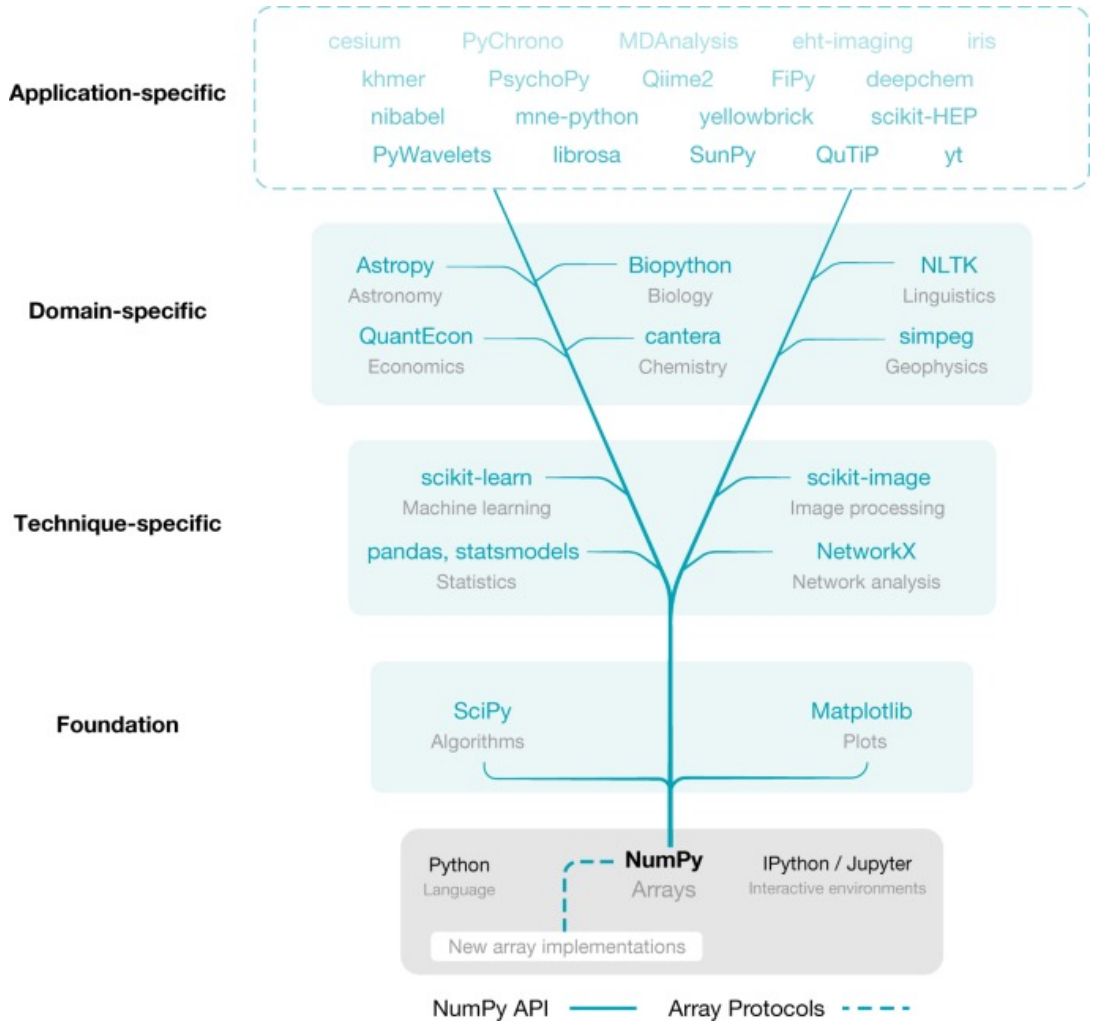
2024년도 1학기
인공지능
Artificial Intelligence

In Last Lecture

❖Numpy 중요성



In Last Lecture

Python For Data Science *Cheat Sheet*

NumPy Basics

Learn Python for Data Science Interactively at www.DataCamp.com

NumPy

The **NumPy** library is the core library for scientific computing in Python. It provides a high-performance multidimensional array object, and tools for working with these arrays.

Use the following import convention:

```
>>> import numpy as np
```

NumPy Arrays

1D array | 2D array | 3D array

axis 1, axis 0 | axis 2, axis 1, axis 0

Creating Arrays

```
>>> a = np.array([1,2,3])
>>> b = np.array([(1.5,2,3), (4,5,6)], dtype = float)
>>> c = np.array([[(1.5,2,3), (4,5,6)], [(3,2,1), (4,5,6)]],
                 dtype = float)
```

Initial Placeholders

`>>> np.zeros((2,4))`	Create an array of zeros
`>>> np.ones((2,3,4),dtype=np.int16)`	Create an array of ones
`>>> d = np.arange(10,25,5)`	Create an array of evenly spaced values (step value)
`>>> np.linspace(0,2,9)`	Create an array of evenly spaced values (number of samples)
`>>> e = np.full((2,2),7)`	Create a constant array
`>>> f = np.eye(2)`	Create a 2X2 identity matrix
`>>> np.random.random((2,2))`	Create an array with random values
`>>> np.empty((3,2))`	Create an empty array

I/O

Saving & Loading On Disk

```
>>> np.save('my_array', a)
>>> np.savez('array.npz', a, b)
>>> np.load('my_array.npy')
```

Saving & Loading Text Files

```
>>> np.loadtxt("myfile.txt")
>>> np.genfromtxt("my_file.csv", delimiter=',')
>>> np.savetxt("myarray.txt", a, delimiter=" ")
```

Data Types

`>>> np.int64`	Signed 64-bit integer types
`>>> np.float32`	Standard double-precision floating point
`>>> np.complex`	Complex numbers represented by 128 floats
`>>> np.bool`	Boolean type storing `TRUE` and `FALSE` values
`>>> np.object`	Python object type
`>>> np.string_`	Fixed-length string type
`>>> np.unicode_`	Fixed-length unicode type

