디지털 영상처리 연구실 연구보고서

김우헌

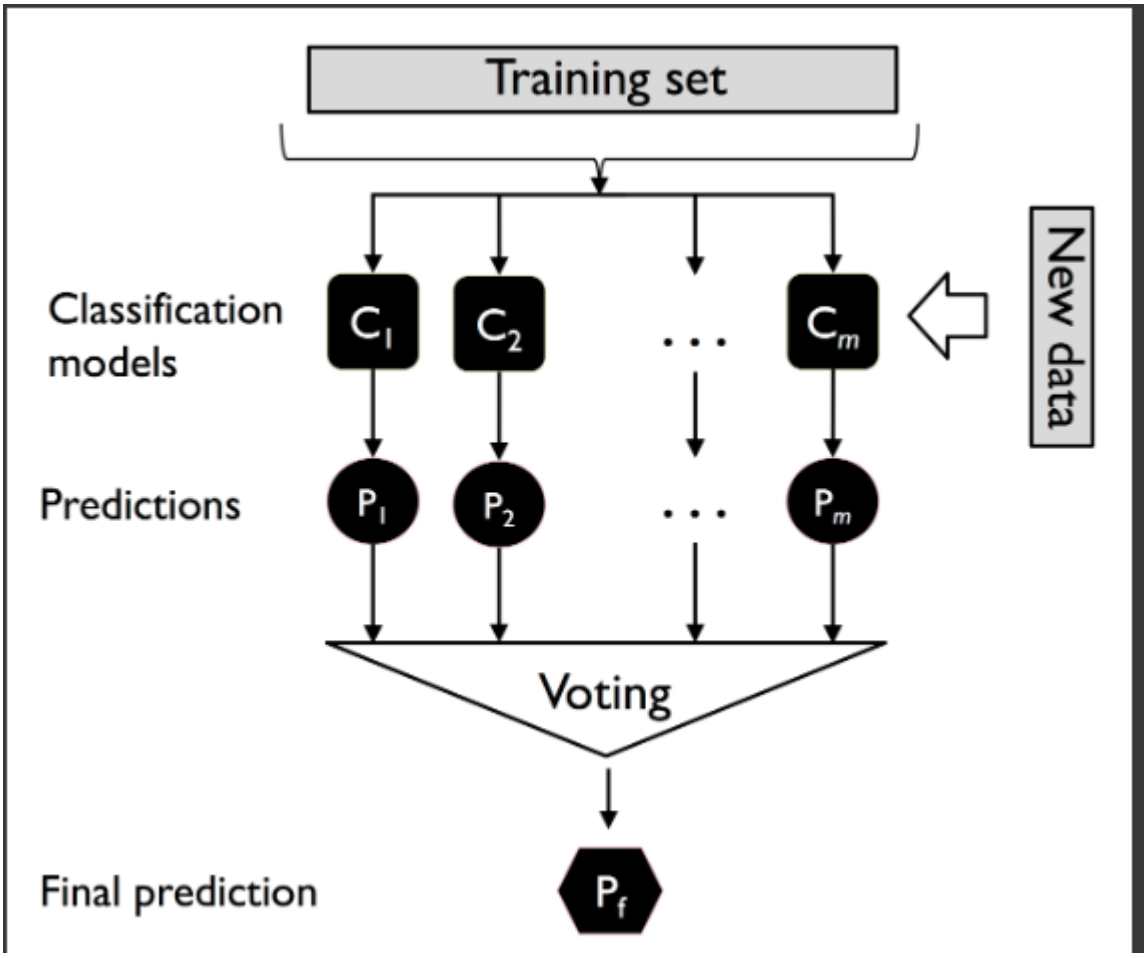
#ensemble

->여러 학습 알고리즘을 조합하여 보다 정확한 예측을 도출

->서로다른 모델을 결합

->보팅,스태킹,배깅,부스팅

##보팅(voting)



