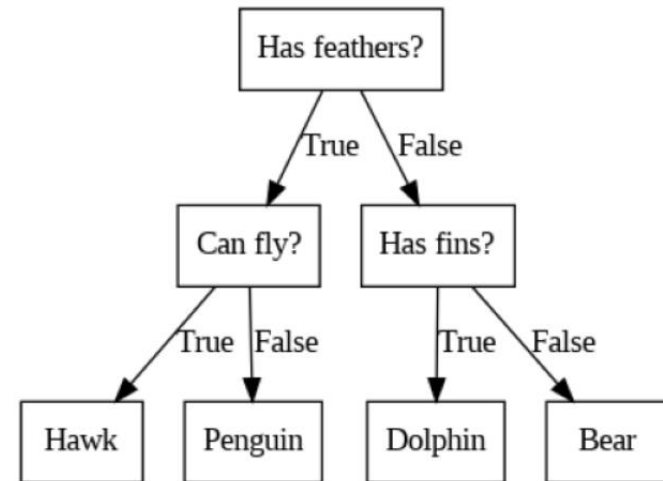


Tree based classification



강의 목표

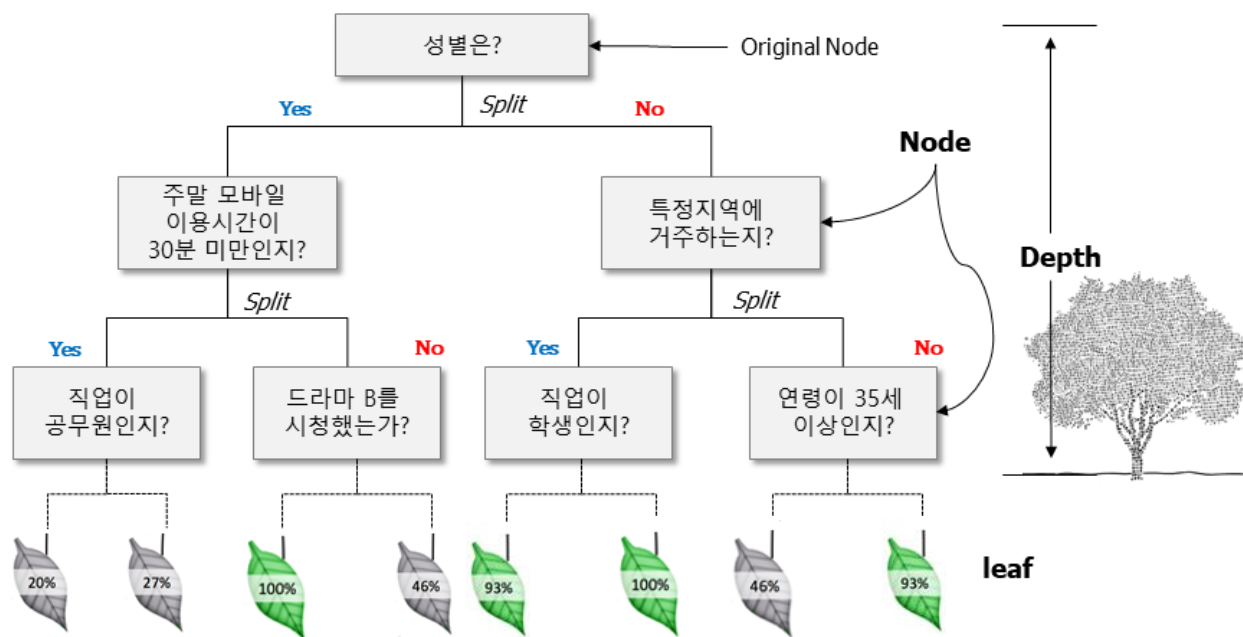
- 분류문제의 해결을 위해 의사결정나무, 배깅, 랜덤 포레스트 모델을 이해하고 파이썬으로 실습

강의 내용	
1. 인공지능의 이해	<ul style="list-style-type: none">▪ 데이터, 알고리즘, 모형의 이해▪ Scikit learn API의 메커니즘 이해 및 실습▪ 분산과 편향, 손실 함수, 최적화▪ 모델의 평가 및 선택
2. 분류/회귀생성 모델	<ul style="list-style-type: none">▪ Logistic Regression & Regression▪ Stochastic Gradient Descent▪ K-Nearest Neighbors
	<ul style="list-style-type: none">▪ Deep learning with Keras▪ Gaussian Naïve Bayes▪ Support vector machine
	<ul style="list-style-type: none">▪ 의사결정나무 (Decision tree)▪ 앙상블 (Ensemble: Bagging, Boosting, Stacking)

의사결정 나무 알고리즘

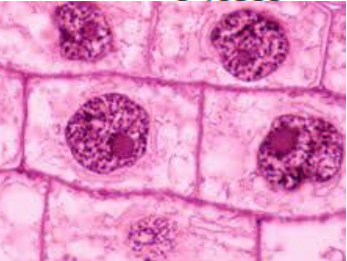
- 가장 적은 질문으로 가장 빠르게 정답에 이르는 스무고개 방식
- 처음 질문의 응답에 따라 다음 질문을 순차적으로 분류 확률이 최대화 될 때까지 질문
- 회귀와 분류 모델에 모두 적용 가능

예제) 성별, 거주지, 직업, 연령, 주말 모방일 이용시간, 드라마장르 선호 등의 정보를 이용하여 상품을 구매할 것인지 아닌지?



데이터, 알고리즘, 모형

- 2개의 특성변수로 적합과 부적합, 이항 분류(Binary Classification) 예측하는데 정확도(Accuracy)를 최대화하는 것을 목적으로 다양한 알고리즘을 적용하여 최적 모형을 개발
- UCI 흉부암(Breast cancer) 세포핵 이미지 자료에서 50개를 예측용 데이터로 사용

