Step 16-1: Pandas and Data Analytics

Instructor: Eunil Park (eunilpark@skku.edu)

SUNG KYUN KWAN
UNIVERSITY

01. Pandas

SUNG KYUN KWAN UNIVERSITY

데이터 다루기

- Excel
- Matrix
- Pandas: 넘파이를 기반
  - 처리속도가 빠름
  - 행과 열로 구조화된 Dataframe을 제공
  - Dataframe 특화 함수 지원

SUNG KYUN KWAN UNIVERSITY

Numpy 2차원 데이터

- 같은 자료값을 가지는 단순한 수치 정보 중심

- Numpy – 다차원 배열 versus Pandas – Series + Dataframe

SUNG KYUN KWAN UNIVERSITY

Pandas - Series

- Series: 같은 자료형의 데이터를 저장하는 인덱싱된 1차원 배열
  - Series 클래스를 이용하여 데이터를 만듦

```
import numpy as np
import pandas as pd

series_example = pd.Series([100, 50, 30, 10])
series_example
```

```
0    100
1     50
2     30
3     10
dtype: int64
```

```
import numpy as np
import pandas as pd

series_example = pd.Series([100, 50, 30, 10, np.NAN])
series_example
```

```
0    100.0
1     50.0
2     30.0
3     10.0
4      NaN
dtype: float64
```

- 결측치는?
  - Pandas에서 데이터를 감시하고,
    결측치를 리턴하는 함수 제공
  - isna()

```
series_example.isna()
```

```
0    False
1    False
2    False
3    False
4     True
dtype: bool
```

SUNG KYUN KWAN UNIVERSITY

Pandas - Series

- Series 내 데이터 접근: 리스트와 동일

```
series_example[0],series_example[1]
```

```
(100.0, 50.0)
```

- index 갈아끼우기도 가능
  - 기존의 숫자 이외에도 원하는 index 타입이 있을 경우, 정의하여 삽입 가능
  - 오른쪽 예시는 index를 100, 99, 98… 로 바꾸었을 때

```
import numpy as np
import pandas as pd

data = [100, 50, 30, 10, np.NAN]
reverse_series = pd.Series(data, index=[100, 99, 98, 97, 96])
print(reverse_series)
```

```
100    100.0
99      50.0
98      30.0
97      10.0
96       NaN
dtype: float64
```

SUNG KYUN KWAN UNIVERSITY

Pandas – Dataframe

- 딕션으로 각 학생의 중간고사 점수 나타내 보기

```
mid_term = {'StudentA': 95, 'StudentB': 90, 'StudentC': 80, 'StudentD': 60}
mid_term_series = pd.Series(mid_term)
print('Midterm Results')
print(mid_term_series)
```

```
Midterm Results
StudentA    95
StudentB    90
StudentC    80
StudentD    60
dtype: int64
```

- 과제 점수나 기말 고사 점수가 추가되어야 한다면?
  - 1차원적인 Series 구조로는 어려움
  - 새로운 Series 구조의 데이터를 만들어 결합이 필요

    → 2차원 기반의 **Dataframe** 활용

SUNG KYUN KWAN UNIVERSITY

Pandas – Dataframe

- DataFrame 활용
  - df vs print(df)

```
import numpy as np
import pandas as pd

student_se = pd.Series(['StudentA', 'StudentB', 'StudentC', 'StudentD'])
mid_term_series = pd.Series([95, 90, 80, 60])
final_series = pd.Series([80, 95, 88, 90])

df=pd.DataFrame({'Name':student_se, 'Mid': mid_term_series, 'Final': final_series})
df
```

	Name	Mid	Final
0	StudentA	95	80
1	StudentB	90	95
2	StudentC	80	88
3	StudentD	60	90

```
   Name  Mid  Final
0  StudentA   95     80
1  StudentB   90     95
2  StudentC   80     88
3  StudentD   60     90
```

- 가장 높은 중간고사 점수와 평균값 출력

```
max_level = np.argmax(mid_term_series)
print('중간고사 점수가 높은 학생:', student_se[max_level])
```

```
중간고사 점수가 높은 학생: StudentA
```

```
print('중간고사 최고점:', mid_term_series.max(), ', 평균점수:',mid_term_series.mean())
```

```
중간고사 최고점: 95 , 평균점수: 81.25
```

SUNG KYUN KWAN UNIVERSITY

Pandas – csv 활용

- CSV – 콤마로 구분한 변수를 뜻함 (Comma Separated Variables)

  *데이터로 사용한 콤마 vs 구분자 콤마

- read_csv()로 파일 읽기
  - 현재 컴퓨터 같은 경로에 있을 때는 아래 (좌)

```
import numpy as np
import pandas as pd

df = pd.read_csv('kborank.csv')
print(df)
```