Inspecting Your Array

`>>> a.shape`	Array dimensions
`>>> len(a)`	Length of array
`>>> b.ndim`	Number of array dimensions
`>>> e.size`	Number of array elements
`>>> b.dtype`	Data type of array elements
`>>> b.dtype.name`	Name of data type
`>>> b.astype(int)`	Convert an array to a different type

Asking For Help

```
>>> np.info(np.ndarray.dtype)
```

Array Mathematics

Arithmetic Operations

`>>> g = a - b`	Subtraction
`>>> np.subtract(a,b)`	Subtraction
`>>> b + a`	Addition
`>>> np.add(b,a)`	Addition
`>>> a / b`	Division
`>>> np.divide(a,b)`	Division
`>>> a * b`	Multiplication
`>>> np.multiply(a,b)`	Multiplication
`>>> np.exp(b)`	Exponentiation
`>>> np.sqrt(b)`	Square root
`>>> np.sin(a)`	Print sines of an array
`>>> np.cos(b)`	Element-wise cosine
`>>> np.log(a)`	Element-wise natural logarithm
`>>> e.dot(f)`	Dot product

Comparison

`>>> a == b`	Element-wise comparison
`>>> a < 2`	Element-wise comparison
`>>> np.array_equal(a, b)`	Array-wise comparison

Aggregate Functions

`>>> a.sum()`	Array-wise sum
`>>> a.min()`	Array-wise minimum value
`>>> b.max(axis=0)`	Maximum value of an array row
`>>> b.cumsum(axis=1)`	Cumulative sum of the elements
`>>> a.mean()`	Mean
`>>> b.median()`	Median
`>>> a.corrcoef()`	Correlation coefficient
`>>> np.std(b)`	Standard deviation

Copying Arrays

`>>> h = a.view()`	Create a view of the array with the same data
`>>> np.copy(a)`	Create a copy of the array
`>>> h = a.copy()`	Create a deep copy of the array

Sorting Arrays

`>>> a.sort()`	Sort an array
`>>> c.sort(axis=0)`	Sort the elements of an array's axis

Subsetting, Slicing, Indexing

Also see Lists

Subsetting

`>>> a[2]`	Select the element at the 2nd index
`>>> b[1,2]`	Select the element at row 1 column 2 (equivalent to `b[1][2]`)

Slicing

`>>> a[0:2]`	Select items at index 0 and 1
`>>> b[0:2,1]`	Select items at rows 0 and 1 in column 1
`>>> b[:1]`	Select all items at row 0 (equivalent to `b[0:1, :]`)
`>>> c[1,...]`	Same as `[1,:,:]`
`>>> a[ : :-1]`	Reversed array `a`

Boolean Indexing

`>>> a[a<2]`	Select elements from `a` less than 2

Fancy Indexing

`>>> b[[1, 0, 1, 0],[0, 1, 2, 0]]`	Select elements `(1,0)`,`(0,1)`,`(1,2)` and `(0,0)`
`>>> b[[1, 0, 1, 0]][:,[0,1,2,0]]`	Select a subset of the matrix's rows and columns

Array Manipulation

Transposing Array

`>>> i = np.transpose(b)`	Permute array dimensions
`>>> i.T`	Permute array dimensions

Changing Array Shape

`>>> b.ravel()`	Flatten the array
`>>> g.reshape(3,-2)`	Reshape, but don't change data

Adding/Removing Elements

`>>> h.resize((2,6))`	Return a new array with shape (2,6)
`>>> np.append(h,g)`	Append items to an array
`>>> np.insert(a, 1, 5)`	Insert items in an array
`>>> np.delete(a,[1])`	Delete items from an array

Combining Arrays

`>>> np.concatenate((a,d),axis=0)`	Concatenate arrays
`>>> np.vstack((a,b))`	Stack arrays vertically (row-wise)
`>>> np.r_[e,f]`	Stack arrays vertically (row-wise)
`>>> np.hstack((e,f))`	Stack arrays horizontally (column-wise)
`>>> np.column_stack((a,d))`	Create stacked column-wise arrays
`>>> np.c_[a,d]`	Create stacked column-wise arrays

Splitting Arrays

`>>> np.hsplit(a,3)`	Split the array horizontally at the 3rd index
`>>> np.vsplit(c,2)`	Split the array vertically at the 2nd index

DataCamp
Learn Python for Data Science Interactively

Day

04

Pandas

CONTENTS

A

What is Pandas?