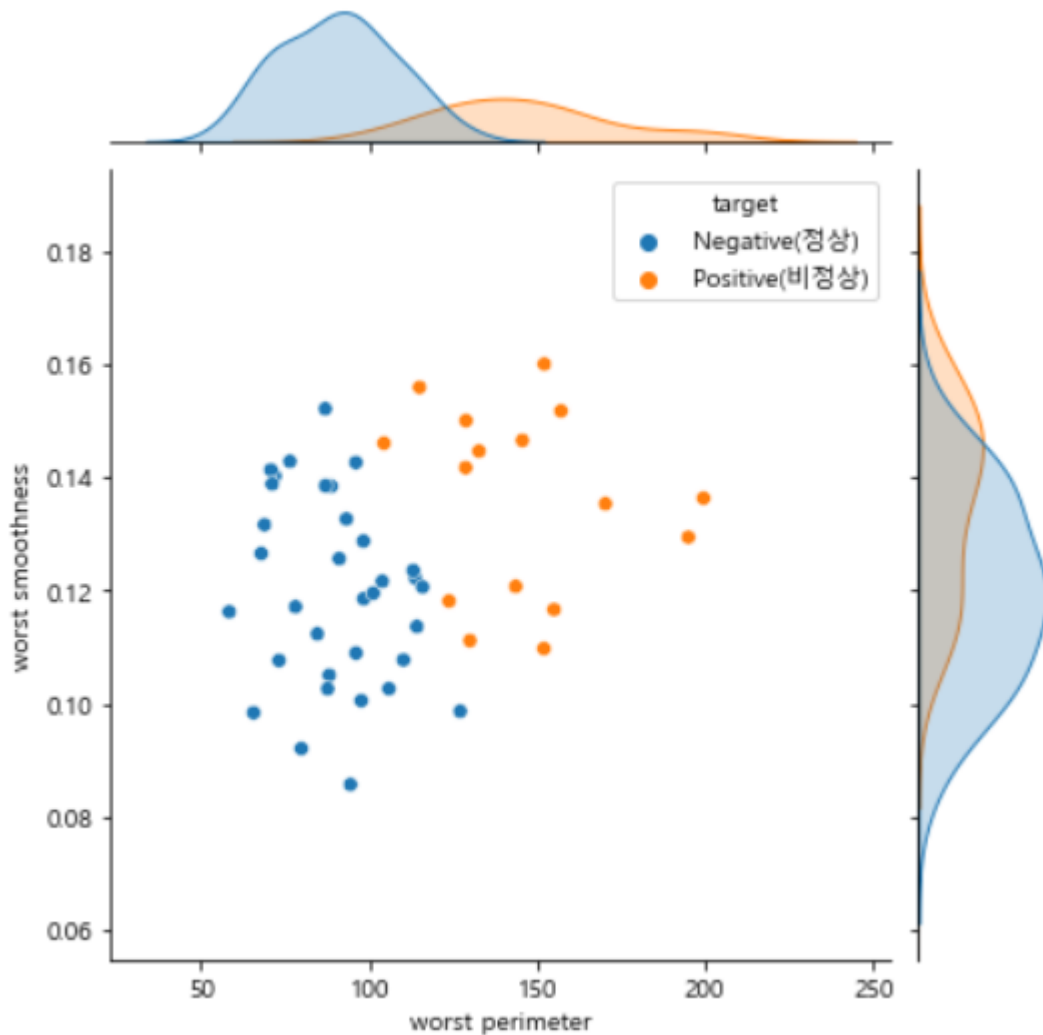
Input (X)		Output (y)
worst smoothness	worst perimeter	target
0.16000	152.10	1
0.10050	97.66	0
0.13160	68.81	0
0.11620	58.36	0
0.11810	123.80	1
	93.22	0
	115.90	0
	113.70	0
	67.88	0

$$f : X \rightarrow y$$



- 모형 = 알고리즘(데이터)
- 평가지표 = 정확도

분류를 가장 잘할 수 있는 방법

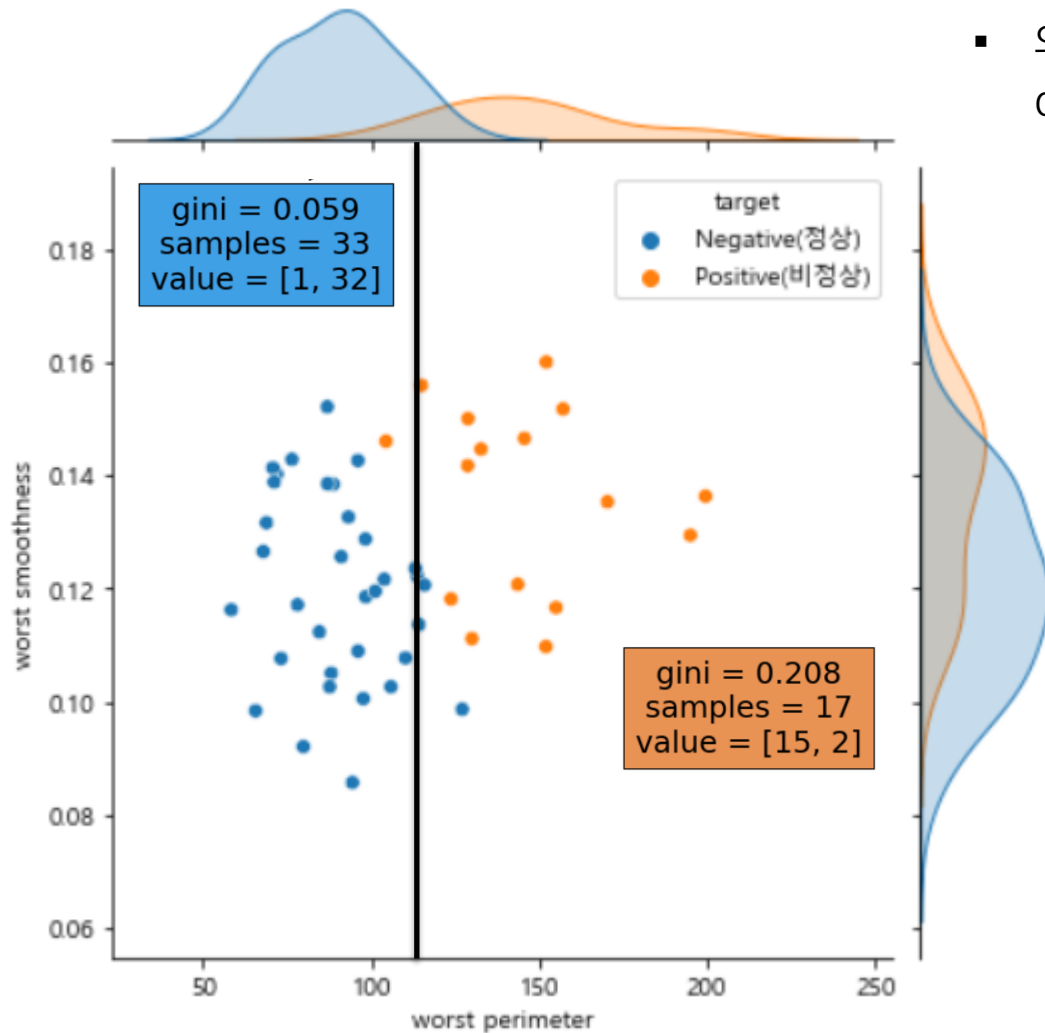


- 어떤 첫번째 질문을 하여야 데이터의 분류(빨강과 파란색)를 가장 잘할 수 있는가 ?

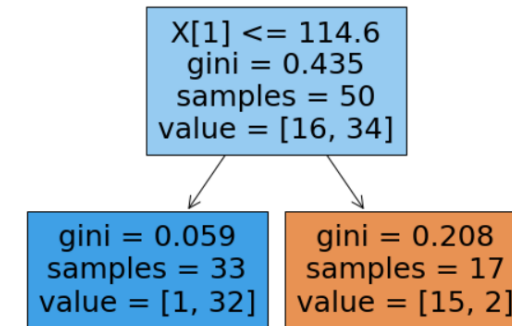
[Level 0 비중]

Target	건수(비율)	예측
정상	34(68%)	Negative (정상)
비정상	16(32%)	

Split Variable and Value_분리 변수와 분리 값의 결정



- 의사결정 나무의 Depth가 '1' 인 경우 (Stump)로 핵심 아이디어는 노드 분리(Node splitting)

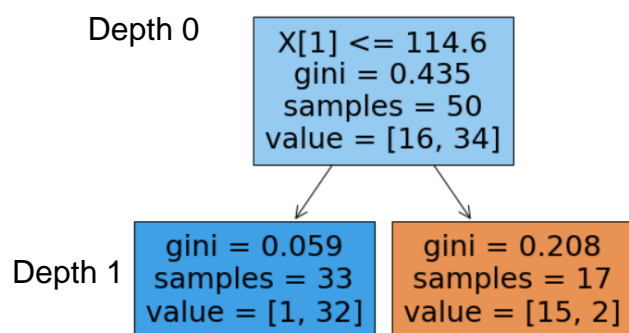


Worst perimeter	Target	건수(비율)	예측
<= 114.6	비정상	1/33(3%)	
	정상	32/33(97%)	정상
> 114.6	비정상	15/17(88%)	비정상
	정상	2/17(12%)	

