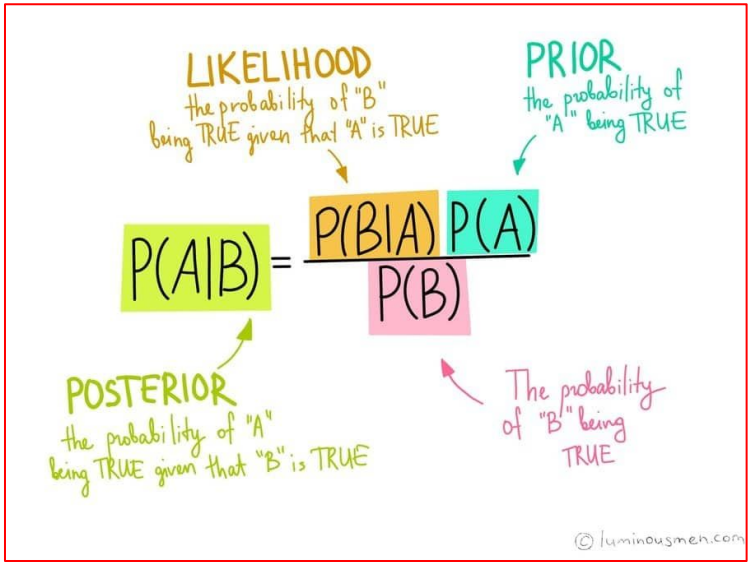
41-1. Naive Bayes Classification Example

- 날씨가 overcast, 기온이 Mild일 때 경기를 할 확률
  - 목표 : P(Play=Yes | Weather=Overcast, Temp=Mild), P(Play=No | Weather=Overcast, Temp=Mild) 확률 비교
  - **P(Weather=Overcast, Temp=Mild | Play=Yes)** P(Play=Yes) / P(Weather=Overcast, Temp=Mild)
  - **P(Weather=Overcast, Temp=Mild | Play=Yes) = P(Overcast|Yes) P(Mild|Yes) = 0.44 * 0.44 = 0.1936**
  - **P(Overcast|Yes) = 4/9 = 0.44, P(Mild|Yes) = 4/9 = 0.44**
  - **P(Yes) = 9/14 = 0.64**
  - **P(Weather=Overcast, Temp=Mild) = P(Weather=Overcast) P(Temp=Mild) = (4/14) * (6/14) = 0.1224**
  - **P(Play=Yes | Weather=Overcast, Temp=Mild) = P(Weather=Overcast, Temp=Mild | Play=Yes) P(Play=Yes) / P(Weather=Overcast, Temp=Mild) = 0.1936 * 0.64 / 0.1224 = 1**
  - 위와 비슷하게 하여 **P(Play=No | Weather=Overcast, Temp=Mild) = 0 * 0.36 / 0.1224 = 0**
- **따라서 overcast이고, Mild 하면 경기 한다.!**

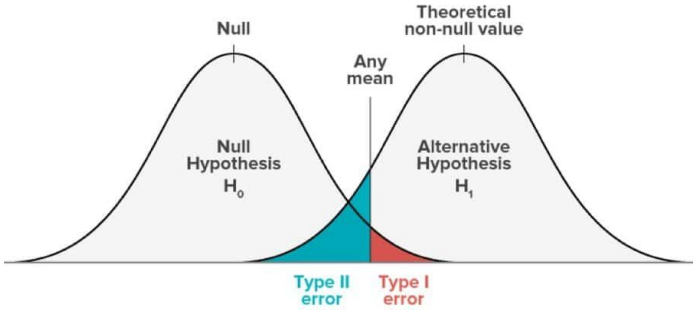
Whether	Temperature	Play
Sunny	Hot	No
Sunny	Hot	No
Overcast	Hot	Yes
Rainy	Mild	Yes
Rainy	Cool	Yes
Rainy	Cool	No
Overcast	Cool	Yes
Sunny	Mild	No
Sunny	Cool	Yes
Rainy	Mild	Yes
Sunny	Mild	Yes
Overcast	Mild	Yes
Overcast	Hot	Yes
Rainy	Mild	No

41-2. Naive Bayes Classification

- Posterior : 목적으로 하는 확률
- Likelihood : 학습 데이터에서 분석한 확률
- Prior : 학습 데이터에서의 클래스 확률