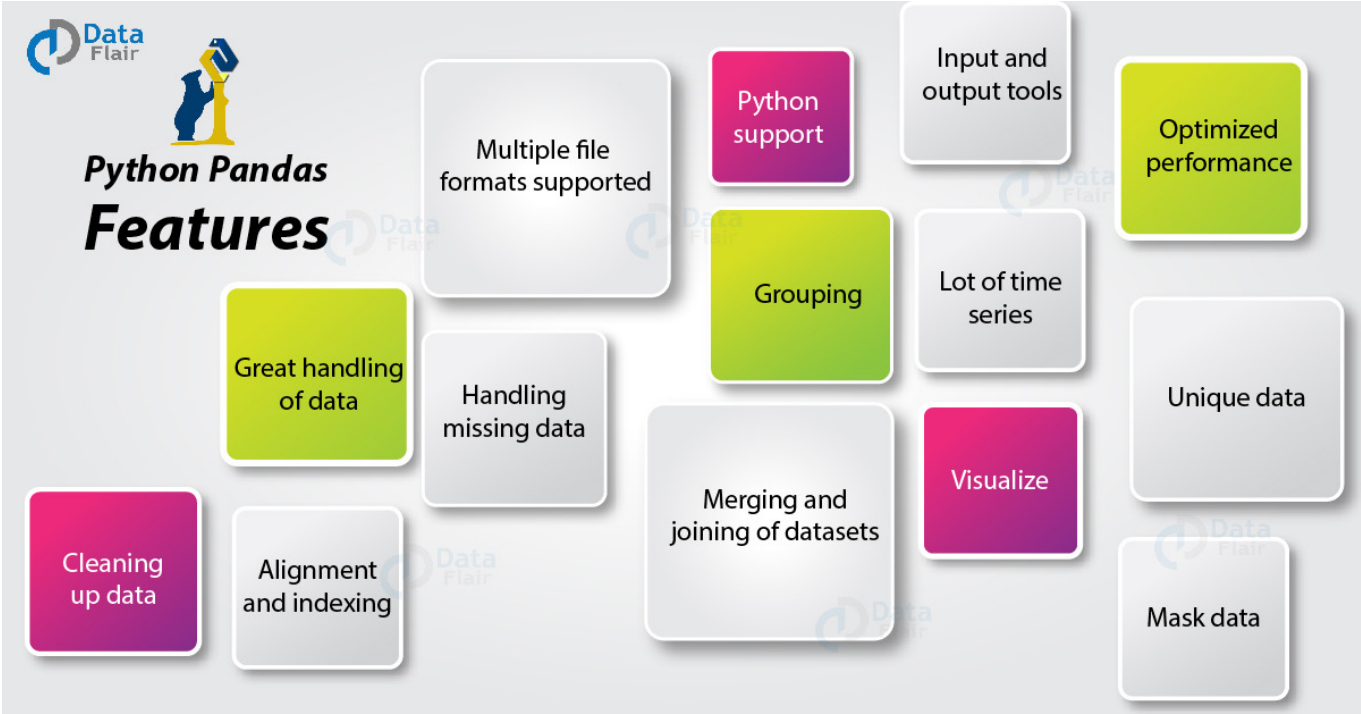
What is Pandas?

- An open-source library that is built on top of NumPy library

- One of the most popular Python libraries for data science
  - Data read, data cleaning, data transforming, and data analysis

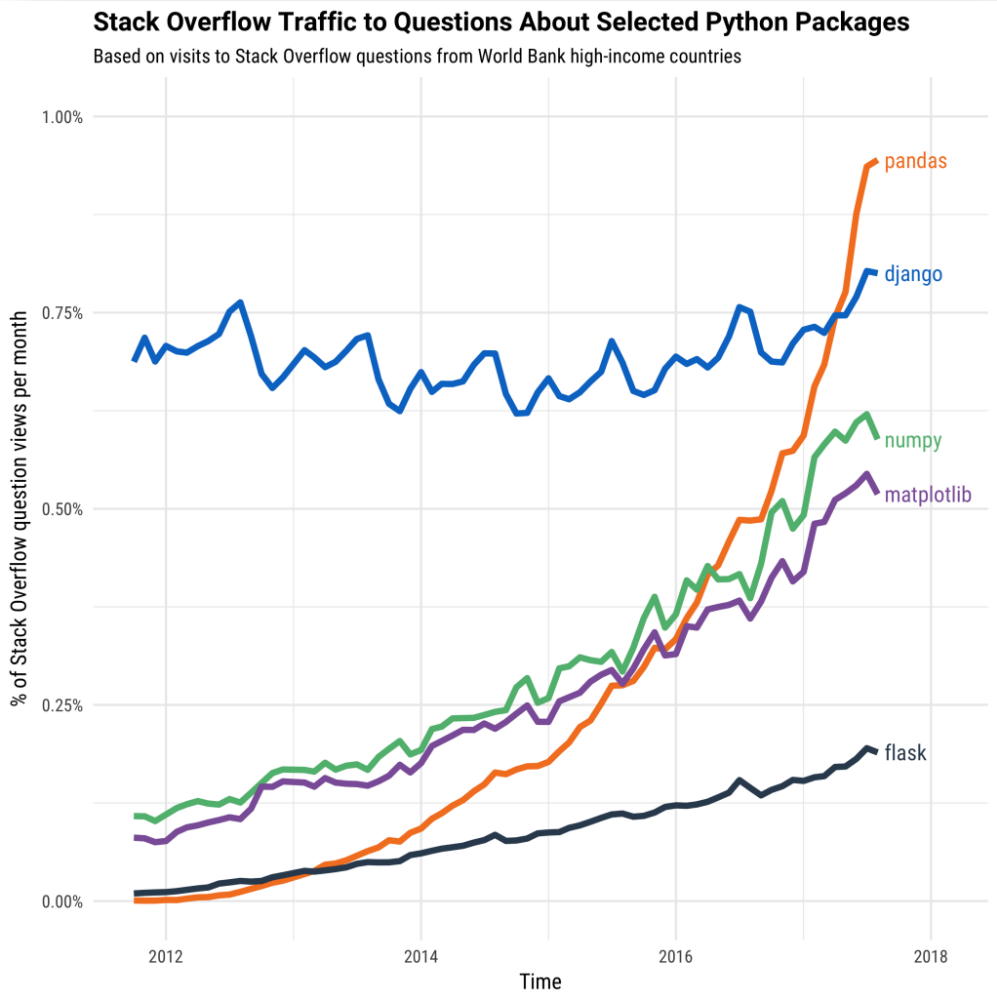
What is Pandas?

