머신러닝응용 제09강

Decision Tree

첨단공학부 김동하교수

제09강 Decision Tree

1	의사결정나무의 개념에 대해 학습한다.
2	나무의 성장과 가지치기에 대해 학습한다.
3	CART 알고리즘에 대해 학습한다.

핵심 단어

- 분리 규칙
- 불순도
- 성장하기
- 가지치기

09강. Decision Tree

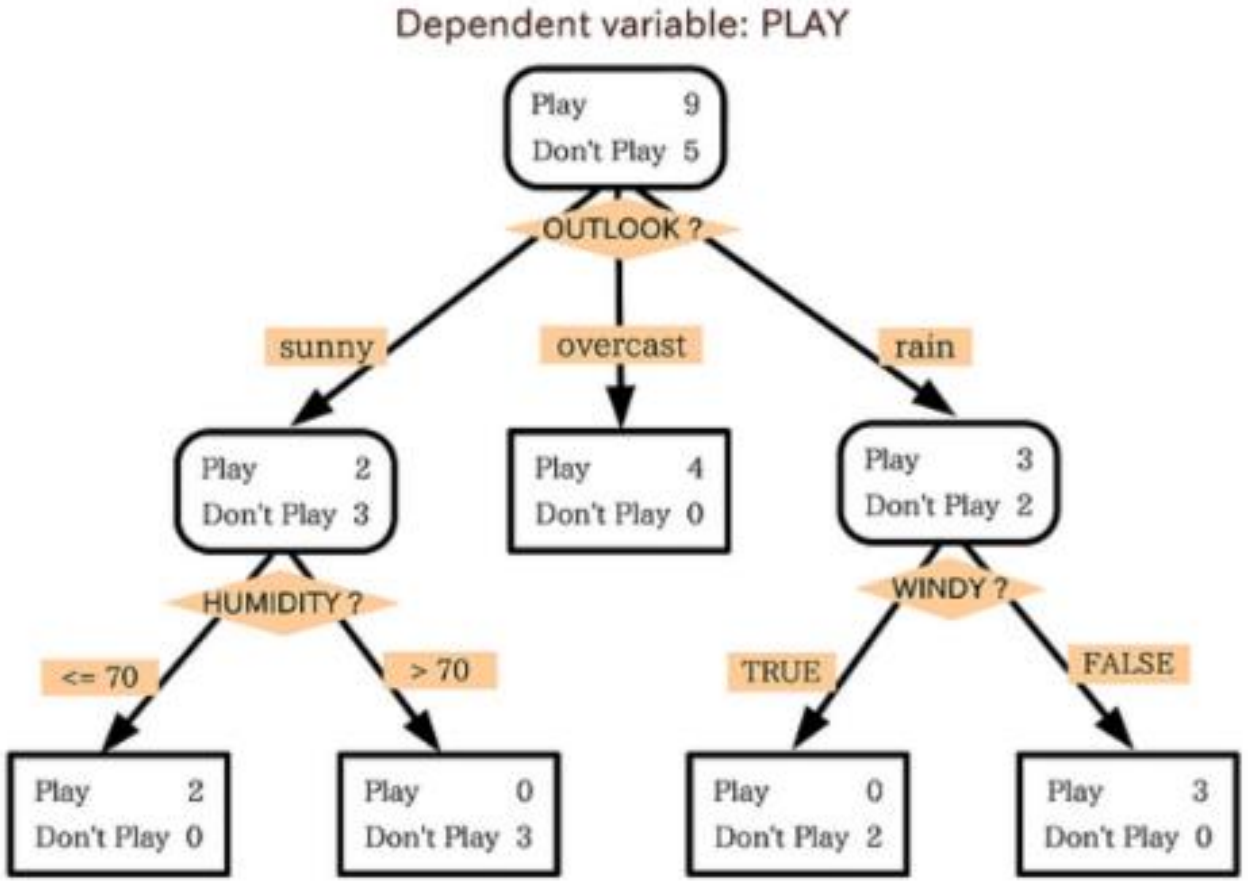
01. 의사결정나무

1) 의사결정나무의 개요

- 지도 학습 기법 중 한 가지.
- 적용 결과에 의해 if-then으로 표현되는 규칙 생성.
- 규칙의 이해가 쉽고 우수한 해석력.

