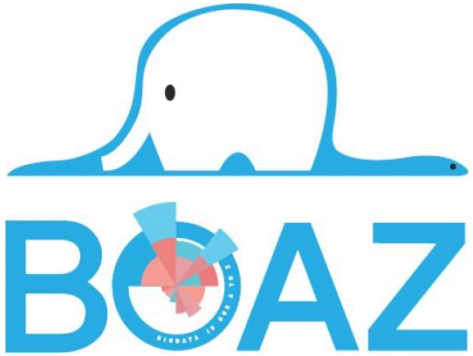


2017.09.28

Decision Tree

의사 결정 나무

발표자 : 9기 박예진

1

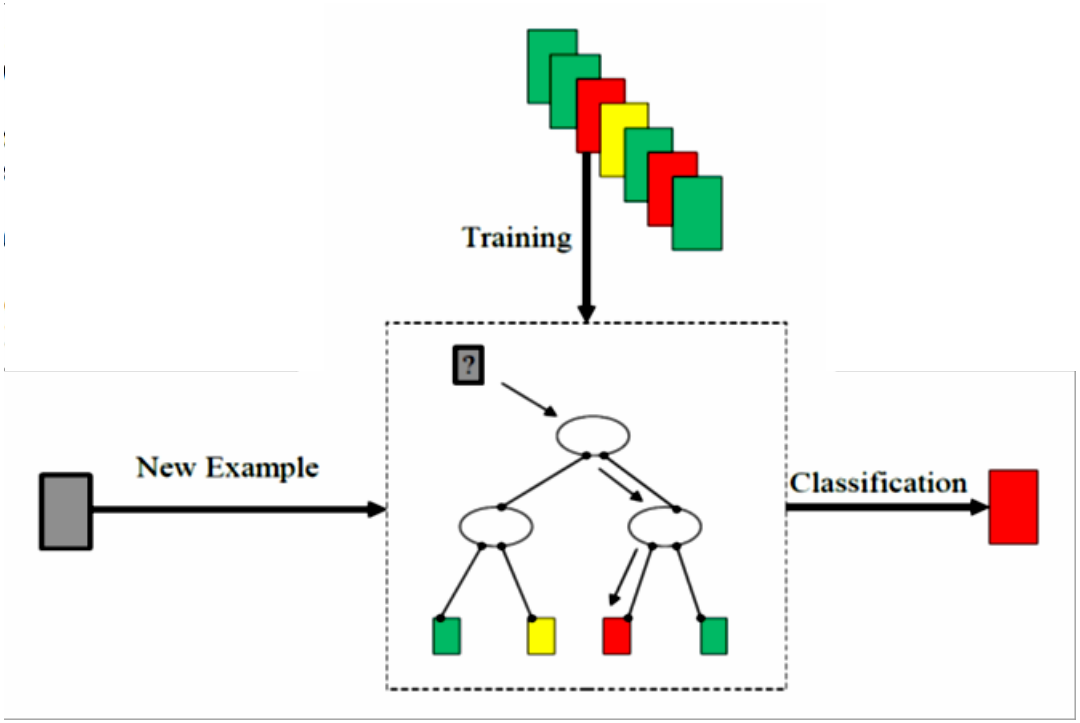
의사 결정 나무란?

1. 의사 결정 나무(Decision Tree)란?

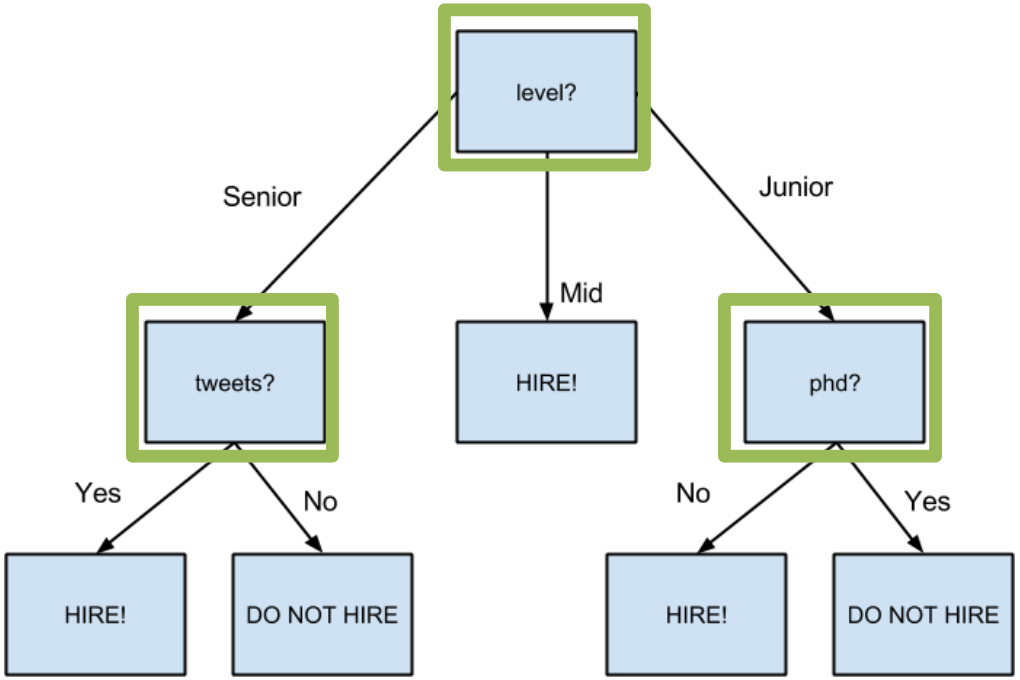
✓ 지도학습(Supervised Learning) : 학습에 사용되는 데이터의 결과가 정해져 있는 경우