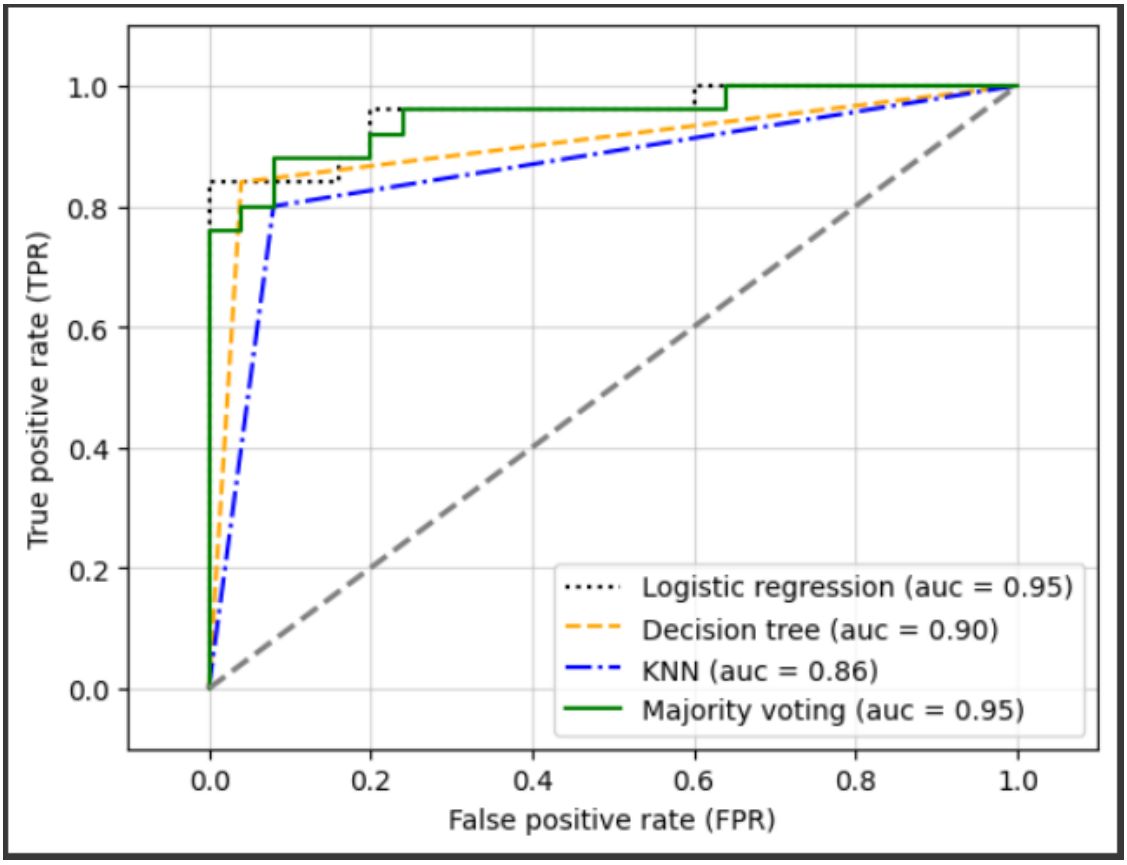
#hard voting? Soft voting?

