Ensemble

Winter Vacation Capstone Study

TEAM Kai.Lib

발표자 : 원철황

2020.01.06 (FRI)

Ensemble Classifier Notation

- **앙상블 분류기**

  앙상블 분류기는
  - 학습 데이터를 사용하여 여러 개의 서로 다른 분류기를 만들고
  - 이들 분류기의 판정 결과를 투표방식이나 가중치 투표 방식으로 결합하여
  - 최종 부류를 결정하는 분류기이다.

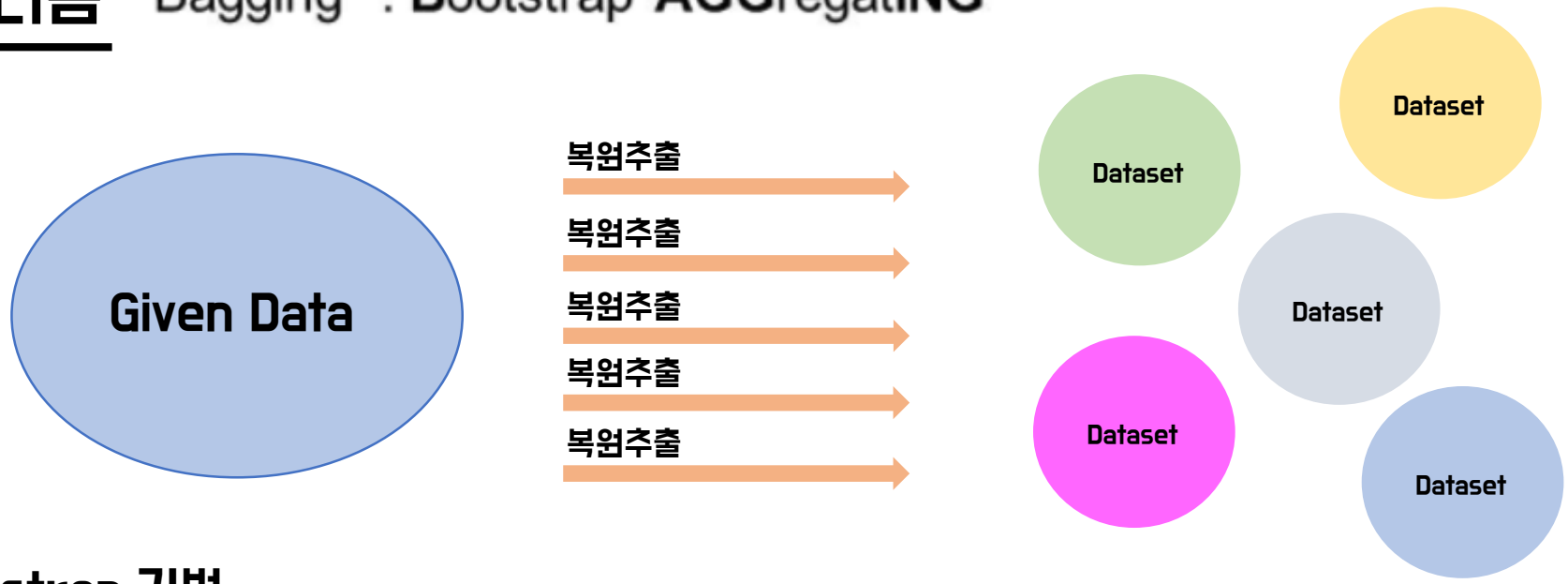
  ※ 앙상블 분류기를 학습할 때,
  서로 다른 분류기를 만들기 위해서 다른 데이터 집합들을 사용하거나 서로 다른 속성들을 사용하는 등의 기법이 적용된다.