✓ 분류문제(Classification)

✓ 비모수적(non-parametric) 방법 : 모집단에 대한 가정이 필요 없음!



1. 의사 결정 나무(Decision Tree)란?

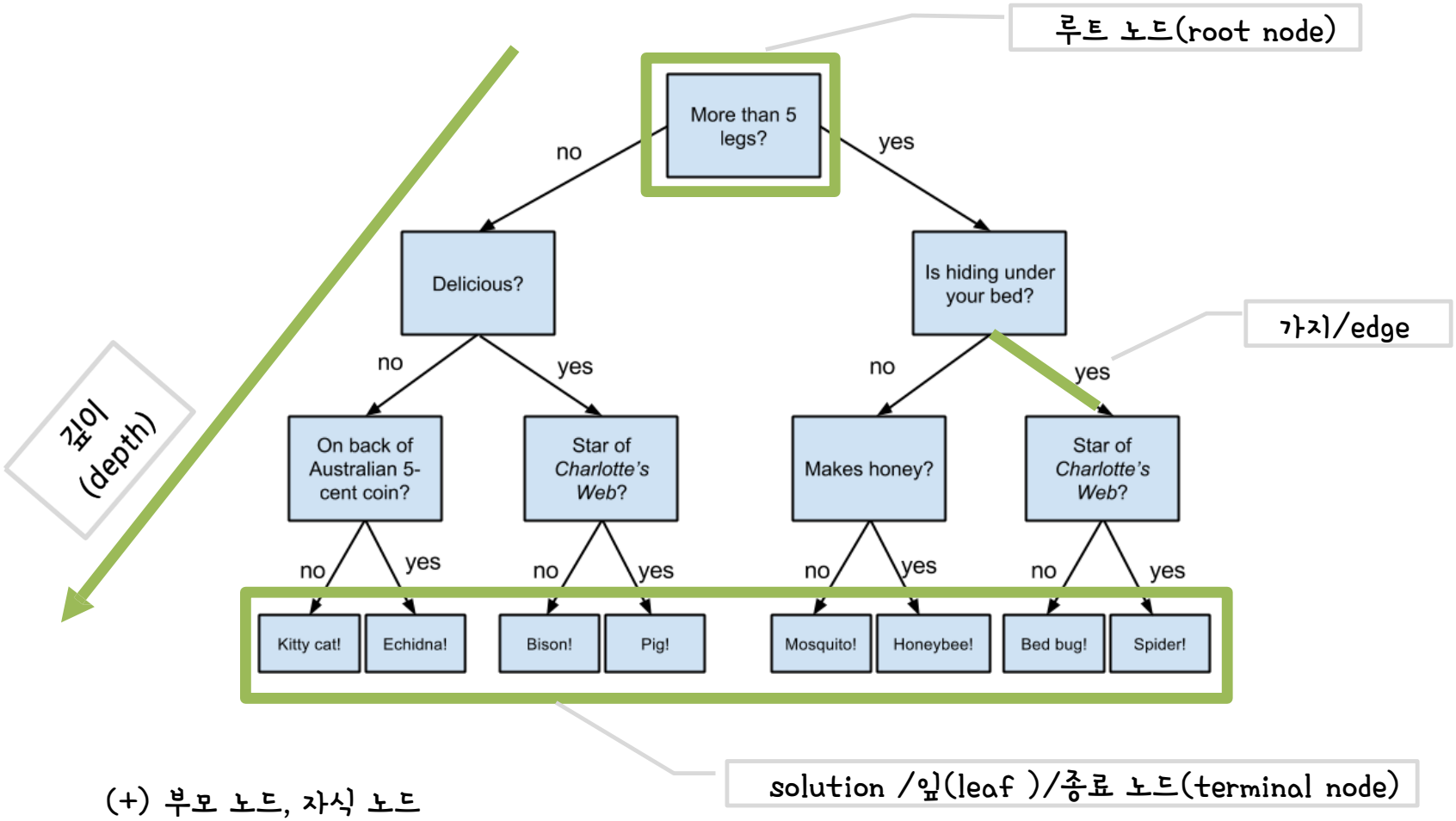


√ 노드(node)