	Unnamed: 0	2018	2019	2020	2021	2022
0	Doosan	1	2	3	4	9
1	SSG	2	1	9	6	1
2	Hanwha	3	9	10	10	10
3	Kiwoom	4	3	5	5	3
4	Samsung	5	8	8	1	7
5	Kia	6	7	6	9	5
6	Lotte	7	10	7	8	8
7	LG	8	4	4	3	2
8	KT	9	6	2	2	4
9	NC	10	5	1	7	6

```
import numpy as np
import pandas as pd

df = pd.read_csv('kborank.csv', index_col = 0)
print(df)
```

	2018	2019	2020	2021	2022
Doosan	1	2	3	4	9
SSG	2	1	9	6	1
Hanwha	3	9	10	10	10
Kiwoom	4	3	5	5	3
Samsung	5	8	8	1	7
Kia	6	7	6	9	5
Lotte	7	10	7	8	8
LG	8	4	4	3	2
KT	9	6	2	2	4
NC	10	5	1	7	6

kborank.csv - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
,2018,2019,2020,2021,2022
Doosan,1,2,3,4,9
SSG,2,1,9,6,1
Hanwha,3,9,10,10,10
Kiwoom,4,3,5,5,3
Samsung,5,8,8,1,7
Kia,6,7,6,9,5
Lotte,7,10,7,8,8
LG,8,4,4,3,2
KT,9,6,2,2,4
NC,10,5,1,7,6
```

*온라인 파일도 가능, 다른 경로에 있을 때는 경로 지정 필요

*read_csv() 함수에 index_col이라는 키워드 매개변수에 인자 0 지정 → 첫 번째 열이 인덱스로 사용 (우)

SUNG KYUN KWAN UNIVERSITY

Pandas – csv 활용

- 컬럼값, 인덱스값 출력
  - columns, index 활용

```
[16] df.columns

Index(['2018', '2019', '2020', '2021', '2022'], dtype='object')

df.index

Index(['Doosan', 'SSG', 'Hanwha', 'Kiwoom', 'Samsung', 'Kia', 'Lotte', 'LG',
       'KT', 'NC'],
      dtype='object')
```

- 특정값 출력: 리스트와 동일 – 컬럼값 활용
- 리스트로 변환 요청 – .tolist() 활용

```
df['2022']

Doosan      9
SSG         1
Hanwha     10
Kiwoom      3
Samsung     7
Kia         5
Lotte       8
LG          2
KT          4
NC          6
Name: 2022, dtype: int64
```

SUNG KYUN KWAN UNIVERSITY

Pandas – csv 활용

- 새로운 열 만들기 (axis : 방향)
  - 전체 평균 랭크
  - 최근 3년 평균 랭크

```
df['average_3'] = (df['2020']+df['2021']+df['2022'])/3
print(df)
```

	2018	2019	2020	2021	2022	average	average_3
Doosan	1	2	3	4	9	3.8	5.333333
SSG	2	1	9	6	1	3.8	5.333333
Hanwha	3	9	10	10	10	8.4	10.000000
Kiwoom	4	3	5	5	3	4.0	4.333333
Samsung	5	8	8	1	7	5.8	5.333333
Kia	6	7	6	9	5	6.6	6.666667
Lotte	7	10	7	8	8	8.0	7.666667
LG	8	4	4	3	2	4.2	3.000000
KT	9	6	2	2	4	4.6	2.666667
NC	10	5	1	7	6	5.8	4.666667

