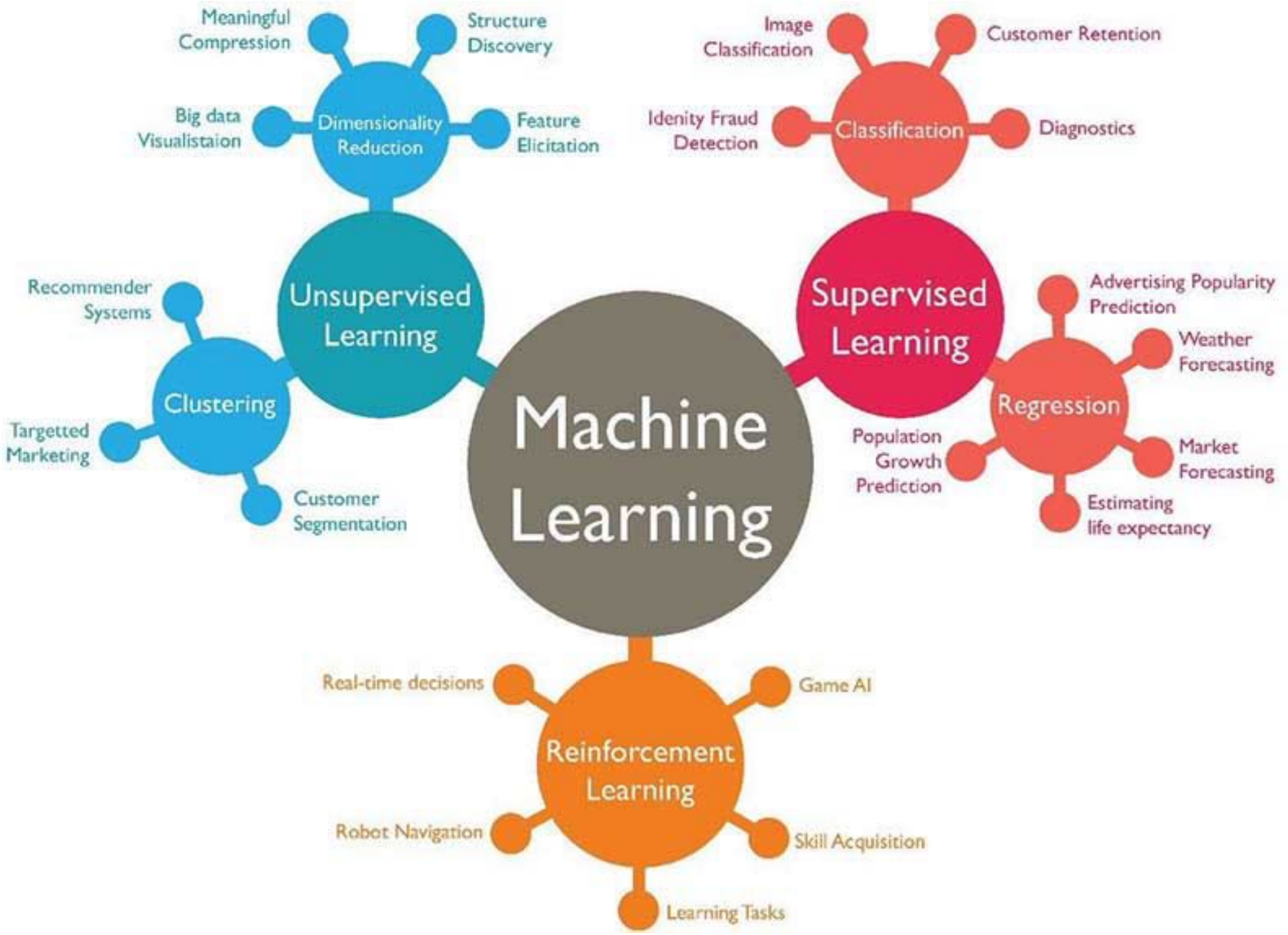
Machine Learning

Intro