Gini Index

- 모든 변수를 대상으로 Gini Index를 기준으로 가장 낮은 순으로 가지치기를 단계적 수행
- 이진분류의 Gini Index 수식 (Cross - Entropy와 수치적으로 유사)

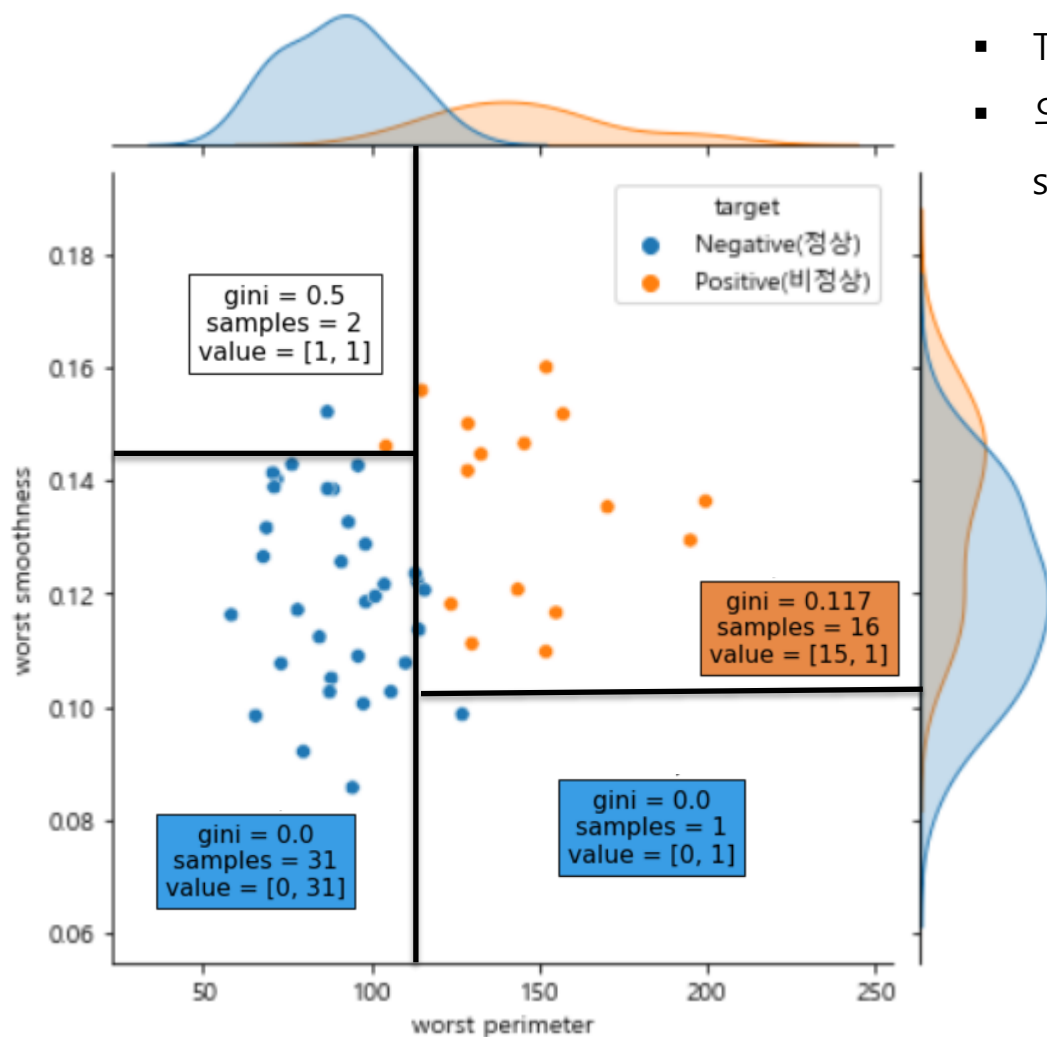
$$\text{지니지수 (Gini Index)} = \sum_{C=1}^2 P_C * (1 - P_C) = 2P_1 * (1 - P_1)$$



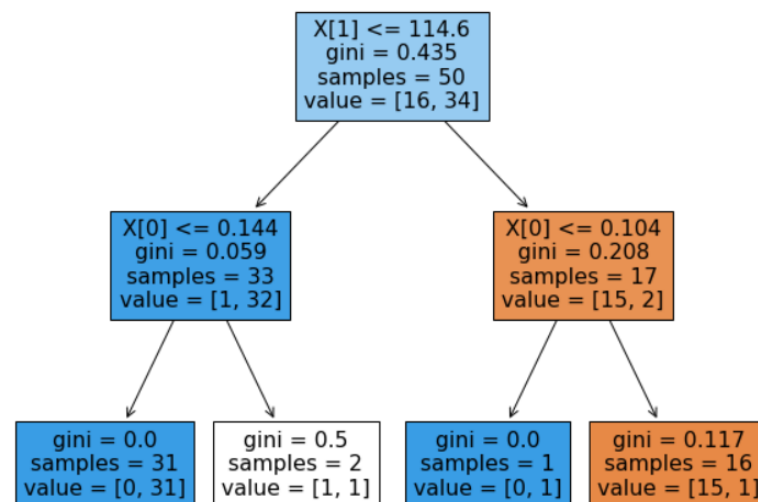
	Red 개수	Blue 개수	소계	a Red 비율	b=(1-a) Blue 비율	2a(1-a) Gini Index
Depth 0	16	34	50	0.32	0.68	0.435
Depth 1_left	1	32	33	0.03	0.97	0.059
Depth 1_right	15	2	17	0.88	0.12	0.208

* 각 노드의 지니 지수는 불순도(Impurity)

Splitting - 가지치기



- Tree의 depth가 '2'
- 의사결정 나무 알고리즘은 데이터 분할, 노드 분리, splitting(가지치기) 으로 Gini Index가 낮춘다.



예
측

정상	?	정상	비정상
100%	50%	100%	88%

데이터

- 수치형 변수 30개와 총 568개의 Instance로 구성된 입력행렬(X)과 예측 변수(target)
- 의사결정나무 → Bagging → Random Forest 모델의 정확도 비교

