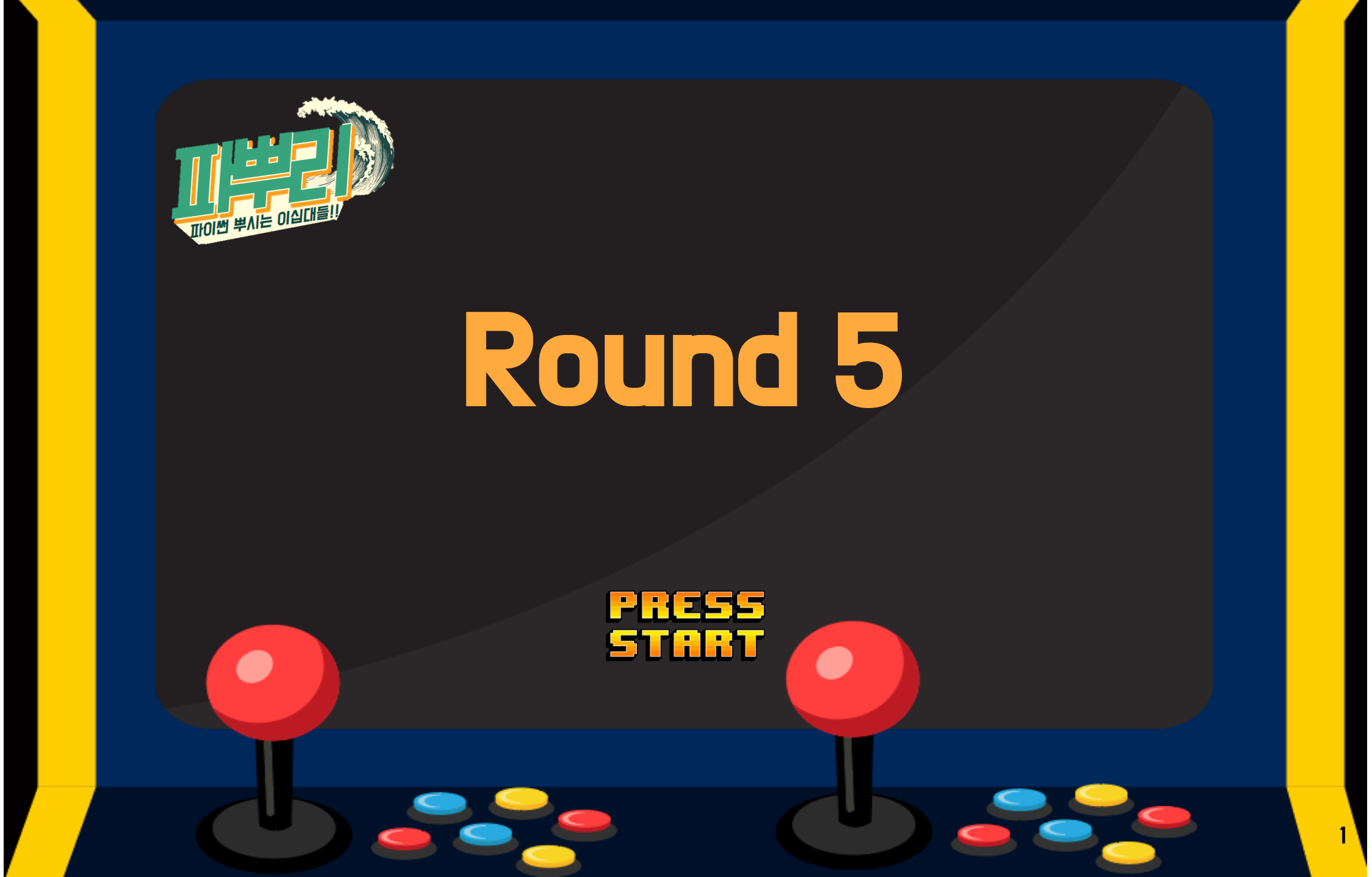

파뿌리
파이썬 뿌시는 이십대들!!
Round 5
PRESS
START

파뿌리
파이썬 뿌시는 이십대들!!