❖ Pandas features



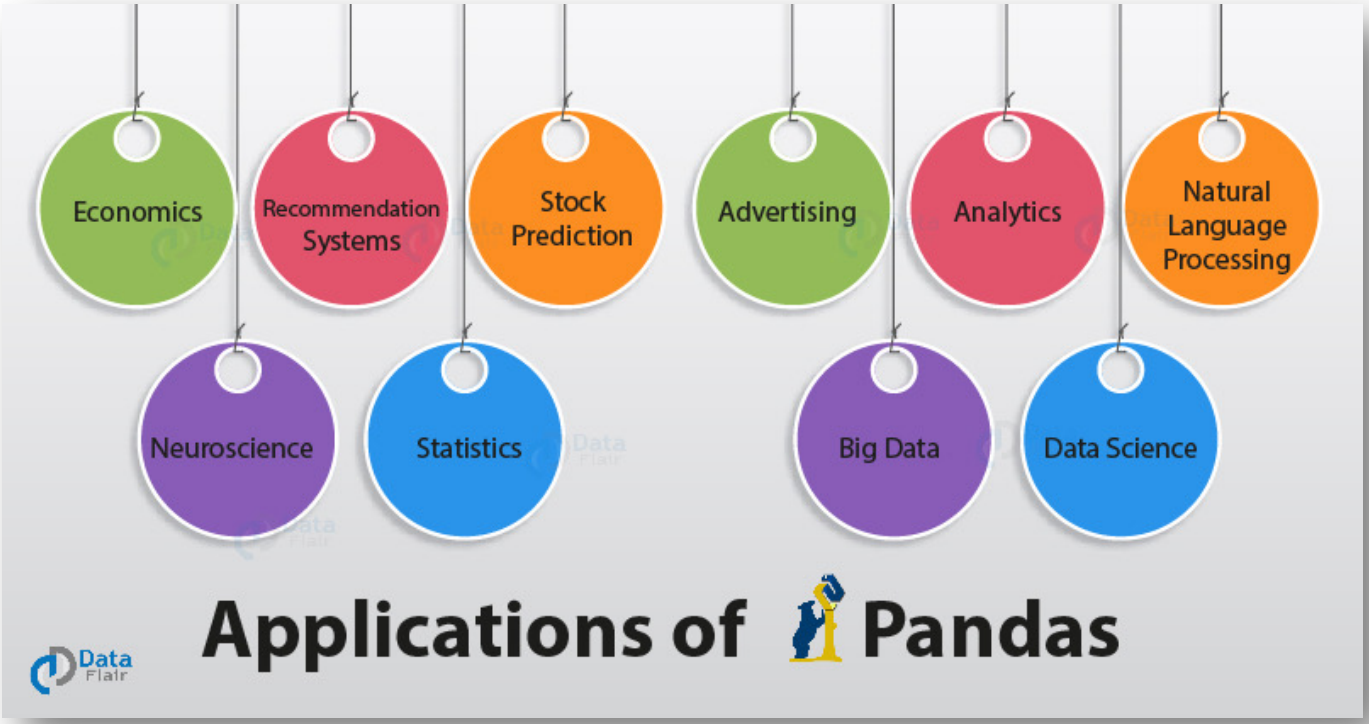
What is Pandas?

- Why Pandas?



What is Pandas?

❖ Pandas Application areas



What is Pandas?

- Jupyter Notebook을 실행
- 명령 프롬프트(CMD)를 실행하고 다음 명령어를 입력
  - pip install pandas
- 다른 라이브러리 또한 설치해야 함
  - Numpy
- pandas 라이브러리 버전 확인

```
import pandas                    #Importing pandas library
print(pandas.__version__)        #Printing pandas library version
```

B

Pandas Data Structure

Pandas Data Structure

❖ Pandas의 두 가지 주요 구성 요소는 Series와 DataFrame이다

❖ Series

- A column

❖ Data frame

- Series 컬렉션으로 구성된 다차원 테이블

Series

	apples
0	3
1	2
2	0
3	1

+

Series

	oranges
0	0
1	3
2	7
3	2

=

DataFrame

	apples	oranges
0	3	0
1	2	3
2	0	7
3	1	2

Pandas Data Structure

❖ Series

- 레이블이 지정된 일차원 배열

- Syntax
  - pandas.Series(data, index, dtype, copy)

  - data
    - Input data in the form of ndarray, list, dict or scalar value
  - index
    - Index of the column
  - dtype
    - Data type
  - copy
    - Copy data

Pandas Data Structure

❖ Series

- Numpy를 이용한 Series 만드는 방법

```
import pandas as pd
import numpy as np

data = np.array(['a','b','c','d'])
series = pd.Series(data, index=[100,101,102,103])

print(series)
```

```
100  a
101  b
102  c
103  d
dtype: object
```

Pandas Data Structure

❖ Series

- dict를 이용한 Series 만드는 방법

```
import pandas as pd
import numpy as np

data = {100 : 'a', 101 : 'b', 102 : 'c', 103 : 'd'}
series = pd.Series(data)

print(series)
```

```
100  a
101  b
102  c
103  d
dtype: object
```

Pandas Data Structure

- **Series**
  - Task: Create a series for the following column

Index	Data
1	'A'
2	'B'
3	'C'
4	'D'
5	'E'

Pandas Data Structure

❖ Series

- Task: Create a series for the following column

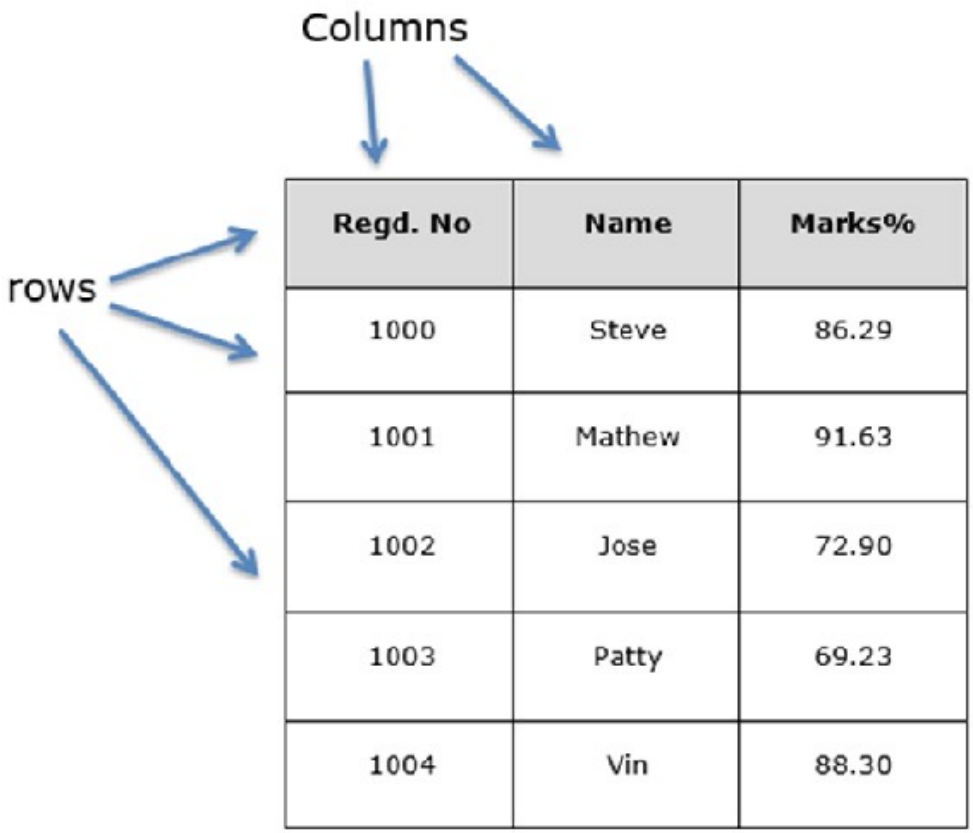
```
import pandas as pd
import numpy as np

data = np.array(['A','B','C','D', 'E'])
series = pd.Series(data, index=[1,2,3,4, 5])

print(series)
```

```
1    A
2    B
3    C
4    D
5    E
dtype: object
```