```
df['average'] = df.mean(axis=1)
print(df)
```

	2018	2019	2020	2021	2022	average
Doosan	1	2	3	4	9	3.8
SSG	2	1	9	6	1	3.8
Hanwha	3	9	10	10	10	8.4
Kiwoom	4	3	5	5	3	4.0
Samsung	5	8	8	1	7	5.8
Kia	6	7	6	9	5	6.6
Lotte	7	10	7	8	8	8.0
LG	8	4	4	3	2	4.2
KT	9	6	2	2	4	4.6
NC	10	5	1	7	6	5.8

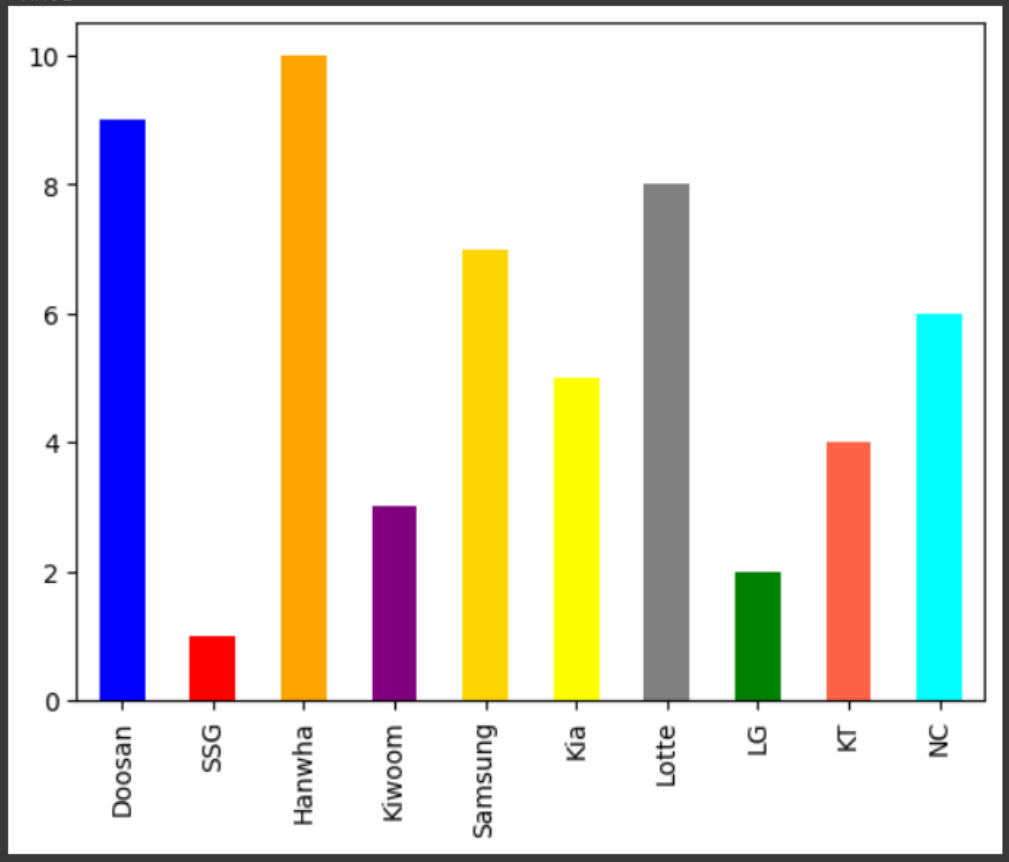
SUNG KYUN KWAN UNIVERSITY

Pandas – Visualization

- Plot 활용 – kind ← plot 종류 (bar: 바 그래프, pie: 파이그래프 등)

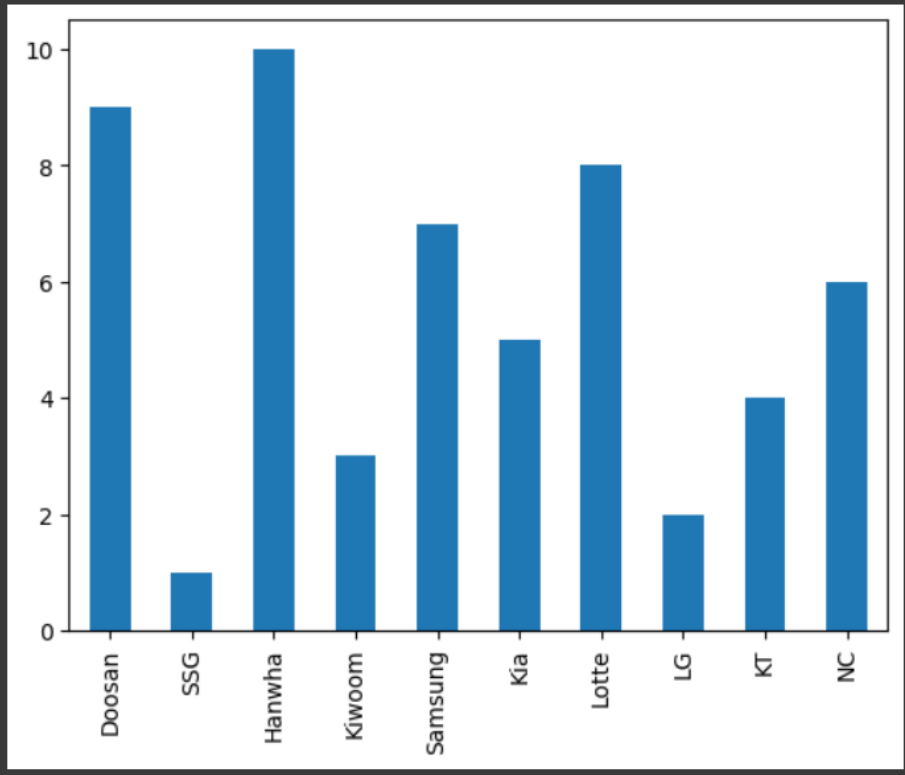
```
df['2022'].plot(kind='bar',color=('blue','red','orange','purple','gold','yellow','gray','green','tomato','cyan'))
```

