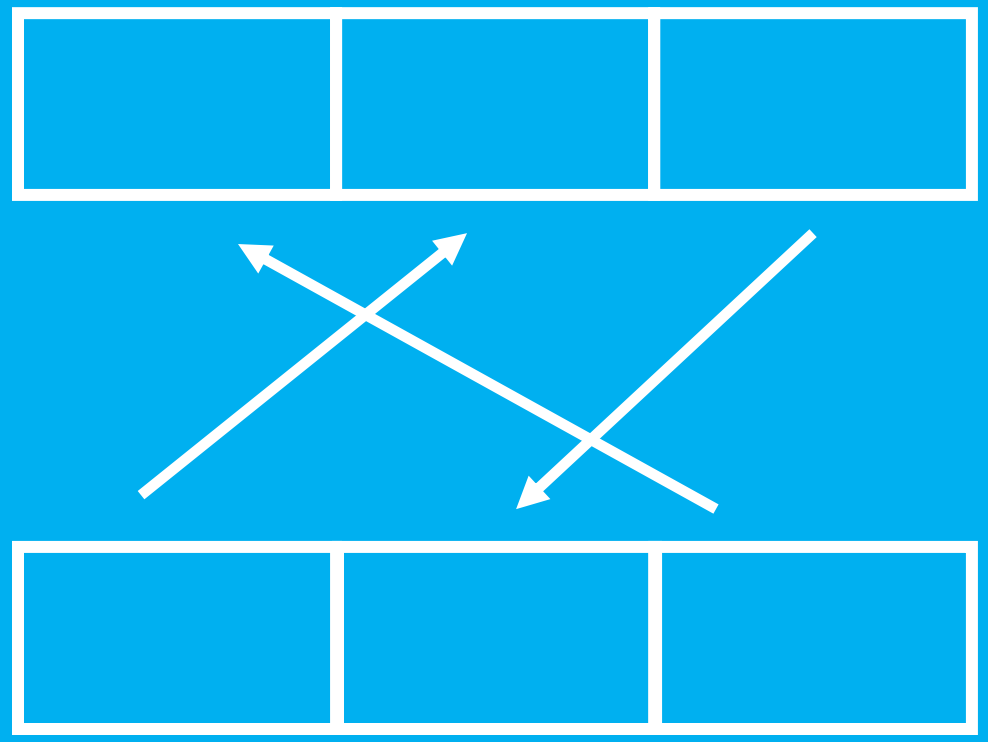
9주차 정규세션

교차 검증과 그리드 서치



진도 세션 개요

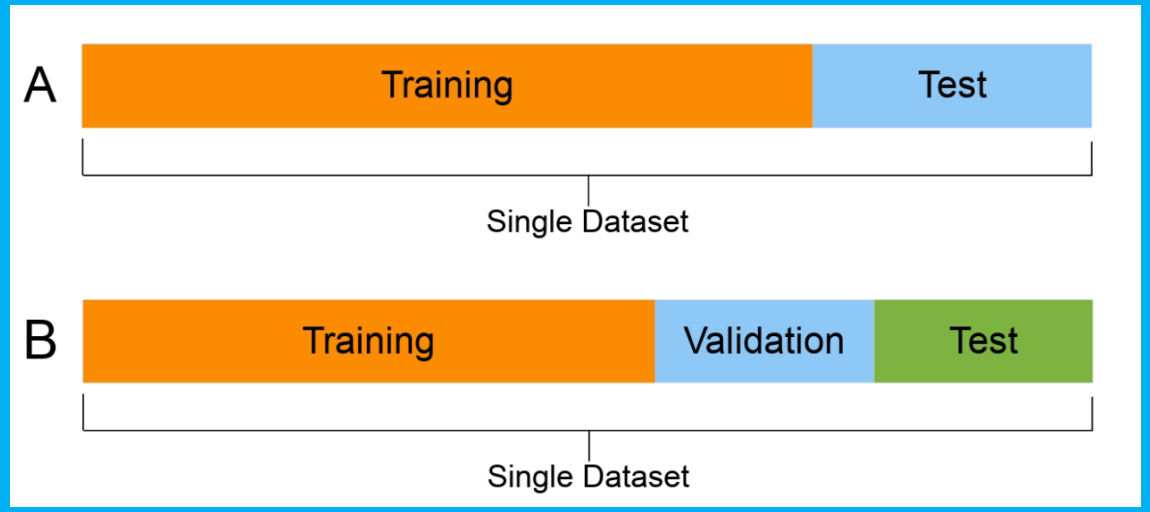
교차 검증

-검증 세트
-K-Fold 교차검증

그리드 서치

-그리드 서치
-랜덤 서치, 베이시안 서치

교차 검증



-검증 세트(Validation Set)

머신러닝 모델을 검증하기 위한 데이터셋

Test Set로 성능을 평가해서 변수들을 조절하면 결국 Test Set에 적합한 모델이 만들어지게 된다

이를 방지하기 위해 검증 세트를 따로 만들어 일반화 성능을 유지
주로 Training Set에서 따로 분리해서 이용

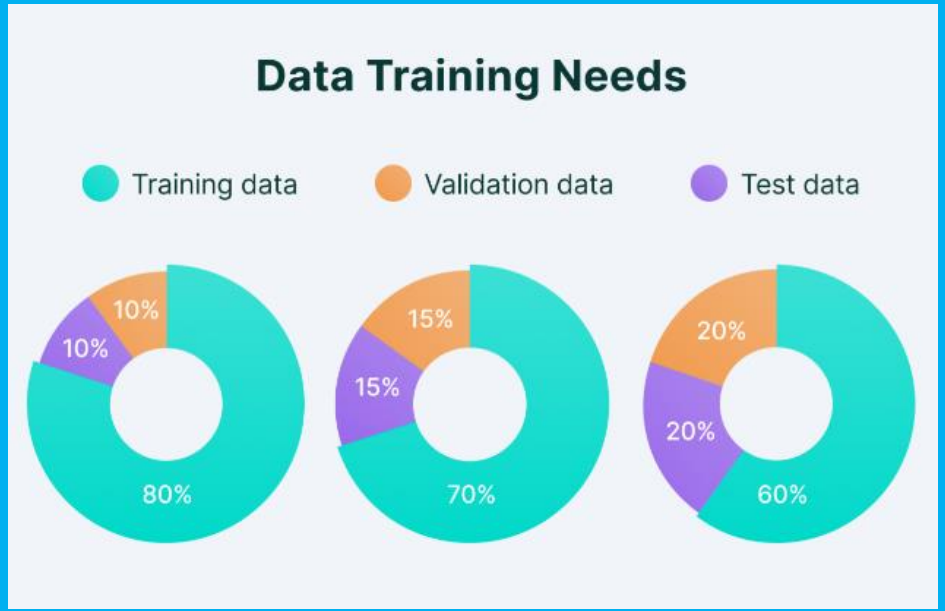
교차 검증

-검증 세트(Validation Set)

검증 세트를 이용해서 검증을 마친 모델은 최종적으로 Test Set으로 평가를 하게 된다

장점: Test Set에 적합한 모델이 안되도록, 일반화된 모델이 될 수 있게끔 도와준다

단점: 전체 데이터양이 적은 경우 변수값이 달라지면 성능 평가 점수가 크게 달라진다



교차 검증

-검증 세트(Validation Set)

검증세트 만들기

```
import pandas as pd
wine = pd.read_csv('https://bit.ly/wine_csv_data')
```

```
data = wine[['alcohol', 'sugar', 'pH']].to_numpy()
target = wine['class'].to_numpy()
```

```
from sklearn.model_selection import train_test_split
train_input, test_input, train_target, test_target = train_test_split(data, target, test_size = 0.2, random_state = 42)
```

```
sub_input, val_input, sub_target, val_target = train_test_split(train_input, train_target, test_size = 0.2, random_state = 42)
```

```
print(data.shape, train_input.shape, sub_input.shape)
```

```
(6497, 3) (5197, 3) (4157, 3)
```

```
print(val_input.shape)
```

