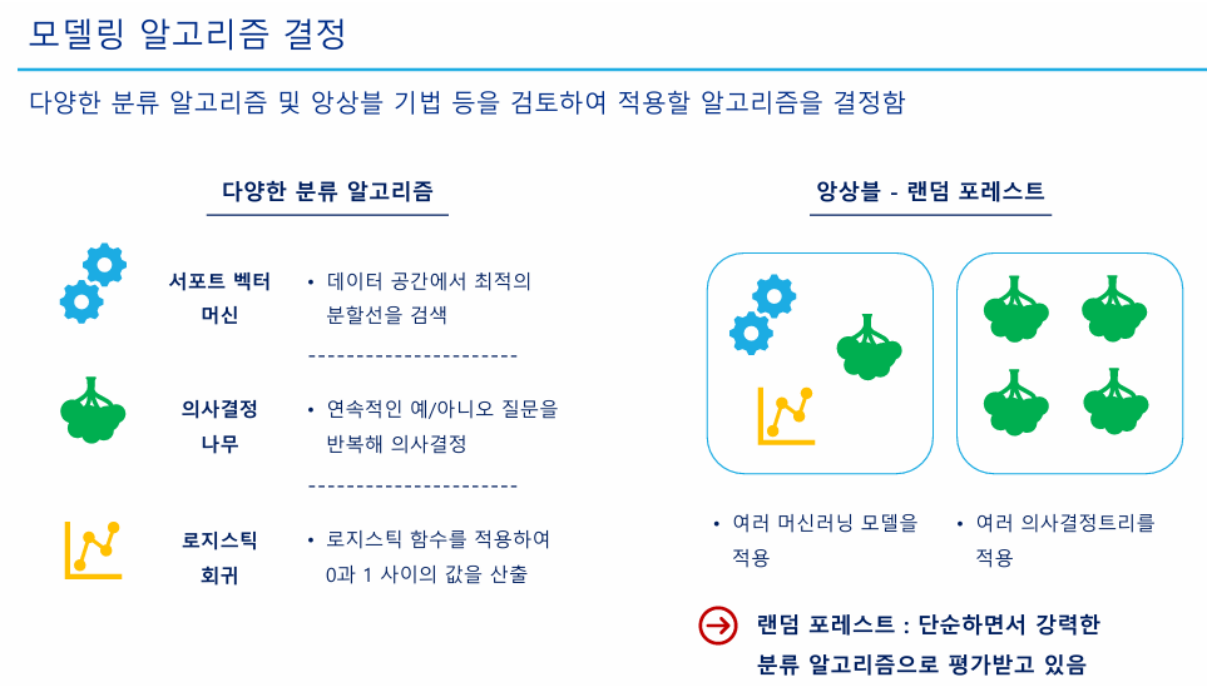
알고리즘

✅ 랜덤 포레스트

머신러닝 - 지도학습 - 분류: 범주예측(Classification)에서 사용됨.

분류 알고리즘?



랜덤 포레스트의 개념

데이터 세트에서 여러 번의 데이터 서브셋을 추출, **의사결정나무**를 각각 적용.
비교적 간단한 방법이지만, 가장 강력한 머신러닝 알고리즘 중 하나.
추가 조정을 위해
하이퍼 파라미터가 이용된다.

Azure 에서는?