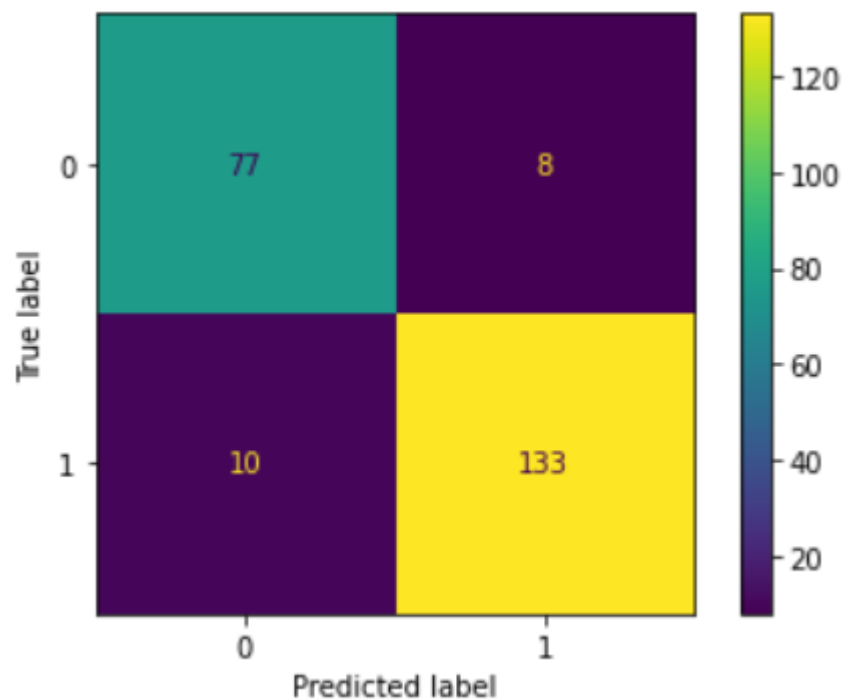
	worst smoothness	worst symmetry	fractal dimension error	worst radius	worst texture	worst perimeter	worst area	worst compactness	worst concavity	worst concave points	worst fractal dimension	target
0	0.16220	0.4601	0.006193	25.380	17.33	184.60	2019.0	0.66560	0.7119	0.2654	0.11890	0
1	0.12380	0.2750	0.003532	24.990	23.41	158.80	1956.0	0.18660	0.2416	0.1860	0.08902	0
2	0.14440	0.3613	0.004571	23.570	25.53	152.50	1709.0	0.42450	0.4504	0.2430	0.08758	0
3	0.20980	0.6638	0.009208	14.910	26.50	98.87	567.7	0.86630	0.6869	0.2575	0.17300	0
4	0.13740	0.2364	0.005115	22.540	16.67	152.20	1575.0	0.20500	0.4000	0.1625	0.07678	0
...
564	0.14100	0.2060	0.004239	25.450	26.40	166.10	2027.0	0.21130	0.4107	0.2216	0.07115	0
565	0.11660	0.2572	0.002498	23.690	38.25	155.00	1731.0	0.19220	0.3215	0.1628	0.06637	0
566	0.11390	0.2218	0.003892	18.980	34.12	126.70	1124.0	0.30940	0.3403	0.1418	0.07820	0
567	0.16500	0.4087	0.006185	25.740	39.42	184.60	1821.0	0.86810	0.9387	0.2650	0.12400	0
568	0.08996	0.2871	0.002783	9.456	30.37	59.16	268.6	0.06444	0.0000	0.0000	0.07039	1

569 rows × 31 columns

의사결정 나무 모형의 오차행렬과 정확도

- 228개 시험데이터, FN, FPR, 썬, Recall, Precision, Accuracy, F1-score

오차 행렬(Confusion Matrix)



$$\text{정확도(Accuracy)} = (77+133)/(85+143) = 92.11\%$$

```
from sklearn.datasets import load_breast_cancer  
X, y = load_breast_cancer(return_X_y=True)
```

1. 데이터 준비

```
from sklearn.tree import DecisionTreeClassifier
```

2. 알고리즘 선택

```
model = DecisionTreeClassifier()
```

3. 알고리즘 객체화

```
model.fit(X, y)
```

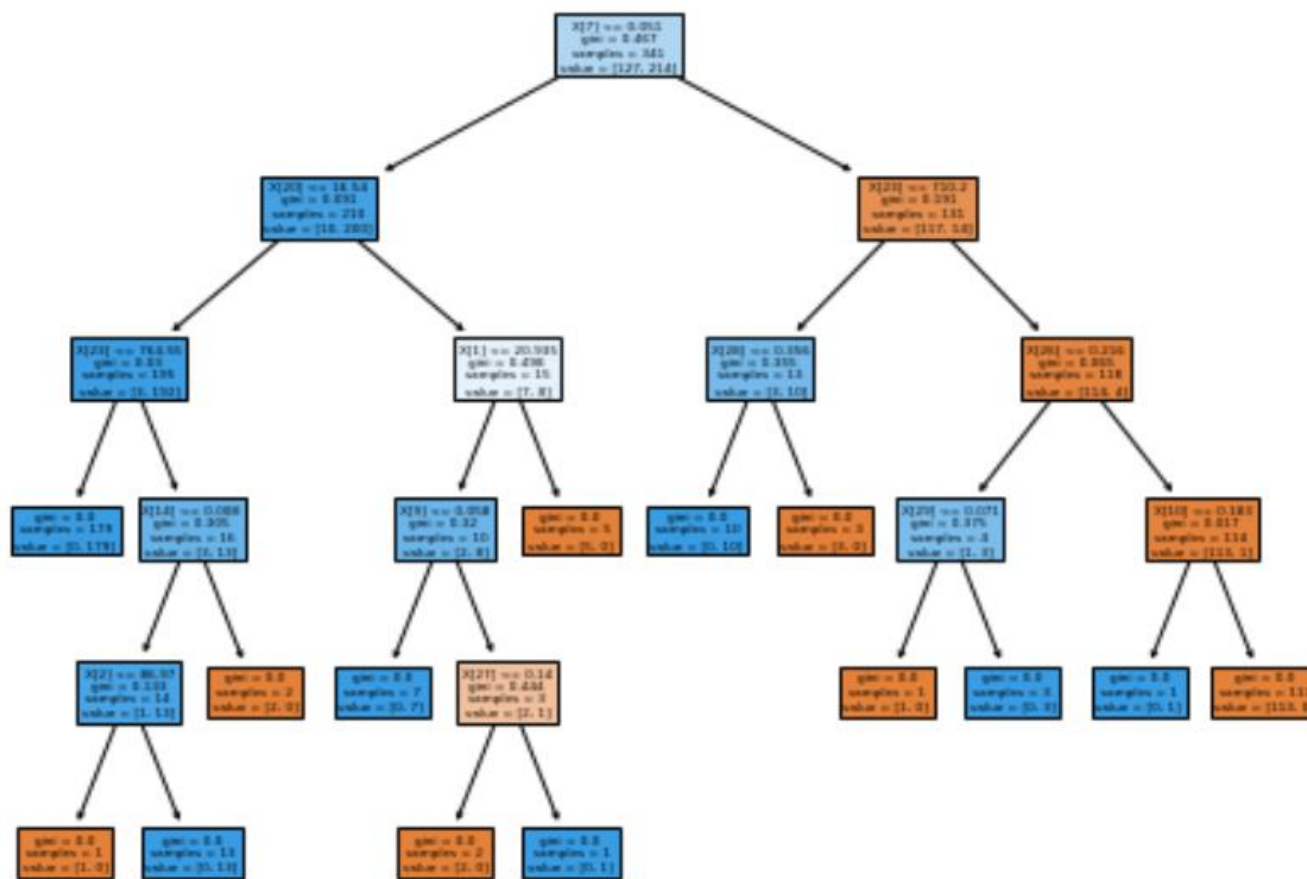
4. 학습(모형)

```
model.predict(X)  
model.predict_proba(X)
```

