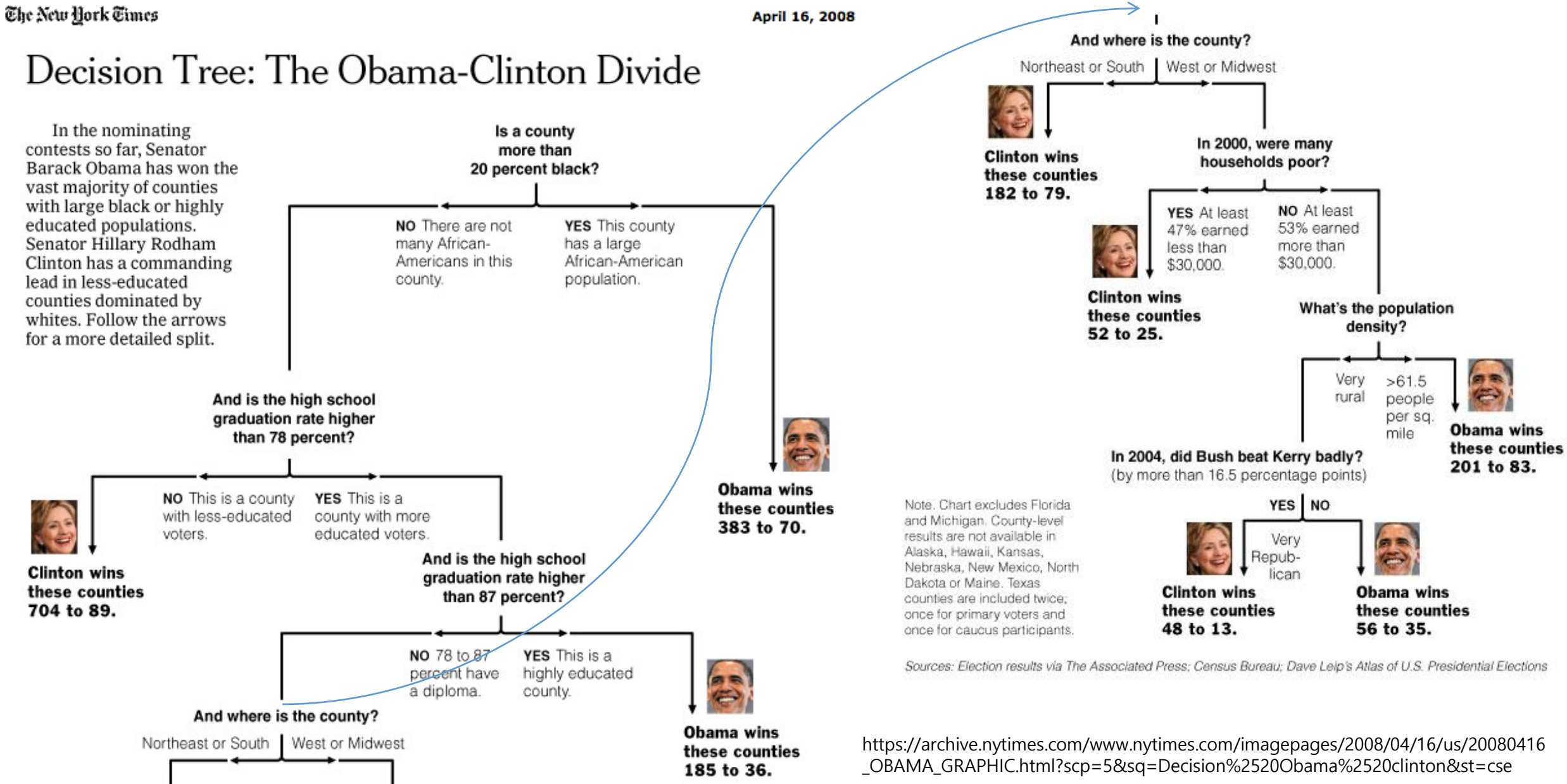
IT와 비즈니스혁신

W8-9. 마이닝 기법 I: 분류 (앙상블) & 평가

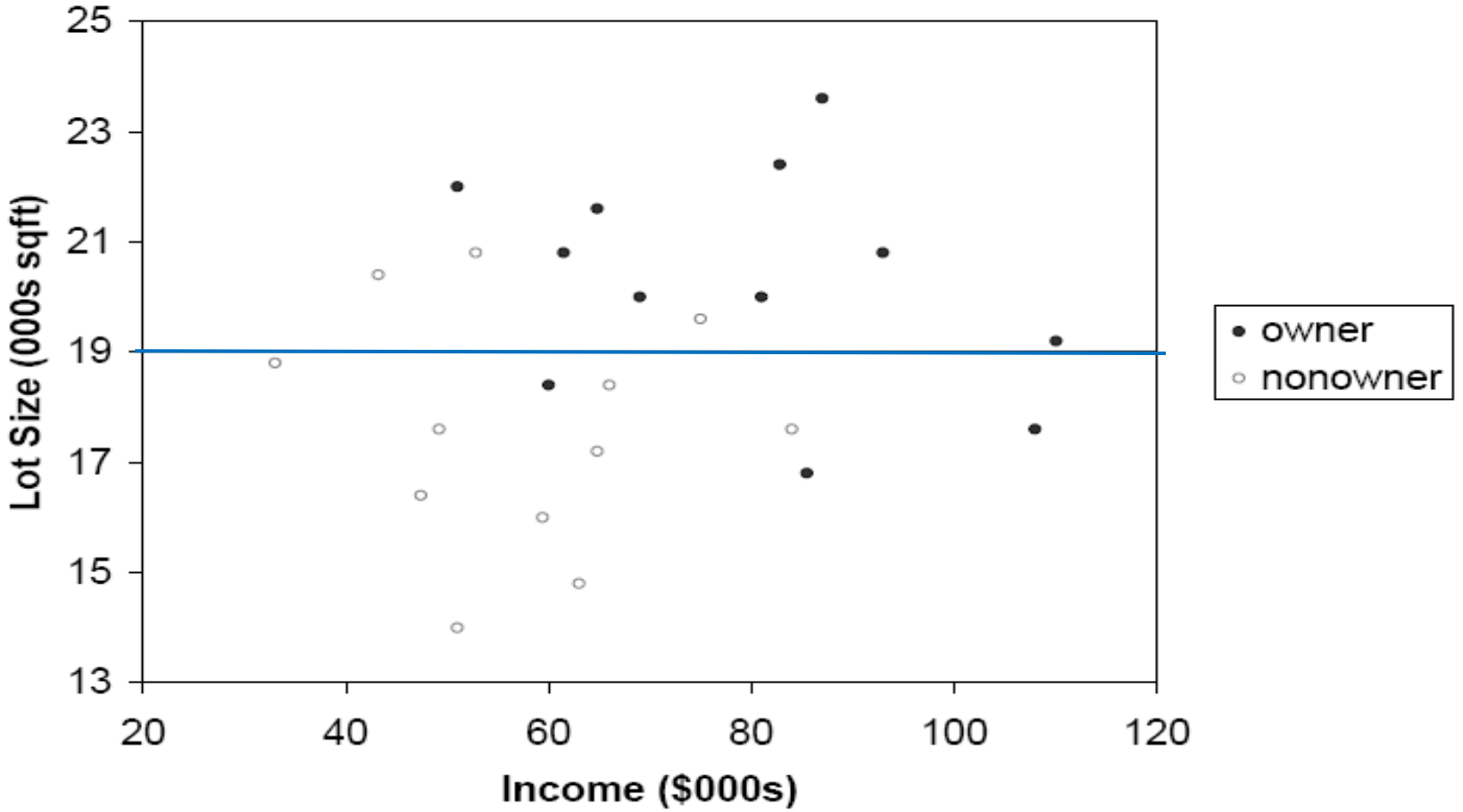
의사결정나무 예시: 선거 예측



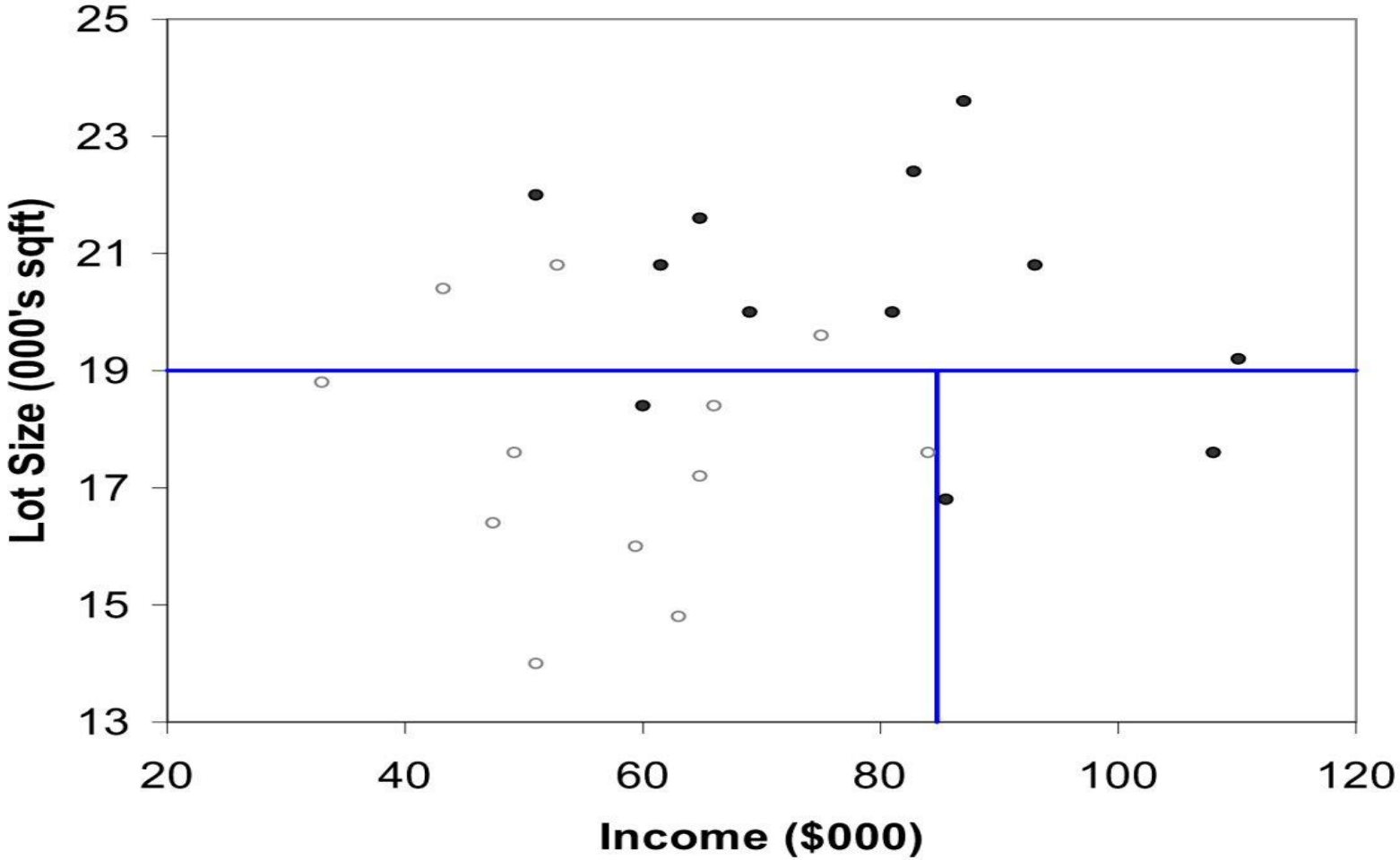
https://archive.nytimes.com/www.nytimes.com/imagepages/2008/04/16/us/20080416_OBAMA_GRAPHIC.html?scp=5&sq=Decision%2520Obama%2520clinton&st=cse

의사결정나무 예시: 누가 잔디깎기 기계를 살 것인가?