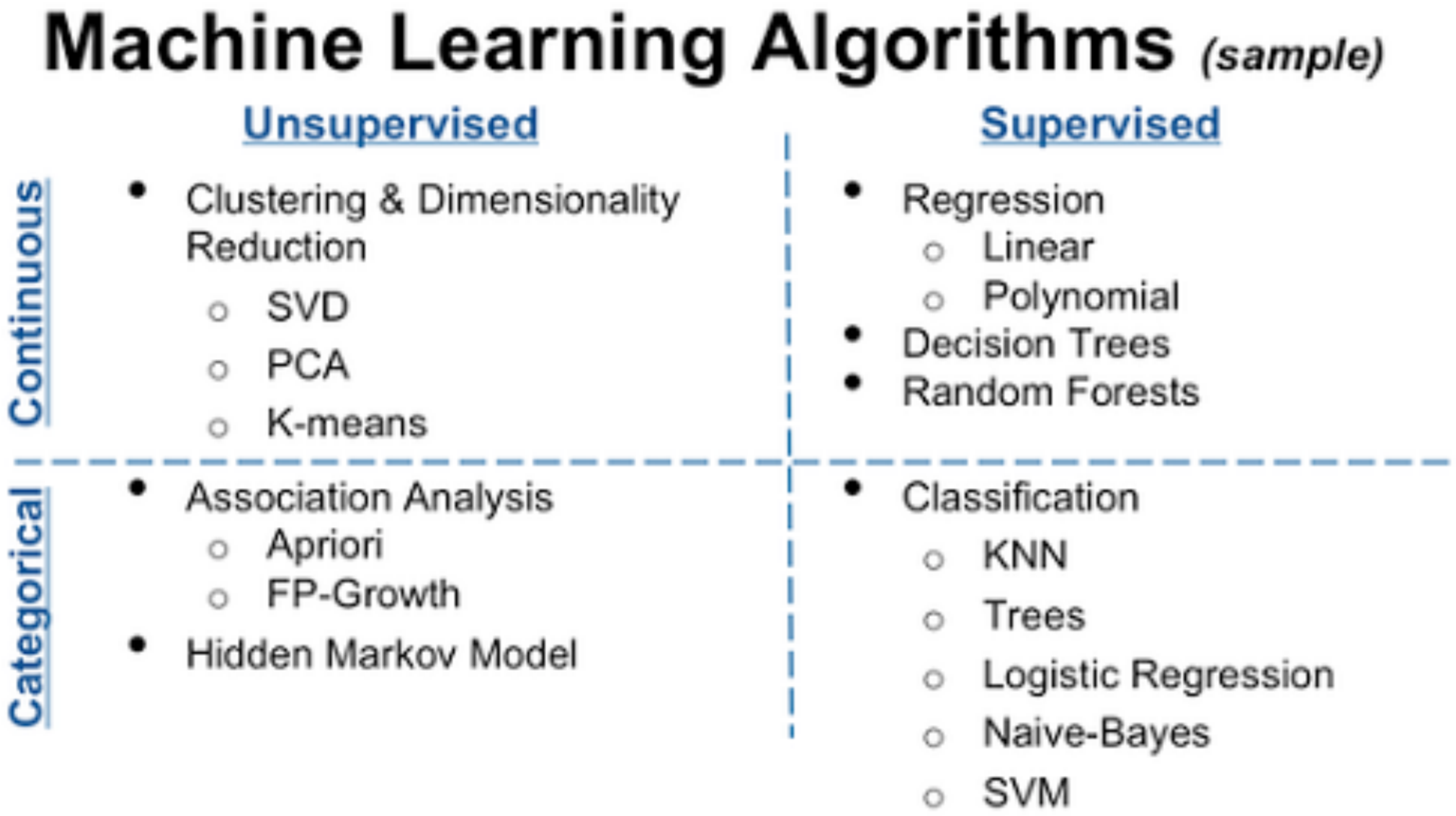
Machine Learning?

