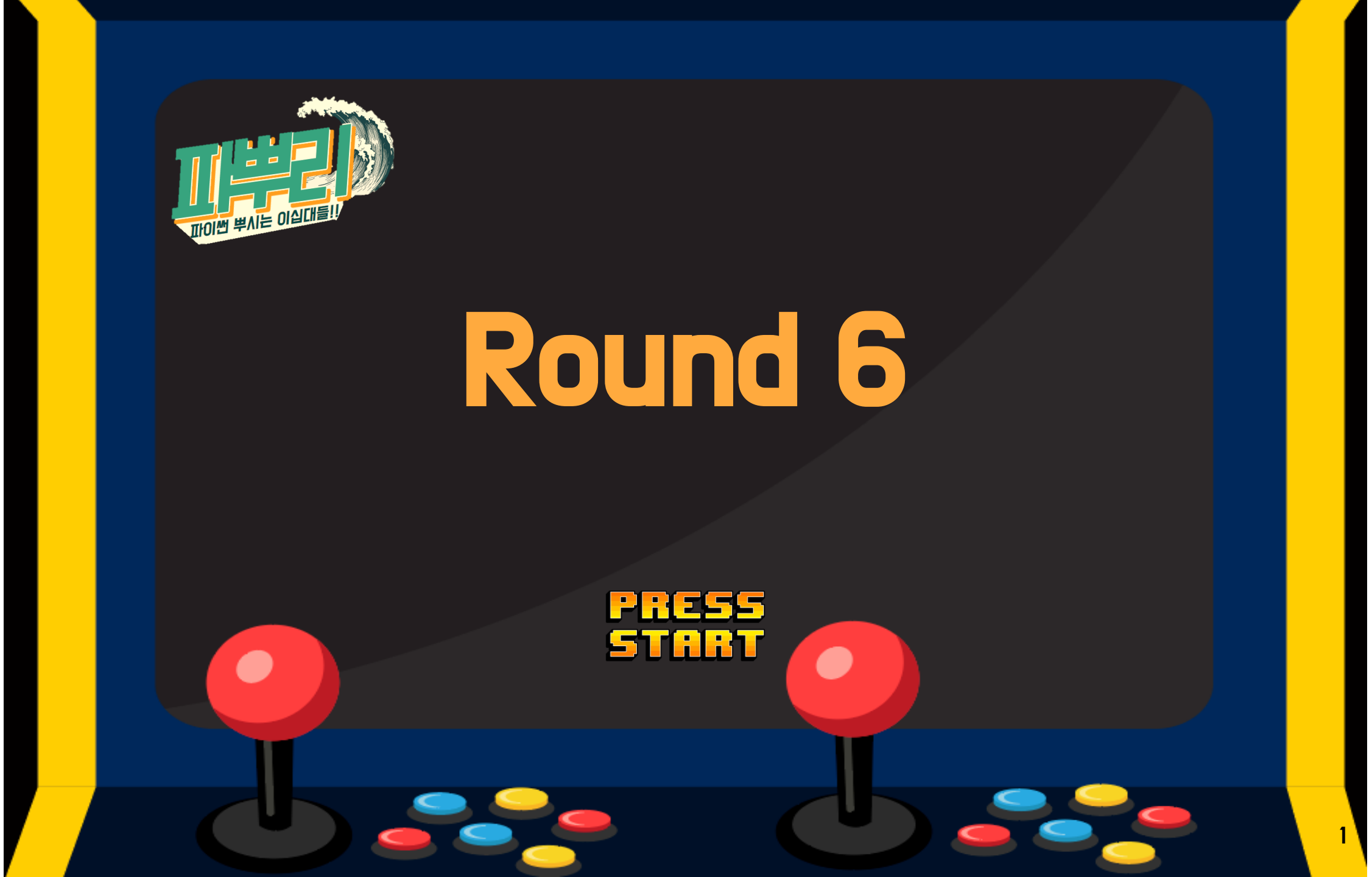

파뿌리
파이썬 뿌시는 이십대들!!
Round 6
PRESS
START

파뿌리
파이썬 뿌시는 이십대들!!

