〈데이터분석과기계학습 4주차〉

Decision Tree / Random Forest / k-NN

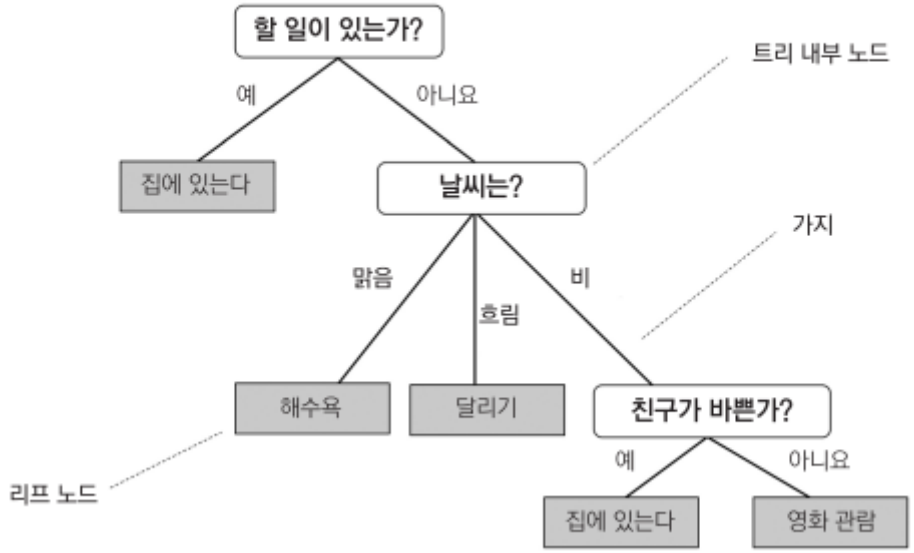
인공지능융합공학부 데이터사이언스전공

곽찬희

3.6. 결정 트리 학습 (Decision Tree, DT)

- **Decision Tree**
  - ✓ 일련의 질문에 대한 결정을 통해 데이터를 분해하는 모델

그림 3-17 어떤 일을 할지 선택하기 위한 결정 트리



3.6.1. 정보 이득 최대화

- 정보 이득(Information Gain, IG)
  - ✓ 가장 정보가 풍부한 특성으로 노드를 나누기 위해 확인하는 정보의 획득 정도를 판별하는 지표
  - ✓ (쉬운 말로) 이 노드를 나누는 것이, 문제 풀이(분류)에 도움이 되는가를 판별하는 지표

$$IG(D_p, f) = I(D_p) - \sum_{j=1}^{m} \frac{N_j}{N_p} I(D_j)$$

  - I: 불순도 지표, f는 분할에 사용할 특성
  - D_p: 부모 노드의 데이터 셋, N_p: 부모 노드에 있는 샘플 개수
  - D_j: 자식 노드의 데이터 셋, N_j: 자식 노드에 있는 샘플 개수
  - 즉, IG = 부모 노드의 불순도 - (자식 노드의 불순도의 합)
  - IG가 크다 =

3.6.1. 정보 이득 최대화

- **정보 이득(Information Gain, IG)**
  - ✓ **Binary DT에 널리 사용되는 세 개의 불순도 지표 (p(i|t) 는 노드 t에서 클래스 i의 비율)**
    1. **지니 불순도 (Gini impurity, I_G)**

$$I_G(t) = \sum_{i=1}^{c} p(i|t)\big(1 - p(i|t)\big) = 1 - \sum_{i=1}^{c} p(i|t)^2$$

- **잘못된 확률을 최소화하기 위한 기준으로 이해 가능**
- **클래스가 완벽히 섞여 있을 때 최대**

- **예) 클래스가 $50\%/50\%$일 때**

$$I_G(t) = 1 - \sum_{i=1}^{c} 0.5^2 = 0.5$$

3.6.1. 정보 이득 최대화

- 정보 이득(Information Gain, IG)
  - ✓ Binary DT에 널리 사용되는 세 개의 불순도 지표 (p(i|t) 는 노드 t에서 클래스 i의 비율)

    2. 엔트로피 (entropy, I_H)

$$I_H(t) = -\sum_{i=1}^{c} p(i|t) \log_2 p(i|t)$$

    - 한 노드의 모든 샘플이 같은 클래스면, 엔트로피는 0이 됨.
    - 한 노드가 두 클래스를 동등하게 가지고 있으면 엔트로피는 1이 됨.

