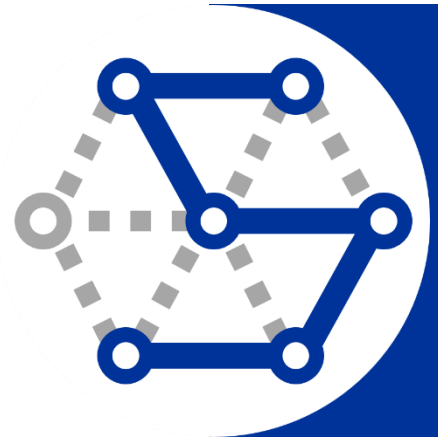
데이터 전처리
Data Preprocessing


06
데이터 축소

목차

데이터 축소Data Reduction

- 방대한 양의 데이터를 대상으로 복잡하게 데이터를 분석하고 마이닝 기법을 적용한다면 매우 많은 시간이 소요되어 분석이 비현실적임 → 데이터 축소 필요
- 데이터 축소 전략
  - 차원적 축소dimensionality reduction: 데이터 인코딩 스키마를 적용하여 압축되거나 축소된 표현 제공
  - 수치적 축소numerosity reduction: 모수적 모형parametric model이나 비모수적 모형non-parametric model을 사용한 데이터 대체
  - 모수적 모형parametric model: 모수의 특성을 활용하는 모형으로 모집단이 정규분포를 띤다는 가정하에 표본 통계량으로 모집단 통계량을 추정
  - 비모수적 모형non-parametric model: 모수의 특성을 활용하지 않는 모형으로 군집화, 표본 추출, 히스토그램 등이 대표적인 예
- 축소된 데이터 집합에 대한 데이터 분석 결과는 원본 데이터 집합에 대한 데이터 분석 결과와 거의 동일한 결과를 산출해야 함

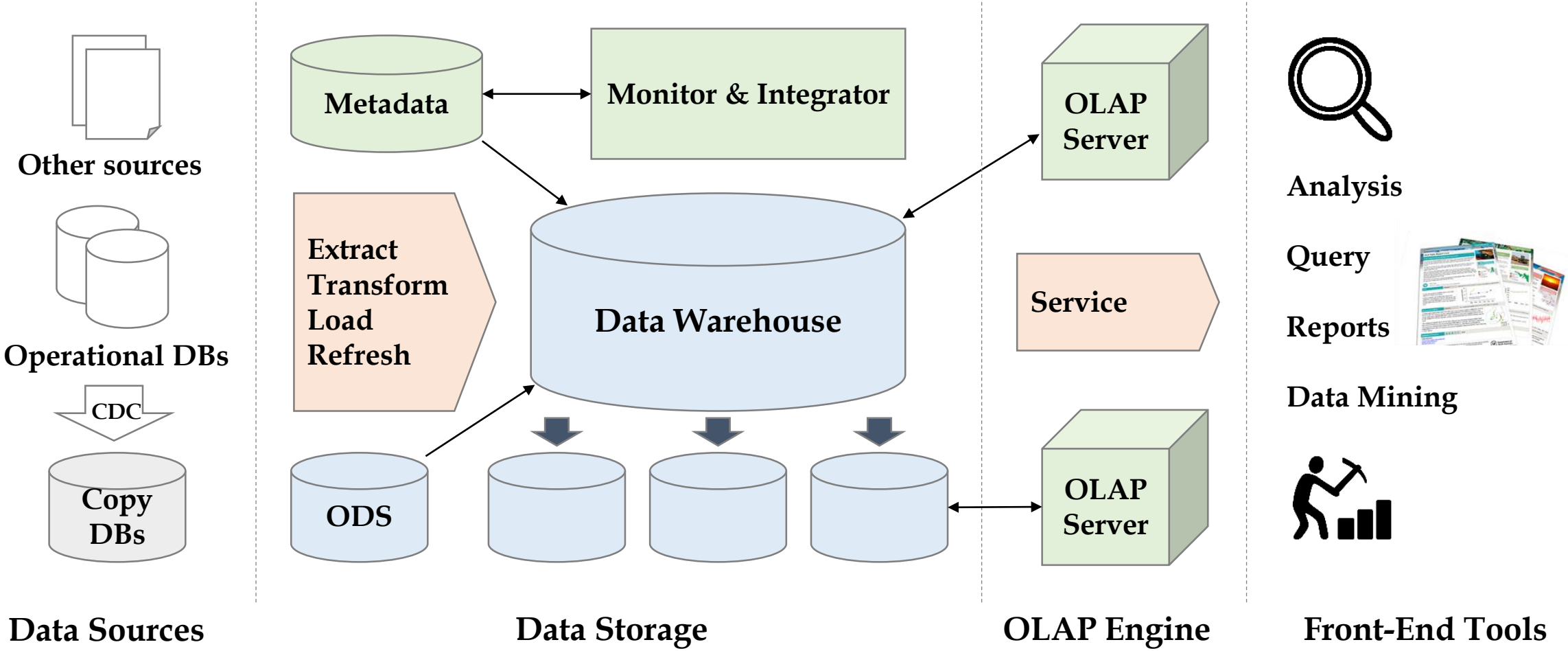


1. 데이터 큐브 집계

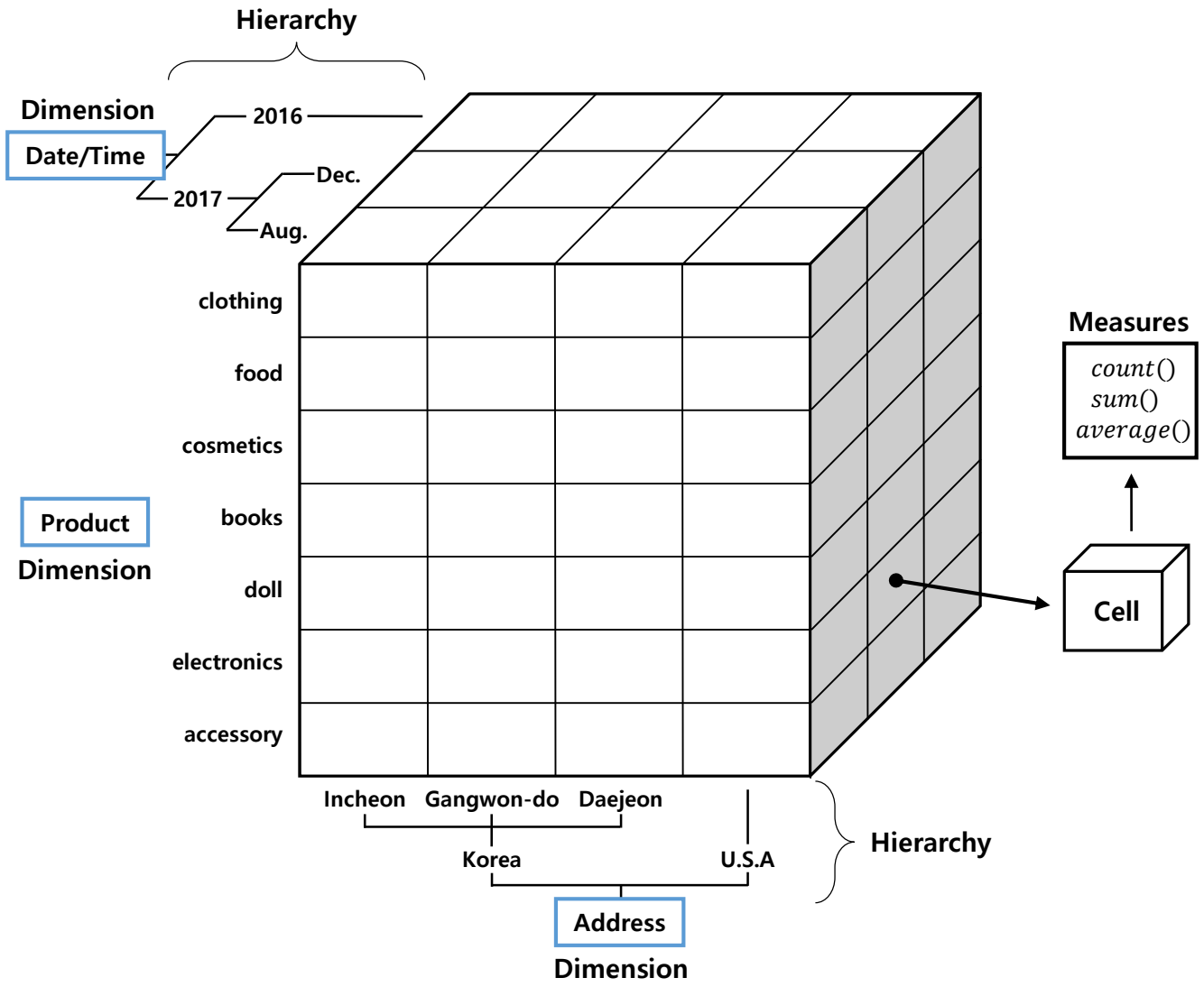
데이터 큐브Data Cube

- 데이터 큐브는 데이터웨어하우스Data Warehouse에서 나오는 용어
- 다차원 집계 정보를 의미
- 데이터 큐브를 구성하는 각 셀은 다차원 공간에서 데이터 포인트에 일치하는 집계된 데이터aggregated data를 가짐
- 원천 데이터를 여러 관점에서 추상화시켜 데이터 축소를 구현
- 데이터 큐브는 사전 계산precomputed되고, 요약된summarized 데이터에 신속히 접근할 수 있도록 하며, 다양한 데이터 분석 처리가 가능