《 Round 6 》

- 데이터 사전처리 개요
- 데이터 확인
- 데이터 표준화
- 결측치 처리

New Assignment

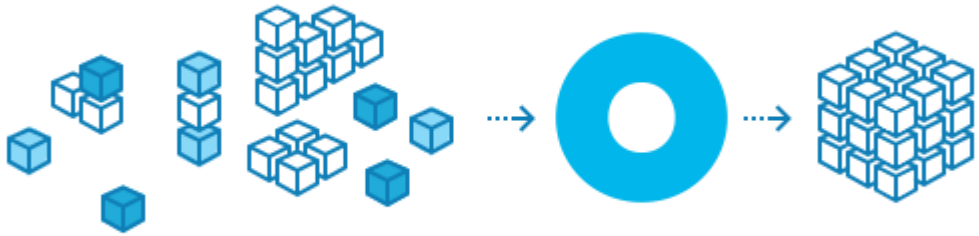
파뿌리
파이썬 뿌시는 이십대들!!

《 Round 6 》

- 데이터 사전처리 개요 《
- 데이터 확인
- 데이터 표준화
- 결측치/중복치 처리

파뿌리
파이썬 뿌시는 이십대들!!

Data preprocessing

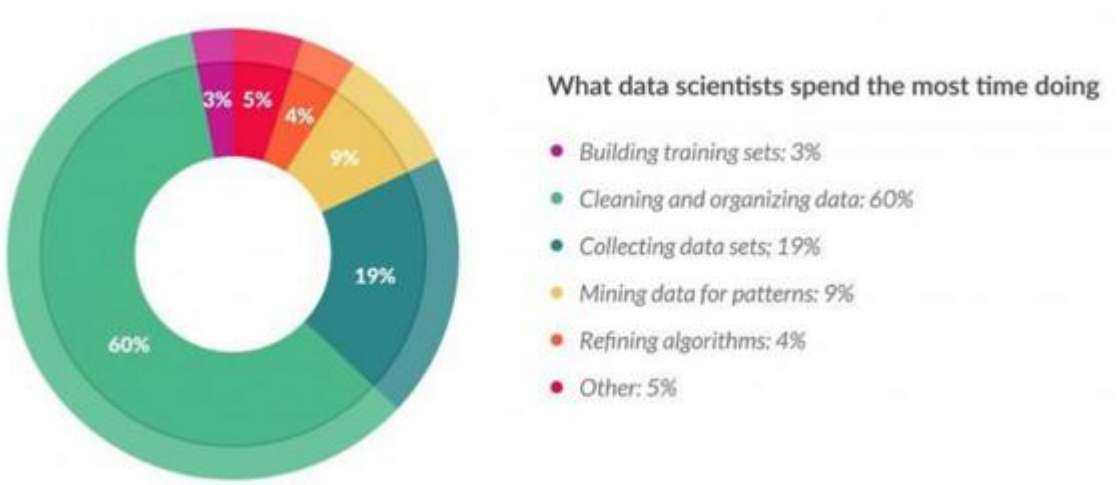


데이터셋은 보통 바로 분석이 불가능할 정도로 지저분(messy)하다.

분석이 가능한 상태로 만드는 것이 Data Preprocessing

파뿌리
파이썬 뿌시는 이십대들!!

Data preprocessing



데이터 분석의 과정의 대부분을 차지할 정도로 굉장히 중요한 작업!

공개 데이터 셋인 엔진 데이터셋(auto_mpg.csv) 으로 수행해보자!

파뿌리
파이썬 뿌시는 이십대들!!

《 Round 6 》

- 데이터 사전처리 개요
- 데이터 확인 《
- 데이터 표준화
- 결측치 처리

파뿌리
파이썬 뿌시는 이십대들!!

데이터 내용 확인하기

```
dfm = pd.read_csv('auto-mpg.csv', header=None)
dfm.columns = ['mpg(연비)', 'cylinders(실린더 수)', 'displacement(배기량)',
               'horsepower(출력)', 'weight(차중)',
                  'acceleration(가속능력)', 'model_year(출시년도)',
               'origin_number(제조국 번호)', 'name(모델 명)']
```

데이터 불러오기 및 컬럼지정

데이터 내용 확인하기

```
   mpg(연비)  cylinders(실린더 수)  displacement(배기량) horsepower(출력)  weight(차중)  \
0     18.0                 8             307.0          130.0      3504.0
1     15.0                 8             350.0          165.0      3693.0
2     18.0                 8             318.0          150.0      3436.0
3     16.0                 8             304.0          150.0      3433.0
4     17.0                 8             302.0          140.0      3449.0

   acceleration(가속능력)  model_year(출시년도)  origin_number(제조국 번호)  \
0              12.0               70                      1
1              11.5               70                      1
2              11.0               70                      1
3              12.0               70                      1
4              10.5               70                      1

                 name(모델 명)
0  chevrolet chevelle malibu
1          buick skylark 320
2         plymouth satellite
3              amc rebel sst
4                ford torino
```

DataFrame.head()
or
DataFrame.tail()