Pandas Data Structure

❖ Data frame

- 행과 열로 구성된 다차원 테이블



Regd. No	Name	Marks%
1000	Steve	86.29
1001	Mathew	91.63
1002	Jose	72.90
1003	Patty	69.23
1004	Vin	88.30

Pandas Data Structure

❖ Data frame

- Syntax
  - pandas.DataFrame(data, index, columns, dtype, copy)

  - data
    - Input data in the form of series, ndarray, list, dict or scalar value
  - index
    - Index of the data frame
  - columns
    - Name/label of columns
  - dtype
    - Data type
  - copy
    - Copy data

Pandas Data Structure

❖ Data frame

- List를 이용한 data frame 만드는 방법

```
import pandas as pd

data = [['Tim',35],['Sonya',30],['Sunny',34]]

df = pd.DataFrame(data,columns=['Name','Age'],dtype=float)

print(df)
```

```
     Name   Age
0    Tim     35.0
1    Sonya 30.0
2   Sunny  34.0
```

Pandas Data Structure

❖ Data frame

- Dict of series를 이용한 data frame 만드는 방법

```
import pandas as pd

data = {'one' : pd.Series([1, 2, 3], index=['a', 'b', 'c']),
        'two' : pd.Series([1, 2, 3, 4], index=['a', 'b', 'c', 'd'])}

df = pd.DataFrame(data)
print(df)
```

```
   one  two
a  1.0    1
b  2.0    2
c  3.0    3
d  NaN    4
```

Pandas Data Structure

❖ Data frame

- Dict of series를 사용하여 다음 테이블에 대한 data frame 만들어보기

	Artist	Genre	Listeners	Plays
0	Billie Holiday	Jazz	1,300,000	27,000,000
1	Jimi Hendrix	Rock	2,700,000	70,000,000
2	Miles Davis	Jazz	1,500,000	48,000,000
3	SIA	Pop	2,000,000	74,000,000

Pandas Data Structure

❖ Data frame

- dict of series를 사용하여 다음 테이블에 대한 data frame 만들어보기

```
import pandas as pd
import numpy as np

data = {'Artist' : pd.Series(['Billie Holiday', 'Jimi Hendrix', 'Miles Davis', 'SIA']),
        'Genre' : pd.Series(['Jazz', 'Rock', 'Jazz', 'Pop']),
        'Listeners' : pd.Series([1300000, 2700000, 1500000, 2000000]),
        'Plays' : pd.Series([27000000, 70000000, 48000000, 74000000])}

df = pd.DataFrame(data)
print(df)
```

실습

❖ 앞선 예제의 코드를 실행해보고 이해해보자

C

Data Processing in Pandas

Data Processing in Pandas

❖ Pandas의 read_csv 함수를 이용하여 데이터 불러오기

- Save IMDB-Movie-Data.csv file in your local storage

- index_col 속성
  - CSVs don't have indexes like our Data Frames, so all we need to do is just designate the index_col when reading

- In our case, we select "Title" column as index

```
import pandas as pd

movies_df = pd.read_csv(r"D:/IMDB-Movie-Data.csv",
                        index_col="Title")
print(movies_df)
```

Data Processing in Pandas

❖ head() 함수

- Print out a first five rows of your data frame
  - movies_df.head()

- We could also pass a number to head()
  - movies_df.head(10)

❖ tail() 함수

- Print out the last five rows
  - movies_df.tail()

- If you want to see the last two records, then pass a number
  - movies_df.tail(2)

Data Processing in Pandas

- **info()**
  - Provides essential details about your dataset
  - Features
    - Number of rows
    - Number of columns
    - Number of non-null values
    - Type of data in each column
    - How much memory the data frame is using

Data Processing in Pandas

- **info()**
  - movies_df.info()

```
<class 'pandas.core.frame.DataFrame'>
Index: 1000 entries, Guardians of the Galaxy to Nine Lives
Data columns (total 11 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Rank                1000 non-null   int64
 1   Genre               1000 non-null   object
 2   Description         1000 non-null   object
 3   Director            1000 non-null   object
 4   Actors              1000 non-null   object
 5   Year                1000 non-null   int64
 6   Runtime (Minutes)   1000 non-null   int64
 7   Rating              1000 non-null   float64
 8   Votes               1000 non-null   int64
 9   Revenue (Millions)  872 non-null    float64
 10  Metascore           936 non-null    float64
dtypes: float64(3), int64(4), object(4)
memory usage: 93.8+ KB
```

Data Processing in Pandas

❖ shape

- 행과 열의 크기
  - movies_df.shape
  - (1000, 11)