```
<Axes: >
```




```
df['2022'].plot(kind='bar')
```

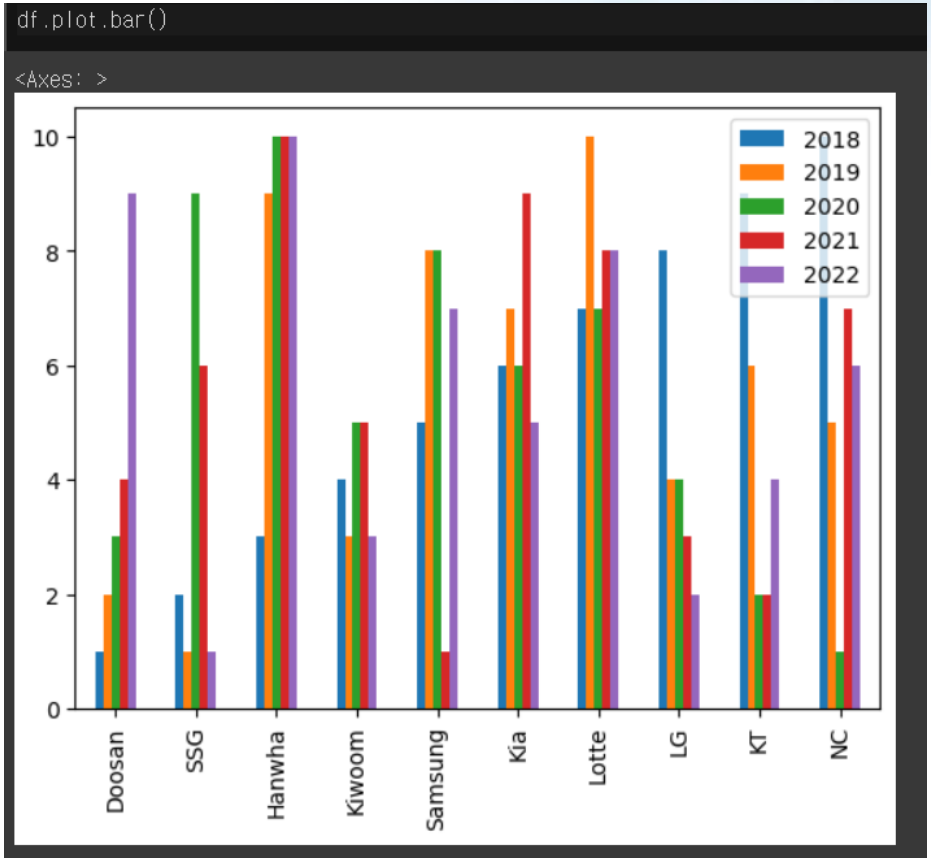
```
<Axes: >
```



SUNG KYUN KWAN UNIVERSITY

Pandas – Visualization

- DataFrame은 matplot 과 호환성이 높아 여러 테이블 제시 가능
  - df.plot.bar() 예시



SUNG KYUN KWAN UNIVERSITY

Pandas

- 슬라이싱과 인덱싱
  - 처음 5행: head() / 마지막 5행: tail()
  - 만일 head()와 tail()에 정수를 인자 제시 → 그 정수만큼의 행을 보여줌

```
df.head(3)
```

	2018	2019	2020	2021	2022
Doosan	1	2	3	4	9
SSG	2	1	9	6	1
Hanwha	3	9	10	10	10

```
df.tail(3)
```

	2018	2019	2020	2021	2022
LG	8	4	4	3	2
KT	9	6	2	2	4
NC	10	5	1	7	6

  - 특정 행 선택 시: loc 활용

    *2개 이상일 때, 이중 사용

```
df.loc['Lotte']
```

```
2018     7
2019    10
2020     7
2021     8
2022     8
Name: Lotte, dtype: int64
```

```
df.loc[['Lotte', 'NC']]
```

	2018	2019	2020	2021	2022
Lotte	7	10	7	8	8
NC	10	5	1	7	6

SUNG KYUN KWAN UNIVERSITY

Pandas

- 슬라이싱과 인덱싱
  - head(), tail() 메소드 : 첫, 마지막 항목 에서 지정 개수 추출
  - [m:n]을 사용 : 지정 구간 m, n의 항목을 추출
  - loc[] : 인덱스에서 특정 레이블이 있는 행 추출
  - iloc[] : 인덱스에서 정수형 인덱스 사용, 특정 위치 행 추출

Thanks

Step 16-1: Pandas and Data Analytics
Instructor: Eunil Park (eunilpark@skku.edu)