Decision Tree

구름
도시공학과 일반대학원
한양대학교

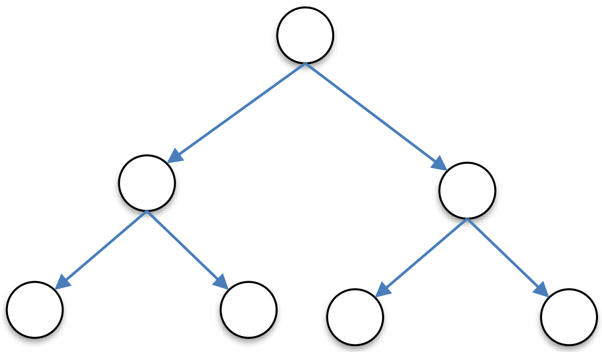
Decision Tree (의사결정 나무)

CHAID Kass, G. V. (1980).
"An exploratory technique for investigating large quantities of categorical data". *Applied Statistics*. **29** (2): 119–127.

CART Breiman, Leo; Friedman, J. H.; Olshen, R. A.; Stone, C. J. (1984).
Classification and regression trees. Monterey, CA: Wadsworth & Brooks/Cole Advanced Books & Software.

ID3 Quinlan, J. R. 1986. Induction of Decision Trees. Mach. Learn. 1, 1 (Mar. 1986), 81–106

C4.5 Quinlan, J. R. C4.5: *Programs for Machine Learning*. Morgan Kaufmann Publishers, 1993.

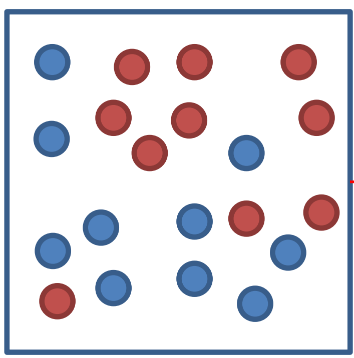


CART : Classification & Regression Tree

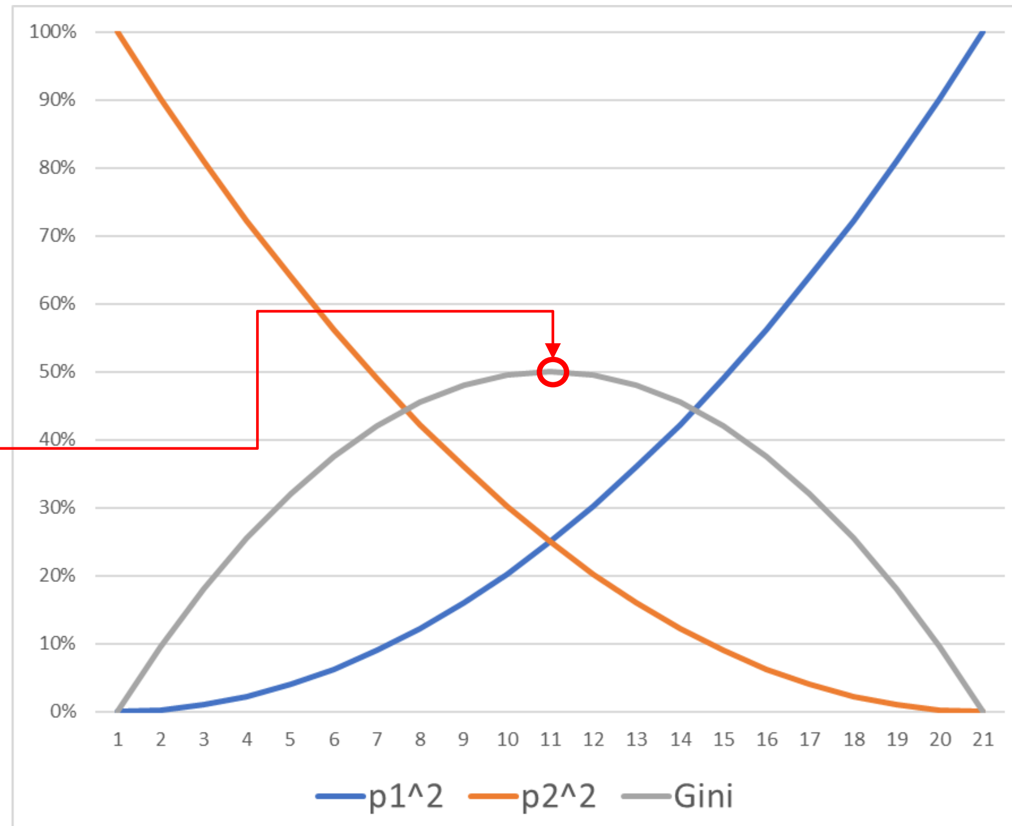
Gini Index(지니인덱스)
Impurity

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

S : 데이터 집합
c : 종속변수 클래스 개수
Pi : 해당 클래스 확률



높은온도(p1)	낮은온도(p2)
-	20
1	19
2	18
3	17
4	16
5	15
6	14
7	13
8	12
9	11
10	10
11	9
12	8
13	7
14	6
15	5
16	4
17	3
18	2
19	1
20	-



