SNS COLLEGE OF ENGINEERING

Kurumbapalayam(Po), Coimbatore – 641 107
Accredited by NAAC-UGC with 'A' Grade
Approved by AICTE, Recognized by UGC & Affiliated to Anna University, Chennai

Department of Artificial Intelligence and Data Science

Course Name – Big Data Analytics
III Year / V Semester

Unit 2 – Data Science using Python

Topic - Pandas

Pandas

- Pandas is an open source Python package that is most widely used for data science/data analysis and machine learning tasks. It is built on top of another package named Numpy, which provides support for multi-dimensional arrays.

- Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. The name Pandas is derived from the word Panel Data – an Econometrics from Multidimensional data.

- In 2008, developer Wes McKinney started developing pandas when in need of high performance, flexible tool for analysis of data.

- Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Pandas

Key Features of Pandas

- Fast and efficient DataFrame object with default and customized indexing.
- Tools for loading data into in-memory data objects from different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of date sets.
- Label-based slicing, indexing and subsetting of large data sets.
- Columns from a data structure can be deleted or inserted.
- Group by data for aggregation and transformations.
- High performance merging and joining of data.
- Time Series functionality.

Pandas

Why Use Pandas?

- Pandas allows us to analyze big data and make conclusions based on statistical theories.
- Pandas can clean messy data sets, and make them readable and relevant.
- Relevant data is very important in data science.

Pandas generally provide two data structures for manipulating data, They are:

- Series
- DataFrame

Pandas

Series

Pandas Series is a one-dimensional labeled array capable of holding data of any type (integer, string, float, python objects, etc.). The axis labels are collectively called indexes. Pandas Series is nothing but a column in an excel sheet. Labels need not be unique but must be a hashable type. Pandas series can be created using list or dictionaries.

Example - Create a simple Pandas Series from a list:

```
import pandas as pd
a = [1, 7, 2]
myvar = pd.Series(a)
print(myvar)
```

Output

```
0    1
1    7
2    2
dtype: int64
```

Pandas

Labels

If nothing else is specified, the values are labeled with their index number. First value has index 0, second value has index 1 etc.This label can be used to access a specified value.

Create Labels

With the index argument, you can name your own labels.

```
import pandas as pd
a = [1, 7, 2]
myvar = pd.Series(a, index = ["x", "y", "z"])
print(myvar)
```

Output

```
x    1
y    7
z    2
dtype: int64
```

Pandas

Key/Value Objects as Series

We can also use a key/value object, like a dictionary, when creating a Series.

Example

Create a simple Pandas Series from a dictionary:

```
import pandas as pd
calories = {"day1": 420, "day2": 380, "day3": 390}
myvar = pd.Series(calories)
print(myvar)
```

Output

```
Day1 420
Day2 380
Day3 390
```

Pandas

Accessing Data from Series with Position

Data in the series can be accessed similar to that in an ndarray.

Example

Retrieve the first element. As we already know, the counting starts from zero for the array, which means the first element is stored at zeroth position and so on.

```
import pandas as pd
s = pd.Series([1,2,3,4,5],index = ['a','b','c','d','e'])

#retrieve the first element
print s[0]
```

Its **output** is as follows −

```
1
```

Pandas

Example
Retrieve the first three elements in the Series. If a : is inserted in front of it, all items from that index onwards will be extracted. If two parameters (with : between them) is used, items between the two indexes (not including the stop index)

```
import pandas as pd
s = pd.Series([1,2,3,4,5],index = ['a','b','c','d','e'])
#retrieve the first three element
print s[:3]
```

Its output is as follows –

```
a  1
b  2
c  3
```

Pandas

Retrieve the last three elements

```
import pandas as pd
s = pd.Series([1,2,3,4,5],index = ['a','b','c','d','e'])
#retrieve the last three element
print s[-3:]
```

Its **output** is as follows –

```
c  3
d  4
e  5
```

Pandas

Retrieve Data Using Label (Index)

A Series is like a fixed-size dict in that you can get and set values by index label.

Example

Retrieve a single element using index label value.

```
import pandas as pd
s = pd.Series([1,2,3,4,5],index = ['a','b','c','d','e'])
#retrieve a single element
print s['a']
```

Its **output** is as follows −

```
1
```

Pandas

DataFrame

Pandas DataFrame is a two-dimensional size-mutable, potentially heterogeneous tabular data structure with labeled axes (rows and columns). A Data frame is a two-dimensional data structure, i.e., data is aligned in a tabular fashion in rows and columns. Pandas DataFrame consists of three principal components, the data, rows, and columns.

Create DataFrame

A pandas DataFrame can be created using various inputs like –

- Lists
- dict
- Series
- Numpy ndarrays
- Another DataFrame

Pandas

Create a DataFrame from two Series:

```
import pandas as pd
data = {
  "calories": [420, 380, 390],
  "duration": [50, 40, 45]
}
myvar = pd.DataFrame(data)
print(myvar)
```

Output

```
   calories  duration
0       420        50
1       380        40
2       390        45
```

Pandas

Create a DataFrame from Lists

The DataFrame can be created using a single list or a list of lists.

```
import pandas as pd
data = [1,2,3,4,5]
df = pd.DataFrame(data)
print df
```

Its output is as follows −

Output

```
0 1
1 2
2 3
3 4
4 5
```

Pandas

Create a DataFrame from Dict of ndarrays / Lists

All the **ndarrays** must be of same length. If index is passed, then the length of the index should equal to the length of the arrays.

If no index is passed, then by default, index will be range(n), where **n** is the array length.

Example 1

```
import pandas as pd
data = {'Name':['Tom', 'Jack', 'Steve', 'Ricky'],'Age':[28,34,29,42]}
df = pd.DataFrame(data)
print df
```

Its **output** is as follows –

```
   Age     Name
0   28      Tom
1   34     Jack
2   29    Steve
3   42    Ricky
```

Pandas

Create a DataFrame from List of Dicts

List of Dictionaries can be passed as input data to create a DataFrame. The dictionary keys are by default taken as column names.

Example

The following example shows how to create a DataFrame by passing a list of dictionaries.

```
import pandas as pd
data = [{'a': 1, 'b': 2},{'a': 5, 'b': 10, 'c': 20}]
df = pd.DataFrame(data)
print df
```

Its **output** is as follows –

```
   a   b     c
0  1   2   NaN
1  5  10  20.0
```

Pandas

The following example shows how to create a DataFrame by passing a list of dictionaries and the row indices.

```
import pandas as pd
data = [{'a': 1, 'b': 2},{'a': 5, 'b': 10, 'c': 20}]
df = pd.DataFrame(data, index=['first', 'second'])
print df
```

Its **output** is as follows −

```
        a   b      c
first   1   2    NaN
second  5   10   20.0
```

Pandas

Series Basic Functionality

Sr.No.

Attribute or Method & Description

1

axes

Returns a list of the row axis labels

2

dtype

Returns the dtype of the object.

3

empty

Returns True if series is empty.

4

ndim

Returns the number of dimensions of the underlying data, by definition 1.

5

size

Returns the number of elements in the underlying data.

6

values

Returns the Series as ndarray.

7

head()

Returns the first n rows.

8

tail()

Returns the last n rows.

THANK YOU