- Used frequently when cleaning and transforming data
  - 예를 들어 일부 기준에 따라 일부 행을 필터링한 다음 제거된 행 수를 빨리 알고 싶을 경우

Data Processing in Pandas

❖ describe()

- Returns descriptive and summary statistics about dataframe
  - describe function for continuous variables
  - movies_df.describe()

	Rank	Year	Runtime (Minutes)	Rating	Votes	Revenue (Millions)	Metascore
count	1000.000000	1000.000000	1000.000000	1000.000000	1.000000e+03	872.000000	936.000000
mean	500.500000	2012.783000	113.172000	6.723200	1.698083e+05	82.956376	58.985043
std	288.819436	3.205962	18.810908	0.945429	1.887626e+05	103.253540	17.194757
min	1.000000	2006.000000	66.000000	1.900000	6.100000e+01	0.000000	11.000000
25%	250.750000	2010.000000	100.000000	6.200000	3.630900e+04	13.270000	47.000000
50%	500.500000	2014.000000	111.000000	6.800000	1.107990e+05	47.985000	59.500000
75%	750.250000	2016.000000	123.000000	7.400000	2.399098e+05	113.715000	72.000000
max	1000.000000	2016.000000	191.000000	9.000000	1.791916e+06	936.630000	100.000000

1.count : 개수
2.mean : 평균
3.std : 표준편차
4.min, max : 최솟값, 최댓값
5.25%, 50%, 75% -> 4분위수 (25% -> 25%의 데이터들이 해당 값보다 낮다)

Data Processing in Pandas

❖ describe()

- We can also use describe() function for categorical variables
  - movies_df['Genre'].describe()

```
count                    1000
unique                    207
top          Action,Adventure,Sci-Fi
freq                       50
Name: Genre, dtype: object
```

Genre
Action,Adventure,Sci-Fi
Adventure,Mystery,Sci-Fi
Horror,Thriller
Animation,Comedy,Family
Action,Adventure,Fantasy
Action,Adventure,Fantasy
Comedy,Drama,Music
Comedy
Action,Adventure,Biography
Adventure,Drama,Romance
Adventure,Family,Fantasy
Biography,Drama,History
Action,Adventure,Sci-Fi
Animation,Adventure,Comedy

데이터 셋에서 총 50번 나옴

  - Features
    - count of rows
    - unique count of categories
    - top category
    - freq of top category

Data Processing in Pandas

❖ columns

- 데이터 셋의 열 이름을 반환
  - `movies_df.columns`

```
Index(['Rank', 'Genre', 'Description', 'Director', 'Actors', 'Year',
       'Runtime (Minutes)', 'Rating', 'Votes', 'Revenue (Millions)',
       'Metascore'],
      dtype='object')c
```

- columns를 통해 열 이름을 변경 할 수 있음

Data Processing in Pandas

❖ index

- 데이터 세트의 인덱스를 출력
  - movies_df.index

```
Index(['Guardians of the Galaxy', 'Prometheus', 'Split', 'Sing',
       'Suicide Squad', 'The Great Wall', 'La La Land', 'Mindhorn',
       'The Lost City of Z', 'Passengers',
       ...
       'Underworld: Rise of the Lycans', 'Taare Zameen Par',
       'Take Me Home Tonight', 'Resident Evil: Afterlife', 'Project X',
       'Secret in Their Eyes', 'Hostel: Part II', 'Step Up 2: The Streets',
       'Search Party', 'Nine Lives'],
      dtype='object', name='Title', length=1000)
```

```
movies_df = pd.read_csv(r"C:\Users\user\Desktop\IMDB-Movie-Data.csv", index_col="Title")
```

Data Processing in Pandas

❖ rename()

- dict을 통해 특정 또는 모든 열의 이름을 변경 할 수 있음

```
movies_df.rename(columns={
        'Runtime (Minutes)': 'Runtime',
}, inplace=True)

movies_df.columns
```

```
Index(['Rank', 'Genre', 'Description', 'Director', 'Actors', 'Year',
        'Runtime', 'Rating', 'Votes', 'Revenue (Millions)', 'Metascore'],
      dtype='object')
```

- Task
  - Change Revenue (Millions) -> 'Revenue_millions'

Data Processing in Pandas

❖ Data frame manipulation

- Output the following table
  - Note: Title is index variable

Title	Genre	Rating
Guardians of the Galaxy	Action,Adventure,Sci-Fi	8.1
Prometheus	Adventure,Mystery,Sci-Fi	7.0
Split	Horror,Thriller	7.3
Sing	Animation,Comedy,Family	7.2
Suicide Squad	Action,Adventure,Fantasy	6.2

- Source code

```
subset = movies_df[['Genre', 'Rating']]

subset.head()
```

Data Processing in Pandas

❖ Data frame manipulation

- Output movies taken in 2012
  - `movies_df[movies_df['Year'] == 2012].head(5)`

Title	Rank	Genre	Description	Director	Actors	Year	Runtime	Rating	Votes	Revenue_millions	Metascore
Prometheus	2	Adventure,Mystery,Sci-Fi	Following clues to the origin of mankind, a te...	Ridley Scott	Noomi Rapace, Logan Marshall-Green, Michael Fa...	2012	124	7.0	485820	126.46	65.0
The Avengers	77	Action,Sci-Fi	Earth's mightiest heroes must come together an...	Joss Whedon	Robert Downey Jr., Chris Evans, Scarlett Johan...	2012	143	8.1	1045588	623.28	69.0
The Dark Knight Rises	125	Action,Thriller	Eight years after the Joker's reign of anarchy...	Christopher Nolan	Christian Bale, Tom Hardy, Anne Hathaway,Gary ...	2012	164	8.5	1222645	448.13	78.0
The Place Beyond the Pines	136	Crime,Drama,Thriller	A motorcycle stunt rider turns to robbing bank...	Derek Cianfrance	Ryan Gosling, Bradley Cooper, Eva Mendes,Craig...	2012	140	7.3	200090	21.38	68.0
Django Unchained	145	Drama,Western	With the help of a German bounty hunter , a fr...	Quentin Tarantino	Jamie Foxx, Christoph Waltz, Leonardo DiCaprio...	2012	165	8.4	1039115	162.80	81.0