구현 방법

배깅 (Bagging, Bootstrap aggregating)
부스팅 기법 (Boosting Algorithm)
보팅 기법 (VotingAlgorithm)
스태킹 기법 (Stacking Algorithm)

Bias-Variance Tradeoff 개념이 앙상블 기법의 본질이므로, 이는 추후 다룰 예정

배깅 알고리즘 "Bagging" : Bootstrap AGGregatING



▷ Bootstrap 기법

:: 주어진 학습 데이터 집합으로부터 복원추출을 하여 여러 개의 **서로 다른 학습 데이터 집합**을 만들어 내는 기법

▷ Bagging

:: 붓스트랩을 통해 만들어진 각 학습 데이터로부터 집합별로 분류기를 만들고, **투표**나 **가중치 투표**를 하여 최종 판정을 하는 방법

▷ Random Forest

:: 분류기로 결정트리를 사용하는 배깅 기법. 결정트리를 만들어가는 과정에서 분할 속성을 결정할 때, **분할 속성 후보**들을 무작위로 선택한 다음, 이들에 대해서만 **정보 이득** 등을 계산하여 분할 속성과 기준을 결정한다. 이렇게 모든 공간의 모든 속성을 고려하는 것이 아니라, 서로 다른 속성들을 고려하도록, 분류기가 서로 다른 **부분공간**만을 사용하도록 설계한다.

배깅 알고리즘

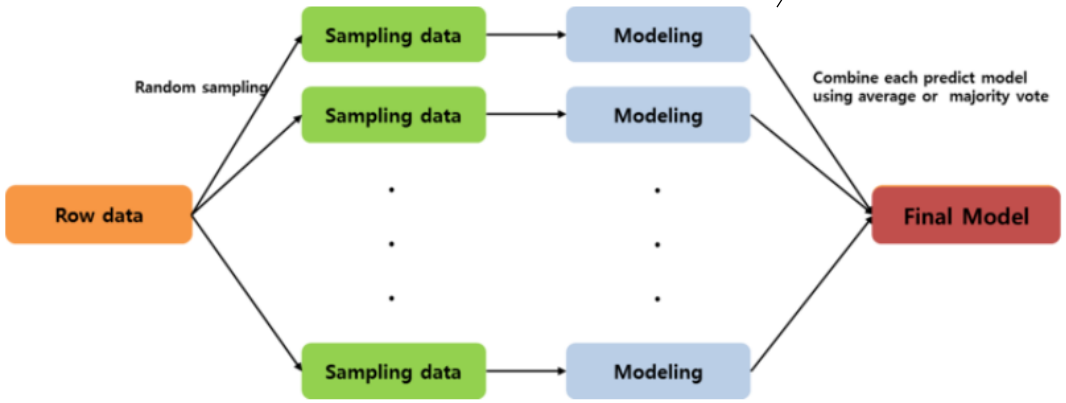
데이터가 충분히 큰 경우, 각 데이터가 하나의 붓스트랩 표본에서 제외될 확률은 36.78%이다

($\lim_{n\to\infty}\left(1-\frac{1}{n}\right)^n = e^{-1} = 0.3678$).

