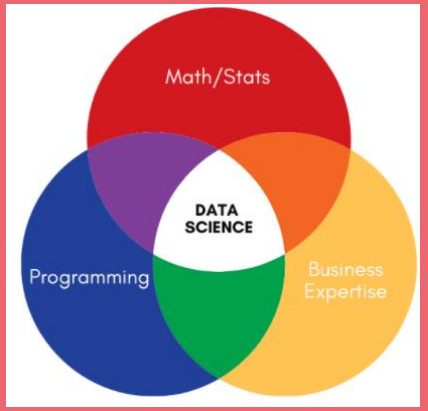
Decision Tree

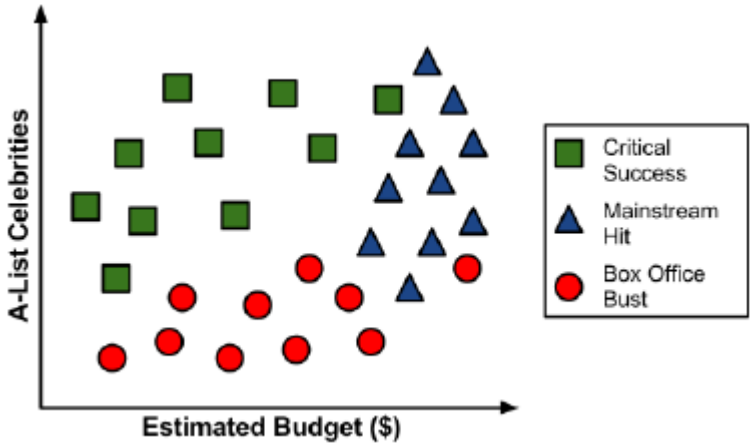
인하대학교
데이터사이언스 학과
김 승 환

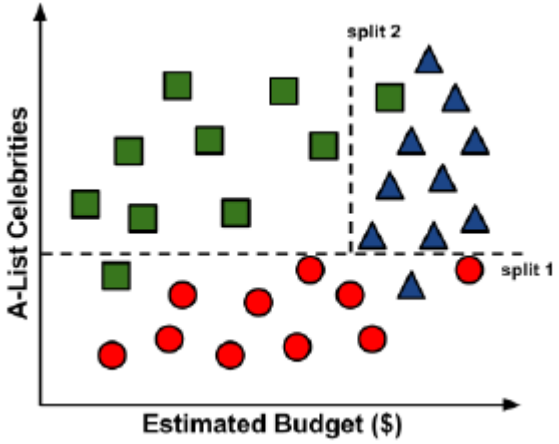
swkim4610@inha.ac.kr



1 Decision Tree

- 좌측 그래프는 영화의 예산액과 유명 배우 수에 따른 영화 흥행 정도를 표시한 것이다.
- Mainstream Hit > Critical Success > Box office Bust 순임
- 흥행 정도를 잘 구분하는 선이 존재한다. 선을 그려본 것이다.
- 영화의 흥행 정도를 선으로 잘 나눌 수 있음을 알 수 있다.
- 이를 응용하면 영화를 개봉하기 전에 위와 같은 그래프를 그려 자신의 영화가 어디에 놓이는지 알면 영화 흥행여부를 예측할 수 있을 것이다.

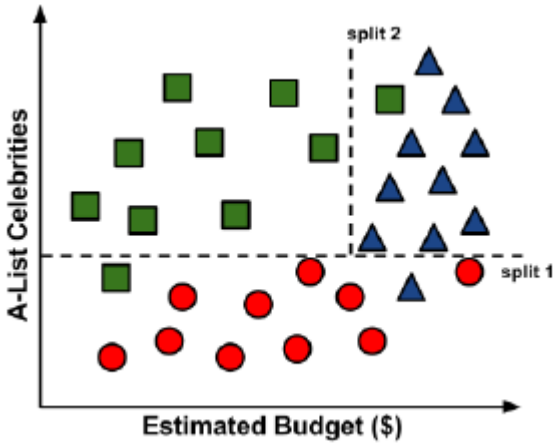


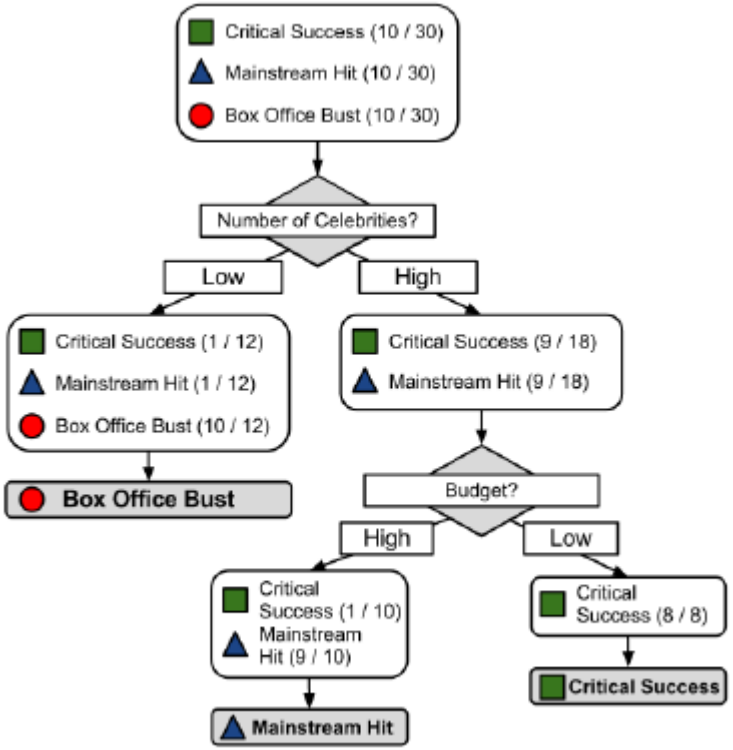


1 Decision Tree

불순도

- 이 로직은 아래와 같은 Tree 형식으로 표현가능 하다.
- 다차원 공간을 나누는데 불순도(impurity)가 낮게 직선으로 나누는 아이디어에서 출발한다.