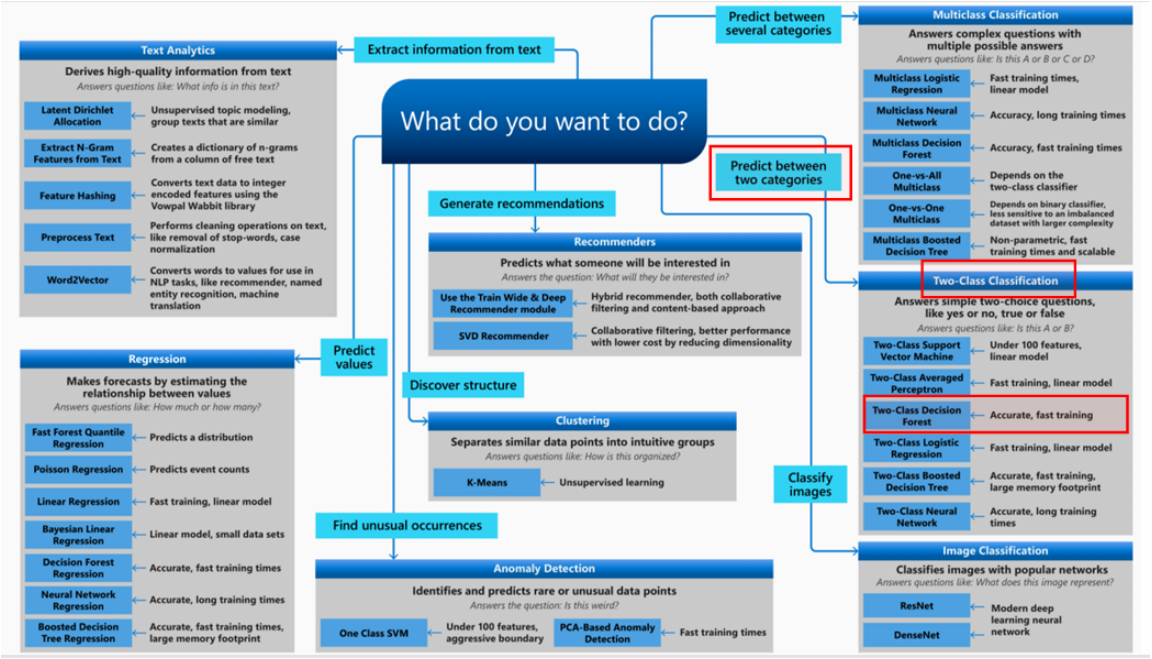
designer 기준

data → select colums in dataset(컬럼 선택) → clean missing data(누락값 처리) → edit metadata (데이터 변환) → convert to indicator values (0,1로 변환) → split data (학습 데이터, 테스트 데이터 분리)

알고리즘 적용 (랜덤 포레스트의 이진분류법 - two-class decision forest)

1. two-class decision forest와 split data의 학습 부분을 train model에 엮고, **정답으로 사용할 컬럼 지정**
2. train model과 split data의 예측 부분을 score model로 적고, true로 세팅 후에
3. evaluate model로 평가 (job 실행)

▼ 알고리즘 표 참조



▼ 데이터 변환 참조

데이터 변환 - Category→Indicator value

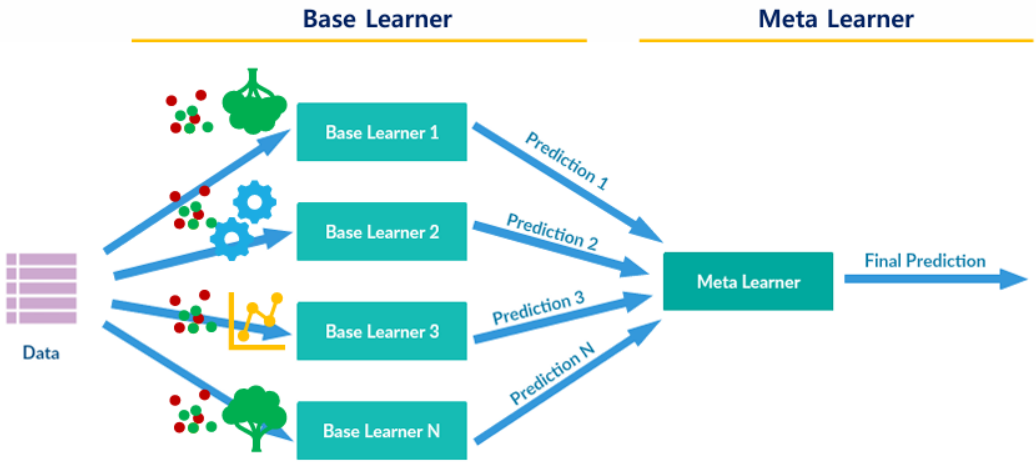
문자열 데이터 형식을 범주형 데이터로 변경 후, 순서가 없는 범주형 데이터의 경우 indicator value로 변환

컬럼명	형식
education	String
marital_status	String
occupation	String
relationship	String
race	String
sex	String

