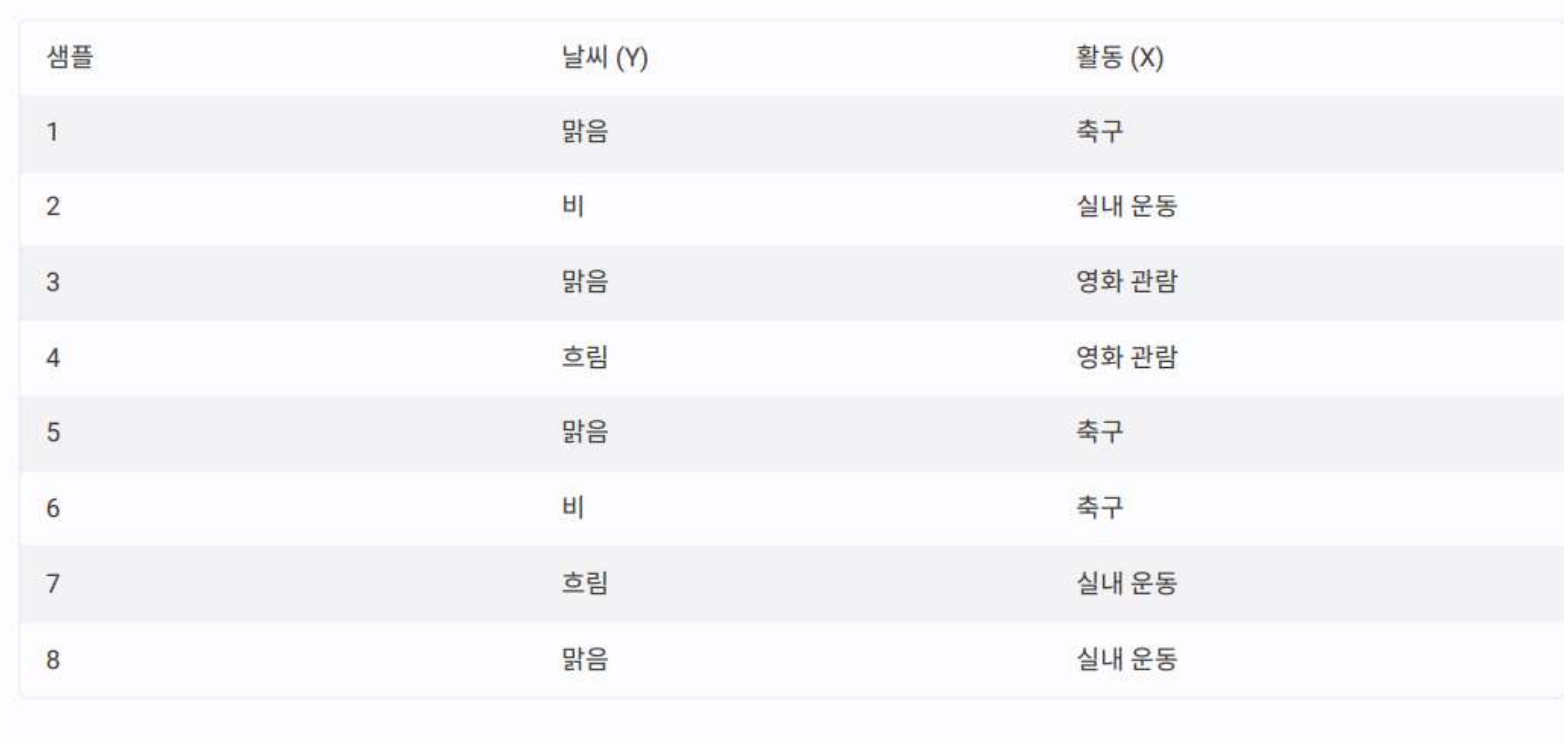
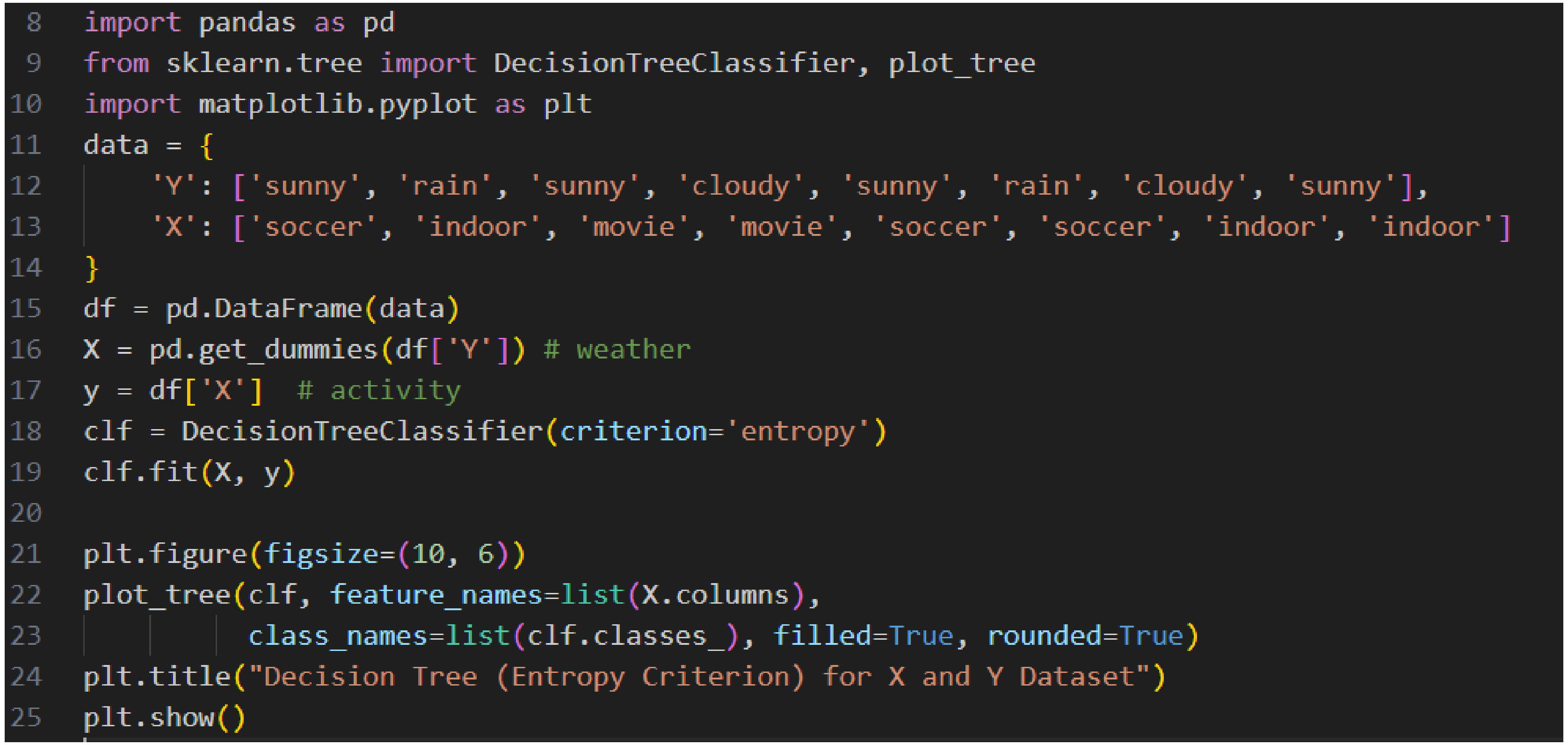
날씨,활동 데이터