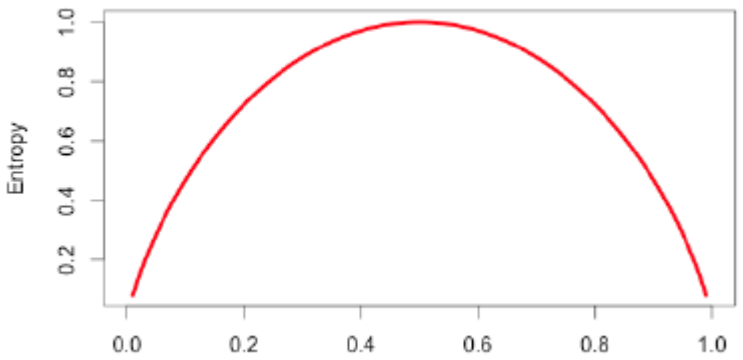




Decision Tree

엔트로피

$$G(S) = 1 - \sum_{i=1}^{c} p_i^2 \qquad \text{Entropy}(S) = \sum_{i=1}^{c} -p_i \log_2(p_i)$$



- 바구니에 Red 구슬이 60%, White가 40% 들어 있다면 이 때 Entropy는
- -0.6*log2(0.6) -0.4*log2(0.4) = 0.97이 된다.
- 지니 계수는 1 - (0.6**2 + 0.4**2) = 0.48이 된다.
- 엔트로피나 지니 계수 모두 p=0.5일 때 불순도가 최고이기 때문에 최대값을 가진다.

Decision Tree

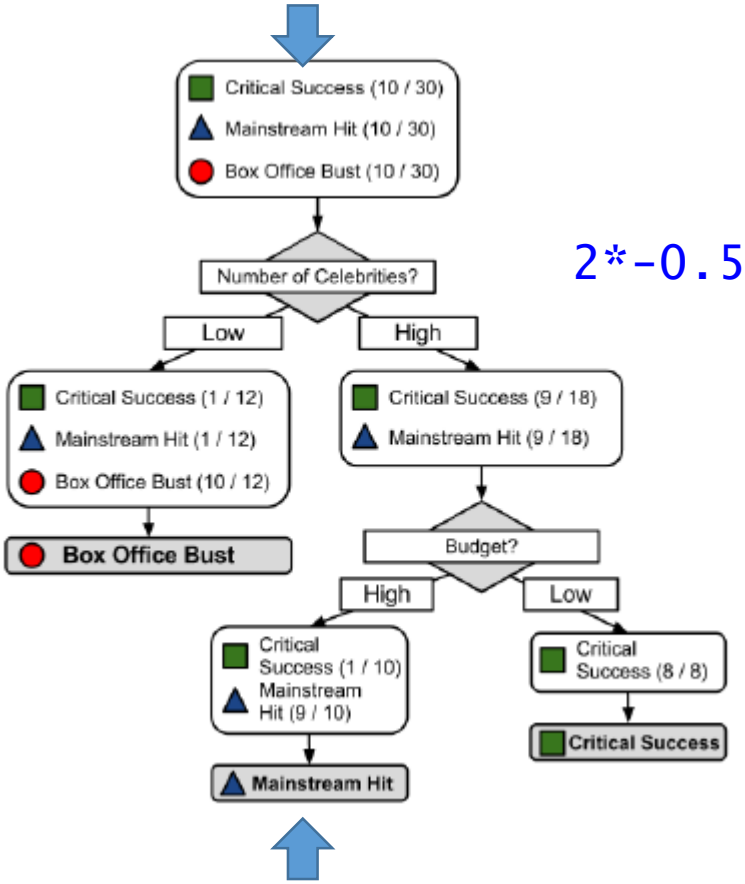
엔트로피

Entropy가 작아지는

방향으로

Tree를 만들어 간다.

3*-0.33*log2(0.33)=1.58



2*-0.5*log2(0.5)=1

2*-1/12*log2(1/12)-10/12*log2(10/12)=0.82

-1*log2(1)=0

-0.1*log2(0.1)-0.9*log2(0.9)=0.47

Decision Tree

Information Gain

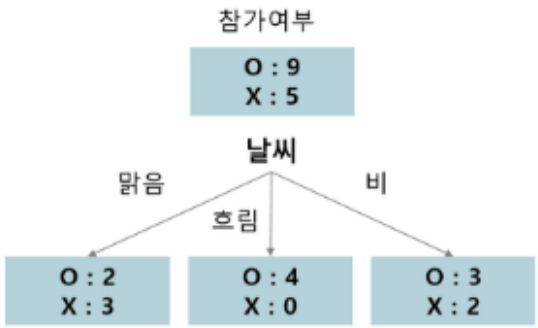
- 트리 분기는 Information Gain에 의해 이루어 진다.
- 루트 다음에 날씨로 분기할 때 Information Gain은 아래와 같이 계산된다.

날짜	날씨	온도	습도	바람	참가여부
D1	맑음	더움	높음	약함	X
D2	맑음	더움	높음	강함	X
D3	흐림	더움	높음	약함	O
D4	비	포근	높음	약함	O
D5	비	서늘	정상	약함	O
D6	비	서늘	정상	강함	X
D7	흐림	서늘	정상	강함	O
D8	맑음	포근	높음	약함	X
D9	맑음	서늘	정상	약함	O
D10	비	포근	정상	약함	O
D11	맑음	포근	정상	강함	O
D12	흐림	포근	높음	강함	O
D13	흐림	더움	정상	약함	O
D14	비	포근	높음	강함	X

$$E(\text{경기}) = -\frac{9}{14}log_2(\frac{9}{14}) - \frac{5}{14}log_2(\frac{5}{14}) = 0.940$$

날씨

$$E(\text{경기}|\text{날씨}) = \frac{5}{14}\left(-\frac{2}{5}log_2\left(\frac{2}{5}\right)-\frac{3}{5}log_2\left(\frac{3}{5}\right)\right) + \frac{4}{14}\left(-\frac{4}{4}log_2\left(\frac{4}{4}\right)-\frac{0}{4}log_2\left(\frac{0}{4}\right)\right) + \frac{5}{14}\left(-\frac{3}{5}log_2\left(\frac{3}{5}\right)-\frac{2}{5}log_2\left(\frac{2}{5}\right)\right) = 0.694$$



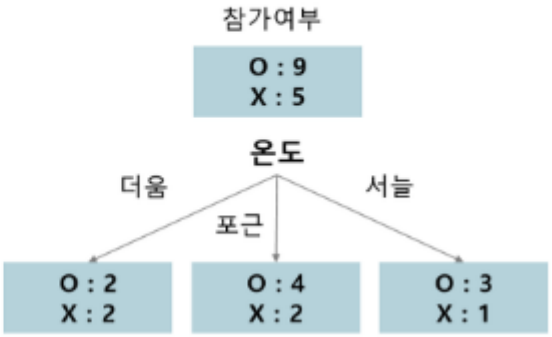
Decision Tree

Information Gain

- 같은 방식으로 온도, 습도 바람에 대해 각각 엔트로피를 계산하면 아래와 같다.

