직장인을 위한 파이썬 데이터 분석

Pandas Cheat Sheet

Pandas

Pandas는 panel data analysis(패널 자료)의 약자로, 파이썬을 활용한 **정형 데이터 분석에서 가장 많이 활용**되는 라이브러리입니다.

라이브러리 로딩

```
import pandas as pd
```

Pandas 데이터 구조

시리즈 (Series)

어떤 데이터 타입도 담을 수 있는 **1차원의 라벨링**(labeled) 된 **배열**입니다.

인덱스 (Index)

A	1
B	2
C	3
D	4

```
s = pd.Series([1, 2, 3, 4],  index=['A',  'B',  'C',  'D'])
```

데이터 프레임 (DataFrame)

인덱스 (Index) 와 열 (Columns) 로 이루어진 2차원의 라벨링(labeled) 된 **데이터 구조**입니다.

열 (Columns)

인덱스 (Index)

	국가	수도	인구(만)
A	한국	서울	9.8
B	중국	베이징	19.6
C	일본	도쿄	14.0

```
data = {'국가': ['한국',  '중국',  '일본'],
        '수도': ['서울',  '베이징',  '도쿄'],
        '인구': [9828094, 19612368, 14049146]}

df = pd.DataFrame(data, columns=['국가',  '수도',  '인구'])
```

슬라이싱 (Slicing) / 인덱싱 (Indexing)

값 선택

```
s['A']
> 1
```
시리즈 (Series) 값 선택

```
df[1:]
>    국가   수도    인구
> 1  중국   베이징  19612368
> 2  일본   도쿄    14049146
```
데이터 프레임 (DataFrame) 값 선택

값 변경

```
s['A'] = 6
```
'A' 인덱스에 위치한 값을 6으로 변경

loc / iloc

```
df.loc(0,  ['국가'])
> '한국'
```
라벨(인덱스 값)을 통한 선택

```
df.iloc(0, 0)
> '한국'
```
위치를 통한 선택

Boolean 인덱싱

```
s[(s < 1) | (s > 2)]
> C    3
> D    4
```
값이 1 미만이거나 2 초과인 값 선택

```
df[df['인구']>12000000]
>    국가   수도    인구
> 1  중국   베이징  19612368
> 2  일본   도쿄    14049146
```
'인구' 열 값이 12,000,000 이상인 데이터 선택

데이터 삭제

```
s.drop(['A', 'B'])
df.drop('국가', axis=1)
```
행 방향 데이터 삭제 (axis = 0)
열 방향 데이터 삭제 (axis = 1)

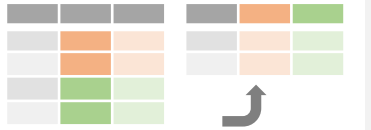
정렬

```
df.sort_index(by='국가')
df.reset_index(drop=True)
```
행 / 열 인덱스 기준 정렬
인덱스 초기화 (기존 인덱스 열 삭제)

데이터 프레임 결합

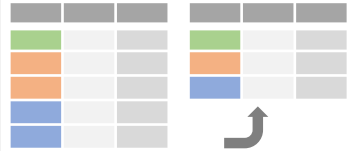
피벗(Pivot) 테이블

```
df.pivot(columns='열', values='값')
```



그룹 연산 (Groupby)

```
df.groupby(by="열이름")
```



데이터 프레임 결합

Merge (겹치는 값을 기준으로 데이터 결합)

```
pd.merge(df1, df2, how='left', on='Key 열')
pd.merge(df1, df2, how='right', on='Key 열')
pd.merge(df1, df2, how='inner', on='Key 열')
pd.merge(df1, df2, how='outer', on='Key 열')
```
Left Merge
Right Merge
Inner Merge
Outer Merge

Concat (행 / 열 방향 데이터 결합)

```
pd.concat([df1,df2], axis=0)
pd.concat([df1,df2], axis=1)
```
행 방향 결합
열 방향 결합

결측 / 중복 처리

결측 값 처리

```
df.dropna()
df.fillna('대체값')
```
결측값 행 삭제
결측값 대체

중복 값 처리

```
df.drop_duplicates()
```
중복 행 삭제

데이터 정보 확인

기본 정보 확인

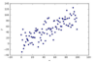
```
df.info()
df.columns
df.index
```
전체 데이터 정보
열 (Columns) 이름
인덱스 (Index) 이름

요약 통계 확인

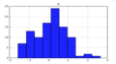
```
df.describe()
df['열이름'].value_counts()
df.sum() / df.mean() / df.min() / df.max()
```
요약 통계 정보
값 개수 집계
합계 / 평균 / 최소 / 최대값

시각화

```
df.plot(kind='scatter')
# 산점도
```



```
df.plot(kind='hist')
# 히스토그램
```



파일 읽기 / 쓰기

CSV 파일

```
df = pd.read_csv('file.csv')     # 파일 읽기
df.to_csv('file.csv')            # 파일 저장
```

EXCEL 파일

```
df = pd.read_excel('file.xlsx')                 # xlsx 파일 읽기
df.to_excel('file.xlsx',  sheet_name='Sheet1')  # 파일 저장
xlsx = pd.ExcelFile('file.xls')                 # xls 파일 읽기
df = pd.read_excel(xlsx,  'Sheet1')
```

데이터 베이스 (SQL) 연결

```
from sqlalchemy import create_engine
engine = create_engine('sqlite:///:memory:')

df=pd.read_sql('SELECT * FROM my_table;', engine) # sql 문으로 로딩
df=pd.read_sql_table('my_table', engine)          # 테이블명으로 로딩
df=pd.read_sql_query('SELECT * FROM my_table;', engine)

df.to_sql('my_dataframe', engine)                 # sql 파일 저장
```

※ read_sql() 은 read_sql_table() 과 read_sql_query() 를 묶어서 사용하는 것과 같습니다.