42. Type I and Type II Error



- 귀무 가설(H0): μ1= μ2 (두 약품의 효과가 동일)
- 대립 가설(H1): μ1≠ μ2 (두 약품의 효과가 동일 X)
- **제1종 오류 : 두 약품이 다르지 않지만 연구자가 귀무 가설을 기각하고 두 약품이 다르다는 결론을 내리는 경우**
- **제2종 오류 : 두 약품이 다르지만 연구자가 귀무 가설을 긍정하고 두 약품이 동일하다는 결론을 내리는 경우**

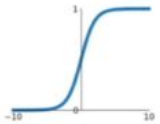
	모집단에 대한 사실	
표본을 기반으로 한 결정	H_0가 참	H_0가 거짓
H_0를 기각할 수 없음	옳은 결정(확률 = 1 - α)	**제2종 오류** - H_0가 거짓인데 기각하지 않음(확률 = β)
H_0를 기각	**제1종 오류** - H_0가 참인데 기각(확률 = α)	올바른 결정(확률 = 1 - β)

43. Activation Function

- 딥러닝 네트워크에서 노드에 입력된 값들을 비선형 함수에 통과시킨 후 다음 레이어로 전달하는 함수
- 비선형 함수 : 진선형태의 함수가 아닌 것 (기울기가 변함)
- **비선형 함수를 사용하는 이유 : 더 많은 layer를 쌓기 위해**
  - 선형 함수를 이용하여 쌓으면 하나의 layer로 합칠 수 있다.
- 종류

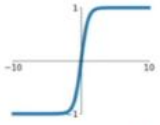
Sigmoid

$\sigma(x) = \frac{1}{1+e^{-x}}$



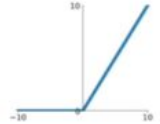
tanh

$\tanh(x)$



ReLU

$\max(0, x)$



Leaky ReLU

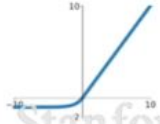
$\max(0.1x, x)$

Maxout

$\max(w_1^T x + b_1, w_2^T x + b_2)$

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



44. Gradient Vanishing 문제

- **Sigmoid, tanh** 같은 활성화 함수를 사용하면 **그레디언트 값**이 점점 작아져 **소실되는 문제**가 발생한다.
  - 미분 값이 특정 구간을 제외하면 거의 0에 수렴
- 해결책 **ReLU**
  - 0 이상의 값을 가지면 기울기가 1로 고정
  - 학습 빠르고, 연산 비용 적고, 구현 간단
- Leaky ReLU
  - ReLU에서 0이하의 기울기가 0으로 고정되어 뉴런이 죽는 경우 발생 (Dying ReLU)
  - Dying ReLU 문제를 해결하기 위해 나온 방법

45. Bagging vs Boosting

- Bagging
  - 데이터로부터 복원추출을 통해 n개의 bootstrap sample 생성.
  - 해당 sample에 대해서 모델 학습.
  - 위 과정을 M번 반복한 후 최종 Bagging 모델을 다음과 같이 정의
  - 대표적인 모델 : Random Forest
  - 병렬적 방식

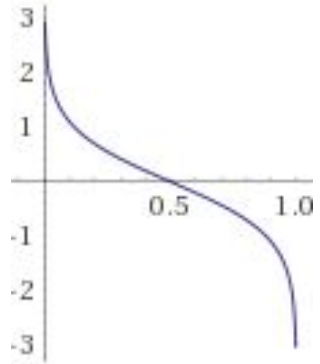
$$\hat{g}_{Bag}(\cdot) = M^{-1} \sum_{k=1}^{M} \hat{g}^{*k}(\cdot)$$

- Boosting
  - weak learner를 생성한 후 Error를 계산.
  - Error에 기여한 Sample마다 다른 가중치를 주고 해당 Error를 감소시키는 새로운 모델 학습.
  - 위 과정을 M번 반복한 후 최종 Boosting model을 다음과 같이 정의
  - 대표적인 모델 : AdaBoost, XGBoost, light GBM
  - 직렬적 방식