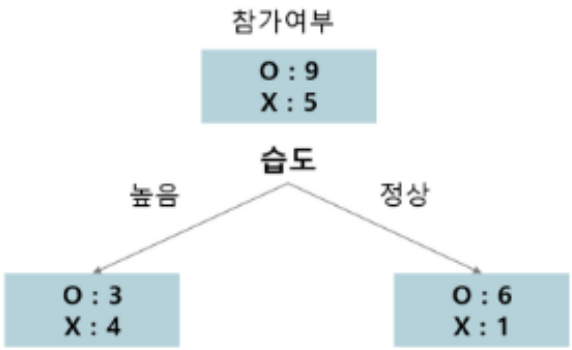
온도

$$E(\text{경기}|\text{온도}) = \frac{4}{14}\left(-\frac{2}{4}log_2\left(\frac{2}{4}\right)-\frac{2}{4}log_2\left(\frac{2}{4}\right)\right) + \frac{6}{14}\left(-\frac{4}{6}log_2\left(\frac{4}{6}\right)-\frac{2}{6}log_2\left(\frac{2}{6}\right)\right) + \frac{4}{14}\left(-\frac{3}{4}log_2\left(\frac{3}{4}\right)-\frac{1}{4}log_2\left(\frac{1}{4}\right)\right) = 0.911$$



습도

$$E(\text{경기}|\text{습도}) = \frac{7}{14}\left(-\frac{3}{7}log_2\left(\frac{3}{7}\right)-\frac{4}{7}log_2\left(\frac{4}{7}\right)\right) + \frac{7}{14}\left(-\frac{6}{7}log_2\left(\frac{6}{7}\right)-\frac{1}{7}log_2\left(\frac{1}{7}\right)\right) = 0.789$$



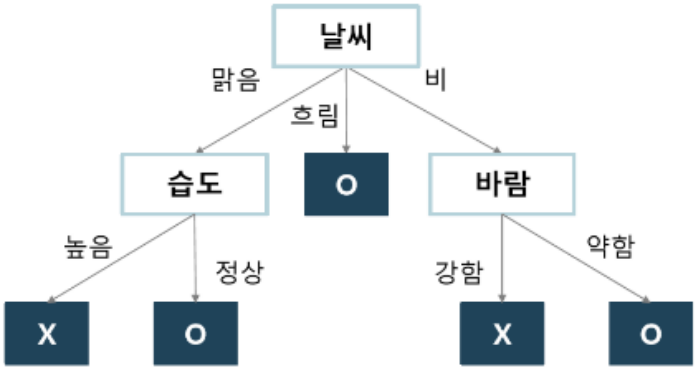
Decision Tree

Information Gain

- 날씨의 information gain이 0.94-0.694 = 0.246 로 가장 크므로 첫 분기는 날씨로 정한다.
- 날씨 맑음을 루트노트로 보고 5개의 관측값에 대해 위의 과정을 반복하여 1개가 남을 때까지 반복한다.
- 이 과정을 맑음, 흐림, 비에 대해 반복한다.

날짜	날씨	온도	습도	바람	참가여부
D1	맑음	더움	높음	약함	X
D2	맑음	더움	높음	강함	X
D8	맑음	포근	높음	약함	X
D9	맑음	서늘	정상	약함	O
D11	맑음	포근	정상	강함	O

최종결과



Decision Tree

Information Gain

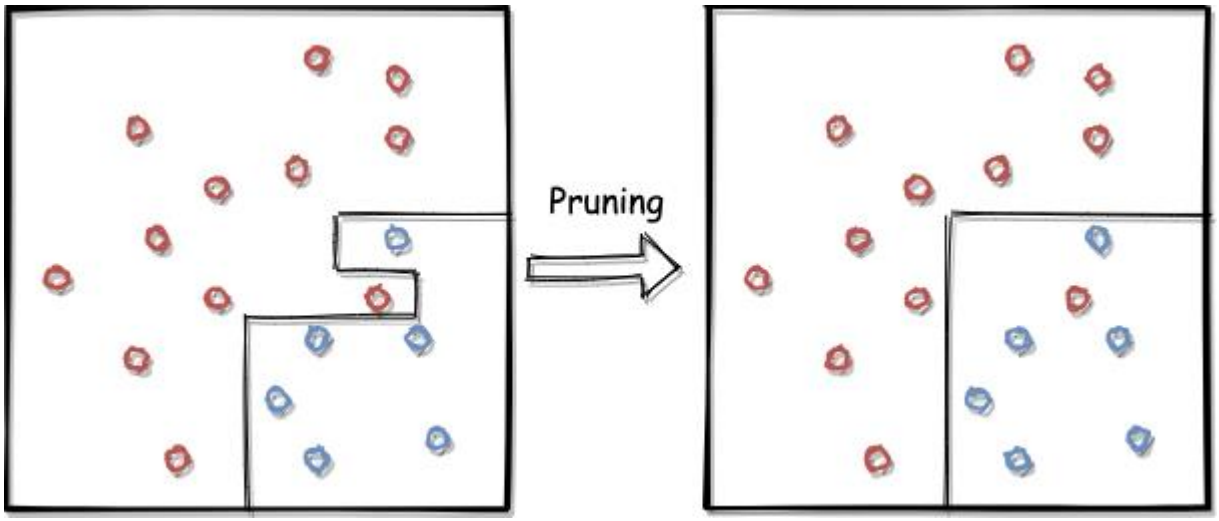
- 독립변수가 연속형이면 이전 예처럼 어느 포인트에서 나눠야 하는지 결정해야 한다.
- 모든 데이터를 하나 하나 기준점으로 삼아 나누는 방법과 중위수나 사분위수로 나누는 방법도 있을 수 있고, 클래스가 바뀌는 지점을 기준으로 하는 경우도 있다.
- Income을 소트한 후, 종속변수가 바뀌는 지점은 (59.4, 60), (64.8, 65), (84, 85.8) 로 각각 두개 값의 평균이 된다.
- 기준값이 결정되면 범주형 자료처럼 그룹 단위로 엔트로피를 계산하면 된다.

INCOME	LOTSIZE	OWNERSHIP
43.2	17.2	X
49.2	17.6	X
52.8	19.6	X
59.4	17.6	X
60	18.4	O
61.5	21	O
64.8	21.6	O
65	20.8	X
84	20.4	X
85.8	16.8	O
87	23.6	O
110.1	19.2	O

Decision Tree

가지치기

- 공간을 선으로 자르면서 뿌리에서 잎으로 갈수록 데이터의 수가 줄어 들어 Overfitting이 된다.
- 이를 방지하기 위해 마지막 레코드 수를 지정하는 것이 필요하다.
- 또한, 전문가가 규칙의 타당성을 검토하여 무리한 가지치기는 제거하는 것도 중요하다.