✅ 앙상블 기법

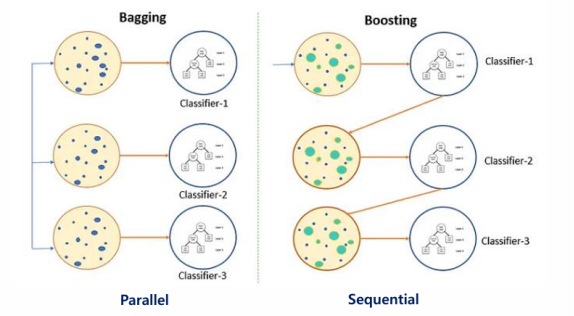
앙상블 기법

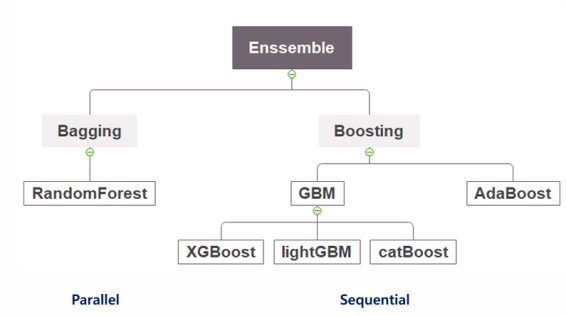
단일 모델이 아닌 여러 모델을 사용하여 예측하는 것을 앙상블 기법이라고 함



고로, 앙상블 기법은 여러 weak learner를 이용하여 strong learner를 구성하는 효과적인 방법.

앙상블의 대표적 기법은 Bagging과 Boosting

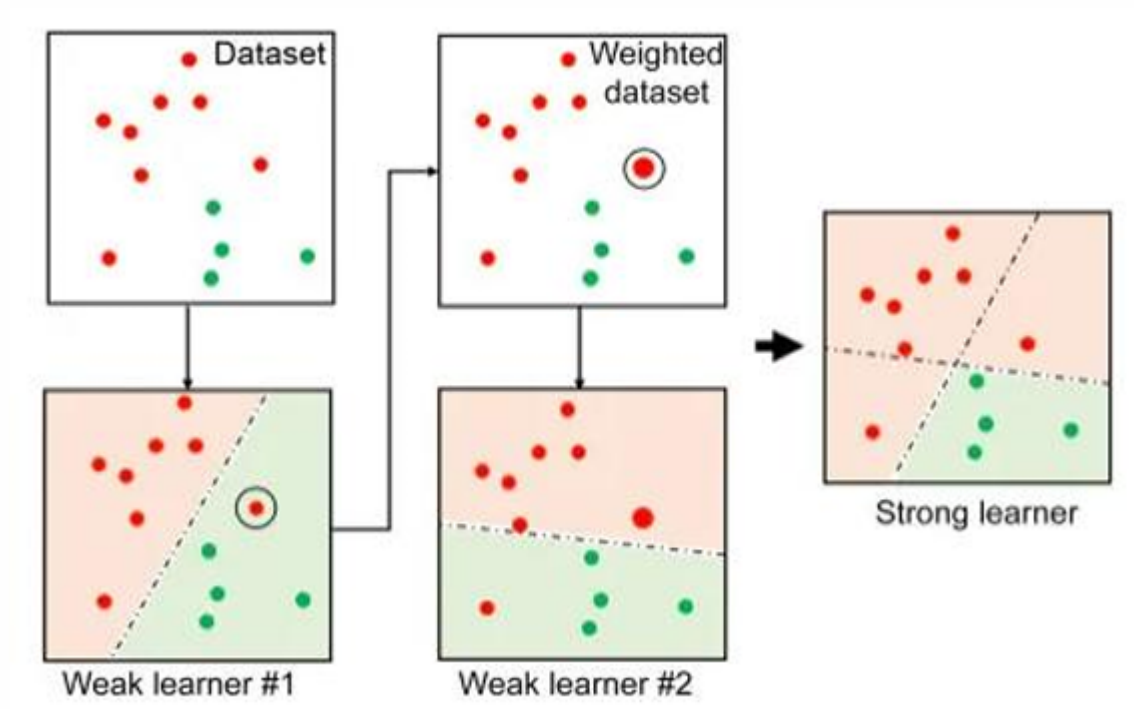




그럼, boosting 알고리즘?