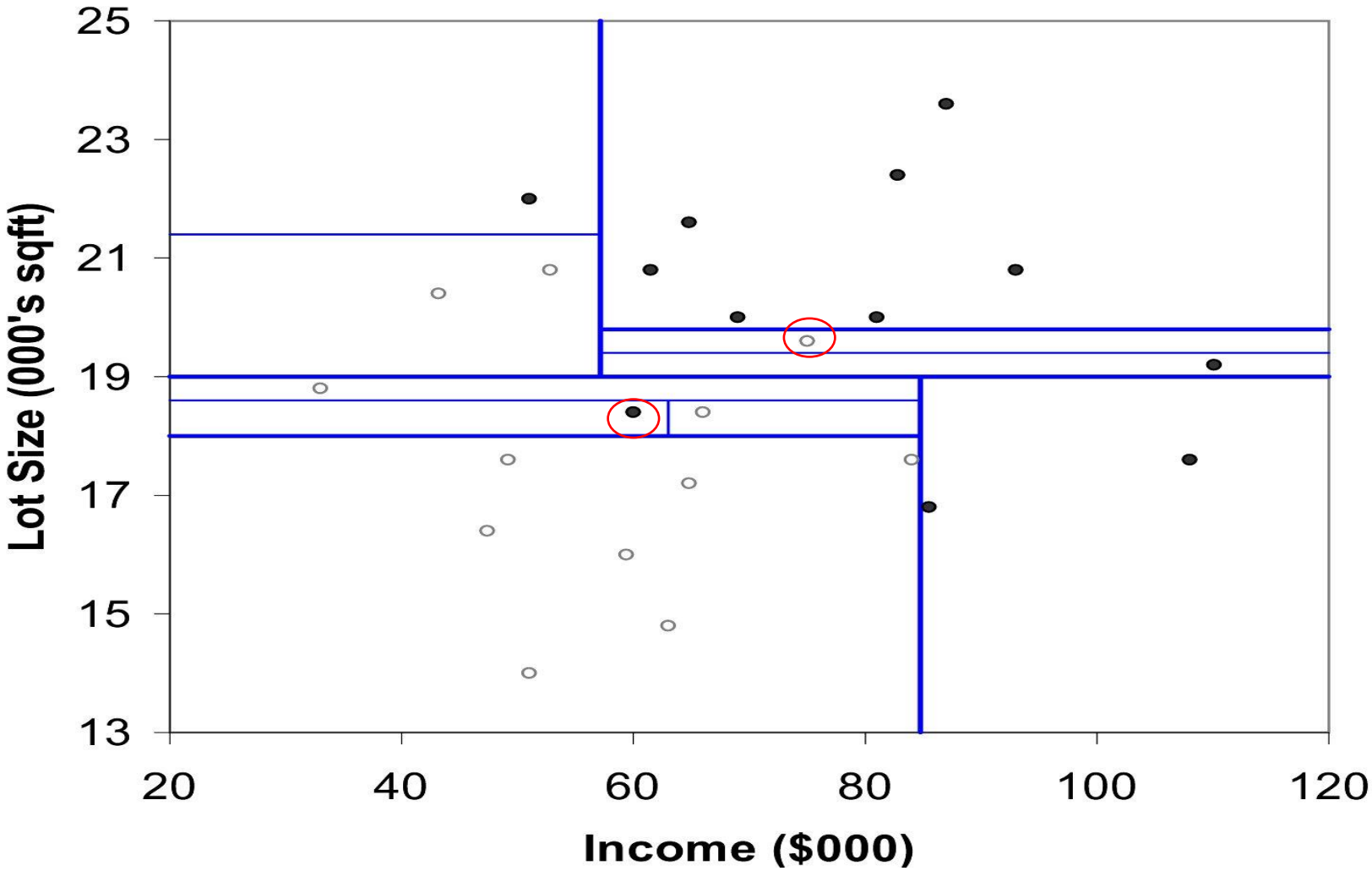
- 분할 기준 1: LotSize >= 19,000



- 분할 기준 2: Income = $84,000



- 완전 분할 후

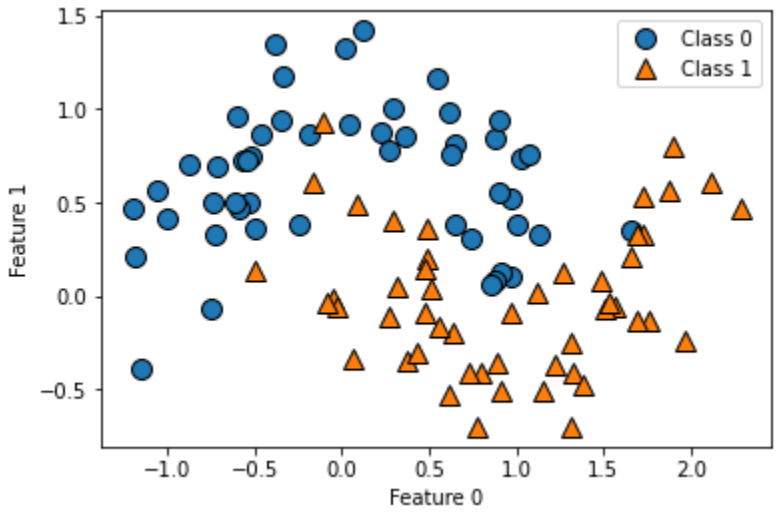

25
23
21
19
17
15
13
Lot Size (000's sqft)
20
40
60
80
100
120
Income ($000)

Contents

* 출처: A.Mueller & S.Guido, 박해선 역, 파이썬 라이브러리를 활용한 머신러닝, 한빛미디어, 2019
권철민.파이썬 머신러닝 완벽 가이드.위키북스.2020

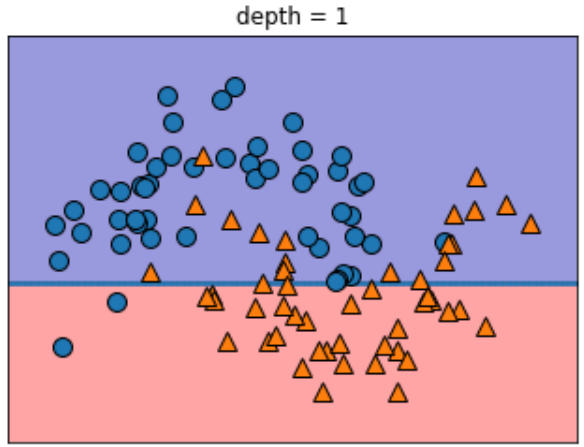
Ⅰ. 앙상블 (Ensemble) 모델

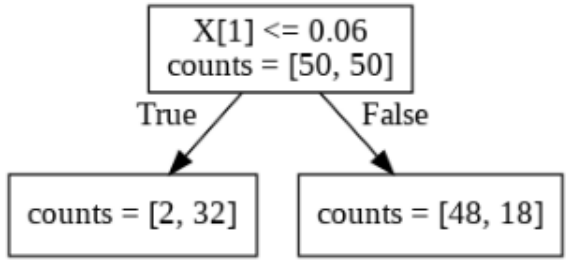
1 의사결정나무 생성 과정

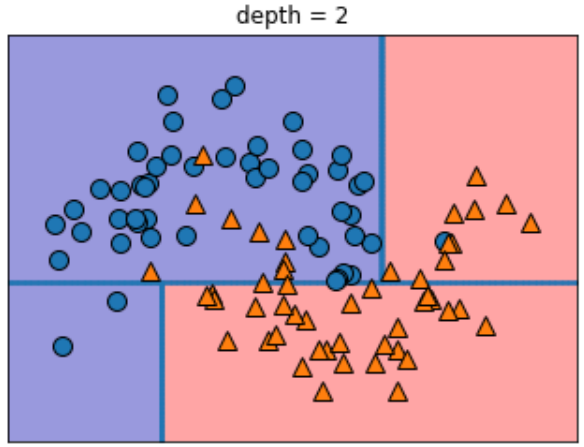


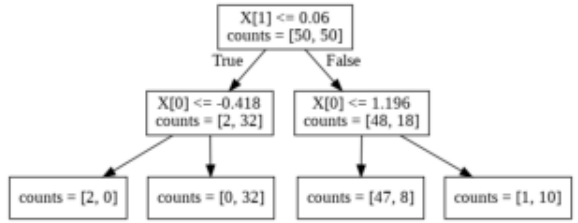
- 분류(Class 0 과 1을 구분)를 위한 의사결정나무 생성
- 나무의 깊이(Depth) 가 깊을수록 정확도가 향상되지만 과적합(overfitting)되는 경향이 있음

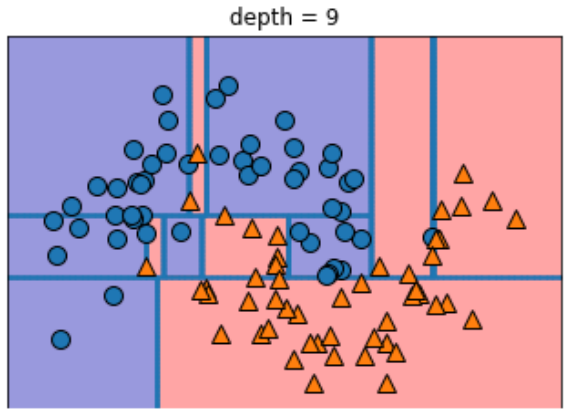
-> 이를 해결하기 위해서 단일 의사결정나무 대신 앙상블 방법 사용

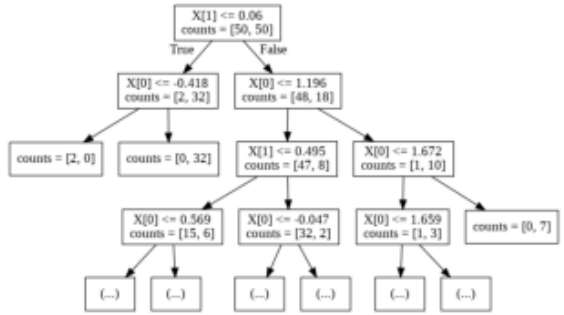










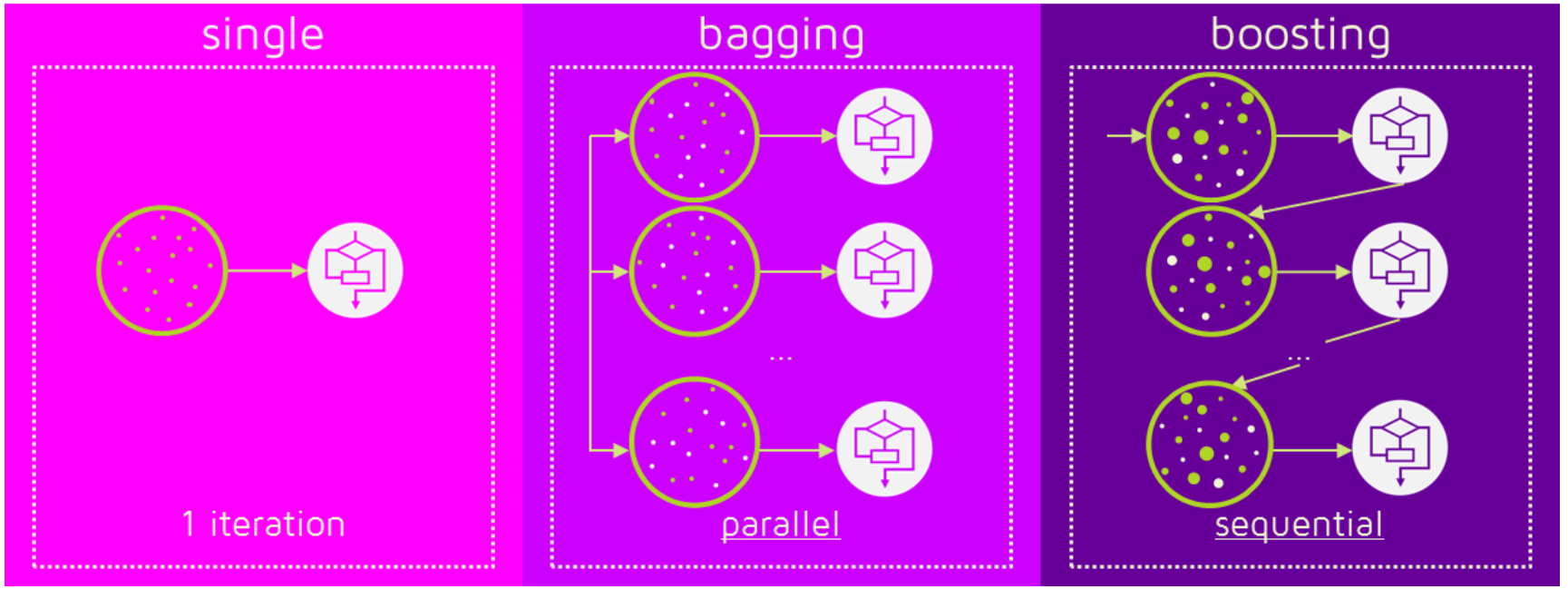


2 앙상블(Ensemble) 모델: 여러 모델을 연결하여 더 강력한 모델을 만드는 기법

여러 개의 모델을 조합하여 성능 향상

- 배깅 (Bagging)
  - 여러 개의 모델을 만들어 서로 다른 학습데이터로 학습 시킨 후(bootstrap), 동일한 테스트 데이터를 예측하고 집계(aggregating) –〉 특히 과적합 문제를 완화하고 성능 향상

