



PART II-03

머신러닝 기본

시간계획

- 오늘도 파이팅 입니다.^ ^

| 시간 | 학습내용 |
|-------------|---|
| 09:00~10:00 | 회귀모델 이해 및 성능 최적화를 위한 feature engineering 이해 |
| 10:20~11:20 | LinearRegression(단순선형회귀) 실습 |
| 11:40~12:40 | 회귀모델 데이터 학습 및 예측 |
| 12:40~14:00 | 즐거운 점심 시간 |
| 14:00~15:00 | 앙상블 이해 및 알고리즘 살펴보기 |
| 15:20~16:20 | 랜덤포레스트 실습 |
| 16:40~17:50 | XGBoost 실습 |

지도 학습(Supervised Learning)

앙상블 학습모델(Ensemble Learning Model)

학습목표

- 앙상블 학습의 부스팅과 배깅을 이해한다.
- 랜덤 포레스트 알고리즘을 활용해 머신러닝 모델을 만든다.
- XGBoost 알고리즘을 이해하고, 머신러닝 모델을 만든다.
- 결측치를 확인하고, 중위값, 최빈값으로 채운다.
- 여러 학습모델의 성능을 비교 평가한다.

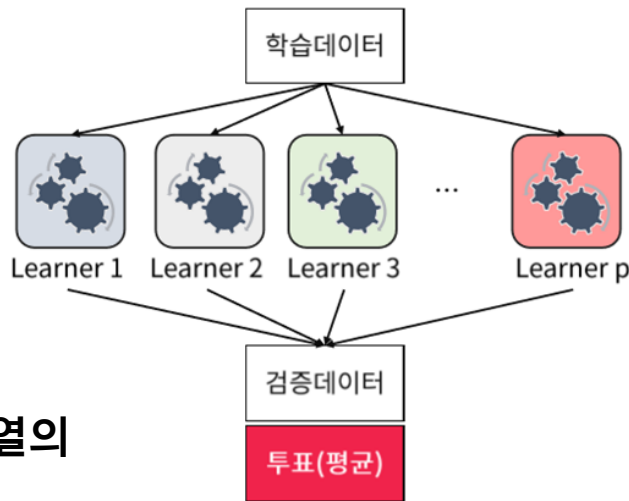
앙상블 학습 개요

- 여러 개의 분류기(Classifier)를 생성하고 그 예측을 결합함으로써 보다 정확한 최종 예측을 도출하는 기법
- 어려운 문제의 결론을 내기 위해 여러 명의 전문가로 위원회를 구성해 다양한 의견을 수렴하고 결정하는 것과 비슷
- 앙상블 학습의 목표는 다양한 분류기의 예측 결과를 결합함으로써 단일 분류기보다 신뢰성이 높은 예측 값을 얻음.



앙상블의 유형

- 보팅(Voting)
- 배깅(Bagging)
 - 랜덤 포레스트(Random forest) 알고리즘
- 부스팅
 - 에이다 부스팅, 그래디언트 부스팅, **XGBoost**, LightGBM
 - **정형 데이터의 분류나 회귀에서 GBM 부스팅 계열의 앙상블이 전반적으로 높은 예측 성능을 나타냄**
- 스택킹(Stacking)
 - 여러가지 모델을 기반으로 메타 모델을 수립함.
- 넓은 의미로 서로 다른 모델을 결합한 것들을 앙상블로 지칭하기도 함



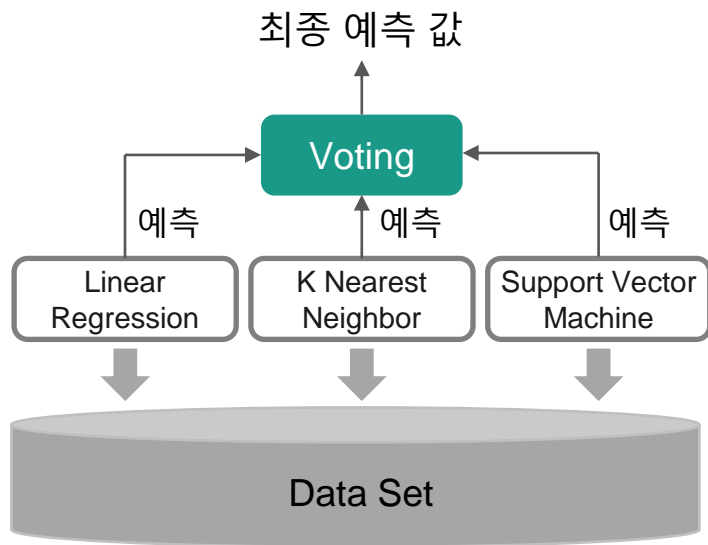
앙상블의 특징

- 단일 모델의 약점을 다수의 모델을 결합하여 보완
- 뛰어난 성능을 가진 모델들로만 구성하는 것보다 성능이 떨어지더라도 서로 다른 유형의 모델을 섞는 것이 오히려 전체 성능에 도움이 될 수 있음.
- 랜덤 포레스트 및 뛰어난 부스팅 알고리즘들은 모두 결정 트리 알고리즘을 기반 알고리즘을 적용함.
- 결정 트리의 단점인 과적합(오버 피팅)을 수십~수천개의 많은 분류기를 결합해 보완하고 장점인 직관적인 분류 기준은 강화함.