데이터의 일부분 확인

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 398 entries, 0 to 397
Data columns (total 9 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   mpg(연비)                398 non-null    float64
 1   cylinders(실린더 수)       398 non-null    int64
 2   displacement(배기량)       398 non-null    float64
 3   horsepower(출력)          398 non-null    object
 4   weight(차중)              398 non-null    float64
 5   acceleration(가속능력)      398 non-null    float64
 6   model_year(출시년도)        398 non-null    int64
 7   origin_number(제조국 번호)  398 non-null    int64
 8   name(모델 명)              398 non-null    object
dtypes: float64(4), int64(3), object(2)
memory usage: 28.1+ KB
```

DataFrame.info()

데이터의 요약정보 확인

데이터 내용 확인하기

	mpg(연비)	cylinders(실린더 수)	displacement(배기량)	horsepower(출력)	weight(차중)	acceleration(가속능력)	model_year(출시년도)
count	398.000000	398.000000	398.000000	398	398.000000	398.000000	398.000000
unique	NaN	NaN	NaN	94	NaN	NaN	NaN
top	NaN	NaN	NaN	150.0	NaN	NaN	NaN
freq	NaN	NaN	NaN	22	NaN	NaN	NaN
mean	23.514573	5.454774	193.425879	NaN	2970.424623	15.568090	76.010050
std	7.815984	1.701004	104.269838	NaN	846.841774	2.757689	3.697627
min	9.000000	3.000000	68.000000	NaN	1613.000000	8.000000	70.000000
25%	17.500000	4.000000	104.250000	NaN	2223.750000	13.825000	73.000000
50%	23.000000	4.000000	148.500000	NaN	2803.500000	15.500000	76.000000
75%	29.000000	8.000000	262.000000	NaN	3608.000000	17.175000	79.000000
max	46.600000	8.000000	455.000000	NaN	5140.000000	24.800000	82.000000

DataFrame.describe(include='all')

데이터의 기술통계 확인

파뿌리
파이썬 뿌시는 이십대들!!

데이터 내용 확인하기

```
print(dfm["origin_number(제조국 번호)"].value_counts())
```

```
1    249
3     79
2     70
Name: origin_number(제조국 번호), dtype: int64
```

DataFrame.describe(include='all')

범주형 데이터의 고유값 개수 확인

파뿌리
파이썬 뿌시는 이십대들!!

《 Round 6 》

- 데이터 사전처리 개요
- 데이터 확인
- 데이터 표준화 《
- 결측치 처리

파뿌리
파이썬 뿌시는 이십대들!!

데이터 표준화

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 398 entries, 0 to 397
Data columns (total 9 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   mpg(연비)                 398 non-null    float64
 1   cylinders(실린더 수)        398 non-null    int64
 2   displacement(배기량)       398 non-null    float64
 3   horsepower(출력)          398 non-null    object
 4   weight(차중)              398 non-null    float64
 5   acceleration(가속능력)      398 non-null    float64
 6   model_year(출시년도)        398 non-null    int64
 7   origin_number(제조국 번호)  398 non-null    int64
 8   name(모델 명)              398 non-null    object
dtypes: float64(4), int64(3), object(2)
memory usage: 28.1+ KB
```

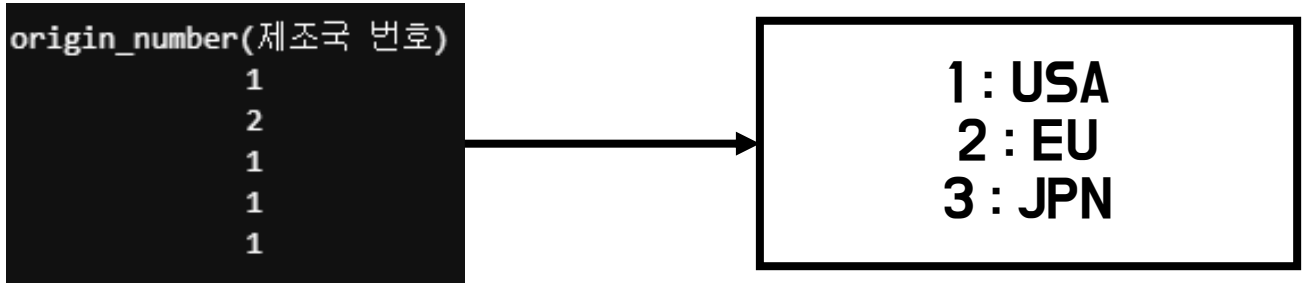
DataFrame.info()