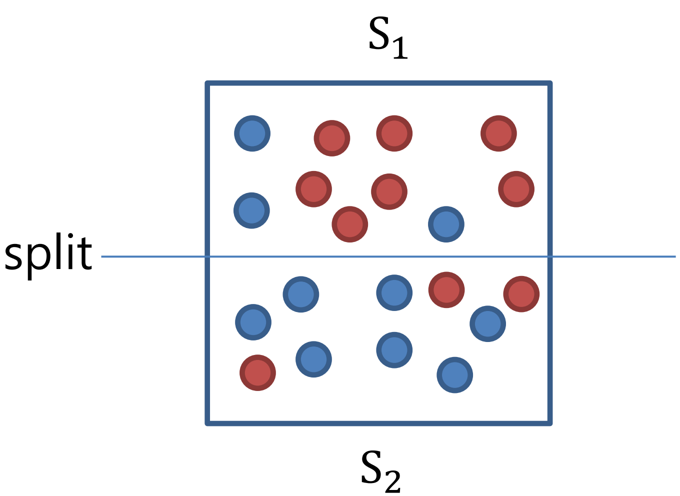
CART : Classification & Regression Tree

Gini Index(지니인덱스)

Impurity

$$I(S) = \sum_{i=0}^{d} R_i \times G(S_i)$$

d : 노드수
Ri : 해당 노드 데이터 비율
Si : 해당 노드 데이터



$$G(S_1) = 1 - \left(\frac{3}{10}\right)^2 - \left(\frac{7}{10}\right)^2 = 0.42$$

$$G(S_2) = 1 - \left(\frac{7}{10}\right)^2 - \left(\frac{3}{10}\right)^2 = 0.42$$

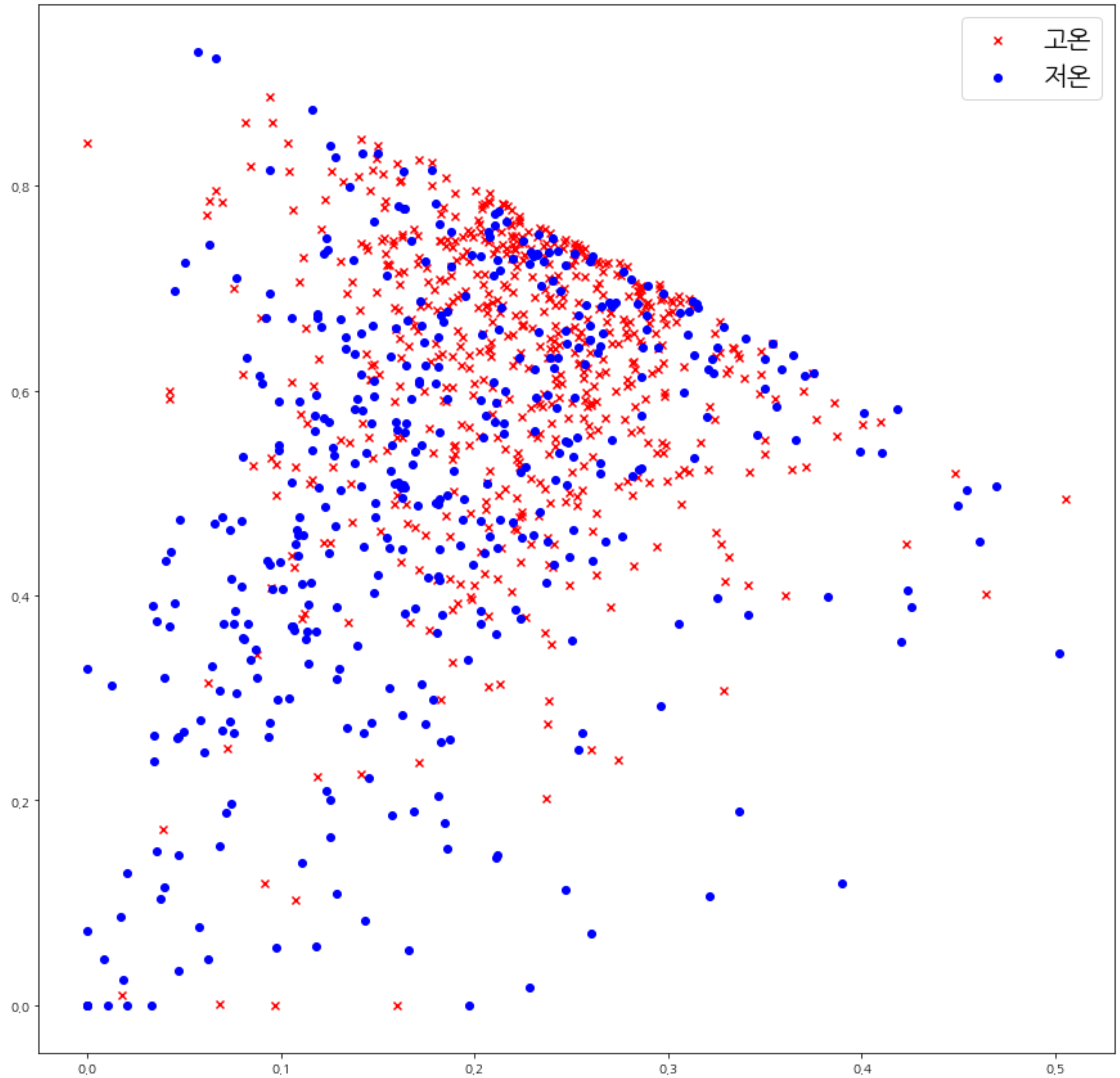
$$I(S) = 0.5 \times 0.42 + 0.5 \times 0.42 = 0.42$$