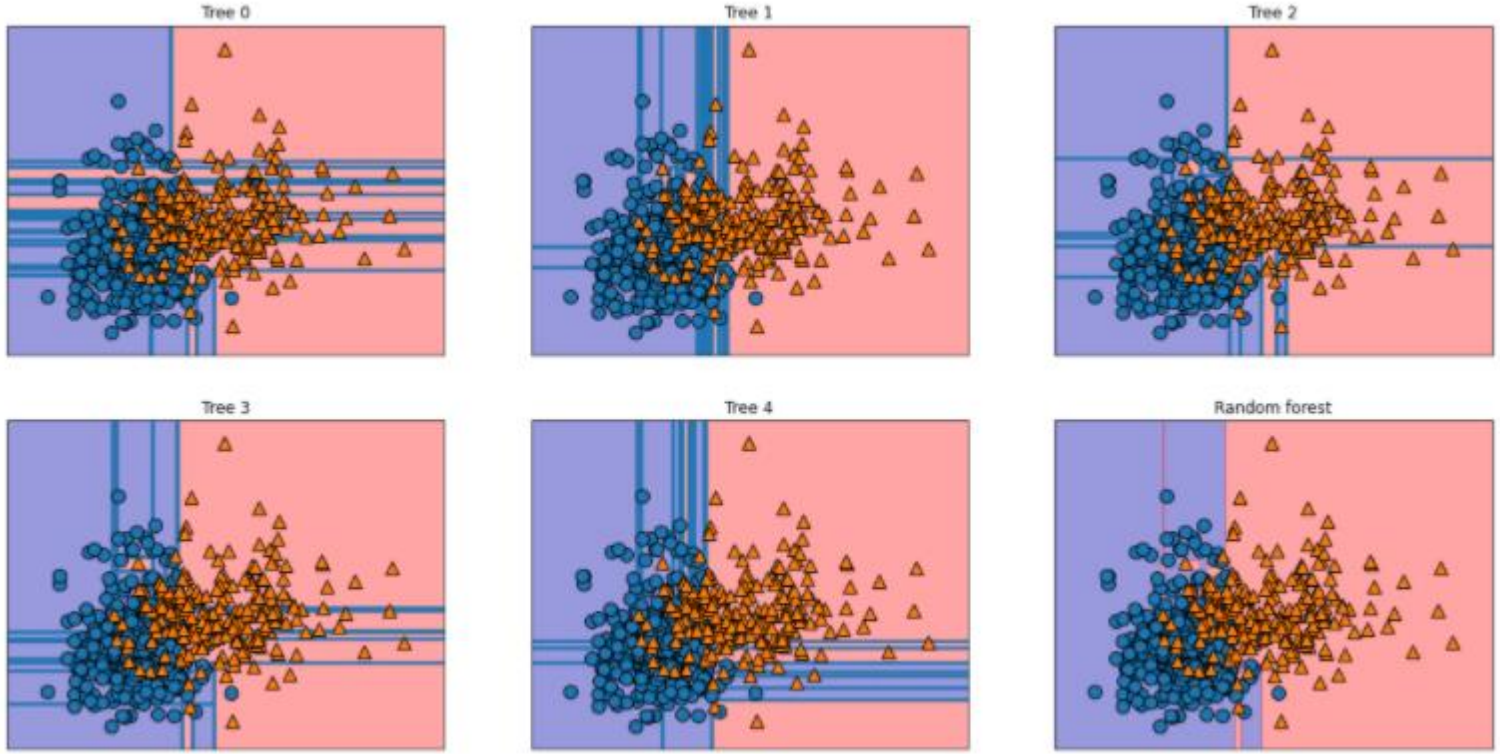
- 부스팅 (Boosting)
  - 병렬적으로 학습하는 배깅과 다르게 동일한 모델을 순차적으로 학습해서 (학습 데이터를 보강해 가면서) 여러 개의 모델을 만든 후 가중 투표를 통해 예측값 결정



랜덤 포레스트 (Random forest)

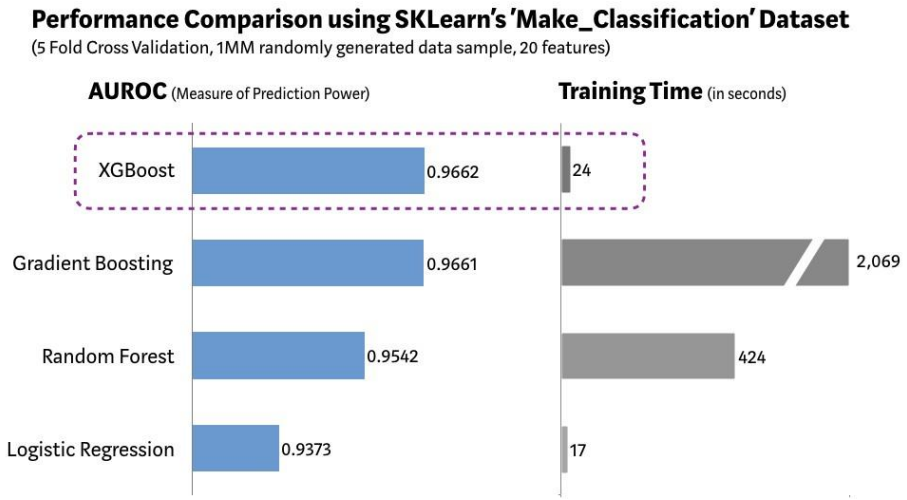
- 의사결정나무를 '랜덤하게' 여러가지로 만들고 이들의 **평균**을 이용하는 방법으로 과적합 문제 완화
  - 모델을 만들기 위해 사용하는 샘플 데이터를 “랜덤하게” 선택
  - 모델에 이용하는 변수(feature)를 “랜덤하게“ 선택

- 장단점
  - 성능이 매우 뛰어나고 매개변수 튜닝을 많이 하지 않아도 잘 작동함
  - 대량의 데이터는 여러 개의 CPU에 나누어 손쉽게 병렬 처리 가능
  - n_estimators(나무의 수)는 클수록 성능은 좋아지나 훈련 시간이 오래 걸림
  - 랜덤하므로 결과 모델이 매번 달라짐
  - 텍스트 데이터처럼 차원이 높고 희소한 데이터에는 잘 동작하지 않음 (선형모델이 더 적합)

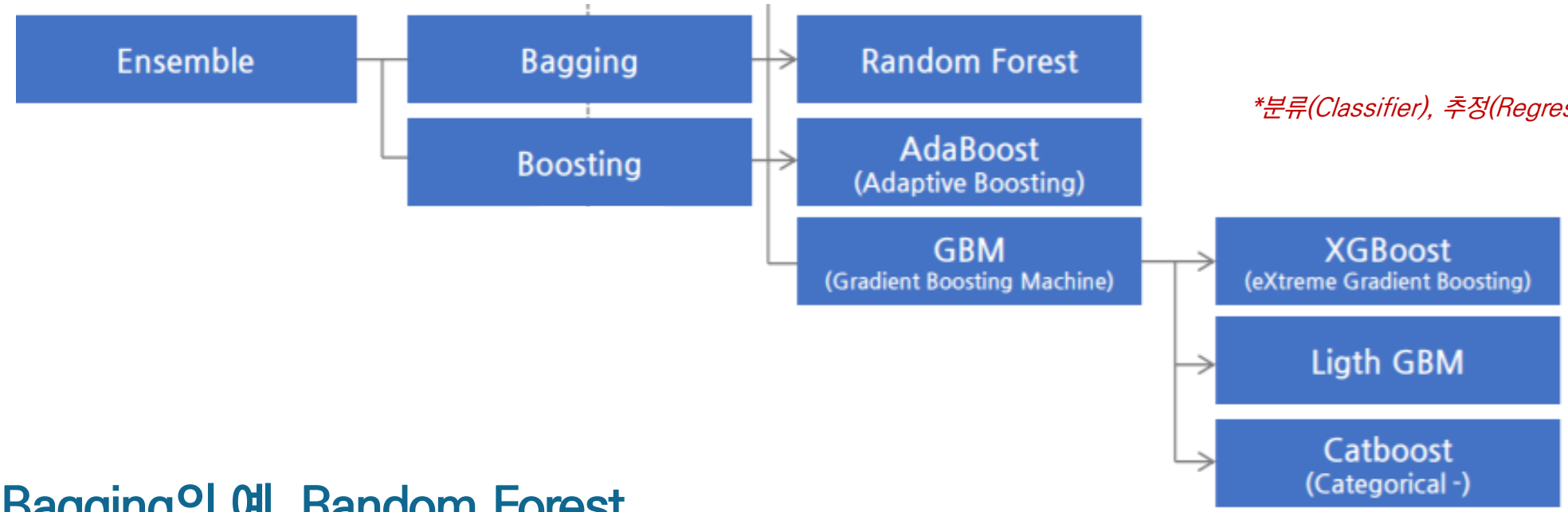


그래디언트 부스팅 (Gradient Boosting)

- 랜덤한 방식으로 앙상블을 구하는 것이 아니라 이전의 트리의 성능을 단계적으로 보완하여 오차를 줄이는 방식으로 나무 생성
- 가장 성능이 좋은 알고리즘으로 분류 및 예측에 모두 사용됨
- 장단점
  - 보통 5 이하의 깊지 않은 나무 (약한 학습기, weak learner)를 사용함으로써 학습 속도가 빠름
  - 이전 나무의 오차를 얼마나 강하게 보정할 것인지를 제어하는 학습 속도(learning rate)를 미세하게 조정해야 함
  - 희소한 고차원 데이터에 대해서는 잘 동작하지 않음
- XGBoost, LightGBM, CatBoost
  - XGBoost(eXtreme Gradient Boosting): 대용량 데이터를 다룰 수 있도록 분산 컴퓨팅 활용, 확장성이 뛰어남
  - LightGBM (Light Gradient Boosting Methods): XGBoost 보다 학습에 걸리는 시간과 메모리 사용량이 적음
  - CatBoost (Categorical Boosting): 범주형 데이터의 경우 예측 성능이 우수함



* 출처: https://www.kaggle.com/getting-started/190987



*분류(Classifier), 추정(Regressor) 모두 가능

Bagging의 예. Random Forest

```
# 모두 동일하며 tree_model2 = DecisionTreeClassifier(max_depth = 4, random_state =0)  대신
# tree_model3 = RandomForestClassifier(n_estimators = 100, random_state =0)  만 다름. 나무 100개를 만들어서
from sklearn.ensemble import RandomForestClassifier    #회귀문제는 ~Regressor

tree_model3 = RandomForestClassifier(n_estimators = 100, max_depth = 5, random_state =0, max_features=8)
tree_model3.fit(X_train, y_train)

print("랜덤포레스트 학습데이터 정확도: ", tree_model3.score(X_train, y_train))
print("랜덤포레스트 테스트데이터 정확도: ", tree_model3.score(X_test, y_test))
```

*DecisionTreeClassifer대신 RandomForestClassifier 사용

```
랜덤포레스트 학습데이터 정확도:  0.7702616464582004
랜덤포레스트 테스트데이터 정확도:  0.7705544933078394
```

* n_estimators: 생성할 나무의 수, 클수록 좋지만,메모리와 시간 이슈
* max_features:몇 개의 변수(feature)를 사용할지 지정할 수 있지만
  기본값도 좋음(분류: sqrt(n_features) ,회귀: 전체)
* 분류: 확률을 고려한 약한 투표 (여러 가능성의 가중평균)
  회귀: 앙상블의 평균치