Decision Tree

장단점

- 장점:
  - ❒ 머신러닝 결과를 이해할 수 있고, 설명이 가능함(당신이 대출이 안되는 이유를 설명함)
  - ❒ 범주형 변수 적용이 쉬움
  - ❒ 실무에 적용이 비교적 쉬움
  - ❒ 비선형적 관계를 가지는 경우, 비교적 높은 정확도를 가짐
  - ❒ 이상치에 덜 민감함

- 단점
  - ❒ 연속형 변수를 이산형 변수로 만들어 예측력을 저하시킴
  - ❒ 학습하지 않은 상황에 예측력이 떨어짐
  - ❒ 과적합 가능성이 높음

Decision Tree

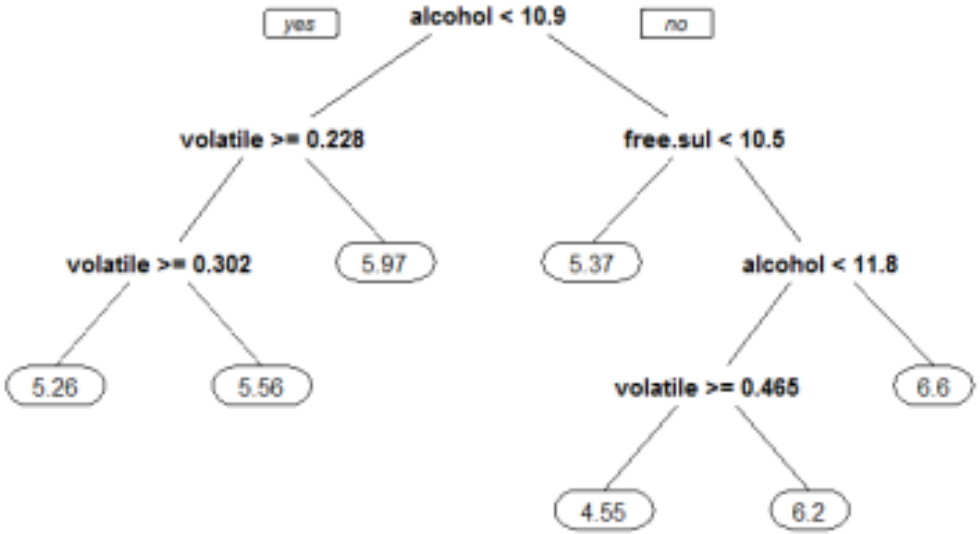
실습

- 버섯은 건강식품이지만 독버섯을 잘못 먹으면 사망할 수 있다.

2 Decision Tree Regression

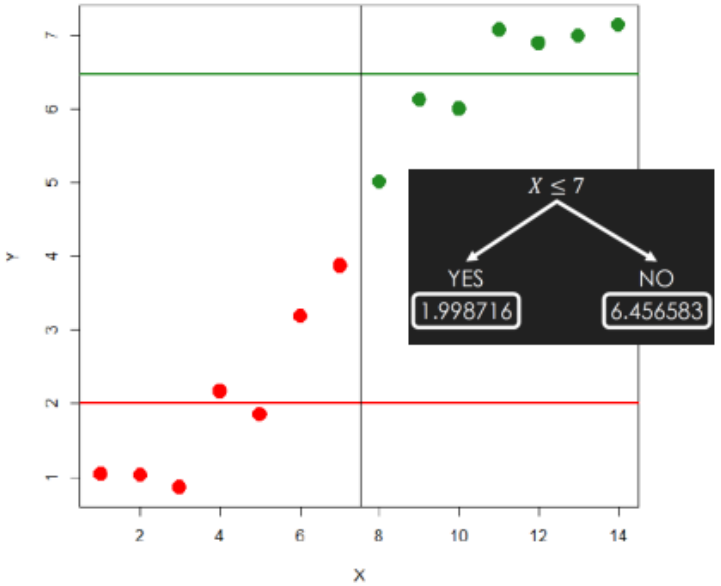
회귀문제

- Decision Tree는 분류 문제 뿐 아니라 회귀문제에도 적용할 수 있다.
- 특히, feature가 많은 다차원 비선형 회귀 문제에 잘 작동하여 빅데이터 분석에서 많이 사용한다.
- 아래는 와인의 Quality 점수를 추정하는 의사결정나무 예다.
- 좋은 와인은 알코올, 황(sulfur) 함량이 크고, 휘발산(volatile) 함량이 작다.



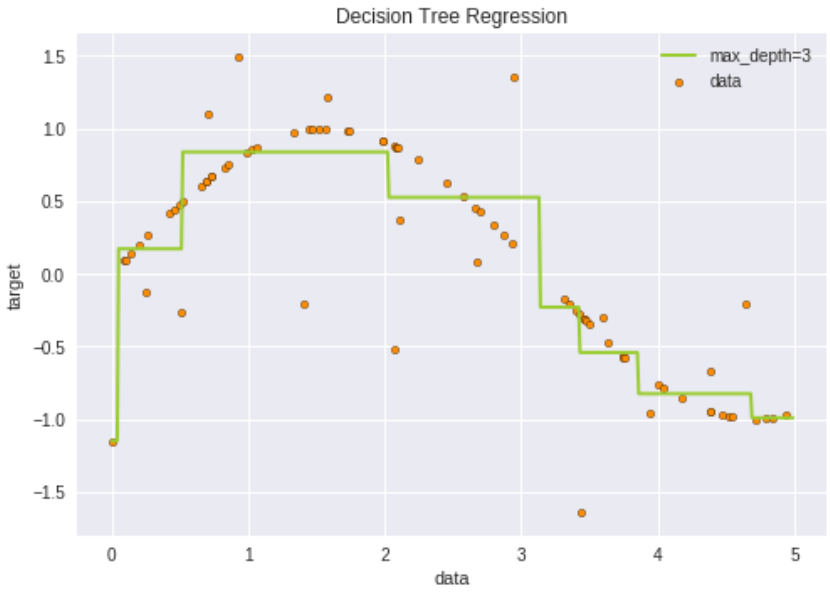
Decision Tree Regression

- Decision Tree Regression은 SDR(Standard Deviation Reduction) 을 최대화하는 방향으로 평면을 분할한다.
- 아래 그림에서 X = 7 을 기준으로 데이터를 나누면 데이터가 쪼개지고 Y 평균이 각각 2와 6.5로 변한다. 이 때, $SDR = SD(Y) - \overline{SD(Y_i)}$, $\overline{SD(Y_i)}$는 두개 그룹의 표준편차 가중 평균이다.
- 이렇게 나누어 가다 보면 동질적인 개체들의 하나의 그룹으로 만들 수 있다.