$$\hat{g}_{boost}(\cdot) = \sum_{k=1}^{M} c_k \hat{g}_k(\cdot)$$

45-1. AdaBoost

- 기초 분류기 선택 : ID3
- 훈련 데이터 : {(x1, y1), (x2, y2), … , (xn, yn)} 이고, yi = -1 or 1 class
- Algorithm
  1. 샘플링 분포 초기화 : D1(i) = 1/N -> 학습 샘플 데이터를 뽑을 확률
  2. t = 1, 2, 3, …, m 반복
     - Dt 분포에 따라 St = {(x1, y1), (x2, y2), … , (xk, yk)} 샘플링
     - St를 사용해 기초 분류기 ht 훈련
     - ht의 오차율 계산 : et = sum(I[pred_i ≠ yi] * Dt(xi)) / k
       - I[pred_i ≠ yi] : 기초 분류기 ht 에서의 틀리게 예측한 개수
     - ht의 가중치 계산 at = log((1 - et) / et)
       - 오차율이 작아질수록 작게 계산
     - Dt+1 분포 업데이트
       - 틀린 데이터라면 더 높은 가중치 Dt 또는 Dt * (1-et)/et
       - 맞은 데이터라면 더 낮은 가중치 Dt * et/(1 - et)
  3. 데이터 z 예측 : sign(sum_t= 1~m [ht(z) * at])
     - 각 기초분류기에서의 결과 * 각 기초분류기의 가중치 들의 합
     - 양수면 1, 음수면 -1로 판단



45-2. Boosting 정리

- 기초 분류기 선택
- 학습 데이터의 가중치 초기화
- 기초 분류기 학습 후 틀린 데이터에서는 가중치를 증가, 맞은 데이터에서는 감소
- 위 과정을 여러번 반복
- 기초 분류기에 대한 가중 융합하여 출력
- 오차율이 낮은 분류기는 더 큰 힘을 얻는다.

46. 모멘텀, AdaGrad, Adam

- 모멘텀 항 : 일차 모멘텀 (과거 기울기와 현재 기울기의 평균)
- AdaGrad : 이차 모멘텀 (과거 기울기의 제곱과 현재 기울기의 제곱 평균)\
- Adam : 일차, 이차 모멘텀

47. Entropy vs Cross Entropy

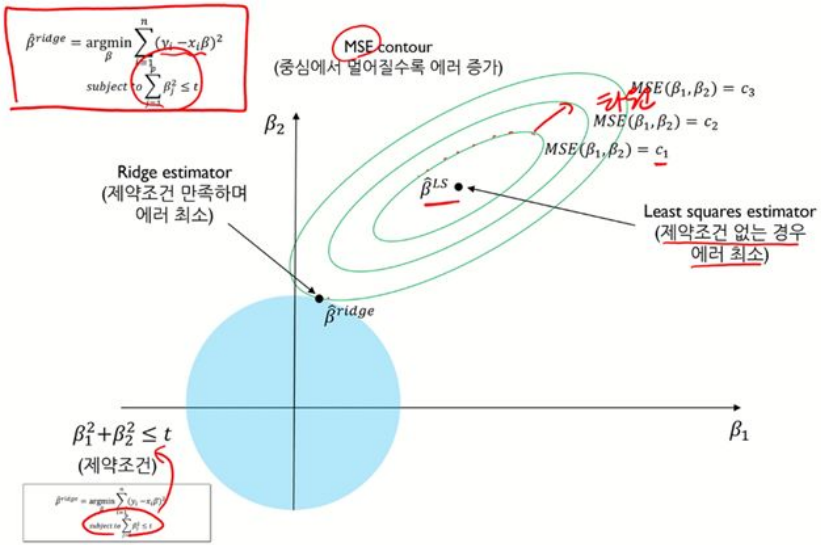
- Entropy 와 Cross Entropy의 차이점
  - P(i) : 실제 확률 (분류된 상태, 정답)
  - Q(i) : 예측 확률 (모델에서 나온 예측 확률)
  - Cross Entropy는 Entropy와는 다르게 예측 확률 값과 실제 확률의 곱으로 표현 된다.

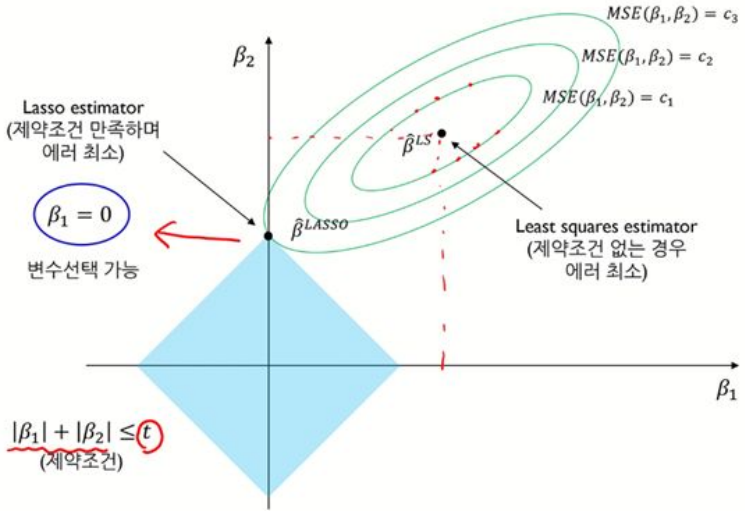
$$H(X) = \sum_{i=1}^{k} log_2 \frac{1}{P_i} * P_i$$

$$H(P,Q) = \sum_{i=1}^{k} log_2 \frac{1}{Q_i} * P_i$$

48. L1, L2 정규화 항

- 정규화 항 : 가중치의 제약의 걸어 가중치의 범위를 줄인다.