II. 분류 실습

목적: 분류 (암진단: 양성, 악성)를 위한 의사결정나무 모델 생성

위스콘신대에서 제공하는 암진단 데이터

- 스노우보드에서 wisc_bc_data.csv 파일 다운로드 받기
- 캐글 사이트에서 다운로드: https://www.kaggle.com/uciml/breast-cancer-wisconsin-data

데이터 정보

- id : 환자 아이디
- diagnosis :진단 결과 (타겟 변수)

  M (malignant, 악성)

  B (benign, 양성)
- 세포에 관한 30개 변수 :

  radius, texture, perimeter, area, smoothness, compactness, concavity, points, symmetry, dimension 에 대한 _mean(평균), _se(표준편차), _worst(가장 큰 값 3개의 평균)

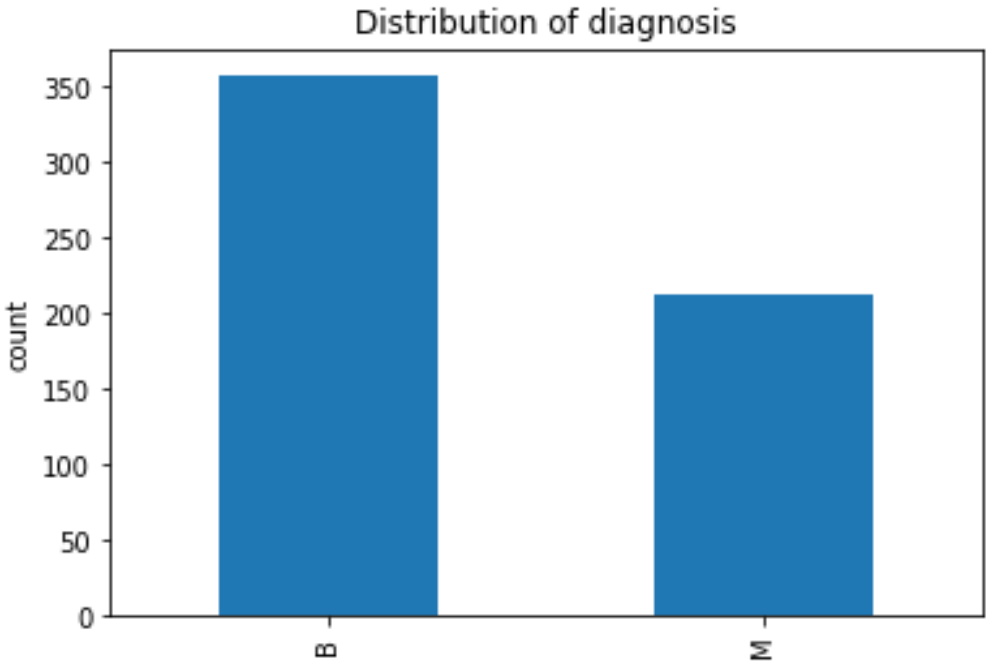
Id, 타겟 변수(diagnosis), 30개 변수(feature)

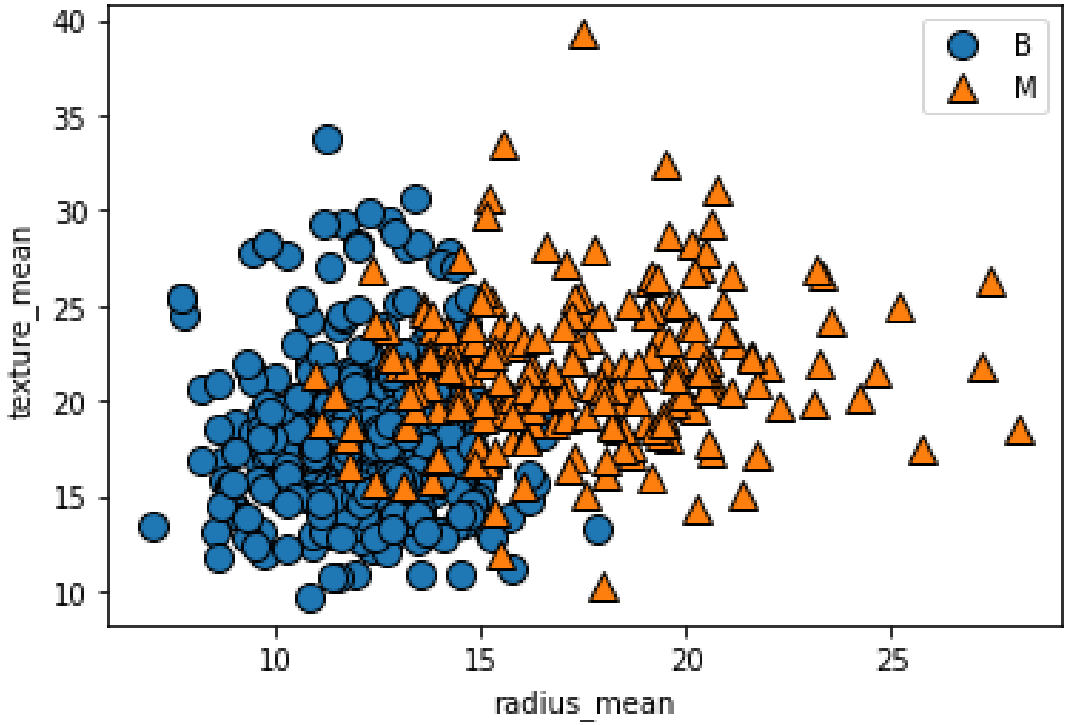
id	diagnosis	radius_mean	texture_mean	:
87139402	B	12.32	12.39	:
8910251	B	10.69	18.95	:
905520	B	11.04	16.83	:
868871	B	11.28	13.39	:
9012568	M	15.19	13.21	:
	:	:	:	:
	:	:	:	:
	:	:	:	:
	:	:	:	:
569행				

데이터 준비 (전처리)

- 데이터 인코딩(Ecoding): 범주형 데이터를 수치형으로 변환
  - 범주형 (A/B/C 등)을 수치형(1/2/3, ..)으로 변경 또는 더미 변수화
- 결측치 처리, 데이터 스케일 조정, 불균형 데이터 처리 등

시각화를 통한 데이터 탐색





사이킷 런 라이브러리의 함수 DecisionTreeClassifier 이용

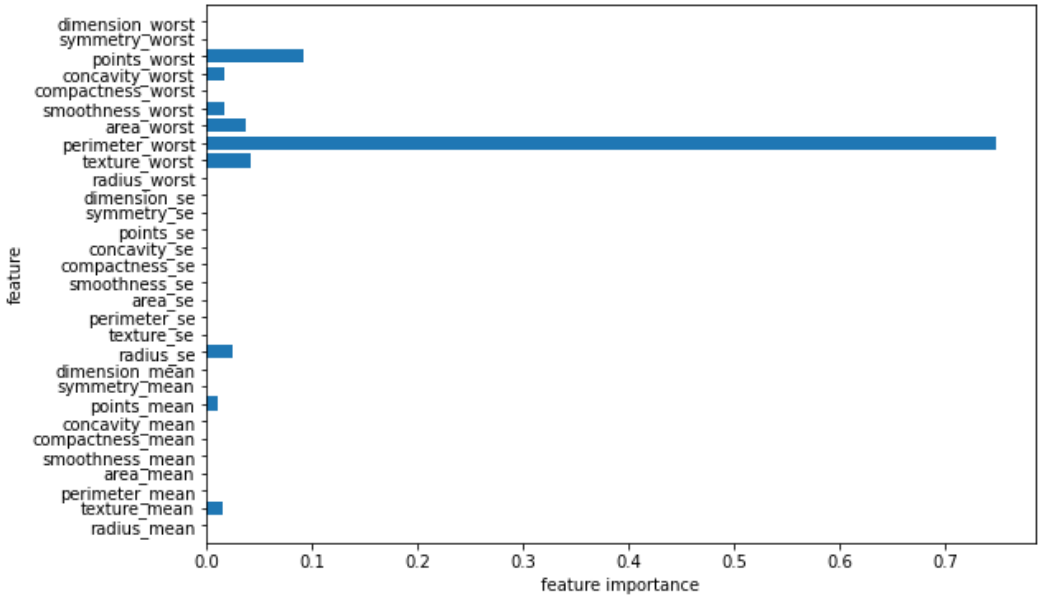
- 모델의 시각화가 직관적이어서 설명하기에 좋음
- 데이터의 스케일에 영향을 받지 않으므로 정규화나 전처리가 필요 없음
- 변수의 데이터 타입이 이진(binary)이나 연속값, 또는 이들이 혼합되어 있어도 잘 동작함
- 단점은 사전 가지치기 (예. 나무의 깊이 줄이기)를 사용해도 과적합(overfitting)되는 경향이 있음

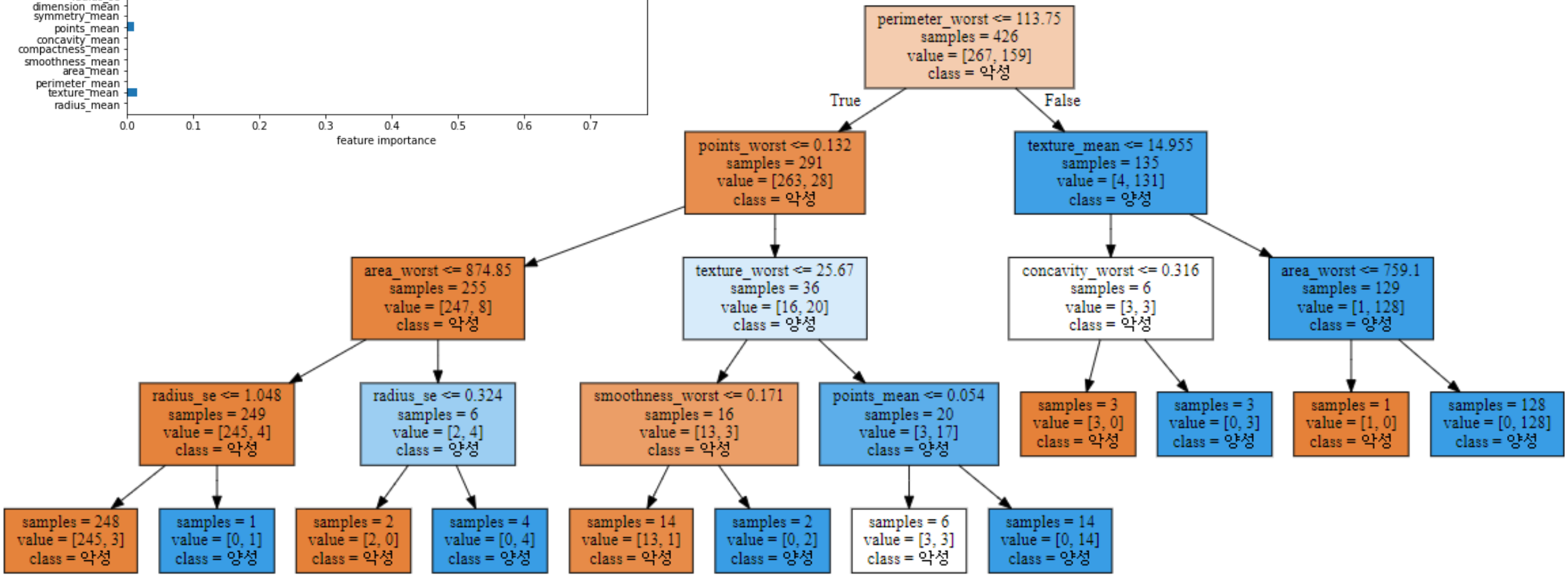
- 파라미터 설정
  - criterion: 분할시 순수도 계산 방법 (gini, entropy, default: gini)
  - splitter: 각 노드에서 분할을 선택하는 데 사용되는 전략 (best, random, default: best)
  - max_depth: 나무의 최대 깊이 (default: none)
  - min_samples_split: 자식 노드를 분할하는데 필요한 최소 샘플 수 (default: 2)
  - min_samples_leaf: 마지막 잎 노드에 있어야 할 최소 샘플 수 (default: 1)
  - max_features: 각 노드에서 분할에 사용할 변수의 최대 수 (auto, sqrt, log2, default: None)
  - random_state: 난수 seed 설정 (max_features 만큼의 변수 선택시)
  - max_leaf_nodes: 잎 노드의 최대수 (default: none)

