

# Kuggle

2021\_02 Kuggle 겨울세션\_W1

2021.12.28

이준희, 김현우



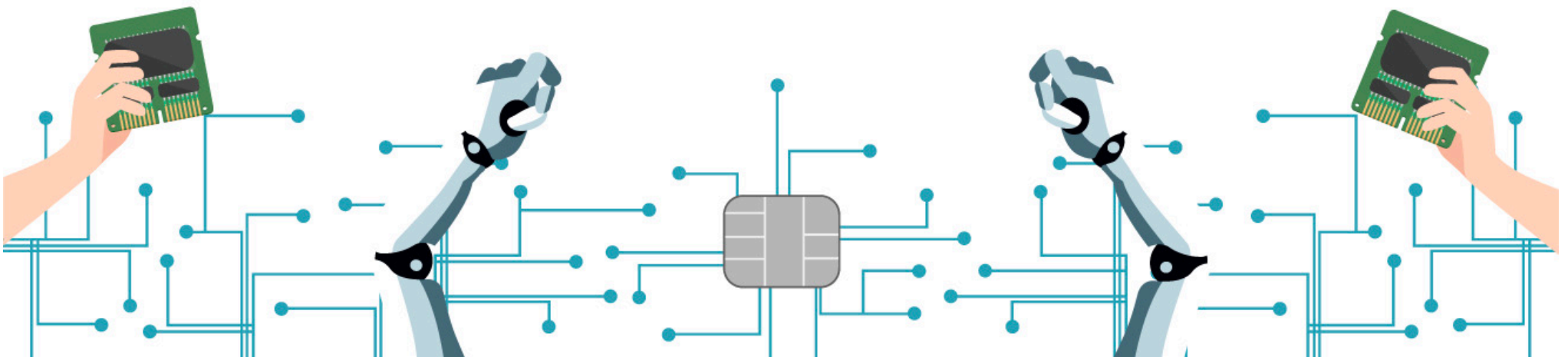
## CONTENTS

**01** ML

**02** Scikit Learn

**03** Model Selection

**04** Preprocessing



## CONTENTS

# 01. ML

- 도대체 ML이란 무엇인가
- 기본 Flow

# 01 ML

도대체 ML이란 무엇인가



## Machine Learning => " 기계 학습 "

- 1) 기계가 학습한다
- 2) 기계가 어떻게 학습을 하는가?
- 3) 주어진 데이터를 기반으로 학습을 한다
- 4) 그러니까 어떻게 학습을 하는가?
- 5) 규칙 기반 / 선형 경계 / 확률 기반 알고리즘을 이용한다

# 01 ML

Flow



1) 데이터 준비 → 데이터 수집, 데이터 전처리

2) 데이터 세트 분리 → 학습, 검증, 평가

3) 모델 학습 → 알고리즘 선택 및 학습 과정

4) 평가 → 결과에 대한 해석

**\*\*기본 흐름이 너무 쉽게 이해되는 것이 장점이자 단점**

-> 학습하는 입장에서 내가 제대로 배운 것이 맞는지 의문이 생김

-> 단계, 단계 마다 가미할 수 있는 기술이 무궁무진함

	키	몸무게	흡연유무	예상수명	
현우	5.5ft	60kg	yes	66	$5.5x_1 + 60x_2 + yes \cdot x_3$
준희	6.0ft	55kg	yes	78	$6.0x_1 + 55x_2 + yes \cdot x_3$
국준	5.0ft	65kg	no	74	$5.0x_1 + 65x_2 + no \cdot x_3$

$$\begin{bmatrix} 5.5 & 60 & yes \\ 6.0 & 55 & yes \\ 5.0 & 65 & no \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 66 \\ 78 \\ 74 \end{bmatrix} \quad \begin{bmatrix} 5.5 \\ 6.0 \\ 5.0 \end{bmatrix} x_1 + \begin{bmatrix} 60 \\ 55 \\ 65 \end{bmatrix} x_2 + \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} x_3$$

$$life-span = (-0.4) \cdot weight + 20 \cdot height + (-20) \cdot is\_smoking$$

- 1) 흡연 유무는 yes / no 인데 어떻게 계산을 할 수 있을까? => Encoding
- 2) 몸무게의 절대값이 키보다 훨씬 큰데 정보량을 해석하는데 영향을 끼치지 않을까? => Scailing
- 3) 키가 크면 몸무게가 많이 나가는 상관성이 있지 않을까? => Reduce Dimension
- 4) 데이터가 3개 밖에 없는데 너무 적은건 아닐까? => Crawling