《 Round 5 》

- 데이터 분석의 기초
- 판다스 기초

New Assignment

파뿌리
파이썬 뿌시는 이십대들!!

《 Round 5 》

- 데이터 분석의 기초 《
- 판다스 기초

데이터 분석이란?

데이터 분석이란?

인사이트를 도출하기 위해

알고리즘과 수학적 처리과정을 적용하여

해당 정보에 대한 결론을 도출하고 패턴을 찾기 위한 목적으로

데이터를 다루는 과학

파뿌리
파이썬 뿌시는 이십대들!!

Exploratory Data Analysis

- 문제 정의
- 시각화 & 변수탐색
- 결측치, 이상치 탐지

Data Preprocessing

- 적절한 데이터 처리
- 정규화
- 교차검증 설정

Feature Engineering

- 변수 생성
- 차원 축소
- 특징 추출

Modeling

- 예측 모델링
- 분류 모델링
- 결과 해석

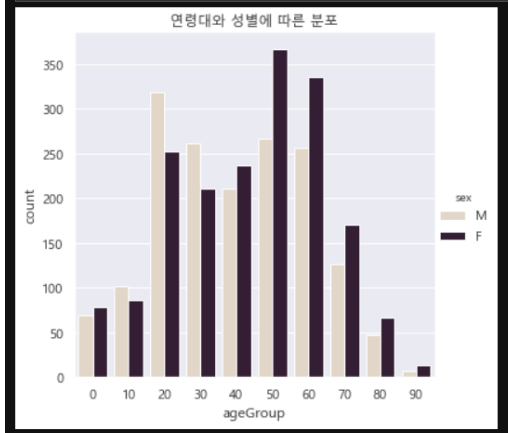
파뿌리
파이썬 뿌시는 이십대들!!

Exploratory Data Analysis

- 문제 정의
- 시각화 & 변수탐색
- 결측치, 이상치 탐지

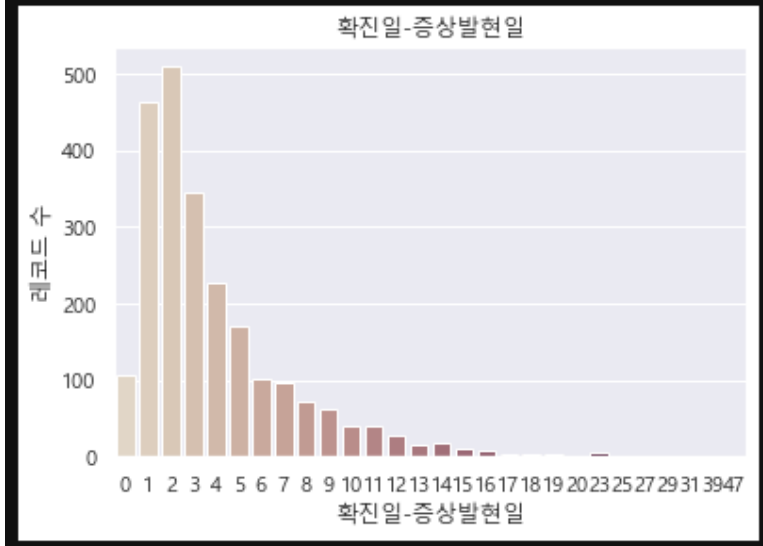
- 연령대에 따라서 어떻게 분포하는가?
- 각 지역에 따라서 어떻게 분포하는가?
- 감염경로별로 어떻게 분포하는가?
- 시간의 흐름에 따른 추이는??
- 증상 발현 및 확진의 관계
- 감염군집
- 접촉자와 감염주요인자간의 비율

```
sns.catplot(x="ageGroup", kind="count", hue="sex", palette="ch:.25", data=cov_df)
plt.title("연령대와 성별에 따른 분포")
plt.show()
```