핵심 문법

```
# 1. 학습/테스트 분할
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test=train_test_split(df[features], df[response_var], test_size=0.2)

#2.데이터 전처리

  :

#3. 의사결정나무 모델 구축 및 예측                    * 수치 추정을 위해서는
from sklearn.tree import DecisionTreeClassifier      DecisionTreeRegressor 이용

dt_model = DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=4,
        max_features=None, max_leaf_nodes=None, min_impurity_decrease=0.0,
        min_impurity_split=None, min_samples_leaf=1, min_samples_split=2,
        min_weight_fraction_leaf=0.0, presort=False, random_state=None, splitter='best')

dt_model.fit(x_train, y_train)
test_preds = dt_model.predict(x_test)

#4. 성능 평가
from sklearn.metrics import classification_repor, accuracy_score, precision_score, recall_score

print(classification_report(y_test, test_preds))
print('[Test] Accuracy: %0.4f' % accuracy_score(y_test, test_preds))
print('[Test] Precision: %0.4f' % precision_score(y_test, test_preds))
print('[Test] Recall: %0.4f' % recall_score(y_test, test_preds))
```

DecisionTreeClassifier(max_depth=4, random_state=10)



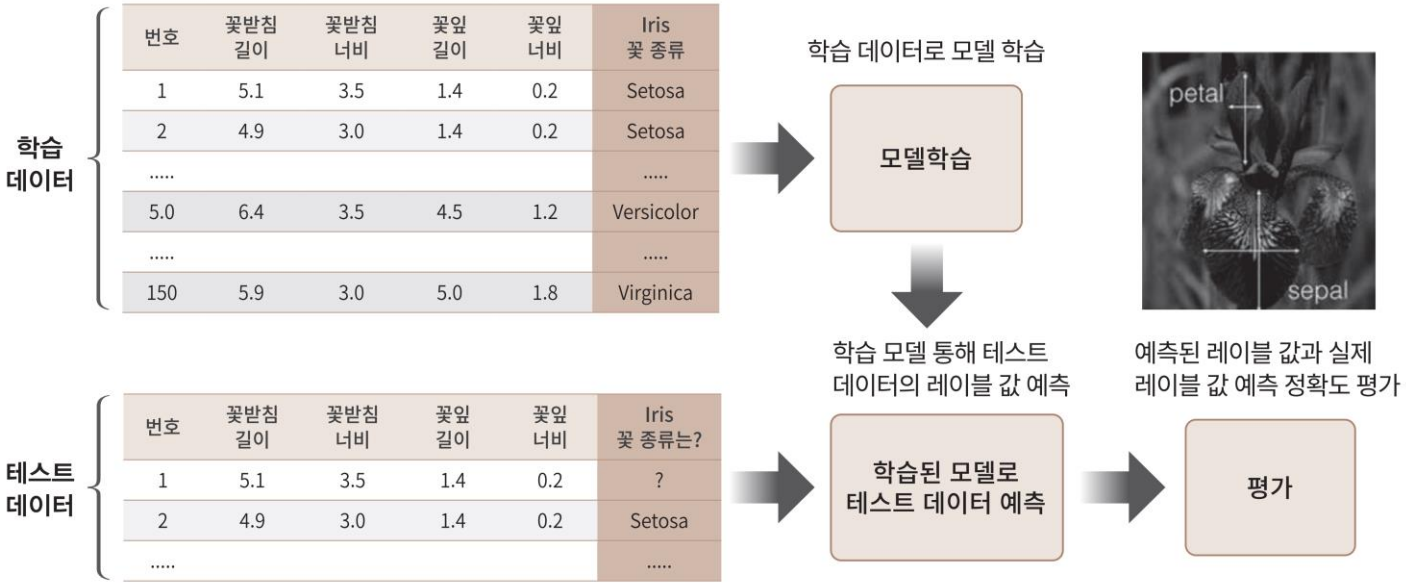


모델을 구축하기 위한 데이터(학습용) 와 모델 평가를 위한 데이터 (시험용) 로 분할

- X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, stratify=y, random_state=42)
  - random_state : random 으로 분할시 사용되는 난수 seed 숫자

시험용 데이터에 모델을 적용하여 정확도 계산

- tree.score(X_test, y_test) = 0.94 (시험용 데이터의 94%를 정확하게 분류함)

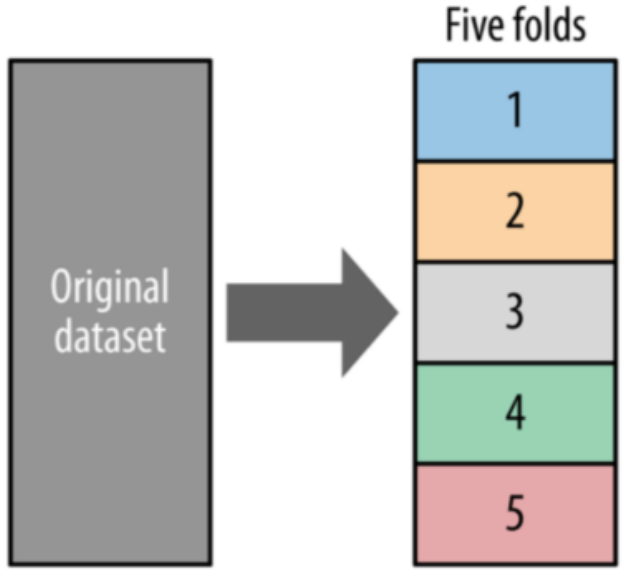


붓꽃 데이터를 이용한 분류 모델 구축과 평가 과정

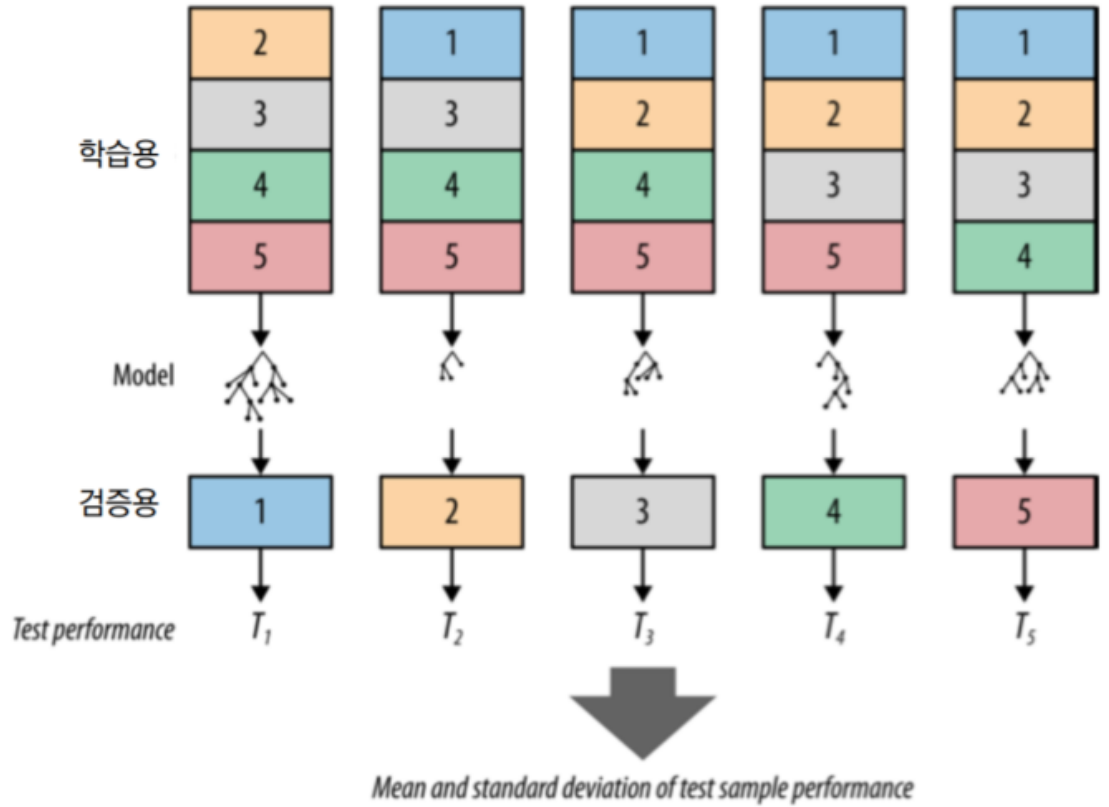
4 (Advanced) 교차 검증 (k-fold cross validation)

k번 교차 검증

- 최적의 모델 도출을 위해 사용 (특히, 데이터 수가 충분치 않을 때 중첩된(Nested) 방식으로 최적의 모델 찾기)
- 데이터를 무작위로 한 번만 나누게 되면 분류하기 쉬운 데이터만 학습에 이용될 수도 있고, 반대로 분류하기 어려운 데이터로만 이용할 수도 있음



*학습용 데이터를 k개 fold 로 나눈 후,
k-1 folds는 학습에 이용하고 1 fold는 모델 검증에
이용하는 방식을 k번 반복해서 이 k번의 성능 평균 파악



k번 수행한 교차 검증 결과의 평균값 이용

- dt_scores = cross_val_score(모델, X_train, y_train, cv=10, scoring='accuracy')

```
# 교차 검증을 10번 수행하여 10번의 교차 검증 평균 정확도를 비교 (10-fold cross validation)
from sklearn.model_selection  import cross_val_score

dt_scores = cross_val_score(tree_model2, X_train, y_train, cv=10, scoring='accuracy')
rf_scores = cross_val_score(tree_model3, X_train, y_train, cv=10, scoring='accuracy')

print("Accuracy")
print("Decision tree: ", dt_scores)
print("Random forest: ", rf_scores)

print("Accuracy mean")
print("Decision tree :{:.3f}".format(dt_scores.mean()))
print("Random forest :{:.3f}".format(rf_scores.mean()))
```

```
Accuracy
Decision tree:  [0.771 0.752 0.732 0.726 0.752 0.72  0.694 0.782 0.756 0.788]
Random forest:  [0.732 0.758 0.752 0.732 0.764 0.752 0.707 0.75  0.776 0.756]
Accuracy mean
Decision tree :0.747
Random forest :0.748
```

* dt_scores에 10개의 accuracy 성능이 들어감

5 (Advanced) GridSearch

그리드 서치를 이용하여 최적의 파라미터 조합 찾기

- 최고의 성능을 가지는 모델을 찾기 위해서 여러 파라미터들을 변경해 보면서 최적의 파라미터 조합을 찾는 함수

```
from sklearn.model_selection import GridSearchCV

parameters = {'max_depth':[2,3,4,5,6], 'max_features':[3,4,5,6,7,8,9]}

rf_tree = RandomForestClassifier(n_estimators = 100, random_state =0)
grid = GridSearchCV(rf_tree, param_grid = parameters, cv = 10, n_jobs = -1)
print(grid)

grid.fit(X_train, y_train)

print('GridSearchCV 최적 파라미터:', grid.best_params_)
print('GridSearchCV 최고 정확도: {0:.4f}'.format(grid.best_score_))
```

```
GridSearchCV 최적 파라미터: {'max_depth': 2, 'max_features': 3}
GridSearchCV 최고 정확도: 0.7511
```

* n_estimators = 나무의 개수
* n_jobs = −1 병렬처리

적합한 모델 선택 (학습용, 시험용 데이터에 적용시 정확도 측정하기)

	모델	파라미터 변경	정확도
Decision Tree	DecisionTreeClassifier(random_state=0)	max_depth= none	학습: 시험:
	DecisionTreeClassifier(max_depth=4, random_state=0)	max_depth= 4로 제한	학습: 시험:
Random Forest	RandomForestClassifier(n_estimators=100, random_state=0)		학습: 시험:
Gradient Boosting	GradientBoostingClassifier(random_state=0)	max_depth= 3 n_estimators=100 learning_rate=0.1	학습: 시험:
	GradientBoostingClassifier(random_state=0, max_depth=1)	max_depth= 1로 제한	학습: 시험:
	GradientBoostingClassifier(random_state=0, learning_rate=0.01)	learning_rate=0.01로 감소	학습: 시험:

Ⅲ. 모델 평가

- 정오분류표
- ROC 곡선과 AUC

1 정오분류표 (Confusion Matrix)