기계(즉 컴퓨터)를 인간처럼 학습시켜 스스로 규칙을 형성하고자 하는 시도에서 시작되었다



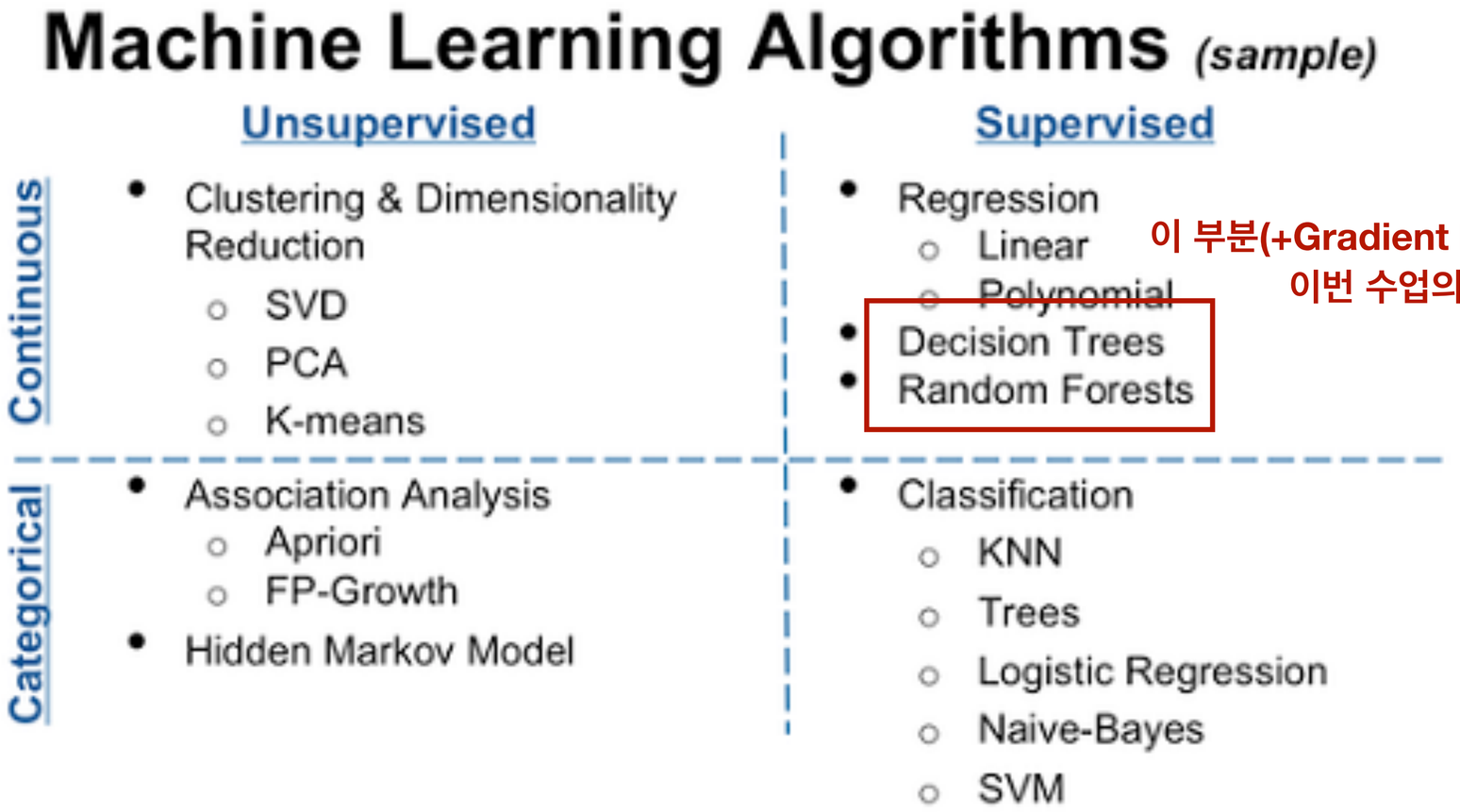
Machine Learning의 분류

크게 Supervised Learning과 Unsuprvised Learning으로 나뉘며,
이 중 우리는 Supervised Learning, 그 중에서도 Tree 기반의 알고리즘을 집중적으로 다룰 것이다



Machine Learning의 분류

크게 Supervised Learning과 Unsuprvised Learning으로 나뉘며,
이 중 우리는 Supervised Learning, 그 중에서도 Tree 기반의 알고리즘을 집중적으로 다룰 것이다



이 부분(+Gradient Boosting Machine)이
이번 수업의 핵심이 될 예정

Structured vs Unstructured Dataset

데이터는 구조화된(Strurctured) 데이터와 비구조화된 데이터(Unstructured)로 나뉘어지며,
우리는 이 중 구조화된 데이터에서 가장 좋은 성능이 나오는 알고리즘을 중점적으로 다룰 것이다.

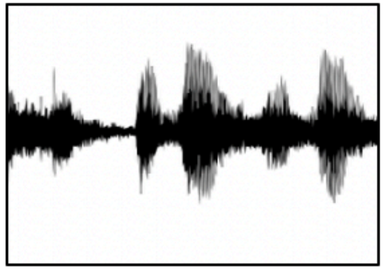
Supervised Learning

Structured Data

Size	#bedrooms	···	Price (1000$s)
2104	3		400
1600	3		330
2400	3		369
⋮	⋮		⋮
3000	4		540

User Age	Ad Id	···	Click
41	93242		1
80	93287		0
18	87312		1
⋮	⋮		⋮
27	71244		1

Unstructured Data



Audio

Image

Four scores and seven years ago...

Text

Coursera's Neural Networks and Deep Learning (Andrew Ng)

앞으로의 스케쥴

가장 기본적인 알고리즘인 Decision Tree부터 시작하여,
가장 최신의 머신러닝 알고리즘과, 이를 실전에 활용하는 방법까지 모두 다루게 될 것.