여러 개의 단순한 모델(weak learner)을 순차적으로 구성하는 앙상블 모델의 일종.
하나의 깊은 트리로 구성하는 랜덤 포레스트와 달리, 깊이가 2인 나무를 여러 개 이용하여 strong learner를 구축.

▼ 표 참고



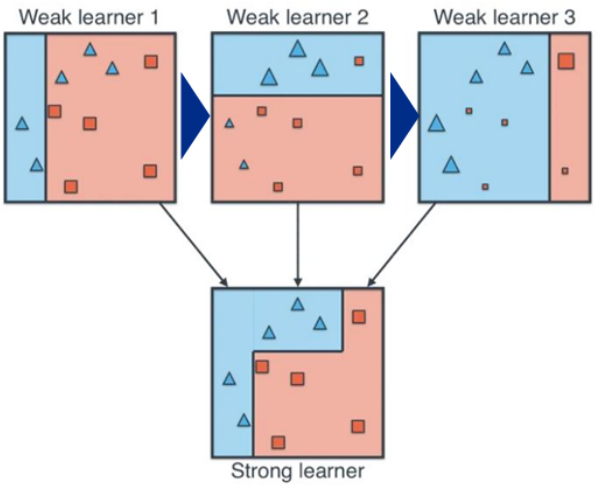
대표적 : AdaBoosting

각 단계에서 이전 단계의 단점을 개선해 나가는 부스팅 알고리즘의 일종으로, **큰 오류에 집중**하여 개선하는 방법을 취함

▼ 표 참고

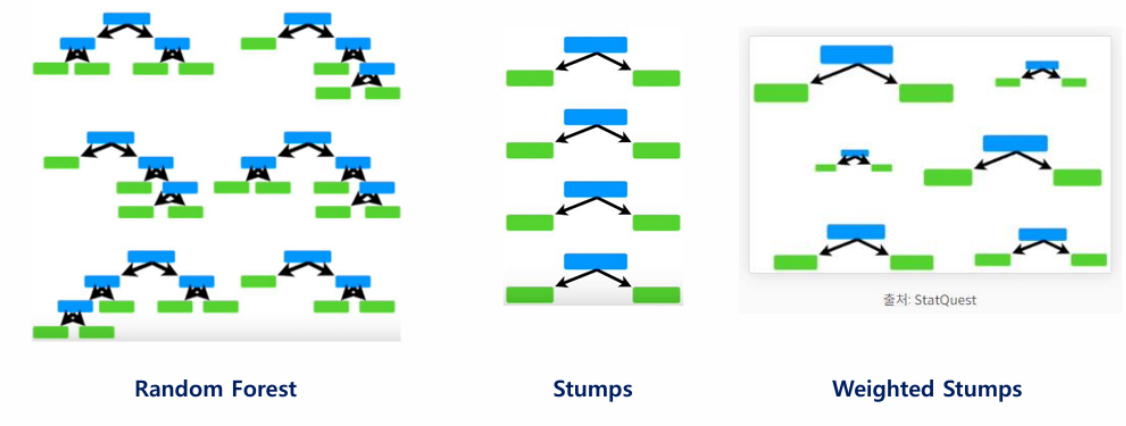
큰 오류에 집중하여 개선하는 방법을 취함

- 이전 단계의 오류 개선(큰 오류에 집중)
  - 이전 단계에서 오류가 큰 데이터의 가중치를 높임
  - 이전 단계에서 오류가 작은 데이터는 가중치를 낮춤
- 이전 단계에서 조정된 가중치에 기반하여 데이터 학습에 적용. (데이터 마다 가중치를 적용)
- 순차적으로 반복 후 단계별 결과물을 종합하는 앙상블 모델 구성



그리고 이는, weak learner로 깊은 tree가 아닌, 깊이가 2인 stump를 사용.
깊은 tree에 비해 예측력이 낮은 여러 stump들의 가중치를 달리하여, 예측력 높은 모델을 구성함.