- 20대와 50대, 60대가 월등히 많았다.
- 내 생각엔 20대는 놀기 위해서 많이 돌아다녔기 때문에고, 50~60대는 교회때문이 아닐지???

```
manTim_df = pd.read_csv("경기 증상 발현 일자.csv")

plot = sns.barplot(x=manTim_df["확진일-증상발현일"], y
plt.title("확진일-증상발현일")
plt.show()

result = ""
n = manTim_df['레코드 수'].sum()
objs = n/2
for i, j in zip(manTim_df["확진일-증상발현일"], manTim
    n = n-j
    if n <= objs  :
        result = i
        break

print("확진일 증상발현일 중앙값 :", result)
```



확진일 증상발현일 중앙값 : 3

파뿌리
파이썬 뿌시는 이십대들!!

Data Preprocessing

- 적절한 데이터 처리
- 정규화
- 교차검증 설정

- '확진자' -> '번호'로 변경
- 'GRP' -> 'primaryGroup'으로 변경
- '구분' -> '감염구분'으로 변경
- '무증상/조사중' -> '특이사항'으로 변경 및 '검사중'인 low 삭제
- columns 이름 영문화
- '구분2', '"구분"', '기준일(발명일, 확진일 선택)', '무증상/조사중 기준일 ' columns 삭제

```
cov_df.drop(cov_df.columns[[13, 14, 16, 17]], axis='columns', inplace=True)
cov_df.columns = ['index', 'number', 'sex', 'age', 'ageGroup', 'ConfirmationDate',
                  'manifestationDate', 'specialNote', 'areaNumber', 'area', 'ReDetection',
                  'infectionRoute', 'primaryGroup', 'infectionType']
cov_df = cov_df[cov_df.specialNote != "조사중"]
cov_df.info(verbose=True)
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3484 entries, 0 to 3510
Data columns (total 14 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   index              3484 non-null   int64
 1   number             3484 non-null   object
 2   sex                3484 non-null   object
 3   age                3484 non-null   int64
 4   ageGroup           3484 non-null   int64
 5   ConfirmationDate   3484 non-null   object
 6   manifestationDate  2304 non-null   object
 7   specialNote        3484 non-null   object
 8   areaNumber         3484 non-null   object
 9   area               3484 non-null   object
 10  ReDetection        3484 non-null   object
 11  infectionRoute     3484 non-null   object
 12  primaryGroup       3484 non-null   object
 13  infectionType      2848 non-null   object
dtypes: int64(3), object(11)
memory usage: 408.3+ KB
```

파뿌리
파이썬 뿌시는 이십대들!!

Feature Engineering

- 변수 생성
- 차원 축소
- 특징 추출

방송일시	노출(분)	마더코드	상품코드	상품명	상품군	판매단가	취급액
-01 06:00:00	20.0	100346	201072	테이트 남성 셀린니트3종	의류	39900	2099000.0
-01 06:00:00	20.0	100346	201079	테이트 여성 셀린니트3종	의류	39900	4371000.0
-01 06:20:00	20.0	100346	201072	테이트 남성 셀린니트3종	의류	39900	3262000.0
-01 06:20:00	20.0	100346	201079	테이트 여성 셀린니트3종	의류	39900	6955000.0
-01 06:40:00	20.0	100346	201072	테이트 남성 셀린니트3종	의류	39900	6672000.0