1) 의사결정나무의 개요

◆ 의사결정나무의 예:



출처: https://ratsgo.github.io/machine%20learning/2017/03/26/tree/

2) 의사결정나무의 구성요소

- 뿌리마디 (Root node)
  - 나무구조가 시작되는 마디
- 자식마디 (Child node)
  - 하나의 마디로부터 분리된 2개 이상의 마디들
- 부모마디 (Parent node)
  - 주어진 마디의 상위 마디

2) 의사결정나무의 구성요소

- 끝마디 (Terminal or leaf node)
  - 자식마디가 없는 마디.
- 중간마디 (Internal node)
  - 부모마디와 자식마디가 모두 있는 마디.
- 가지 (Branch)
  - 뿌리마디로부터 끝마디까지 연결된 마디들
- 깊이 (Depth)
  - 뿌리마디로부터 끝마디까지 분리한 횟수

3) 의사결정나무의 특징

장점

- 이해하기 쉬운 규칙 (if-then) 이용해 생성된다.
- 연속형, 범주형 자료를 모두 다 취급할 수 있다.
- 이상치에 덜 민감하다.
- 모형의 가정 (예: 선형성, 등분산성 등)이 필요 없다.

3) 의사결정나무의 특징

단점

- 회귀 모형에서는 그 예측력이 떨어진다.
- 나무가 너무 깊은 경우에는 예측력이 나쁠 뿐만 아니라 해석 또한 쉽지 않다.
- 계산량이 많을 수 있다.
- 결과가 불안정하다.

09강. Decision Tree

02. 의사결정나무의 형성

1) 의사결정나무의 형성과정

- 나무의 성장 (Growing)
  - 각 마디에서 적절한 최적의 분리규칙 (splitting rule)을 찾아서 나무를 성장시킨다.
  - 정지규칙 (stopping rule)을 만족하면 성장을 중단한다.
- 가지치기 (Pruning)
  - 오분류율을 크게 할 위험이 높거나 부적절한 추론규칙을 가지고 있는 가지를 제거.
  - 불필요한 가지를 제거하는 과정.

1) 의사결정나무의 형성과정

타당성 평가

- 각 끝마디에 예측값을 할당
- 이익도표 (Lift chart), 검증 자료 (validation data)의 사용, 또는 교차 타당성 (cross-validation) 등을 이용하여 의사결정나무를 평가.

해석 및 예측

- 구축된 나무모형을 해석하고 예측.

2) 의사결정나무로 예측하기

- 입력값은 뿌리마디에서 출발
- 분리 조건에 따라 자식마디로 내려감.
- 끝마디에 도착할 때까지 계속 내려감.

2) 의사결정나무로 예측하기

회귀 문제일 경우

- 입력값이 도착한 끝마디에 속하는 모든 훈련자료 출력값의 평균으로 예측.

분류 문제일 경우

- 입력값이 도착한 끝마디에 속하는 모든 훈련자료 출력값의 최빈값으로 예측.

3) 분리 규칙

- 각 마디에서의 분리규칙
  - 입력 변수와 분리 기준을 정해야 함.

- 연속 변수의 경우
  - 변수 X와 분리 기준 c
  - 변수 X의 값이 c보다 작으면 왼쪽 자식마디, 크면 오른쪽 자식마디

3) 분리 규칙

- 범주형 변수의 경우
  - 전체 범주를 두 개의 부분집합으로 나눔.
  - 예: 전체 범주가 1,2,3,4일 때
    - 1,2,4 중 하나 -> 왼쪽 자식마디
    - 3 -> 오른쪽 자식마디

09강. Decision Tree

03. 의사결정나무 만들기

1) 분리 규칙의 선정

- 각 마디에서는 목표 변수의 분포를 가장 잘 구별해주는 변수와 분리 기준을 설정.
  - 불순도 (impurity)를 사용
  - 불순도를 최소화하는 방향
    - 생성된 두 개의 자식마디의 불순도의 합이 최소

1) 분리 규칙의 선정

- 불순도 측정량
- 분류 모형
  - 카이제곱 통계량 (Chi-square statistic)
  - 지니 지수 (Gini index)
  - 엔트로피 지수 (Entropy index)
- 회귀 모형
  - 분산 분석에 의한 F-통계량 (F-statistic)
  - 분산의 감소량

2) 카이 제곱 통계량

◆특정 분리 변수와 분리 기준에 의해 다음과 같이 노드를 분리했다고 하자.