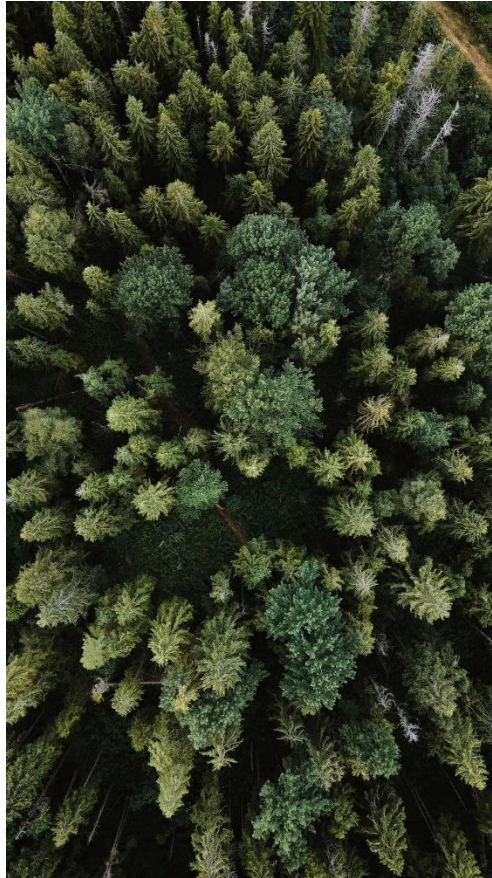
5. 예측

의사결정 나무 모형의 과적합

- 모델의 특성상 자료의 레이블이 완전히 분류될 때까지 splitting(가지치기)를 하여 훈련하게 되어 과적합(Overfitting)이 발생하기 쉬움 → pruning 필요

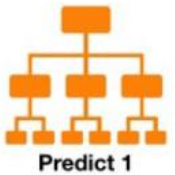
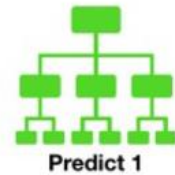
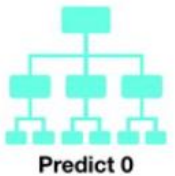
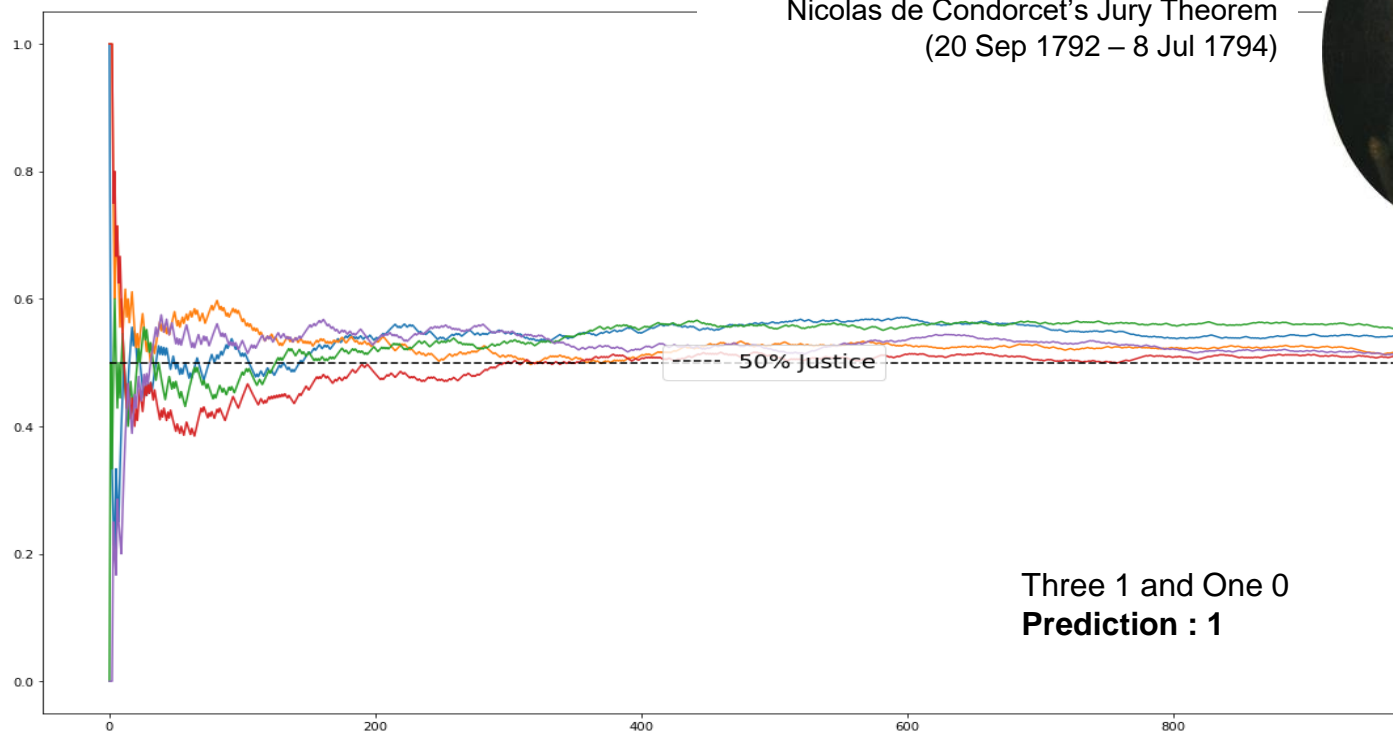


Ensemble Model



배심원 정리

- 근소하게 올바른 판단력을 가진 배심원이 모여서 독립적으로 판결하게 되면 항상 올바른 판단을 내린다(집단 지성, the wisdom of crowds)
- 여러 개의 의사결정 나무를 만들어 다수의 결과로 예측

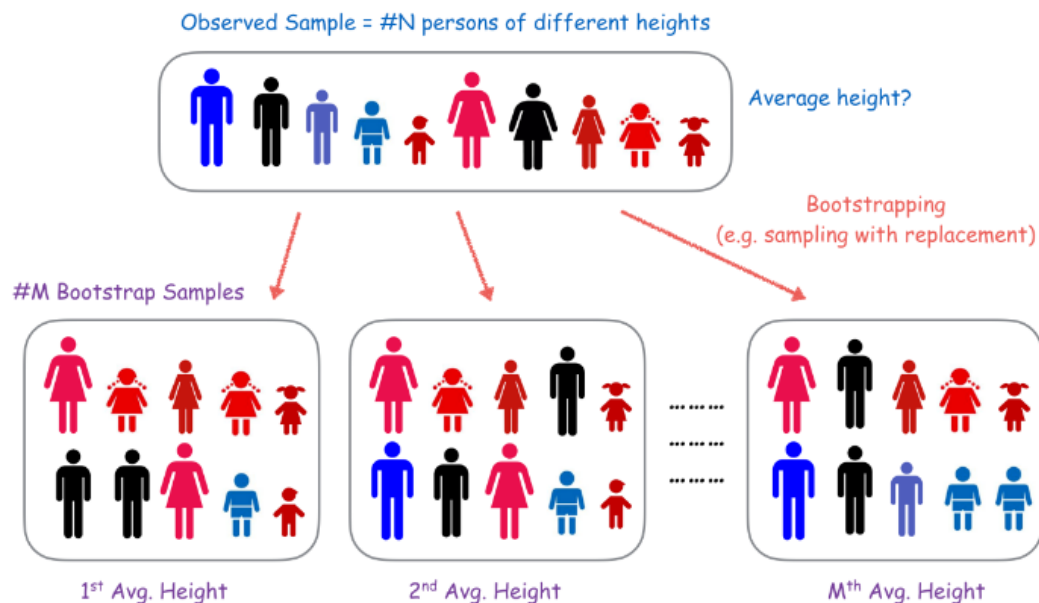


부트스트래핑 샘플링

- 부트스트래핑(Bootstrapping)은 표본의 수가 적거나 추가 자료를 얻기 어려워 통계가설을 적용하기 힘든 경우 확보된 샘플에서 반복적으로 다시 샘플링하는 방법
- 재표본 추출(Resampling) = 부트스트래핑(Boostrapping)
- 표본 N개에서 k개를 복원 추출(중복 선택)하여 모수 추정치의 표준오차를 줄임