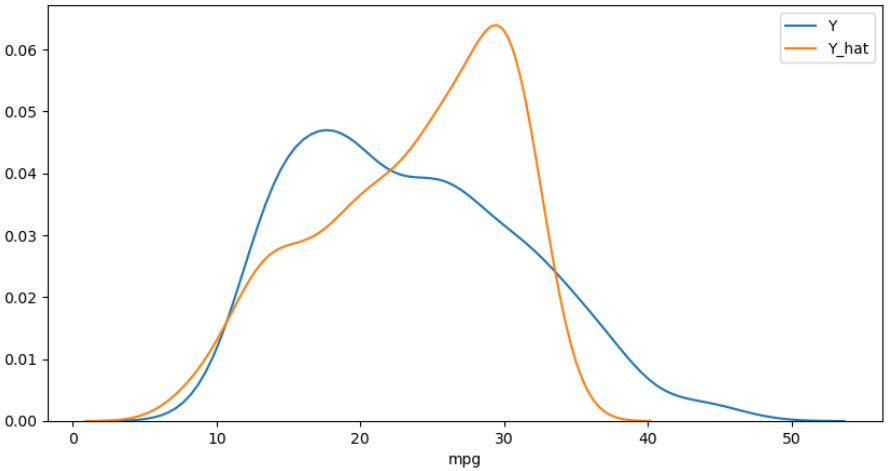
방송일시	노출(분)	마더코드	상품코드	상품명	상품군	판매단가	취급액	주문량	month	day	hour	minute	weekday	season	holiday
-01 06:00:00	20.0	100346	201072	테이트 남성 셀린니트3종	의류	39900	2099000.0	52.606516	1	1	6	0	1	3	1
-01 06:00:00	20.0	100346	201079	테이트 여성 셀린니트3종	의류	39900	4371000.0	109.548872	1	1	6	0	1	3	1
-01 06:20:00	20.0	100346	201072	테이트 남성 셀린니트3종	의류	39900	3262000.0	81.754386	1	1	6	20	1	3	1
-01 06:20:00	20.0	100346	201079	테이트 여성 셀린니트3종	의류	39900	6955000.0	174.310777	1	1	6	20	1	3	1
-01 06:40:00	20.0	100346	201072	테이트 남성 셀린니트3종	의류	39900	6672000.0	167.218045	1	1	6	40	1	3	1

Modeling

- 예측 모델링
- 분류 모델링
- 결과 해석

```python
# sklearn 라이브러리에서 선형회귀분석 모듈 가져오기
from sklearn.linear_model import LinearRegression

# 단순회귀분석 모형 객체 생성
lr = LinearRegression()

## 학습 시작
# train data를 가지고 모형 학습
lr.fit(X_train, Y_train)

# 학습을 마친 모형에 test data를 적용하여 결정계수(R^2) 계산
r_square = lr.score(X_test, Y_test)

# 회귀식과 결정계수(R^2) 산출
print('회귀식 :', float(lr.coef_),'X +',lr.intercept_)
print('결정계수(R^2) :', r_square)
print('\n')

# 모형에 전체 X 데이터를 입력하여 예측한 값 y_hat을 실제 값 y와 비교
Y_hat = lr.predict(X)

plt.figure(figsize=(10, 5))
ax1 = sns.distplot(Y, hist=False, label="Y")
ax2 = sns.distplot(Y_hat, hist=False, label="Y_hat", ax=ax1)
plt.show()
plt.close()
```

```
train data 개수:  274
test data 개수:  118

회귀식 : -0.007753431671236769 X + 46.7103662572801
결정계수(R^2) : 0.6822458558299325
```



파뿌리
파이썬 뿌시는 이십대들!!

데이터 분석을 하기 위해서 가장 필요한 것?

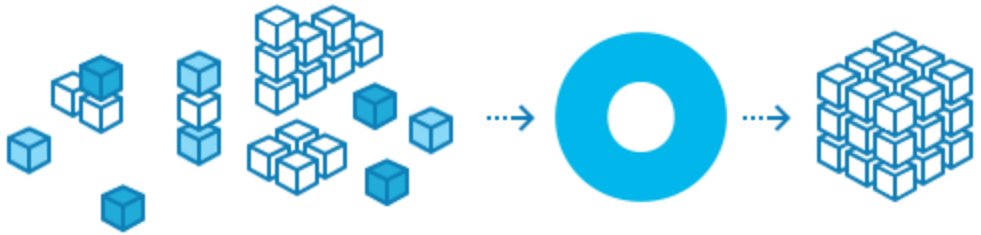
각종 통계적 기법??