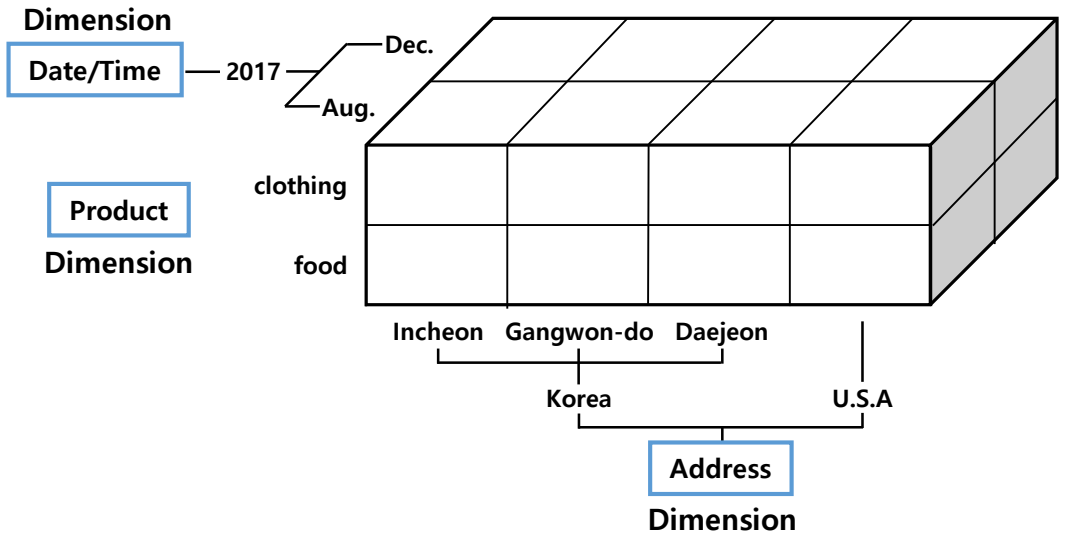
데이터 웨어하우스Data Warehouse



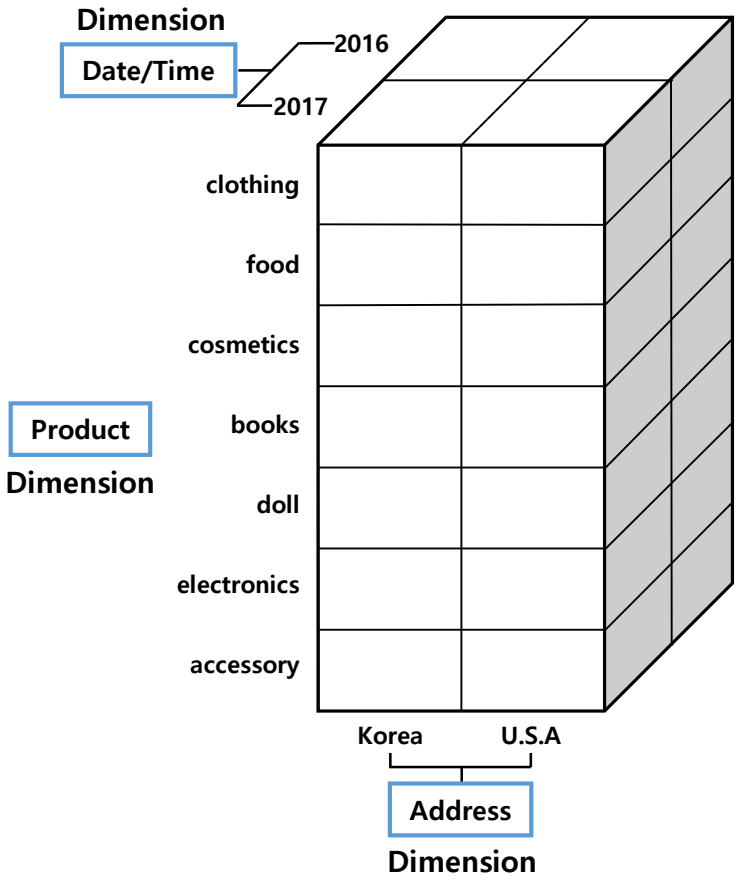
데이터 큐브 예제 (MOLAP)



데이터 큐브 슬라이싱 예제 (MOLAP)



데이터 큐브 슬라이싱 예제 (MOLAP)



데이터 큐브 예제

연도	총판매 수
2009	2560
2008	2210
2007	2100

2009	2560
2008	2210
2007	2100

1차원 데이터 모델과 큐브

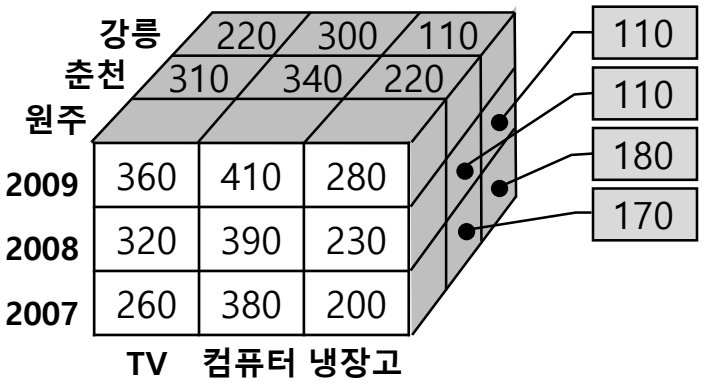
	지역 = "원주"			지역 = "춘천"			지역 = "강릉"		
연도	TV	컴퓨터	냉장고	TV	컴퓨터	냉장고	TV	컴퓨터	냉장고
2009	360	410	280	310	340	220	220	300	110
2008	320	390	230	210	310	180	190	270	110
2007	260	380	200	220	310	170	180	270	110

3차원 데이터 모델

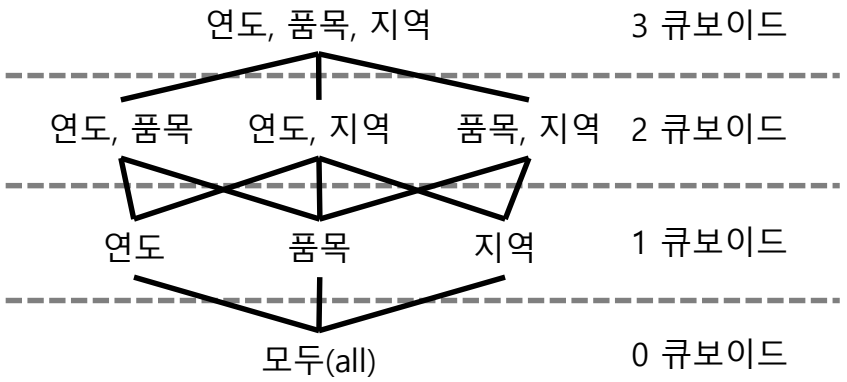
연도	TV	컴퓨터	냉장고
2009	890	1050	620
2008	720	970	520
2007	660	960	480

2009	890	1050	620
2008	720	970	520
2007	660	960	480
	TV	컴퓨터	냉장고

2차원 데이터 모델과 큐브



3차원 데이터 큐브



3차원 큐브 격자

데이터 큐브 예제 (ROLAP)

R

A	B	C	D
a_1	b_1	c_1	d_1
a_1	b_1	c_1	d_2
a_1	b_2	c_1	d_3
a_1	b_2	c_2	d_1
a_1	b_2	c_2	d_2
a_1	b_2	c_2	d_3

A	B	C	D	M
a_1	b_1	c_1	d_1	1
a_1	b_1	c_1	d_2	1
a_1	b_2	c_1	d_3	1
a_1	b_2	c_2	d_1	1
a_1	b_2	c_2	d_2	1
a_1	b_2	c_2	d_3	1

```
SELECT   A, B, C, D, COUNT(M)
FROM     R
CUBE BY  A, B, C, D;
```

```
SELECT   A, B, C, COUNT(M)
FROM     R
GROUP BY A, B, C;
```

cuboid ACD

A	B	C	M
a_1	b_1	c_1	2
a_1	b_2	c_1	1
a_1	b_2	c_2	1
a_1	b_2	c_2	2

A	B	D	M
a_1	b_1	d_1	1
a_1	b_1	d_2	1
a_1	b_2	d_1	1
a_1	b_2	d_2	1
a_1	b_2	d_3	2

A	C	D	M
a_1	c_1	d_1	1
a_1	c_1	d_2	1
a_1	c_1	d_3	1
a_1	c_2	d_1	1
a_1	c_2	d_2	1
a_1	c_2	d_3	1

B	C	D	M
b_1	c_1	d_1	1
b_1	c_1	d_2	1
b_2	c_1	d_3	1
b_2	c_2	d_1	1
b_2	c_2	d_2	1
b_2	c_2	d_3	1

A	B	M
a_1	b_1	3
a_1	b_2	3

cell $\langle a_1, b_2; 3 \rangle$

A	C	M
a_1	c_1	3
a_1	c_2	3

A	D	M
a_1	d_1	2
a_1	d_2	2
a_1	d_3	2

B	C	M
b_1	c_1	2
b_2	c_1	1
b_2	c_2	3

B	D	M
b_1	d_1	1
b_1	d_2	1
b_2	d_1	1
b_2	d_2	1
b_2	d_3	2

C	D	M
c_1	d_1	1
c_1	d_2	1
c_1	d_3	1
c_2	d_1	1
c_2	d_2	1
c_2	d_3	1

A	M
a_1	6

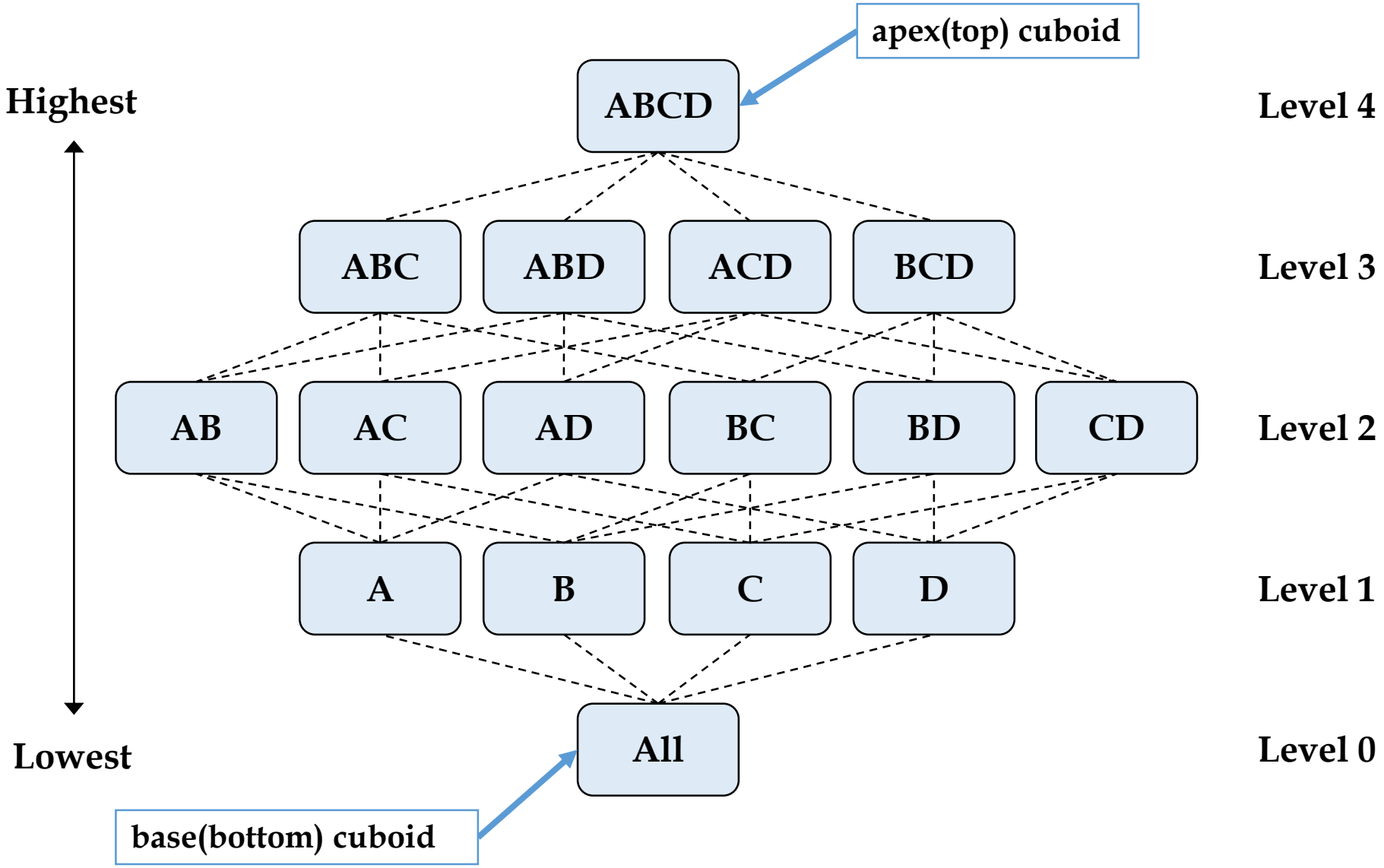
B	M
b_1	3
b_2	3

C	M
c_1	3
c_2	3

D	M
d_1	2
d_2	2
d_3	2

M
6

큐브 격자Cube Lattice



2. 속성 부분집합 선택

속성 부분집합 선택

- 속성 부분집합 선택Attribute Subset Selection은 연관성이 낮거나 중복되는 데이터 속성을 제거하여 데이터 집합의 크기를 줄이는 기법을 의미
- 분석하려는 데이터가 너무 많은 속성을 포함한다면, 데이터 분석 작업의 시간 효율이 떨어질 수 밖에 없음
- 속성 중에서 데이터 분석에 영향을 미치지 않거나 타 속성과 중복적 성격을 가지는 것도 많이 존재
- 너무 많은 속성을 제거하지 않고 분석 작업을 할 경우, 분석 알고리즘에 혼동을 줄 수도 있고, 이로 인해 분석 결과 패턴의 품질에도 악영향을 미칠 수 있음
- 연관성이 낮거나 중복된 데이터 속성(차원)을 제거하여 데이터 집합의 크기를 줄이는 노력이 필요
- 목표는 전체 속성에 가장 가까운 데이터 범주의 확률 분포와 최소의 속성 집합을 찾는 것