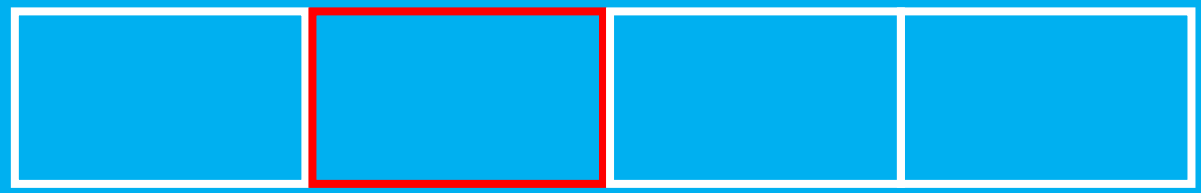
```
(1040, 3)
```

교차 검증

-교차 검증

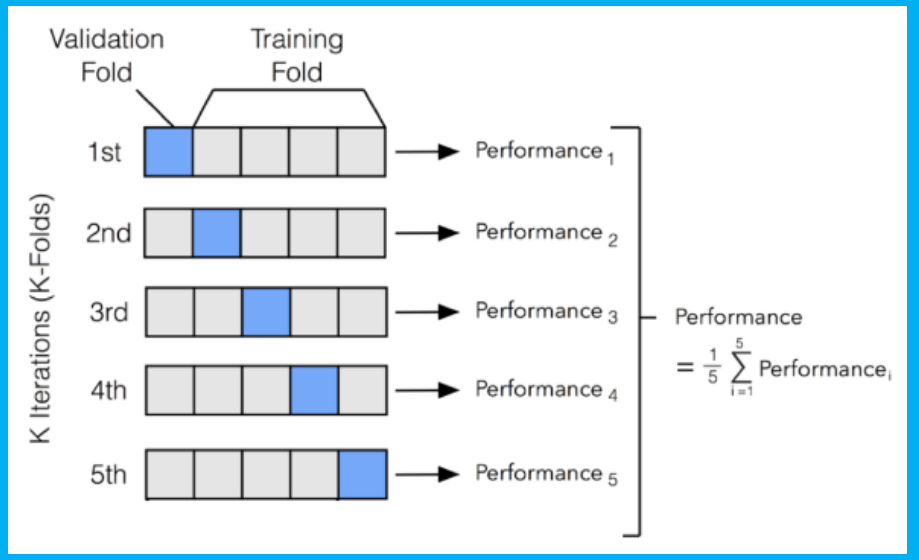
검증 세트의 데이터양이 너무 적으면 검증을 진행 할 때
점수의 차이가 많이 난다

교차 검증을 통해서 데이터의 다양한 부분들을 검증 세트로
이용하고 평균을 내 안정적인 점수를 도출한다



교차 검증

-K-Fold Cross Validation



K-Fold가 보편적으로 사용되는 검증 방법
- 전체 데이터셋을 Train Set & Test Set으로 나눈다
- Train Set를 K개의 Fold로 나눈다
- Fold들을 Validation Set로 사용하고 나머지는 Train Set
- 모든 Fold들에 대해 똑같이 진행하고 평균 내어 점수 확인

교차 검증

-K-Fold Cross Validation

사이킷런을 이용해 교차검증 실행

```
from sklearn.model_selection import cross_validate
scores = cross_validate(dt, train_input, train_target)
print(scores)
```

```
{'fit_time': array([0.01230454, 0.00722599, 0.00739336, 0.00730777, 0.00713015]),
'score_time': array([0.00107765, 0.00074458, 0.00073695, 0.00072265, 0.00156212]),
'test_score': array([0.86923077, 0.84615385, 0.87680462, 0.84889317, 0.83541867])}
```

```
import numpy as np
print(np.mean(scores['test_score']))
```

```
0.855300214703487
```

