

pandas

- 쉽고 직관적인 관계형 또는 분류 데이터로 작업 할 수 있도록 설계된 빠르고 유연한 데이터 구조를 제공하는 python 패키지
- 데이터 구조와 조작, 결측값 처리, 데이터 시각화 등 다양한 기능을 제공하여 데이터 처리 작업을 보다 쉽게 수행 가능
- 인덱스(Index)에 따라 데이터를 나열하므로 사전(Dictionary) 자료형에 가까움
- 시리즈(Series)를 기본적인 자료형으로 사용

1. 주요 데이터 구조

- Series : 1차원 배열과 같은 데이터 구조로 데이터와 인덱스를 포함
- DataFrame : 2차원 배열과 같은 데이터 구조로, 행과 열로 이루어진 데이터를 저장

2. Pandas 기능

- 데이터 읽기 : SV, Execl, SQL 등 다양한 형식의 파일을 읽어올 수 있음
- 데이터 저장 : 처리한 데이터를 CSV, Execl, SQL 등 다양한 형식으로 저장 가능
- 데이터 조작 : 데이터 필터링, 정렬, 그룹화, 합치기 등 다양한 데이터 조작 가능
- 결측값 처리 : 결측값 처리 기능을 제공하며, 누락된 데이터를 처리하고 대체할 수 있음
- 데이터 시각화 : Matplotlib와 함께 사용하여 데이터 시각화 가능

3. Pandas 사용법

Series

	Name
0	영지
1	은지
2	미현
3	유진

Index

DataFrame

	Name	Age	Major	Score
0	영지	20	Electrical	80
1	은지	22	Computer	65
2	미현	21	Electrical	70
3	유진	24	Mechanical	85

(1) Series

- 인덱스와 값으로 구성

```
In [ ]: import pandas as pd

data = ['영지', '은지', '미현', '유진']

stu_name = pd.Series(data, name='Name')

print(stu_name)
```

(2) DataFrame

- 데이터프레임이란 행과 열로 구성된 2차원 데이터 구조
- 표 형식의 데이터를 다룰 때 유용
- 데이터 조작, 필터링, 정렬, 시각화 등 다양한 데이터 처리 작업을 수행할 때 유용

```
In [ ]: import numpy as np
import pandas as pd
```

```
In [ ]: # 딕셔너리를 사용한 데이터 만들기
```

```
data1 = {"Name" : ['영지', '은지', '미현', '유진'],
         "Age" : [20, 22, 21, 24],
         "Major" : ['Electrical', 'Computer', 'Electrical', 'Mechanical'],
         "Score" : [80, 65, 70, 85]}

data1
```

```
In [ ]: # DataFrame 형식으로 저장
df = pd.DataFrame(data1)
print(df)
```

```
In [ ]: # 열(Row) 이름 지정
df = pd.DataFrame(data1, index = ['row1', 'row2', 'row3', 'row4'])
```

```
In [ ]: # print로 출력할 경우와 그냥 출력할 경우 차이
print(df)
df
```

```
In [ ]: # 리스트를 이용한 데이터 만들기
```

```
data2 = [['철수', 20, 100.0],
         ['영희', 25, 95.0],
         ['길동', 23, 80.0]]

print(data2)
```

```
In [ ]: # 리스트를 데이터 프레임으로 변환
```

```
df = pd.DataFrame(data2)
df
```

```
In [ ]: # df의 행열 이름 지정하기
```

```
df = pd.DataFrame(data2,
                  index = ['row1', 'row2', 'row3'],
                  columns = ['Name', 'Age', 'Score'])

df
```

2. Subset

- 데이터프레임에서 필요한 행 또는 열을 선택하는 것을 의미

2-1. [] / loc / iloc

```
In [ ]: # 행(column) 데이터 가져오기
# 행의 이름과 데이터 타입도 함께 출력
df['Name']
```

```
In [ ]: # 특정 행 여러개를 불러오는 경우
df[['Name', 'Score']]
```

```
In [ ]: # 열(row) 데이터 가져오기
df.loc['row1']
```

```
In [ ]: # 특정 열 여러개를 가져오는 경우
df.loc[['row1', 'row3']]
```

```
In [ ]: # 특정 데이터 하나만 가져오기
df.loc['row1', 'Name']
```

```
In [ ]: # loc을 사용해 특정 행 데이터 모두 가져오기
df.loc[:, 'Name']
```

```
In [ ]: # loc을 사용해 여러개 행 데이터 모두 가져오기
df.loc[:, ['Name', 'Score']]
```

```
In [ ]: # slicing
# loc을 사용해 1행부터 3행까지 데이터 모두 가져오기
df.loc[:, 'Name' : 'Score']
```

```
In [ ]: # iloc
# 숫자를 사용한 indexing/slicing
df.iloc[:, [0,1]]
```

```
In [ ]: df.iloc[:,2, [0,2]]
```

```
In [ ]: df.iloc[-1::,:]
```

2-2. head() / tail()

```
In [ ]: # head()
# 위에서부터 원하는 row 갯수 출력
df.head(2)
```

```
In [ ]: # tail()
# 아래에서부터 원하는 row 갯수 출력
df.tail(2)
```