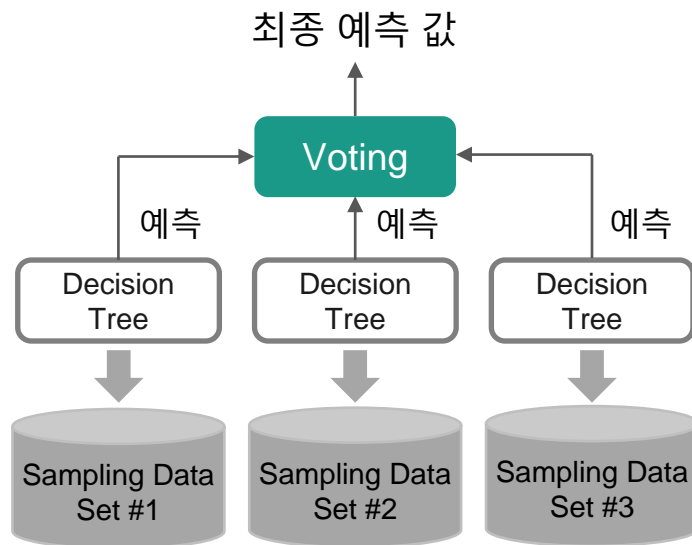
보팅(Voting)과 배깅(Bagging)

보팅(Voting) 방식



보팅은 서로 다른 분류 알고리즘이 같은 데이터 세트를 학습하고 예측한 결과를 보팅을 통해 최종 예측결과를 선정함

배깅(Bagging) 방식

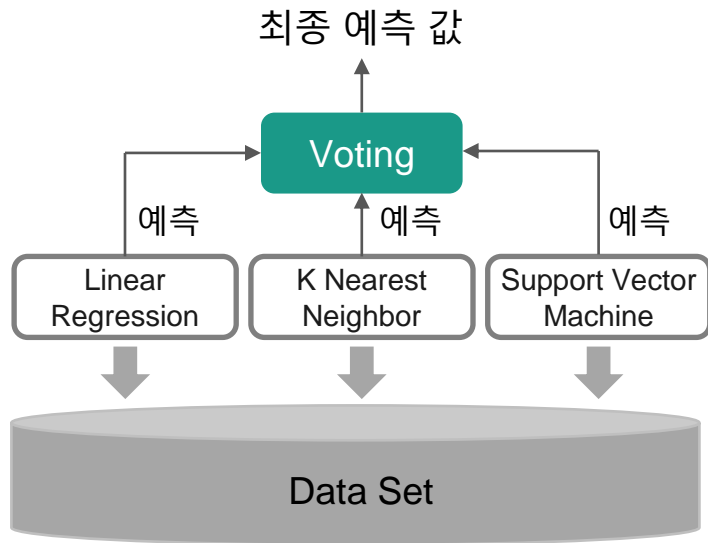


배깅은 각각의 분류기가 모두 같은 유형의 알고리즘 기반이지만, 데이터 샘플링을 서로 다르게 학습 수행하여 보팅 수행

앙상블 학습(Ensemble Learning) 랜덤 포레스트(Random forest)

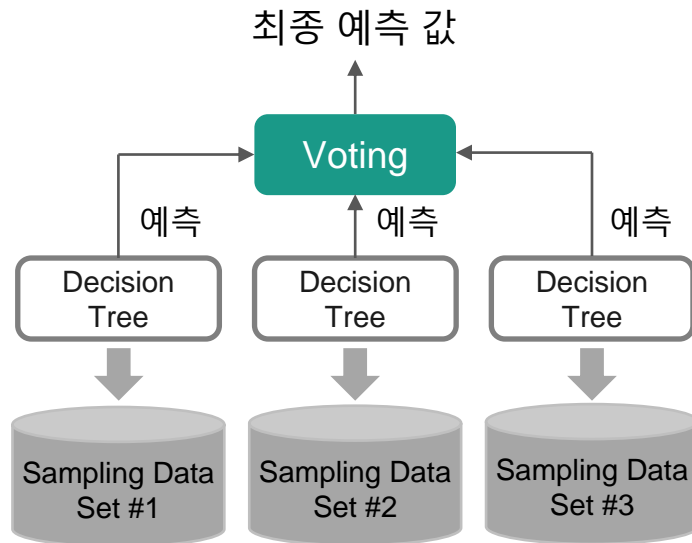
앙상블 학습 - 배깅(Bagging)