  - ✓ 실제 사용 시, 지니 불순도와 엔트로피가 비슷한 결과를 보임
  - ✓ DT의 경우 불순도 조건을 바꾸기보다 가지치기(pruning)을 통해 튜닝 진행

3.6.1. 정보 이득 최대화

- 정보 이득(Information Gain, IG)
  - ✓ Binary DT에 널리 사용되는 세 개의 불순도 지표 (p(i|t) 는 노드 t에서 클래스 i의 비율)
    3. 분류 오차 (classification error, I_E)

$$I_E(t) = 1 - \max\{p(i|t)\}$$

    - 엔트로피와 마찬가지로, 한 노드의 모든 샘플이 같은 클래스면 0(최소), 두 클래스가 동등하면 0.5(최대)

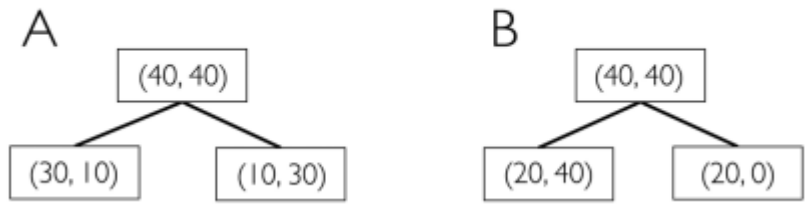
  - ✓ 노드의 클래스 확률 변화에 덜 민감하기 때문에, 가지치기에 주로 사용 (트리 구성엔 X)

트리 구성 시나리오

- **클래스가 40/40 인 노드를 나누는 두 가지 방법**

$$I_E(D_p) = 1 - 0.5 = 0.5$$

그림 3-18 두 개의 분할 시나리오



- **분류 오차일 때 (classification error): 0.25 vs 0.25**

A

$$I_E(D_{left}) = 1 - \frac{3}{4} = 0.25$$

$$I_E(D_{right}) = 1 - \frac{3}{4} = 0.25$$

$$IG_E = 0.5 - \frac{4}{8}0.25 - \frac{4}{8}0.25 = \mathbf{0.25}$$

B

$$I_E(D_{left}) = 1 - \frac{4}{6} = \frac{1}{3}$$

$$I_E(D_{right}) = 1 - 1 = 0$$

$$IG_E = 0.5 - \frac{6}{8} \times \frac{1}{3} - 0 = \mathbf{0.25}$$

트리 구성 시나리오

- **클래스가 40/40 인 노드를 나누는 두 가지 방법**

$$I_G(D_p) = 1 - (0.5^2 + 0.5^2) = 0.5$$

- **지니 불순도 (Gini impurity): 0.125 vs. <u>0.16</u>**

A

$$I_G(D_{left}) = 1 - \left(\left(\frac{3}{4}\right)^2 + \left(\frac{1}{4}\right)^2\right) = 0.375$$

$$I_G(D_{right}) = 1 - \left(\left(\frac{1}{4}\right)^2 + \left(\frac{3}{4}\right)^2\right) = 0.375$$

$$IG_G = 0.5 - \frac{4}{8}0.375 - \frac{4}{8}0.375 = \mathbf{0.125}$$

B

$$I_G(D_{left}) = 1 - \left(\left(\frac{2}{6}\right)^2 + \left(\frac{4}{6}\right)^2\right) = 0.4$$

$$I_G(D_{right}) = 1 - \left((1)^2 + (0)^2\right) = 0$$

$$IG_G = 0.5 - \frac{6}{8}0.4 - 0 = \mathbf{0.16}$$

트리 구성 시나리오

- **클래스가 40/40 인 노드를 나누는 두 가지 방법**

$$I_H(D_p) = -(0.5\log_2 0.5 + 0.5\log_2 0.5) = 1$$

- **엔트로피 (Entropy): 0.19 vs. 0.31**

A

$$I_H(D_{left}) = -\left(\frac{3}{4}\log_2\frac{3}{4} + \frac{1}{4}\log_2\frac{1}{4}\right) = 0.81$$