▼ 표 참고

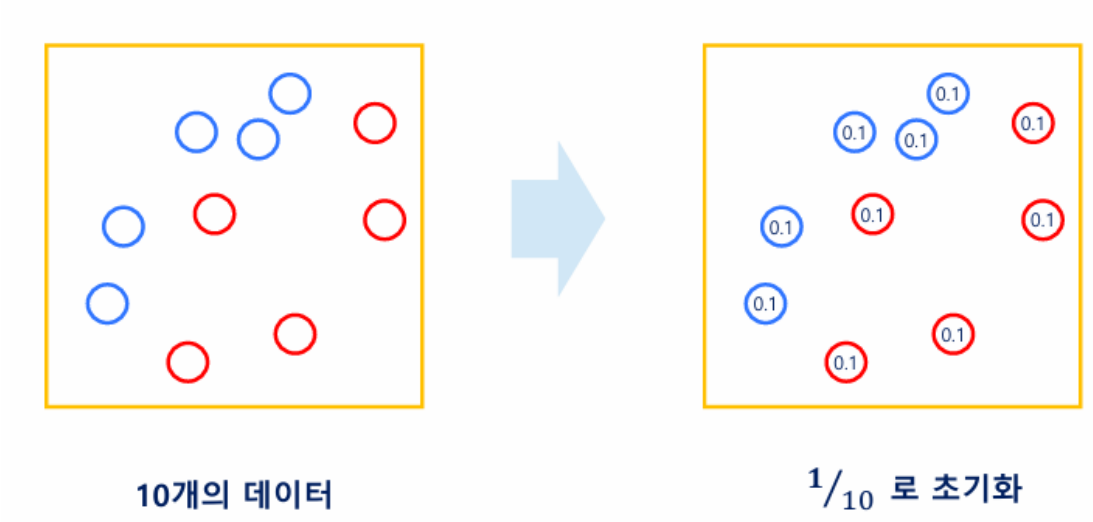


AdaBoosting 알고리즘 계산법

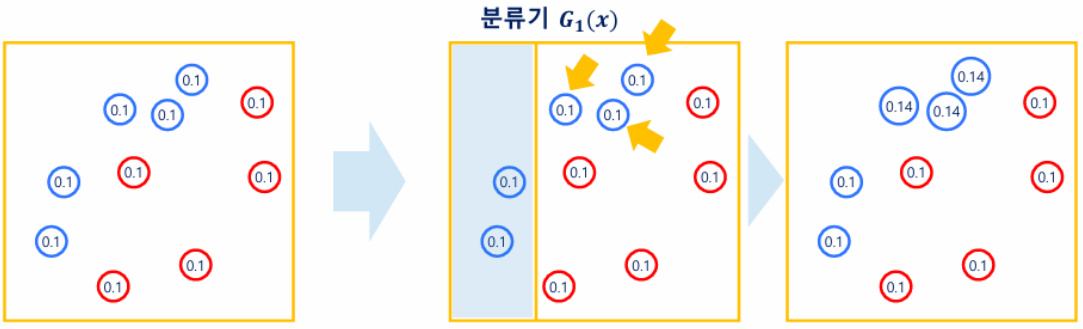
1. 모든 데이터의 가중치를 1/n로 초기화
2. 분류기 G1(x) 적용 후 가중치 갱신
3. 분류기 G2(x) 적용 후 가중치 갱신
4. 분류기 G3(x) 적용 후 가중치 갱신