보팅(Voting) 방식



보팅은 서로 다른 분류 알고리즘이 같은 데이터 세트를 학습하고 예측한 결과를 보팅을 통해 최종 예측결과를 선정함

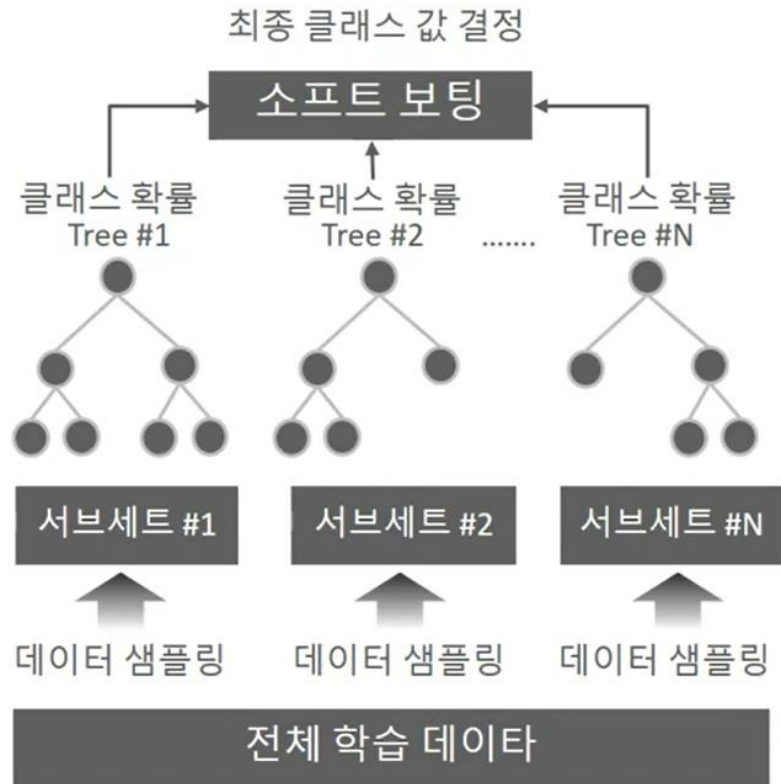
배깅(Bagging) 방식



배깅은 각각의 분류기가 모두 같은 유형의 알고리즘 기반이지만, 데이터 샘플링을 서로 다르게 학습 수행하여 보팅 수행

배깅 - 랜덤 포레스트(Random Forest)

- 다재 다능한 알고리즘
- 앙상블 알고리즘 중 비교적 빠른 수행 속도를 가짐
- 다양한 영역에서 높은 예측 성능을 보이고 있음.
- 여러 개의 결정 트리
분류기가 전체 데이터에서 배깅 방식으로 각자의 데이터를 샘플링해서 개별적으로 학습 후 보팅을 통해 예측

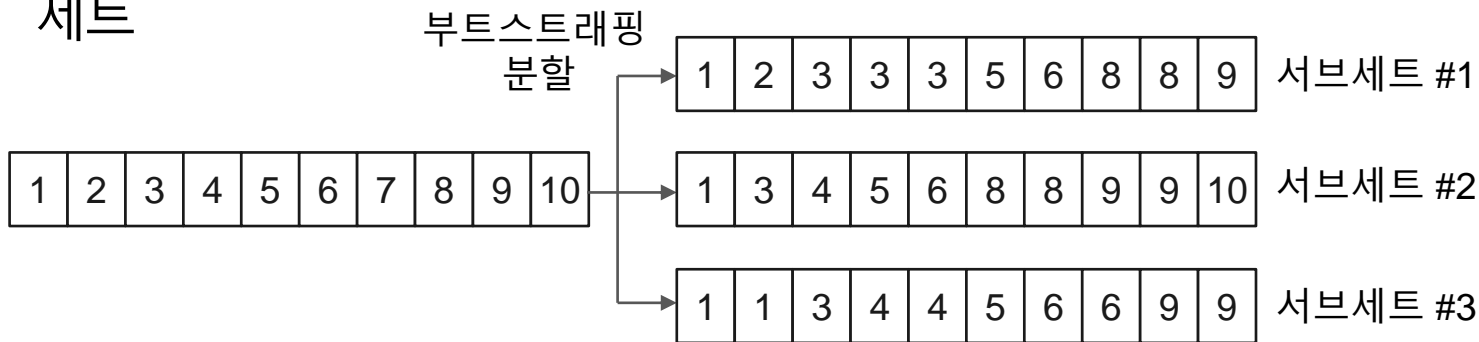


배경 - 랜덤 포레스트(Random Forest)

- 랜덤 포레스트(Random Forest) 의 장점
 - 많은 하이퍼 파라미터 튜닝을 거치지 않아도 일반적으로 안정적으로 좋은 성능을 발휘함
 - 병렬 처리를 이용해서 여러개의 트리를 한번에 학습시킬 수 있음
(`n_jobs = -1`로 지정 시, 시스템의 모든 processor 사용)
- 랜덤 포레스트(Random Forest)의 단점
 - 학습시간이 상대적으로 오래 걸림

랜덤 포레스트 - 부트 스트래핑 분할

- 개별 분류기 : 결정 트리
- 개별 분류기의 학습 데이터 세트
 - * Bagging : bootstrap aggregating
 - 부트스트래핑(bootstrapping) 분할 방식
 - : 전체 데이터에서 일부가 중복되게 샘플링 된 데이터 세트
- 원본 데이터의 건수가 10개인 학습 데이터 세트에 랜덤 포레스트를 3개의 결정 트리기반으로 학습($n_{\text{estimators}}=3$) 할 때의 데이터 서브 세트



Random Forest Estimator



scikit-learning의 랜덤 포레스트 Estimator

- `sklearn.ensemble.RandomForestClassifier` (분류문제에서 사용)
- `sklearn.ensemble.RandomForestRegressor` (회귀문제에서 사용)

랜덤 포레스트 하이퍼 파라미터

RandomForestClassifier Hyper-Parameter

- `n_estimators` : **랜덤 포레스트에서 결정트리의 개수 지정**, 디폴트 10개 많이 설정할 수록 좋은 성능을 기대할 수 있지만, 계속 증가시킨다고 성능이 무조건 향상되는 것은 아님. 늘릴수록 학습 시간이 오래 걸림
- `max_features` : **결정트리의 최대 feature 수**, 디폴트는 `auto(sqrt)`. 랜덤 포레스트의 트리를 분할 하는 피처를 참조할 때 `sqrt`(전체 피처 개수) 만큼 참조(전체 피처가 16개면 4개로 참조)
- `max_depth`, `min_samples_leaf` : 결정트리와 동일하게 과적합을 개선하기 위해 사용

[참고] DataFrame의 replace() 함수 복습

- DataFrame.replace() 함수를 이용해서 컬럼의 값을 원하는 형태로 변경할 수 있음.
- 숫자형 값을 문자형으로, 숫자형 값을 다른 숫자로

```
# No/Yes 값을 0, 1로 변경
df['RainToday'].replace({ 'No': 0, 'Yes': 1 }, inplace=True)
df['RainTomorrow'].replace({ 'No': 0, 'Yes': 1 }, inplace=True)
```

<https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.replace.html>

원-핫 인코딩(One-Hot-Encoding)

- 범주형 값을 이진화 된 값으로 바꿔서 표현하는 것
- Label Encoding 으로
 - 개 = 1, 고양이 = 2, 토끼 = 3 이라고 했을 때,
 - 머신러닝은 $1+2 = 3$, 개 + 고양이 = 토끼라고 학습할 경향성을 띌 수도 있음
- 이 문제를 해결하기 위해 One-Hot-Encoding을 사용함.