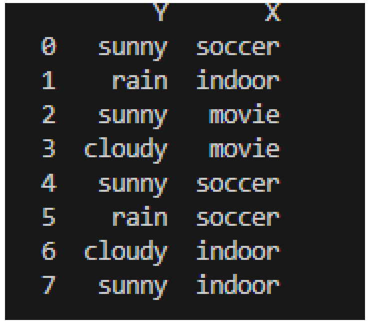


샘플	날씨 (Y)	활동 (X)
1	맑음	축구
2	비	실내 운동
3	맑음	영화 관람
4	흐림	영화 관람
5	맑음	축구
6	비	축구
7	흐림	실내 운동
8	맑음	실내 운동

DecisionTree Classification – entropy 설정

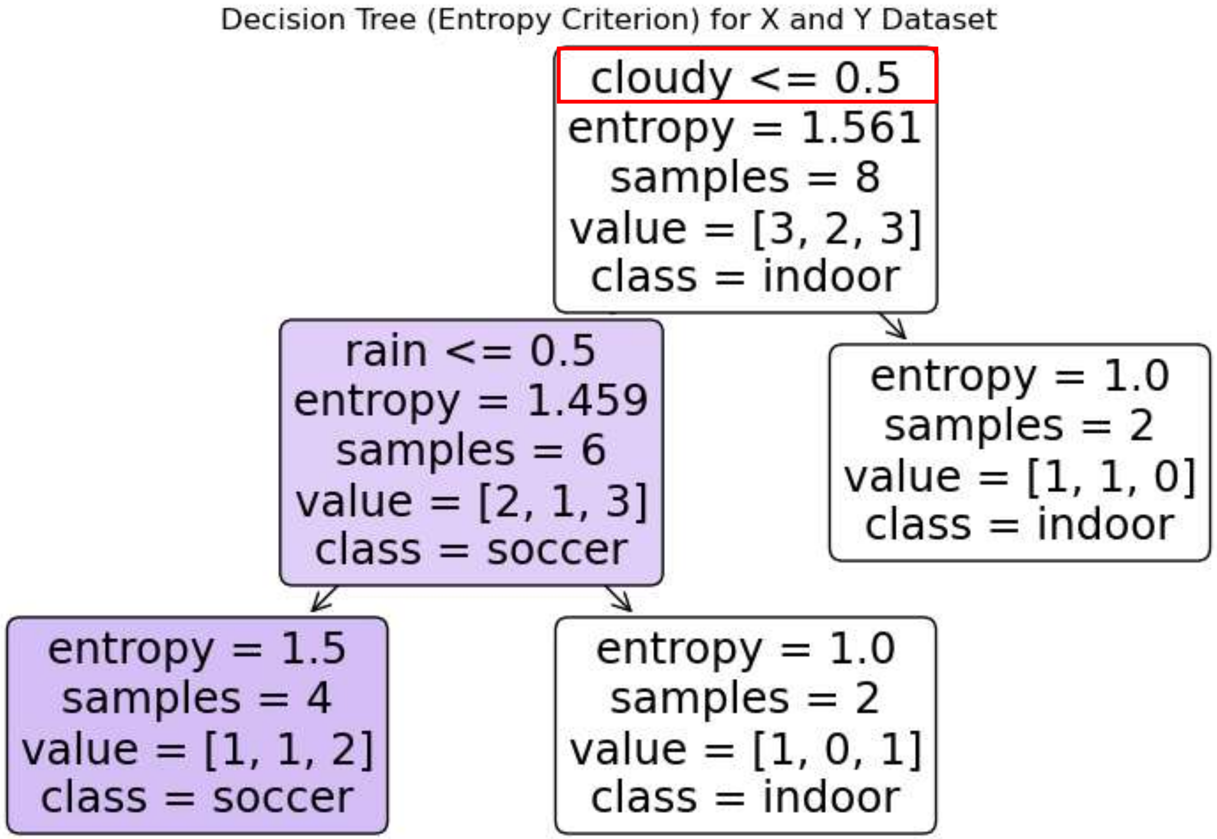


```
import pandas as pd
from sklearn.tree import DecisionTreeClassifier, plot_tree
import matplotlib.pyplot as plt
data = {
    'Y': ['sunny', 'rain', 'sunny', 'cloudy', 'sunny', 'rain', 'cloudy', 'sunny'],
    'X': ['soccer', 'indoor', 'movie', 'movie', 'soccer', 'soccer', 'indoor', 'indoor']
}
df = pd.DataFrame(data)
X = pd.get_dummies(df['Y']) # weather
y = df['X']  # activity
clf = DecisionTreeClassifier(criterion='entropy')
clf.fit(X, y)

plt.figure(figsize=(10, 6))
plot_tree(clf, feature_names=list(X.columns),
          class_names=list(clf.classes_), filled=True, rounded=True)
plt.title("Decision Tree (Entropy Criterion) for X and Y Dataset")
plt.show()
```

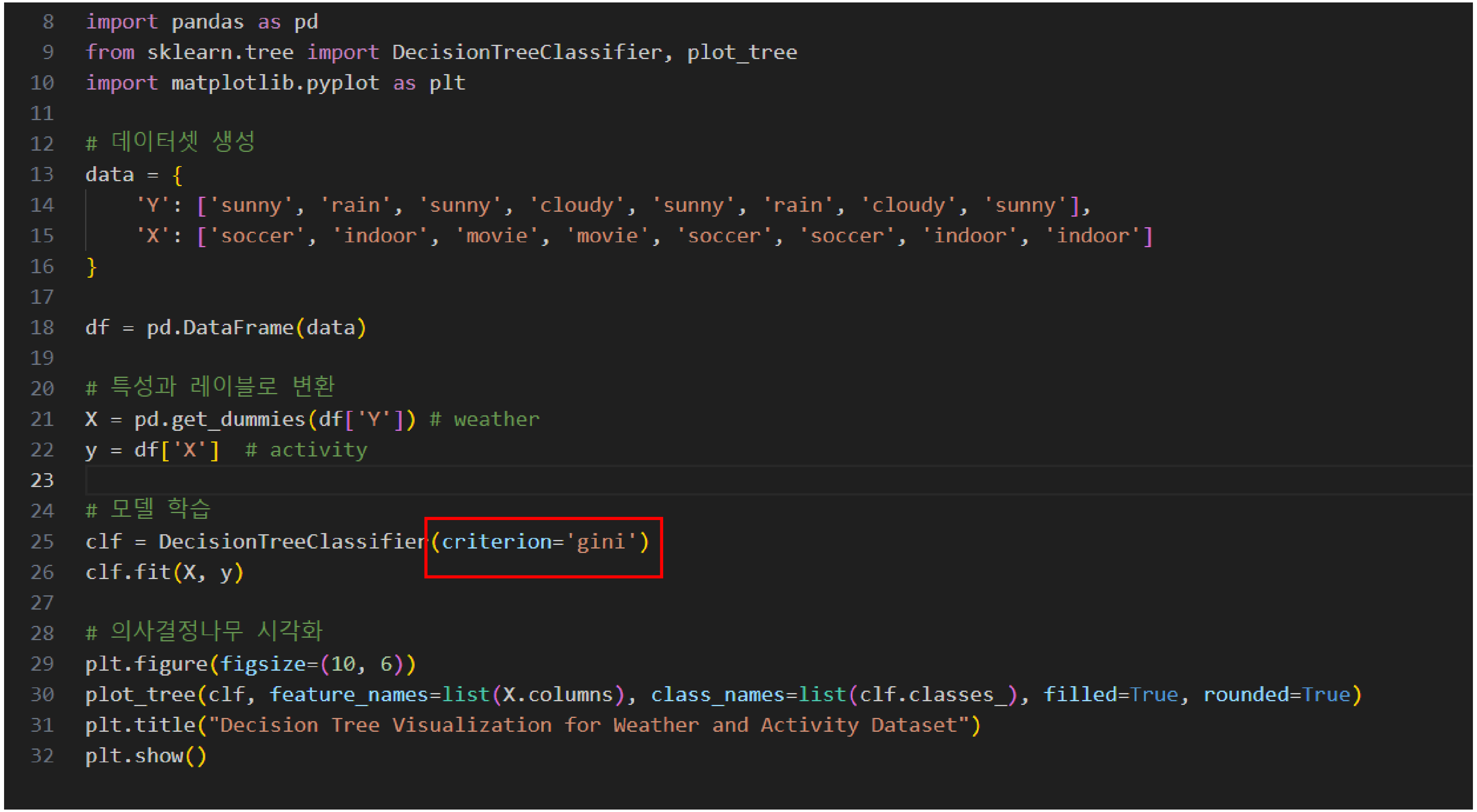


```
        Y       X
0   sunny  soccer
1    rain  indoor
2   sunny   movie
3  cloudy   movie
4   sunny  soccer
5    rain  soccer
6  cloudy  indoor
7   sunny  indoor
```