## CONTENTS

# 02. Scikit Learn

- Scikit Learn
- Pytorch, Tensorflow

# 01 Scikit Learn

Scikit Learn



-> 앞에서 말한 일련의 과정들을 도와주는 라이브러리

-> 기본 골자는 fit()과 predict()

-> 이외에도 dataset 제공, feature 처리, algorithm, evaluation 등등의  
기능

\*\* 자세한 내용은 [https://scikit-learn.org/stable/user\\_guide.html](https://scikit-learn.org/stable/user_guide.html) 참조



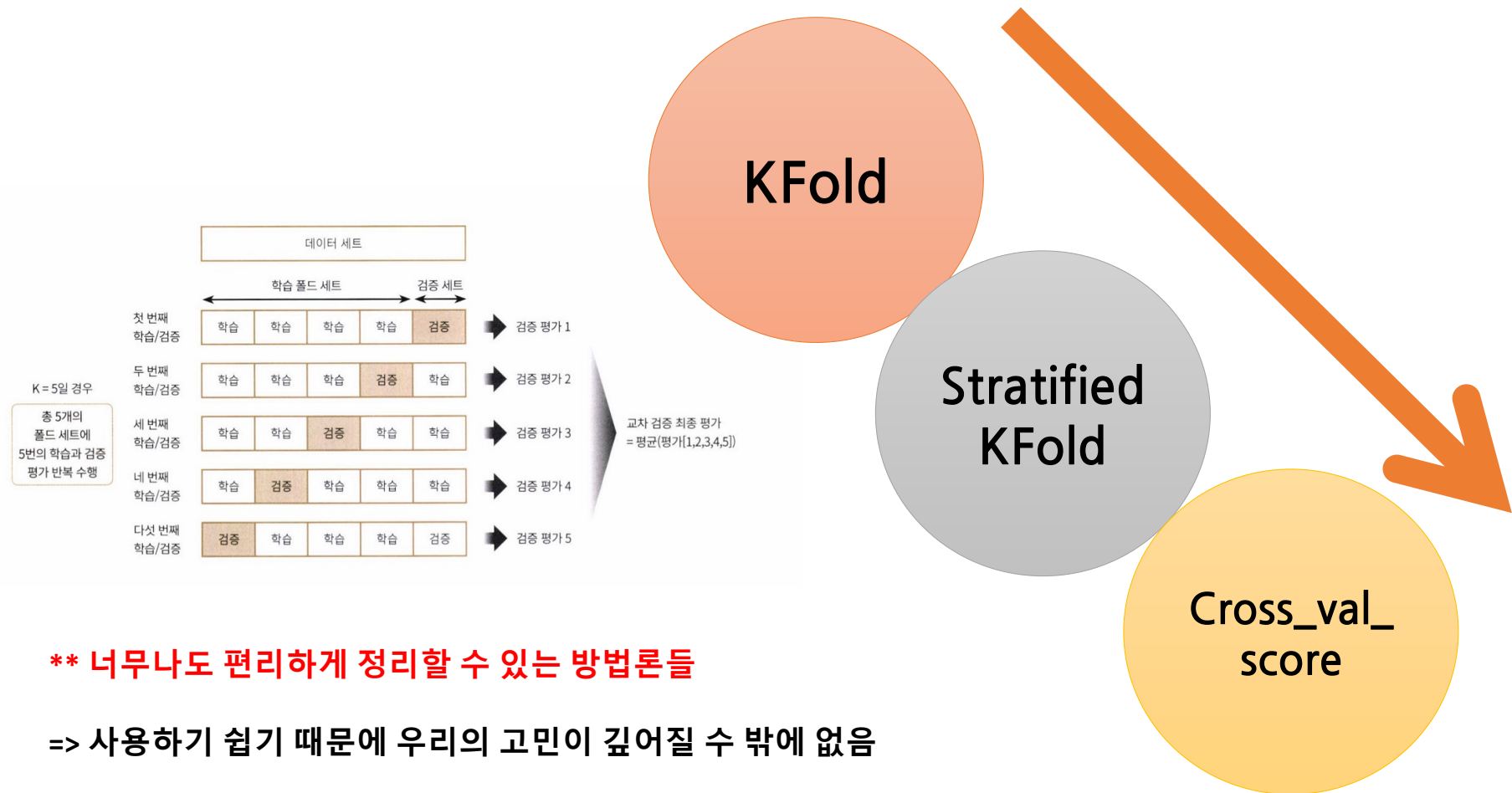


## CONTENTS

# 03. Model Selection

- `Train_test_split()`
- Cross Validation
- `GridSearchCV()`

```
1  import pandas as pd
2  import numpy as np
3
4  def cs_train_test_split(X, y, test_size, random_state):
5
6      df = pd.concat([pd.DataFrame(X), pd.DataFrame(y)], axis = 1)
7
8      test_len = round(len(df)*test_size)
9
10     if(random_state <= len(df) - test_len):
11         test_df = df[random_state : random_state + test_len]
12         train_df = df[:random_state] + df[random_state+test_len:]