$$I_H(D_{right}) = -\left(\frac{1}{4}\log_2\frac{1}{4} + \frac{3}{4}\log_2\frac{3}{4}\right) = 0.81$$

$$IG_H = 0.5 - \frac{4}{8}0.81 - \frac{4}{8}0.81 = \mathbf{0.19}$$

B

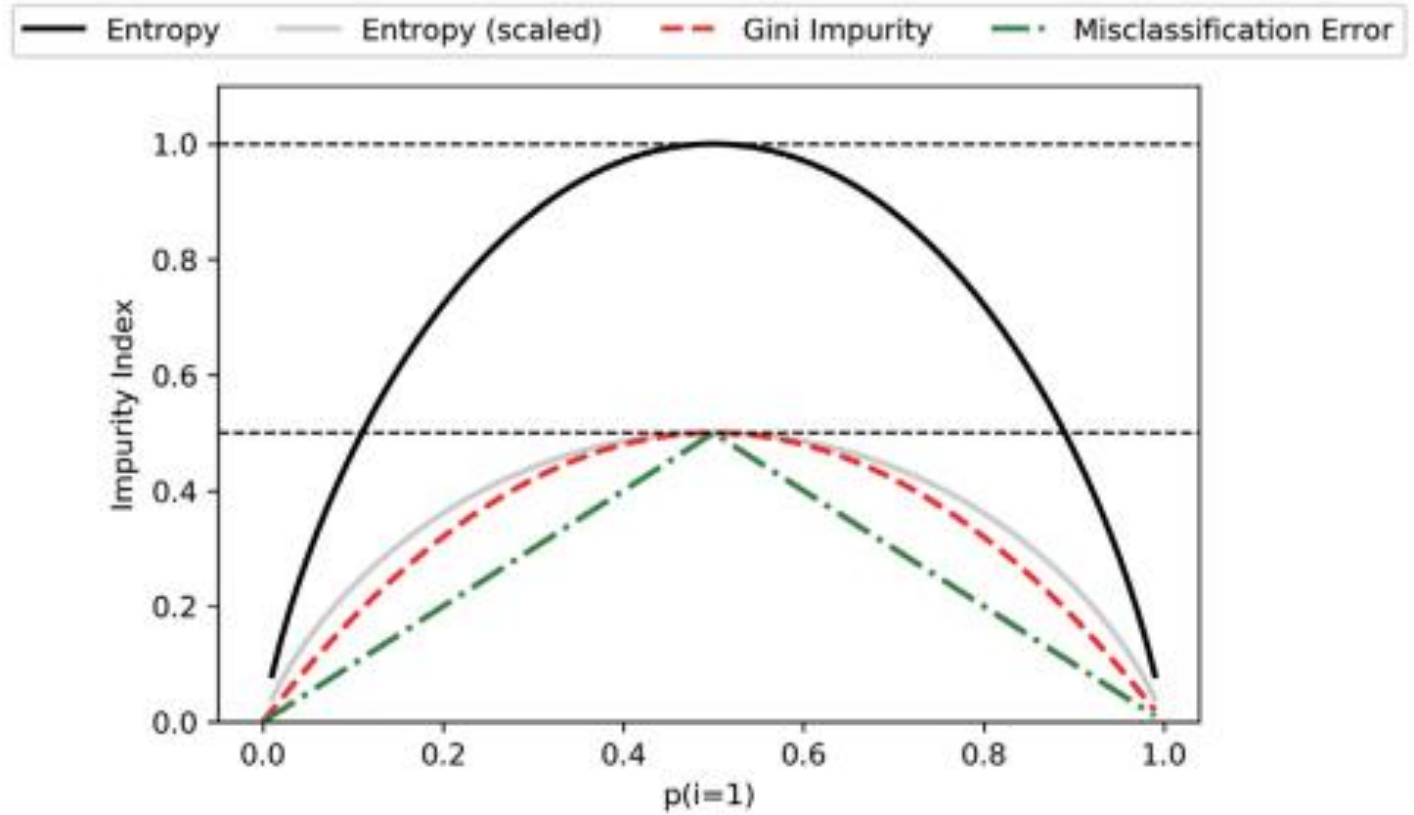
$$I_H(D_{left}) = -\left(\frac{2}{6}\log_2\frac{2}{6} + \frac{4}{6}\log_2\frac{4}{6}\right) = 0.92$$