5. 최종 모델

▼ 표 참고

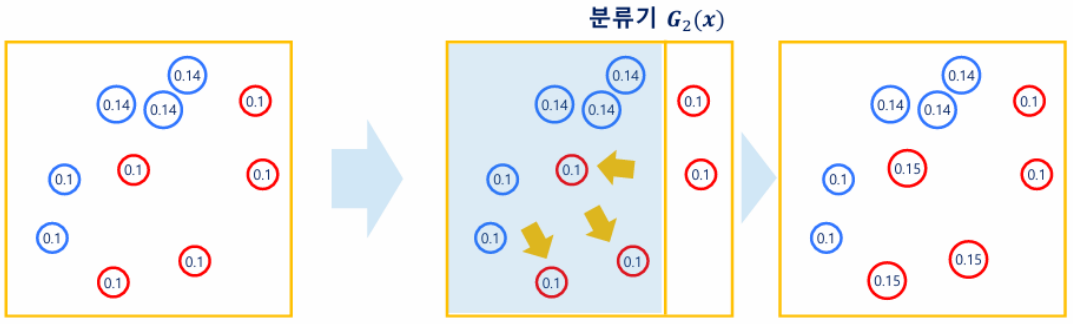


(2) 분류기 $G_1(x)$ 적용 후 가중치 갱신



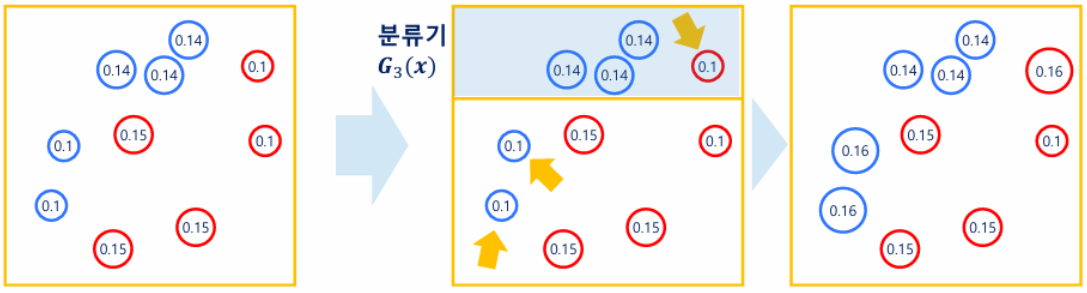
손실 = 0.3
α_1 = 0.37
못맞춘 데이터의 가중치 갱신 = 0.14

(3) 분류기 $G_2(x)$ 적용 후 가중치 갱신



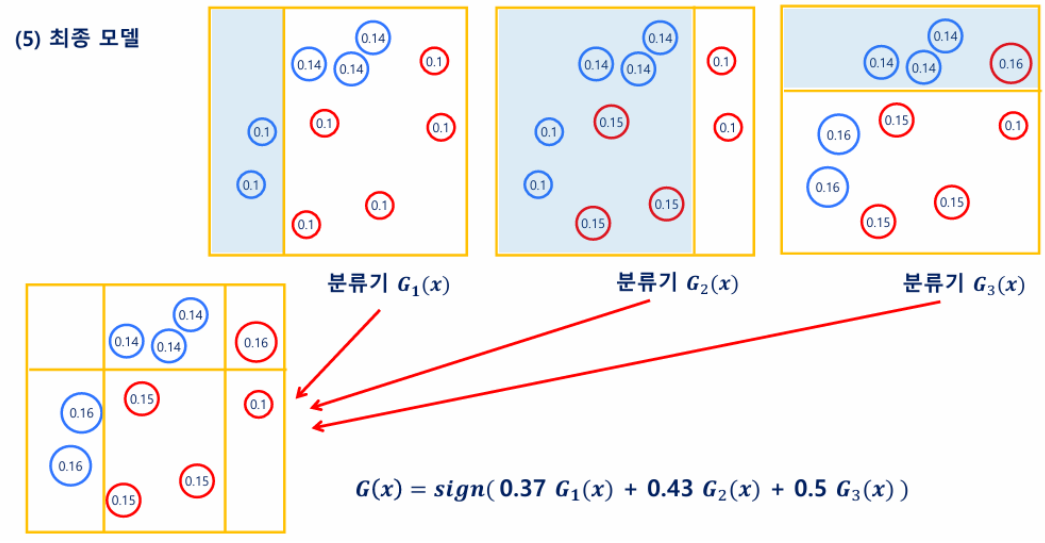
손실 = 0.27
α_2 = 0.43
못맞춘 데이터의 가중치 갱신 = 0.15