Popular Machine Learning Algorithms

https://www.ubuntupit.com/machine-learning-algorithms-for-both-newbies-and-professionals/

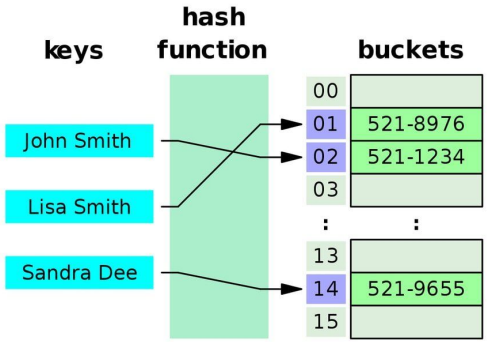
List

1. **Naive Bayes**
2. **Support Vector Machine**
3. **Linear Regression**
4. **Logistic Regression**
5. **K-Nearest Neighbor**
6. **K-means Clustering**
7. **Decision Tree**
8. **Random Forest**
9. CART
10. Apriori Algorithm
11. **Principal Component Analysis**
12. **CatBoost**
13. Iterative Dichotomiser 3
14. Hierarchical Clustering
15. **Back Propagation**
16. **AdaBoost**
17. Deep Learning
18. **Gradient Boosting Algorithm**
19. Hopfield Network
20. C4.5

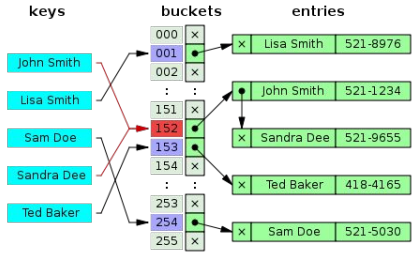
Computer Structures

01. Hash Table

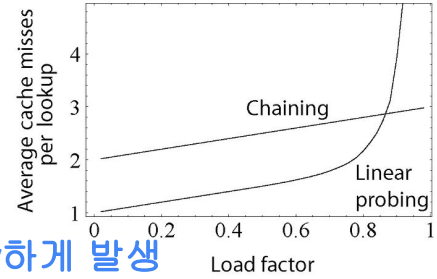
- key를 value에 매핑한 데이터 구조
- **Hash Function** : key값으로 저장되어 있는 주소를 산출하는 함수
- Hash Function을 이용하여 삽입, 검색 속도 빠름
- Python에서는 Dictionary로 구현
- **Hash Collision** 문제 발생 가능성 있음
  - Bucket 은 유한하므로 비둘기집 원리에 의해 중복되는 경우가 발생



02. Hash Collision



- 서로 다른 키를 가진 레코드들이 하나의 버킷에 매핑되는 경우
- **bucket overflow** : Collision 버킷에 충분한 공간이 없어 추가 할 수 없는 상태
- Chaining (Open Hashing, Closed Addressing)
  - 버켓 내에 연결리스트(Linked List)를 할당
  - 해시 충돌 발생 시 연결리스트로 데이터들을 연결하는 방식
  - 데이터의 주소값 바뀌지 않음
  - **장점 : 복잡한 계산 사용 X, 해시테이블이 채워질수록 성능저하가 Linear하게 발생**



- Open Addressing (Closed Hashing)
  - 해시 충돌 발생 시 다른 버켓에 데이터를 삽입하는 방식
  - 선형 탐색(Linear Probing): 해시충돌 시 다음 버켓, 혹은 몇 개를 건너뛰어 데이터를 삽입
  - 제곱 탐색(Quadratic Probing): 해시충돌 시 제곱만큼 건너뛴 버켓에 데이터를 삽입
  - 이중 해싱(Double Hashing): 해시충돌 시 다른 해시함수를 한 번 더 적용한 결과를 이용
  - **장점 : 체이닝처럼 포인터 필요 X, 추가적인 저장공간 X, 삽입,삭제시 오버헤드가 적다. 데이터가 적을 때 더 유리**