	Good	Bad	Total
Left	32	48	80
Right	178	42	220
Total	210	90	300

2) 카이 제곱 통계량

◆ 앞의 표에서 각 셀에 대한 기대도수를 구할 수 있다.

	Good	Bad	Total
Left	80/300*210/300*300=54	80/300*90/300*300=24	80
Right	154	66	220
Total	210	90	300

2) 카이 제곱 통계량

- 실제 도수와 기대 도수를 이용.

$$\text{카이제곱통계량} = \sum \frac{(\text{기대도수} - \text{실제도수})^2}{\text{기대도수}}$$

- 앞의 표에서 카이제곱 통계량은 다음과 같다.

$$\frac{(56-32)^2}{56} + \frac{(24-48)^2}{24} + \frac{(154-178)^2}{154} + \frac{(66-42)^2}{66}$$
$$= 46.75$$

- 최소의 카이제곱통계량을 갖는 분리기준을 탐색.

3) 지니 지수

- 지니 지수는 다음과 같이 계산.

  지니지수
  $=$ 왼쪽에서 good일 확률 $*$ 왼쪽에서 bad일 확률
  $+$ 오른쪽에서 good일 확률 $*$ 오른쪽에서 bad일 확률

- 앞의 표에서 지니 지수는 다음과 같다.

$$\frac{32}{80} * \frac{48}{80} + \frac{178}{220} * \frac{42}{220} = 0.3944$$

- 최소의 지니 지수를을 갖는 분리기준을 탐색.

4) 엔트로피 지수

- 엔트로피 지수는 다음과 같이 계산.

  엔트로피 =
  = 왼쪽에서 good일 확률 * log(왼쪽에서 good일 확률)
  + 왼쪽에서 bad일 확률 * log(왼쪽에서 bad일 확률)
  + 오른쪽에서 good일 확률 * log(오른쪽에서 good일 확률)
  + 오른쪽에서bad일 확률 * log(오른쪽에서bad일 확률)

- 앞의 표에서 엔트로피를 구하면 약 0.4796
- 엔트로피를 가장 작게 하는 분리 기준을 탐색.

5) 분리 방법: 예제

- 아래 자료에서 지니 지수를 이용하여 최적의 분리를 찾아보자.

Temperature	Humidity	Windy	Class
Hot	High	False	N
Hot	High	True	N
Hot	High	False	P
Mild	High	False	P
Cool	Normal	False	P
Cool	Normal	True	N
Cool	Normal	True	P

Temperature	Humidity	Windy	Class
Mild	High	False	N
Cool	Normal	False	N
Mild	Normal	False	P
Mild	Normal	True	P
Mild	High	True	P
Hot	Normal	False	N
Mild	High	True	P

5) 분리 방법: 예제

- Temperature 변수를 기준으로 분리할 경우
  - Left node={Hot}, Right node={Mild, Cold}
  - 지니지수=3/4*1/4+3/10*7/10=0.3975

	N	P	Total
Left	3	1	4
Right	3	7	10
Total	6	8	14

5) 분리 방법: 예제

- Temperature 변수를 기준으로 분리할 경우
  - Left node={Mild}, Right node={Hot, Cold}
  - 지니지수=1/6*5/6+5/8*3/8=0.373

	N	P	Total
Left	1	5	6
Right	5	3	8
Total	6	8	14

5) 분리 방법: 예제

- Temperature 변수를 기준으로 분리할 경우
  - Left node={Cold}, Right node={Hot, Mild}
  - 지니지수=2/4*2/4+4/10*6/10=0.49

	N	P	Total
Left	2	2	4
Right	4	6	10
Total	6	8	14

5) 분리 방법: 예제

- Humidity 변수를 기준으로 분리할 경우
  - Left node={High}, Right node={Normal}
  - 지니지수=3/7*4/7+3/7*4/7=0.489

	N	P	Total
Left	3	4	7
Right	3	4	7
Total	6	8	14

5) 분리 방법: 예제

- Windy 변수를 기준으로 분리할 경우
  - Left node={False}, Right node={True}
  - 지니지수=4/8*4/8+2/6*4/6=0.472