Information Gain = 0.5 - 0.42 = 0.08

Sdot Data

Sdot 반경 200m 이내의 지목 면적 비율, 도로(x축)와 대지(y축)

도	대	종속
0.194081	0.750344	1
0.179794	0.783107	0
0.187805	0.720704	0
0.23362	0.480936	0
0.253656	0.742705	1
0.325552	0.397471	0
0.094508	0.534529	1
0.160991	0.803961	1
0.314716	0.685284	0
0.294449	0.642433	0
0.259395	0.649318	0
0.259483	0.726472	0
0.312998	0.687002	0
0.264695	0.519184	0



Gini Index 계산

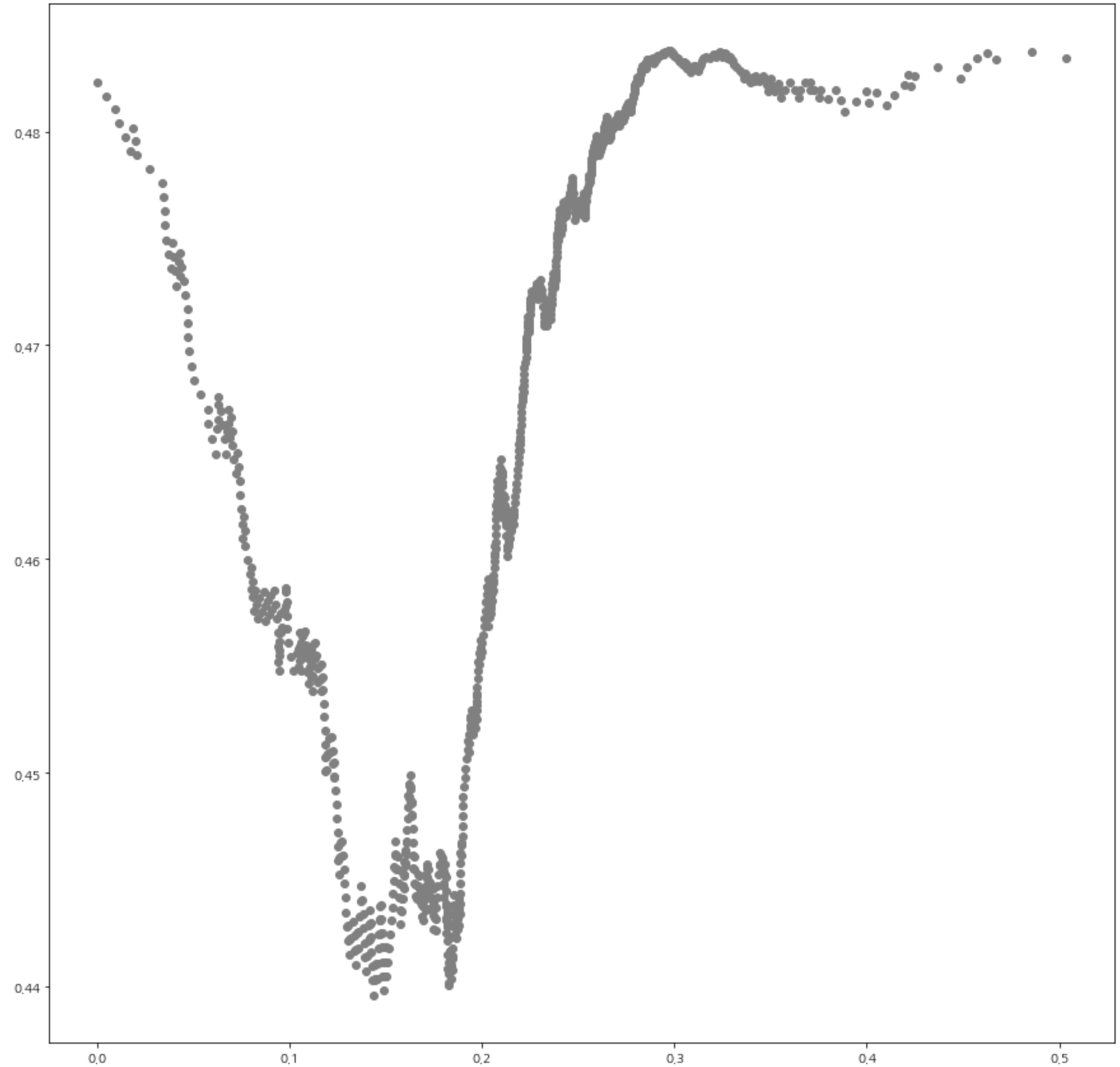
```
tmp = sdot_data_total[['도', '대', '종속']]
tmp['종속']

x = np.array(sdot_data_total[['도', '대']].fillna(0).astype('float').values)
y = np.array(sdot_data_total['종속'].values)
y = y.reshape(y.shape[0], 1)


def GiniIndex(y):
    total = len(y)
    G = 1
    for c in np.unique(y):
        # print(str(c) + "값 : " + str(np.power(np.where(y == c, 1, 0).sum() / total, 2)))
        G = G - np.power(np.where(y == c, 1, 0).sum() / total, 2)
    return G

print(str(GiniIndex(y)))
```