교차 검증

-K-Fold Cross Validation

K-Fold 분할 수 설정하기
StratifiedKFold 의 n_splits 값을 지정하여 분할 수를 바꿀 수 있다

```
from sklearn.model_selection import StratifiedKFold
scores = cross_validate(dt, train_input, train_target, cv = StratifiedKFold())
print(np.mean(scores['test_score']))
```

```
0.855300214703487
```

```
splitter = StratifiedKFold(n_splits=10, shuffle = True, random_state = 42)
scores = cross_validate(dt, train_input, train_target, cv = splitter)
print(np.mean(scores['test_score']))
```

```
0.8574181117533719
```

교차 검증

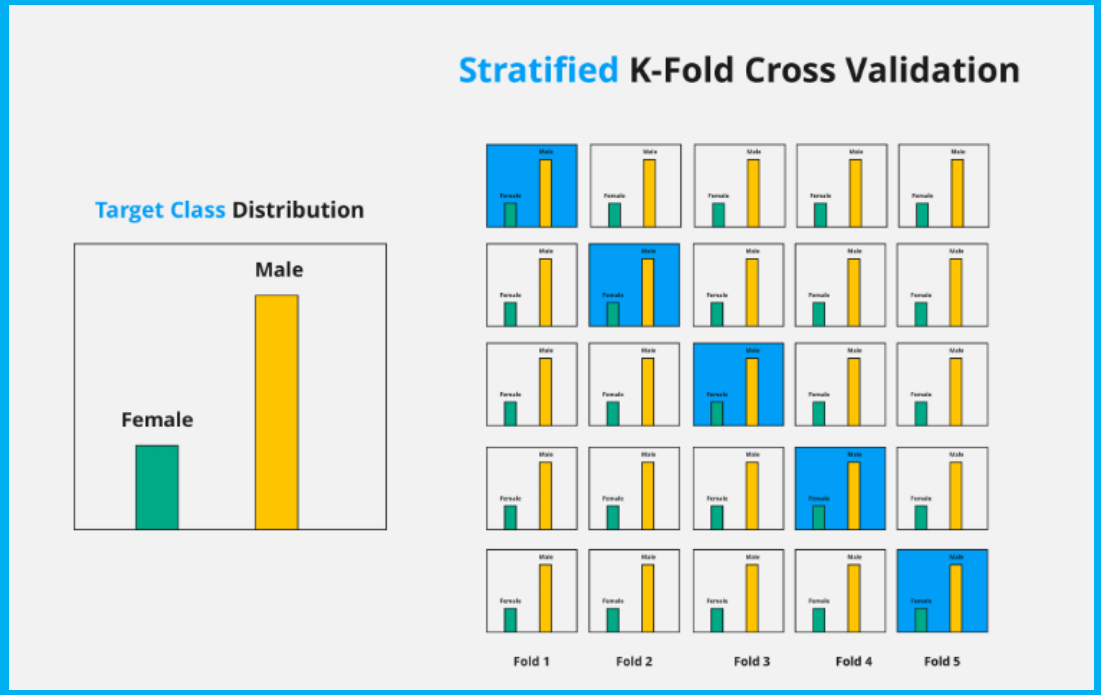
-K-Fold Cross Validation 단점

모델 훈련 및 평가 소요시간 증가
분할 수가 많아질수록 데이터가 일반화되지 못함
다른 Fold에 같은 데이터가 존재하면 성능 악화
순서가 고려된 데이터가 섞이지 않은 경우