	N	P	Total
Left	4	4	8
Right	2	4	6
Total	6	8	14

5) 분리 방법: 예제

- 가장 작은 지니 지수를 갖는 분리 기준은
  - Temperature 변수 사용
  - Left node={Mild}, Right node={Hot,Cold}
  - 지니 지수는 0.373

6) 회귀 모형에서의 불순도

- 왼쪽 자식 마디와 오른쪽 자식 마디의 평균의 차이를 검정하는 F-통계량의 유의 확률이 가장 작은 분리 변수와 분리 기준을 사용하여 분리를 수행.

- 왼쪽 자식 마디의 자료의 분산과 오른쪽 자식 마디의 자료의 분산의 합이 가장 작은 분리를 선택하여 분리를 수행.

7) 정지 규칙

- 현재의 마디가 더 이상 분리가 일어나지 못하게 하는 규칙.
- 대게 다음의 규칙 중 하나를 정지 규칙으로 사용한다.
  - 모든 자료가 한 그룹에 속할 때 (목표 변수가 범주형일 때에만 해당).
  - 마디에 속하는 자료가 일정 수 이하일 때.
  - 불순도의 감소량이 아주 작을 때.
  - 뿌리 마디로부터의 깊이가 일정 수 이상일 때.

8) 가지 치기

- 지나치게 많은 마디를 가지는 의사결정나무는 새로운 자료에 적용했을 때 예측 오차가 매우 클 가능성이 있다. -> 과적합

- 성장이 끝난 나무의 가지를 적당히 제거하여 적당한 크기를 갖는 나무 모형을 최종적인 예측모형으로 선택하는 것이 예측력의 향상에 도움이 된다.

8) 가지 치기

- 적당한 크기를 결정하는 방법은 평가용 자료를 사용하거나 교차 확인을 이용하여 예측에러를 구하고 이 예측에러가 가장 작은 나무 모형을 선택한다.

09강. Decision Tree

04. CART 알고리즘

1) CART

- Classification and Regression Tree
- 1984년 Breiman 과 그의 동료들이 발명
- 가장 널리 사용되는 의사결정나무 알고리즘

2) CART의 나무 성장

- 이진 분류 (Binary split) 을 이용.
- 분류 문제의 경우 불순도를 지니 지수를 이용하고 회귀 문제의 경우 분산을 이용.
- 각 마디의 자료의 수가 일정 수보다 작거나 불순도의 감소량이 일정 양 이하이면 성장을 정지.

3) CART의 가지치기

- 주어진 나무 T 와 양수 α 에 대해 비용 복잡도 (cost complexity) 를 다음과 같이 정의한다:

$$C_\alpha(T) = \text{나무}T\text{의 오분류율} + \alpha \cdot |T|$$

  - 여기서, $|T|$는 나무 T의 끝마디 개수.

- 나무성장과정을 통해 생성된 큰 나무 T_0에 대하여, 주어진 α 에 대해 C_α를 최소로 만드는 T_0의 부분나무를 $T(\alpha)$라 하자.

3) CART의 가지치기

- α를 0에서 시작해서 계속 증가시키면서 대응되는 나무 $T(\alpha)$들을 찾아나간다.
- 평가용 자료나 교차 확인 방법을 이용하여 $T(\alpha)$의 오분류율을 계산한다.
- 이 중에서 오분류율이 가장 작은 나무를 최종 나무 모형으로 선택한다.

09강. Decision Tree

05. Python을 이용한 실습

1) 데이터 설명

- Penguins 데이터셋
  - 남극 펭귄 344마리에 대한 데이터
  - species: 펭귄 종류 (총 3가지)
  - island: 서식하는 남극섬 종류
  - bill_length_mm: culmen length (mm)
  - bill_depth_mm: bill_depth (mm)
  - flipper_length_mm: 물갈퀴 길이 (mm)
  - body_mass_g: 몸무게 (g)
  - sex: 성별

1) 데이터 설명

- 의사결정나무를 이용하여 펭귄의 종을 예측하는 분류 모형을 만들자.