$G(S) =$ 0.4837581124932395

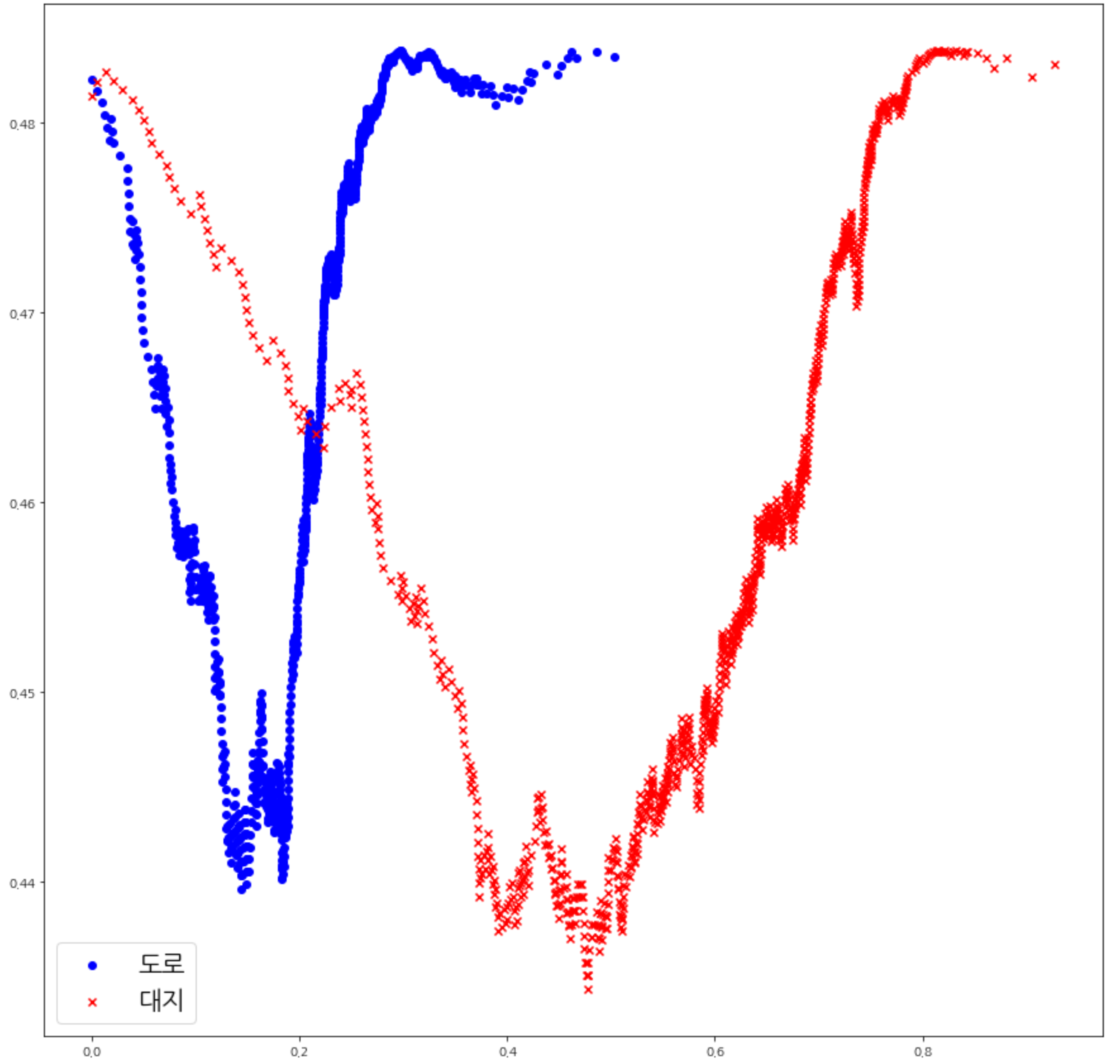
Split Loop

```
criteria = x[:,0]
criteria = np.sort(np.unique(criteria))
total = len(y)
I = np.array([])
for f,l in zip(criteria[:-1], criteria[1:]):
    split = np.mean([f, l])

    s1 = y[np.where(x[:,0] < split, True, False)]
    s2 = y[np.where(x[:,0] > split, True, False)]

    Gini = len(s1) / total *  GiniIndex(s1) + len(s2) / total * GiniIndex(s2)

    I = np.append(I, np.array([f, l, split, Gini]))

I = I.reshape(int(I.shape[0]/4), 4)
```