교차 검증

-Stratified K-Fold Cross Validation

데이터 클래스 별 분포를 고려하여 나눠서 K-Fold 진행
데이터가 편향되어 있는 경우를 해소
주로 분류 모델에 사용

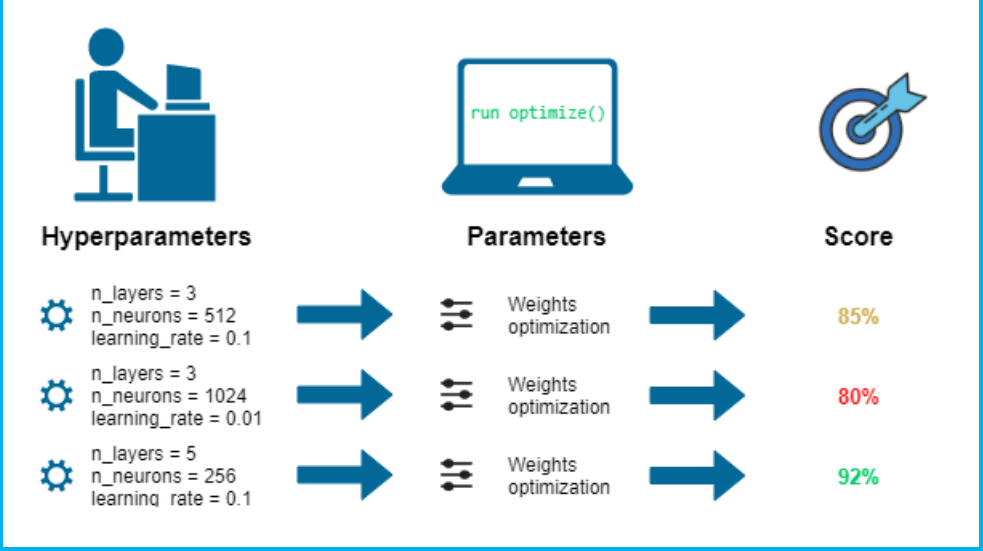


그리드 서치

- 하이퍼 파라미터

하이퍼 파라미터 = 학습 전에 직접 인간이 지정하는 파라미터

Ex) Lasso 모델의 alpha값과 max_iter값



그리드 서치

- 효율적인 파라미터 조정

하이퍼 파라미터 튜닝은 매개변수들을 조정해가며 진행
그러나 변수의 종류가 몹시 많으면 전부 비교해보기 힘들다

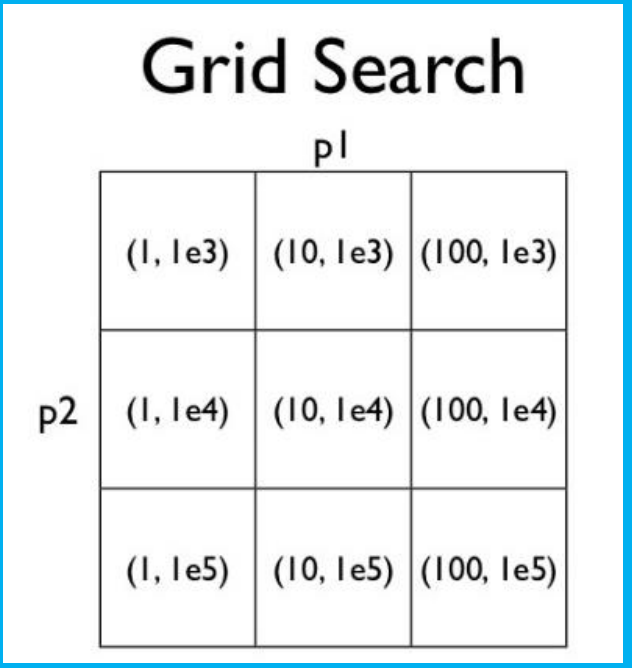
이를 해결하기 위해서 그리드 서치를 사용

그리드 서치

- 그리드 서치

Grid Search(격자 검색)으로 모델에 입력 할 하이퍼 파라미터들을 순서에 따라 입력한 후 최고 성능을 내는 하이퍼 파라미터를 도출

순서에 따라 모든 값들을 넣고 비교하는 만큼 훈련에 소요되는 시간이 많이 든다는 것이 단점



그리드 서치

- 그리드 서치 만들어보기

```
from sklearn.model_selection import GridSearchCV
params = {'min_impurity_decrease' : [0.0001, 0.0002, 0.0003, 0.0004, 0.0005]}

gs = GridSearchCV(DecisionTreeClassifier(random_state=42), params, n_jobs= -1)

gs.fit(train_input, train_target)
```

```
GridSearchCV(estimator=DecisionTreeClassifier(random_state=42), n_jobs=-1,
             param_grid={'min_impurity_decrease': [0.0001, 0.0002, 0.0003,
                                                   0.0004, 0.0005]})
```