$$I_H(D_{right}) = 0$$

$$IG_H = 0.5 - \frac{6}{8}0.92 - 0 = \mathbf{0.31}$$

불순도 지표 비교

그림 3-19 불순도 지표 비교

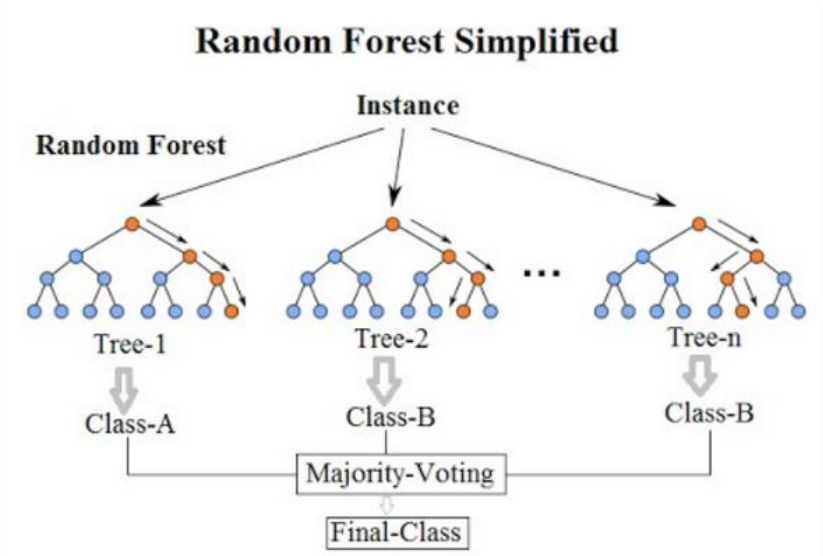


3.6.3. 랜덤 포레스트

- 랜덤 포레스트(random forest)
  - ✓ Tree 를 랜덤성을 고려해서 많이 만들어보자(나무나무나무나무나무->숲)
  - ✓ 트리의 앙상블(ensemble)
  - ✓ 다음과 같은 단계로 요약 가능
    1. n개의 랜덤한 부트스트랩(bootstrap) 샘플을 추출 (훈련 셋에서 중복을 허용하면서 랜덤하게 n개 샘플 추출)
    2. 부트스트랩 샘플에서 결정 트리를 학습. 각 노드에 대해
       a. 중복을 허용하지 않고 랜덤하게 d개의 특성을 선택
       b. 정보 이득(IG)와 같은 목적 함수를 기준으로 최선의 분할을 만드는 특성을 사용해 노드 분할
       c. a, b를 k번 반복
       d. 각 트리의 예측을 모아 다수결 투표(majority voting)의 결과로 클래스 레이블 할당

3.6.3. 랜덤 포레스트

- 랜덤 포레스트(random forest)
  - ✓ DT만큼 해석이 쉽지 않지만, 하이퍼파라미터 튜닝이 간단함
  - ✓ 가지치기의 필요가 없음 (ensemble 모델의 효과)
  - ✓ 실전에서 중요하게 결정해야 할 파라미터는 트리 개수 하나임.
    - 트리가 많을 수록 성능이 좋아짐 (but 계산이 더 필요)
  - ✓ 그 외에 샘플의 크기 n, 선택할 특성의 개수 d 등도 튜닝 가능
    - n을 통해 편향-분산 트레이드오프를 조절

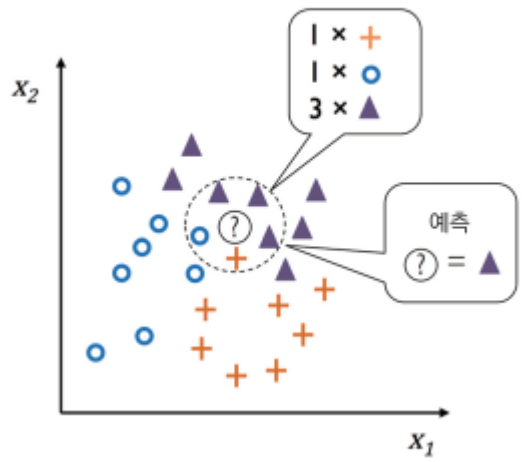


https://upload.wikimedia.org/wikipedia/commons/7/76/Random_forest_diagram_complete.png

k-최근접 이웃(k-NN, k-Nearest Neighbor)