Scikit-learning OneHotEncoding

- numpy 행렬을 입력 값으로 해야함.
- 벡터 입력을 허용하지 않음. Reshape을 이용해 Matrix로 변환필요
예제 파일 참고 : one_hot_encoding_(sklearn, pandas)_ex.ipynb

```
from sklearn.preprocessing import OneHotEncoder  
ohe = OneEncoder(sparase=False)
```

```
ohe_data = ohe_data.astype(int)  
print(ohe_data)
```

```
[[1 0 0 0]  
 [0 1 0 0]  
 [1 0 0 0]  
 [0 1 0 0]  
 [0 0 0 1]  
 [0 0 1 0]  
 [0 0 1 0]]
```

sparase=False
결과

```
ohe_data = ohe_data.astype(int)  
print(ohe_data)
```

```
(0, 0)      1  
(1, 1)      1  
(2, 0)      1  
(3, 1)      1  
(4, 3)      1  
(5, 2)      1  
(6, 2)      1
```

sparase=True
결과

원-핫 인코딩(One-Hot-Encoding)

- 피처 값의 유형에 따라 새로운 피처를 추가해 고유 값에 해당하는 컬럼에만 1을 표시하고 나머지 컬럼에는 0을 표시하는 방식

원본

원-핫 인코딩 – 이진코드로 범주 값을 표기함

| 상품분류 | 상품분류_a001 | 상품분류_a002 | 상품분류_b001 | 상품분류_b002 |
|------|-----------|-----------|-----------|-----------|
| a001 | 1 | 0 | 0 | 0 |
| a002 | 0 | 1 | 0 | 0 |
| a001 | 1 | 0 | 0 | 0 |
| a002 | 0 | 1 | 0 | 0 |
| b002 | 0 | 0 | 0 | 1 |
| b001 | 0 | 0 | 1 | 0 |
| b001 | 0 | 0 | 1 | 0 |

Pandas의 원-핫 인코딩

- `get_dummies()` 활용 예)

```
import pandas as pd
```

```
df = pd.DataFrame({'item': ['TV', '냉장고', '전자레인지', '컴퓨터', '선풍기', '믹서', '믹서']})
```

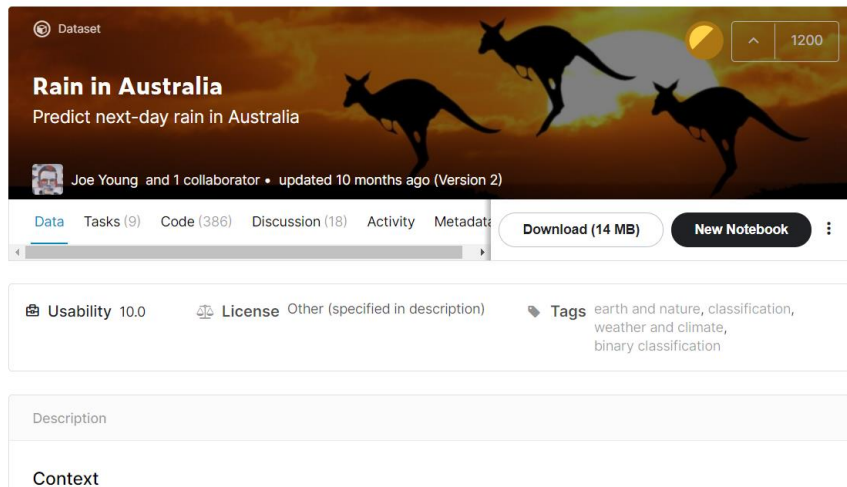
```
pd.get_dummies(df)
```

| | a001 | a002 | b001 | b002 |
|---|------|------|------|------|
| 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 2 | 1 | 0 | 0 | 0 |
| 3 | 0 | 1 | 0 | 0 |
| 4 | 0 | 0 | 0 | 1 |
| 5 | 0 | 0 | 1 | 0 |
| 6 | 0 | 0 | 1 | 0 |

[실습1] 랜덤 포레스트 – 내일 날씨 예측

Rain in Australia 데이터, 내일 날씨 예측하기

- <https://www.kaggle.com/uciml/breast-ca>
- 호주 기상청에서 2010년에 발표한 데이터셋
- 데이터 columns : Date, Location, RainToday, RainTomorrow...
 - 23 개 컬럼
- 데이터 개수 : 145,460



[실습1] 랜덤 포레스트 – 내일 날씨 예측

- Feature 데이터 columns : Date, Location, RainToday ...
 - 22 Features (RainTomorrow 제외)
- Target Value : RainTomorrow 컬럼
 - Yes(내일 비가 옴) or No(내일 비가 오지 않음)
 - Binary Classification
- 사용 알고리즘(Estimator)
 - DecisionTreeClassifier
 - RandomForestClassifier
- 적용기법
 - Data Cleansing(결측치 처리)
 - 정확도 비교 평가