최소 속성 집합 찾기

- n개의 속성들에 대해서 조합하면 2^n개의 속성 조합이 가능
- 소모적 탐색법exhausted search
- 경험적 방법heuristic

소모적 탐색법exhausted search

- 가장 쉽게 생각할 수 있는 방법
- 2^n개의 가능한 속성 조합 모두를 탐색
- n이 증가할수록 엄청난 비용이 발생

경험적 방법heuristic

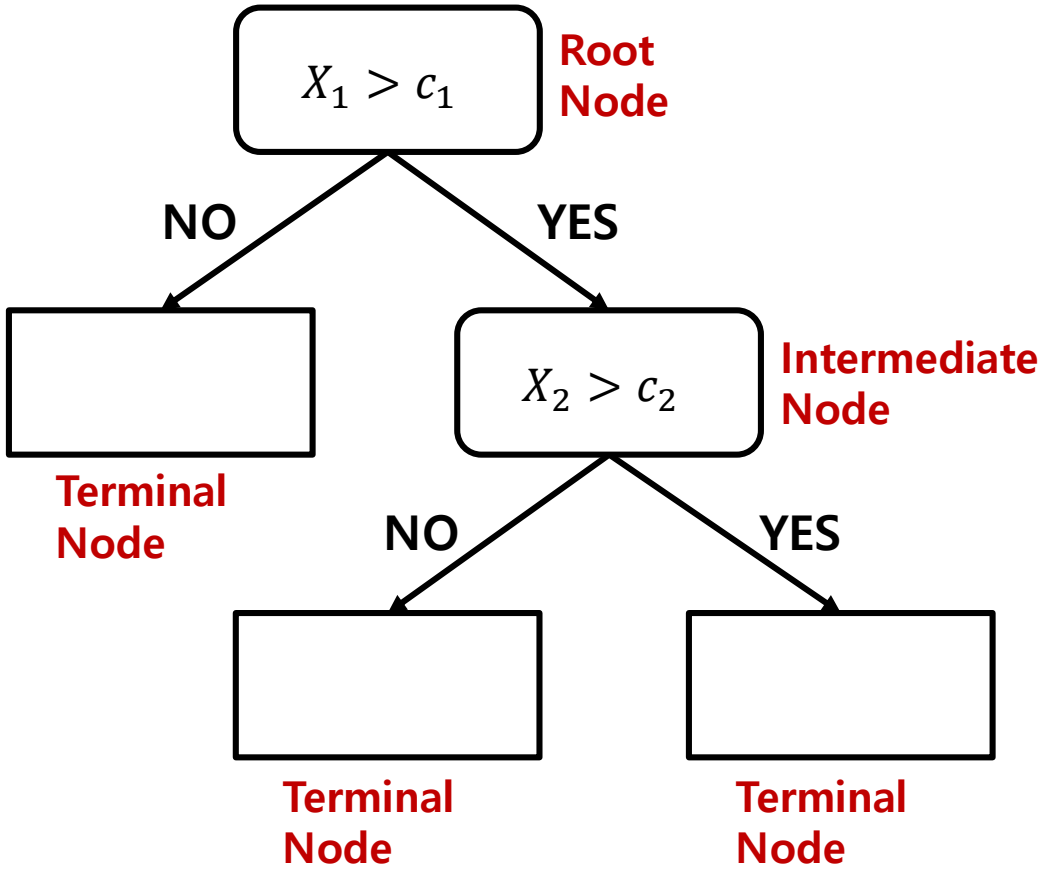
- 일반적으로 많이 사용
- 검색 공간을 축소하여 탐색
- 속성 공간을 탐색하는 동안에 매 회마다 최선으로 보이는 것을 선택 (탐욕적greedy)
- 전역적으로globally 최적이 될 것이라는 기대 속에 지역적으로locally 최적인 해를 선택해가는 방법
- 최적 혹은 최악의 속성들은 그것들이 서로 독립적이라고 가정하는 통계적 유의성statistical significance 검정을 통해여 주로 결정
- 분류classification를 위한 의사결정트리decision tree 생성에 사용되는 정보 이득information gain과 같은 속성 평가 척도를 사용 가능

경험적 기법의 종류

- 단계적 전진 선택법stepwise forward selection: 속성의 공집합으로 시작해서 최적의 속성들을 하나씩 추가하는 방법
- 단계적 후진 제거법stepwise backward elimination: 속성의 전체집합으로 시작해서 최악의 속성들을 하나씩 제거하는 방법
- 전진 선택법과 후진 제거법의 결합combination of forward selection and backward elimination: 전진 선택법과 후진 제거법을 결합하여 각 단계마다 최선의 속성을 선택하고 최악의 선택을 제거하는 방법
- 의사결정트리 귀납법decision tree induction: 의사결정트리는 데이터마이닝 기법 중 분류classification를 위해 고안되었고, 흐름도flowchart와 유사한 구조를 가짐

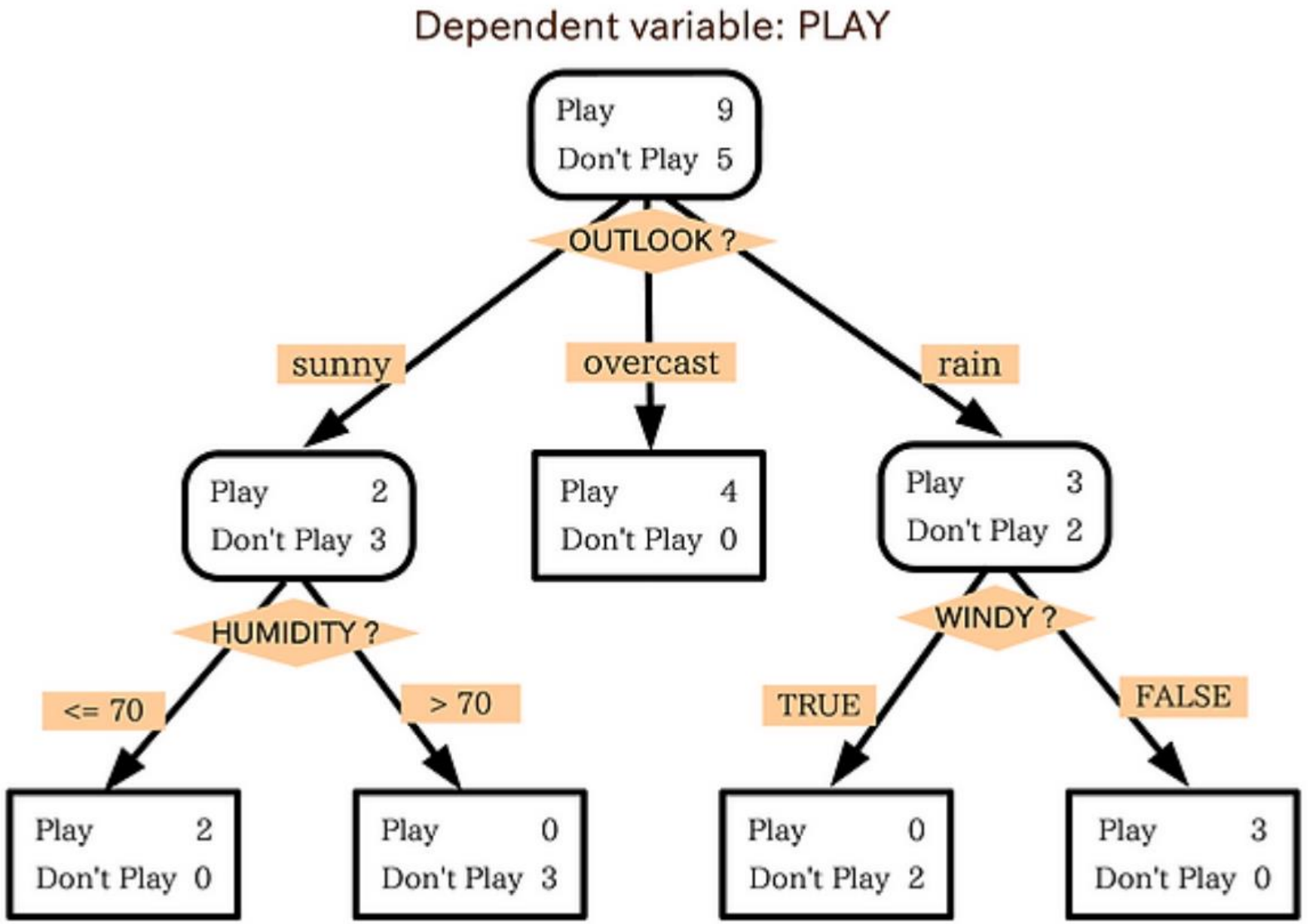
의사결정트리decision tree

- 나무를 뒤집은 것과 같은 모양
- 분기가 거듭되면 데이터의 개수는 줄어듬
- 의사결정트리는 분류classification와 회귀regression 모두 사용 가능
- 범주형 또는 연속형 데이터에 대해서 예측 가능
- Root node: 트리의 초기 지점
- Terminal node의 수가 분리된 집합의 수
- Terminal node를 합하면 전체 데이터의 수와 동일



의사결정트리 예제

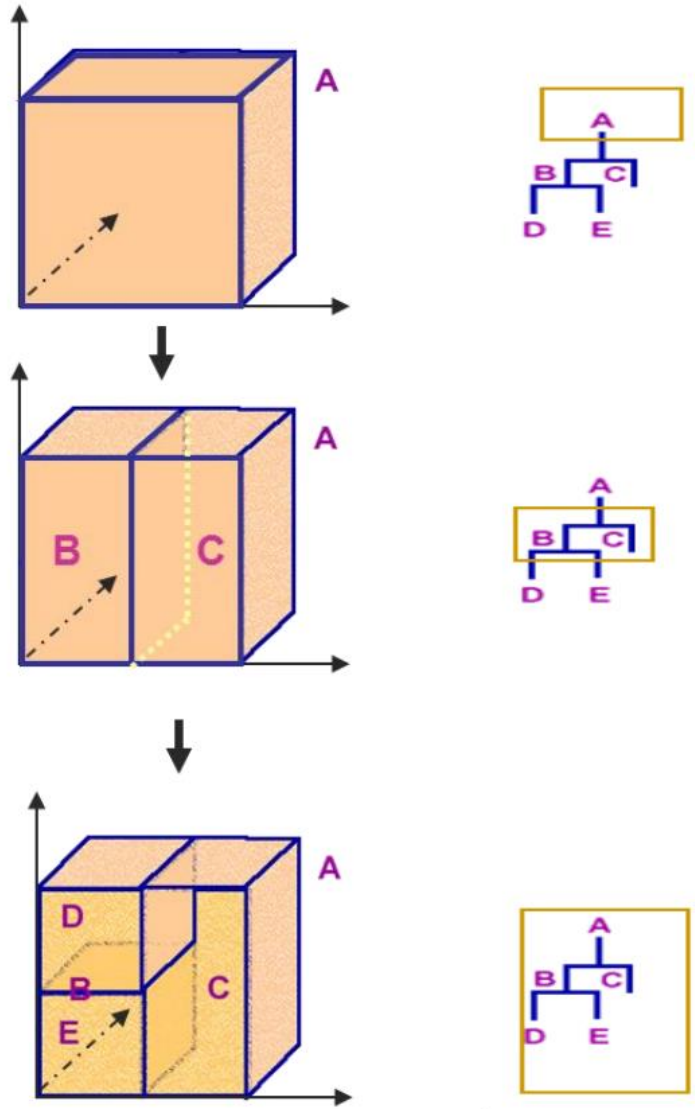
https://ratsgo.github.io/machine%20learning/2017/03/26/tree/