결과

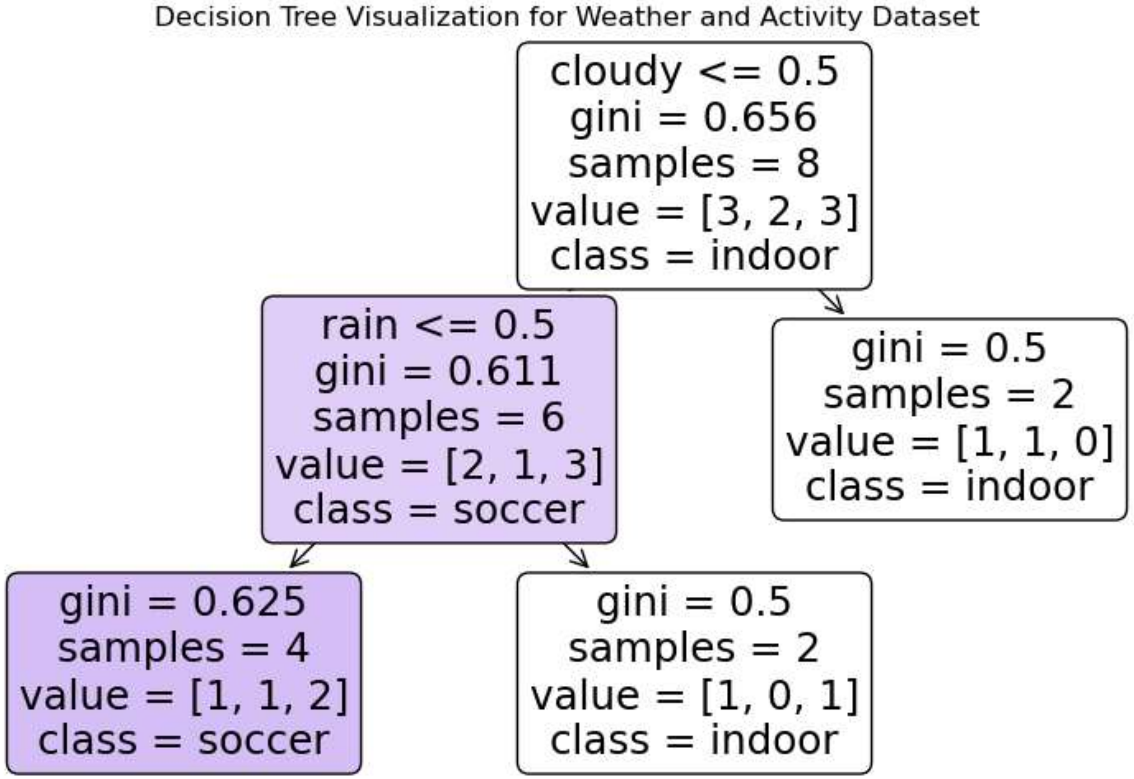

Decision Tree (Entropy Criterion) for X and Y Dataset
cloudy <= 0.5
entropy = 1.561
samples = 8
value = [3, 2, 3]
class = indoor
rain <= 0.5
entropy = 1.459
samples = 6
value = [2, 1, 3]
class = soccer
entropy = 1.0
samples = 2
value = [1, 1, 0]
class = indoor
entropy = 1.5
samples = 4
value = [1, 1, 2]
class = soccer
entropy = 1.0
samples = 2
value = [1, 0, 1]
class = indoor