[실습1] 랜덤 포레스트 – 내일 날씨 예측

- Features

| | |
|---------------|------------|
| Date | : 일시 |
| Location | : 지역 |
| MinTemp | : 최저 기온 |
| MaxTemp | : 최고 기온 |
| Rainfall | : 강우량(mm) |
| Evaporation | : 증발량(mm) |
| Sunshine | : 해가 보인 시간 |
| WindGustDir | : 자정까지 풍향 |
| WindGustSpeed | : 자정까지 풍속 |
| WindDir9am | : 아침풍향 |
| WindDir3pm | : 낮풍향 |

| | |
|--------------|---------------|
| WindSpeed9am | : 아침풍속 |
| WindSpeed3pm | : 낮풍속 |
| Humidity9am | : 아침습도 |
| Humidity3pm | : 낮습도 |
| Pressure9am | : 아침대기압 |
| Pressure3pm | : 낮대기압 |
| Cloud9am | : 아침 구름량 _x/8 |
| Cloud3pm | : 낮 구름량 _x/8 |
| Temp9am | : 아침 기온 |
| Temp3pm | : 낮 기온 |
| RainToday | : 당일강수여부 |
| RainTomorrow | : 내일강수여부 |

[실습1] 랜덤 포레스트 - 내일 날씨 예측

- 결측치가 많은 데이터

내일 비가 올지 예측해보기

- 파일 : 3-5-1_

Random_Forest_내일_비가올지_예측해보기
(Rain in Australia).ipynb



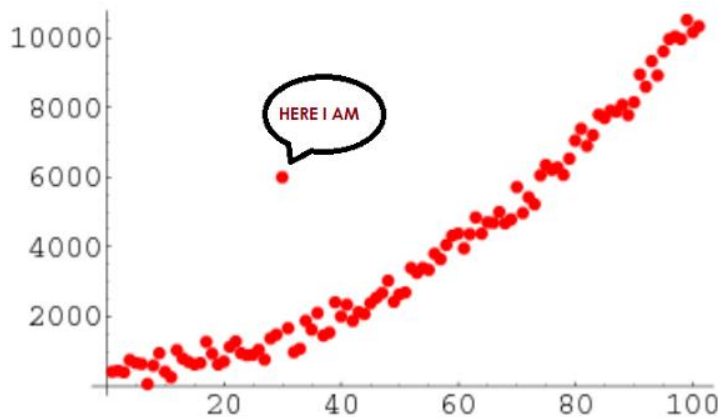
df.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 145460 entries, 0 to 145459
Data columns (total 23 columns):
#   Column              Non-Null Count  Dtype
---  -
0   Date                145460 non-null  object
1   Location            145460 non-null  object
2   MinTemp             143975 non-null  float64
3   MaxTemp             144199 non-null  float64
4   Rainfall            142199 non-null  float64
5   Evaporation         82670 non-null   float64
6   Sunshine            75625 non-null   float64
7   WindGustDir         135134 non-null  object
8   WindGustSpeed       135197 non-null  float64
9   WindDir9am          134894 non-null  object
10  WindDir3pm          141232 non-null  object
11  WindSpeed9am        143693 non-null  float64
12  WindSpeed3pm        142398 non-null  float64
13  Humidity9am         142806 non-null  float64
14  Humidity3pm         140953 non-null  float64
15  Pressure9am         130395 non-null  float64
16  Pressure3pm         130432 non-null  float64
17  Cloud9am            89572 non-null   float64
18  Cloud3pm            86102 non-null   float64
19  Temp9am             143693 non-null  float64
20  Temp3pm             141851 non-null  float64
21  RainToday           142199 non-null  object
22  RainTomorrow        142193 non-null  object
dtypes: float64(16), object(7)
memory usage: 25.5+ MB
```

이상치(Outlier) 제거

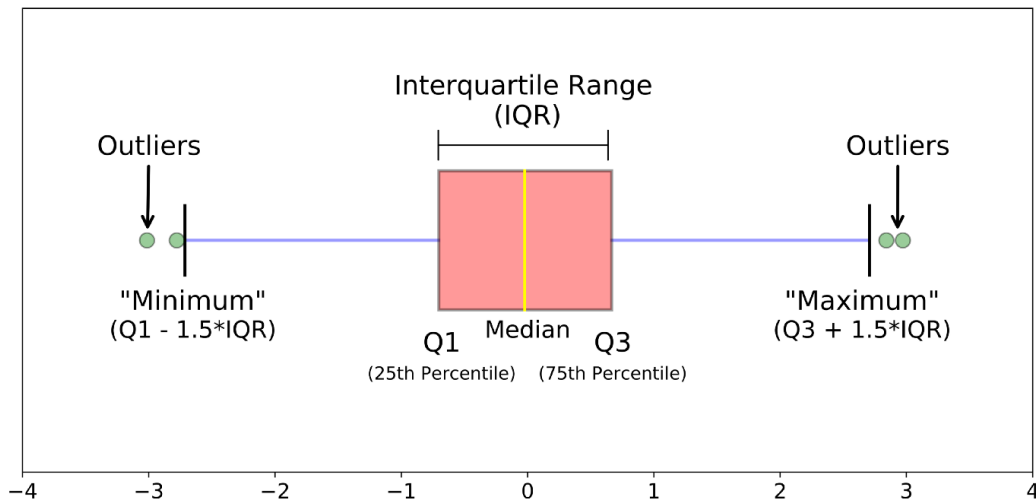
이상치 제거(Remove Outlier)

- 이상치는 다른 데이터와 **크게 다른 값을** 갖는 데이터를 의미함
- 이상치는 오히려 **머신러닝 모델의 학습에 방해** 됨
- 이상치 제거(Remove Outlier)는 **머신러닝 모델의 성능향상에 도움**이 됨



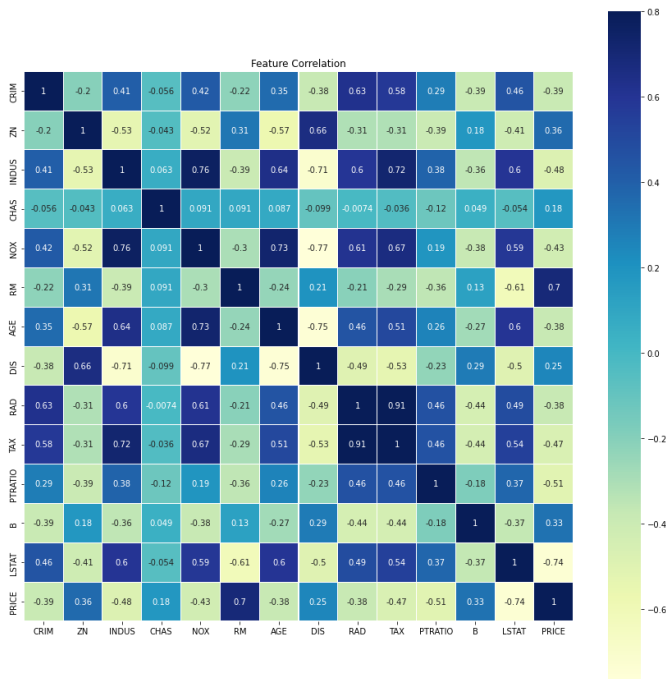
이상치 제거 – IQR(Inter Quantile Range) 활용

- IQR(Inter Quantile Range) 방법을 이용해 이상치(Outlier) 제거
- 사분위 값을 이용하여 데이터의 분포 모양, 대칭성, 극단 값을 쉽게 파악할 수 있음
- seaborn의 `boxplot()` 활용



이상치 제거 – 상관분석(Correlation analysis) 활용

- 상관분석(Correlation analysis)을 활용한 이상치 제거 컬럼 선택
- target과 연관관계가 높은 feature의 이상치 제거



1에 가까운 값 : 두 변수들 간의 양의 상관관계가 있음.

0에 가까운 값 : 두 변수들 간의 상관관계가 없음.

-1에 가까운 값 : 두 변수들 간의 음의 상관관계가 있음.

이상치 제거 – IQR 방식 활용

- IQR 방식으로 이상치 제거 구현하기