데이터 분할

https://ratsgo.github.io/machine%20learning/2017/03/26/tree/

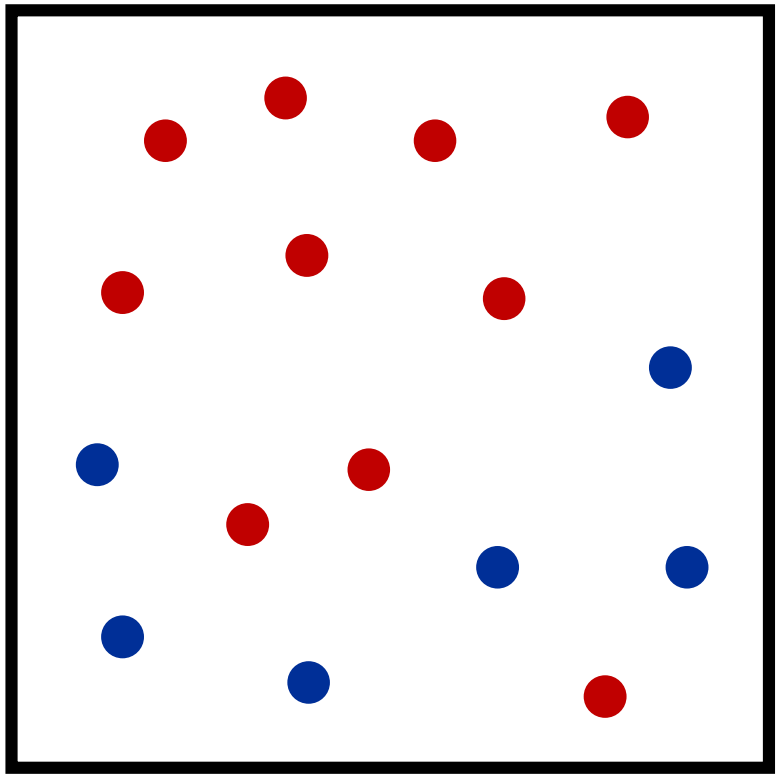
- 의사결정 트리의 분기에 따라 데이터가 분할
- A를 B와 C로 분할
- B는 D와 E로 분할
- Terminal node: C, D, E
- 전체 데이터 A가 부분 집합 C, D, E로 분할



불순도/불확실성

https://ratsgo.github.io/machine%20learning/2017/03/26/tree/

- 타겟 변수 Y가 범주형 변수인 분류나무
- 분류나무는 순도homogeneity가 증가, 불순도impurity 혹은 불확실성uncertainty이 최대한 감소하도록 하는 방향으로 학습
- 순도가 증가/불확실성이 감소하면 정보이론에서는 정보획득information gain이라함



엔트로피entropy

https://ratsgo.github.io/machine%20learning/2017/03/26/tree/

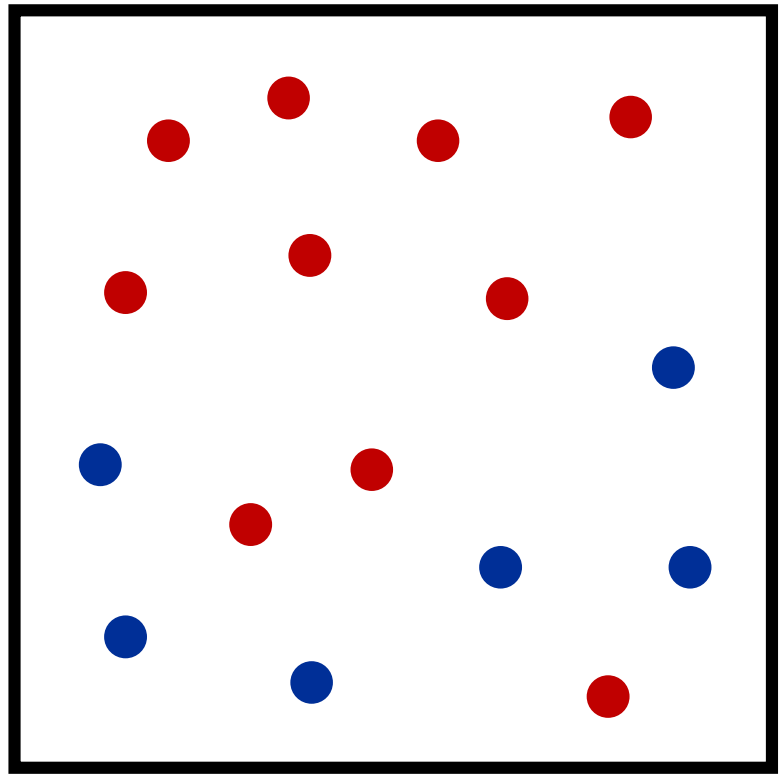
- m개의 레코드가 속한 A 영역에 대한 엔트로피

$$Entropy(A) = -\sum_{k=1}^{m} P_k \log_2(P_k)$$

- 엔트로피 예제

$$Entropy(A) = -\frac{10}{16}\log_2\left(\frac{10}{16}\right) - \frac{6}{16}\log_2\left(\frac{6}{16}\right) \approx 0.95$$

- A 영역에 속한 모든 레코드가 동일한 범주에 속한 경우 엔트로피 0 (불확실성 최소, 순도 최대)
- 반대로 범주가 둘 뿐이고 해당 개체의 수가 동일하게 반반씩 섞여 있을 경우 엔트로피 0.5 (불확실성 최대, 순도 최소)



엔트로피entropy

https://ratsgo.github.io/machine%20learning/2017/03/26/tree/

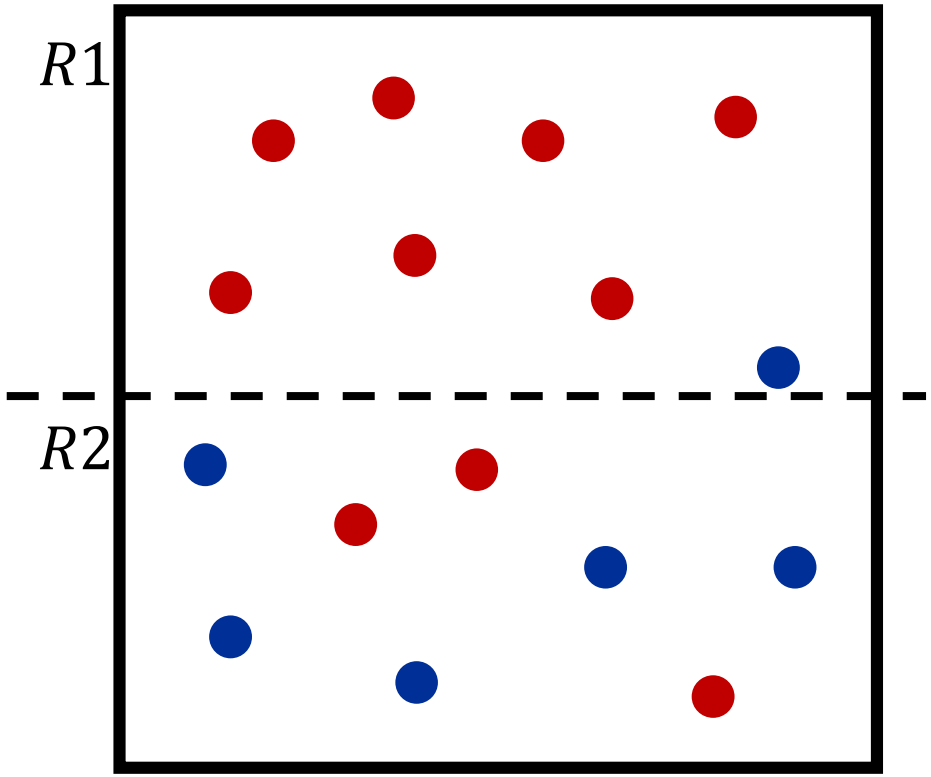
- 전체 영역 A에서 부분집합 $R1, R2$로 분할

$$Entropy(A) = \sum_{i=1}^{d} R_i \left(-\sum_{k=1}^{m} P_k \log_2(P_k) \right)$$

- 엔트로피 계산

$$Entropy(A) = 0.5 \times \left(-\frac{7}{8}\log_2\left(\frac{7}{8}\right) - \frac{1}{8}\log_2\left(\frac{1}{8}\right) \right)$$

$$+0.5 \times \left(-\frac{3}{8}\log_2\left(\frac{3}{8}\right) - \frac{5}{8}\log_2\left(\frac{5}{8}\right) \right) \approx 0.75$$



- 의사결정트리는 구분 한 뒤 각 영역의 순도homogeneity가 증가/불확실성(엔트로피)가 최대한 감소하도록 하는 방향으로 학습

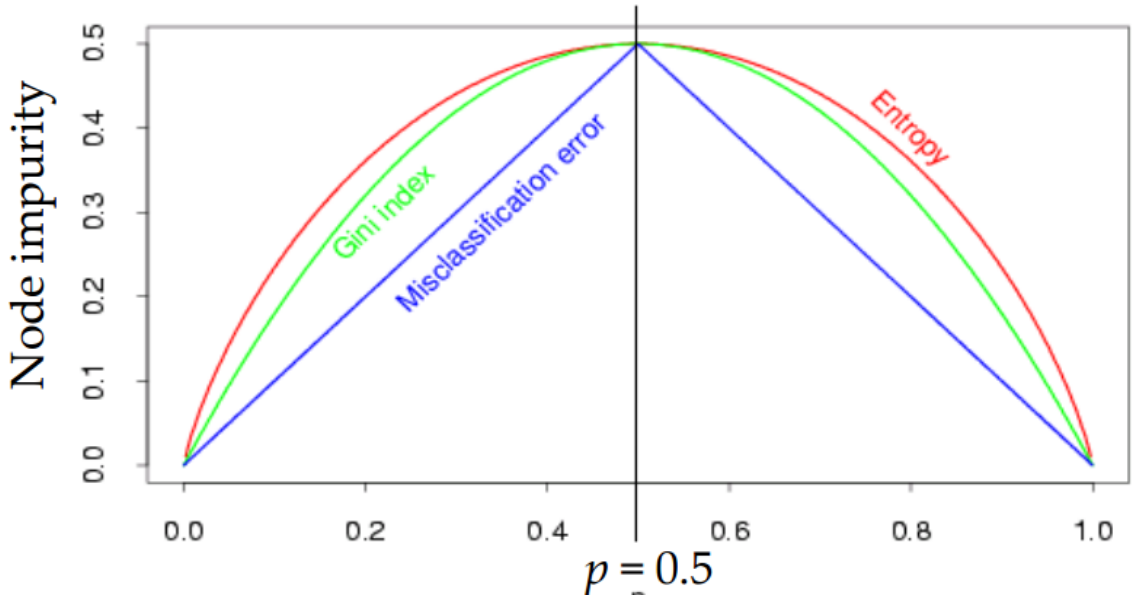
지니 계수Gini Index

https://ratsgo.github.io/machine%20learning/2017/03/26/tree/

- 지니계수 공식

$$G.I(A) = \sum_{i=1}^{d} \left(R_i \left(1 - \sum_{k=1}^{m} p_{ik}^2 \right) \right)$$