Decision Tree Regression

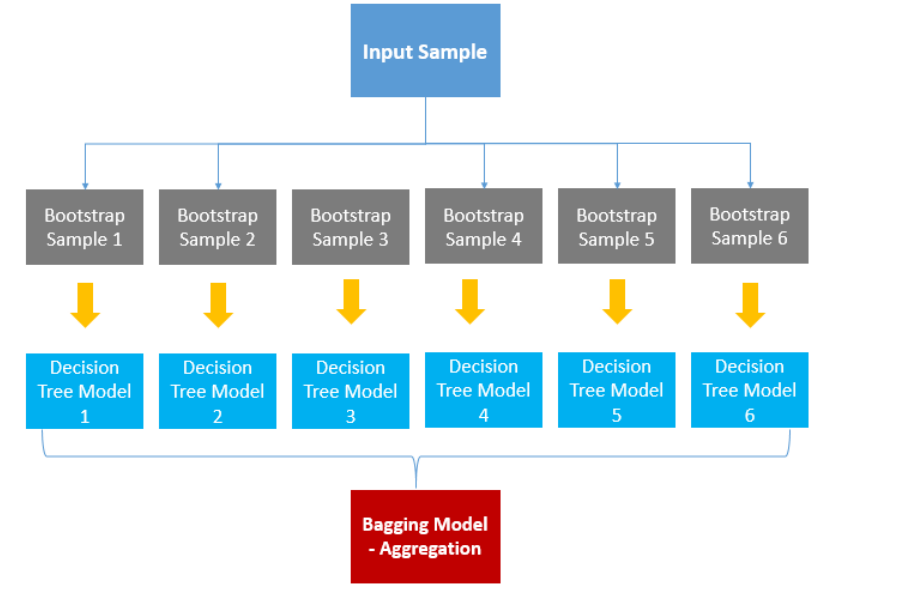
- 아래는 Decision Tree Regression을 통해 SDR이 어떻게 회귀를 수행하는지 알 수 있다.



3 Ensemble

앙상블

- 앙상블은 여러 모형을 합쳐서 하나의 결과를 만드는 기법으로 Decision Tree를 앙상블을 통해 정확도를 개선할 수 있다. 앙상블 방법은 Bagging, Boosting 방법으로 나눠진다.
- 배깅은 Bootstrap Sampling을 이용해 여러 개의 Tree를 만들고 Voting 한다.
- 배깅은 직전 결과와 상관없이 다음 표본이 만들어지지만, 부스팅은 직전결과를 반영하여 다음 표본을 만들어 간다.
- 앙상블 방법은 k개의 노드에 병렬처리가 가능하여 빅데이터 처리에 적합하다.



Ensemble

Bagging

- 부스트랩은 n개 표본에서 m개의 표본을 with-replacement 방식으로 선택한다.
- 그러면 하나의 표본이 또 선택될 가능성이 있으므로 100개에서 100개의 추출하면 100개 보다 작은 숫자의 표본이 만들어 진다. 이론적으로는 1 - 0.99**100 = 63.4% 만 추출된다.
- 즉, 특정 표본 'i' 가 부스트랩 표본에 속할 확률은 (1 – 안 뽑힐 확률) 이다.
- 안 뽑힐 확률은 100번 모두 선택되지 않을 확률로 0.99^{100} 이다.

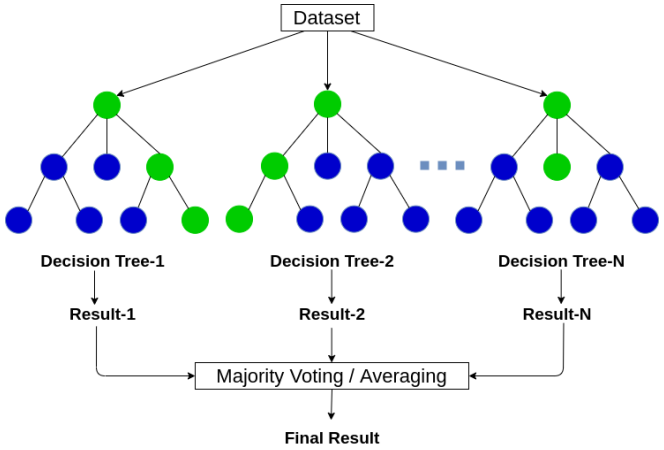
$$Pr(i \in \text{Bootstrap Sample}) = 1 - 0.99^{100} = 0.634$$

```
import random
sample = list(range(0,100))
bootstrap_sample = set()

for i in range(100):
    bootstrap_sample.add(random.choices(sample)[0])

len(bootstrap_sample)
```

Ensemble

Random Forest

- 랜덤 포레스트는 의사결정나무를 배깅(Bagging) 알고리즘으로 구현한 모형이다.
- 배깅은 부스트랩(Bootstrap) 표본 추출법을 이용하여 하나의 데이터셋을 여러 개의 데이터 셋으로 만들어 표본 변동에 강한 의사결정 룰을 만드는 기법이다.
- k개의 의사결정나무는 부스트랩 표본으로 데이터를 재구성하고 feature 역시 부스트랩으로 추출한다.
- 그러므로, k개 트리는 데이터도 다르고, 사용된 feature도 다르므로 약간씩 다른 결과를 낼 것이다.
- 랜덤 포레스트는 이렇게 다른 k개의 결과를 모아 투표에 의해 최종 결과를 산출한다.
- 이는 앙상블(Ensemble) 모형의 일종으로 분류, 회귀 문제에 다 적용 가능하다.



Boosting

AdaBoost

- Ada Boost는 Adaptive Boosting의 줄임 말로 부스팅 모형이다.
- 부스트랩 표본을 추출하는 과정에서 이전에 오 분류된 개체가 다음 표본에 포함될 확률을 증가시켜 오 분류 표본의 학습량을 늘려주는 기법이다.
- 또한, k개의 부스트랩 트리의 결과를 통합할 때에도 오 분류가 낮은 트리의 가중치를 높여 최종결과를 산출한다.
- $D_1(i)$는 AdaBoost 시작 시점에 i번째 표본이 추출될 확률로 시작 시점에는 확률을 uniform 하게 준다.
- ε_1은 첫 번째 모형의 오분류율이다. ε_1 은 0.5 보다 작은 값으로 가정한다. 만약, 0.5 이상이면 해당 모형은 버린다.
- α_1 은 첫번째 모형의 중요도를 나타내는 가중값으로 ε_1이 0.5일 때, 0이 되고, 0.5 보다 작아지면 + 이다.