- 데이터셋에 있는 변수의 이름에 해당
- 해당 변수는 언제나 범주형(nominal) 변수여야함

```
inputs = [
    ({'level':'Senior', 'lang':'Java', 'tweets':'no', 'phd':'no'},    False),
    ({'level':'Senior', 'lang':'Java', 'tweets':'no', 'phd':'yes'},   False),
    ({'level':'Mid',    'lang':'Python', 'tweets':'no', 'phd':'no'},   True)
```

1. 의사 결정 나무(Decision Tree)란?



(+) 부모 노드, 자식 노드

2

의사 결정 나무의 종류

CART

- Classification And Regression Tree

2. 의사 결정 나무의 종류

1 Classification Tree
분류나무

√ 일반적으로 의사 결정 나무라고 하면, 분류 나무를 의미함!

√ 타겟: 범주형 변수의 알맞은 부류

√ 규칙 선정 기준: 자식 노드의 purity를 최대한 높게 만드는 것을 택함

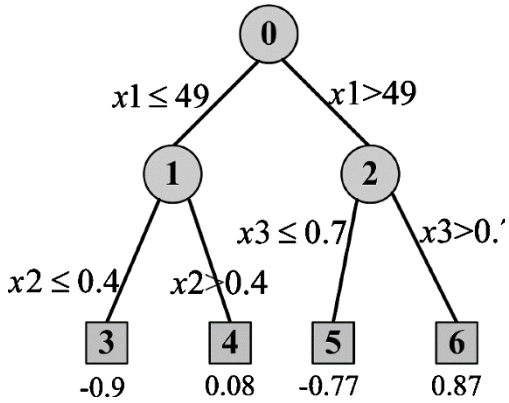
√ 결과 : 범주형으로 입력한 데이터가 분류되는 클래스 레이블을 출력

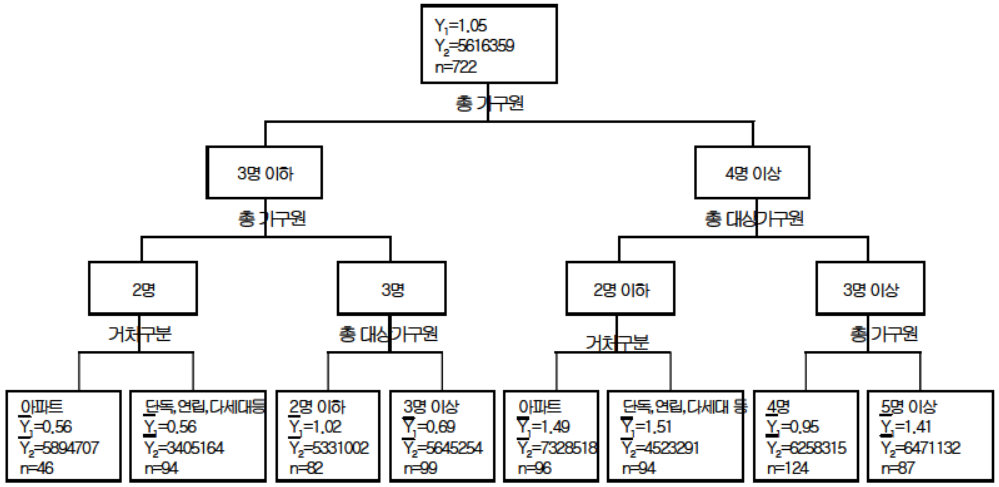
2 Regression Tree 회귀나무

√ 타겟 : 정량적(quantitative, 수량화가가능한)인 변수의 예측

√ 규칙 선정 기준 : 자식 노드의 분산을 최소화하는 것을 택함

√ 결과 : 수치형. 각 잎에 해당하는 값을, 해당 잎으로 분류된 데이터의 예측값으로 부여





3

의사 결정 나무의 분석 단계

3. 의사 결정 나무의 분석 단계

1 의사 결정 나무의 형성

- 분석할 데이터에 따라 적절한 알고리즘, 분리기준, 정지규칙을 지정

2 가지치기(Pruning)