(4) 분류기 $G_3(x)$ 적용 후 가중치 갱신



손실 = 0.24
α_3 = 0.5
못맞춘 데이터의 가중치 갱신 = 0.16

(5) 최종 모델

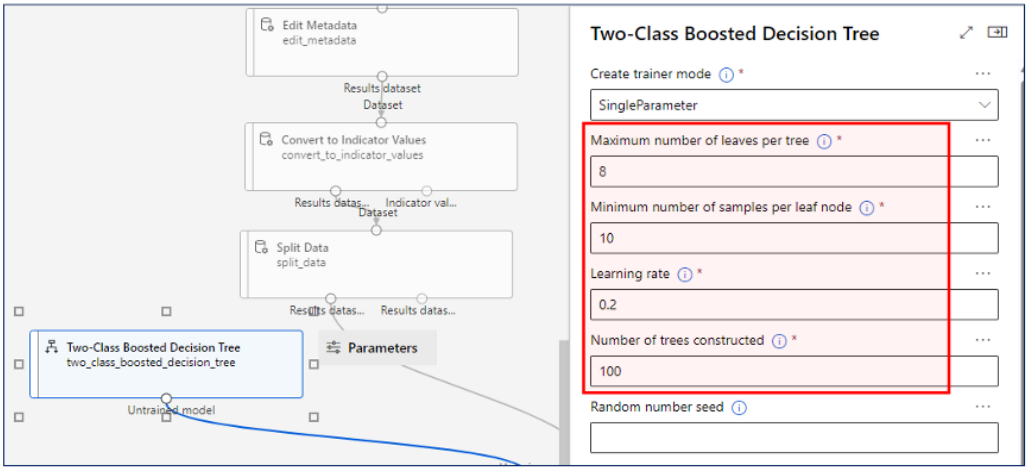


$$G(x) = sign(\,0.37\ G_1(x) + 0.43\ G_2(x) + 0.5\ G_3(x)\,)$$

▼ 하이퍼 파라미터 참고

모델링 알고리즘 선택

Designer로 돌아가서, 알고리즘의 하이퍼파라미터 설정



132

▼ 요약

wo-Class Boosted Decision Tree 하이퍼파라미터 요약

1. **Maximum number of leaves per tree**
   - **설명**: 한 트리에서 가질 수 있는 최대 리프 노드 개수.
   - **역할**: 모델 복잡도 조절, 값이 클수록 복잡한 모델 생성.
   - **기본값**: 8.
2. **Minimum number of samples per leaf node**
   - **설명**: 각 리프 노드에 필요한 최소 샘플 수.
   - **역할**: 값이 클수록 노드 분할이 제한되어 과적합 방지.
   - **기본값**: 10.
3. **Learning rate**
   - **설명**: 부스팅 학습 속도를 조절.
   - **역할**: 값이 작으면 학습 속도가 느리지만 일반화 잘됨, 값이 크면 빠르게 학습하나 과적합 가능성 높아짐.
   - **기본값**: 0.2.
4. **Number of trees constructed**
   - **설명**: 학습 과정에서 생성되는 트리의 개수.

- **역할**: 트리 개수가 많을수록 복잡한 패턴 학습 가능.
- **기본값**: 100.

5. **Random number seed**
    - **설명**: 난수 생성의 시드 값으로, 결과 재현성을 보장.
    - **역할**: 동일한 설정에서 실험을 반복 가능하게 함.
    - **기본값**: 설정하지 않으면 매번 다른 결과 생성.

✅ 두 알고리즘의 차이

1. 기본 알고리즘의 차이

특징	Two-Class Decision Forest	Two-Class Boosted Decision Tree
알고리즘	랜덤 포레스트 (Random Forest)	부스팅 (Boosting) 기반 결정 트리
학습 방식	여러 트리를 병렬로 학습 (병렬 학습)	이전 트리의 오차를 기반으로 새 트리를 학습 (순차 학습)
트리 간 관계	각 트리는 독립적으로 작동	이전 트리를 보완하여 점진적으로 학습
목적	데이터의 과적합 방지 및 높은 안정성 제공	높은 정확도를 목표로 오차를 줄이는 데 집중

2. 선택 기준

1. **Two-Class Decision Forest**를 선택하세요:
    - 데이터 크기가 크고 복잡하지 않은 경우.
    - 빠른 학습과 안정적인 성능이 중요한 경우.
2. **Two-Class Boosted Decision Tree**를 선택하세요:
    - 데이터가 불균형하거나 복잡도가 높은 경우.
    - 예측 정확도를 최대화하는 것이 중요한 경우.
    - 충분한 계산 리소스가 제공될 때.