다량의 데이터를 처리할 만한 풍부한 컴퓨팅 자원??

편리한 라이브러리와 시각화 도구??

데이터 그 자체가 가장 중요!

데이터분석 과정의 80%는
변수를 탐색하고, 데이터를 다듬고, 분석에 더 적합하게 만드는 것

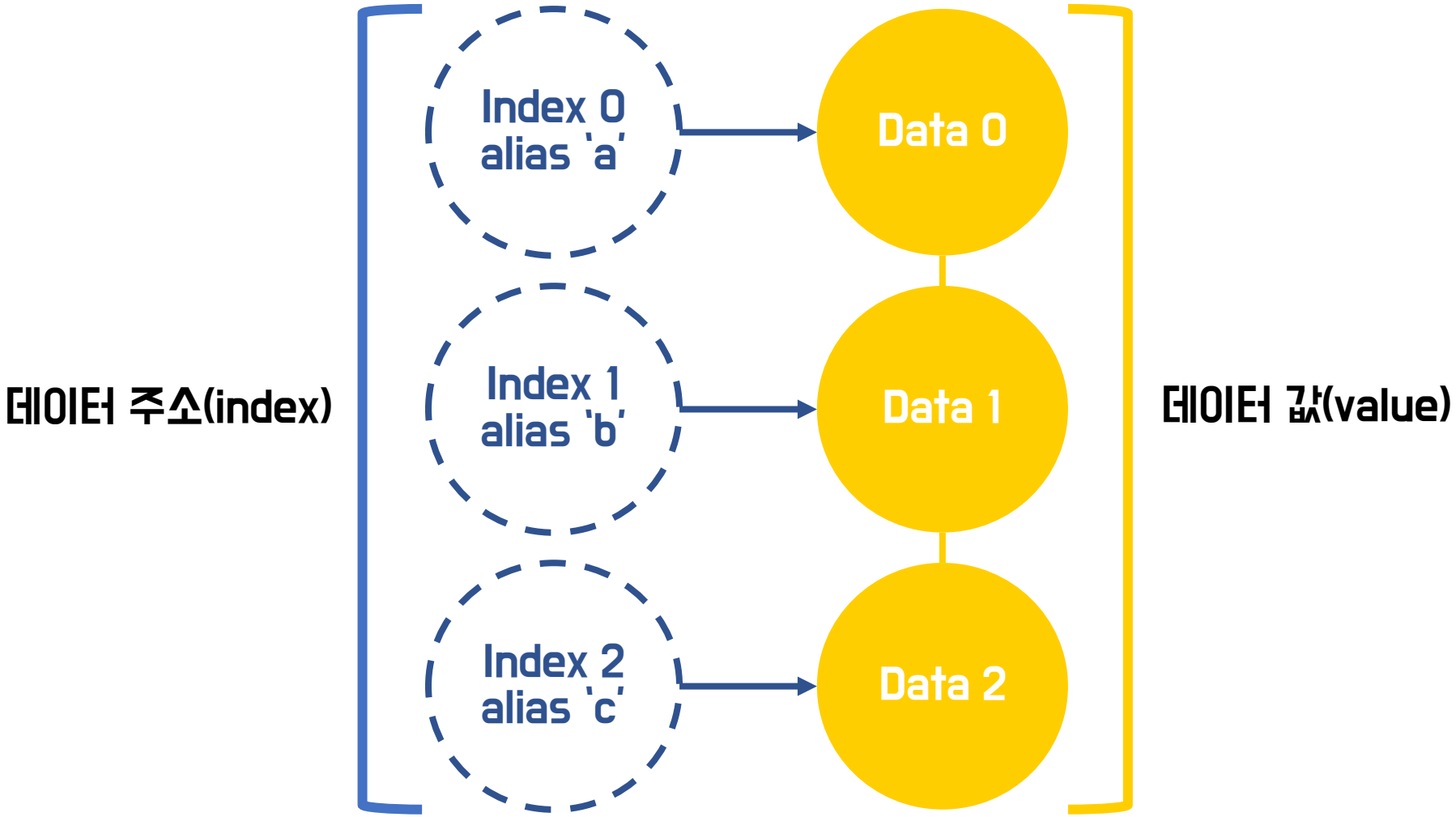


파뿌리
파이썬 뿌시는 이십대들!!

《 Round 5 》

- 데이터 분석의 기초 - complete
- 판다스 기초 《

Series


데이터 주소(index)
Index 0
alias 'a'
Index 1
alias 'b'
Index 2
alias 'c'
Data 0
Data 1
Data 2
데이터 값(value)

파뿌리
파이썬 뿌시는 이십대들!!

Series

- index와 value가 일대일 대응(dictionary와 비슷한 구조)

```python