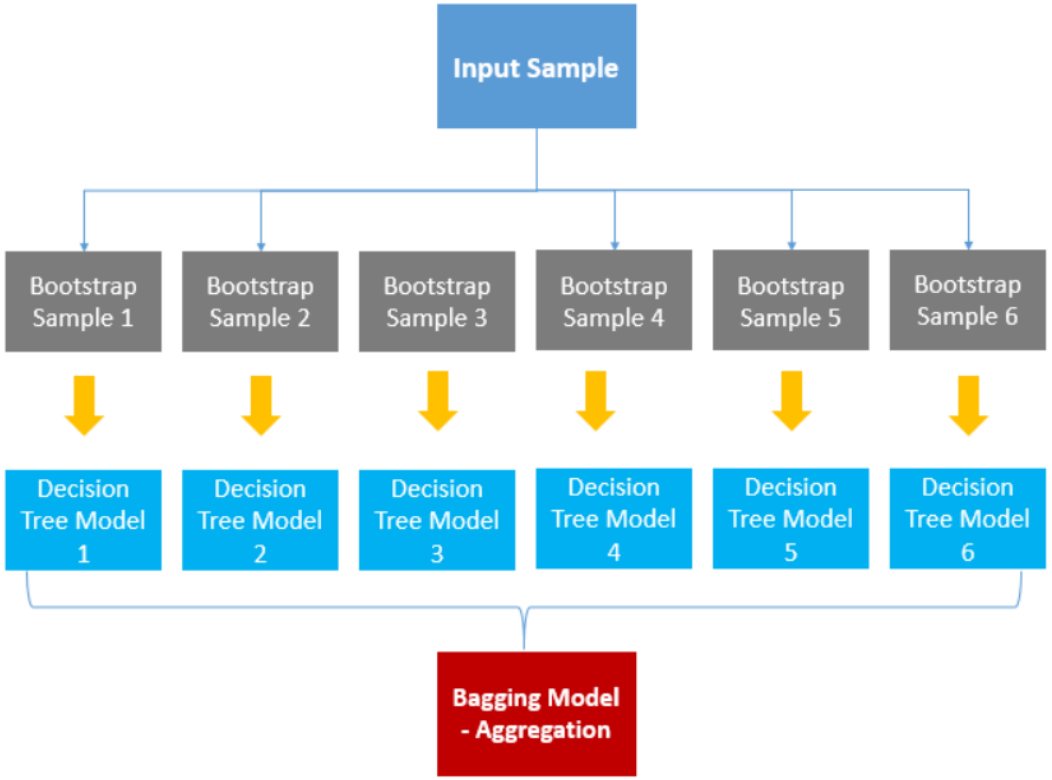
일반적인 Bagging 알고리즘

배깅(bagging)은 bootstrap aggregating의 준말로서 **주어진 데이터에 대해서 여러 개의 붓스트랩(bootstrap) 자료를 생성하고 각 붓스트랩 자료를 모델링(modeling)한 후 결합하여 최종의 예측 모형을 산출하는 방법입니다.** 여기서 **붓스트랩 자료란 단순 복원 임의 추출법(random sampling)을 통해 원자료(raw data)로부터 크기가 동일한 여러 개의 표본 자료를 말합니다.**