모델을 이용한 분류 결과의 요약

- 모델을 이용하여 구한 예측 분류와 실제 분류를 비교하여 발생 빈도수를 보여줌
- 이진 (binary) 분류인 경우 다음과 같이 4 가지 결과로 요약됨

		실제 분류	
		positive	negative
예측분류	Y	TP (True Positive, 참긍정)	FP (False Positive, 거짓긍정)
	N	FN (False Negative, 거짓부정)	TN (True Negative, 참부정)

선거결과 예시

Country별 선거결과/예측		실제 분류	
		오바마 승	오바마 패
예측 분류	오바마 승	TP (True Positive, 참긍정) 80	FP (False Positive, 거짓긍정) 40
	오바마 패	FN (False Negative, 거짓부정) 20	TN (True Negative, 참부정) 60

분류 결과의 정확도 평가 척도

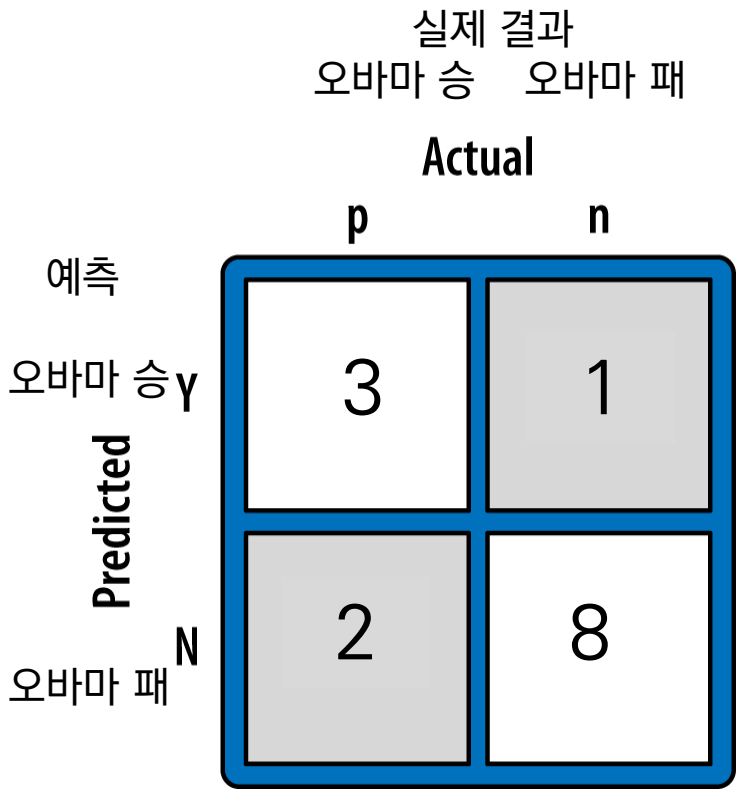
- 최적의 분류 모델을 선택하기 위한 평가 척도
- 민감도와 재현율은 같은 값을 가짐
- 실제 문제에서 오분류에 대한 위험성 또는 비용이 동일하지 않기 때문에 비용을 고려하여 적절한 평가 척도를 선택해야 함.
  - 예. 질병 예측의 경우, 실제 양성인 경우를 음성으로 분류하는 거짓부정이 실제 음성인 경우를 양성으로 분류하는 거짓긍정보다 오분류 비용이 큼

Country별 선거결과/예측		실제 분류	
		오바마 승	오바마 패
예측 분류	오바마 승	TP (True Positive, 참긍정) 80	FP (False Positive, 거짓긍정) 40
	오바마 패	FN (False Negative, 거짓부정) 20	TN (True Negative, 참부정) 60

평가 척도	정의	계산식
정확도 (accuracy)	전체 데이터 중 올바르게 분류한 경우의 비율	(TP+TN) / (TP+FP+FN+TN) = (오바마 승예측& 승결과+오바마 패예측&패결과)/전체
오분류율	오분류된 경우의 비율	(FP+FN) / (TP+FP+FN+TN) =1 – 정확도 = (오바마 승예측&패결과+오바마 패예측&승결과)/전체
민감도 (sensitivity)	실제로 긍정인 경우 중 긍정이라고 분류한 경우의 비율	TP / (TP+FN)= 오바마 승예측/(오바마 승예측&승결과+패예측&승결과)
특이도 (specificity)	실제로 부정인 경우 중 부정으로 분류한 경우의 비율	TN / (TN+FP) = 오바마 패예측/(오바마 승예측&패결과+패예측&패결과)
정밀도 (precision)	긍정이라고 분류한 경우 중 실제로 긍정인 경우의 비율	TP / (TP+FP) = 오바마 승결과/(오바마 승예측&패결과+승예측&승결과)
재현율 (recall)	실제로 긍정인 경우 중 긍정이라고 분류한 경우의 비율	TP / (TP+FN)
F1-measure	정밀도와 재현율의 조화평균	2 X (Precision X Recall) / (Precision + Recall)

연습문제 1. 분류 결과를 보고 각 평가척도의 값을 구하시오.

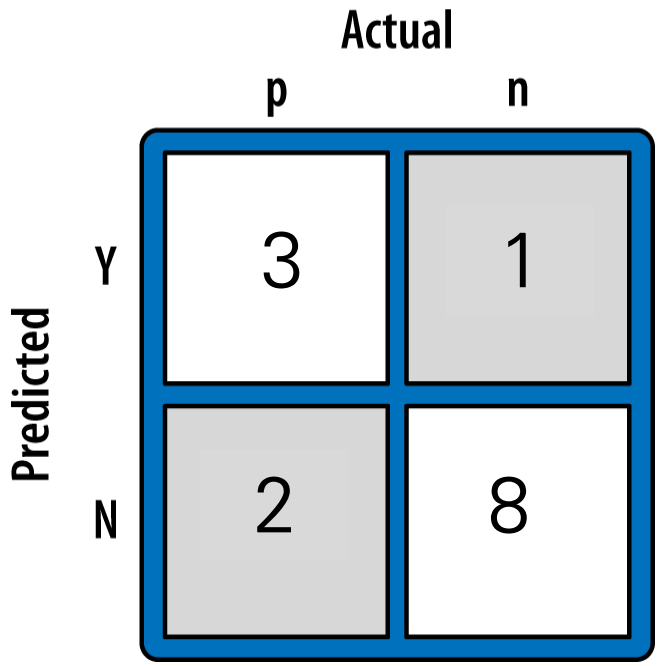
	실제분류	예측분류
1	n	N
2	p	Y
3	n	N
4	n	N
5	n	N
6	p	Y
7	n	N
8	n	N
9	n	N
10	p	N
11	n	N
12	n	Y
13	p	N
14	p	Y



평가 척도	계산값
정확도 (accuracy)	
오분류율	
민감도 (sensitivity)	
특이도 (specificity)	
정밀도 (precision)	

연습문제 1 답안

	실제분류	예측분류
1	n	N
2	p	Y
3	n	N
4	n	N
5	n	N
6	p	Y
7	n	N
8	n	N
9	n	N
10	p	N
11	n	N
12	n	Y
13	p	N
14	p	Y



평가 척도	계산값
정확도 (accuracy)	11/14
오분류율	3/14
민감도 (sensitivity)	3/5
특이도 (specificity)	8/9
정밀도 (precision)	3/4

2 ROC (Receiver Operator Characteristic, 수신자 동작 특성 곡선)

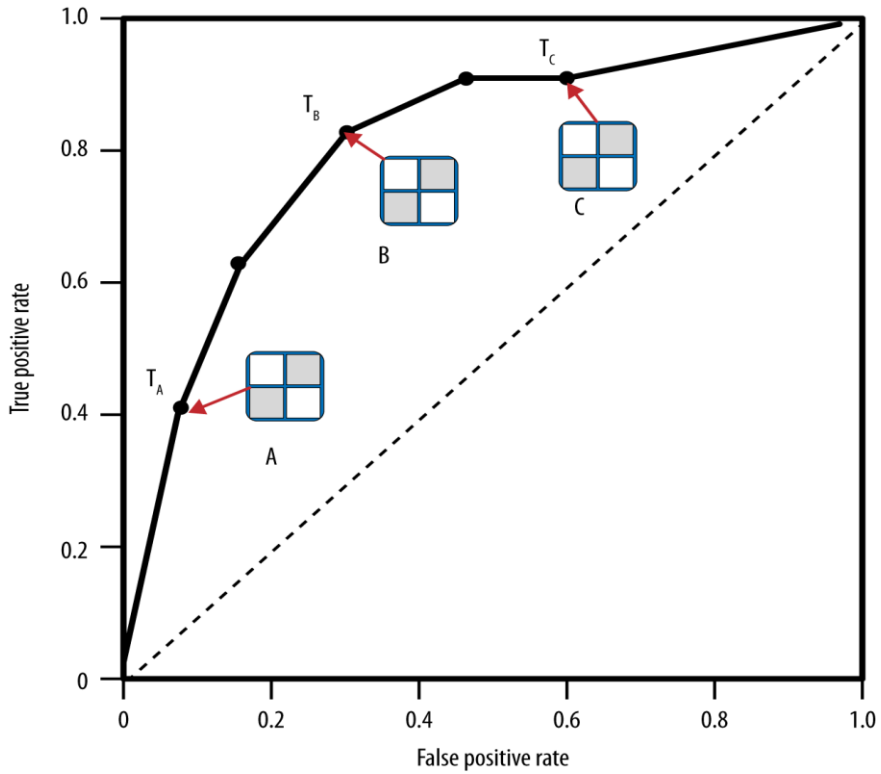
민감도와 특이도의 관계를 표현한 그래프

- 일반적으로 민감도와 특이도를 동시에 증가시키는 것은 불가능
- 모델의 분류기준값(cutoff)을 조정해 가면서 1-특이도 (x축)와 민감도(y축)를 도식화 하여 보여줌
- 하나의 분류 문제에 대해 여러 분류 모델을 비교하여 가장 적합한 모델 선택 가능
  - 예. 질병 예측의 경우, 여러 평가 척도 중 민감도 (실제로 양성인 경우 양성으로 분류하는 비율) 가 중요하므로 이를 고려하여 모델 선택

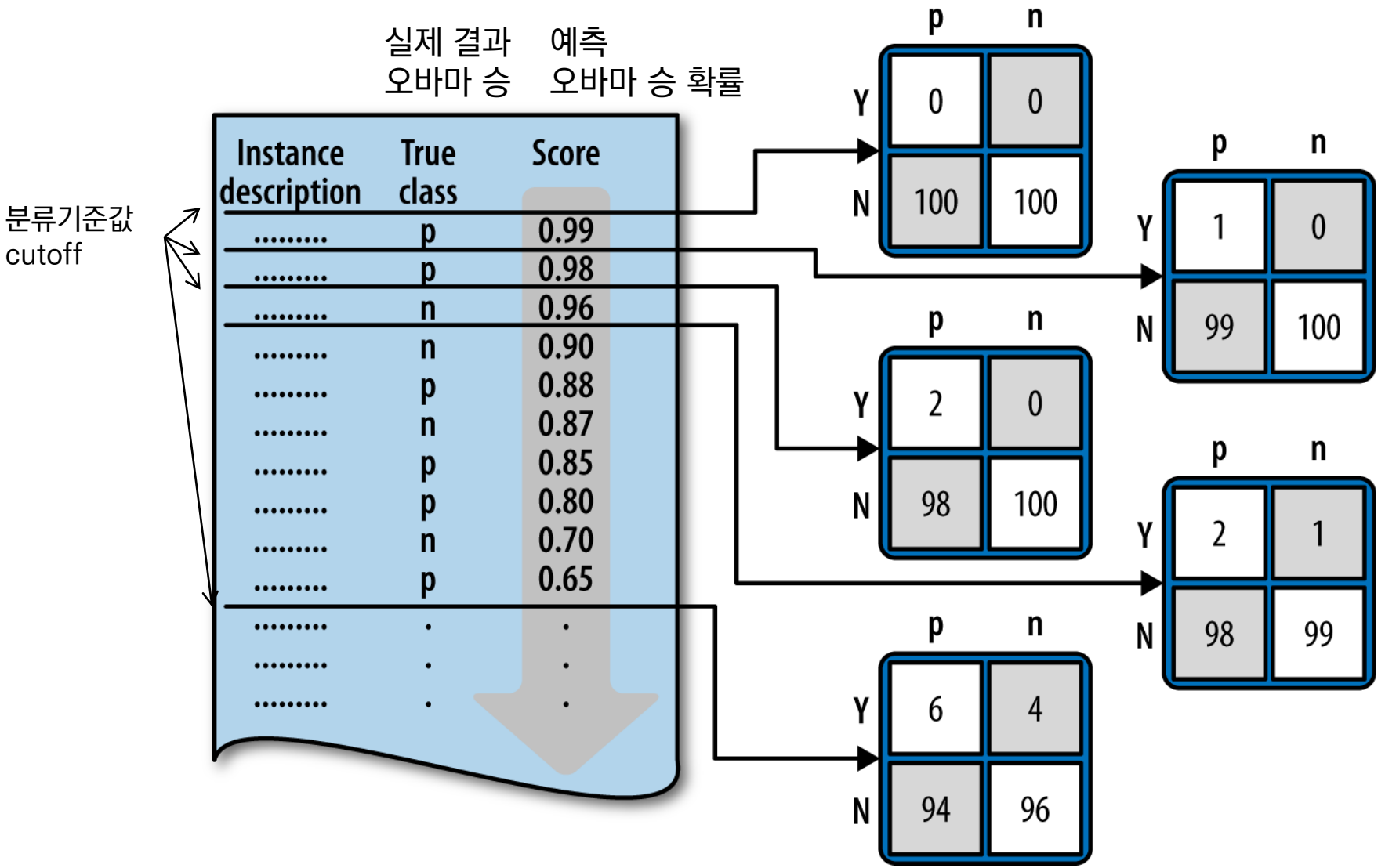
* ROC 곡선은 2차 세계 대전 당시 레이더 화면에 뜬 광점이 적선인지 무해한지를 판별하는 레이더 운영원들의 능력을 측정하기 위해 개발됨