DecisionTree Classification – gini 설정

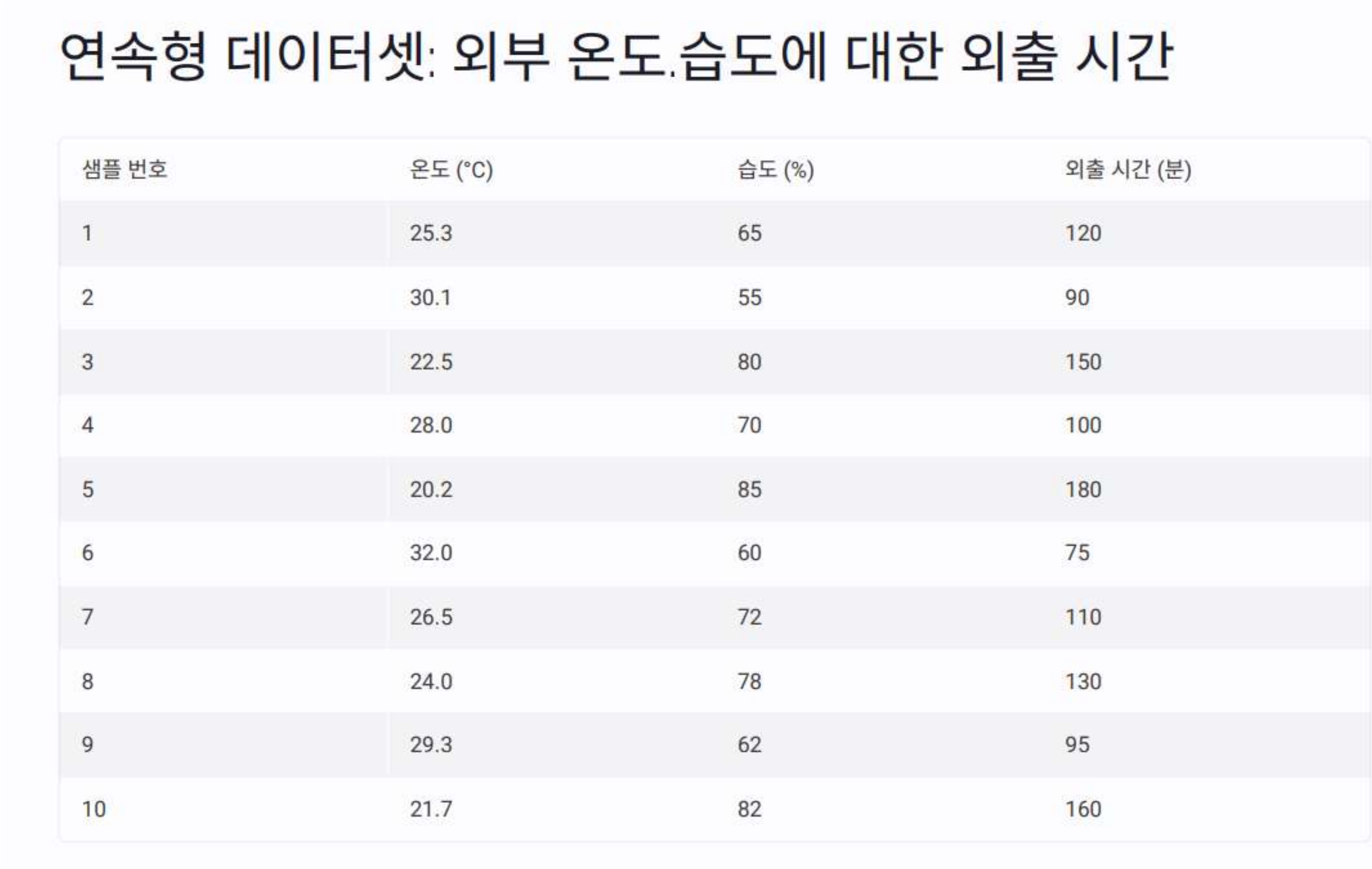


```python
import pandas as pd
from sklearn.tree import DecisionTreeClassifier, plot_tree
import matplotlib.pyplot as plt

# 데이터셋 생성
data = {
    'Y': ['sunny', 'rain', 'sunny', 'cloudy', 'sunny', 'rain', 'cloudy', 'sunny'],
    'X': ['soccer', 'indoor', 'movie', 'movie', 'soccer', 'soccer', 'indoor', 'indoor']
}

df = pd.DataFrame(data)

# 특성과 레이블로 변환
X = pd.get_dummies(df['Y']) # weather
y = df['X']  # activity

# 모델 학습
clf = DecisionTreeClassifier(criterion='gini')
clf.fit(X, y)

# 의사결정나무 시각화
plt.figure(figsize=(10, 6))
plot_tree(clf, feature_names=list(X.columns), class_names=list(clf.classes_), filled=True, rounded=True)
plt.title("Decision Tree Visualization for Weather and Activity Dataset")
plt.show()
```

결과


Decision Tree Visualization for Weather and Activity Dataset
cloudy <= 0.5
gini = 0.656
samples = 8
value = [3, 2, 3]
class = indoor
rain <= 0.5
gini = 0.611
samples = 6
value = [2, 1, 3]
class = soccer
gini = 0.5
samples = 2
value = [1, 1, 0]
class = indoor
gini = 0.625
samples = 4
value = [1, 1, 2]
class = soccer
gini = 0.5
samples = 2
value = [1, 0, 1]
class = indoor

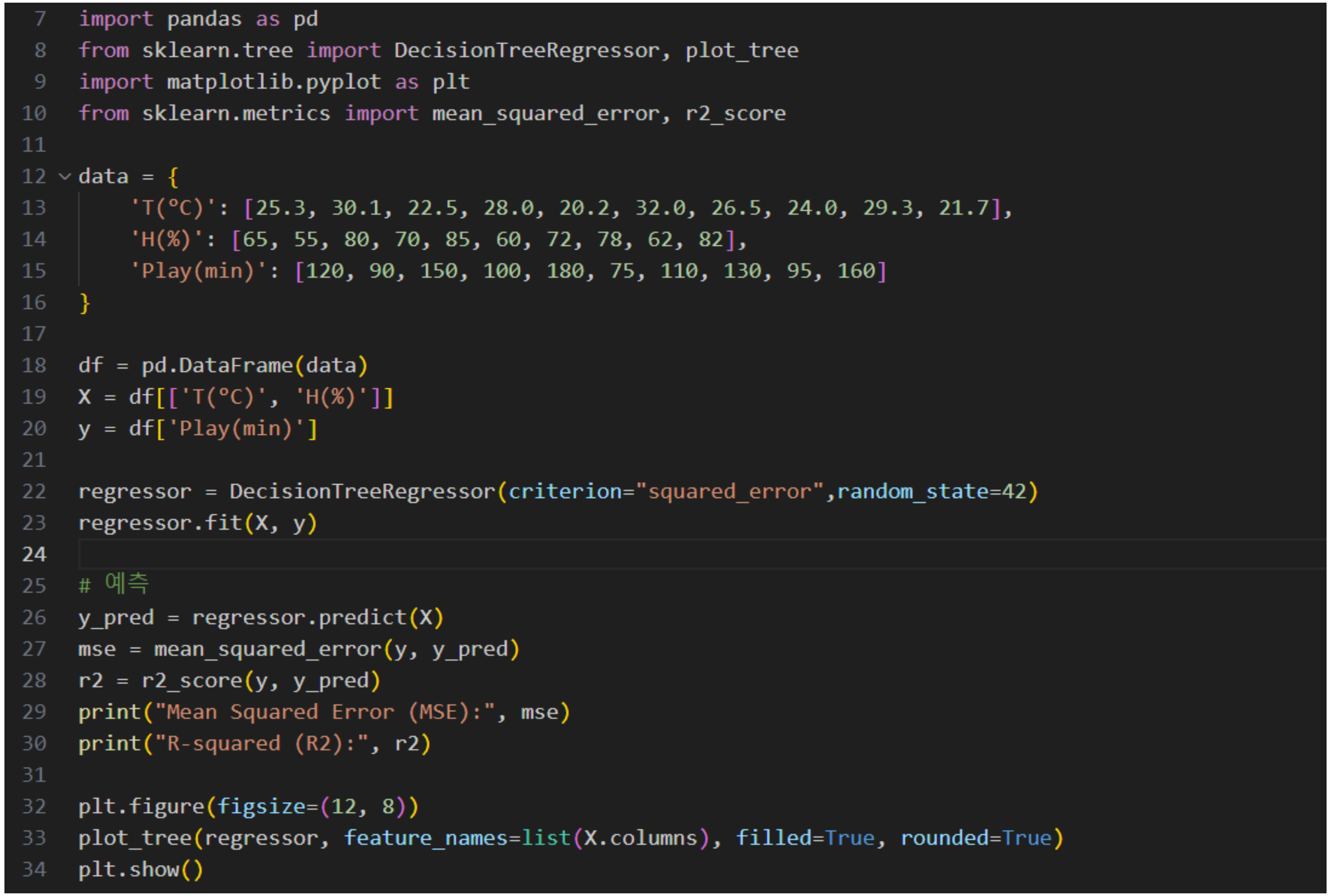
온도,습도,외출시간 데이터



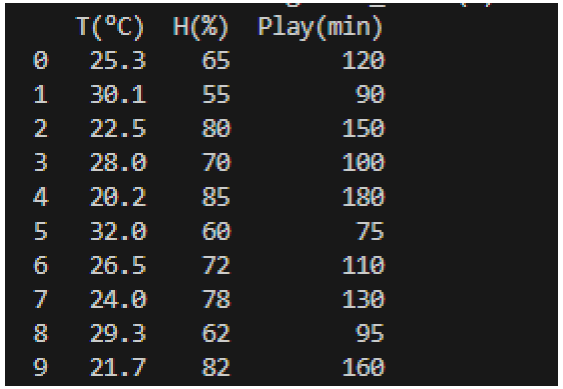
연속형 데이터셋: 외부 온도.습도에 대한 외출 시간

샘플 번호	온도 (°C)	습도 (%)	외출 시간 (분)
1	25.3	65	120
2	30.1	55	90
3	22.5	80	150
4	28.0	70	100
5	20.2	85	180
6	32.0	60	75
7	26.5	72	110
8	24.0	78	130
9	29.3	62	95
10	21.7	82	160