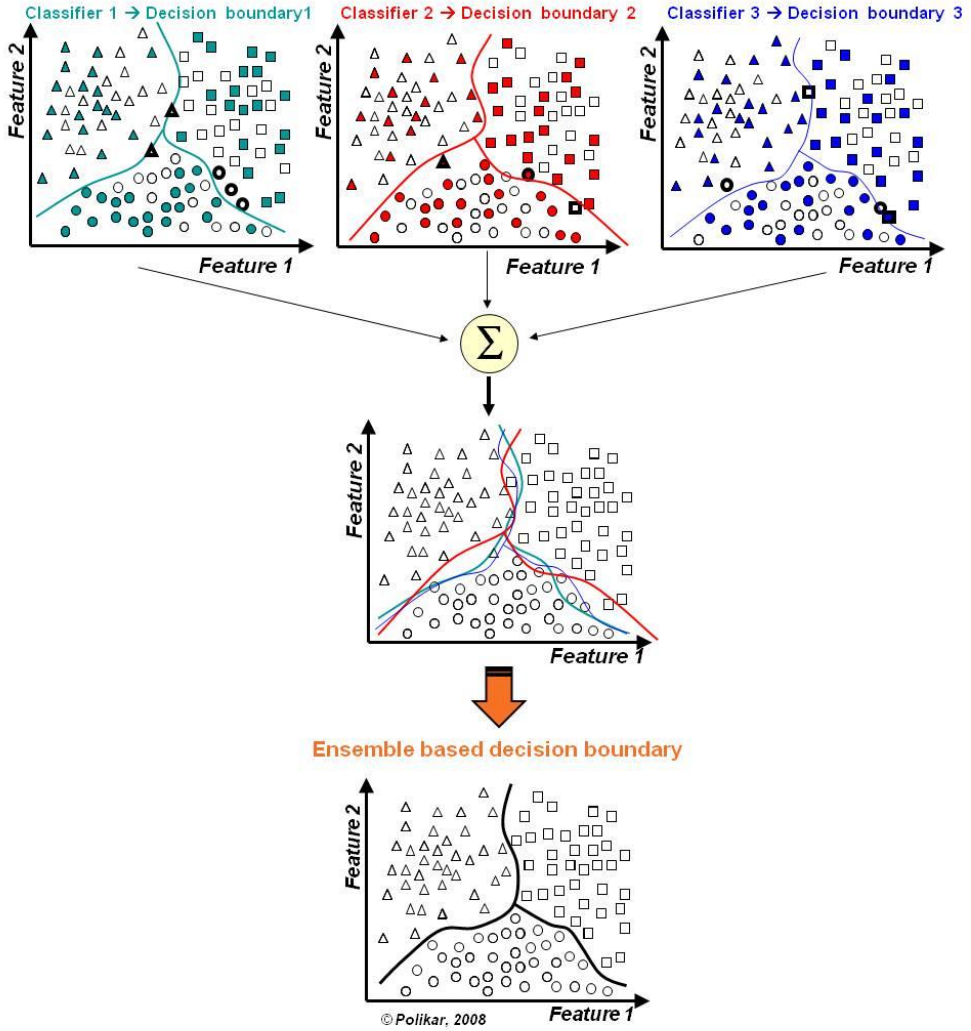
배깅은 예측 모형의 변동성이 큰 경우 예측모형의 변동성을 감소시키기 위해 사용됩니다. 즉, 원자료로부터 여러 번의 복원 샘플링(sampling)을 통해 예측 모형의 분산을 줄여 줌으로써 예측력을 향상 시키는 방법을 배깅이라고 합니다. 따라서 배깅은 일반적으로 과대적합 된 모형, 편의가 작고 분산이 큰 모형에 사용하는 것이 적합합니다.



Decision Tree 에의 적용



배깅 알고리즘: Random Forest Model



▷ Random Forest

:: 분류기로 결정트리를 사용하는 배깅 기법.
결정트리를 만들어가는 과정에서 분할 속성을 결정할 때, **분할 속성 후보**들을 무작위로 선택한 다음, 이들에 대해서만 **정보 이득** 등을 계산하여 분할 속성과 기준을 결정.
즉, 모든 공간의 모든 속성을 고려하는 것이 아니라, 서로 다른 속성들을 고려하도록 분류기가 서로 다른 **부분공간**만을 사용하도록 설계하는 것.
예시로 왼쪽과 같은 그림을 보면 각 Data에 대해 서로 다른 세 개의 기준으로 이들을 나눈 것을 확인할 수 있다. 기준 속성은 Classification의 항목이 될 수 있다.