※ n_jobs = -1 병렬 실행에 사용할 CPU 코어 수 지정(-1은 전부 사용)

그리드 서치

- 그리드 서치 만들어보기

```
dt = gs.best_estimator_
```

```
print(dt.score(train_input,train_target))
```

```
0.9615162593804117
```

```
print(gs.best_params_)
```

```
{'min_impurity_decrease': 0.0001}
```

```
print(gs.cv_results_['mean_test_score'])
```

```
[0.86819297 0.86453617 0.86492226 0.86780891 0.86761605]
```

```
best_index = np.argmax(gs.cv_results_['mean_test_score'])
print(gs.cv_results_['params'][best_index])
```

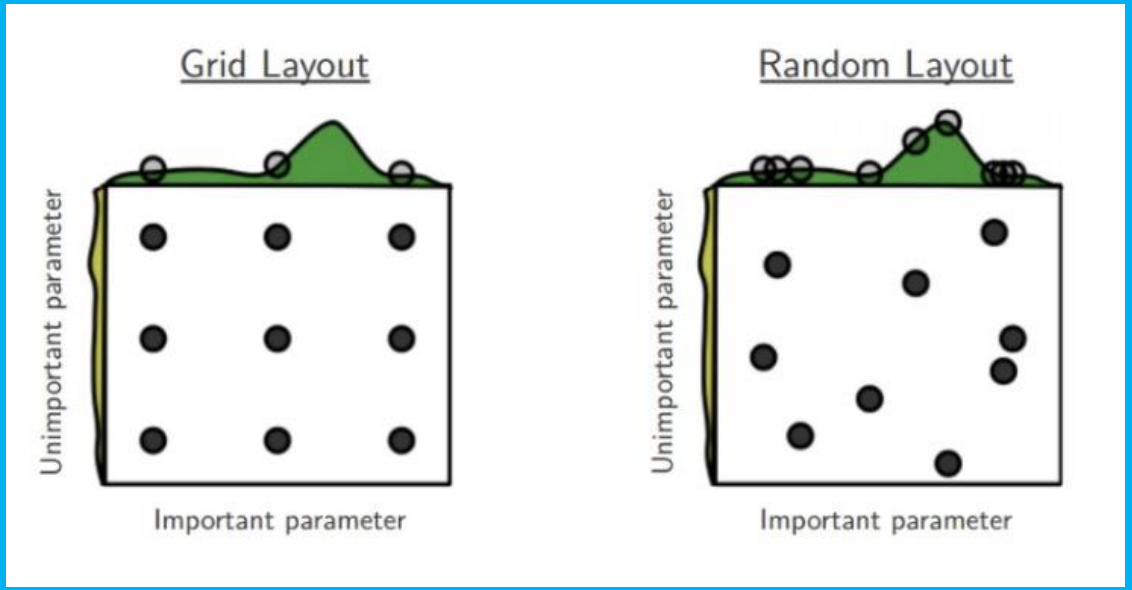
```
{'min_impurity_decrease': 0.0001}
```

그리드 서치

- 그 외에 유사한 방법

Random Search:
하이퍼 파라미터 값을 랜덤하게 대입하고 최고 성능을 내놓은 모델을 생성

불필요한 탐색 횟수를 줄여 소요시간을 단축



그리드 서치

- Random Search 만들어 보기

Randint : 정수형 Uniform: 실수형

```
from scipy.stats import uniform, randint
```

```
rgen = randint(0, 10)
rgen.rvs(10)
```

```
array([7, 3, 2, 3, 6, 2, 0, 9, 7, 6])
```

```
np.unique(rgen.rvs(1000), return_counts=True)
```

```
(array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]),
 array([ 88, 102,  93, 113, 112,  86,  84, 127,  97,  98]))
```

```
ugen = uniform(0, 1)
ugen.rvs(10)
```

```
array([0.023086  , 0.6474147 , 0.69557191, 0.76014318, 0.7604146 ,
       0.21343706, 0.18987216, 0.13216411, 0.19135411, 0.69311863])
```