2) 환경설정

◆ 필요한 패키지 불러오기

```
import os
import pandas as pd
import numpy as np
import sklearn
from sklearn.tree import DecisionTreeClassifier
from sklearn.tree import export_text, export_graphviz
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns
```

3) 데이터 불러오기

데이터 불러오기

```
penguins = sns.load_dataset('penguins')
print(penguins.shape)
penguins.head()
```

```
(344, 7)
```

	species	island	bill_length_mm	bill_depth_mm
0	Adelie	Torgersen	39.1	18.7
1	Adelie	Torgersen	39.5	17.4
2	Adelie	Torgersen	40.3	18.0

4) 데이터 전처리

- sex: 남성은 0, 여성은 1로 변환
- island: 3개의 범주값을 가짐
  - Torgersen, Biscoe, Dream
  - Biscoe, Dream에 대한 가변수 생성
- 수치형 변수에 대해서 평균값으로 결측값 대체

4) 데이터 전처리

```
penguins['bill_length_mm'].fillna(value=penguins['bill_length_mm'].mean(), inplace=True)
penguins['bill_depth_mm'].fillna(value=penguins['bill_depth_mm'].mean(), inplace=True)
penguins['flipper_length_mm'].fillna(value=penguins['flipper_length_mm'].mean(), inplace=True)
penguins['body_mass_g'].fillna(value=penguins['body_mass_g'].mean(), inplace=True)

penguins['sex'] =penguins['sex'].apply(lambda x: 1 if x == 'MALE' else 0)
penguins['Biscoe'] = penguins['island'].apply(lambda x: 1 if x == 'Biscoe' else 0)
penguins['Dream'] = penguins['island'].apply(lambda x: 1 if x == 'Dream' else 0)
```

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex	Biscoe	Dream
0	Adelie	Torgersen	39.10000	18.70000	181.000000	3750.000000	1	0	0
1	Adelie	Torgersen	39.50000	17.40000	186.000000	3800.000000	0	0	0
2	Adelie	Torgersen	40.30000	18.00000	195.000000	3250.000000	0	0	0
3	Adelie	Torgersen	43.92193	17.15117	200.915205	4201.754386	0	0	0
4	Adelie	Torgersen	36.70000	19.30000	193.000000	3450.000000	0	0	0

4) 데이터 전처리

데이터 분할하기

```
colnames = ['bill_length_mm','bill_depth_mm',
            'flipper_length_mm','body_mass_g','sex',
           'Biscoe','Dream']
X = penguins[colnames]
y = penguins.iloc[:,0]
```

```
X_train, X_test, y_train, y_test = \
train_test_split(X, y, test_size = 0.3, random_state=123)
```

5) 의사결정나무 적합하기

- 범주형 불순도는 지니 지수
- 최대 깊이는 3

```
pen_tree = DecisionTreeClassifier(criterion = 'gini',
                                  max_depth = 3,
                                  random_state = 0).fit(X_train, y_train)
```