13     else:
14         random_state = 0
15         df.sample(frac = 1)
16
17     return train_df[:-1], test_df[:-1], train_df[-1], test_df[-1]
```

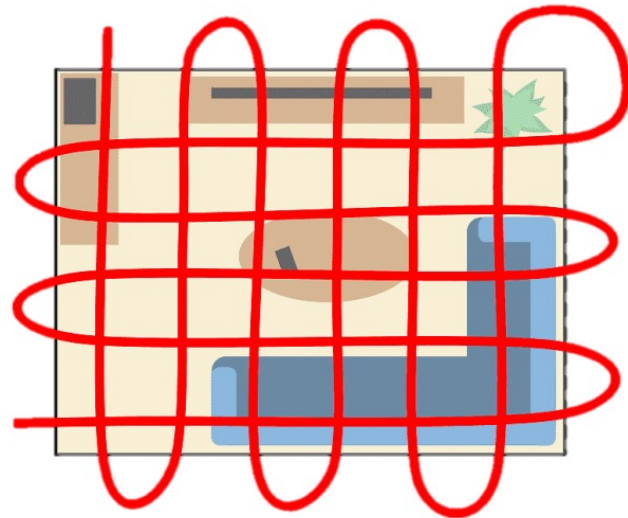




```
1  def cs_kfold(model, data, cv):
2
3      acc_lst = []
4      data = pd.DataFrame(data.sample(frac=1))
5      initial_ratio = round(1 / cv)*len(data) # 단위
6      ratio = initial_ratio
7
8      for i in range(cv):
9          val_data = data[i*initial_ratio: ratio]
10         train_data = data.drop(data[i*initial_ratio:ratio], axis=0)
11
12         X_train, X_test, y_train, y_test = train_data[:-
13                                             1], train_data[-1], val_data[:-1], val_data[-1]
14
15         model.fit(X_train, y_train)
16         pred = model.predict(X_test)
17         acc_lst.append(accuracy_score(y_test, pred))
18
19         ratio += initial_ratio
20
21     return acc_lst
```