- 과적합(overfitting) 발생시,
혹은 분류의 오류를 크게 할 위험이 높거나
부적절한 추론 규칙을 가지고 있는 가지를 제거

3 타당성 평가

4 해석 및 예측

4

분리(분할) 기준
(Splitting Criteria)

4. 분리(분할) 기준 (Splitting Criteria)

√ 노드를 나누는 기준이자, 가지를 분리하는 방식

(분리에 사용될 변수를 선택하는 기준)

√ 엔트로피(Entropy), 정보이득(Information gain),

지니계수(Gini Index), 카이제곱통계량(χ^2) 등

4. 분리(분할) 기준 (Splitting Criteria)

1 엔트로피지수 Entropy Index

√ Entropy(x) : x에 대한 정보를 나타내기 위해 평균적으로 알아야 할 비트(bit)의 수.
즉, 정보의 기댓값

√ 불확실성(uncertainty), 불순도(impurity), 무질서도(disorder)를 나타냄

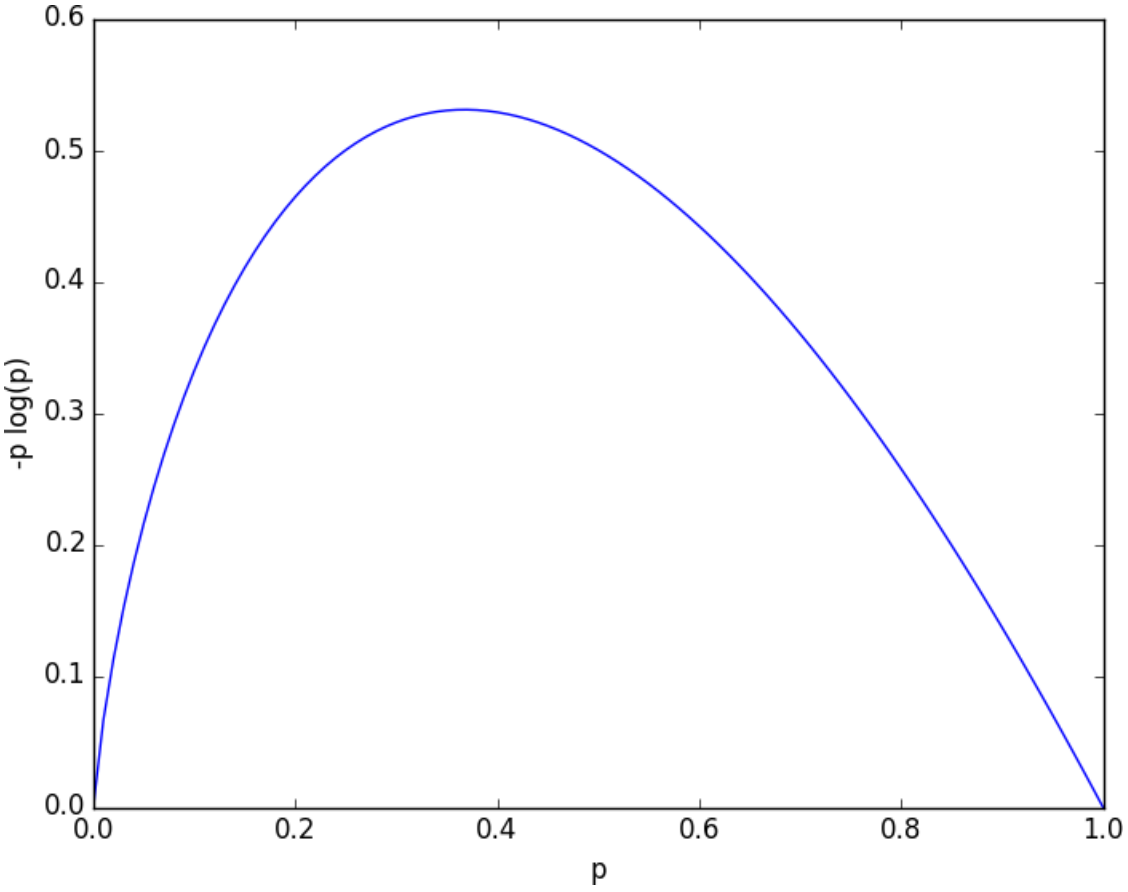
√ 엔트로피가 높다 = 불순도가 높다 = 데이터가 고르게 분포한다.
엔트로피가 낮다 = 불순도가 낮다 = 대부분의 데이터가 몰려있다.