Data Processing in Pandas

❖ Data frame manipulation

- Output the movies that have a rating of 8.6
  - `movies_df[movies_df['Rating'] >= 8.6].head(5)`

Title	Rank	Genre	Description	Director	Actors	Year	Runtime	Rating	Votes	Revenue_millions	Metascore
Interstellar	37	Adventure,Drama,Sci-Fi	A team of explorers travel through a wormhole ...	Christopher Nolan	Matthew McConaughey, Anne Hathaway, Jessica Ch...	2014	169	8.6	1047747	187.99	74.0
The Dark Knight	55	Action,Crime,Drama	When the menace known as the Joker wreaks havo...	Christopher Nolan	Christian Bale, Heath Ledger, Aaron Eckhart,Mi...	2008	152	9.0	1791916	533.32	82.0
Inception	81	Action,Adventure,Sci-Fi	A thief, who steals corporate secrets through ...	Christopher Nolan	Leonardo DiCaprio, Joseph Gordon-Levitt, Ellen...	2010	148	8.8	1583625	292.57	74.0
Kimi no na wa	97	Animation,Drama,Fantasy	Two strangers find themselves linked in a biza...	Makoto Shinkai	Ryûnosuke Kamiki, Mone Kamishiraishi, Ryô Nari...	2016	106	8.6	34110	4.68	79.0
Dangal	118	Action,Biography,Drama	Former wrestler Mahavir Singh Phogat and his t...	Nitesh Tiwari	Aamir Khan, Sakshi Tanwar, Fatima Sana Shaikh,...	2016	161	8.8	48969	11.15	NaN

Data Processing in Pandas

❖ Data frame manipulation

- Output movies, which was directed by Christopher Nolan OR Ridley Scott
  - Hint we can use OR (¦) operator
  - movies_df[(movies_df['Director'] == 'Christopher Nolan') ¦ (movies_df['Director'] == 'Ridley Scott')].head(5)

Title	Rank	Genre	Description	Director	Actors	Year	Runtime	Rating	Votes	Revenue_millions	Metascore
Prometheus	2	Adventure,Mystery,Sci-Fi	Following clues to the origin of mankind, a te...	Ridley Scott	Noomi Rapace, Logan Marshall-Green, Michael Fa...	2012	124	7.0	485820	126.46	65.0
Interstellar	37	Adventure,Drama,Sci-Fi	A team of explorers travel through a wormhole ...	Christopher Nolan	Matthew McConaughey, Anne Hathaway, Jessica Ch...	2014	169	8.6	1047747	187.99	74.0
The Dark Knight	55	Action,Crime,Drama	When the menace known as the Joker wreaks havo...	Christopher Nolan	Christian Bale, Heath Ledger, Aaron Eckhart,Mi...	2008	152	9.0	1791916	533.32	82.0
The Prestige	65	Drama,Mystery,Sci-Fi	Two stage magicians engage in competitive one-...	Christopher Nolan	Christian Bale, Hugh Jackman, Scarlett Johanss...	2006	130	8.5	913152	53.08	66.0
Inception	81	Action,Adventure,Sci-Fi	A thief, who steals corporate secrets through ...	Christopher Nolan	Leonardo DiCaprio, Joseph Gordon-Levitt, Ellen...	2010	148	8.8	1583625	292.57	74.0

Data Processing in Pandas

❖ Data frame manipulation

- sort_value() function for sorting by values
  - "by" argument that indicates the column name of data frame to be sorted

- Example of sort_value()
  - movies_df.sort_values(by='Year', ascending=False).head(5)

Title	Rank	Genre	Description	Director	Actors	Year	Runtime	Rating	Votes	Revenue_millions	Metascore
Nine Lives	1000	Comedy,Family,Fantasy	A stuffy businessman finds himself trapped ins...	Barry Sonnenfeld	Kevin Spacey, Jennifer Garner, Robbie Amell,Ch...	2016	87	5.3	12435	19.64	11.0
Free Fire	162	Action,Comedy,Crime	Set in Boston in 1978, a meeting in a deserted...	Ben Wheatley	Sharlto Copley, Brie Larson, Armie Hammer, Cil...	2016	90	7.0	6946	1.80	63.0
Tall Men	648	Fantasy,Horror,Thriller	A challenged man is stalked by tall phantoms i...	Jonathan Holbrook	Dan Crisafulli, Kay Whitney, Richard Garcia, P...	2016	133	3.2	173	NaN	57.0
The Huntsman: Winter's War	235	Action,Adventure,Drama	Eric and fellow warrior Sara, raised as member...	Cedric Nicolas-Troyan	Chris Hemsworth, Jessica Chastain, Charlize Th...	2016	114	6.1	66766	47.95	35.0
Popstar: Never Stop Never Stopping	654	Comedy,Music	When it becomes clear that his solo album is a...	Akiva Schaffer	Andy Samberg, Jorma Taccone,Akiva Schaffer, Sa...	2016	87	6.7	30875	9.39	68.0

실습

❖ 앞선 예제의 코드를 실행해보고 이해해보자

Data Processing in Pandas

❖ Data frame manipulation

- loc[] is used to select rows and columns by Names/Labels
  locate

- iloc[] is used to select rows and columns by Integer Index/Position.
  zero based index position.