# GridSearch (cf. Randomized Search)

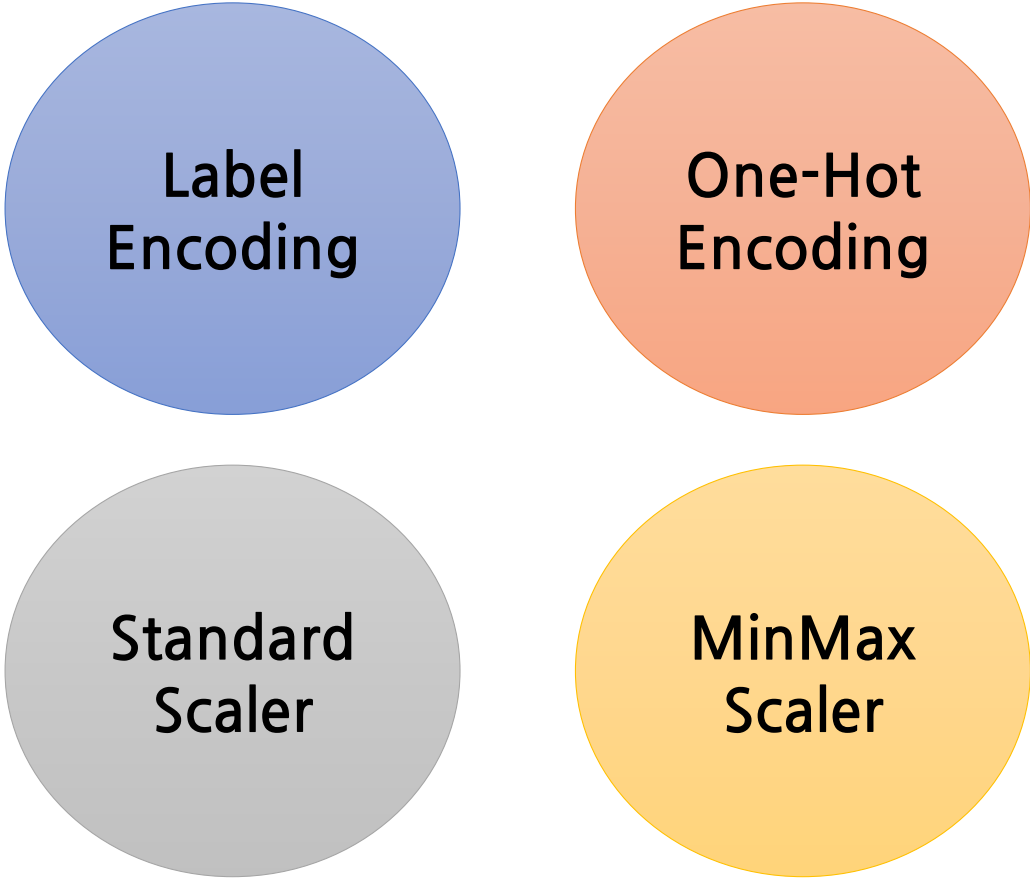
**Kfold(Stratified) +  
Hyper Parameter Tuning**



## CONTENTS

# 04. 전처리

- Dacon 사례
- 전처리의 중요성



**Label  
Encoding**

**One-Hot  
Encoding**

**Standard  
Scaler**

**MinMax  
Scaler**



1) 이름 -> 분석에 필요하지 않기 때문에 삭제

2) Continent, position, prefer\_foot -> 문자열 데이터 => Encoding 필요 (label / get\_dummies)

3) Stat\_overall, stat\_potential vs stat\_skill\_moves -> 비슷한 정보, 다른 데이터 범위 => Scaling 필요?  
(naïve, minmax, standard)

	name	age	continent	contract_until	position	prefer_foot	reputation	stat_overall	stat_potential	stat_skill_moves	value
id											
0	L. Messi	31	south america	2021	ST	left	5.0	94	94	4.0	110500000.0
3	De Gea	27	europa	2020	GK	right	4.0	91	93	1.0	72000000.0
7	L. Suárez	31	south america	2021	ST	right	5.0	91	91	3.0	80000000.0
8	Sergio Ramos	32	europa	2020	DF	right	4.0	91	91	3.0	51000000.0
9	J. Oblak	25	europa	2021	GK	right	3.0	90	93	1.0	68000000.0
...	...	...	...	...	...	...	...	...	...	...	...
16925	S. Adewusi	18	africa	2019	MF	right	1.0	48	63	3.0	60000.0
16936	C. Ehlich	19	europa	2020	DF	right	1.0	47	59	2.0	40000.0
16941	N. Fuentes	18	south america	2021	DF	right	1.0	47	64	2.0	50000.0
16942	J. Milli	18	europa	2021	GK	right	1.0	47	65	1.0	50000.0
16948	N. Christoffersson	19	europa	2020	ST	right	1.0	47	63	2.0	60000.0

8932 rows x 11 columns

# Result

	Naïve + label	Naïve + dummy	Standard + label	Standard + dummy	Minmax + label	Minmax + dummy	Dacon (상위 9%)
RMSE	912,146	929,097	796,927	920,958	874,793	901,762	156,674

# 전처리의 중요성

-> 특정 방법론을 적용할 때 결과가 좋아지는 경향성 존재, 그러나 획기적인 성능 향상은 X

-> 도대체 전처리는 무엇인가!

## **\*\*우리가 고민하는 지점**

전처리 모델을 다 사용해보고 좋은 결과를 내는 모델로 선택 => 과연 내가 전처리를 잘 하고 있는 것일까?

-> Encoder, Scaler 적용 외에도 다양한 전처리 방법들이 존재함

-> 적용해야 하는 데이터의 상황에 맞게, 본인이 적용하는 방법들에 대한 당위를 부여할 수 있게 연습