√ 0 ≤Entropy ≤1

- Entropy = 1 : 모든 데이터가 각 클래스에 동일한 수 만큼 들어 있는 경우
- Entropy = 0 : 모든 데이터가 하나의 클래스에 속할 경우

4. 분리(분할) 기준 (Splitting Criteria)

1 엔트로피지수 Entropy Index



2 정보이득 Information Gain

✓ 가지를 칠 때, 정보 이득을 가장 높게 만드는 변수(조건)를 다음 노드로 삼는다.

✓ 정보가 Before 상태에서 After 상태로 변화하였다면,

$$\text{정보이득}\,\Delta = \text{Entropy(Before)} - \text{Entropy(After)}$$

4. 분리(분할) 기준 (Splitting Criteria)

3 지니지수 Gini Index

√ 불순도(impurity)를 측정

$0 \leq$ 지니지수 $\leq 1 - \frac{1}{k}$ (k: 범주형 변수의 수준 수)

- 지니지수 $= 1 - \frac{1}{k}$: 모든 데이터가 각 클래스에 동일한 수만큼 들어있는 경우
- 지니지수 $= 0$: 모든 데이터가 하나의 클래스에 속할 경우

지니지수의 계산

$$Gini = \sum_{i \neq j} p(i)p(j)$$

4 카이제곱 통계량 $\chi 2$ Chi-squared statistic

√ 통계학에서 카이제곱 독립성 검정에 쓰이는 통계량과 동일

→ p-값이 가장 작은 변수를 다음 노드로 지정한다.

5

사용 알고리즘

5. 사용 알고리즘

- ID3 (IterativeDichotomiser3)
- C4.5
- C5.0

머신러닝 / AI분야에서 개발

- CART(ClassificationandRegressionTree)
- CHAID(Chi-squaredAutomatic Interaction Detector)

통계학분야에서 개발

- QUEST(Quick, Unbiased,Efficient, Statistical Tree)
- MARS (Multivariate Adaptive RegressionSplines)
- 조건부추론트리(ConditionalInferenceTrees)
- etc…

5. 사용 알고리즘

1 ID3
Iterative Dichotomiser 3

✓ 범주형(categorical) 속성에만 적용 가능

✓ 분리(분할) 기준: 엔트로피(Entropy), 정보이득(Infogain)

✓ 가지를뻗는방법: 다지분리(multiwaysplit)

5. 사용 알고리즘

1 ID3
Iterative Dichotomiser 3

√ 알고리즘작동

1. 전체 데이터를 포함하는 루트 노드(루트 노트는 가장 많은 정보를 제공)를 생성한다.

2. 루트 노드에서 특성을 평가(test)한 후 자식 노드를 생성한다.

3. 그 다음으로 정보이득이 높은(=데이터들을 가장 잘 구분할 수 있는) 속성을 선택하여 가지(branch)를 뻗어 하위 노드를 생성한다.

4. 정지 조건을 만족할 때까지 3의 과정을 반복하고, 만약 정지조건을 만족했다면 최종적으로 생성된 노드는 잎이 되고, 해당 노드에 분류된 샘플들에 해당 노드의 레이블을 부여한다.

1 ID3
Iterative Dichotomiser 3

√ 한계점

1. 수치형(numeric) 속성이나 결측값(missing value)은 다루지 못한다.

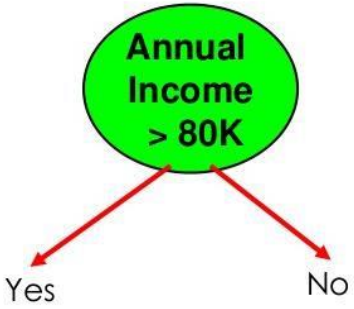
2. 범주형 속성의 모든 수준에 대해 하위 노드를 생성하기 때문에,
수준이 다양할 경우 하위 노드의 가지 수가 매우 많아져
과적합(overfitting)의 문제가 발생할 수 있다.

2 C4.5
ID3의 한계점을 보완한 알고리즘

✓ 범주형(categorical)과 수치형(numeric) 속성 모두에 적용 가능

✓ 분리(분할) 기준: 엔트로피(Entropy), 정보이득(Infogain)

✓ 가지를 뻗는 방법: 다지 분리(multiwaysplit)와 이진 분리(binarysplit)



2 C4.5
ID3의 한계점을 보완한 알고리즘

√ ID3의 한계점을 보완

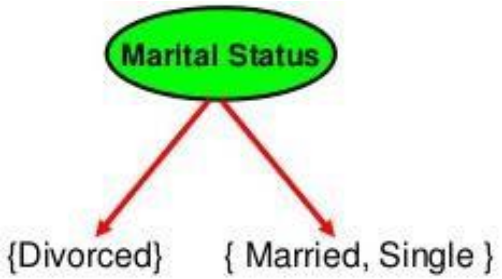
1. 범주형 속성만 취급하는 ID3와 달리, 수치형 속성까지 취급할 수 있다.
2. 결측값(missingvalue)의 문제를 해결할 수 있다.
3. 가지치기(Pruning)가 가능하므로 과적합 문제의 해결이 가능하다.