$$D_1(i) = \frac{1}{m} \quad \alpha_1 = \frac{1}{2} log\left(\frac{1-\varepsilon_1}{\varepsilon_1}\right)$$

- 이후, t 번째 표본의 i 번째 관측값에 대한 추출 확률은 i 번째 값이 정분류일 경우, $\exp(-\alpha_t)$ 즉, 1보다 작은 값을 곱해 확률을 감소시키고, 오분류일 경우 1보다 큰 값을 곱해 추출될 확률을 증가시킨다.

$$if\ h_t(x_i) = y_i, D_{t+1}(i) = \frac{D_t(i)}{Z_t}\exp(-\alpha_t) \quad \leftarrow \text{정분류의 경우}$$

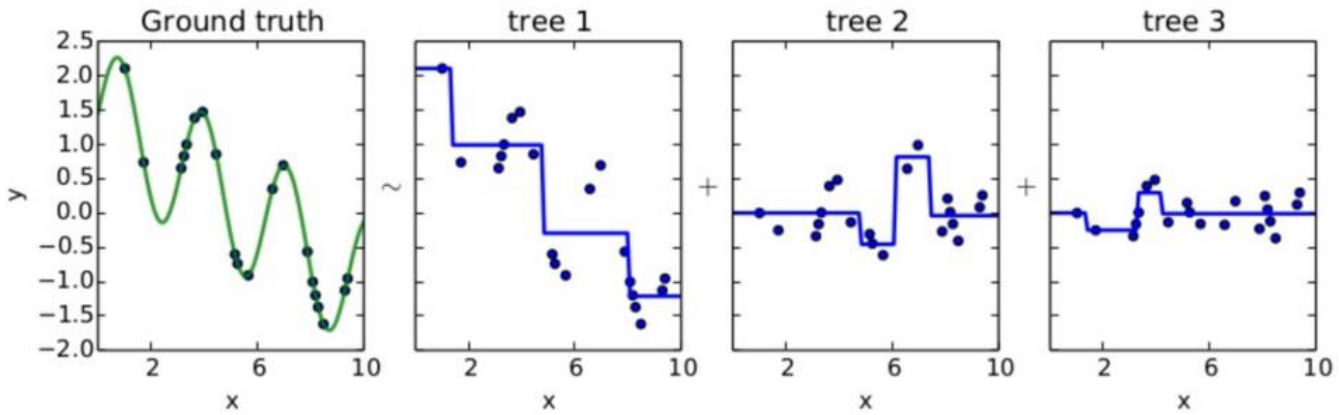
$$if\ h_t(x_i) \neq y_i, D_{t+1}(i) = \frac{D_t(i)}{Z_t}\exp(\alpha_t) \quad \leftarrow \text{오분류의 경우}$$

여기서, Z_t 는 가중치의 합이 "1" 이 되도록 하는 조정 값이다.

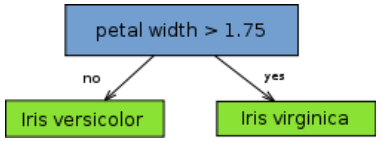
4 Boosting

Gradient Boost

- 그래디언트 부스트는 앙상블 알고리즘에 경사하강(Gradient Descent) 알고리즘을 적용한 것이다.
- Gradient Boost는 일종의 Residual 학습모형이다.
- Residual 학습이란 첫번째 모형으로 예측을 하고 얻어진 잔차를 다음 모형이 학습하는 것이다.
- 이렇게 이전 모형의 결과에 다음 모형 결과를 합쳐 간다.



- 각 모형은 stump tree와 같은 weak classifier를 사용한다.



Boosting

실습

- 아래와 같은 데이터에서 Weight를 예측하는 모형을 만든다고 가정하자.
- 처음에는 아무런 정보가 없으므로 Weak Model은 몸무게 평균값을 사용하는 것으로 하자. 평균값은 71.2 이므로 모든 사람의 키를 71.2로 예측한다. 예측오차가 발생한다.
- 너무나 간단한 모형으로 예측했기 때문에 오차(residual)에는 독립변수인 키, 선호 색상, 성별로 부터 설명가능한 요인이 많이 남아 있을 것이다.

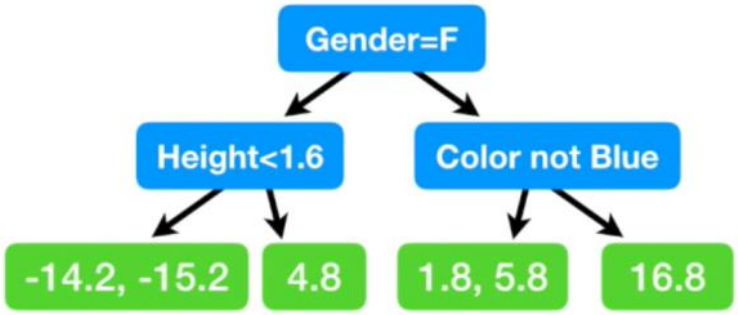
Height (m)	Favorite Color	Gender	Weight (kg)
1.6	Blue	Male	88
1.6	Green	Female	76
1.5	Blue	Female	56
1.8	Red	Male	73
1.5	Green	Male	77
1.4	Blue	Female	57

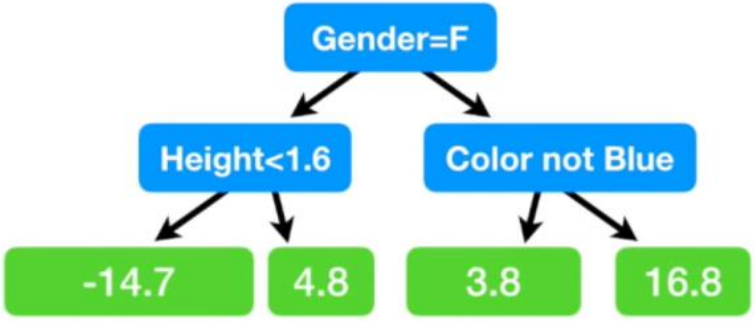
Height (m)	Favorite Color	Gender	Weight (kg)	Residual
1.6	Blue	Male	88	16.8
1.6	Green	Female	76	4.8
1.5	Blue	Female	56	-15.2
1.8	Red	Male	73	1.8
1.5	Green	Male	77	5.8
1.4	Blue	Female	57	-14.2

4 Boosting

실습

- Gradient Boosting은 잔차를 독립변수로 부터 설명하는 Weak Model을 만들어 잔차의 크기를 줄여가는 과정을 반복한다.
- 이 때, 잔차의 크기가 줄어들어 추정 값이 관찰 값에 가까이 가는 과정이 Gradient Descending algorithm과 같아 Gradient Boosting 모형이라고 부른다.
- Weak Model이 아래와 같다고 가정하자.
  성별이 여자고, 키가 1.6 미만인 경우는 (-14.2-15.2)/2 = -14.7 로 잔차를 예측한다는 의미다.
- 여자이면서 키가 작으면 체중이 작으므로 초기값 71.2로 예측하는 것은 보다는 몸무게를 작게 71.2 - 14.7 = 56.5로 예측해야 한다는 의미다.
- 그 다음으로 여자이고 키가 1.6 이상이면 초기값 71.2+4.8 로 예측한다는 의미다.





