

13

랜덤포레스트(분류)

[랜덤포레스트 - 분류]

랜덤포레스트 회귀분석과 설정해야 하는 매개변수는 같습니다.

00:00

jupyter 13-6_랜덤포레스트(분류) (autosaved)

File Edit View Insert Cell Kernel Navigate Widgets Help Trusted Python 3 (ipykernel)

분류의 경우

```
sklearn.ensemble.RandomForestClassifier(...)
```

- `n_estimators`: RandomForest에서 나무의 수 (int, default = 100)
- `criterion`: 분열의 품질을 측정하는 방법 ("gini", "entropy", default="gini")
- `max_depth`: 나무의 최대 깊이. (int, default = None)
None이면 모든 잎이 순수하거나 잎에 `min_sample_split` 미만의 샘플만 존재할 때까지 노드 확장
- `min_samples_split`: 내부 노드를 분할하는 데 필요한 최소 샘플 수 (int or float, default=2)
- `min_samples_leaf`: 리프 노드에 있어야 하는 최소 샘플 수 (int or float, default=1)
- `max_leaf_nodes`: 리프 노드의 최대 개수 (int, default = None)
None이면 리프 노드 수에 제한이 없음
- `bootstrap`: 나무를 만들 때 부트 스트랩 샘플이 사용되는지 여부 (bool, default=True)
False이면 전체 데이터 세트가 각 트리를 작성하는 데 사용
- `oob_score`: 정확도 측정을 위해 out-of-bag 샘플을 사용할지 여부 (bool, default=False)

```
In [9]: import pandas as pd
from sklearn.model_selection import train_test_split

credit = pd.read_csv("../data/credit_final.csv")
X = credit[credit.columns.difference(['credit.rating'])]
y = credit['credit.rating']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=321)
X_train.head()
```

```
Out[9]: account.balance age apartment.type bank.credits credit.amount credit.duration.months credit.purpose current.assets dependents employment.duration
```

[랜덤포레스트 - 그리드서치]

이번 예제에서는 그리드 서치를 통해 최적의 모델을 찾아보겠습니다.

03:05

jupyter 13-6_랜덤포레스트(분류) (autosaved)

File Edit View Insert Cell Kernel Navigate Widgets Help Trusted Python 3 (ipykernel)

RandomForestClassifier (AUC = 0.80)

```
In [5]: clf.get_params()
```

```
Out[5]: {'bootstrap': True,
'ccp_alpha': 0.0,
'class_weight': None,
'criterion': 'gini',
'max_depth': None,
'max_features': 'auto',
'max_leaf_nodes': None,
'max_samples': None,
'min_impurity_decrease': 0.0,
'min_impurity_split': None,
'min_samples_leaf': 1,
'min_samples_split': 2,
'min_weight_fraction_leaf': 0.0,
'n_estimators': 100,
'n_jobs': None,
'oob_score': False,
'random_state': None,
'verbose': 0,
'warm_start': False}
```

```
In [10]: from sklearn.model_selection import GridSearchCV
param_grid = {'max_depth': range(2, 10, 2), 'min_samples_leaf': range(2, 10, 2)}
model_grid_rf = GridSearchCV(RandomForestClassifier(), param_grid, cv=5)
model_grid_rf.fit(X_train, y_train)
```

```
Out[10]: GridSearchCV(cv=5, estimator=RandomForestClassifier(),
param_grid={'max_depth': range(2, 10, 2),
```