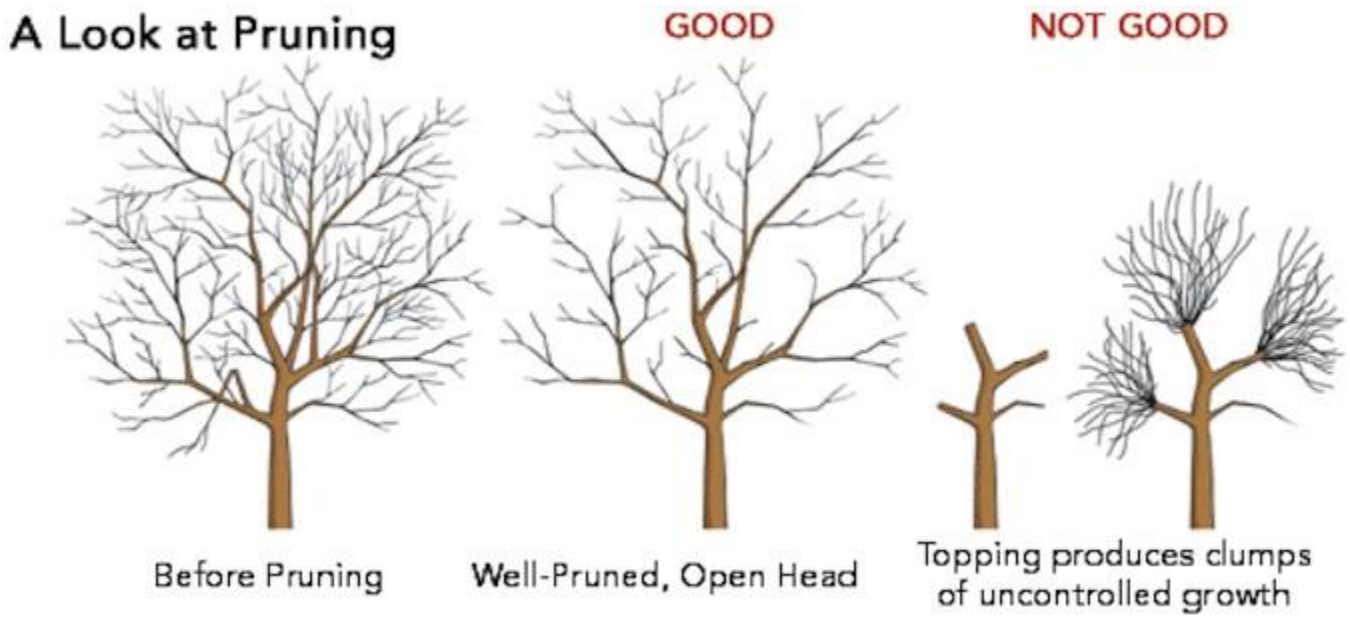
- 오분류오차misclassification error: 불순도 측정이 가능하지만 미분이 불가능하여 자주 쓰이지 않음



의사결정트리 모델 학습

https://ratsgo.github.io/machine%20learning/2017/03/26/tree/

- 재귀적 분기recursive partitioning: 입력 변수 영역을 두개로 구분
- 가지치기pruning: 너무 자세하게 구분된 영역을 통합



https://www.quickcrop.co.uk/blog/wp-content/uploads/2018/03/Tree-Pruning-diagram.jpg

재귀적 분기recursive partitioning

https://ratsgo.github.io/machine%20learning/2017/03/26/tree/

소득	주택크기	잔디깍기기계 여부
60.0	18.4	Owner
85.5	16.8	Owner
64.8	21.6	Owner
61.5	20.8	Owner
87.0	23.6	Owner
110.1	19.2	Owner
108.0	17.6	Owner
82.8	22.4	Owner
69.0	20.0	Owner
93.0	20.8	Owner
51.0	22.0	Owner
81.0	20.0	Owner

소득	주택크기	잔디깍기기계 여부
75.0	19.6	Non-owner
52.8	20.8	Non-owner
64.8	17.2	Non-owner
43.2	20.4	Non-owner
84.0	17.6	Non-owner
49.2	17.6	Non-owner
59.4	16.0	Non-owner
66.0	18.4	Non-owner
47.4	16.4	Non-owner
33.0	18.8	Non-owner
51.0	14.0	Non-owner
63.0	14.8	Non-owner

재귀적 분기recursive partitioning

https://ratsgo.github.io/machine%20learning/2017/03/26/tree/

- 설명변수(X): 소득, 주택크기
- 종속변수(Y): 잔디깍기 기계 구입 여부
- 한 변수(주택 크기)를 기준으로 정렬
- 가능한 모든 분기점에 대해 엔트로피/지니 계수를 구해 분기 전과 비교해 정보 획득을 조사
- 1번 레코드와 나머지 2~24번 레코드 간의 엔트로피 계산 후 분기 전 엔트로피와 비교
- 이후에 1~2번 레코드와 나머지 3~24번 레코드 간의 엔트로피 계산 후 비교
- 다른 변수인 소득을 기준으로 정렬하고 다시 같은 작업 반복
- 모든 경우의 수 가운데 정보 획득이 가장 큰 변수와 그 지점으로 첫 번째 분기를 선택
- 이후 같은 작업을 반복 수행
- 1회 분기에 계산하는 경우의 수: $d(n-1)$, 개체 n개, 변수 d개

재귀적 분기recursive partitioning

https://ratsgo.github.io/machine%20learning/2017/03/26/tree/

소득	주택크기	잔디깍기기계 여부
51.0	14.0	Non-owner
63.0	14.8	Non-owner
59.4	16.0	Non-owner
47.4	16.4	Non-owner
85.5	16.8	Owner
64.8	17.2	Non-owner
108.0	17.6	Owner
84.0	17.6	Non-owner
49.2	17.6	Non-owner
60.0	18.4	Owner
66.0	18.4	Non-owner
33.0	18.8	Non-owner

소득	주택크기	잔디깍기기계 여부
110.1	19.2	Owner
75.0	19.6	Non-owner
69.0	20.0	Owner
81.0	20.0	Owner
43.2	20.4	Non-owner
61.5	20.8	Owner
93.0	20.8	Owner
52.8	20.8	Non-owner
64.8	21.6	Owner
51.0	22.0	Owner
82.8	22.4	Owner
87.0	23.6	Owner

재귀적 분기recursive partitioning

https://ratsgo.github.io/machine%20learning/2017/03/26/tree/

- 분기전 엔트로피

$$= -\frac{1}{2}\log_2\left(\frac{1}{2}\right) - \frac{1}{2}\log_2\left(\frac{1}{2}\right) = 1$$

- 분기후 엔트로피

$$\frac{1}{24}(-\log_2 1) + \frac{23}{24}\left(-\frac{12}{23}\log_2\left(\frac{12}{23}\right) - \frac{11}{23}\log_2\left(\frac{11}{23}\right)\right) \approx 0.96$$

- 분기후 정보획득

$$= 1 - 0.96 = 0.04$$

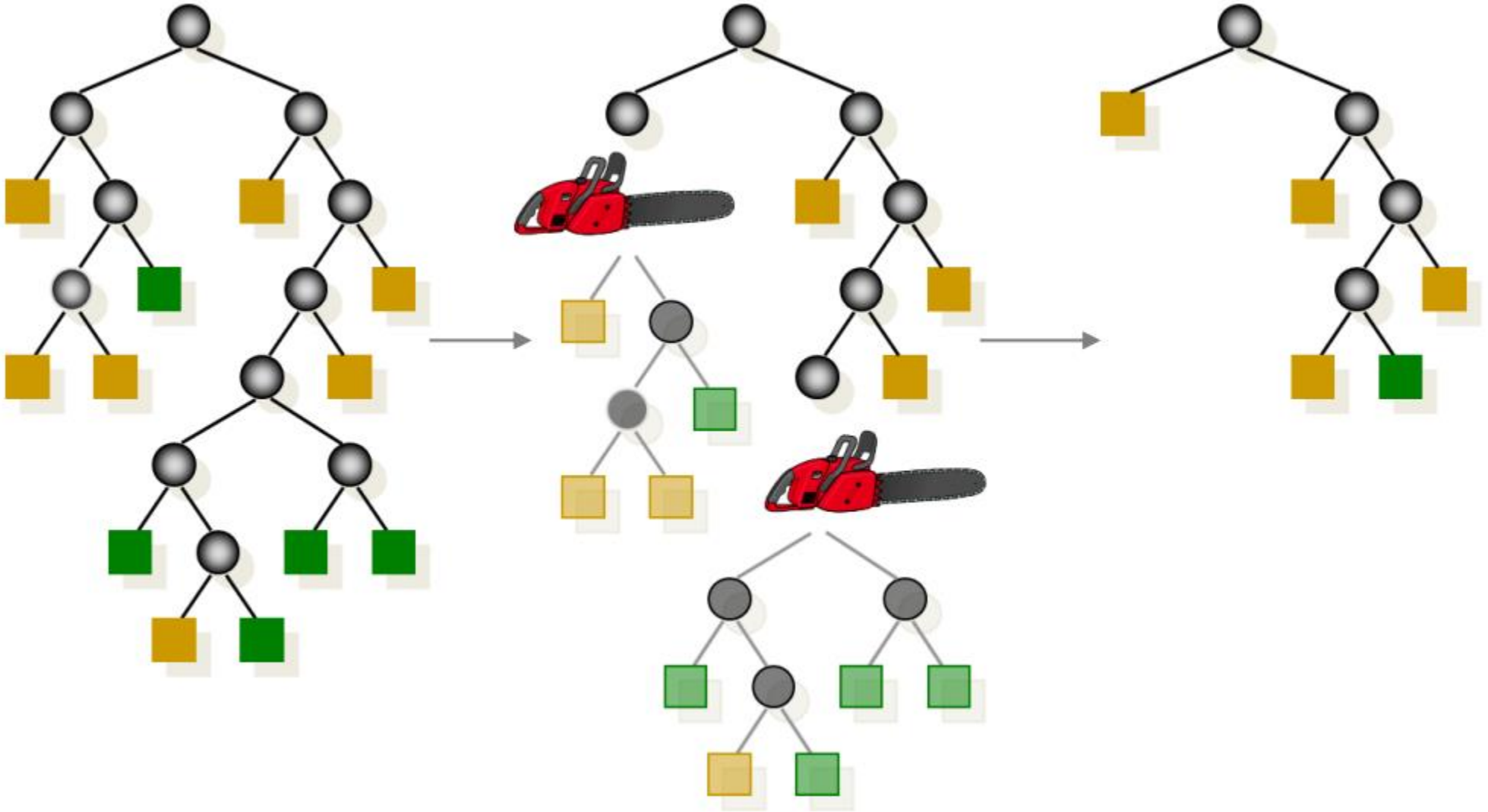
가지치기pruning

https://ratsgo.github.io/machine%20learning/2017/03/26/tree/

- Full tree: 모든 terminal node의 순도가 100%인 상태
- Full tree 생성 후 적절한 수준에서 terminal node를 결합해주어야 함
- Terminal node가 너무 많으면 새로운 데이터에 대한 예측 성능인 일반화generalization 능력이 매우 떨어짐
- 분기가 너무 많으면 학습 데이터에 과적합overfitting 될 수 있음
- 의사결정트리의 분기 수가 증가할 때, 처음에는 새로운 데이터에 대한 오분류율이 감소하나 일정 수준 이상이 되면 우분류율이 증가하는 현상이 발생
- 이러한 문제를 해결하기 위해 검증데이터에 대한 오분류율이 증가하는 시점에 가지치기 수행
- 의사결정트리에서 가지치기는 잘라내는 개념보다 분기를 합치는 merge 개념