▶일반적으로 Categorical Data인 경우 투표 방식 (Voting)으로 집계

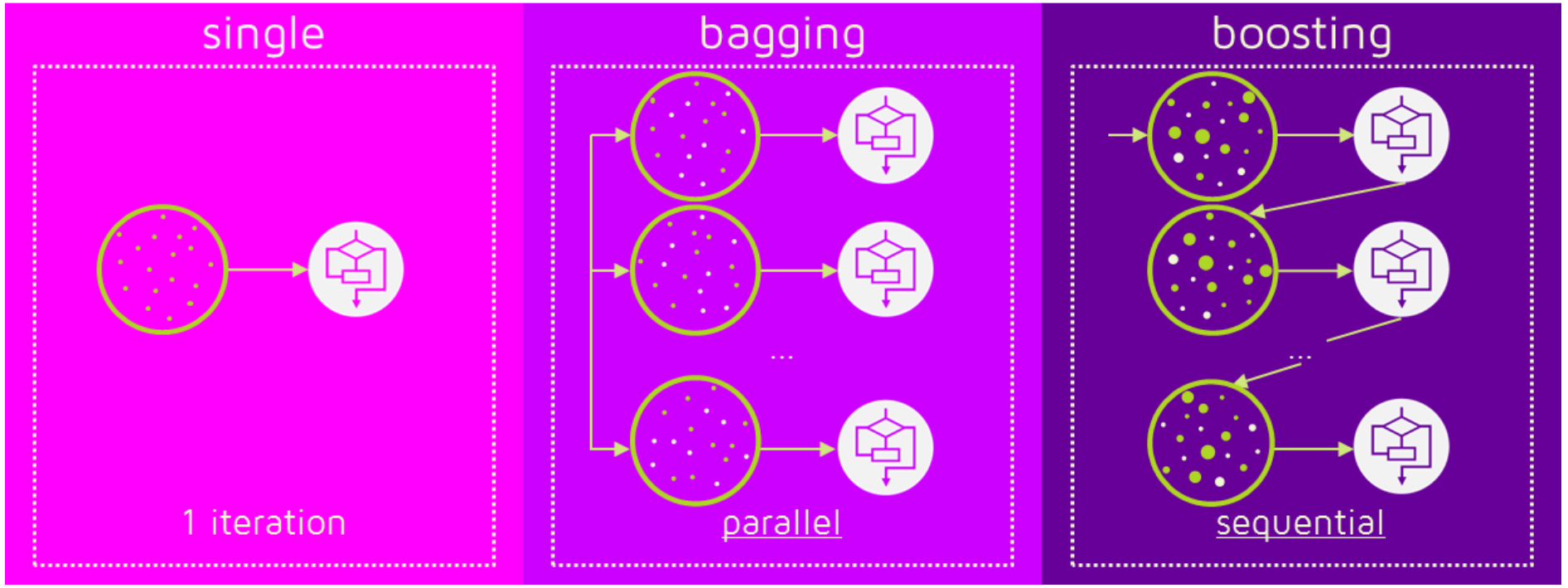
▶Continuous Data인 경우 평균(Average)으로 집계

▶본 KAI.Lib Model에 대해서는 Coefficient Matrix Model을 가중치를 두어 평균을 두는 방식을 적용하면 Overfitting을 피하는 방법이 가능하지 않을까라는 생각을 해봄. 실제 3등팀이 적용한 방식도 이와 같을 것이라 생각함.

부스팅 알고리즘

▷ Booting Algorithms

:: 여러 개의 분류기를 순차적으로 만들어 가는 앙상블 분류기 생성 방법. 여러 학습 데이터 집합을 만드는 것이 아니라, 각 학습 데이터의 가중치를 변경해 가면서 분류기를 만든다. 그리고 가중치는 학습 데이터 각각에 대한 오류를 계산하여 결정된다. 즉, 부스팅은 이전 단계의 분류기에서 잘못 분류된 데이터의 가중치는 높이고 제대로 분류된 데이터의 가중치를 낮춘다.