Boosting

실습

- 그런데, 현재 사용한 모형은 weak model이기 때문에 전적으로 신뢰할 수 없다.
- 그래서 예측값 갱신에 learning rate(eta)를 사용한다.
- eta = 0.1 이라고 가정하면 잔차예측값의 10% 만큼만 반영하여 한걸음 다가가는 것을 의미한다.

새로운 예측값 = 이전예측값 + eta * 잔차 예측값

- Gradient Boosting은 위의 식처럼 잔차가 작아지는 방향으로 예측을 수행하는 기법을 말한다.

```
F_new = np.mean(y) #초기화 모형
tree = DecisionTreeRegressor(max_depth =1) #약한 학습기
eta = 0.1 #학습률
T = 100
models = []
for t in range(T):
    F_old = F_new
    r     = y - F_old          # 잔차
    r_fit = tree.fit(x,r)      # 잔차 적합
    gamma = r_fit.predict(x)
    models.append(copy.deepcopy(r_fit))
    F_new = F_old + eta * gamma # 새로운 예측값
```

- GBM의 단점(느리다, 오버피팅)을 보완하는 연구로 탄생한 것이 XGBoost 알고리즘이다.
- XG Boost는 Extreme Gradient Boosting을 의미하며 R, 파이썬 라이브러리로 제공된다.
- 속도개선을 위해 멀티 스레드를 멀티코어 CPU에 배분하여 병렬처리를 지원한다.
- Loss Regularization을 통해 오버피팅 이슈를 해결하였다. (Ridge Regression 참고)

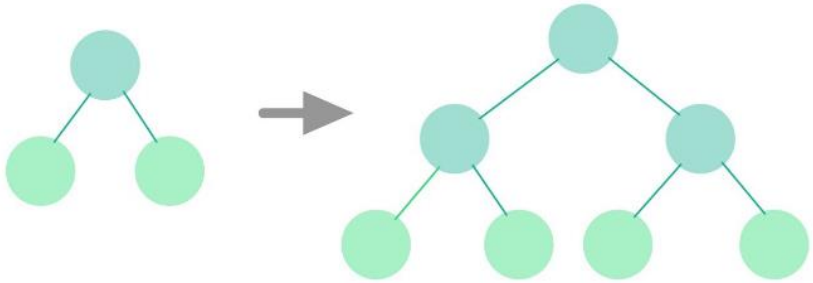
In ridge regression the objective is to minimize

$$\min_{\beta \in \mathbb{R}^p} \left\{ \frac{1}{N} \|y - X\beta\|_2^2 + \lambda \|\beta\|_2^2 \right\}$$

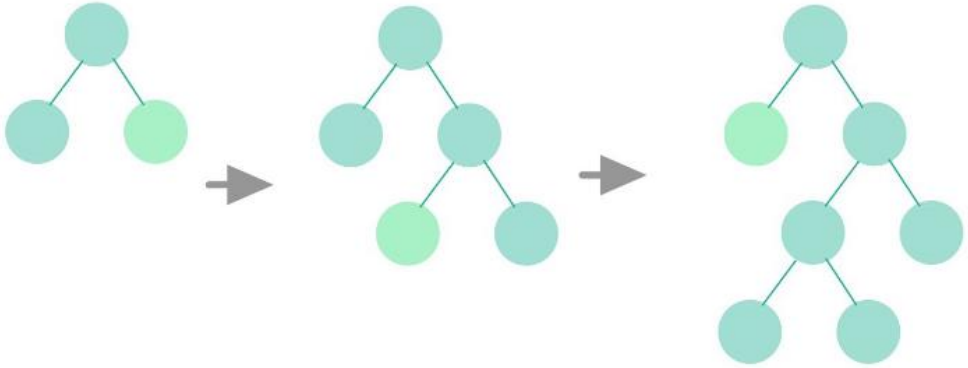
- 그 밖에도 최적 분기점 찾기, 결측값 자동처리(정말 편리함), Tree Pruning 자동화 기능을 제공하여 Kaggle 에서 LightGBM과 함께 가장 인기있는 알고리즘이다.

- Light GBM은 가벼워 속도가 빠른 Gradient Boosting 알고리즘이다.
- 또한, 오버피팅 단점을 피하기 위해 아래와 같이 각 노드에서 Stump Tree를 이용해 이분할 하는 일반적 GBM과 달리 Loss가 큰 노드만 골라 이분할을 깊게 수행하여 시간을 절약한다.

일반적 GBM



Light GBM



4 Boosting

Grid Search

- XGBoost, LightGBM 모형은 모두 Gradient Boosting 모형이다.
- 이 모형을 사용할 때 결정해줘야 하는 파라미터가 있는데 파라미터 조합이 너무 많아 이를 쉽게 찾을 수 있도록 도와 주는 것이 그리드 탐색법이다.
- 주요 파라미터는 아래와 같다.

boosting: (gdbt, rf, dart, goss) 중 선택

max_depth: 개별 트리 최대 깊이(이 값이 크면 오버피팅한다)

min_data_in_leaf: leaf 노드 최소 레코드 수(이 값이 작으면 오버피팅한다)

feature_fraction: boosting = rf에서 각 트리에서 사용할 feature 비율(0.8: 80% 피처만 사용)

bagging_fraction: 매 iteration 시 전체 데이터에서 사용할 데이터 비율

lambda: Ridge Regression 조절모수

learning_rate: 경사하강 속도조절 모수(0.001, 0.01, 0.1 등)

num_boost_round: 부스팅 회수(100~)