Lecture 1: Decision Tree

- Decision Tree의 기본적인 원리
- 이를 파이썬으로 직접 구현하기
- Decision Tree에 있는 Hyperparameter를 배우기

Lecture 2: Ensemble Methods

- Decision Tree의 성능을 올릴 수 있는 앙상블 방식을 배우기
- 배깅(Bagging, Bootstrap Aggregating)를 직접 구현하기
- 그래디언트 부스팅(Gradient Boosting, GBM)을 직접 구현하기

Lecture 3: Practical usage of GBM

- GBM을 가장 실용적으로 사용할 수 있는 여러 툴을 리뷰하기
- 이 중 LightGBM의 사용 방법을 숙지하기
- LightGBM의 주요 하이퍼패러미터를 살펴보기

Lecture 4: Otto Group Product Classification Challenge

- LightGBM을 이용하여 상품을 분류하는 캐글 경진대회에 참여하기
- 순수 Hyperparameter Tuning만으로 상위 10%에 근접하기

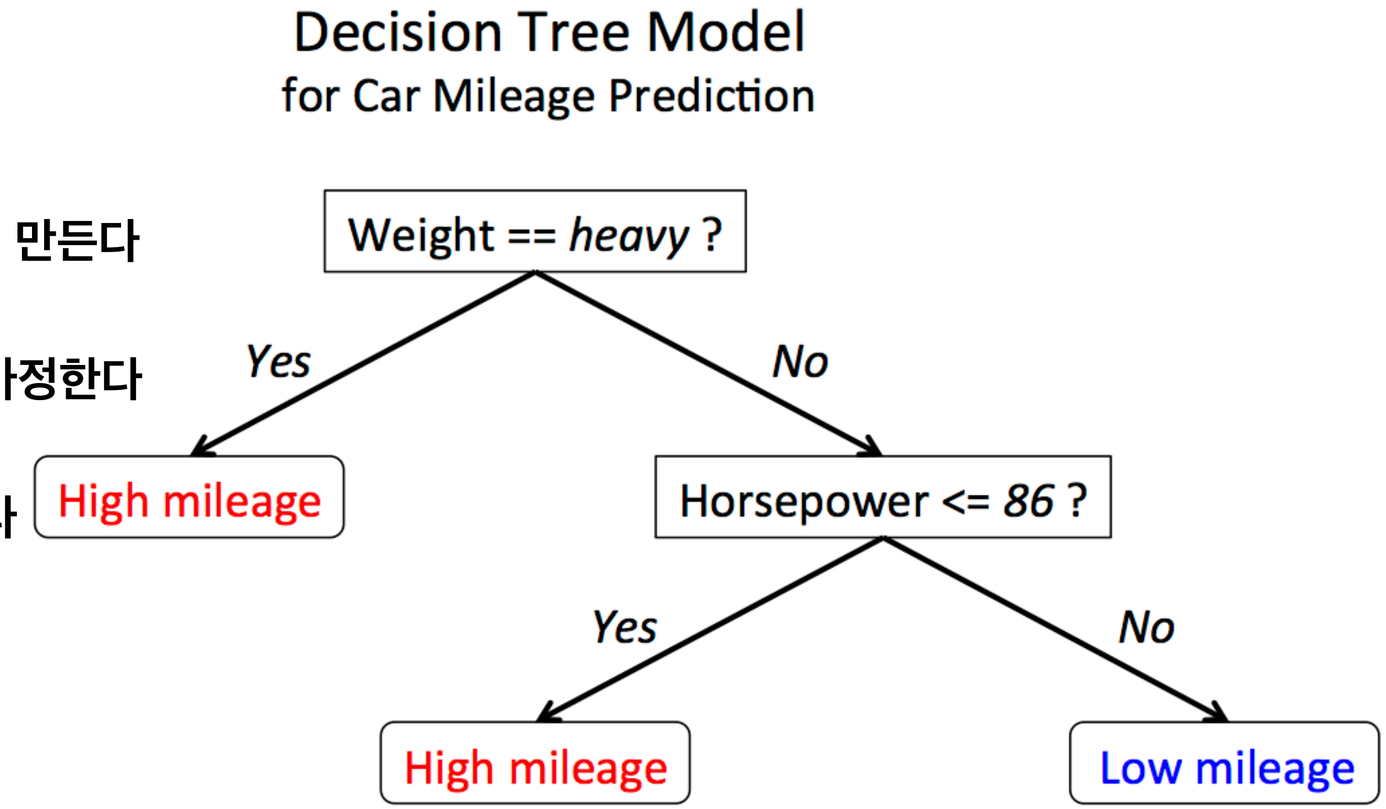
Decision Tree

Decision Tree

가장 기본적인 머신러닝 알고리즘 중 하나
사람이 가장 이해하기 쉬운 머신러닝 알고리즘 중 하나이며, 그 성능도 좋은 편에 속한다

동작 원리

1. feature와 label를 받는다
2. feature를 활용해 만들 수 있는 모든 조건(condition)을 만든다
3. 모든 조건마다의 평균 gini impurity를 구한다
4. 평균 gini impurity가 가장 낮은 조건을 좋은 조건이라 가정한다
5. 이 조건을 활용해 나무의 가지를 친다
6. 이를 gini impurity가 더 낮아지지 않을 때 까지 반복한다