		실제 분류	
		positive	negative
예측 분류	Y	TP (True Positive, 참긍정)	FP (False Positive, 거짓긍정)
	N	FN (False Negative, 거짓부정)	TN (True Negative, 참부정)

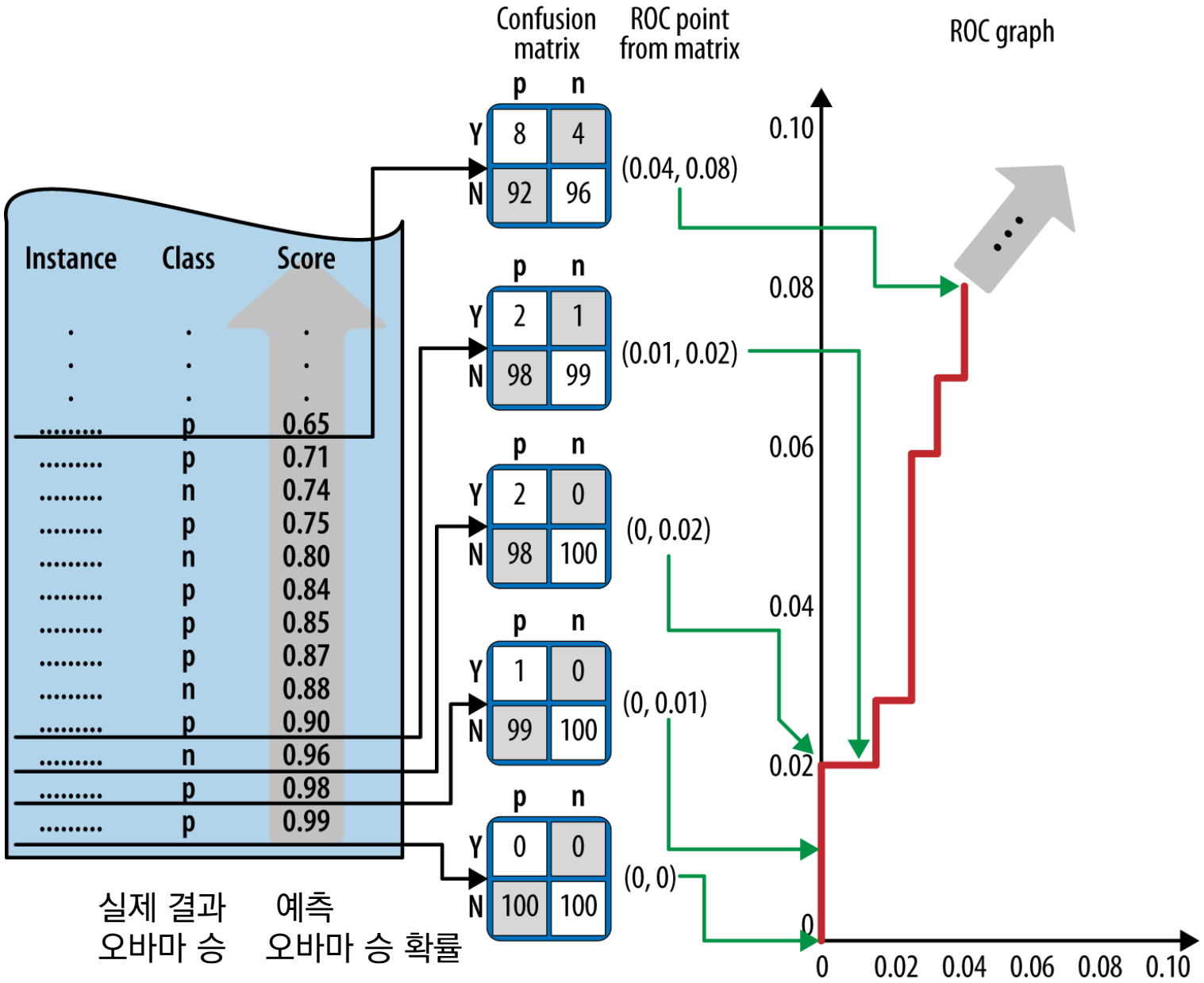
True Positive Rate(참긍정 비율)
= 민감도(sensitivity)
= TP/(TP+FN)



False Positive Rate(거짓긍정 비율, 긍정오류)
=1- 특이도(specificity)
= FP / (TN+FP)



* 출처: F.Provost & T.Fawcett, 강권학 역, 비즈니스를 위한 데이터 과학 : 빅데이터를 바라보는 데이터 마이닝과 분석적 사고, 한빛미디어, 2014

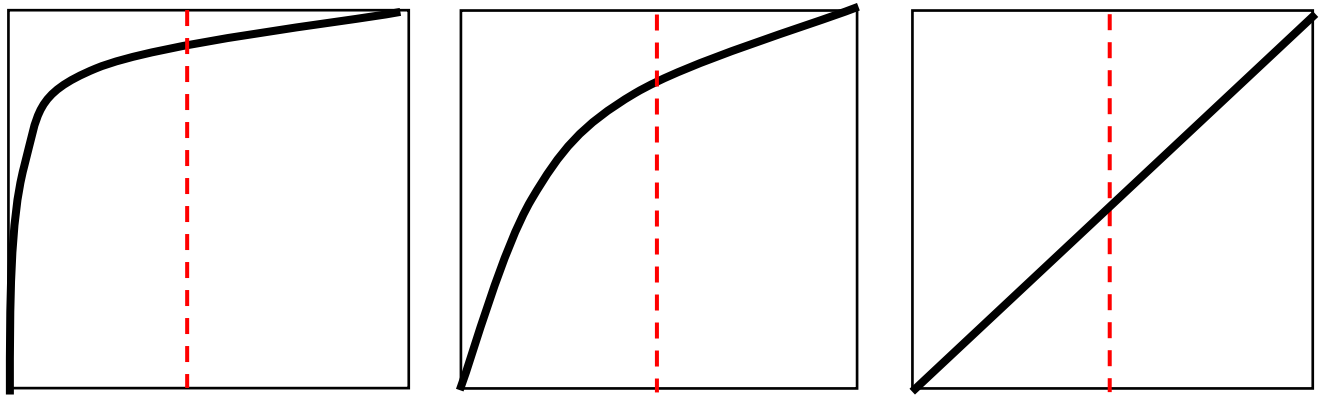


ROC 곡선 그리는 방법

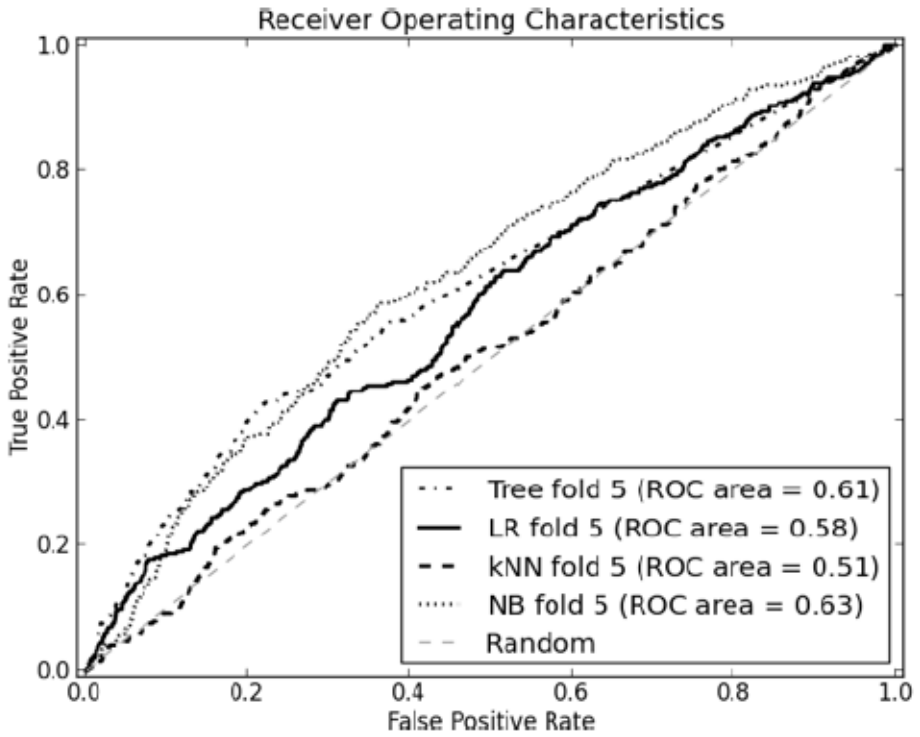
- 분류 결과를 분류의 신뢰도 또는 점수 에 따라 내림차순으로 정렬
- 첫 행부터 차례대로 긍정으로 분류될 경우의 결과에 대한 TP(p 경우의 수), FP(n 경우의 수) 누적수를 계산
- 각 행의 누적 TP, FP 수를 이용하여 민감도, 1-특이도를 계산하고 x, y 값으로 하여 산점도 그리기
- 각 점들을 연결하여 곡선 만들기

ROC 곡선의 비교

- ROC 곡선의 아래 면적이 클수록 민감도와 특이도의 정확도가 높으므로 더 우수한 모델이라 할 수 있음
- 그림 (a): 같은 (1 – 특이도) 값에 대해 민감도 값이 가장 높음
- 그림 (c): 모델 사용의 효과가 전혀 없음 (대각선의 경우 50%)
- 여러 분류 모델을 동시에 비교 가능