# pandas 불러오기 Alias pd
import pandas as pd

# Dictionary 생성
dict_data = {'a': 1, 'b': 2, 'c': 3}

# Series로 변환
sr = pd.Series(dict_data)

# sr의 자료형 출력
print(type(sr))

# sr에 저장된 시리즈 객체 출력
print(sr)
```

파뿌리
파이썬 뿌시는 이십대들!!

Series

- 파이썬의 모든 선형 자료형을 series화 가능

```
tuple_data = ('광종', '1997-07-10', 3, True)
tsr = pd.Series(tuple_data, index=["이름", "생년월일", "학년", "재학여부"])
print(tsr)
```

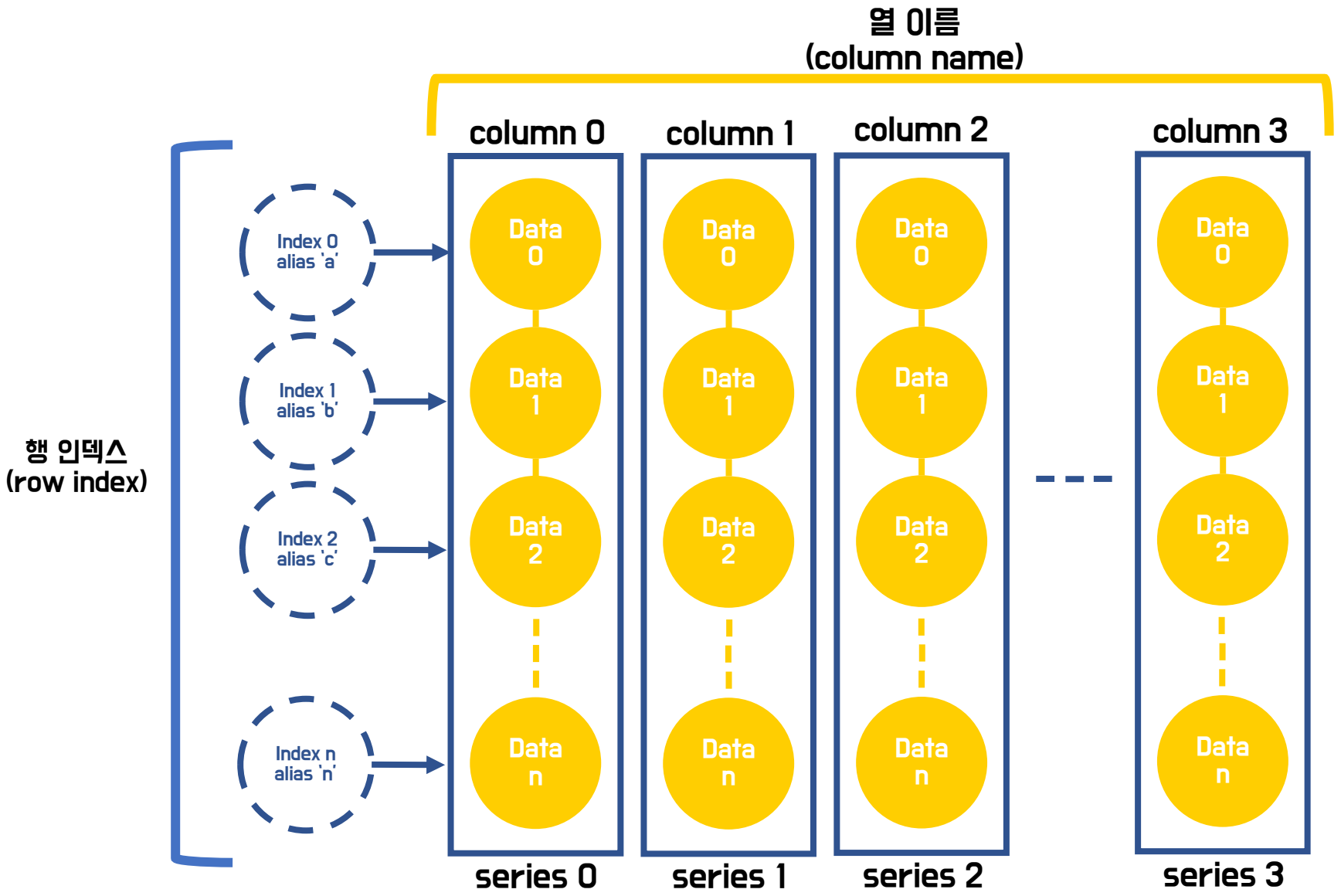
(tuple을 series화 해서 이름형 index를 붙혀주는 모습)

