Python for Data Science

Proprietary content. © Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited.

Agenda

1. NumPy - Scientific Computing
2. Pandas - Data Manipulation
3. NumPy - Key Operations
4. Pandas - Key Operations
5. Use of merge and join

Proprietary content. © Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited.

Gauge Your Understanding

1. What are the different libraries for data manipulation in Python?
2. What are the key operations that can be performed using NumPy & Pandas?

Proprietary content. © Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited.

Key Libraries for Data Manipulation - NumPy & Pandas

Numpy

- **Num**erical **Py**thon
- Fundamental package for scientific computing
- A powerful N-dimensional array object - ndarray
- Useful in linear algebra, vector calculus, and random number capabilities, etc.

Pandas

- Extremely useful for data manipulation and exploratory analysis
- Offers two major data structures - ***Series*** & ***DataFrame***
- A ***DataFrame*** is made up of several ***Series*** - Each column of a ***DataFrame*** is a ***Series***.
- In a DataFrame, each column can have its own data type unlike NumPy array which creates all entries with the same data type.

Proprietary content. © Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited.

NumPy Key Operations

NumPy provides many useful operations for data manipulation. Some of the most commonly used operations and functions of NumPy are:

Operation	Numpy Function
Declare a NumPy array or convert a list into a NumPy array	array()
Reshape an n-dimensional array without changing the data inside the array	reshape()
Concatenate two or more arrays along a specified axis	concatenate()
Create evenly spaced elements in an interval, particularly useful while working with loops	arange(), linspace()
Working with matrices and perform different operations on them	dot(), transpose(), eye()

Proprietary content. © Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited.

Pandas Key Operations (1/2)

Pandas is one of the most famous data manipulation tool which is built on top of NumPy. Some of the commonly used operations and functions of Pandas are:

Operation	Pandas Function
Load or import the data from different sources/formats	read_csv(), read_excel(),
Information about the data - dimension, column dtypes, non-null values and memory usage	info()
View of basic statistical details of numeric data - quartiles, min, max, mean, std	describe()
Merge two data frames with different types of join - inner join, left join, right join, and full outer join	merge()
Explore data frames by different groups, and apply summary functions on each group	groupby()

Proprietary content. © Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited.

Pandas Key Operations (2/2)

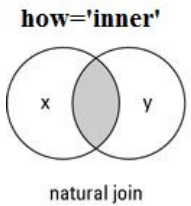
Operation	Pandas Function
To get top n (5 by default) rows of a DataFrame or Series	head()
To concatenate pandas objects (Series or DataFrames) along a particular axis with optional set logic along the other axes	concat()
To get a Series (in descending order) containing counts of unique values	value_counts()
To convert a particular column data type to another data type	astype()
To see the unique values in a particular column	unique()

Proprietary content. © Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited.

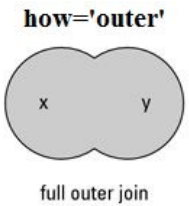
Merge vs Join

- **Join –** The **join** method works best when we are joining dataframes on their indexes (though you can specify another column to join on for the left dataframe).
- **Merge –** The **merge** method is more versatile and allows us to specify columns besides the index to join on for both dataframes.

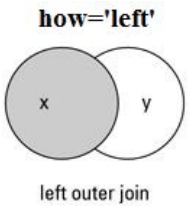
Natural join - Intersection	Full outer join - Union	Left outer join	Right outer join
To keep only rows that match from the data frames how=**'inner'.**	To keep all rows from both data frames, how=**'outer'.**	To include all the rows of your data frame x and only those from y that match how =**'left'.**	To include all the rows of your data frame y and only those from x that match, how=**'right'.**



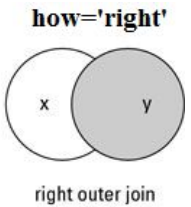
natural join



full outer join



left outer join



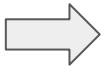
right outer join

Proprietary content. © Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited.

Uses of Merge and Join

id	Name	Age
1	Alex	21
2	Amy	24
3	Allen	25
4	Alice	22
5	Ayoung	19

id	Country	Income
1	India	30000
2	US	24000
3	Brazil	34000
4	UK	14000
5	China	28000



id	Name	Age	Country	Income
1	Alex	21	India	30000
2	Amy	24	US	24000
3	Allen	25	Brazil	34000
4	Alice	22	UK	14000
5	Ayoung	19	China	28000

```
import pandas as pd
left = pd.DataFrame({
   'id':[1,2,3,4,5],
   'Name': ['Alex', 'Amy', 'Allen', 'Alice', 'Ayoung'],
   'Age':['21','24','25','22','19']})
print (left)
```

```
right = pd.DataFrame(
   {'id':[1,2,3,4,5],
   'Country': ['India', 'US', 'Brazil', 'UK', 'China'],
   'Income':['30000','24000','34000','14000','28000']})
print (right)
```

```
print (pd.merge(left,right,on='id'))
```

Proprietary content. © Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited.

Great Learning

Happy Learning !