부스팅 알고리즘

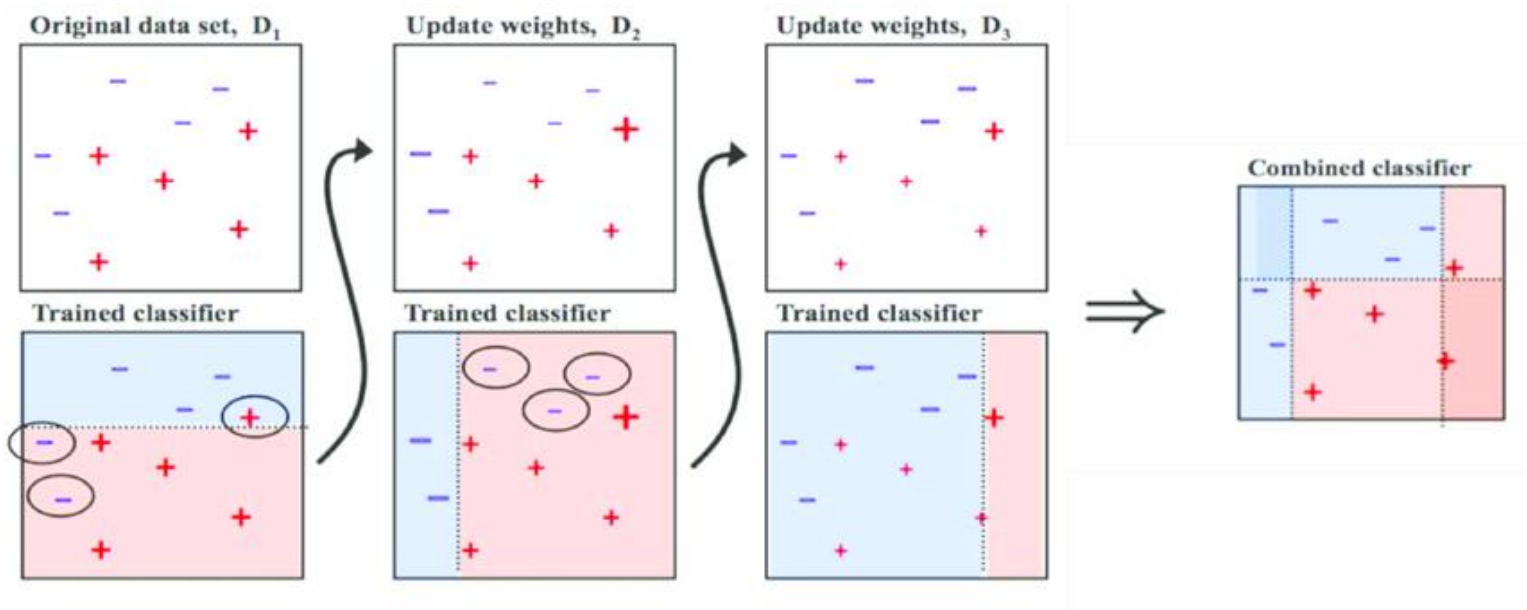
Boosting도 Bagging과 동일하게 복원 랜덤 샘플링을 하지만, 가중치를 부여한다는 차이점이 있습니다. Bagging이 병렬로 학습하는 반면, Boosting은 순차적으로 학습시킵니다. 학습이 끝나면 나온 결과에 따라 가중치가 재분배됩니다.

오답에 대해 높은 가중치를 부여하고, 정답에 대해 낮은 가중치를 부여하기 때문에 오답에 더욱 집중할 수 있게 되는 것 입니다. Boosting 기법의 경우, 정확도가 높게 나타납니다. 하지만, 그만큼 Outlier에 취약하기도 합니다.

AdaBoost, XGBoost, GradientBoost 등 다양한 모델이 있습니다. 그 중에서도 XGBoost 모델은 강력한 성능을 보여줍니다. 최근 대부분의 Kaggle 대회 우승 알고리즘이기도 합니다.

부스팅 알고리즘 예시

▷+ 와 - 로 구성된 데이터셋을 분류하는 문제



▶D1에서는 2/5 지점을 횡단하는 구분선으로 데이터를 나눔. 하지만 위쪽 +는 잘못 분류되었고. 아래쪽 두 - 잘못 분류되었음. 잘못 분류된 데이터는 가중치를 높여주고 잘 분류된 데이터는 가중치를 낮춰 줌

▶D2를 보면 D1에서 잘 분류된 데이터는 크기가 작아졌고. 잘못 분류된 데이터는 크기가 커짐. D3에서는 가중치가 큰 - 3개와 + 1개를 나누는 Classification을 진행하고. 최종적으로 결합하여 Combined Classifier를 형성한다.

▶부스팅은 배깅에 비해 error가 적다. 그러나 좋은 성능을 얻음과 동시에 속도가 느리고 오버 피팅의 가능성이 있다. 상황에 따라 낮은 성능이 문제일 때는 Boositing Algoritm. Overfitting Problem에는 Bagging 기법을 사용하면 된다.