3 CART
Classification And Regression Tree

√ 범주형(categorical)과 수치형(numeric) 속성 모두에 적용 가능

√ 분리(분할) 기준: 지니지수(GiniIndex), 분산의 차이

√ 가지를 뻗는 방법 : 이진분리(binarysplit)



√ 회귀나무를 만들 수 있는 알고리즘

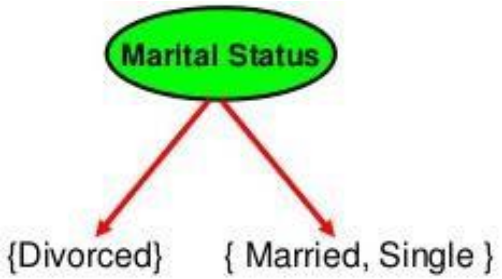
√ 후보나무를 여러 개 생성하고,

그 중에서 최적의 나무를 찾아내는 방법을 사용한다. (최대강점)

5. 사용 알고리즘

4 CHAID
Chi-squared Automatic Interaction Detection

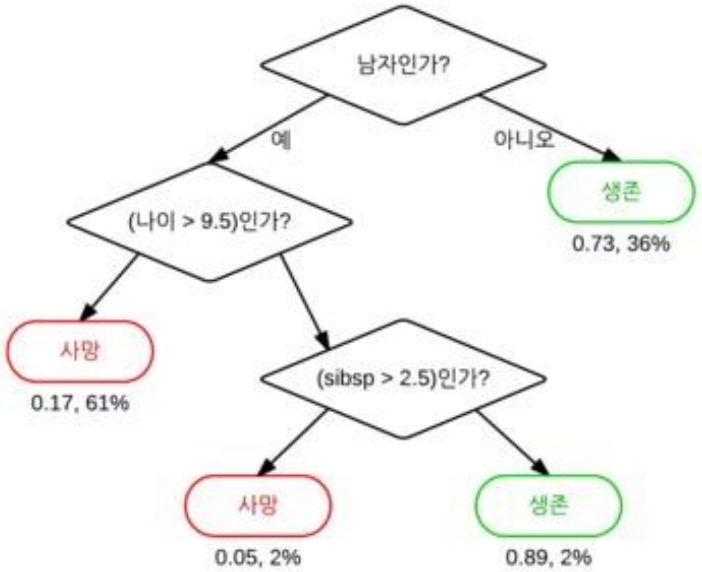
- 원래는 명목형(nominal) 속성만 다룰 수 있었으나,
  수치형 속성은 범주형으로 변환한 후에 적용 가능
- 분리(분할) 기준: 카이 제곱 검정(χ^2), F 검정
- 가지를 뻗는 방법: 다지분리(multiwaysplit)
- 원래 변수들 간의 통계적 관계를 찾는 것이 그 목적
- 가지치기(pruning)가능



6

의사결정나무의 특징

6. 의사 결정 나무의 특징



√ If –Then 구조

√ 순환/반복적/재귀적분할(recursivepartitioning)
: 부모 노드로부터 자식 노드가 생성되고, 이 자식 노드가 또 부모 노드가 되어 다음 자식 노드를 생성

√ 하향식 결정 트리 귀납법(Top-Down Induction of DecisionTrees, TDIDT)
: 루트 노드에서부터 잎 노드까지

7

의사결정나무의 장단점

7. 의사결정나무의 장단점

1 장점

√ 결과를 직관적으로 이해하고 해석하기가 쉽다.

√ 범주형 자료와 수치형 자료 모두에 적용이 가능하다.

√ 시각화된 모형을 보여주므로, 어떤 속성이 분류값에 어떤 영향을 주는지 쉽게 파악할 수 있다.

√ 비모수적 모형이므로, 모형의 가정(선형성, 등분산성 등)이 필요하지 않다.

√ 이상점(Outlier)에 대해 상대적으로 덜 민감하며, 결측치도 처리가 가능하다.

2 단점

✓ 모델이불안정하다.(정확도문제)

✓ 선형성, 추세 파악이 어렵다.

✓ 나무의 깊이(depth)가 너무 깊어질 경우,

과적합의 문제가 발생할 뿐만 아니라 해석도 힘들어진다.

✓ 연속형 속성(주가, 소득등)을 처리/예측하는 능력이

다른 머신러닝 혹은 통계 기법에 비해 떨어진다.