도로 비율 분할 기준에 따른 Gini Index 변화



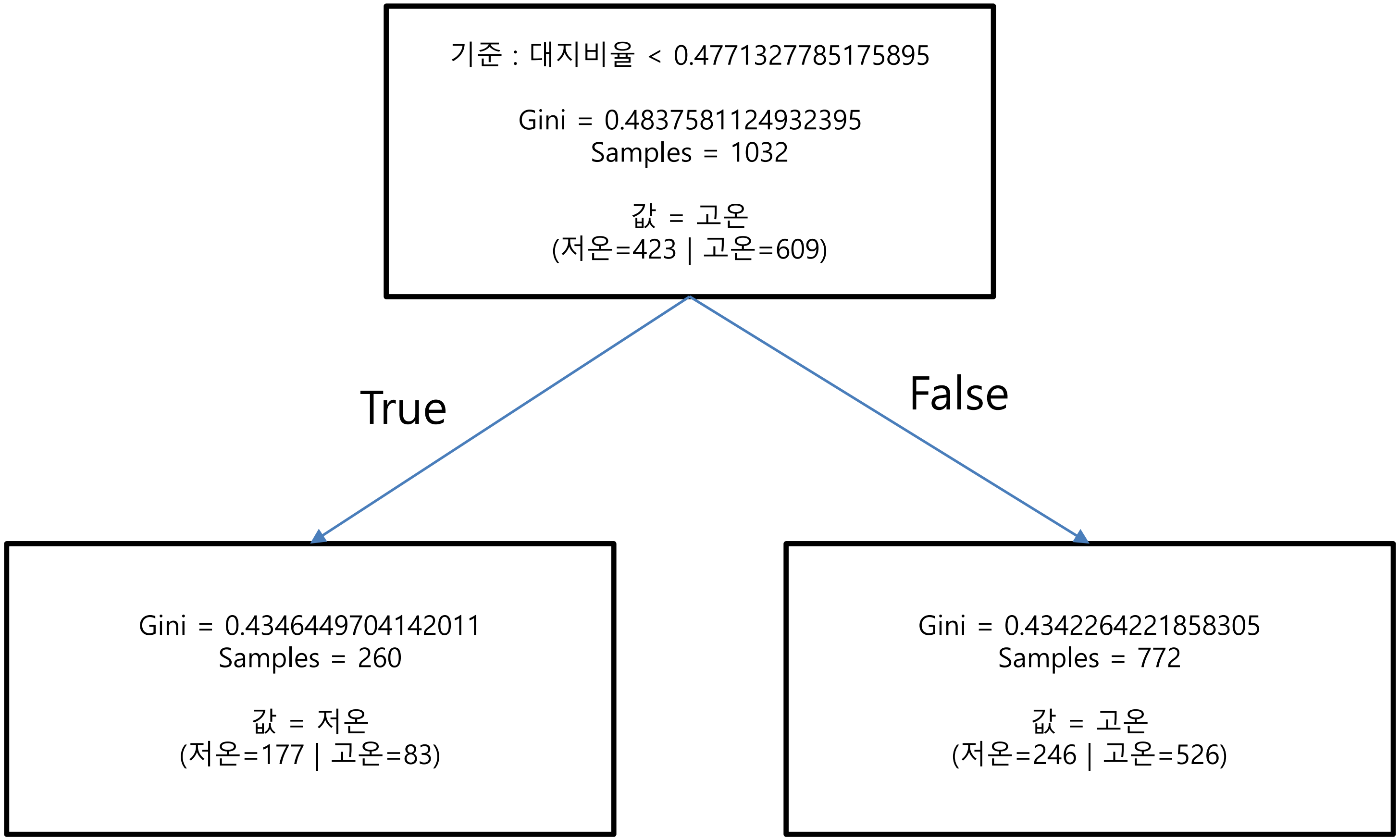
도로와 대지 비율 분할 기준에 따른 Gini Index 변화



도로분할시 최저 Gini : 0.4396058202905748

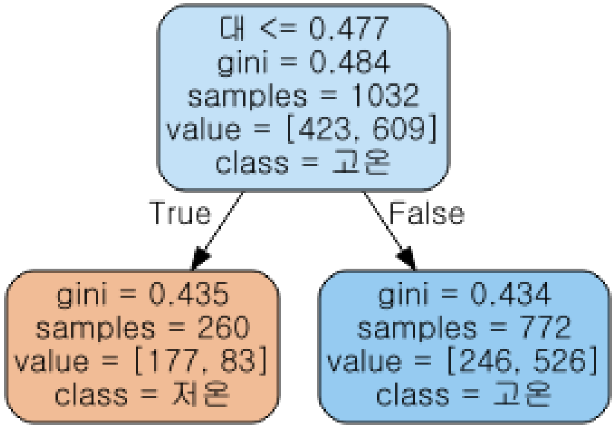
대지분할시 최저 Gini : 0.4343318703829006

Gini Index Dtree

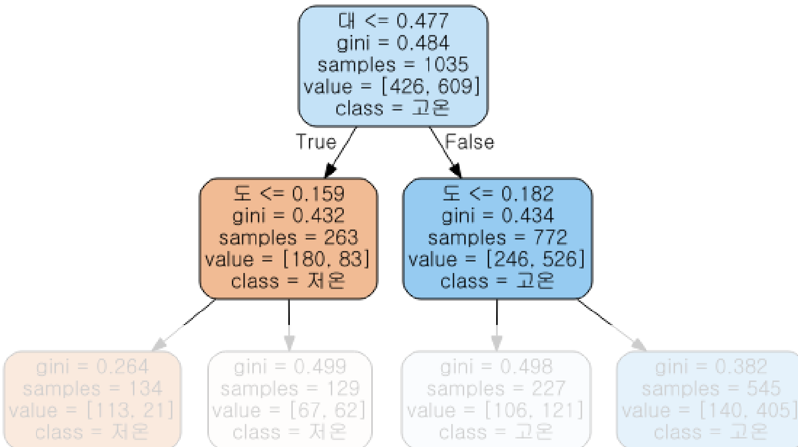


Sklearn 라이브러리 활용시

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.tree import export_graphviz
from graphviz import Source


tree_clf = DecisionTreeClassifier(max_depth=1)
tree_clf.fit(x,y)

score_tr = tree_clf.score(x, y)


dt_dot_data  = export_graphviz(tree_clf,
                               feature_names=['도', '대'],
                               class_names=['저온', '고온'],        # 종속변수
                               rounded = True,
                               filled = True)