DecisionTreeRegressor

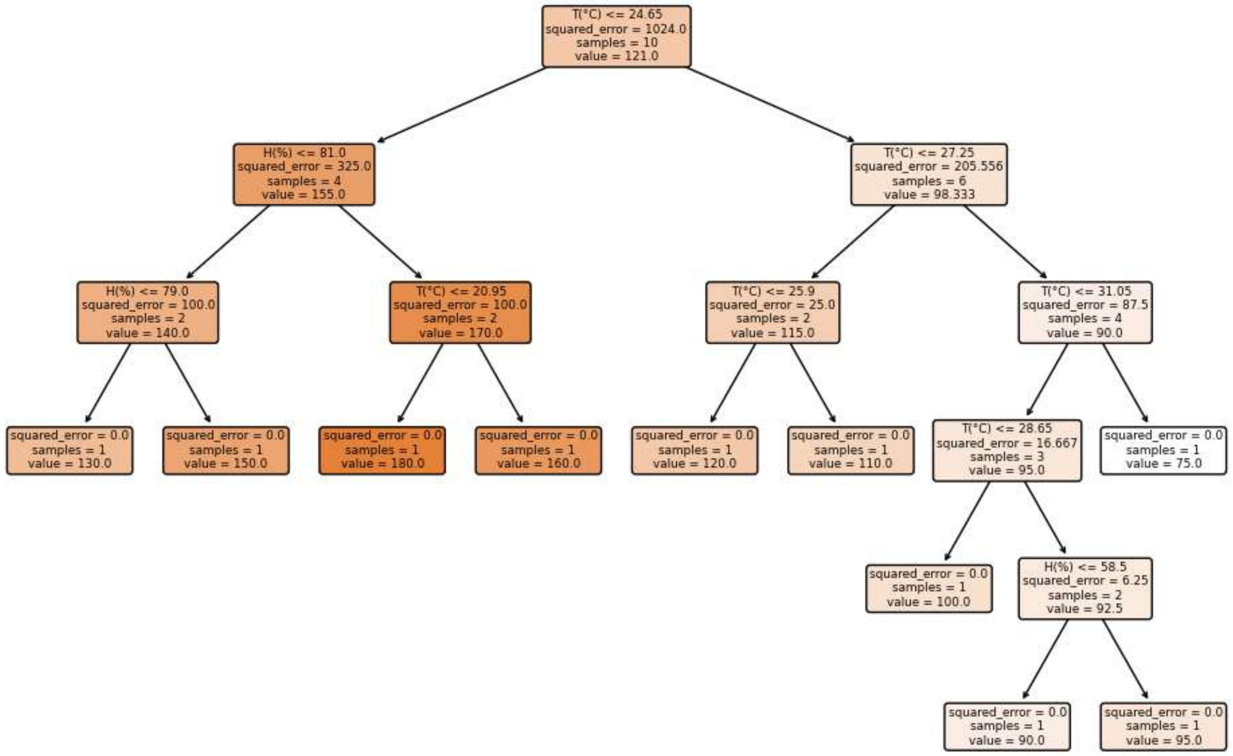


```python
import pandas as pd
from sklearn.tree import DecisionTreeRegressor, plot_tree
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_error, r2_score

data = {
    'T(°C)': [25.3, 30.1, 22.5, 28.0, 20.2, 32.0, 26.5, 24.0, 29.3, 21.7],
    'H(%)': [65, 55, 80, 70, 85, 60, 72, 78, 62, 82],
    'Play(min)': [120, 90, 150, 100, 180, 75, 110, 130, 95, 160]
}

df = pd.DataFrame(data)
X = df[['T(°C)', 'H(%)']]
y = df['Play(min)']

regressor = DecisionTreeRegressor(criterion="squared_error",random_state=42)
regressor.fit(X, y)

# 예측
y_pred = regressor.predict(X)
mse = mean_squared_error(y, y_pred)
r2 = r2_score(y, y_pred)
print("Mean Squared Error (MSE):", mse)
print("R-squared (R2):", r2)

plt.figure(figsize=(12, 8))
plot_tree(regressor, feature_names=list(X.columns), filled=True, rounded=True)
plt.show()
```

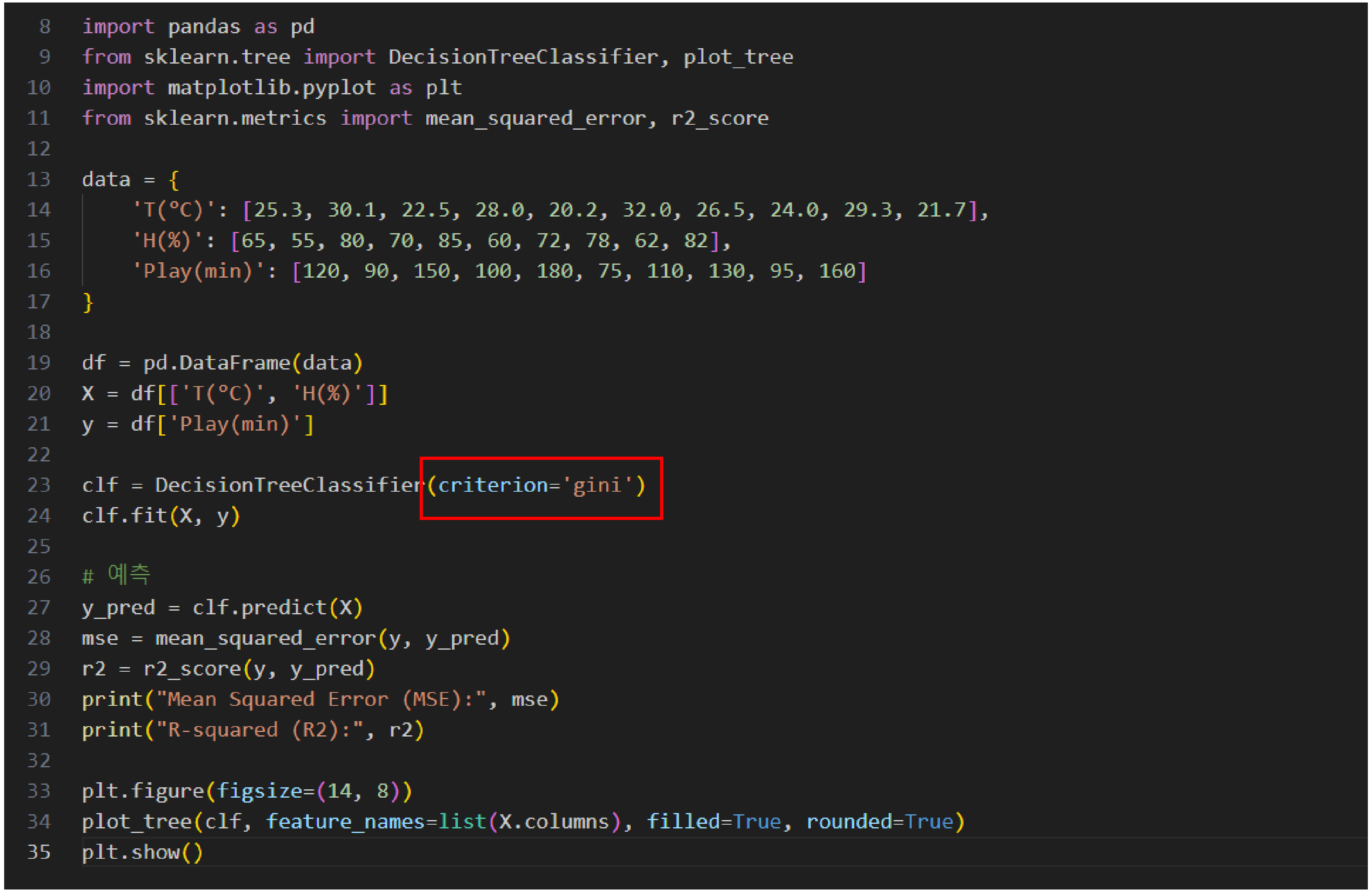


```
   T(°C)  H(%)  Play(min)
0   25.3    65        120
1   30.1    55         90
2   22.5    80        150
3   28.0    70        100
4   20.2    85        180
5   32.0    60         75
6   26.5    72        110
7   24.0    78        130
8   29.3    62         95
9   21.7    82        160
```