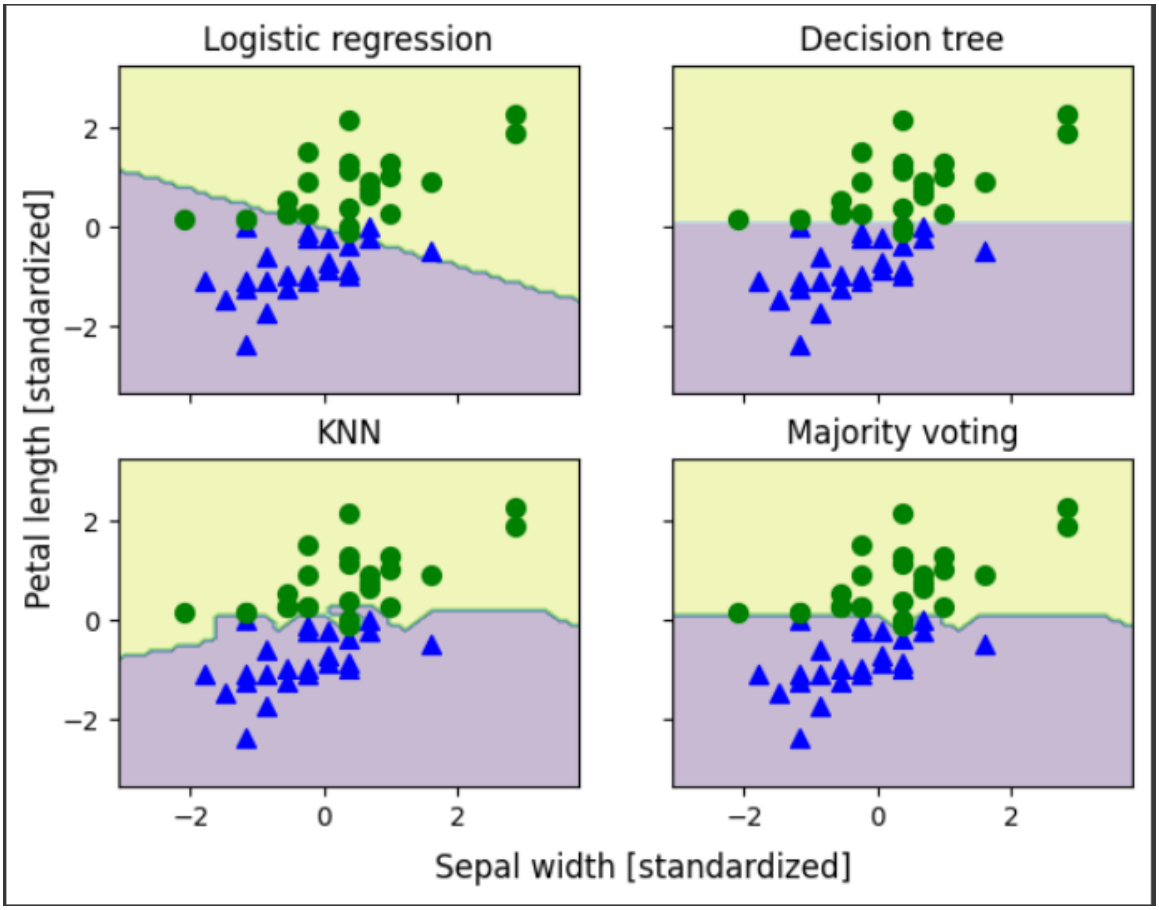
[0.2, 0.8], [0.51, 0.49], [0.55, 0.45] -> [0.42, 0.58]

```
clf1 = LogisticRegression(penalty='l2',
                          C=0.001,
                          random_state=1)

clf2 = DecisionTreeClassifier(max_depth=1,
                              criterion='entropy',
                              random_state=0)

clf3 = KNeighborsClassifier(n_neighbors=1,
                            p=2,
                            metric='minkowski')

pipe1 = Pipeline([['sc', StandardScaler()],
                  ['clf', clf1]])
pipe3 = Pipeline([['sc', StandardScaler()],
                  ['clf', clf3]])
```