- series는 정수형 인덱스와 이름형 인덱스로 모두 접근 가능

```
print(sr['a'])
print(sr[0])
```

(이름 안 붙혀줄 경우 정수형 index로만 접근가능)

Dataframe


열 이름
(column name)
column 0
column 1
column 2
column 3
행 인덱스
(row index)
Index 0
alias 'a'
Index 1
alias 'b'
Index 2
alias 'c'
Index n
alias 'n'
Data 0
Data 1
Data 2
Data n
series 0
series 1
series 2
series 3

파뿌리
파이썬 뿌시는 이십대들!!

Dataframe

- dataframe을 만들기 위해선 같은 길이의 1차원 배열 여러 개가 필요

```
dict_data = {'c0': [1,2,3], 'c1': [4,5,6], 'c2': [7,8,9], 'c3': [10,11,12]}
df = pd.DataFrame(dict_data)
print(type(df))
print('\n')
print(df)
```

```
<class 'pandas.core.frame.DataFrame'>


   c0  c1  c2  c3
0   1   4   7  10
1   2   5   8  11
2   3   6   9  12
```

Dataframe

- 요렇게도 가능

```
import pandas as pd

df = pd.DataFrame([[15, '남', '덕영중'], [17, '여', '강남중'],
                   [19, '남', '상정고'], [14, '여', '산곡여중']],
                   index =['팡종', '슙인', '주년', '션'],
                   columns =['나이', '성별', '학교'])
print(df)
```

	나이	성별	학교
팡종	15	남	덕영중
슙인	17	여	강남중
주년	19	남	상정고
션	14	여	산곡여중

파뿌리
파이썬 뿌시는 이십대들!!

Dataframe

- DataFrame으로 저장된 데이터들에 통계함수를 직접적으로 사용가능

- 다양한 포맷으로 import/export할 수 있음
(CSV, JSON, Excel…)

- 행에 대한 접근은 DataFrame.loc[인덱스 이름]과 .iloc[정수형 위치 인덱스]

- 열에 대한 접근은 DataFrame["열이름"] 혹은 ["정수형 위치 인덱스"]

- DataFrame의 깊은 복사는 s.copy()