Data Processing in Pandas

- **loc[] and iloc[] examples**
  - Let's create the following DataFrame

```
# Pandas.DataFrame.iloc[] usage
import pandas as pd
technologies = {
    'Courses':["Spark","PySpark","Hadoop","Python","pandas"],
    'Fee' :[20000,25000,26000,22000,24000],
    'Duration':['30day','40days','35days','40days','60days'],
    'Discount':[1000,2300,1200,2500,2000]
            }

index_labels=['r1','r2','r3','r4','r5']
df = pd.DataFrame(technologies,index=index_labels)
print(df)
```

Data Processing in Pandas

- **loc[] and iloc[] examples**
  - Let's create the following DataFrame

```
# Outputs:
# r1    Spark  20000    30day      1000
# r2  PySpark  25000   40days      2300
# r3   Hadoop  26000   35days      1200
# r4   Python  22000   40days      2500
# r5   pandas  24000   60days      2000
```

Data Processing in Pandas

❖ loc[] and iloc[] examples

- Select Single Value Using loc[] vs iloc[]

```
# Select Single Row by Index Label
print(df.loc['r2'])  지정한 하나의 행 출력 (r2)

# Select Single Row by Index
print(df.iloc[1])  지정한 하나의 행 출력 (1번째 인덱스)
```

```
# Outputs:
# Courses     PySpark
# Fee         25000
# Duration    40days
# Discount    2300
# Name: r2, dtype: object
```

Data Processing in Pandas

❖ loc[] and iloc[] examples

- In order to select column by label and Index use below

```
# Select Single Column by label
print(df.loc[:, "Courses"])  Courses 열의 전체 행 출력 (인덱스와 함께)

# Select Single Column by Index
print(df.iloc[:, 0])  0번째 열의 전체 행 출력 (인덱스와 함께)
```

```
# Outputs:
#    Courses
# r1    Spark
# r2  PySpark
# r3   Hadoop
# r4   Python
# r5   pandas
```

Data Processing in Pandas

❖ loc[] and iloc[] examples

- To select multiple rows and columns, use the labels or integer index as a list to loc[] and iloc[] attributes

```
# Select Multiple Rows by Label
print(df.loc[['r2','r3']])
```
지정한 행 출력 (r2, r3)

```
# Select Multiple Rows by Index
print(df.iloc[[1,2]])
```
지정한 행 출력 (1, 2번째 인덱스)

```
# Outputs:
#    Courses    Fee Duration  Discount
# r2  PySpark  25000   40days      2300
# r3   Hadoop  26000   35days      1200
```

Data Processing in Pandas

❖ loc[] and iloc[] examples

- Similarly to select multiple columns from pandas DataFrame

```
# Select Multiple Columns by labels
print(df.loc[:, ["Courses","Fee","Discount"]])
```

Courses, Fee, Discount 열의 전체 행 출력 (인덱스와 함께)

```
# Select Multiple Columns by Index
print(df.iloc[:, [0,1,3]])
```

0, 1, 3번째 열의 전체 행 출력 (인덱스와 함께)

```
# Outputs:
#    Courses    Fee  Discount
# r1    Spark  20000      1000
# r2  PySpark  25000      2300
# r3   Hadoop  26000      1200
# r4   Python  22000      2500
# r5   pandas  24000      2000
```

Data Processing in Pandas

❖ loc[] and iloc[] examples

- By using loc[] and iloc[], you can also select rows and columns by range

```
# Select Rows Between two Index Labels
# Includes both r1 and r4 rows
print(df.loc['r1':'r4'])  지정한 행 출력 (r1부터 r4)

# Select Rows Between two Indexs
# Includes Index 0 & Execludes 4
print(df.iloc[0:4])  지정한 행 출력 (0번째 인덱스부터 세번째 인덱스 까지)
```

```
# Outputs:
#    Courses    Fee Duration  Discount
# r1    Spark  20000    30day      1000
# r2  PySpark  25000   40days      2300
# r3   Hadoop  26000   35days      1200
# r4   Python  22000   40days      2500
```

Data Processing in Pandas

❖ loc[] and iloc[] examples

- Selects all columns between Fee and Discount column labels

```
# Select Columns between two Labels
# Includes both 'Fee' and 'Discount' columns
print(df.loc[:,'Fee':'Discount'])
```

Fee 열부터 Discount 열까지 전체 행 출력 (인덱스와 함께)

```
# Select Columns between two Indexes
# Includes Index 1 & Execludes 4
print(df.iloc[:,1:4])
```

1번째 열부터 3번째 열까지 전체 행 출력 (인덱스와 함께)

```
# Outputs:
#      Fee Duration  Discount
# r1  20000    30day      1000
# r2  25000   40days      2300
# r3  26000   35days      1200
# r4  22000   40days      2500
# r5  24000   60days      2000
```

Data Processing in Pandas

❖ loc[] and iloc[] examples

- Selects rows or columns by steps

```
# Select Alternate rows By indeces
print(df.loc['r1':'r4':2])
```
지정한 행 출력 (r1부터 r4, step : 2)

```
# Select Alternate rows By Index
print(df.iloc[0:4:2])
```
지정한 행 출력 (0번째 인덱스부터 세번째 인덱스 까지, step : 2)

```
# Outputs:
#   Courses    Fee Duration  Discount
# r1   Spark  20000    30day      1000
# r3  Hadoop  26000   35days      1200
```

Data Processing in Pandas

❖ loc[] and iloc[] examples

- Selects rows or columns by steps

```
# Select Alternate Columns between two Labels
print(df.loc[:,'Fee':'Discount':2])
```
Fee 열부터 Discount 열까지 전체 행 출력 (인덱스와 함께, step : 2)

```
# Select Alternate Columns between two Indexes
print(df.iloc[:,1:4:2])
```
1번째 열부터 3번째 열까지 전체 행 출력 (인덱스와 함께, step : 2)

```
# Output:
#      Fee  Discount
# r1  20000      1000
# r2  25000      2300
# r3  26000      1200
# r4  22000      2500
# r5  24000      2000
```

실습

❖ 앞선 예제의 코드를 실행해보고 이해해보자

Homework for Lecture 4

❖ Task 1

1. Read usedcars.csv
2. Show summary using head(),tale(), info() and describe()
3. Change name of columns
4. Perform at least three loc and iloc data manipulation
5. Perform at least three conditional data manipulation
6. Challenging task: Create a new column and add it to your dataframe

감사합니다!