스태킹 알고리즘

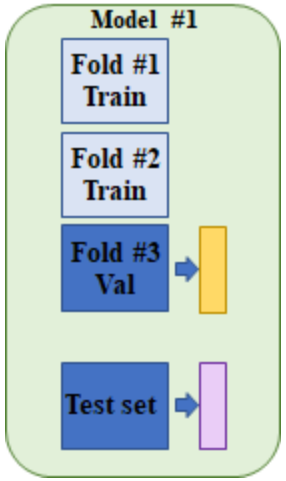
설명

먼저 스태킹은 'Stacked generalization'이라는 제목으로 publish 된 David H. Wolpert의 논문 [1]에서 출발한 개념인데, 줄여서 Stacking이라고 칭하는 경우가 대부분이다. 캐글에서는 모델의 성능을 끌어올리기 위해서 앙상블 기법들을 다양하게 활용한 커널들을 찾아 볼 수 있는데, 이 중 스태킹 기법을 이용하는 경우도 굉장히 많다. 그만큼 스태킹 앙상블이 모델의 성능 향상에 많은 도움을 줄 수 있음을 시사하는 것이다.

- **Stacking은 서로 다른 모델들을 조합해서 최고의 성능을 내는 모델을 생성하는 것.**
- **여러가지 다른 모델의 예측 결과값을 다시 학습 데이터로 만들어서 다른 모델(메타모델)로 재 학습시켜 결과를 예측하는 방법**
- **SVM, Random Forest, KNN 등 다양한 알고리즘을 사용할 수 있고, 이들 알고리즘의 조합을 통해 서로의 장점은 취하고 약점은 보완할 수 있게 되는 것이다.**
- **그러나 어마어마한 연산량을 초래한다.**

스태킹(Stacking)

▷cross-validation 기반 스태킹



간단하고 쉽게 한번 살펴보자. CV fold는 3으로 고정하였고, test set은 별도로 존재하는 것으로 가정한다.
그림 1과 같이, 한 모델이 있다. train fold를 이용하여 1번 모델을 훈련하고, validation fold를 이용하여 모델 예측 값을 계산해 낸다.