```
def get_outlier_indices(data, columns):  
    outlier_indices = []  
    for column in columns:  
        Q1 = data[column].quantile(0.25)  
        Q3 = data[column].quantile(0.75)  
        IQR = Q3 - Q1  
        min_value = Q1 - 1.5*IQR  
        max_value = Q3 + 1.5*IQR  
        filter = ((data[column] < min_value) | (data[column] > max_value))  
  
        outlier_data = data[column][filter]  
        outlier_index = outlier_data.index  
        outlier_indices.extend(outlier_index)  
    return outlier_indices
```

```
print('outlier 삭제하기 전의 데이터 프레임 : ', df.shape)  
delete_indices = get_outlier_indices(df, ['Humidity9am', 'Pressure9am', 'Pressure3pm', 'WindGustSpeed'])  
df = df.drop(delete_indices, axis=0).reset_index(drop=True)  
print('dataframe after removing outlier (etc) : ', df.shape)
```

[실습2] Outlier 제거 적용 Random Forest Classifier

Rain in Australia에 이상치 제거를 적용해서 성능 향상시키기

- 데이터 : <https://www.kaggle.com/uciml/breast-ca>
- 호주 기상청에서 2010년에 발표한 데이터셋
- 데이터 columns : Date, Location, RainToday, RainTomorrow...
 - 23 Dimension
- 데이터 개수 : 145,460

앙상블 학습(Ensemble Learning)

XGBoost(eXtreme Gradient Boosting)

Kaggle 우승자들이 애용하는 알고리즘

XGBoost 설치(on ubuntu 18.04)

현재 실습하고 있는 가상환경에 설치하기