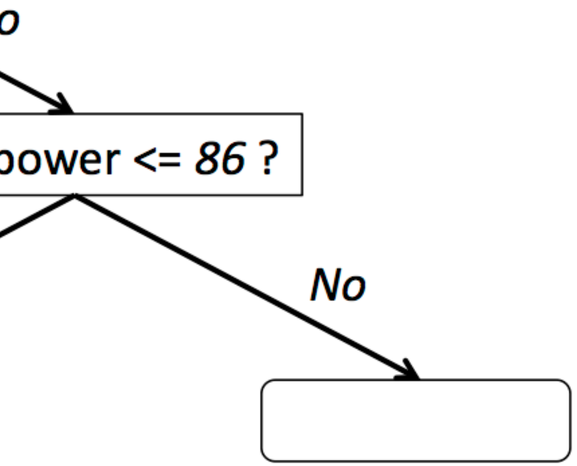


$$I_G(p) = p_t(1 - p_t) + p_f(1 - p_f)$$

gini impurity
(차후에 자세히 설명)

Gini Impurity

데이터의 불순도(impurity)를 측정하기 위한 공식이며,
데이터를 가르는 조건이 좋은 조건인지 좋지 않은 조건인지 판가름하는 기준이 되는 공식이다



이 조건에 100개의 데이터가 있을 경우

100명이 True, 0명이 False인 경우라면 (굉장히 좋은 경우)

$$I_G(p) = p_t(1-p_t) + p_f(1-p_f) = 1.0 * (1-1.0) + 0.0 * (1-0.0) = 0.0$$

50명이 True, 50명이 False인 경우라면 (굉장히 안 좋은 경우)

$$I_G(p) = p_t(1-p_t) + p_f(1-p_f) = 0.5 * (1-0.5) + 0.5 * (1-0.5) = 0.5$$

Gini Impurity는 0에 가까울수록 좋은 조건이며,
0.5에 가까울수록 좋지 않은 조건이라 볼 수 있다.

Decision Tree의 동작 원리

Feature마다의 Gini Impurity를 계산 후
이 Gini Impurity가 좋을 수록(=낮을 수록) 더 좋은 Feature(내지는 조건)이라고 가정한다

name	짱절미	셀스타그램	우산	like
Kang	True	False	False	True
Kim	False	False	False	False
Choi	False	True	False	True
Park	False	False	False	True
Yoon	False	False	False	False

보유하고 있는 데이터

Decision Tree의 동작 원리

Feature마다의 Gini Impurity를 계산 후
이 Gini Impurity가 좋을 수록(=낮을 수록) 더 좋은 Feature(내지는 조건)이라고 가정한다

Features **Label**

name	짱절미	셀스타그램	우산	like
Kang	True	False	False	True
Kim	False	False	False	False
Choi	False	True	False	True
Park	False	False	False	True
Yoon	False	False	False	False

보유하고 있는 데이터

Decision Tree의 동작 원리

Feature마다의 Gini Impurity를 계산 후
이 Gini Impurity가 좋을 수록(=낮을 수록) 더 좋은 Feature(내지는 조건)이라고 가정한다

Features **Label**

name	짱절미	셀스타그램	우산	like
Kang	True	False	False	True
Kim	False	False	False	False
Choi	False	True	False	True
Park	False	False	False	True
Yoon	False	False	False	False

보유하고 있는 데이터

```
짱절미 == True

셀스타그램 == True

우산 == True
```

Feature를 활용해
가능한 모든 조건(condition)을 만든다

Decision Tree의 동작 원리

Feature마다의 Gini Impurity를 계산 후
이 Gini Impurity가 좋을 수록(=낮을 수록) 더 좋은 Feature(내지는 조건)이라고 가정한다

Features **Label**

name	짱절미	셀스타그램	우산	like
Kang	True	False	False	True
Kim	False	False	False	False
Choi	False	True	False	True
Park	False	False	False	True
Yoon	False	False	False	False

보유하고 있는 데이터

짱절미 == True

셀스타그램 == True

우산 == True

Feature를 활용해
가능한 모든 조건(condition)을 만든다

```
condition = make_condition(binary_condition, "짱절미", Tru
gini_impurity = evaluate_average_gini_impurity(data, con

"짱절미 == True", f"{gini_impurity:.6f}"
```

```
('짱절미 == True', '0.400000')
```

```
condition = make_condition(binary_condition, "셀스타그램",
gini_impurity = evaluate_average_gini_impurity(data, con

"셀스타그램 == True", f"{gini_impurity:.6f}"
```

```
('셀스타그램 == True', '0.400000')
```

```
condition = make_condition(binary_condition, "우산", True
gini_impurity = evaluate_average_gini_impurity(data, con

"우산 == True", f"{gini_impurity:.6f}"
```

```
('우산 == True', '0.480000')
```