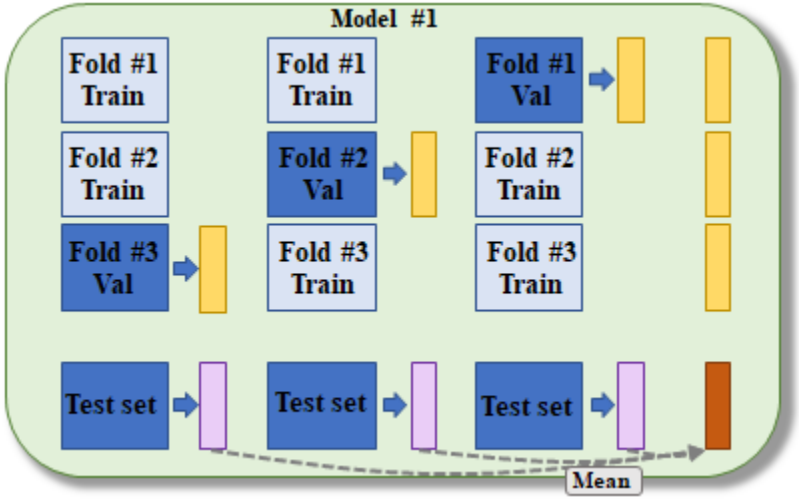
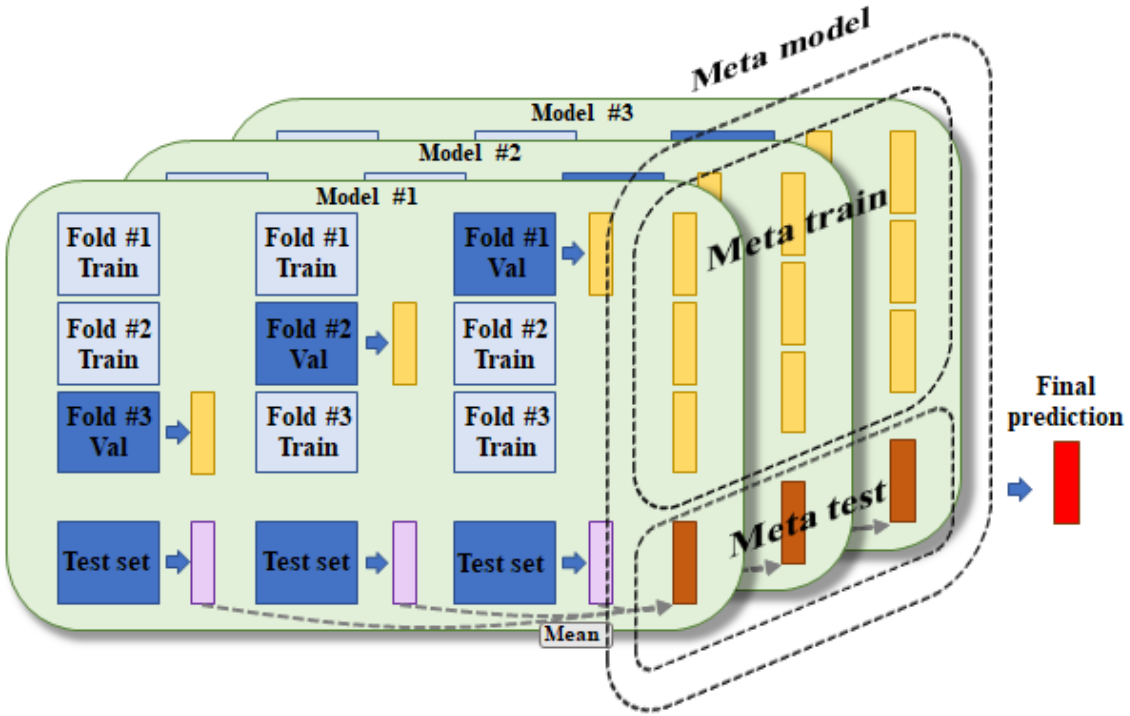


그림 2와 같이 각 valdiation fold 마다 예측 값을 계산하여 이를 concatenate 시키고, test set을 이용한 예측 값 3개(그림 2의 보라색 박스)는 평균을 내어 meta 모델에 사용되는 test set으로 만든다. 그림에서 정렬된 3개의 노란 박스는 결국 한 모델에서 계산되어진, 모든 샘플에 대한 예측 값들이 된다.

스태킹(Stacking)



여기까지의 과정을 다수의 모델에 대해서 반복하고, 그 출력 값들을 concatenate 하게 되면 결국 meta train 데이터와 meta test data를 얻게 된다. 그림에서 보는 것과 같이, meta train 데이터의 dimension은 test set을 제외한 총 샘플수 x 모델 갯수가 될 것이고, meta test 데이터의 dimension은 test set의 샘플 수 x 모델 갯수가 될 것이다. 이를 통해 meta 모델이라고 불리는 2차 모델이 훈련 및 최종 예측 값 출력을 하게 되는 것이다.

앙상블 알고리즘

- 앙상블 방법은 개별 모형에 비해 다음의 장점을 가진다.
  - 평균을 취함으로써 편의를 제거해준다: 치우침이 있는 여러 모형의 평균을 취하면, 어느 쪽에도 치우치지 않는 결과(평균)를 얻게 된다.
  - 분산을 감소시킨다: 한 개 모형으로부터의 단일 의견보다 여러 모형의 의견을 결합하면 변동이 작아진다.
  - 과적합의 가능성을 줄여준다: 과적합이 없는 각 모형으로부터 예측을 결합(평균, 가중 평균, 로지스틱 회귀)하면 과적합의 여지가 줄어든다.