PYTHON PANDAS - SORTING

There are two kinds of sorting available in Pandas. They are –

- By label
- By Actual Value

Let us consider an example with an output.

```
import pandas as pd
import numpy as np

unsorted_df=pd.DataFrame(np.random.randn(10,2),index=[1,4,6,2,3,5,9,8,0,7],colu
mns=['col2','col1'])
print unsorted_df
```

Its **output** is as follows –

```
        col2       col1
1  -2.063177   0.537527
4   0.142932  -0.684884
6   0.012667  -0.389340
2  -0.548797   1.848743
3  -1.044160   0.837381
5   0.385605   1.300185
9   1.031425  -1.002967
8  -0.407374  -0.435142
0   2.237453  -1.067139
7  -1.445831  -1.701035
```

In **unsorted_df**, the **labels** and the **values** are unsorted. Let us see how these can be sorted.

By Label

Using the **sort_index** method, by passing the axis arguments and the order of sorting, DataFrame can be sorted. By default, sorting is done on row labels in ascending order.

```
import pandas as pd
import numpy as np

unsorted_df = pd.DataFrame(np.random.randn(10,2),index=[1,4,6,2,3,5,9,8,0,7],colu
   mns = ['col2','col1'])

sorted_df=unsorted_df.sort_index()
print sorted_df
```

Its **output** is as follows –

```
        col2       col1
0   0.208464   0.627037
1   0.641004   0.331352
2  -0.038067  -0.464730
3  -0.638456  -0.021466
4   0.014646  -0.737438
5  -0.290761  -1.669827
6  -0.797303  -0.018737
7   0.525753   1.628921
8  -0.567031   0.775951
9   0.060724  -0.322425
```

Order of Sorting

By passing the Boolean value to ascending parameter, the order of the sorting can be controlled. Let us consider the following example to understand the same.

```
import pandas as pd
import numpy as np

unsorted_df = pd.DataFrame(np.random.randn(10,2),index=[1,4,6,2,3,5,9,8,0,7],colu
   mns = ['col2','col1'])

sorted_df = unsorted_df.sort_index(ascending=False)
print sorted_df
```

Its **output** is as follows –

```
         col2        col1
9    0.825697    0.374463
8   -1.699509    0.510373
7   -0.581378    0.622958
6   -0.202951    0.954300
5   -1.289321   -1.551250
4    1.302561    0.851385
3   -0.157915   -0.388659
2   -1.222295    0.166609
1    0.584890   -0.291048
0    0.668444   -0.061294
```

Sort the Columns

By passing the axis argument with a value 0 or 1, the sorting can be done on the column labels. By default, axis=0, sort by row. Let us consider the following example to understand the same.

```
import pandas as pd
import numpy as np

unsorted_df = pd.DataFrame(np.random.randn(10,2),index=[1,4,6,2,3,5,9,8,0,7],colu
   mns = ['col2','col1'])

sorted_df=unsorted_df.sort_index(axis=1)

print sorted_df
```

Its **output** is as follows –

```
          col1        col2
1   -0.291048    0.584890
4    0.851385    1.302561
6    0.954300   -0.202951
2    0.166609   -1.222295
3   -0.388659   -0.157915
5   -1.551250   -1.289321
9    0.374463    0.825697
8    0.510373   -1.699509
0   -0.061294    0.668444
7    0.622958   -0.581378
```

By Value

Like index sorting, **sort_values** is the method for sorting by values. It accepts a 'by' argument which will use the column name of the DataFrame with which the values are to be sorted.

```
import pandas as pd
import numpy as np

unsorted_df = pd.DataFrame({'col1':[2,1,1,1],'col2':[1,3,2,4]})
   sorted_df = unsorted_df.sort_values(by='col1')

print sorted_df
```

Its **output** is as follows –

```
    col1  col2
1    1    3
2    1    2
3    1    4
0    2    1
```

Observe, col1 values are sorted and the respective col2 value and row index will alter along with col1. Thus, they look unsorted.

'by' argument takes a list of column values.

```
import pandas as pd
import numpy as np

unsorted_df = pd.DataFrame({'col1':[2,1,1,1],'col2':[1,3,2,4]})
   sorted_df = unsorted_df.sort_values(by=['col1','col2'])

print sorted_df
```

Its **output** is as follows –

```
  col1 col2
2   1   2
1   1   3
3   1   4
0   2   1
```

Sorting Algorithm

sort_values provides a provision to choose the algorithm from mergesort, heapsort and quicksort. Mergesort is the only stable algorithm.

Live Demo

```
import pandas as pd
import numpy as np

unsorted_df = pd.DataFrame({'col1':[2,1,1,1],'col2':[1,3,2,4]})
sorted_df = unsorted_df.sort_values(by='col1' ,kind='mergesort')

print sorted_df
```

Its **output** is as follows –

```
  col1 col2
1    1    3
2    1    2
3    1    4
0    2    1
```