3. Summarizing Data

DataFrame에서 자주 사용하는 함수

- info() : 데이터프레임의 구조와 열에 대한 정보를 출력
- describe() : 데이터프레임의 열에 대한 기술 통계를 요약하여 출력
- nunique() : 열에서 고유한 값의 개수를 반환
- value_count() : 열의 각 고유한 값과 해당 값의 개수를 반환
- count() : 열에서 비어있지 않은 값 (결측치가 아닌 값)의 개수를 반환
- sum() : 열의 값들의 합을 계산
- max() : 열에서 최댓값을 반환
- std() : 열의 값들의 표준편차를 계산

```
In [ ]: # info()
df.info()
```

```
In [ ]: # describe()
# mean: 평균
# std: 표준편차
# min: 최솟값
# 25%: 1사분위수 (25% 지점)
# 50%: 중앙값 (50% 지점)
# 75%: 3사분위수 (75% 지점)
# max: 최댓값
df.describe()
```

```
In [ ]: # nunique()
# unique한 값의 수를 반환
df2 = df.copy()
df2.loc['row2', 'Score'] = np.NaN

df2
```

```
In [ ]: df2.nunique()
```

```
In [ ]: df2['Score'].nunique()
```

```
In [ ]: # value_counts()

Value_cnt = df2['Score'].value_counts()

print("value count : \n",Value_cnt)
print("\n")

# count()

Count = df2['Score'].count()

print("count : ", Count)
print("\n")

# sum()

Sum = df2.sum()

print("sum",Sum)
print("\n")

# std()

Std = df2['Score'].std()

print("std : ", Std)
print("\n")
```

4. Column Exchange

- 칼럼 순서 바꾸기

```
In [ ]: df4 = df.copy()

df4
```

```
In [ ]: df4 = df4.iloc[:, [0,2,1]]
```

```
df4
```

5. 논리연산 데이터 필터

```
In [ ]: data = {
    'Class' : [ '전자전기', '기계공학', '반도체', '전자전기', '기계공학', '반도체', '전자전기' ],
    'Name' : [ 'bob', 'zoe', 'sam', 'ben', 'amy', 'ava', 'leo' ],
    'Age' : [ 20,21,23,21,24,22,25 ],
    'Score' : [ 60,90,40,95,70,75,80 ]
}

df = pd.DataFrame(data)

df
```

```
In [ ]: # 80점 이상 학생 출력
df['Score'] >= 80
```

```
In [ ]: df.loc[df['Score'] >= 80]
```

```
In [ ]: # 80점 이상 학생의 이름과 나이만 출력
df.loc[df['Score'] >= 80 , ['Name', 'Age']]
```

```
In [ ]: # 80점 이상 학생 'Result' 컬럼 만들어주기
df['Result']='None'
df
```

```
In [ ]: df.loc[df['Score'] >= 80 , 'Result'] = 'Pass'
df.loc[df['Score'] < 80 , 'Result'] = 'Fail'

df
```

```
In [ ]: df['Result']=='Pass'

idx = (df['Result']=='Pass')

df.loc[idx]
```

```
In [ ]: df_sorted = df.loc[idx].sort_values('Score')

df_sorted
```

```
In [ ]: # 데이터 엑셀 저장
df_sorted.to_excel('data_sorted.xlsx')

# 엑셀 파일 불러오기
df_import = pd.read_excel('data_sorted.xlsx')

df_import
```

6. 그룹 분석

```
In [ ]: df
```

```
In [ ]: df.groupby(by='Class').mean()  
  
#df.groupby(by='Class').count()  
  
#df.groupby(by='Class').min()  
  
#df.groupby(by='Class').std()
```

```
In [ ]:
```

7. Plotting

```
In [ ]: # matplotlib에서 발생하는 글자체 문제 해결방법  
  
import matplotlib.pyplot as plt  
plt.rcParams['font.family'] = 'Malgun Gothic'
```

```
In [ ]: # plot.bar  
  
#df.plot.bar('Name', 'Score')  
  
df.plot.bar('Name', ['Score', 'Age'])
```

8. 빈데이터 처리하기

```
In [ ]: df.loc[2:5, 'Score'] = np.NaN  
  
df
```

```
In [ ]: # isnull()  
# 값이 null이면 true, 아니면 false 출력  
  
df.isnull()  
  
# dropna()  
# 데이터가 없는 열은 drop  
  
df.dropna()  
  
# fillna()  
# 데이터가 없는 셀을 원하는 값으로 채우기  
value = 0  
  
df.fillna(value)  
  
# replace()  
# 특정 값을 내가 원하는 값으로 변환  
df.replace(np.nan, -1)  
  
# interpolate  
# 주위의 평균값으로 빈 값 채우는 것  
df.interpolate()
```

9. DataFrame 함수 적용

```
In [ ]: def mul_ten(x) :  
        return x*10  
  
        # apply()  
        # 함수를 적용시켜준다  
  
df['Age'].apply(mul_ten)
```

10. DataFrame 합치기

```
In [ ]: # 세로 방향으로 합치기  
  
df_vertical = pd.concat([df,df])  
  
df_vertical  
  
# 가로 방향으로 합치기  
  
df_horizontal = pd.concat([df,df], axis = 1)  
  
df_horizontal
```