- k-NN
  - ✓ k개의 가까운 이웃을 보고, 클래스를 판단함
  - ✓ 게으른 학습기(lazy learner)
    - 훈련 데이터를 함수로 판별하는 것이 아닌, 데이터셋을 메모리에 저장 후 판별
  - ✓ k-NN 알고리즘 요약
    1. 숫자 k와 거리 측정 기준을 선택
    2. 분류하려는 샘플에서 k개의 최근접 이웃을 찾음
    3. 다수결 투표를 통해 클래스 레이블 할당
       - 만약 다수결 결과가 같다면, 가장 가까운 것으로 선택
       - 홀수 k 를 선정하면 동점이 되지 않음

그림 3-25 k-최근접 이웃의 다수결 투표

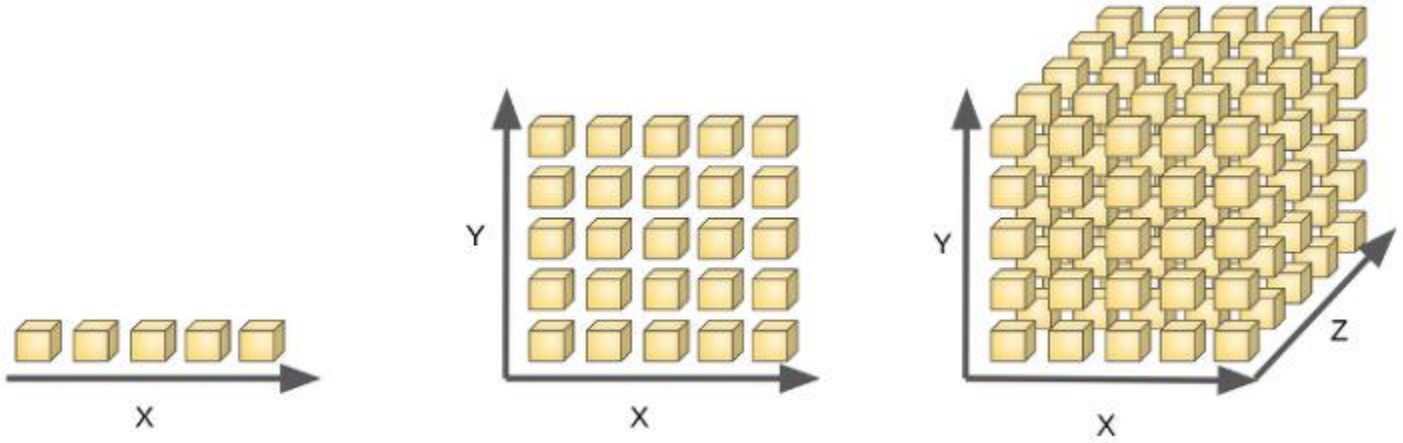


k-최근접 이웃(k-NN, k-Nearest Neighbor)

- 거리를 구하는 법
  - ✓ 맨하탄 거리
    - $|x2 - x1| + |y2 - y1|$
  - ✓ 유클리디안 거리 (feat 피타고라스)
    - $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$
    - 차원이 늘어나도 가능
  - ✓ 위를 일반화한 것이 minkowski distance
    - $d\left(x^{(i)}, x^{(j)}\right) = \sqrt[p]{\sum_k |x_k^i - x_k^j|}$
- 거리 측정에는 이 밖에도 다양한 방식 존재
- 거리 측정 전 변수표준화 중요. Why?

차원의 저주

- 차원의 저주(the curse of dimensionality)
  - ✓ 고정된 크기의 훈련 데이터셋이 차원이 늘어남에 따라 특성 공간이 점점 희소해지는 현상
  - ✓ 차원이 늘어날 수록 더 다양한 데이터가 필요한데, 차원을 늘리면 차원의 수가 기하급수적으로 증가해 빈 차원(공간이) 늘어남



https://img1.daumcdn.net/thumb/R720x0.q80/?scode=mtistory2&fname=http%3A%2F%2Fcfile1.uf.tistory.com%2Fimage%2F99BFBF425DBFA8BC2A7DB9

자 이제 실습을!