Decision Tree & Ensemble

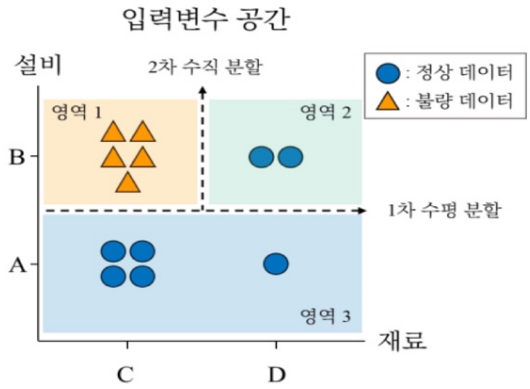
Decision Tree

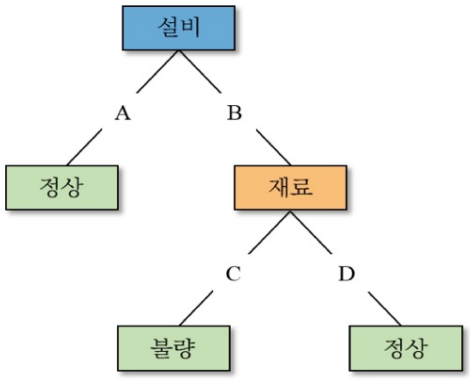
지도학습의 종류: 회귀(Regression) / 분류(Classification)

DT는 기본적으로 Classification Task를 위한 방법론 → Regression Tree로 확장 가능

Decision Tree란?

- 모델의 의사결정 규칙을 나무 형태로 표현하는 모델
- 나무의 경로는 하나의 의사결정 규칙을 의미하며, 입력변수 데이터가 들어오면 규칙에 따라 범주형 출력변수를 예측(일련의 필터 과정 또는 스무고개)
- 분할된 영역에 동일한 클래스 데이터가 최대한 많이 존재하도록 함





불순도(Impurity)

- 불순물이 포함된 정도 = 해당 범주 안에 서로 다른 데이터가 얼마나 섞여 있는지
- DT는 불순도를 최소화하는 방향으로 학습 진행
- 불순도 지표로 Entropy 사용

CART(Classification and Regression Tree)

- 지니 불순도(Gini Impurity) 사용

$$Gini(S) = 1 - p_+^2 - p_-^2$$

S : 분할된 영역 내에 존재하는 데이터 집합

P+ : '정상' 데이터의 비율

P- : '불량' 데이터의 비율

CART 알고리즘에서는 모든 조합에 대해 Gini Impurity를 계산한 후, 가장 낮은 지표를 찾음

- 정보 이득

Information_Gain(S,A)는 영역 S의 데이터를 입력변수 A로 분할하는 경우의 혼잡도 감소량

→ 어떤 질문을 기준으로 나눠야 하는가에 대한 지표로서 작용

$$Information\_Gain(S,A) = Gini(S) - \sum_{v \epsilon Values(A)} \frac{|S_v|}{|S|} Gini(S_v)$$

입력변수 A로 데이터를 분류하기 전의 혼잡도

입력변수 A로 데이터를 분류한 후의 가중 평균 혼잡도

DT는 정보 이득을 **최대화**하는 방향으로 학습을 결정

- 나무 구조 생성 과정
  - CART는 가지 분기 시, 이진 분할(Binary Split) 수행
  - 입력변수들 중에서 IG가 가장 큰 입력변수 선택하여 입력
  - 분류 정확도가 최대화 될 때까지 재귀적으로 반복
  - 오버피팅 문제를 막기 위해 끝 노드의 일부를 제거하는 가지치기 작업을 수행
    - 사전 가지치기 / 사후 가지치기

- 회귀나무

분기 지표를 선택할 때 사용하는 index를 불순도가 아닌 실제값과 예측값의 오차를 사용(= 분산)

- 연속형 입력변수

데이터를 구간(Interval)으로 분할하여 범주형 변수로 전환

단일 경계값 방법(Binary Discretization) 사용

Ensemble

다수의 데이터 집합에 대해서 각각의 머신 러닝 모델을 학습하고, 학습된 모델의 예측을 결합하여 최종 예측 수행

여러 Weak Learner들이 모여 투표(Voting)를 통해 Stronger Learner 구성

Hard Voting

- 클래스별 예측확률을 제시 → 최종 예측값 계산(다수결 투표)

Soft Voting

- Average
  - 예측 확률값을 단순 평균내어 확률이 더 높은 클래스 선정
- Weighted Sum
  - 가중치를 부여하여 가중치 합 사용

Algorithm

- Bagging
- Bootstrap(복원추출): 주어진 데이터셋에서 Random Sampling 하여 새로운 데이터셋 생성
  - 모델 결합
    - 출력변수가 범주형(분류 문제): 각 모델의 예측값들을 다수결 투표하여 최종 예측
    - 출력변수가 연속형(회귀 문제): 각 모델의 예측값들을 평균 내어 최종 예측
  - Random Forest
- Boosting
  - 모델을 반복 업데이트 → 이전 iteration의 결과에 따라 데이터셋 샘플에 대한 가중치 부여

        - GBM: 잔차를 지속적으로 학습
        - XGBoost: 이전 라운드에서의 예측 오류를 다음 라운드의 모델 학습에 반영 (GBM과 동일)
            - 규제항(트리의 복잡성에 패널티를 부여하는 항)이 추가됨
- Stacking
    - Meta learner로 학습시켜 최종 예측값 결정