결과

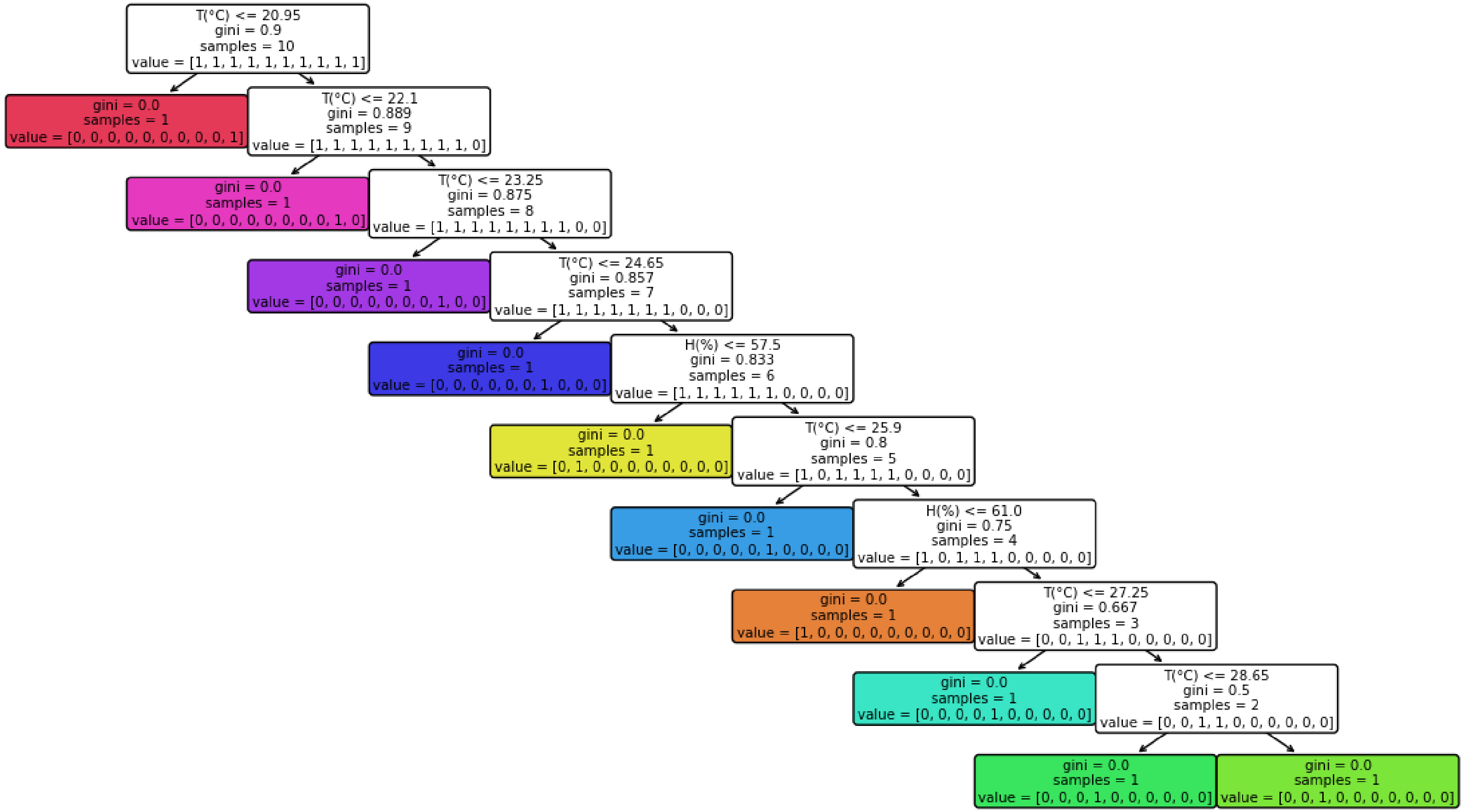

T(°C) <= 24.65
squared_error = 1024.0
samples = 10
value = 121.0
H(%) <= 81.0
squared_error = 325.0
samples = 4
value = 155.0
T(°C) <= 27.25
squared_error = 205.556
samples = 6
value = 98.333
H(%) <= 79.0
squared_error = 100.0
samples = 2
value = 140.0
T(°C) <= 20.95
squared_error = 100.0
samples = 2
value = 170.0
T(°C) <= 25.9
squared_error = 25.0
samples = 2
value = 115.0
T(°C) <= 31.05
squared_error = 87.5
samples = 4
value = 90.0
squared_error = 0.0
samples = 1
value = 130.0
squared_error = 0.0
samples = 1
value = 150.0
squared_error = 0.0
samples = 1
value = 180.0
squared_error = 0.0
samples = 1
value = 160.0
squared_error = 0.0
samples = 1
value = 120.0
squared_error = 0.0
samples = 1
value = 110.0
T(°C) <= 28.65
squared_error = 16.667
samples = 3
value = 95.0
squared_error = 0.0
samples = 1
value = 75.0
squared_error = 0.0
samples = 1
value = 100.0
H(%) <= 58.5
squared_error = 6.25
samples = 2
value = 92.5
squared_error = 0.0
samples = 1
value = 90.0
squared_error = 0.0
samples = 1
value = 95.0