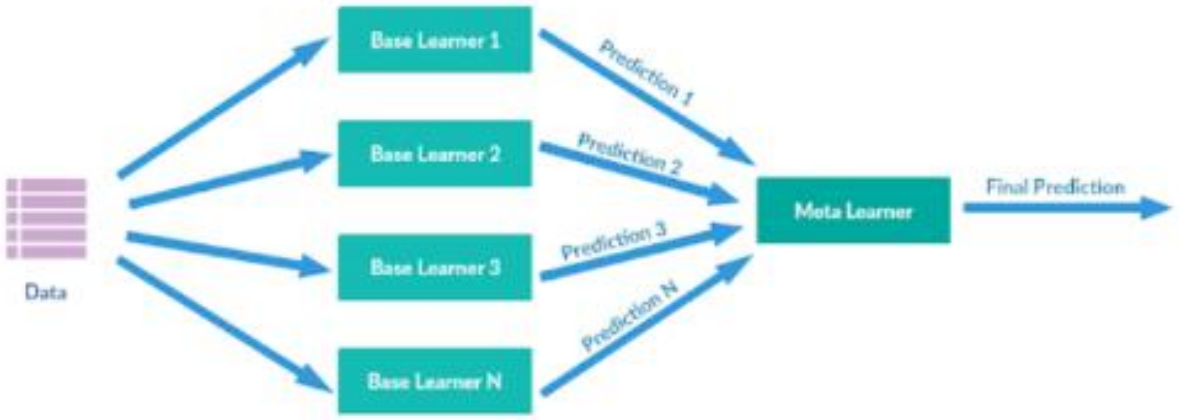
```
vc = VotingClassifier(estimators=[
    ('lr', pipe1), ('dt', clf2), ('knn', pipe3)], voting='soft')

scores = cross_validate(estimator=vc, X=X_train, y=y_train,
                        cv=10, scoring='roc_auc')
```

```
ROC AUC: 0.92 (+/- 0.15) [Logistic regression]
ROC AUC: 0.87 (+/- 0.18) [Decision tree]
ROC AUC: 0.85 (+/- 0.13) [KNN]
ROC AUC: 0.98 (+/- 0.05) [Majority voting]
```


True positive rate (TPR)
False positive rate (FPR)
Logistic regression (auc = 0.95)
Decision tree (auc = 0.90)
KNN (auc = 0.86)
Majority voting (auc = 0.95)


Logistic regression
Decision tree
KNN
Majority voting
Petal length [standardized]
Sepal width [standardized]

##스태킹(stacking)




```
clf1 = LogisticRegression(penalty='l2',
                          C=0.001,
                          random_state=1)

clf2 = DecisionTreeClassifier(max_depth=1,
                              criterion='entropy',
                              random_state=0)

clf3 = KNeighborsClassifier(n_neighbors=1,
                            p=2,
                            metric='minkowski')

pipe1 = Pipeline([['sc', StandardScaler()],
                  ['clf', clf1]])
pipe3 = Pipeline([['sc', StandardScaler()],
                  ['clf', clf3]])
```

```
stacking_lr = StackingClassifier(
    estimators=[
        ('dt', pipe1),
        ('lr', clf2),
        ('knn', pipe3)
    ],
    final_estimator=LogisticRegression()
)
```