```
(MLvenv)$ pip3.8 install xgboost
```

Tip : lightGBM 설치

```
$ pip3.8 install lightgbm
```

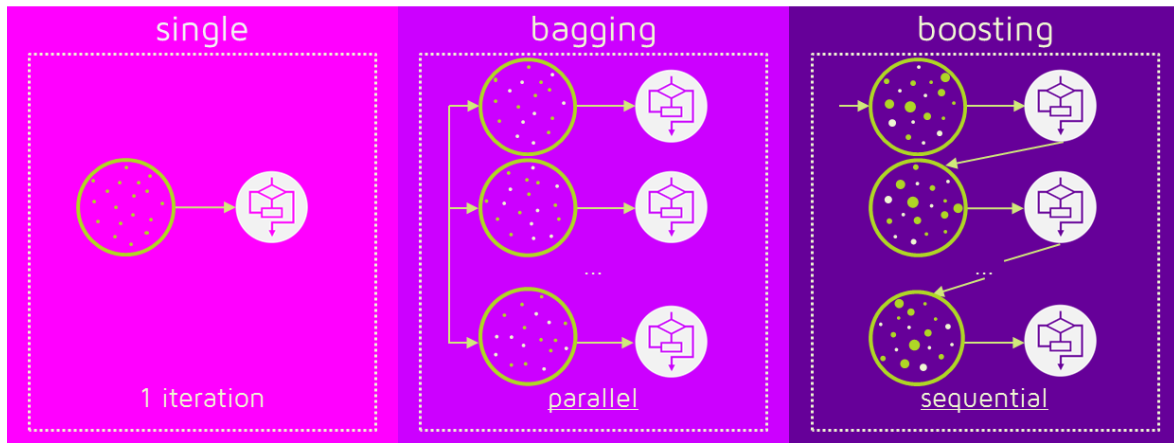

XGBoost(eXtreme Gradient Boosting)

- 공식문서: <https://xgboost.readthedocs.io/en/latest/index.html>
- scikit-learning 외부 라이브러리
- XGBoost는 **병렬적이고, 효율적이고 최적화된** GBM 알고리즘
- XGBoost는 **Grandient Boosting** 알고리즘에 기반함.

dmlc
XGBoost

부스팅 – GBM(Gradient Boost Machine)

- Bagging 방식 : 매번 랜덤하게 샘플을 뽑아서 독립적으로 학습시킨 분류기들의 결과를 종합해서 앙상블 러닝 수행 - 랜덤 포레스트
- **Boosting** 방식 : 매번 샘플을 뽑아서 학습시키되, 순차적으로 오차가 큰 샘플들이 뽑힐 확률이 높아지도록 가중치를 부여하고, 다음 단계에 샘플을 뽑아서 학습시키는 알고리즘



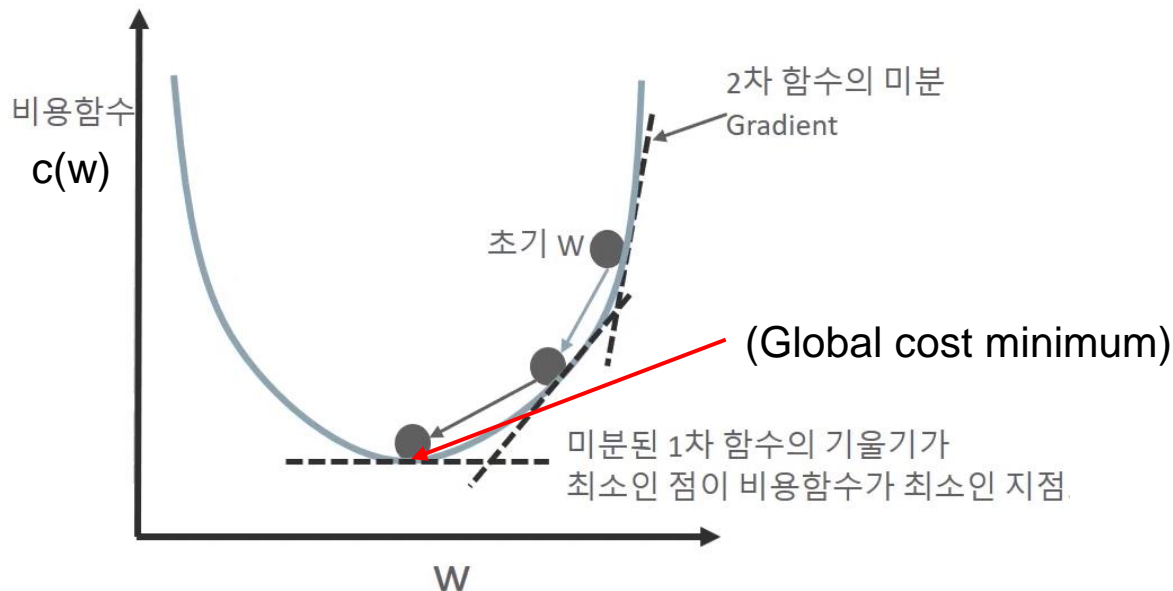
앙상블 학습 - 부스팅(Boosting)

- 여러 개의 약한 학습기(weak learner)를 순차적으로 학습-예측하면서 잘못 예측한 데이터에 가중치 부여를 통해 오류를 개선해 가면서 학습하는 방식
- AdaBoost(Adaptive boosting)와 GBM(Gradient Boosting Machine), **XGBoost**, LightGBM

부스팅(Boosting) - GBM

GBM(Gradient Boosting Machine) 알고리즘

- 학습과정에서 파라미터를 최적화하는데 Gradient Descent 알고리즘 사용
- 오차가 큰 샘플들이 많이 뽑히도록 할 때 Gradient Descent 알고리즘 사용



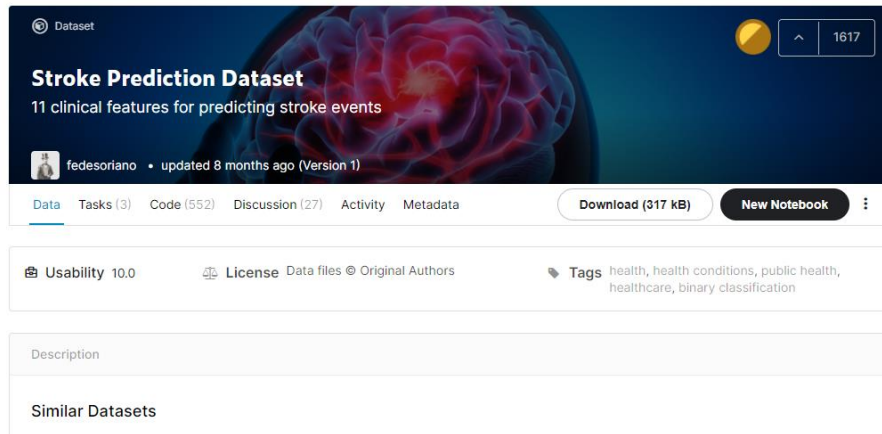
XGBoost의 장점과 단점



- XGBoost의 장점
 - 대부분의 상황에서 안정적으로 좋은 성능을 발휘함.
 - Feature Engineering을 많이 적용하지 않아도 안정적인 성능을 보여줌
- XGBoost의 단점
 - 하이퍼 파라미터가 방대해서 튜닝하는 것이 상대적으로 어려움.

[실습3]Stroke Prediction Dataset

- 데이터 : <https://www.Kaggle.com/fedesoriano/stroke-prediction-dataset>
- 나이, 성별, 고혈압 유무 등을 토대로 뇌졸중을 가진 사람인지 아닌지를 예측해 볼 수 있는 데이터셋
- Feature columns : id, gender, age, hypertension, ... , stroke
 - 12개 컬럼
 - Target Value : stroke(뇌졸중 환자) or not stroke(뇌졸중 아님)