예시) 오른쪽 그림의 경우 부트스트래핑 방법

- ✓ N=10개인 표본(Observed Sample)
- ✓ 재표본 수 10개를 복원 추출
- ✓ M회 반복
- ✓ 총 M*10개의 재표본 생성
- ✓ 모수의 평균신장에 대한 표준오차를 줄임



Ensemble Bootstrapping

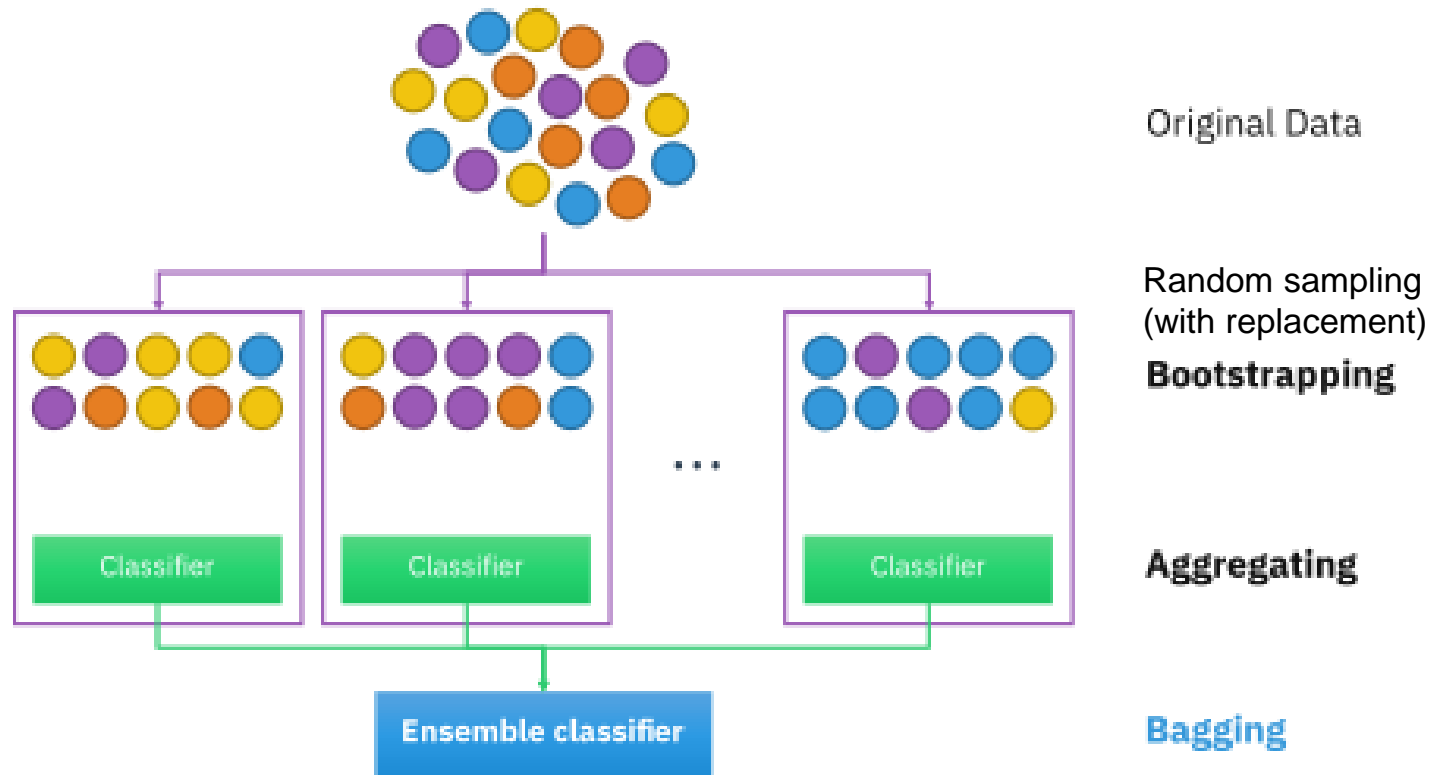
- 표본의 수가 적거나 추가 자료를 얻기 어려워 통계가설이 부재
- 재표본 추출(Resampling) = 부트스트래핑(Boostrapping)
- 표본 N개에서 k개를 복원 추출(중복 선택)하여 모수 추정치의 표준오차를 줄이게 된다.
 - $N=100$ 개인 표본에서 200회 재표본(재표본 수는 100개) 추출하는 부트스트래핑을 적용하면
관측치 수가 100개인 표본 200개를 생성

Data diversity ~ bootstrap aggregating, cross validation



Bagging 모델

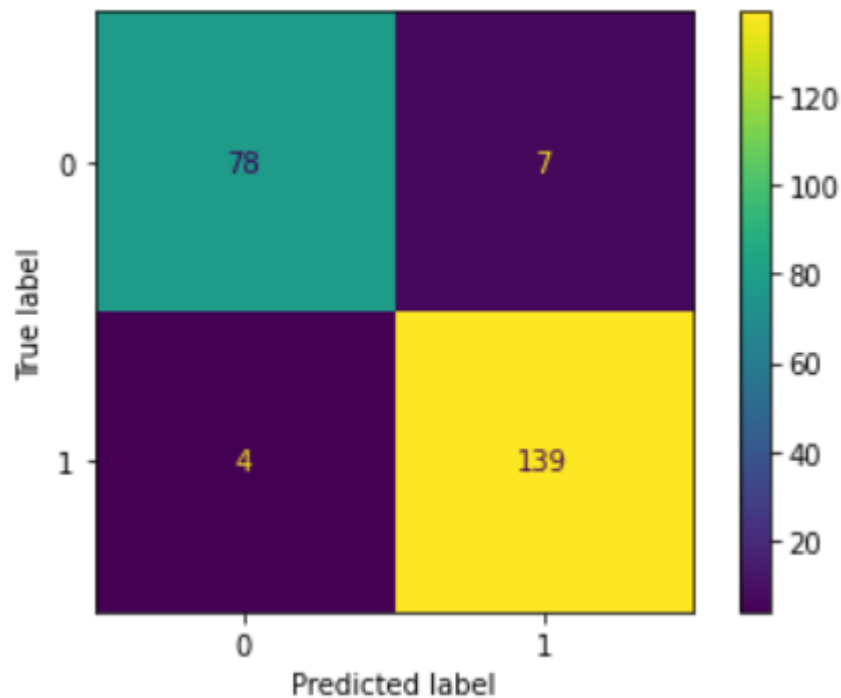
- Bagging(Bootstrapping aggregation)은 bootstrap((Resampling) tree들의 총합(aggregation)
- Bootstrap 표본을 쓰므로 예측 오류의 추정에서 예상외의 효과가 있음