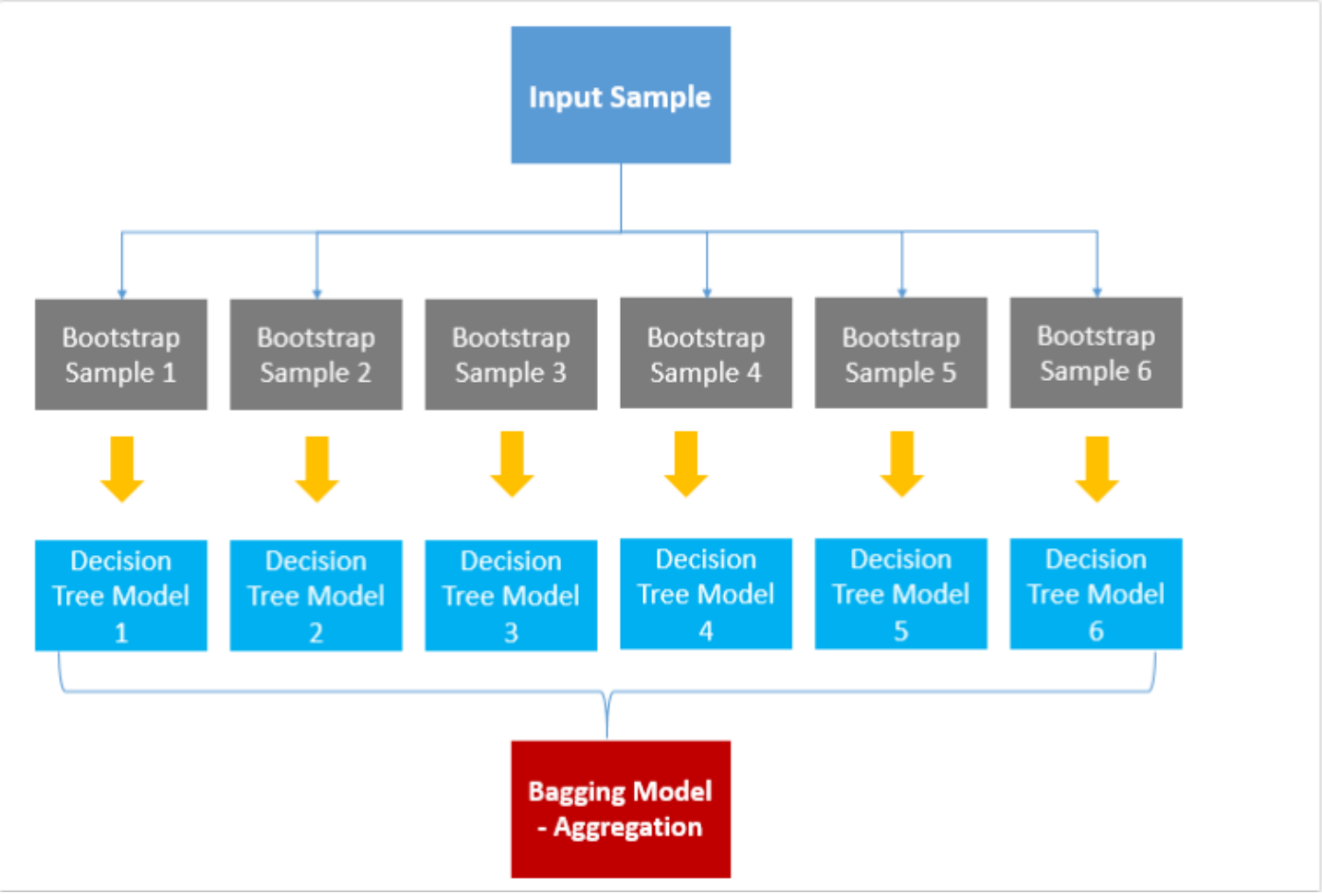
```
ROC AUC: 0.93 (+/- 0.12) [Logistic regression]
ROC AUC: 0.93 (+/- 0.10) [Decision tree]
ROC AUC: 0.89 (+/- 0.10) [KNN]
ROC AUC: 0.96 (+/- 0.06) [Stacking (Logistic Regression)]
ROC AUC: 0.94 (+/- 0.09) [Stacking (Random Forest)]
```

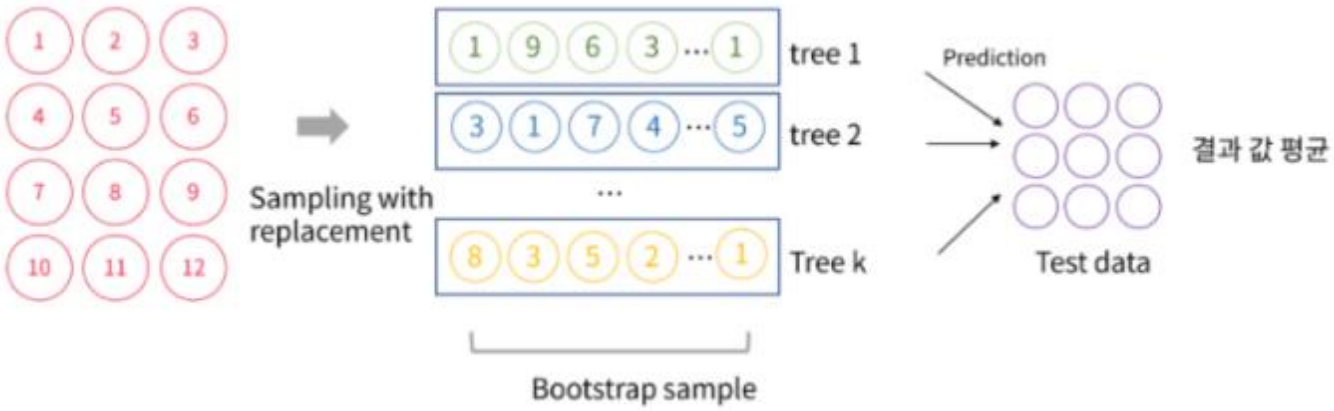
->단일 모델에 비해서 많은 성능 향상이 있지만 더 많은 모델을 사용해야 기대값이 높습니다.