8

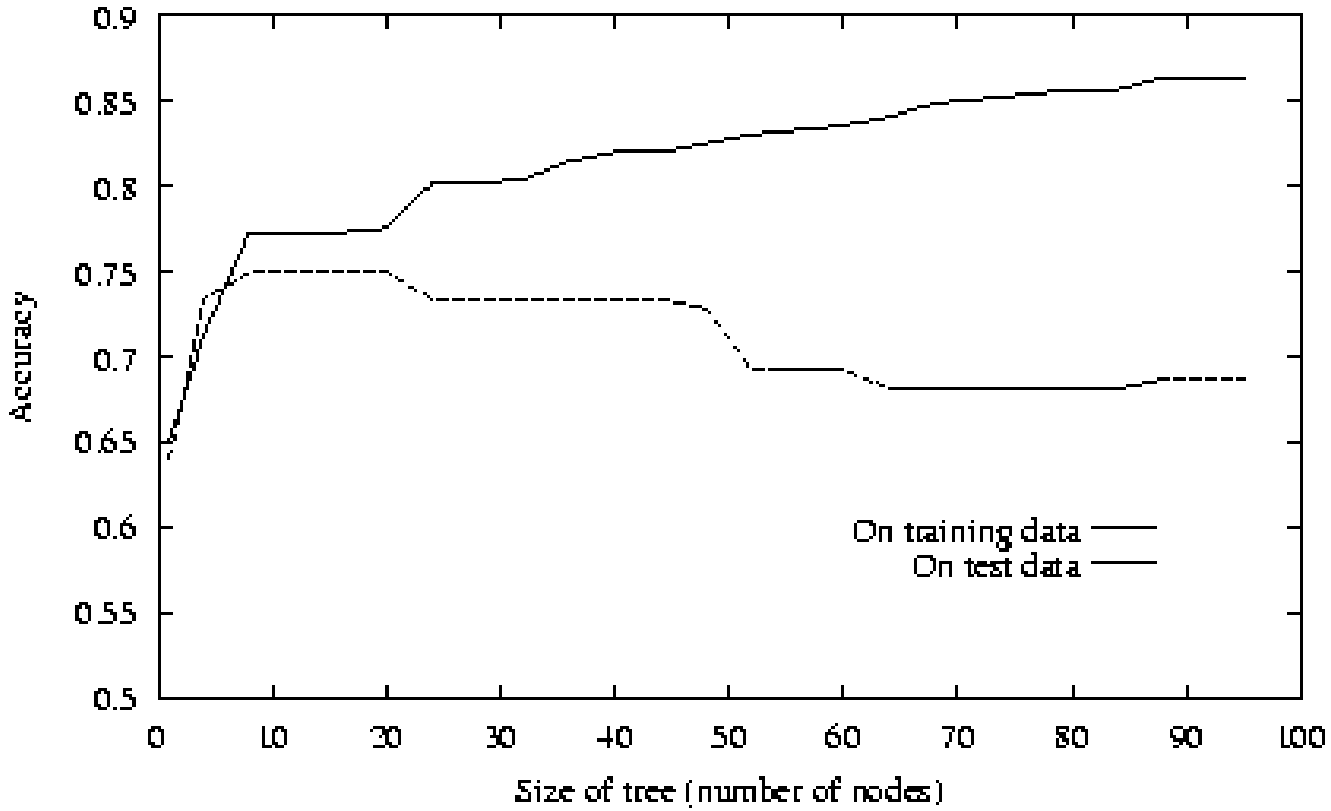
과적합의 해결

- 가지치기(Pruning)

8. 과적합의 해결 – 가지치기(Pruning)

- 과적합 발생! :(

  Overfitting



8. 과적합의 해결 – 가지치기(Pruning)

ㅇ 가지치기 Pruning

√ 의사 결정 나무의 크기를 줄임(가지를 쳐냄)으로써 분류의 정확도를 높인다!