Bagging 모델의 오차행렬과 정확도

- 30개의 의사결정 나무(재 표본 수 30개)와 원표본의 80%만을 붓스트래핑하여 예측을 하면 성능이 높아짐

오차 행렬(Confusion Matrix)



$$\text{정확도(Accuracy)} = (78+139)/(85+143) = 95.18\%$$

```
from sklearn import ensemble
```

```
dt = DecisionTreeClassifier()
```

```
Bag = ensemble.BaggingClassifier(dt,  
                                n_estimators = 30,  
                                max_samples = 0.8)
```

```
model.fit(X, y)
```

```
model.predict(X, y)
```

Random Forest 모델

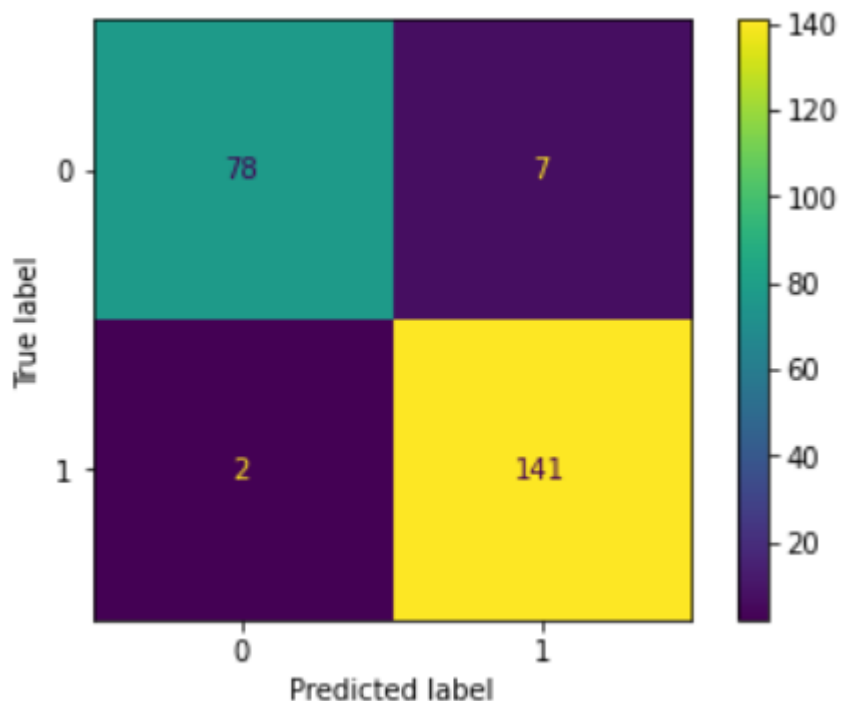
- Bagging은 다수의 tree model들의 총합으로 부분적인 중복성이 모형의 성능을 저하시키는데 이런 문제를 개선한 것이 랜덤 포레스트
 - 1) 원 표본에서 중복을 허용하여 같은 크기의 부표본을 추출하여 훈련자료로 사용하고 뽑히지 않은 개체들로 테스트
 - 2) 각 노드 분리에서 전체 p개의 변수 중에서 임의로(random) 선택된 m개의 변수를 비복원 임의 추출



Random Forest 모델의 오차 행렬과 정확도

- 30개의 의사결정 나무와 4개의 변수만 추출하여 붓스트래핑하여 예측을 하면 성능이 높아짐

오차 행렬(Confusion Matrix)



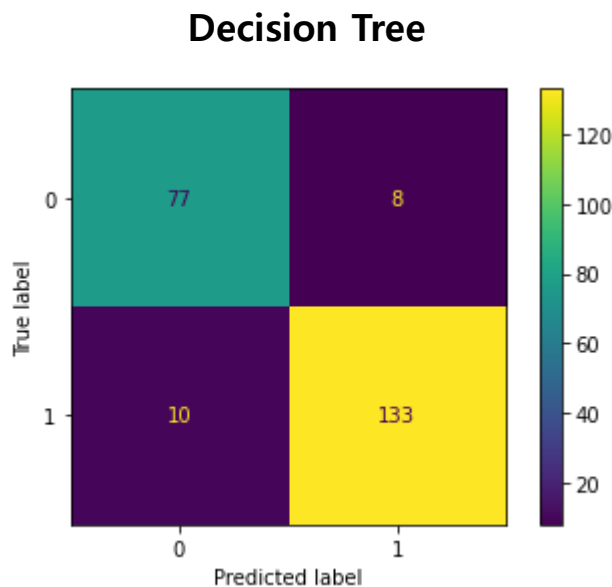
```
Rf = RandomForestClassifier(n_estimators=30,  
                             max_features=4,  
                             oob_score=True)
```

```
model.fit_predict(X, y)
```

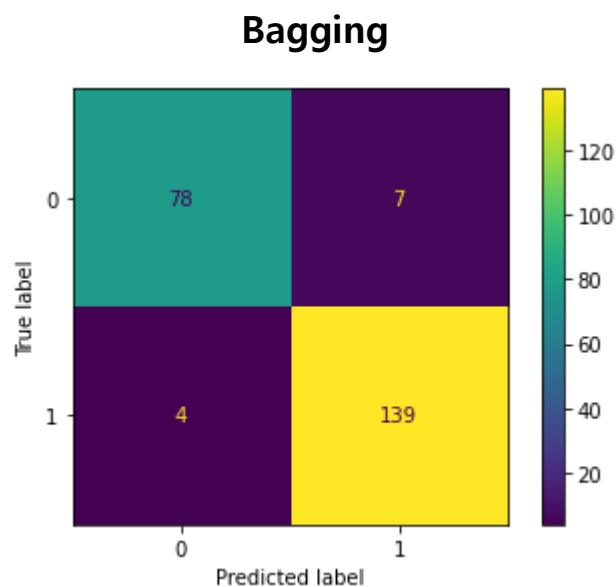
정확도(Accuracy) = $(78+141)/(85+143) = 96.05\%$

모델 비교

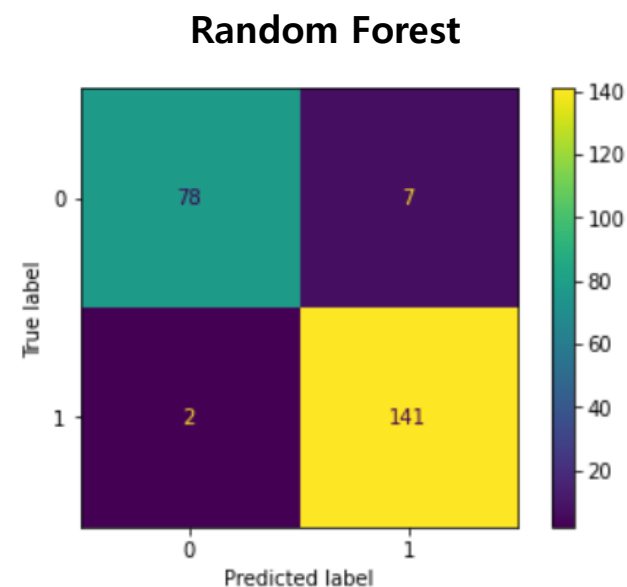
- 모형의 정확도(Accuracy)를 기준으로 평가하면 Random Forest가 가장 좋음
 - recall, precision, f1-score를 기준으로 하면?