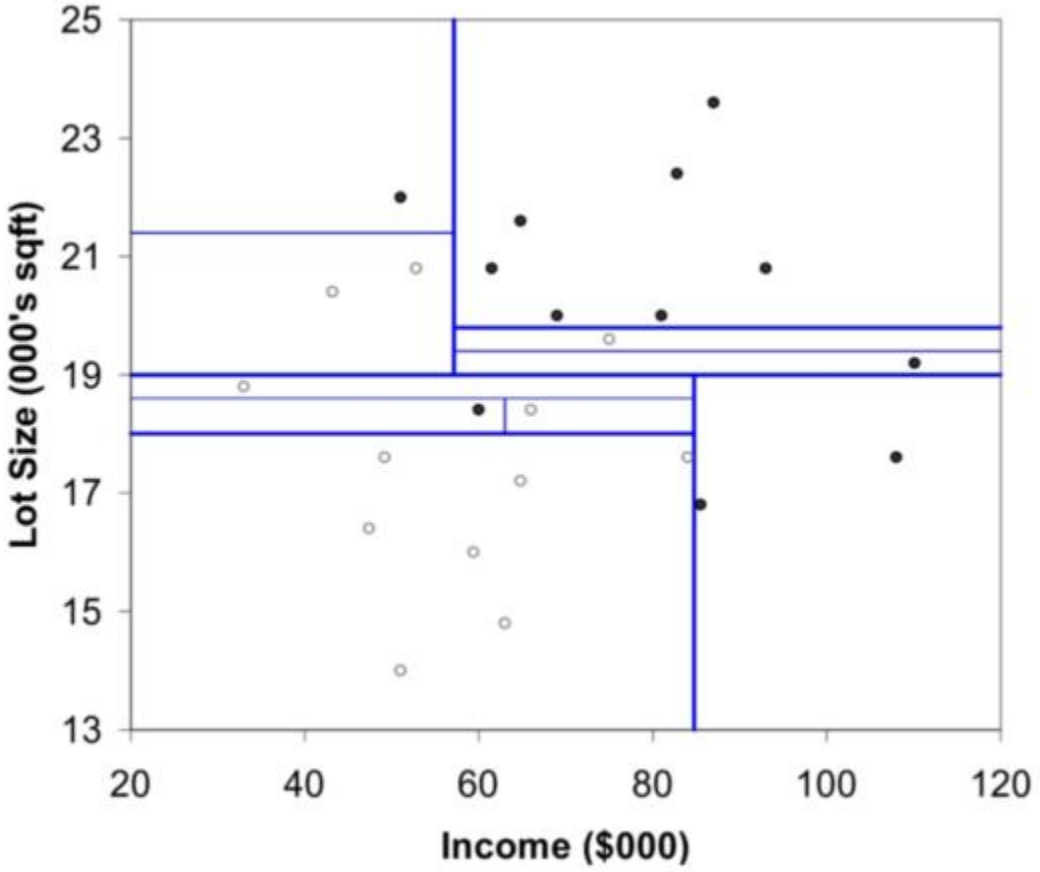
가지치기pruning

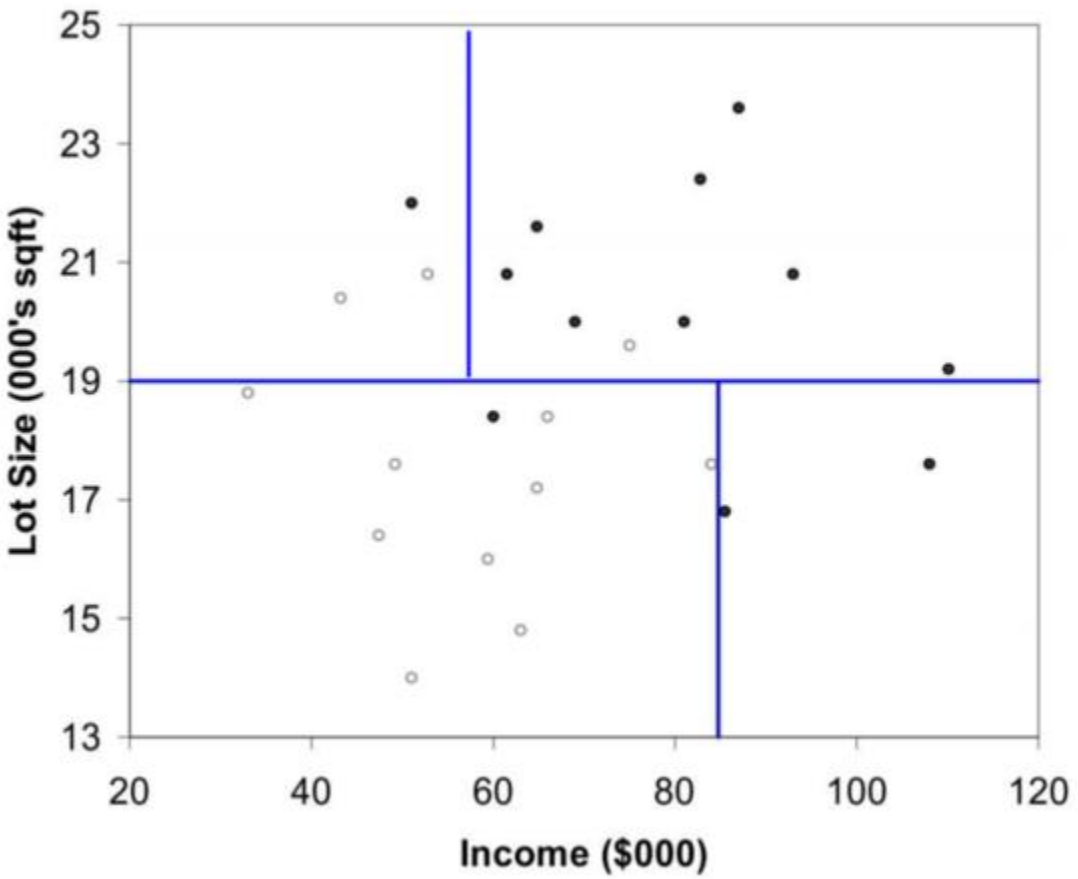
https://ratsgo.github.io/machine%20learning/2017/03/26/tree/



가지치기pruning

https://ratsgo.github.io/machine%20learning/2017/03/26/tree/





가지치기 비용 함수

https://ratsgo.github.io/machine%20learning/2017/03/26/tree/

- 가지치기 비용 함수

$$CC(T) = Err(T) + \alpha \times L(T)$$

- $CC(T)$: 의사결정트리의 비용 복잡도(오류가 적으면서 terminal node 수가 적은 단순한 모델일수록 작은 값)
- $ERR(T)$: 검증데이터에 대한 오분류율
- $L(T)$: terminal node의 수(구조의 복잡도)
- α: $ERR(T)$와 $L(T)$를 결합하는 가중치(사용자에 의해 부여, 보통 0.01~0.1의 값을 사용)

3. 차원 축소

차원 축소dimensionality reduction

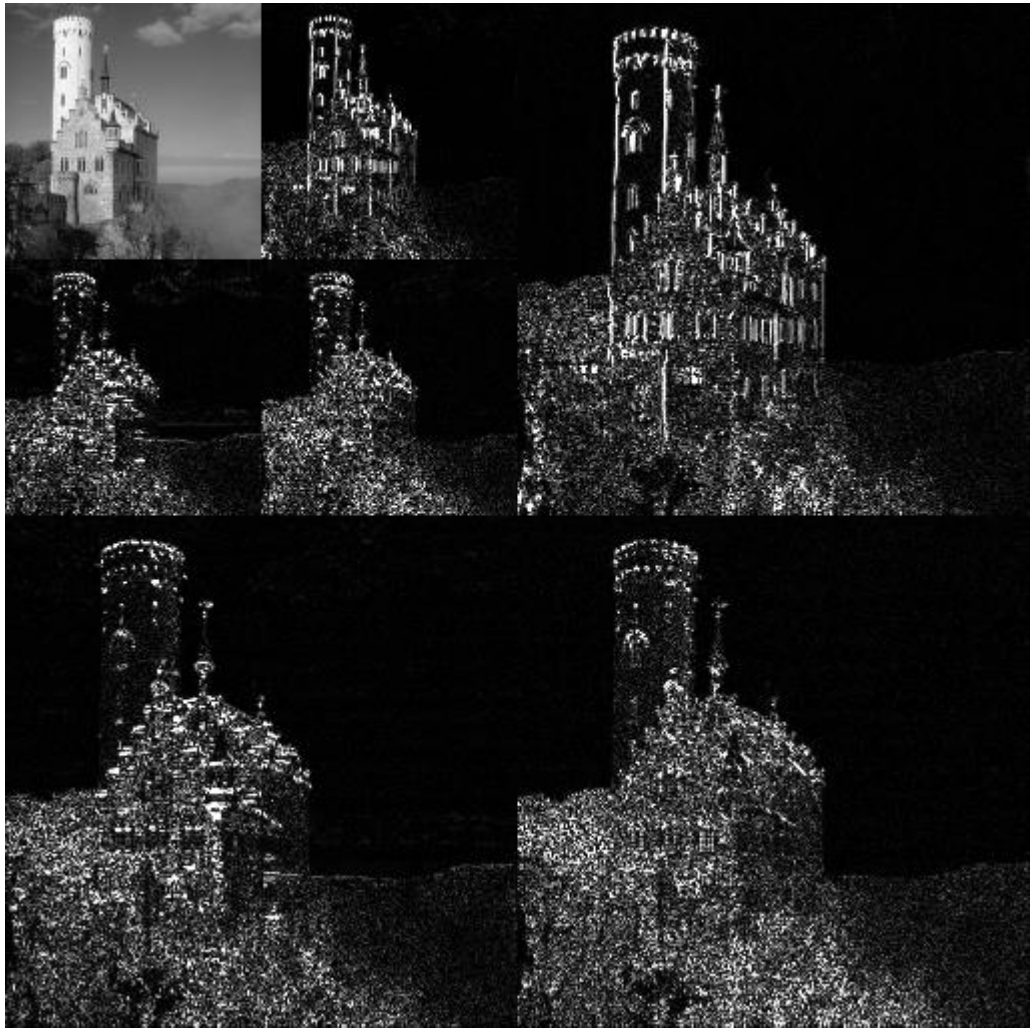
- 원천 데이터의 축소판(압축판)을 얻기 위한 데이터 부호화 또는 데이터 변환의 적용
- 원천 데이터가 정보의 손실 없이 압축된다면 무손실loseless
- 원천 데이터의 근사치만으로 축소된다면 손실lossy
- 일반적으로 많이 사용되며 효과적인 손실 차원 축소 방법
  - 웨이블릿 변환wavelet transform
  - 주성분 분석principal components analysis

웨이블릿 변환wavelet transform

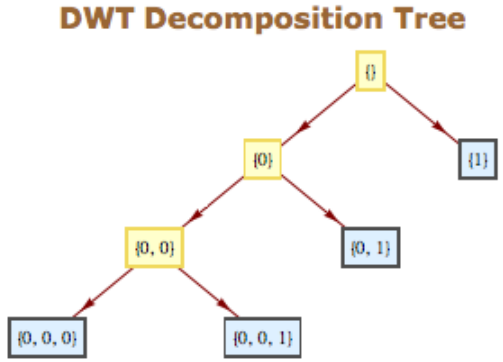
- 이산 웨이블릿 변환discrete wavelet transform, DWT: 데이터 벡터 X를 다른 수치적 벡터numerically vector X'으로 변환 (X와 X'의 길이는 동일)
- 각 튜플을 n 차원 데이터 벡터로 간주하면, 벡터 $X = (x_1, x_2, \cdots, x_n)$를 각 튜플로 고려
- 웨이블릿 변환 데이터가 원천 데이터와 같은 길이(속성 수)를 가지지만 데이터 축소로 볼 수 있는 것은 변환 데이터가 압축되어 보이기 때문
- 웨이블릿 계수 중 가장 유력한 일부만을 저장함으로써 데이터 근사치를 유지
- 예를 들어, 사용자가 정한 어떤 임계값보다 큰 모든 웨이블릿 계수들만 값을 유지하고 나머지 계수들을 0으로 간주하면, 결과적인 데이터 표현은 매우 희소해지며, 데이터 희소성data sparsity은 데이터 연산의 복잡도를 크게 감소시킬 수 있음
- 데이터의 주요 특징들은 보존하면서도 잡음을 제거하는 역할을 하기도 하므로 데이터 정제를 위해서도 효과적임