그리드 서치

- Random Search 만들어 보기

파라미터 지정

Min_samples_split : 리프가 되기 위한 최소 샘플 수

```
params = {'min_impurity_decrease': uniform(0.0001, 0.001),
          'max_depth': randint(20, 50),
          'min_samples_split': randint(2, 25),
          'min_samples_leaf': randint(1, 25),
          }
```

그리드 서치

- Random Search 만들어 보기

RandomizedSearchCV 이용해서 모델 학습

```
from sklearn.model_selection import RandomizedSearchCV

gs = RandomizedSearchCV(DecisionTreeClassifier(random_state=42), params,
                        n_iter=100, n_jobs=-1, random_state=42)
gs.fit(train_input, train_target)
```

```
RandomizedSearchCV(estimator=DecisionTreeClassifier(random_state=42),
                   n_iter=100, n_jobs=-1,
                   param_distributions={'max_depth': <scipy.stats._distn_infrastructure.rv_frozen object at
0x7fcaeaa45110>,
                                        'min_impurity_decrease': <scipy.stats._distn_infrastructure.rv_frozen object at
0x7fcaeaa45410>,
                                        'min_samples_leaf': <scipy.stats._distn_infrastructure.rv_frozen object at
0x7fcaeabbec10>,
                                        'min_samples_split': <scipy.stats._distn_infrastructure.rv_frozen object at
0x7fcaeaa45c50>},
                   random_state=42)
```

그리드 서치

- Random Search 만들어 보기

최상의 파라미터 도출

```
print(gs.best_params_)
```

```
{'max_depth': 39, 'min_impurity_decrease': 0.00034102546602601173, 'min_samples_leaf': 7, 'min_samples_split': 13}
```

```
print(np.max(gs.cv_results_['mean_test_score']))
```

```
0.8695428296438884
```

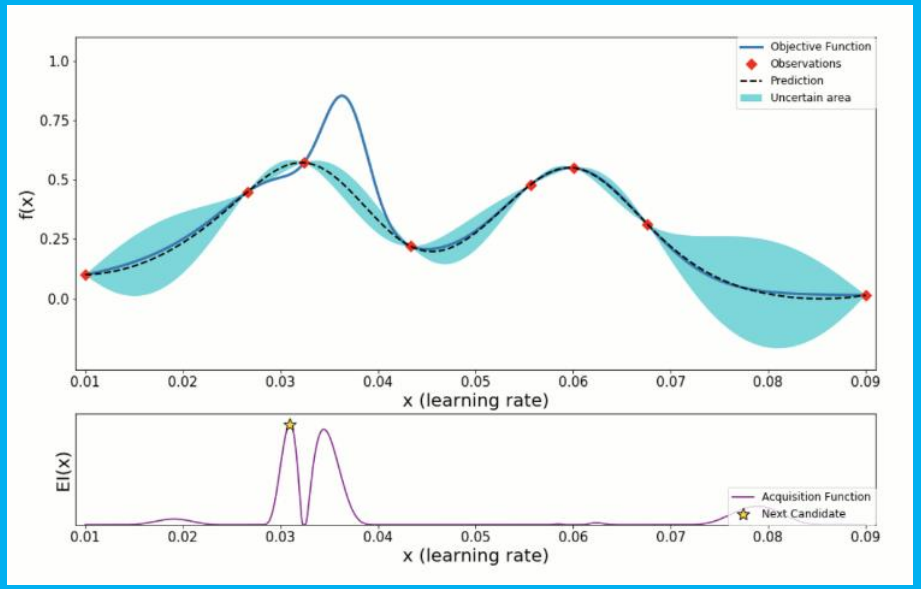
```
dt = gs.best_estimator_

print(dt.score(test_input, test_target))
```

```
0.86
```

그리드 서치

- 그 외에 유사한 방법

Bayesian Optimization(베이즈 최적화):



미지수 x를 받는 미지의 목적함수 f(x)를 상정하여 f(x)를 최대로 만드는 해를 찾아가는 것

순차적으로 하이퍼 파라미터들을 조정해가며 평가를 통해 실시간으로 최적의 하이퍼 파라미터 조합을 탐색