Source(dt_dot_data)
```



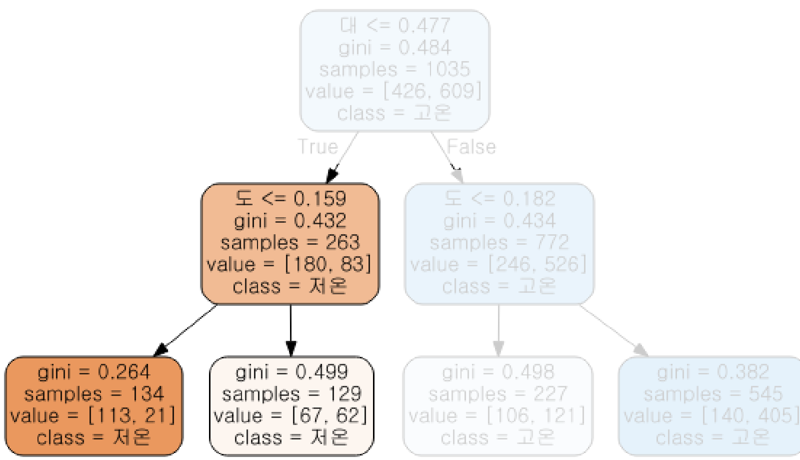
Feature Importance 변수 중요도



Gini = 0.484369

Gini = 0.431985*25.41% + 0.434226*74.59% = 0.433657

Gain = 0.050712 = 대의 중요도



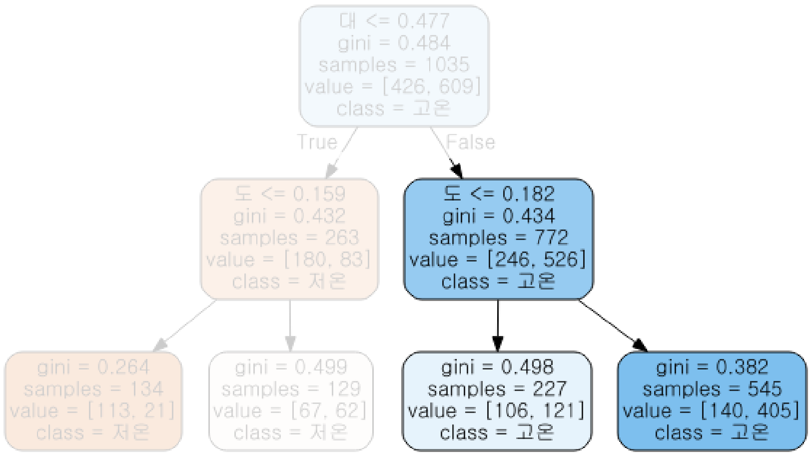
Gini = 0.431985

Gini = 0.264313*50.95% + 0.499249*49.05% = 0.379548

Gain = 0.052437 * 25.41%(parent node) = 0.013325

= 도의 중요도

Feature Importance 변수 중요도



Gini = 0.434226

Gini = 0.497817*29.40% + 0.381786*70.56% = 0.415904

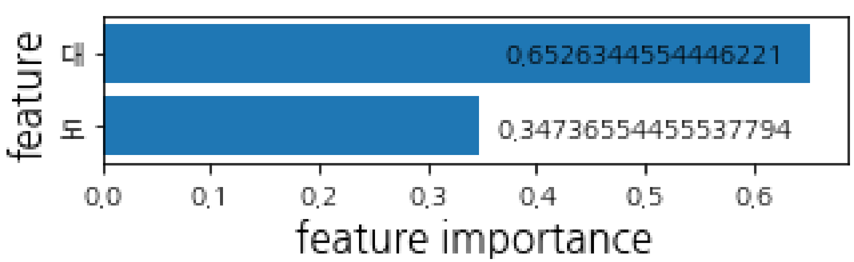
Gain = 0.018322 * 74.59%(parent node) = 0.013666

= 도의 중요도

대의 중요도 = 0.050712 = 0.050712 = 65.26%