 - Binary classification
- 데이터 개수 : 5,110개



[실습3]Stroke Prediction Dataset

- 뇌졸중(腦卒中, 영어: stroke, 문화어: **뇌졸증**, 腦卒症)
- 뇌혈류 이상으로 인해 갑작스레 유발된 국소적인 신경학적 결손 증상을 통칭하는 말이다.
- **증상**: 편측 마비, 수용성언어상실증, 표현언어상...
- **다른 이름**: 뇌중풍(Cerebrovascular accident)
- **진단 방식**: 증상 및 의학 영상에 기반
- **병인**: 뇌 허혈, 두개내출혈

[실습3]Stroke Prediction Dataset 특징

- id : 구분 id
- gender : "Male", "Female", "Other"
- age : 환자의 나이
- hypertension : 고혈압 환자 1, 고혈압 환자 아니면 0
- heart_disease : 심장병 환자 1, 심장병 환자 아니면 0
- ever_married : "No", "Yes"
- work_type : "children", "Govt_jov", "Never_worked", "Private" or "Self-employed"
- Residence_type : "Rural" or "Urban"
- avg_glucose_level : 평균 포도당 수치
- bmi : body mass index(체질량지수: 체중/키²=체지방양 추정)
- smoking_status : "formerly smoked", "neversmoked", "smoked", "unknown"
- stroke : 뇌졸중 환자 1, 뇌졸중 환자 아니면 0

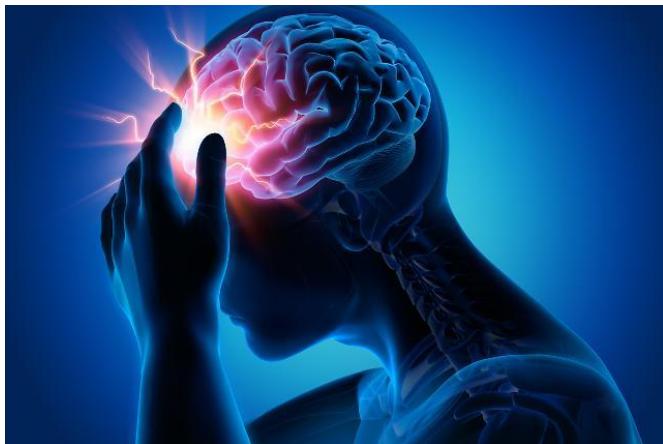
[실습3]Stroke Prediction – 예측모델 셋팅

- 입력 데이터
 - id, gender, age, hypertension, ..., smoking_status
 - 11 features
- Target data
 - stroke 컬럼 : 1(뇌졸중 환자), 0(뇌졸중 환자 아님)
 - binary Classification
- 사용 알고리즘(Estimator)
 - DecisionTreeClassifier
 - RandomforestClassifier
 - XGBoostClassifier
- 추가적인 적용 기법
 - Data Cleansing(결측치 처리)

[실습3]Stroke Prediction

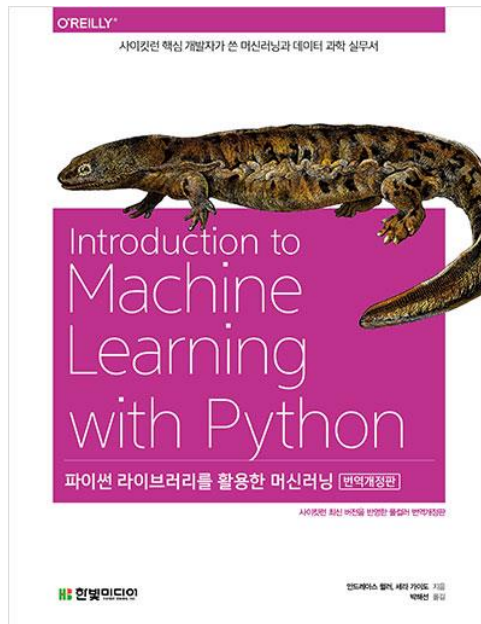
- XGBoost 알고리즘 적용 뇌졸중 환자인지 예측해 보기

실습파일 : 3-5-3_XGBoost_stroke_patient_pred.ipynb



교재 소스

https://github.com/rickiepark/introduction_to_ml_with_python



실전으로 더 깊이 공부 하고 싶다면

- 파이썬 머신러닝 완벽 가이드, 위키북스
- 핸즈온 머신러닝 – 한빛 미디어