##배깅(Bagging)

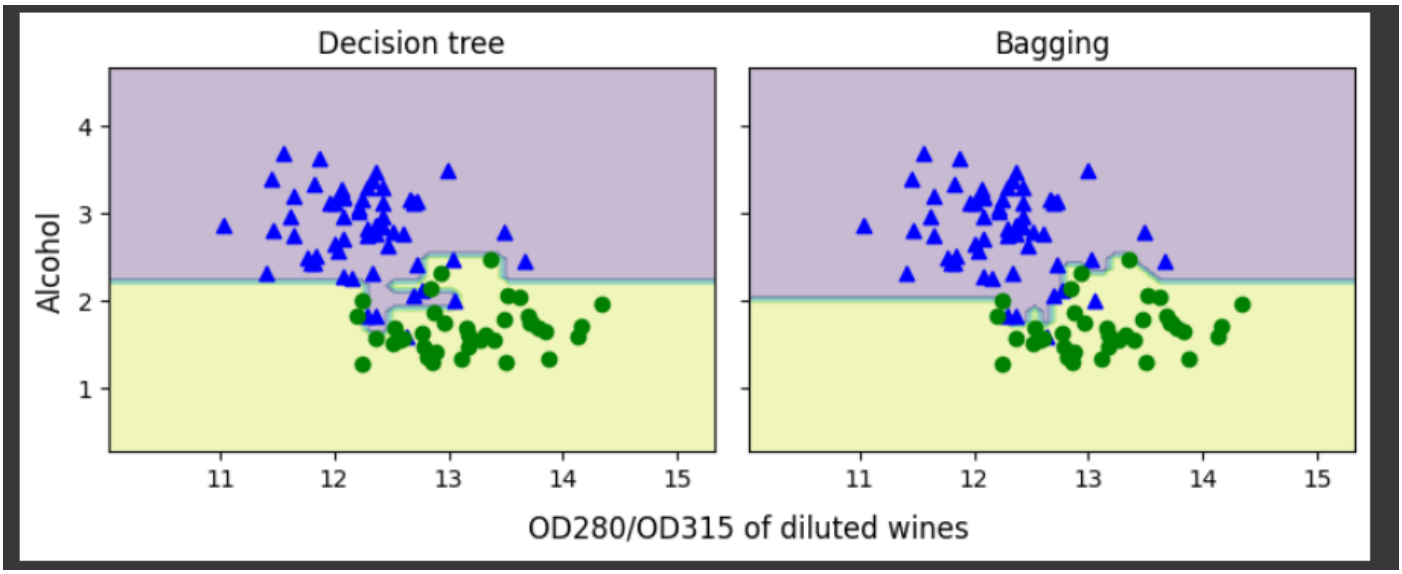
-> Bootstrap Aggregation의 약자

-> Bootstrap=복원 추출을 사용한 표준 추출방법+Aggregation=통합






```
결정 트리의 훈련 정확도/테스트 정확도 1.000/0.833
배깅의 훈련 정확도/테스트 정확도 1.000/0.917
```