4 Boosting

실습

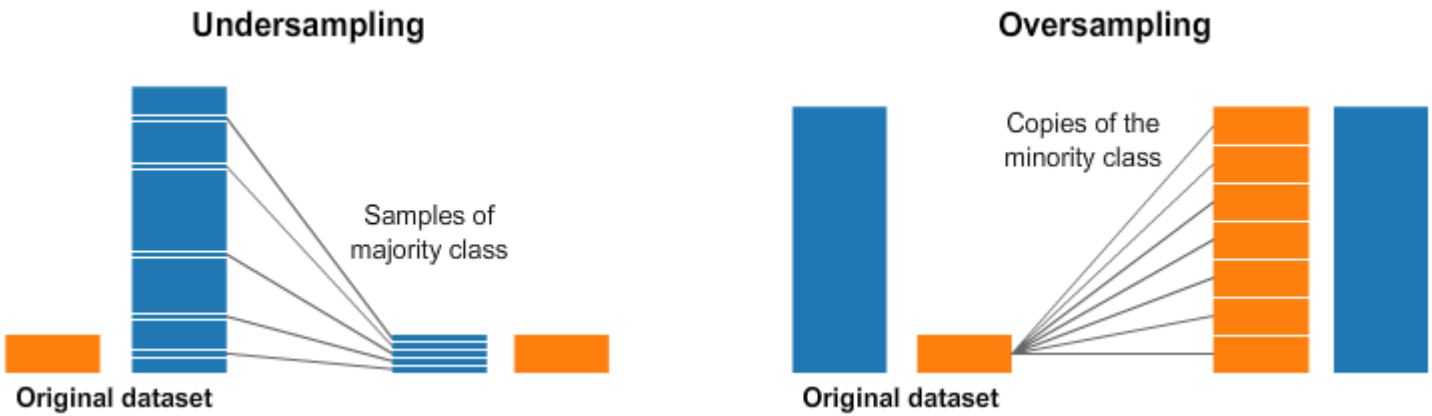
- 아래는 보스톤의 집값 데이터다.
- 분석 목적은 아래 독립변수로 집값을 추정하는 것이다.

```
Boston house prices dataset
---------------------------

**Data Set Characteristics:**

    :Number of Instances: 506

    :Number of Attributes: 13 numeric/categorical predictive. Median Value (attribute 14) is usually the target.

    :Attribute Information (in order):
        - CRIM     per capita crime rate by town
        - ZN       proportion of residential land zoned for lots over 25,000 sq.ft.
        - INDUS    proportion of non-retail business acres per town
        - CHAS     Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
        - NOX      nitric oxides concentration (parts per 10 million)
        - RM       average number of rooms per dwelling
        - AGE      proportion of owner-occupied units built prior to 1940
        - DIS      weighted distances to five Boston employment centres
        - RAD      index of accessibility to radial highways
        - TAX      full-value property-tax rate per $10,000
        - PTRATIO  pupil-teacher ratio by town
        - B        1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town
        - LSTAT    % lower status of the population
        - MEDV     Median value of owner-occupied homes in $1000's
```

5 Class Imbalance

Over/Down Sample

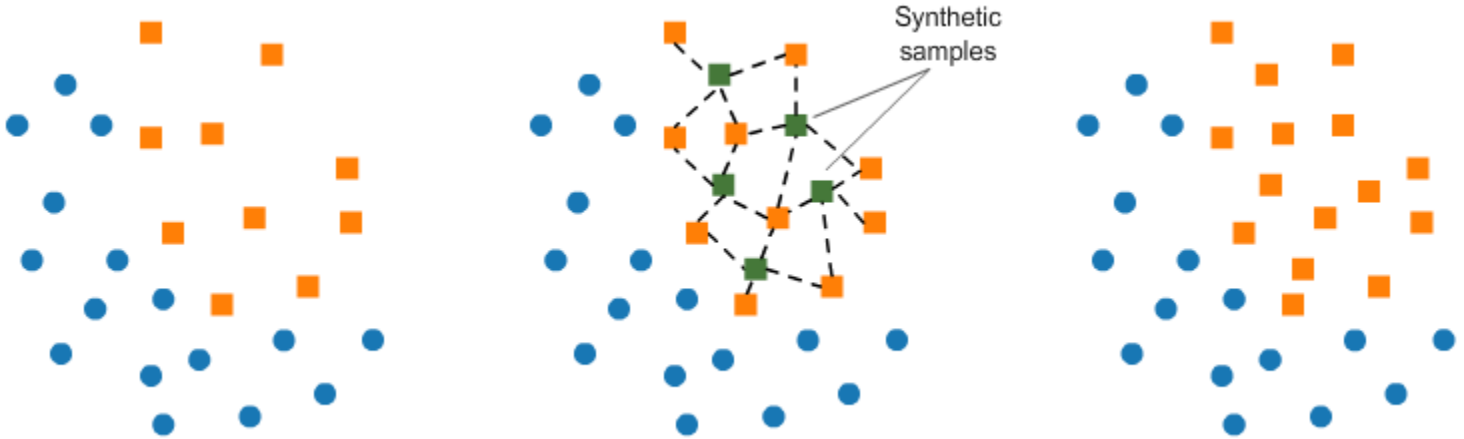
- 분류문제에서 흔히 겪는 문제로 클래스 숫자의 불균형 문제가 있다.
- kNN에서 실습한 암환자의 경우, Benign이 458, Malignant가 241 Case 였다.
  이 경우, Classifier는 훈련 데이터에서 오분유율을 낮추기 위해 Benign 쪽으로 판정하려는 경향이 생긴다.
- 해법으로 P(악성) > 0.4 이면 악성으로 판정해 악성으로 판정을 많이 하도록 하는 방법을 사용할 수 있다.
- 또 다른 해법으로 Over Sampling, Down Sampling을 이용하는 것이다.



5 Class Imbalance

SMOTE

- 그 밖에 SMOTE라는 데이터 생성기법이 있다.
- SMOTE는 Minor Case에서 생성자료를 만드는 기법으로 기존의 점에서 kNN을 이용해 최근접 이웃을 찾고 나와 이웃 사이에 보간법을 이용해 포인트를 추가하는 방식이다.
- SMOTE가 앞의 기법보다 더 좋다는 보장은 없다.



Over_Under.ipynb

5 Class Imbalance

실습

- Pima 인디언 관련 당뇨병 데이터를 보고 당뇨병을 예측(#9 Class Variable)하는 것이다.

```
# 1. Number of times pregnant(임신회수)
# 2. Plasma glucose concentration a 2 hours in an oral glucose tolerance test: 글루코즈
# 3. Diastolic blood pressure (mm Hg) : 혈압
# 4. Triceps skin fold thickness (mm) : 팔근육량
# 5. 2-Hour serum insulin (mu U/ml) : 인슐린
# 6. Body mass index (weight in kg/(height in m)^2): BMI
# 7. Diabetes pedigree function : 가족력
# 8. Age (years) : 나이
# 9. Class variable (0 or 1)
```

감사합니다

인하대학교
데이터사이언스 학과
김 승 환

swkim4610@inha.ac.kr