도의 중요도 = 0.013325 + 0.013666 = 0.026991 = 34.73%

0.077703



Dtree 과적합 문제를 해결하기위해 = Pruning(가지치기)

Depth = 20

Depth	R2
1	0.681202
2	0.681202
3	0.712209
4	0.732558
5	0.744186
6	0.767442
7	0.793605
8	0.828488
9	0.850775
10	0.880814
11	0.897287
12	0.921512
13	0.935078
14	0.944767
15	0.959302
16	0.963178
17	0.968023
18	0.971899
19	0.98062

2진 분류 성능평가지표

		실제 정답	
		True	False
분류 결과	True	True Positive	False Positive
	False	False Negative	True Negative

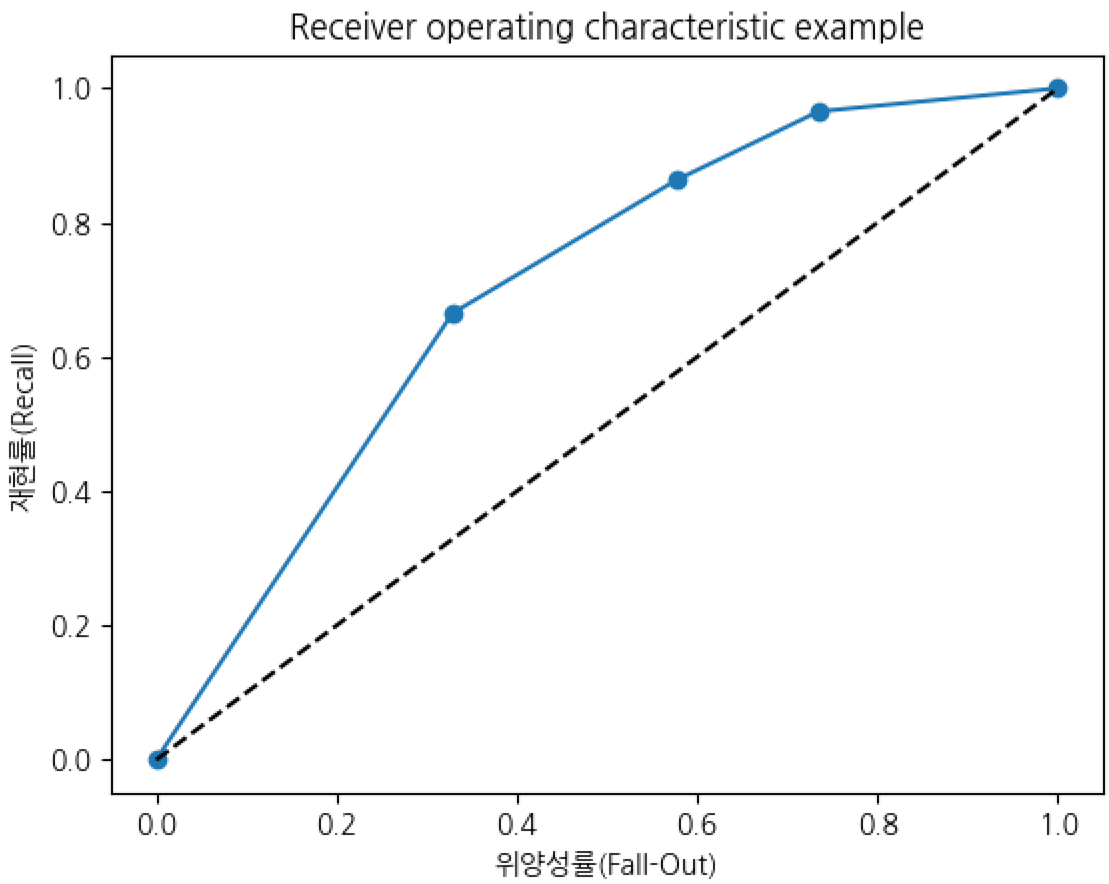
$$(Precision) = \frac{TP}{TP + FP}$$

$$(Recall) = \frac{TP}{TP + FN}$$