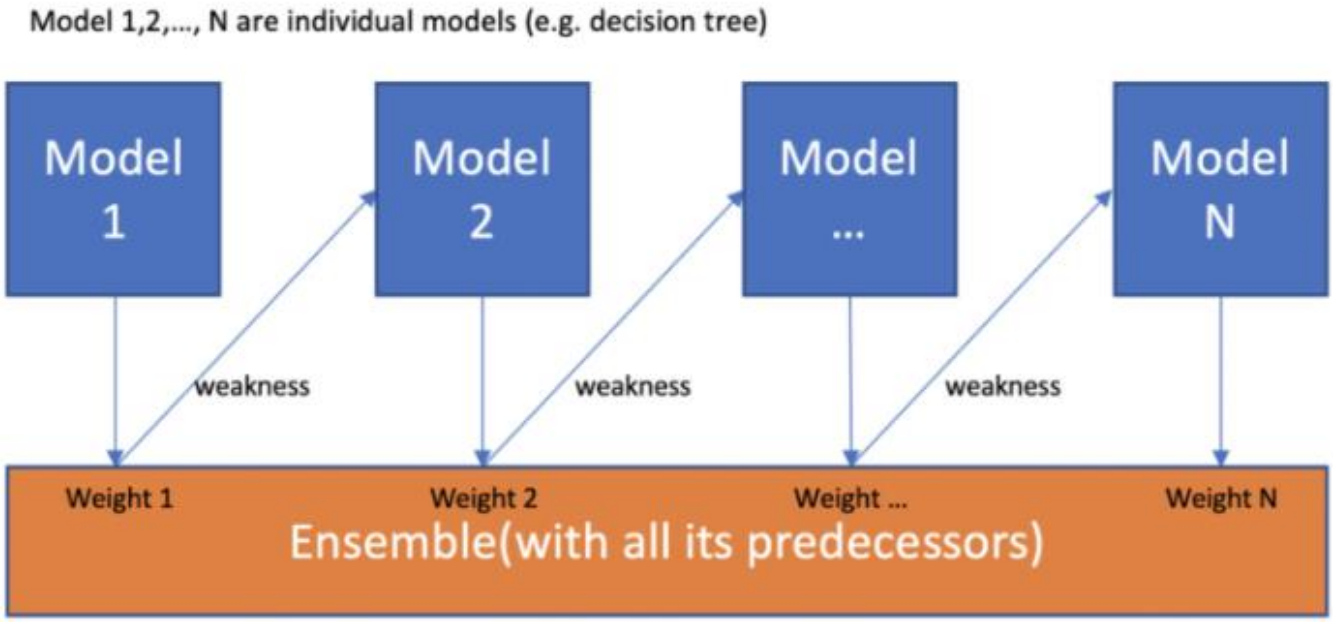


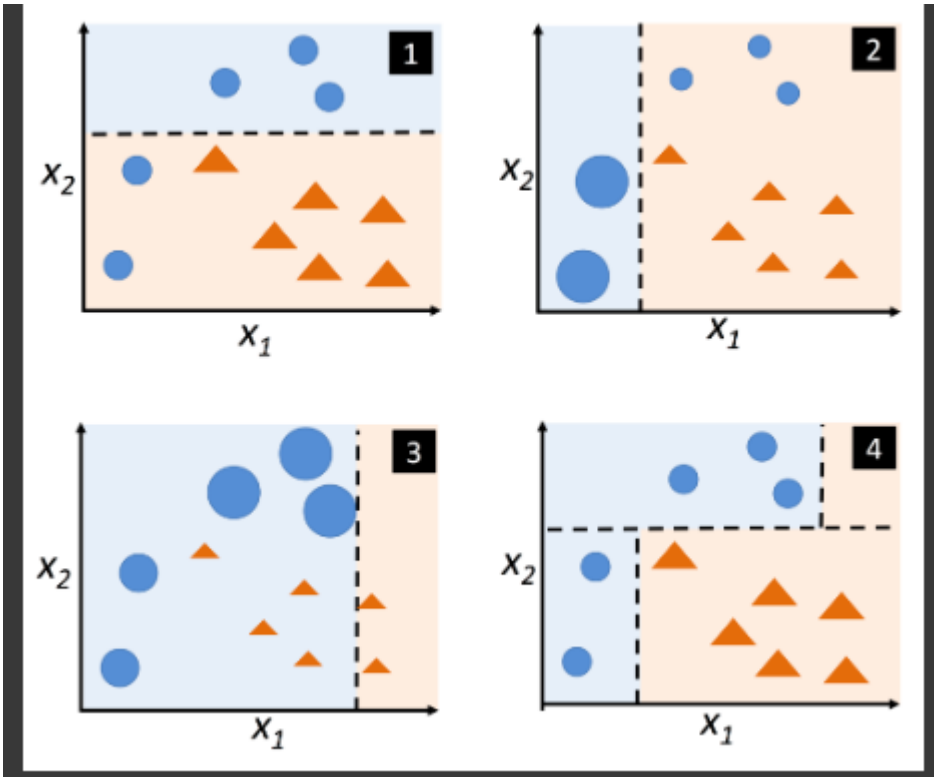
->장점:overfiting에 강함

->단점:데이터의 중복을 허용하기 때문에 특정 샘플이 여러 번 사용되어 편향될 가능성이 존재

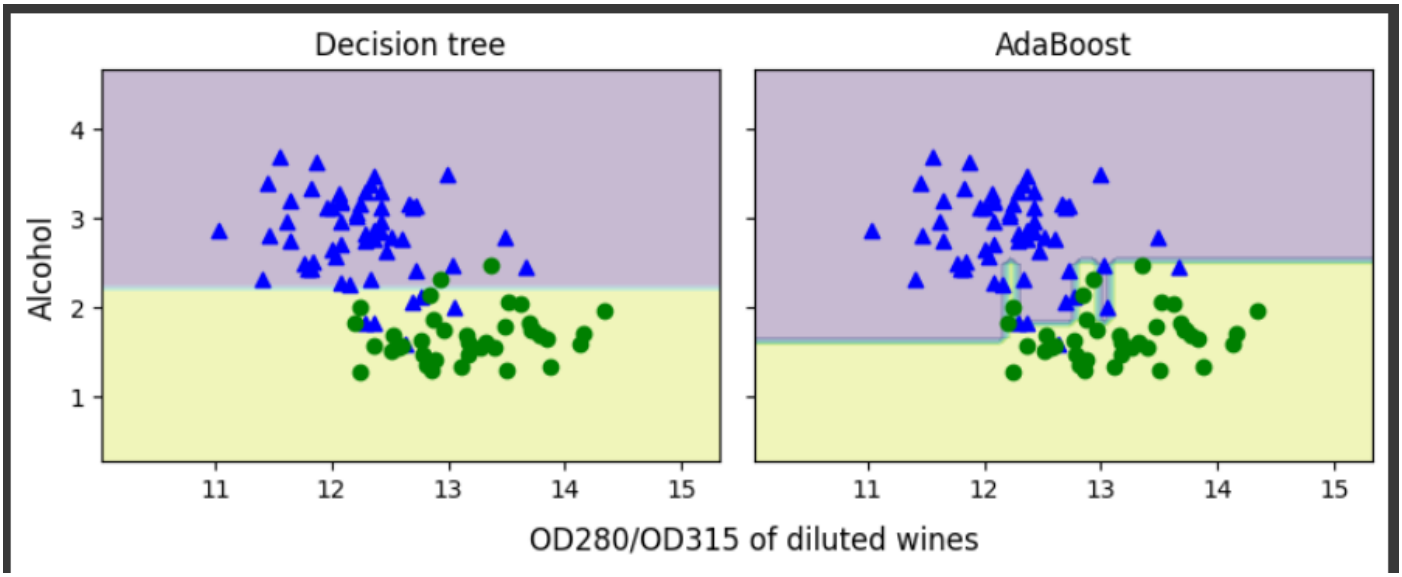
##부스팅(Boosting)

->가중치를 활용하여 약 분류기를 강 분류기로 만드는 방법






```
결정 트리의 훈련 정확도/테스트 정확도 0.916/0.875
에이다부스트의 훈련 정확도/테스트 정확도 1.000/0.917
```



->모델의 성능 향상 but overfiting가능성 존재

#배깅과 부스팅 차이