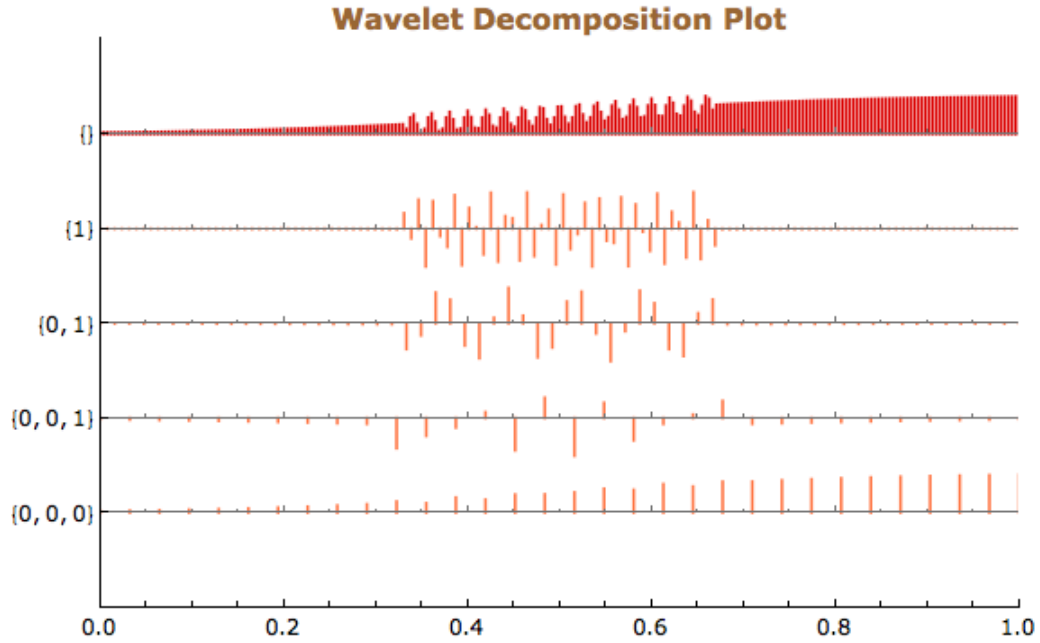
웨이블릿 변환wavelet transform

이현호, Python과 SQL을 활용한 실전 데이터 전처리, 카오스북, 2018.



https://ko.wikipedia.org/wiki/%EC%9B%A8%EC%9D%B4%EB%B8%94%EB%A6%BF_%EB%B3%80%ED%99%98





https://www.wolfram.com/mathematica/new-in-8/wavelet-analysis/discrete-wavelet-transform-(dwt).ko.html

DWT 적용 절차

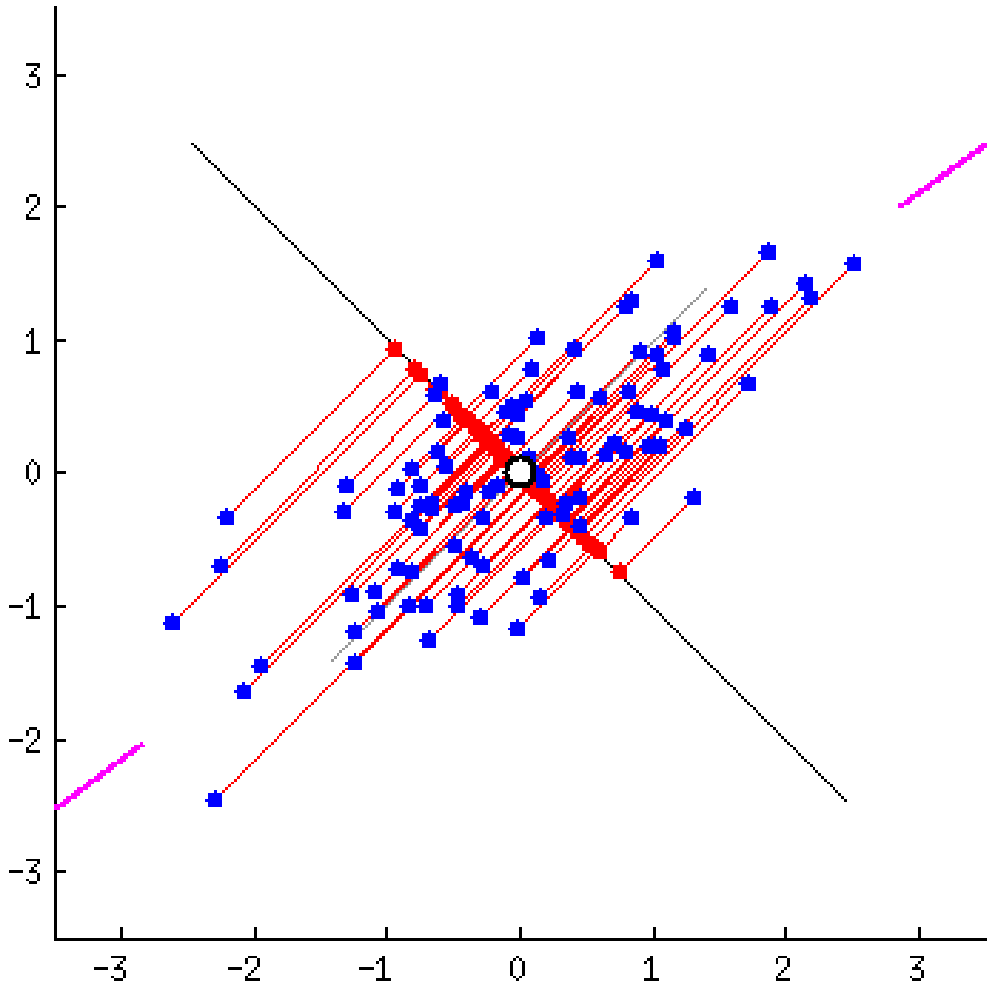
- 각 반복 때마다 데이터를 반으로 나눠서 계산 속도를 향상시키는 피라미드 알고리즘 pyramid algorithm 사용
- 웨이블릿 변환 과정은 웨이블릿 원형 함수를 적용
- 원형 함수를 분석 웨이블릿 혹은 모웨이블릿이라고 하며 두 가지가 존재
  - 고주파 버전: 시계열 분석에 사용
  - 저주파 버전: 빈도 분석에 사용

피라미드 알고리즘pyramid algorithm

1) 입력 벡터의 길이 $L(L \geq n)$을 2의 정수 제곱으로 만들기 위해 필요한 만큼 0을 패딩으로 채움
2) 각 변환에 두 개의 함수를 적용
   - 합이나 가중 평균을 적용한 데이터 평활화smoothing
   - 데이터의 세부적인 특성을 두드러지게 하는 가중차weighted difference 계산
3) 데이터 벡터 X의 두 데이터 포인트 쌍인 $(x2i, x2i+1)$에 두 개의 함수가 적용되고, 그 결과로 길이가 $L/2$인 두 데이터 셋을 생성
   - 빈도 분석에 사용되는 저주파 버전
   - 시계열 분석에 사용되는 고주파 버전
4) 위 과정을 길이 L이 2가 될 때까지 반복
5) 위 반복에서 구해진 데이터 집합으로부터 선택된 값은 변형된 데이터의 웨이블릿 계수로 지정

주성분 분석Principal Components Analysis, PCA

- 주성분 분석Principal Components Analysis, PCA은 n 개의 속성을 가진 튜플(n 차원의 데이터 벡터)에 대하여 데이터를 표현하는데 최적으로 사용될 수 있는 n 차원 직교벡터orthogonal vector들에 대한 k를 찾음 $(k \leq n)$
  → 감소된 차원의 공간을 갖는 데이터 공간 생성 (차원 축소)



https://stats.stackexchange.com/questions/2691/making-sense-of-principal-component-analysis-eigenvectors-eigenvalues

주성분 분석 절차

https://www.potatogim.net/wiki/%EC%A3%BC%EC%84%B1%EB%B6%84_%EB%B6%84%EC%84%9D

1. 입력 데이터를 표준화하여 같은 범위에 속하게 함
(표준화를 통해 큰 범위를 갖는 속성들이 작은 범위를 갖는 속성들을 압도하지 않도록 하기 위함)
2. 표준화된 입력 데이터를 위한 기저base를 제공하는 직교 벡터orthonormal vector들을 계산
이들을 주성분Principal Component라고 하며, 입력 데이터는 주성분의 선형 조합linear combination
3. 주성분은 중요도의 내림차순으로 정렬
주성분은 데이터에 대한 새로운 축의 집합으로서의 역할
(즉, 정렬된 첫 번째 축은 가장 큰 분산을 보여주며, 두 번째 축은 그 다음으로 높은 분산을 보여주는 식)
4. 내림차순 정렬이 되어있어 약한 주성분weak principal component를 제거함으로써 데이터 크기 감소 (즉, 가장 강한 성분들을 사용함으로써 크기가 축소된 원천 데이터의 높은 근사치 구성 가능)

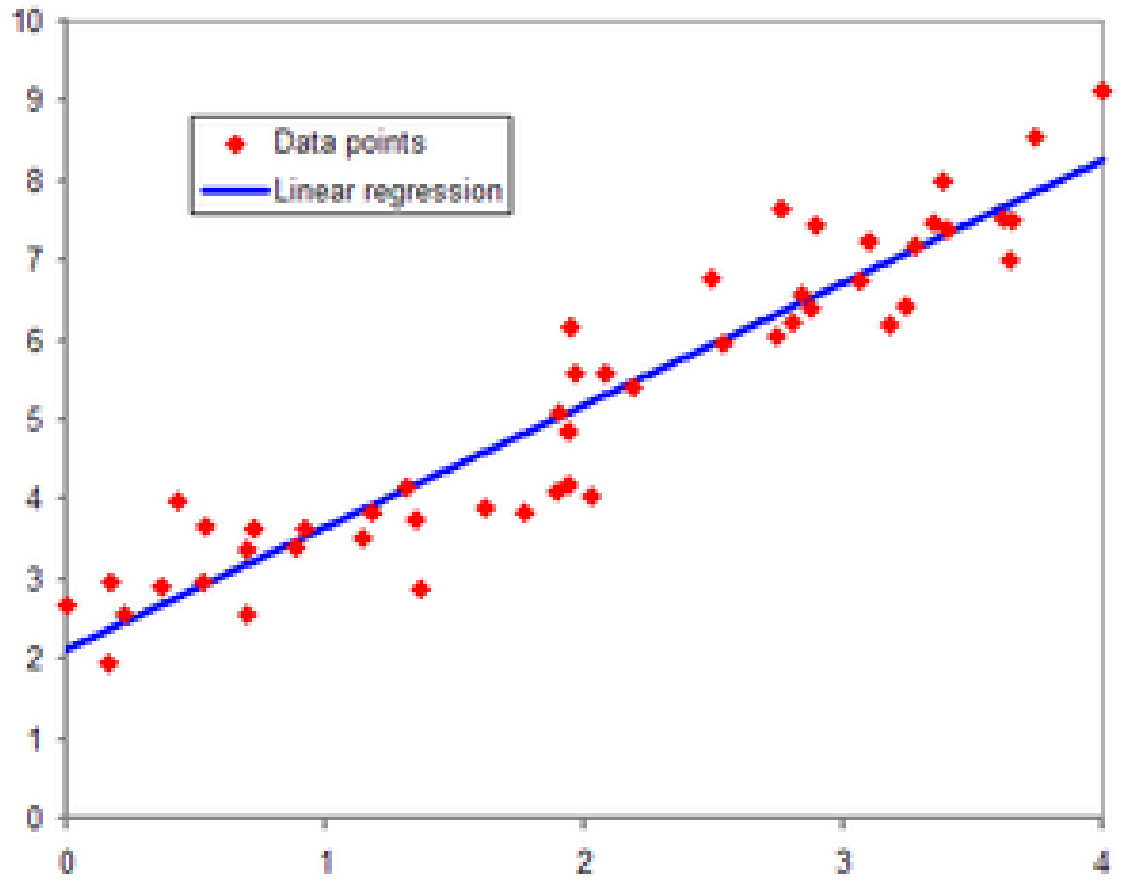
주성분 분석Principal Components Analysis, PCA

- 속성 부분집합 선택은 속성의 초기 집합의 부분집합을 유지하며 속성 집합의 크기를 줄임
- 주성분 분석 PCA
  - 필수적인 속성들의 핵심을 결합
  - 기대하지 않았던 관계를 보여주기도 하여 평범하지 않은 결과 해석이 가능
  - 비용 효과적이며, 순서화된 속성이나 순서화되지 않은 속성에 두루 적용 가능
  - 희소 데이터와 비대칭 데이터에 모두 적용 가능
  - 다차원 데이터를 2차원으로 축소 가능
- 데이터 축소 기법에서는 웨이블릿 변환은 고차원 데이터에 더 적합하고, PCA는 희소 데이터 취급에 더 유리

회귀 모형regression model

이현호, Python과 SQL을 활용한 실전 데이터 전처리, 카오스북, 2018.