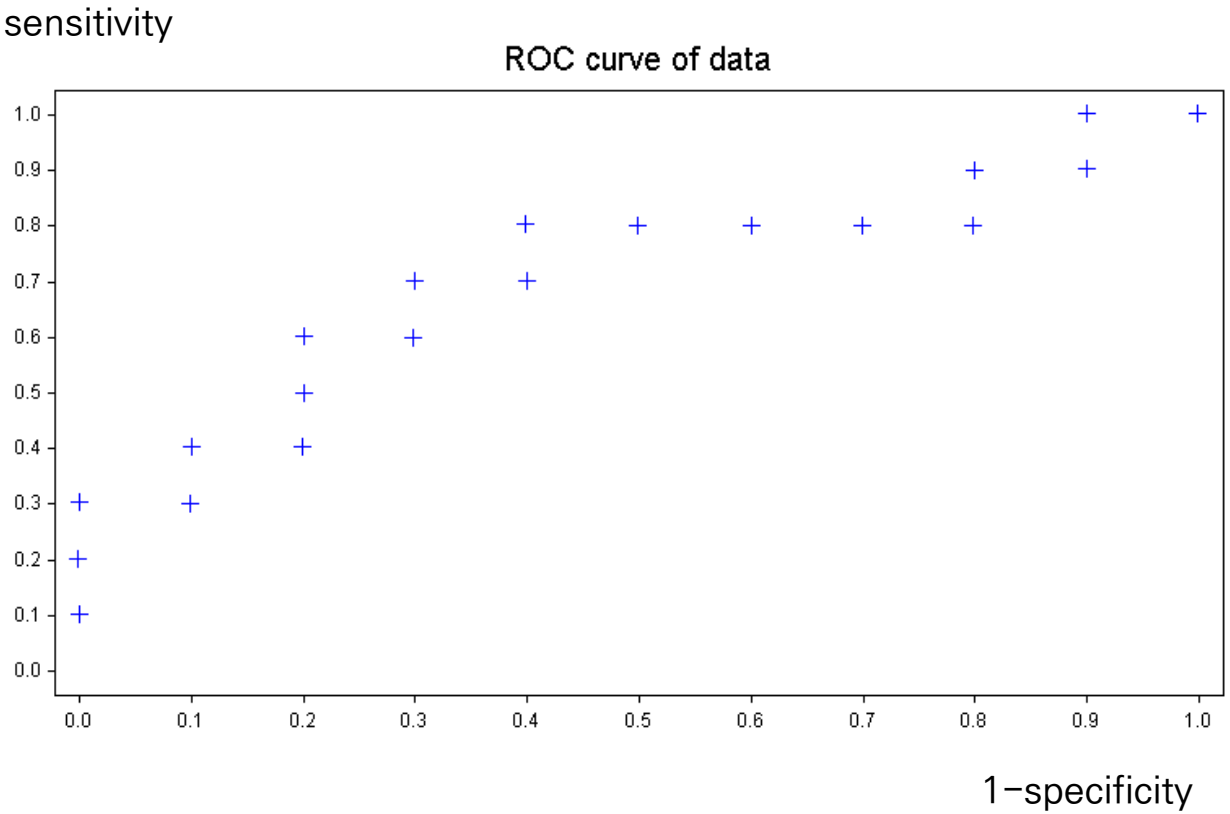


(a) (b) (c)



연습문제 2. 분류기준값(cutoff)를 변경하면서 ROC 곡선을 그리고, 다음 빈 칸을 채우시오.

case	질병판정 실제결과 p:yes n:no	판정 예측 확신도 confidence (yes)	민감도	1-특이도
1	p	0.65	0.1	0.0
2	p	0.56	0.2	0.0
3	p	0.55	0.3	0.0
4	n	0.51	0.3	0.1
5	p	0.50	0.4	0.1
6	n	0.47	0.4	0.2
7	p	0.47	0.5	0.2
8	p	0.45	0.6	0.2
9	n	0.44	0.6	0.3
10	p	0.44	0.7	0.3
11	n	0.42	0.7	0.4
12	p	0.42		
13	n	0.38	0.8	0.5
14	n	0.38	0.8	0.6
15	n	0.38	0.8	0.7
16	n	0.37	0.8	0.8
17	p	0.36	0.9	0.8
18	n	0.33	0.9	0.9
19	p	0.31	1.0	0.9
20	n	0.29	1.0	1.0

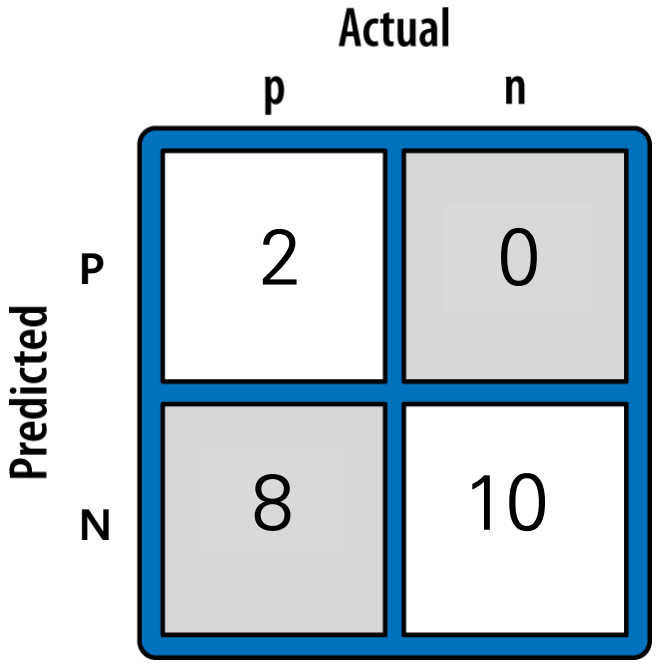


양성 판정의 분류기준값 (cuoff)이 0. 42 인 경우,
민감도(sensitivity)= 80 % 특이도(specificity) = 60 %.

연습문제 2. 분류기준값(cutoff)를 변경하면서 ROC 곡선을 그리고, 다음 빈 칸을 채우시오.

case	질병판정 실제결과 p:yes n:no	판정 예측 확신도 confidence (yes)	예측 결과	민감도	1-특이도
1	p	0.65	P	0.1	0.0
2	p	0.56	P	0.2	0.0
3	p	0.55	N	0.3	0.0
4	n	0.51	N	0.3	0.1
5	p	0.50	N	0.4	0.1
6	n	0.47	N	0.4	0.2
7	p	0.47	N	0.5	0.2
8	p	0.45	N	0.6	0.2
9	n	0.44	N	0.6	0.3
10	p	0.44	N	0.7	0.3
11	n	0.42	N	0.7	0.4
12	p	0.42	N		
13	n	0.38	N	0.8	0.5
14	n	0.38	N	0.8	0.6
15	n	0.38	N	0.8	0.7
16	n	0.37	N	0.8	0.8
17	p	0.36	N	0.9	0.8
18	n	0.33	N	0.9	0.9
19	p	0.31	N	1.0	0.9
20	n	0.29	N	1.0	1.0

Cut off =0.56

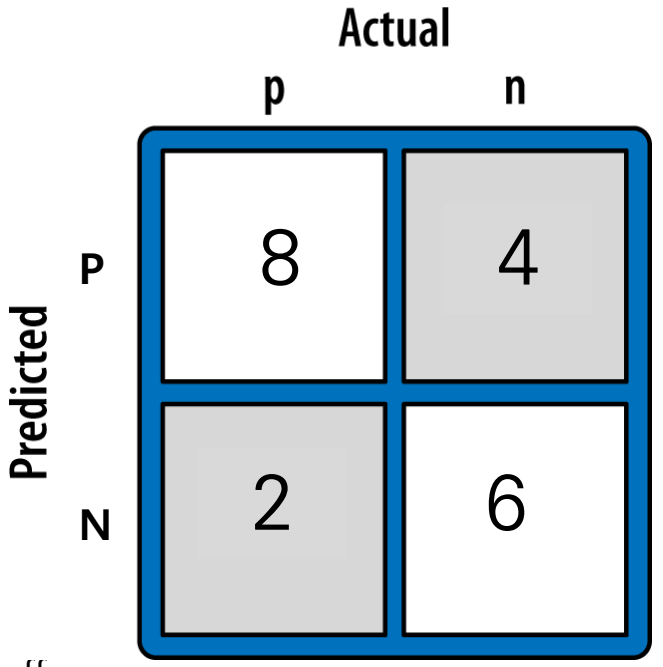


양성 판정의 분류기준값 (cuoff)이 0. 56 인 경우,
민감도(sensitivity)= 2(실제 p& 예측P)/10 (실제 p)=0.2
특이도(specificity) = 10(실제 n & 예측 N)/10 (실제 n)= 1
1-특이도 = 0

연습문제 2. 분류기준값(cutoff)를 변경하면서 ROC 곡선을 그리고, 다음 빈 칸을 채우시오.

case	질병판정 실제결과 p:yes n:no	판정 예측 확신도 confidence (yes)	예측 결과	민감도	1-특이도
1	p	0.65	P	0.1	0.0
2	p	0.56	P	0.2	0.0
3	p	0.55	P	0.3	0.0
4	n	0.51	P	0.3	0.1
5	p	0.50	P	0.4	0.1
6	n	0.47	P	0.4	0.2
7	p	0.47	P	0.5	0.2
8	p	0.45	P	0.6	0.2
9	n	0.44	P	0.6	0.3
10	p	0.44	P	0.7	0.3
11	n	0.42	P	0.7	0.4
12	p	0.42	P		
13	n	0.38	N	0.8	0.5
14	n	0.38	N	0.8	0.6
15	n	0.38	N	0.8	0.7
16	n	0.37	N	0.8	0.8
17	p	0.36	N	0.9	0.8
18	n	0.33	N	0.9	0.9
19	p	0.31	N	1.0	0.9
20	n	0.29	N	1.0	1.0

Cut off =0.42



양성 판정의 분류기준값 (cuoff)이 0. 42 인 경우,
민감도(sensitivity)= 8 (실제 p& 예측P) /10 (실제 p) =0.8
특이도(specificity) = 6 (실제 n& 예측N)/10 (실제 n)= 0.6
1- 특이도 = 0.4

"코로나19 환자 증상 발현 전 전파율 40%"

2020.04.28 15:41

【 임시선별검사소 코로나19 검사법 3종 안내 】

구분	비인두도말 PCR법	신속항원검사법	타액 PCR법
시간	24시간 이내	30분 이상 (양성시, 추가 24시간 필요)	24시간 이내
민감도/ 특이도*	민감도 : 98% 이상 특이도 : 100%	민감도 : 90% 특이도 : 96%	민감도 : 92% 특이도 : 100%
검체	비인두도말	비인두도말	타액(침)
절차	검체채취 → PCR ⇒ 확진	검체채취 → 현장검사 ⇒ (양성) → 검체 재채취 → PCR ⇒ 확진	타액 채취 → PCR ⇒ 확진

* 현재까지 허가 또는 긴급사용승인된 제품에 대한 민감도 및 특이도

출처: 질병관리청

그러면서 그는 "다양한 플랫폼을 이용해 치료제가 발굴되거나 개발된다고 하더라돈 이 치료제가 대량으로 사용될 경우 약제 내성도 등장할 수 있다고 전문가들은 경고한다"며 "비유하자면 아직 첫 번째 산을 넘기도 전이지만 그 뒤에 연달아 또다른 산들이 기다리고 있음을 알고 있고 또 긴장하고 있다"고 말했다. 결국 백신이 개발돼 지역사회 접종이 완료될 때까지는 방심할 수 없는 상황이 지속될 것이라는 판단이다.

항체 검사 시기와 실시 지역, 방법에 대해서는 아직 정해지지 않았다. 권 부본부장은 "대구, 경북 지역이 가장 사례수가 많은 지역이기 때문에 우선적으로 항체 검사를 실시하는 것이 합리적이라는 생각을 하고 있다"며 "장기적으로 전국민 또는 합리적인 표본을 대상으로 시행하는 것도 필요할 것"이라고 말했다.

권 부본부장은 또 "항체 검사를 할 때 결정해야 할 기술적이고 전문적인 내용이 많다"며 "시약의 정확도와 민감도, 특이도 등을 해석해 합리적인 결과가 나올 수 있는 시약이 필요하고 검체 확보 방안과 시기 등을 고려해야 한다"고 말했다.

그는 또 "설령 아주 소규모의 환자나 또는 그런 집단발병의 단초만 보인다 하더라도 훨씬 더 많은 환자가 있을 수 있다는 생각을 해야 한다"며 "접촉자가 있게 되면 90% 이상을 찾아내서 관리를 해야만 전파를 차단했다고 얘기할 수 있다고 전문가들이 얘기하기 때문에 전체 신규 환자 규모가 줄어들고 있다 하더라도 방역에 최선을 기울이도록 하겠다"고 덧붙였다.

http://dongascience.donga.com/news.php?idx=36351
https://www.docdocdoc.co.kr/news/articleView.html?idxno=2005869

모델을 이용한 분류 결과의 정확도를 평가하는 방법

- 정오분류표를 이용하여 분류 결과와 실제 값을 비교하여 나타냄
- 종합적인 평가 척도인 정확도 외에 민감도, 특이도, 정밀도, 재현율 등 다양한 평가 척도 사용 가능
- ROC 곡선과 AUC를 사용하여 다양한 모델의 특성을 비교, 요구조건에 적합한 모델 선택 가능