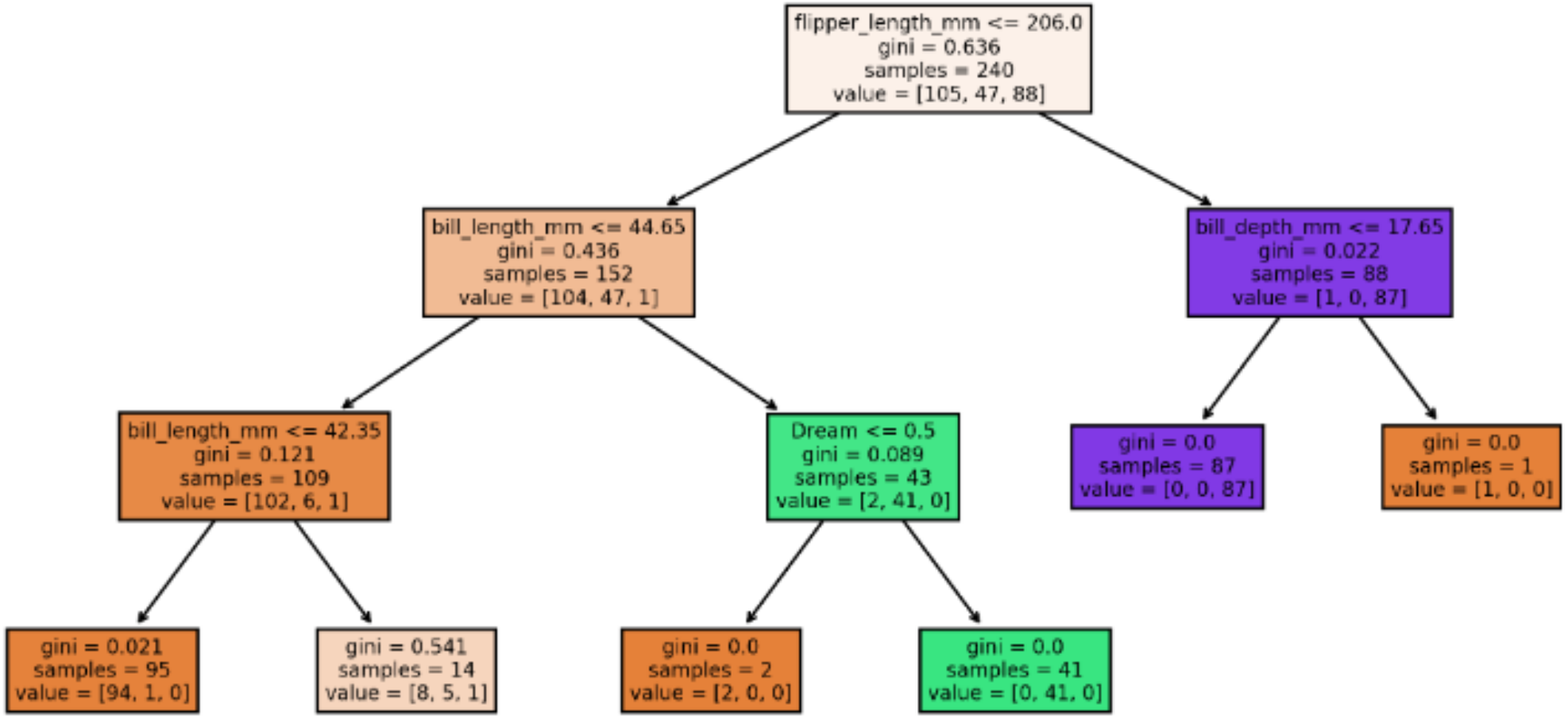
5) 의사결정나무 적합하기

◆ 적합한 나무 시각화하기

```
fig, axes = plt.subplots(nrows = 1, ncols = 1, figsize = (10,5), dpi = 300)
plotResult = sklearn.tree.plot_tree(pen_tree,
                                    feature_names = colnames,
                                    filled = True)
```

5) 의사결정나무 적합하기

적합한 나무 시각화하기

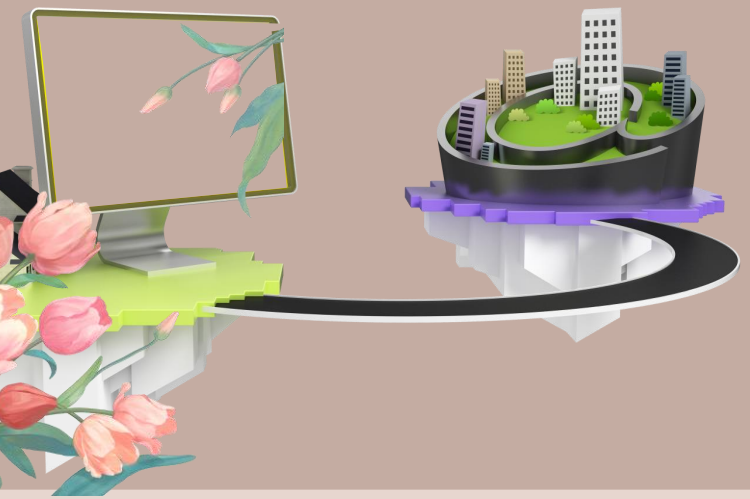


5) 의사결정나무 적합하기

◆ 적합한 나무의 성능 확인하기

```
print(pen_tree.score(X_test, y_test))
pred_y = pen_tree.predict(X_test)
print(confusion_matrix(y_test, pred_y))
```

```
0.9423076923076923
[[47  0  0]
 [ 5 16  0]
 [ 1  0 35]]
```



다음시간안내

제10강

Ensemble Learning 1.