각 조건마다의 평균 gini impurity를 구한다.
(해당 조건이 참일때, 거짓일때 gini impurity의 평균)

Decision Tree의 동작 원리

Feature마다의 Gini Impurity를 계산 후
이 Gini Impurity가 좋을 수록(=낮을 수록) 더 좋은 Feature(내지는 조건)이라고 가정한다

Features **Label**

name	짱절미	셀스타그램	우산	like
Kang	True	False	False	True
Kim	False	False	False	False
Choi	False	True	False	True
Park	False	False	False	True
Yoon	False	False	False	False

보유하고 있는 데이터

짱절미 == True

셀스타그램 == True

우산 == True

Feature를 활용해
가능한 모든 조건(condition)을 만든다

```
condition = make_condition(binary_condition, "짱절미", Tru
gini_impurity = evaluate_average_gini_impurity(data, con

"짱절미 == True", f"{gini_impurity:.6f}"
```

```
('짱절미 == True', '0.400000')
```

```
condition = m                                    셀스타그램",
gini_impurity                                    (data, con

"셀스타그램 == True", f"{gini_impurity:.6f}"
```

두 조건 모두 Gini Impurity가 0.4로 동일하게 좋음

```
('셀스타그램 == True', '0.400000')
```

```
condition = make_condition(binary_condition, "우산", True
gini_impurity = evaluate_average_gini_impurity(data, con

"우산 == True", f"{gini_impurity:.6f}"
```

```
('우산 == True', '0.480000')
```

각 조건마다의 평균 gini impurity를 구한다.
(해당 조건이 참일때, 거짓일때 gini impurity의 평균)

Decision Tree의 동작 원리

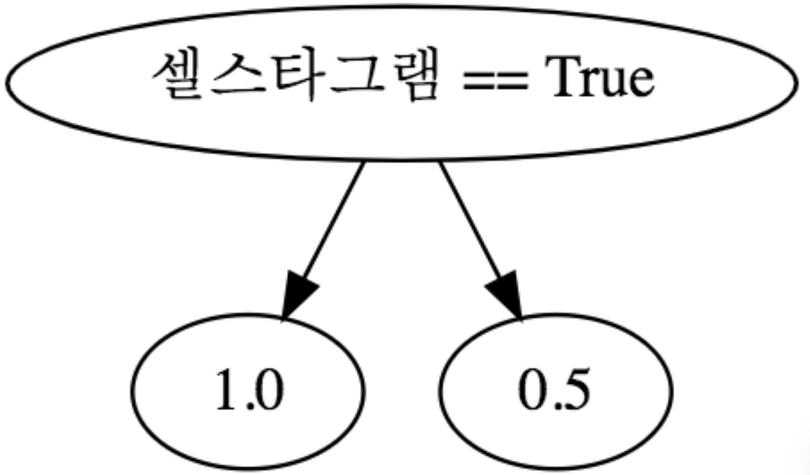
Feature마다의 Gini Impurity를 계산 후
이 Gini Impurity가 좋을 수록(=낮을 수록) 더 좋은 Feature(내지는 조건)이라고 가정한다

Label

name	짱절미	셀스타그램	우산	like
Kang	True	False	False	True
Kim	False	False	False	False
Choi	False	True	False	True
Park	False	False	False	True
Yoon	False	False	False	False

보유하고 있는 데이터

Gini Impurity가 가장 낮은(==0.4)
“셀스타그램 == True” 조건으로 나눈다



Decision Tree의 동작 원리

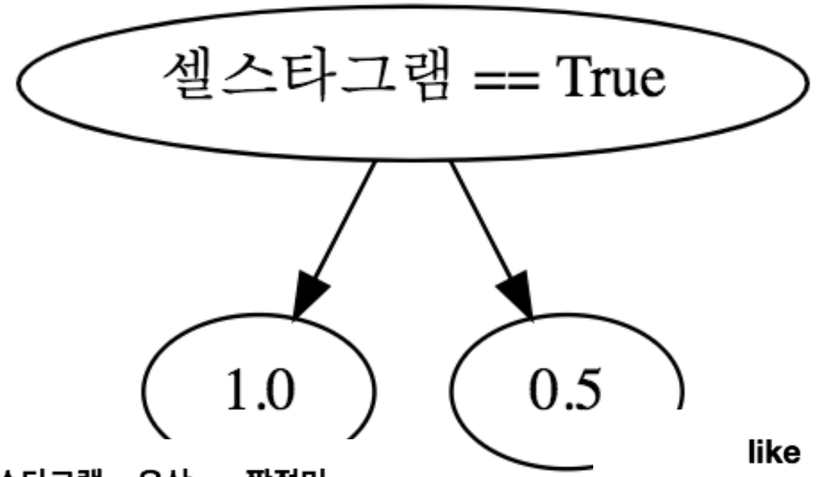
Feature마다의 Gini Impurity를 계산 후
이 Gini Impurity가 좋을 수록(=낮을 수록) 더 좋은 Feature(내지는 조건)이라고 가정한다