데이터 표준화

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 398 entries, 0 to 397
Data columns (total 9 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   mpg(연비)                 398 non-null    float64
 1   cylinders(실린더 수)        398 non-null    int64
 2   displacement(배기량)       398 non-null    float64
 3   horsepower(출력)          398 non-null    object
 4   weight(차중)              398 non-null    float64
 5   acceleration(가속능력)       398 non-null    float64
 6   model_year(출시년도)        398 non-null    int64
 7   origin_number(제조국 번호)  398 non-null    int64
 8   name(모델 명)              398 non-null    object
dtypes: float64(4), int64(3), object(2)
memory usage: 28.1+ KB
```

해당 데이터는 int보단 object나 category 타입이 더 잘 어울림

데이터 표준화




```
dfm["origin_number(제조국 번호)"].replace({1:"USA", 2:"EU", 3:"JPN"}, inplace=True)
dfm["origin_number(제조국 번호)"] = dfm["origin_number(제조국 번호)"].astype('category')
print(dfm["origin_number(제조국 번호)"].unique())
```

```
[USA, JPN, EU]
Categories (3, object): [USA, JPN, EU]
```

표준화를 통해 가독성 상승시키고. int -> category형변환을 통해 효율성 상승

파뿌리
파이썬 뿌시는 이십대들!!

《 Round 6 》

- 데이터 사전처리 개요
- 데이터 확인
- 데이터 표준화
- 결측치 처리 《

파뿌리
파이썬 뿌시는 이십대들!!

결측치

말 그대로 비어있는 값이다. 0과는 다름.
결측치가 있으면 모델링을 할 수 없고, 각종 통계치를 확인 할 수 없게 된다.
데이터 전처리시 가장 먼저 처리해야 하는 값

결측치의 종류 :

None, "", "?", np.NaN, "-" 등…

파이썬에서 NaN은 numpy의 numpy.nan을 사용함

파뿌리
파이썬 뿌시는 이십대들!!

결측치

결측치의 처리 방법의 종류 :

a. 삭제 : 간편하나 데이터의 수가 적어짐

b. 대체 : 다른 관측치들의 평균값, 최빈값, 중간값, 인근값 등으로 대체하는 것

c. 예측 값 삽입 : 관측치들 만을 이용해서 결측치을 예측하는 모델을 만들어 결측치를 예측하는 방법

결측치 확인

```
print(dfm.isnull().sum())
```

```
mpg(연비)                  0
cylinders(실린더 수)       0
displacement(배기량)       0
horsepower(출력)           0
weight(차중)               0
acceleration(가속능력)     0
model_year(출시년도)       0
origin_number(제조국 번호) 0
name(모델 명)              0
dtype: int64
```

	mpg(연비)	cylinders(실린더 수)	displacement(배기량)	horsepower(출력)
count	398.000000	398.000000	398.000000	398
unique	NaN	NaN	NaN	94
top	NaN	NaN	NaN	150.0
freq	NaN	NaN	NaN	22
mean	23.514573	5.454774	193.425879	NaN
std	7.815984	1.701004	104.269838	NaN
min	9.000000	3.000000	68.000000	NaN
25%	17.500000	4.000000	104.250000	NaN
50%	23.000000	4.000000	148.500000	NaN
75%	29.000000	8.000000	262.000000	NaN
max	46.600000	8.000000	455.000000	NaN

Isnull() 에는 검출이 되지 않지만, 기술통계를 확인할 때 특정열에 결측치가 존재함을 알 수 있다.

결측치 처리

```
dfm.replace('?', np.nan, inplace=True)
dfm = dfm.astype({'horsepower(출력)' : np.float})
mean_power = dfm['horsepower(출력)'].mean()
dfm['horsepower(출력)'].fillna(mean_power, inplace=True)
print(dfm.describe(include='all'))
```

	mpg(연비)	cylinders(실린더 수)	displacement(배기량)	horsepower(출력)
count	398.000000	398.000000	398.000000	398.000000
unique	NaN	NaN	NaN	NaN
top	NaN	NaN	NaN	NaN
freq	NaN	NaN	NaN	NaN
mean	23.514573	5.454774	193.425879	104.469388
std	7.815984	1.701004	104.269838	38.199187
min	9.000000	3.000000	68.000000	46.000000
25%	17.500000	4.000000	104.250000	76.000000
50%	23.000000	4.000000	148.500000	95.000000
75%	29.000000	8.000000	262.000000	125.000000
max	46.600000	8.000000	455.000000	230.000000

1. '?' -> numpy.nan으로 교체
2. numpy.float으로 형변환
3. 평균 값으로 결측치 대체

파뿌리!
파이썬 뿌시는 이십대들!!

NEXT STAGE