$$(77+133)/(85+143) = 92.11\%$$



$$(78+139)/(85+143) = 95.18\%$$

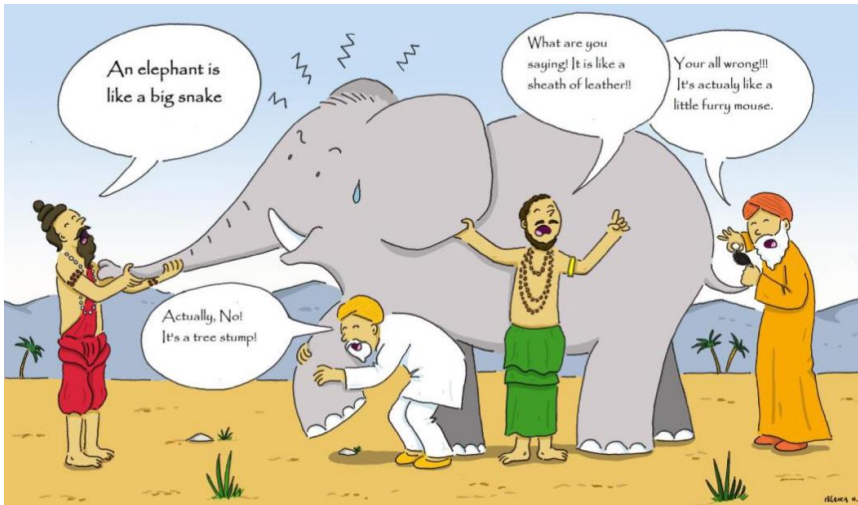


$$(78+141)/(85+143) = 96.05\%$$



Ensemble

Better predictions with bagging, boosting & stacking



<https://becominghuman.ai/ensemble-learning-bagging-and-boosting-d20f38be9b1e>



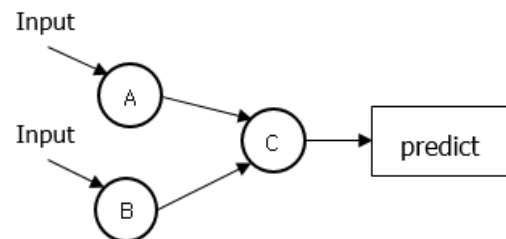
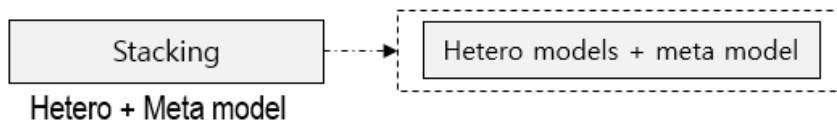
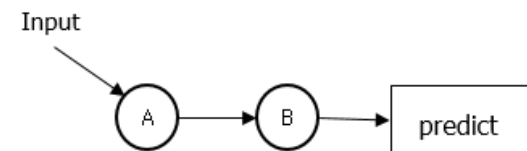
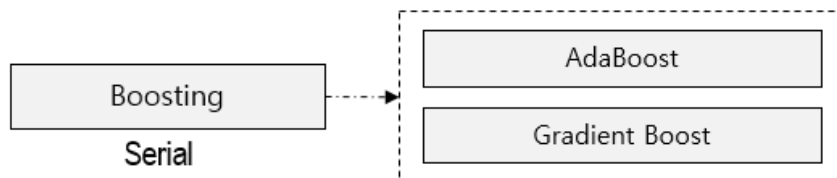
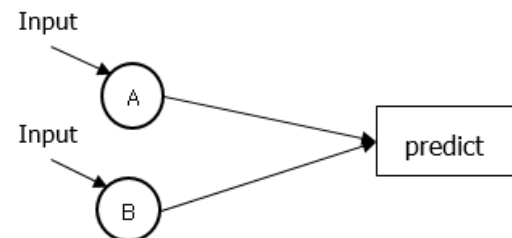
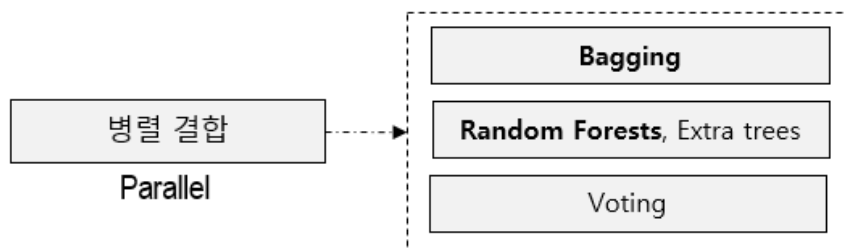
Bagging



Boosting

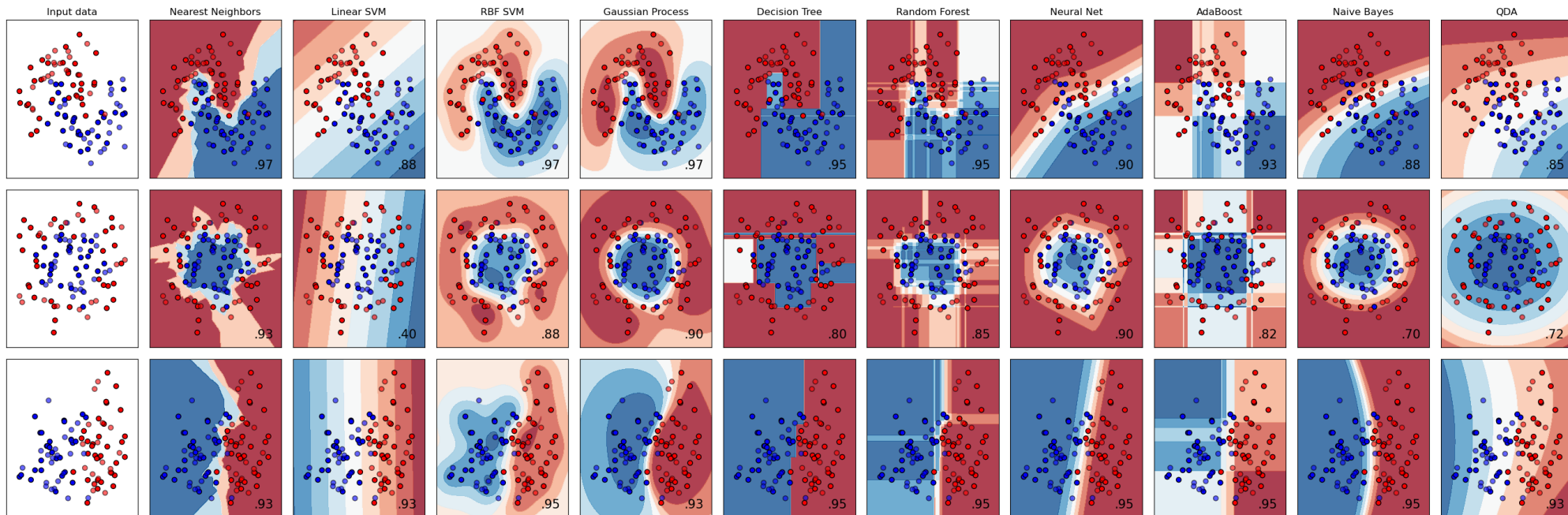
앙상블 모델

- 병렬 결합 방법은 병렬적(parallel) 형태의 여러 모델(A, B)의 다수결 예측
- 부스팅(Boosting) 방법은 선행 모델의 예측 결과를 후행 모델이 학습하는 순차적 직렬(Serial)형태 예측



Classifier comparison with the decision boundaries

- 알고리즘별로 예측성능(정확도)과 의사결정경계선을 나타내는 그림



https://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html