Label

name	짱절미	셀스타그램	우산	like
Kang	True	False	False	True
Kim	False	False	False	False
Choi	False	True	False	True
Park	False	False	False	True
Yoon	False	False	False	

보유하고 있는 데이터

Gini Impurity가 가장 낮은(==0.4)
"셀스타그램 == True" 조건으로 나눈다



name	like	셀스타그램	우산	짱절미
Choi	True	True	False	False

name	like	셀스타그램	우산	짱절미
Kang	True	False	False	True
Kim	False	False	False	False
Park	True	False	False	False
Yoon	False	False	False	False

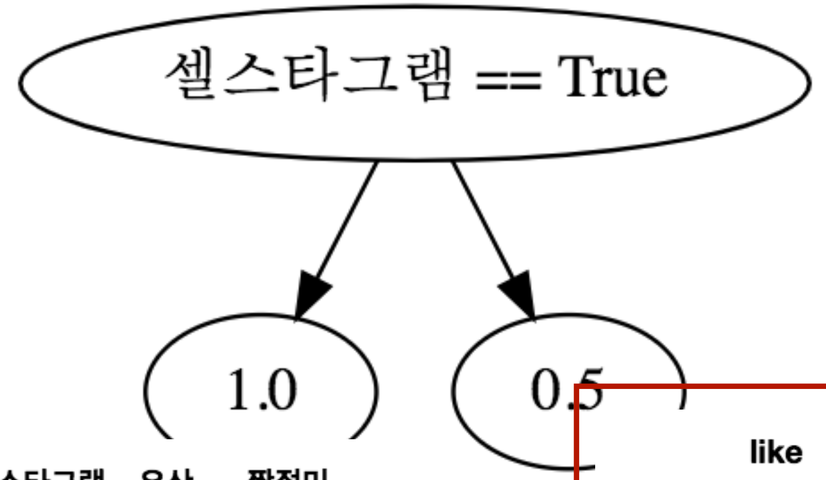
Decision Tree의 동작 원리

Feature마다의 Gini Impurity를 계산 후
이 Gini Impurity가 좋을 수록(=낮을 수록) 더 좋은 Feature(내지는 조건)이라고 가정한다

Label

name	짱절미	셀스타그램	우산	like
Kang	True	False	False	True
Kim	False	False	False	False
Choi	False	True	False	True
Park	False	False	False	True
Yoon	False	False	False	

보유하고 있는 데이터



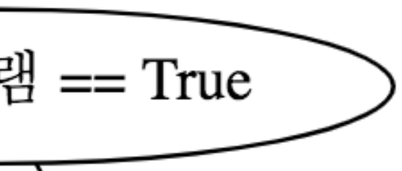
name	like	셀스타그램	우산	짱절미
Choi	True	True	False	False

오른쪽 데이터는 한 번 더 나눌 수 있다

name	like	셀스타그램	우산	짱절미
Kang	True	False	False	True
Kim	False	False	False	False
Park	True	False	False	False
Yoon	False	False	False	False

Decision Tree의 동작 원리

Feature마다의 Gini Impurity를 계산 후
이 Gini Impurity가 좋을 수록(=낮을 수록) 더 좋은 Feature(내지는 조건)이라고 가정한다

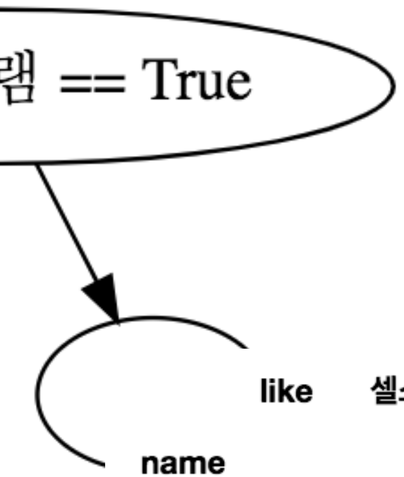


name	like	셀스타그램	우산	짱절미
Kang	True	False	False	True
Kim	False	False	False	False
Park	True	False	False	False
Yoon	False	False	False	False

트리의 우측 데이터

Decision Tree의 동작 원리

Feature마다의 Gini Impurity를 계산 후
이 Gini Impurity가 좋을 수록(=낮을 수록) 더 좋은 Feature(내지는 조건)이라고 가정한다



name	like	셀스타그램	우산	짱절미
Kang	True	False	False	True
Kim	False	False	False	False
Park	True	False	False	False
Yoon	False	False	False	False

트리의 우측 데이터

```
condition = make_condition(binary_condition, "짱절미",
gini_impurity = evaluate_average_gini_impurity(right_

"짱절미 == True", f"{gini_impurity:.6f}"
```

```
('짱절미 == True', '0.333333')
```