DecisionTree Classifier

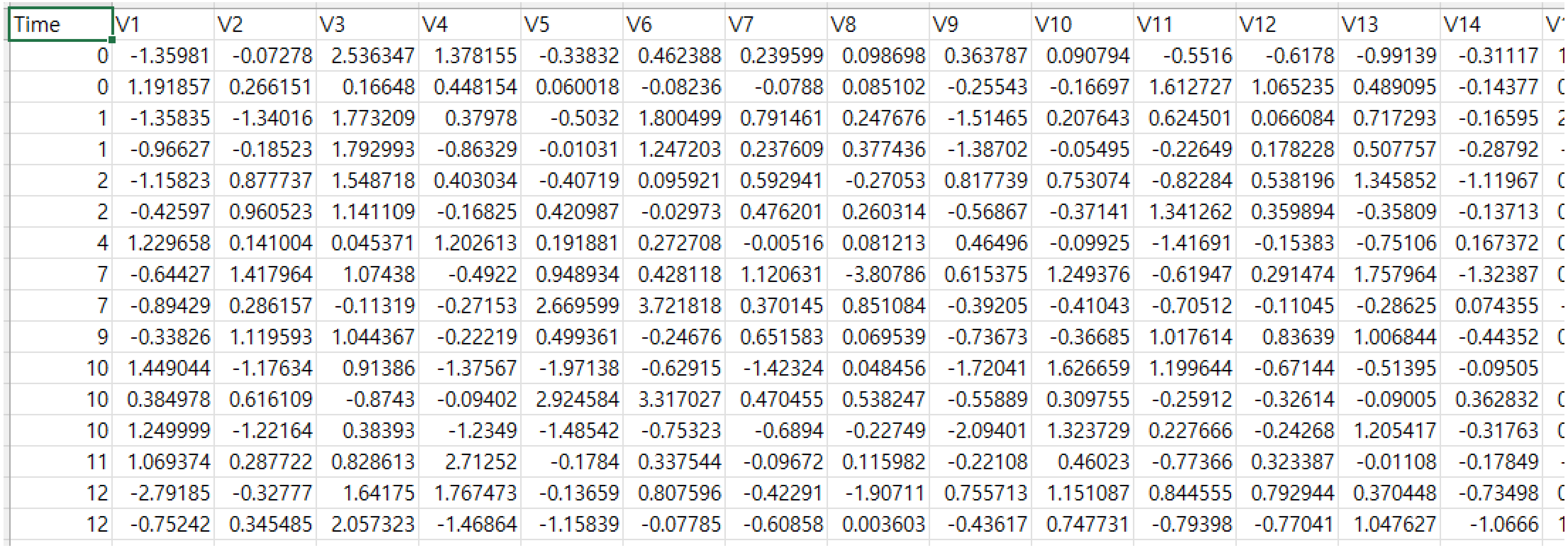


```python
import pandas as pd
from sklearn.tree import DecisionTreeClassifier, plot_tree
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_error, r2_score

data = {
    'T(°C)': [25.3, 30.1, 22.5, 28.0, 20.2, 32.0, 26.5, 24.0, 29.3, 21.7],
    'H(%)': [65, 55, 80, 70, 85, 60, 72, 78, 62, 82],
    'Play(min)': [120, 90, 150, 100, 180, 75, 110, 130, 95, 160]
}

df = pd.DataFrame(data)
X = df[['T(°C)', 'H(%)']]
y = df['Play(min)']

clf = DecisionTreeClassifier(criterion='gini')
clf.fit(X, y)

# 예측
y_pred = clf.predict(X)
mse = mean_squared_error(y, y_pred)
r2 = r2_score(y, y_pred)
print("Mean Squared Error (MSE):", mse)
print("R-squared (R2):", r2)

plt.figure(figsize=(14, 8))
plot_tree(clf, feature_names=list(X.columns), filled=True, rounded=True)
plt.show()
```