- 주어진 데이터의 근사치를 구하는데 사용
- 선형회귀linear regression은 확률변수random variable y를 예측변수predicator variable인 x의 선형함수로 모형화: $y = wx + b$
- 계수coefficient w와 b는 데이터를 분리하는 실제 선과 그 선의 추정치 사이의 오류를 최소화해주는 최소제곱법method of least square에 의해 구할 수 있음
- 다중 회귀multiple regression는 확률변수 y가 두 개 이상의 예측 변수 x에 의해 모형화되도록 단순선형회귀를 확장한 것



https://ko.wikipedia.org/wiki/%ED%9A%8C%EA%B7%80_%EB%B6%84%EC%84%9D

로그-선형 모형log-linear model

- n개의 속성으로 표현되는 n차원에서 주어진 n개의 튜플 집합을 n차원 공간의 한 점으로 생각하는 이산 다차원 확률분포의 근사치를 구함
- 차원 조합의 가장 작은 부분집합에 기반하여 이산화된 속성들의 집합에 대해 다차원 공간 내의 각 점의 확률을 평가하는 데 사용 가능
- 이렇게 함으로써 저차원 공간으로부터 고차원 공간 생성 가능
- 차원 축소와 데이터 평활화에 유용
  - 차원 축소: 저차원의 점은 원래 데이터보다 적은 공간을 차지
  - 데이터 평활화: 저차원 공간의 평가를 응집하는 것은 고차원 공간의 평가보다 표본 변동에 덜 민감
- 회귀 방법이 고차원 데이터에 적용될 경우 계산 비용이 기하급수적으로 늘어남
- 로그-선형모형은 10차원 정도까지는 우수한 확장 가능성scalability를 보여줌

Generalized Linear Models

Model	Equation	Interpretation
Level-Level Regression	$Y = \alpha + \beta X$	One unit change in X leads to β unit change in Y
Log-Linear Regression	$log(Y) = \alpha + \beta X$	One unit change in X leads to $100 * \beta$ percent change in Y
Linear-Log Regression	$Y = \alpha + \beta\, log(X)$	One percent change in X leads to $\beta/100$ unit change in Y
Log-Log Regression	$log(Y) = \alpha + \beta\, log(X)$	One percent change in X leads to β percent change in Y

https://www.kdnuggets.com/2017/10/learn-generalized-linear-models-glm-r.html/2

4. 수량 축소

수량 축소 방법

- 표본 추출sampling
- 히스토그램histogram
- 군집화clustering

표본 추출sampling

- 큰 데이터 집합을 많은 수의 임의 데이터 샘플(부분집합)로 표현 가능
- 대용량 데이터 집합 D가 N개의 튜플을 포함하고 있다고 가정

https://idatassist.com/need-know-random-sampling/

표본 추출sampling 방법

- 비복원 단순 무작위표본Simple Random Sample WithOut Replacement, SRSWOR: D로부터 N개의 튜플 중에서 임의의 s개를 취하는 방법으로서 모든 튜플들의 표본으로 추출될 확률은 같음
- 복원 단순 무작위표본Simple Random Sample With Replacement With Replacement, SRSWR: 각 튜플이 D로부터 추출될 때마다 기록된 후 다시 제자리로 복원replace 된다는 것을 제외하면 SRSWOR와 유사, 각 튜플은 추출된 다음에 다시 추출될 수 있도록 D에 되돌려짐
- 집략표본Cluster Sample: D에 있는 튜플들이 M개의 상호 배반적 군집cluster으로 묶여 있는 가운데 s개의 군집을 단순 무작위로 추출 ($s < M$)
- 층화표본Stratified Sample: D가 층strata이라 불리는 상호 배반적 부분들로 분할되어 있다면, 각 층에서 하나씩 단순 무작위로 추출 (예, 고객의 나이 그룹 각각에 대하여 하나의 층이 생성되어 있는 고객 데이터로부터 층화표본을 얻음)

히스토그램histogram

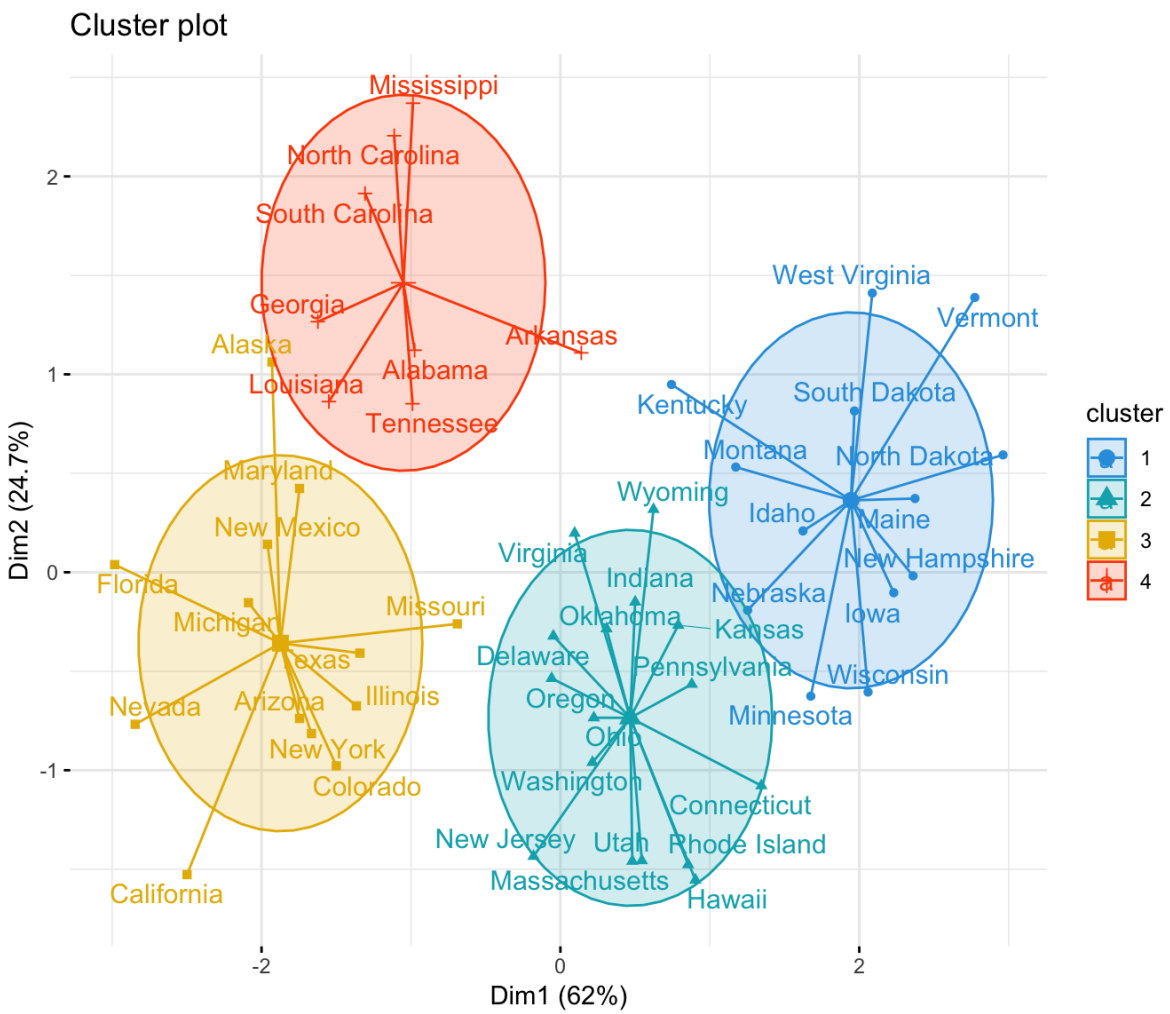
- 구간화를 사용하여 데이터 분포의 근사치를 구하는 데이터 축소의 전형적 형태
- 속성 A의 데이터를 버킷bucket 혹은 빈bin이라 불리는 분리 집합disjoint subset으로 나눔
- 각 버킷이 단일한 속성 값/빈도의 쌍으로 표현되기도 하고, 주어진 속성에 대한 연속 범위continuous range를 나타내기도 함
- 히스토그램은 희소 데이터나 밀집 데이터 모두에 효과적, 비대칭적 데이터와 균일한 데이터 모두 매우 효과적
- 단일 속성에 대한 히스토그램은 다중 속성에 대한 것으로 확장 가능
- 다차원 히스토그램에서는 속성 간의 의존성 포착 가능
- 일반적으로 5개 까지의속성을 가진 데이터의 근사치를 구하는 데 효과적이라고 알려짐

버킷 결정 및 속성 값 분할 방법

- 동등 폭Equal-width: 각 버킷의 범위는 균일
- 동등 빈도Equal-frequency: 각 버킷의 빈도가 일정 (각 버킷이 같은 수의 데이터 표본을 포함)
- V-최적V-optimal: 최소 분산을 갖는 히스토그램을 의미, 히스토그램 분산은 각 버킷이 나타내는 데이터 값들의 가중합weighted sum이며, 버킷 가중치는 버킷에 있는 값들의 개수와 동일
- 최대 차이Max-Diff: 인접한 값들의 각 쌍 사이의 차이를 고려, 사용자 정의 버킷의 수 β에 대하여, $\beta - 1$개의 최대 차이를 갖는 쌍들에 대한 각 쌍 사이에 버킷 경계가 정해짐

군집화clustering

- 데이터 튜플을 객체로 간주하고, 각 객체들을 군집cluster이라는 그룹으로 나눔
- 한 군집 내 객체들과는 유사하면서도 다른 군집 내 객체들과는 유사하지 않도록 군집화
- 유사성은 공간 내에서 객체들이 어떻게 가까운지의 관점에 따라 거리 함수에 기반하여 정의
- 클러스터의 품질은 지름diameter의 표현으로 나타내고, 지름은 클러스터의 두 객체 간 최대 거리로 표현
- 클러스터 간 중심 거리centroid distance는 클러스터 중심 간 거리로서 클러스터 품질로 대체 측정
- 클러스터의 지름은 짧을수록(클러스터 내 객체 간의 유사성이 강할수록), 클러스터 간 중심 거리는 길수록(클러스터 간 유사성은 약할수록) 군집화의 품질이 높다고 볼 수 있음



https://www.datanovia.com/en/lessons/k-means-clustering-in-r-algorith-and-practical-examples/

Q & A