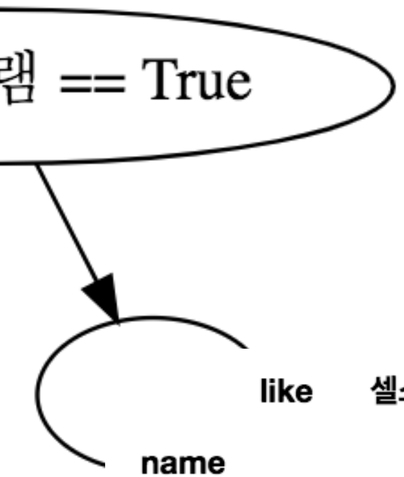
```
condition = make_condition(binary_condition, "우산", T
gini_impurity = evaluate_average_gini_impurity(right_

"우산 == True", f"{gini_impurity:.6f}"
```

```
('우산 == True', '0.500000')
```

우측 데이터의 평균 Gini Impurity를 구한다

Decision Tree의 동작 원리

Feature마다의 Gini Impurity를 계산 후
이 Gini Impurity가 좋을 수록(=낮을 수록) 더 좋은 Feature(내지는 조건)이라고 가정한다



name	like	셀스타그램	우산	짱절미
Kang	True	False	False	True
Kim	False	False	False	False
Park	True	False	False	False
Yoon	False	False	False	False

트리의 우측 데이터

```
condition = make_condition(binary_condition, "짱절미",
gini_impurity = evaluate
```

짱절미 == True 조건이 Gini Impurity가 높다

```
"짱절미 == True", f"{gini_impurity:.6f}"
```

```
('짱절미 == True', '0.333333')
```

```
condition = make_condition(binary_condition, "우산", T
gini_impurity = evaluate_average_gini_impurity(right_

"우산 == True", f"{gini_impurity:.6f}"
```

```
('우산 == True', '0.500000')
```

우측 데이터의 평균 Gini Impurity를 구한다

Decision Tree의 동작 원리

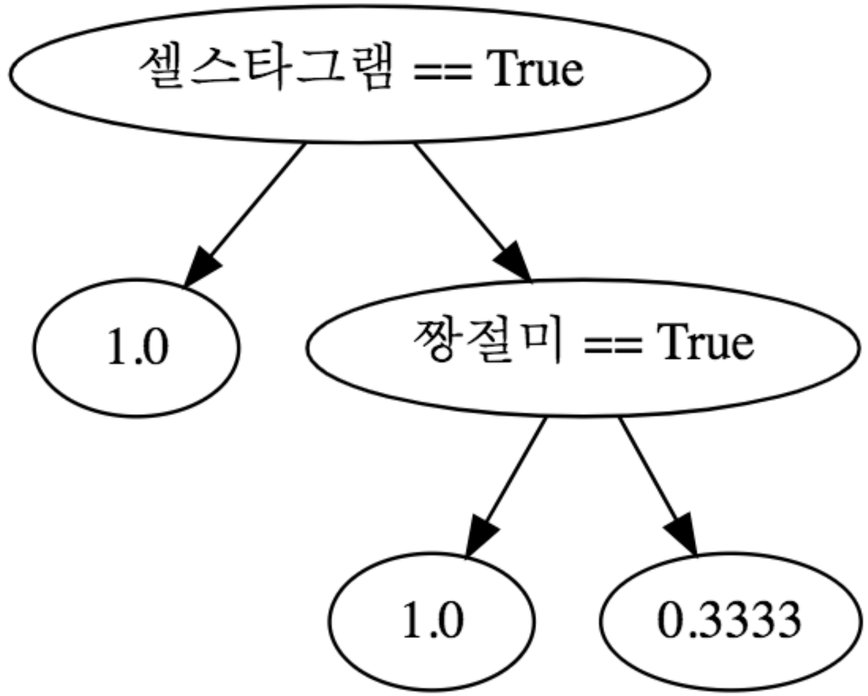
Feature마다의 Gini Impurity를 계산 후
이 Gini Impurity가 좋을 수록(=낮을 수록) 더 좋은 Feature(내지는 조건)이라고 가정한다

```
condition = make_condition(binary_condition, "짱절미",
gini_impurity = evaluate
"짱절미 == True", f"{gini_impurity:.6f}"
('짱절미 == True', '0.333333')
```

짱절미 == True 조건이 Gini Impurity가 높다

```
condition = make_condition(binary_condition, "우산", T
gini_impurity = evaluate_average_gini_impurity(right_

"우산 == True", f"{gini_impurity:.6f}"
('우산 == True', '0.500000')
```

우측 데이터의 평균 Gini Impurity를 구한다



조건을 추가하여 Tree를 구축한다

Q & A