결과

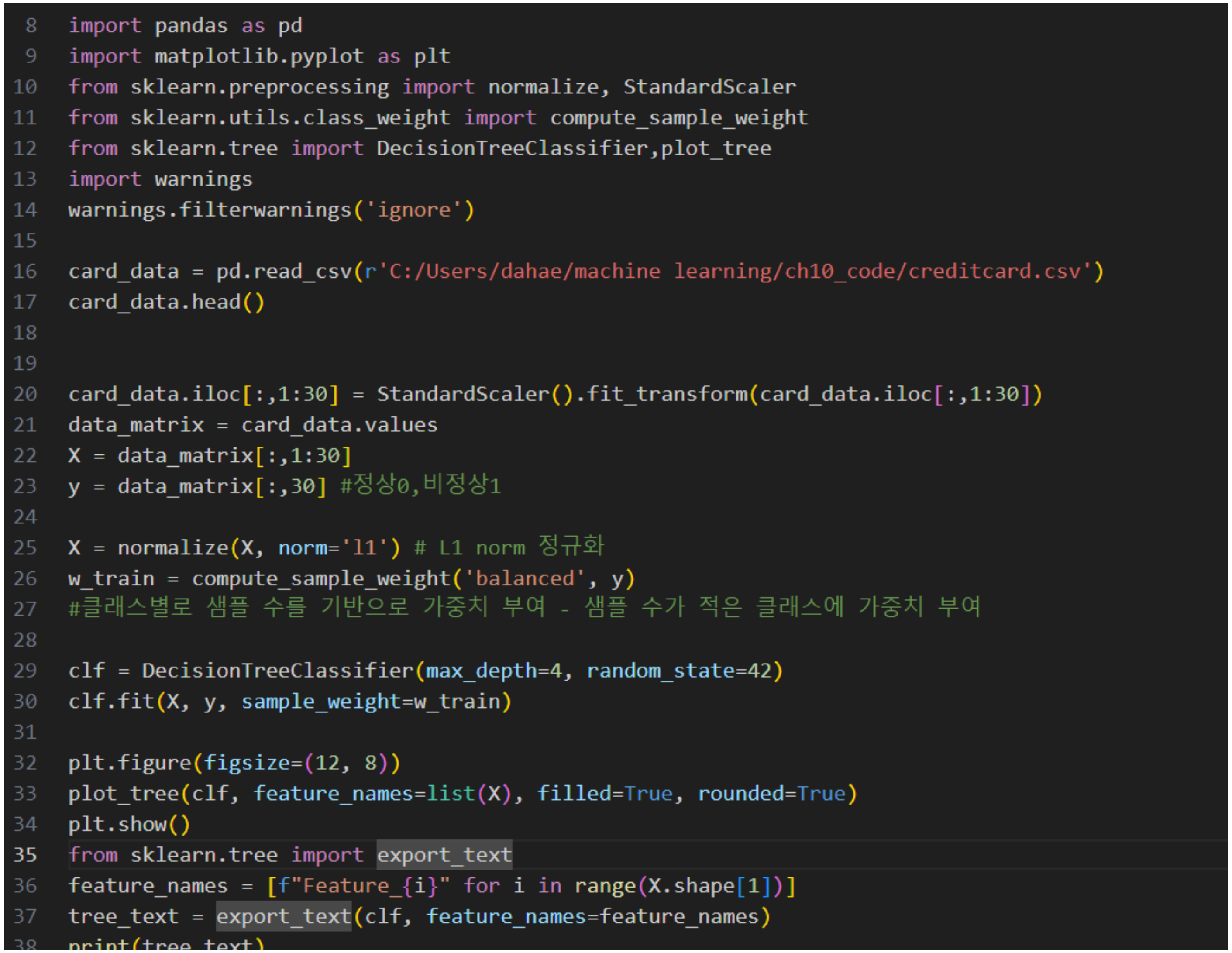

T(°C) <= 20.95
gini = 0.9
samples = 10
value = [1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
gini = 0.0
samples = 1
value = [0, 0, 0, 0, 0, 0, 0, 0, 0, 1]
T(°C) <= 22.1
gini = 0.889
samples = 9
value = [1, 1, 1, 1, 1, 1, 1, 1, 1, 0]
gini = 0.0
samples = 1
value = [0, 0, 0, 0, 0, 0, 0, 0, 1, 0]
T(°C) <= 23.25
gini = 0.875
samples = 8
value = [1, 1, 1, 1, 1, 1, 1, 1, 0, 0]
gini = 0.0
samples = 1
value = [0, 0, 0, 0, 0, 0, 0, 1, 0, 0]
T(°C) <= 24.65
gini = 0.857
samples = 7
value = [1, 1, 1, 1, 1, 1, 1, 0, 0, 0]
gini = 0.0
samples = 1
value = [0, 0, 0, 0, 0, 0, 1, 0, 0, 0]
H(%) <= 57.5
gini = 0.833
samples = 6
value = [1, 1, 1, 1, 1, 1, 0, 0, 0, 0]
gini = 0.0
samples = 1
value = [0, 1, 0, 0, 0, 0, 0, 0, 0, 0]
T(°C) <= 25.9
gini = 0.8
samples = 5
value = [1, 0, 1, 1, 1, 1, 0, 0, 0, 0]
gini = 0.0
samples = 1
value = [0, 0, 0, 0, 0, 1, 0, 0, 0, 0]
H(%) <= 61.0
gini = 0.75
samples = 4
value = [1, 0, 1, 1, 1, 0, 0, 0, 0, 0]
gini = 0.0
samples = 1
value = [1, 0, 0, 0, 0, 0, 0, 0, 0, 0]
T(°C) <= 27.25
gini = 0.667
samples = 3
value = [0, 0, 1, 1, 1, 0, 0, 0, 0, 0]
gini = 0.0
samples = 1
value = [0, 0, 0, 0, 1, 0, 0, 0, 0, 0]
T(°C) <= 28.65
gini = 0.5
samples = 2
value = [0, 0, 1, 1, 0, 0, 0, 0, 0, 0]
gini = 0.0
samples = 1
value = [0, 0, 0, 1, 0, 0, 0, 0, 0, 0]
gini = 0.0
samples = 1
value = [0, 0, 1, 0, 0, 0, 0, 0, 0, 0]

신용카드 발급사기 판단



Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14	V1
0	-1.35981	-0.07278	2.536347	1.378155	-0.33832	0.462388	0.239599	0.098698	0.363787	0.090794	-0.5516	-0.6178	-0.99139	-0.31117	1
0	1.191857	0.266151	0.16648	0.448154	0.060018	-0.08236	-0.0788	0.085102	-0.25543	-0.16697	1.612727	1.065235	0.489095	-0.14377	0
1	-1.35835	-1.34016	1.773209	0.37978	-0.5032	1.800499	0.791461	0.247676	-1.51465	0.207643	0.624501	0.066084	0.717293	-0.16595	2
1	-0.96627	-0.18523	1.792993	-0.86329	-0.01031	1.247203	0.237609	0.377436	-1.38702	-0.05495	-0.22649	0.178228	0.507757	-0.28792	-
2	-1.15823	0.877737	1.548718	0.403034	-0.40719	0.095921	0.592941	-0.27053	0.817739	0.753074	-0.82284	0.538196	1.345852	-1.11967	0
2	-0.42597	0.960523	1.141109	-0.16825	0.420987	-0.02973	0.476201	0.260314	-0.56867	-0.37141	1.341262	0.359894	-0.35809	-0.13713	0
4	1.229658	0.141004	0.045371	1.202613	0.191881	0.272708	-0.00516	0.081213	0.46496	-0.09925	-1.41691	-0.15383	-0.75106	0.167372	0
7	-0.64427	1.417964	1.07438	-0.4922	0.948934	0.428118	1.120631	-3.80786	0.615375	1.249376	-0.61947	0.291474	1.757964	-1.32387	0
7	-0.89429	0.286157	-0.11319	-0.27153	2.669599	3.721818	0.370145	0.851084	-0.39205	-0.41043	-0.70512	-0.11045	-0.28625	0.074355	-
9	-0.33826	1.119593	1.044367	-0.22219	0.499361	-0.24676	0.651583	0.069539	-0.73673	-0.36685	1.017614	0.83639	1.006844	-0.44352	0
10	1.449044	-1.17634	0.91386	-1.37567	-1.97138	-0.62915	-1.42324	0.048456	-1.72041	1.626659	1.199644	-0.67144	-0.51395	-0.09505	
10	0.384978	0.616109	-0.8743	-0.09402	2.924584	3.317027	0.470455	0.538247	-0.55889	0.309755	-0.25912	-0.32614	-0.09005	0.362832	0
10	1.249999	-1.22164	0.38393	-1.2349	-1.48542	-0.75323	-0.6894	-0.22749	-2.09401	1.323729	0.227666	-0.24268	1.205417	-0.31763	0
11	1.069374	0.287722	0.828613	2.71252	-0.1784	0.337544	-0.09672	0.115982	-0.22108	0.46023	-0.77366	0.323387	-0.01108	-0.17849	-
12	-2.79185	-0.32777	1.64175	1.767473	-0.13659	0.807596	-0.42291	-1.90711	0.755713	1.151087	0.844555	0.792944	0.370448	-0.73498	0
12	-0.75242	0.345485	2.057323	-1.46864	-1.15839	-0.07785	-0.60858	0.003603	-0.43617	0.747731	-0.79398	-0.77041	1.047627	-1.0666	1

신용카드 발급사기 판단



```python
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import normalize, StandardScaler
from sklearn.utils.class_weight import compute_sample_weight
from sklearn.tree import DecisionTreeClassifier,plot_tree
import warnings
warnings.filterwarnings('ignore')

card_data = pd.read_csv(r'C:/Users/dahae/machine learning/ch10_code/creditcard.csv')
card_data.head()


card_data.iloc[:,1:30] = StandardScaler().fit_transform(card_data.iloc[:,1:30])
data_matrix = card_data.values
X = data_matrix[:,1:30]
y = data_matrix[:,30] #정상0,비정상1

X = normalize(X, norm='l1') # L1 norm 정규화
w_train = compute_sample_weight('balanced', y)
#클래스별로 샘플 수를 기반으로 가중치 부여 - 샘플 수가 적은 클래스에 가중치 부여

clf = DecisionTreeClassifier(max_depth=4, random_state=42)
clf.fit(X, y, sample_weight=w_train)

plt.figure(figsize=(12, 8))
plot_tree(clf, feature_names=list(X), filled=True, rounded=True)
plt.show()
from sklearn.tree import export_text
feature_names = [f"Feature_{i}" for i in range(X.shape[1])]
tree_text = export_text(clf, feature_names=feature_names)
print(tree_text)
```

신용카드 발급사기 판단