$$(Accuracy) = \frac{TP + TN}{TP + FN + FP + TN}$$

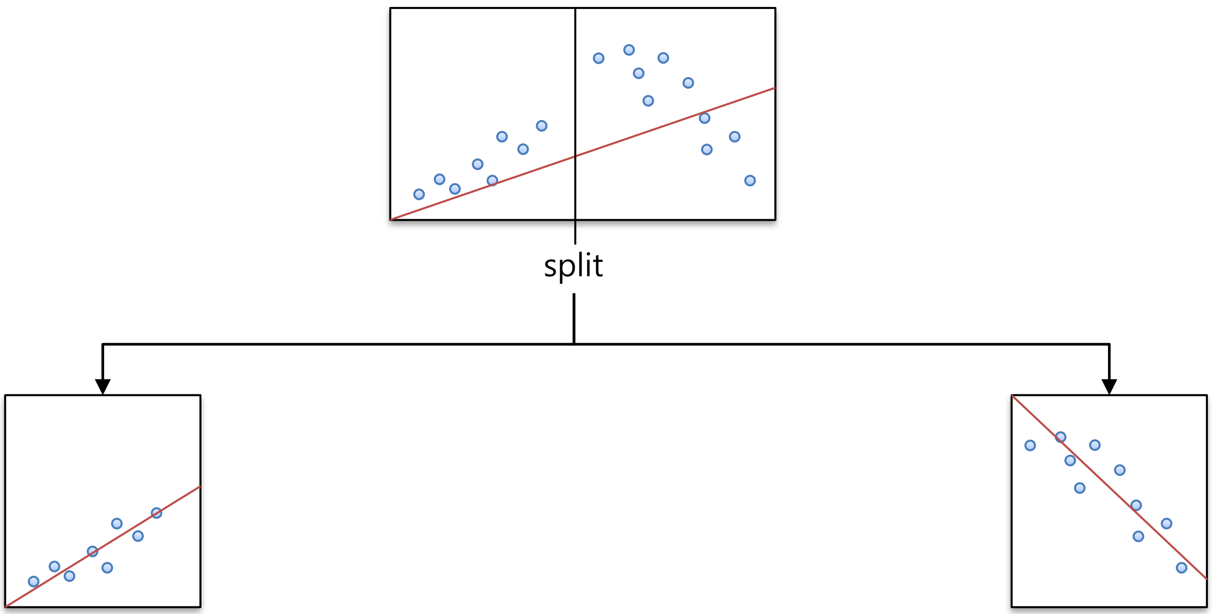
$$(F1\text{-}score) = 2\times\frac{1}{\frac{1}{Precision}+\frac{1}{Recall}} = 2\times\frac{Precision \times Recall}{Precision + Recall}$$

ROC curve / AUC



Regression Tree

$$SSE = \sum_{i=1}^{n}\left((y_i - \hat{y})^2\right) = 100$$



$$SSE_1 = \sum_{i=1}^{n_1}\left((y_i - \hat{y})^2\right) = 12$$

$$SSE_2 = \sum_{i=1}^{n_2}\left((y_i - \hat{y})^2\right) = 13$$

$$GAIN = SSE - (SSE_1 + SSE_2)$$

ID3 entropy (C4.5, C5.0)

$$Entropy(S) = \sum_{i=1}^{c} p_i * I(x_i) \quad , \quad I(x) = log_2 \frac{1}{p(x)}$$