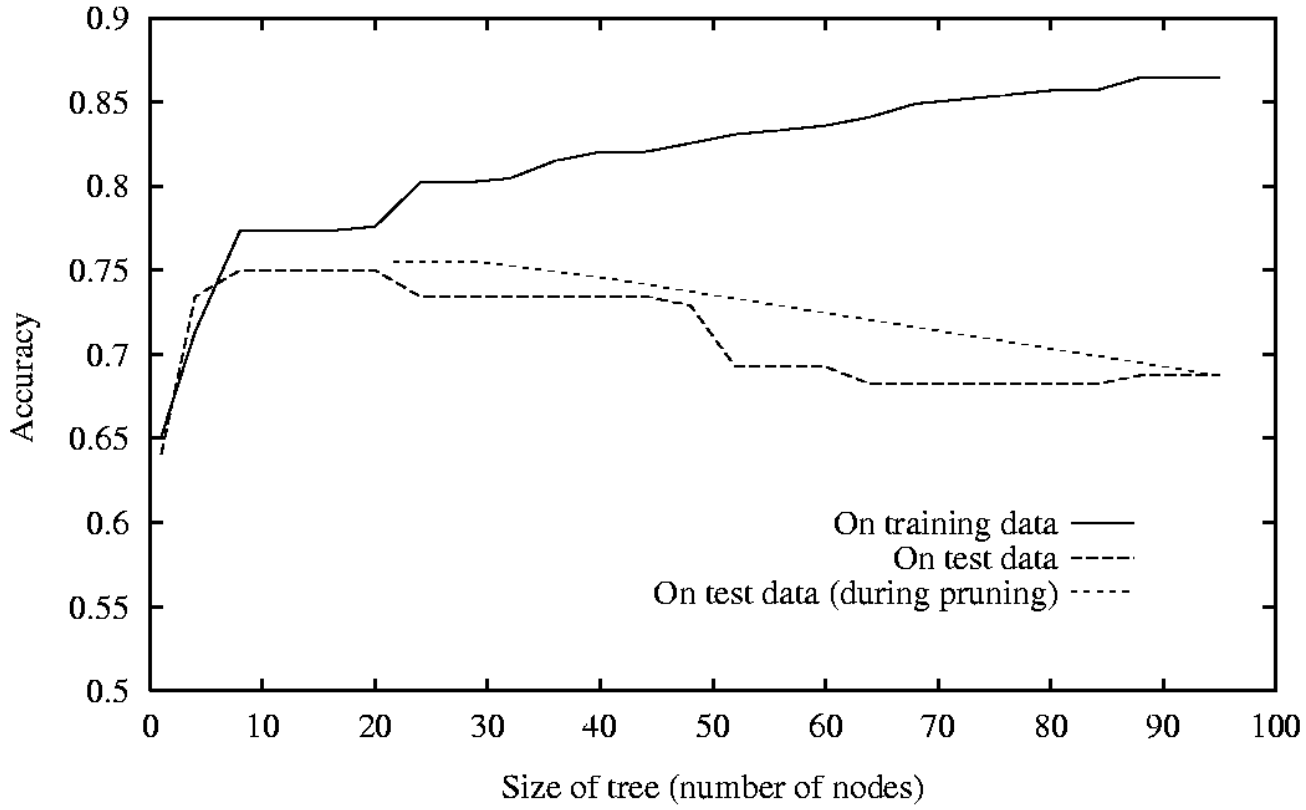
→ 과적합 해결 : training data에만 너무 최적화된 나무는,
Test data나 새로 들어올 데이터의 분류에 비효율적이다.

√ 다양한 방식의 가지치기가 존재한다!

- 사전 가지치기(Pre-Pruning)
- 사후 가지치기(Post-Pruning)
- Pessimistic Pruning
- Rule Post-Pruning
- Etc…

8. 과적합의 해결 – 가지치기(Pruning)

ㅇ 가지치기 Pruning



8. 과적합의 해결 – 가지치기(Pruning)

1 사전 가지치기 Pre-Pruning

✓ Early Stopping Rule / Top-Down Fashion
→ 나무가 다 자라기 전에 알고리즘을 멈춘다.
→ 너무 빨리 성장을 멈춰서 어떤 패턴을 놓칠 수도 있다.

✓ 사후 가지치기(Post-Pruning)보다 빠르나,
실제 분석 시에는 사후 가지치기를 더 선호함

✓ 다양한 사전 가지치기
- Chi-square Pruning
- Minimum No. Of Object Pruning

8. 과적합의 해결 – 가지치기(Pruning)

1 사전 가지치기 Pre-Pruning

√ 어느 노드에서, 언제 멈출까?

1. 모든 데이터가 같은 클래스에 속할 때
2. 모든 속성값들이 같을 때
3. 사용자가 지정한 임계값(threshold)의 수보다 데이터 수가 적을 때
4. 나무를 더 이상 확장해도 불순도(impurity)가 줄지 않을 때

2 사후가지치기 Post-Pruning

✓ Grow to its entirety/ Bottom-Up Fashion

→ 나무가 다 자라고 난 다음 가지를 친다.
→ 특정 노드 아랫 부분(subtree)를 잎 노드로 대체한다.

✓ 다양한 사후 가지치기

- Reduced Error Pruning
- Error Complexity Pruning
- Minimum Error Pruning
- Cost BasedPruning

9

파이썬 예제 코드 실습

감사합니당 :)