TheDataLytics

python™

ANUURAG EDLABADKAR

#Prof Ram Meghe College of Engineering & Management, Badnera - Amravati

#https://colab.research.google.com/

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn

# Create Series

var_lst =
['and','as','assert','break','class','continue','def','del','elif']
ser1 = pd.Series(var_lst)
ser1

0         and
1          as
2      assert
3       break
```

```
4       class
5    continue
6         def
7         del
8        elif
dtype: object
```

```
# Create Series

var_tup =
('else','except','False','finally','for','from','global','if','import'
,'in','is','lambda')
ser2 = pd.Series(var_tup)
ser2
```

```
0        else
1      except
2       False
3     finally
4         for
5        from
6      global
7          if
8      import
9          in
10         is
11     lambda
dtype: object
```

```
# Create Series with Dictionary

var_dict =
{'None':'nonlocal','not':'or','pass':'raise','return':'True','try':'wh
ile','with':'yield'}
ser3 = pd.Series(var_dict)
ser3
```

```
None      nonlocal
not             or
pass         raise
return        True
try          while
with         yield
dtype: object
```

```
arr = np.array([91,82,73,64,55,46,37,28,19,0])
ser4 = pd.Series(arr)
ser4
```

```
0    91
1    82
```

```
2    73
3    64
4    55
5    46
6    37
7    28
8    19
9     0
dtype: int64
```

```
arr =
np.array(['Canada','China','Indonesia','India','Japan','Mexico','Taiwa
n','United States'])
ser5 = pd.Series(arr)
ser5
```

```
0           Canada
1            China
2        Indonesia
3            India
4            Japan
5           Mexico
6           Taiwan
7    United States
dtype: object
```

```
ser1.index
```

```
RangeIndex(start=0, stop=9, step=1)
```

```
ser1.dtype
```

```
dtype('O')
```

```
ser1.dtype, ser2.dtype, ser3.dtype, ser4.dtype, ser5.dtype
```

```
(dtype('O'), dtype('O'), dtype('O'), dtype('int64'), dtype('O'))
```

```
ser1.size
```

```
9
```

```
ser1.size, ser2.size, ser3.size, ser4.size, ser5.size
```

```
(9, 12, 6, 10, 8)
```

```
ser1.nbytes
```

```
72
```

```
ser1.nbytes, ser2.nbytes, ser3.nbytes, ser4.nbytes, ser5.nbytes
```

```
(72, 96, 48, 80, 64)
```

```
ser1.shape
```

```
(9,)
```

```
ser1.shape, ser2.shape, ser3.shape, ser4.shape, ser5.shape
```

```
((9,), (12,), (6,), (10,), (8,))
```

```
ser1.ndim
```

```
1
```

```
ser1.ndim, ser2.ndim, ser3.ndim, ser4.ndim, ser5.ndim
```

```
(1, 1, 1, 1, 1)
```

```
len(ser1)
```

```
9
```

```
len(ser1), len(ser2), len(ser3), len(ser4), len(ser5)
```

```
(9, 12, 6, 10, 8)
```

```
ser1.count()
```

```
9
```

```
ser1.count(), ser2.count(), ser3.count(), ser4.count(), ser5.count()
```

```
(9, 12, 6, 10, 8)
```

```
ser1.index, ser2.index, ser3.index, ser4.index, ser5.index
```

```
(RangeIndex(start=0, stop=9, step=1),
 RangeIndex(start=0, stop=12, step=1),
 Index(['None', 'not', 'pass', 'return', 'try', 'with'],
dtype='object'),
 RangeIndex(start=0, stop=10, step=1),
 RangeIndex(start=0, stop=8, step=1))
```

```
ser6 = pd.Series(['a1','b2','c3','d4'], index=['aa','bb','cc','dd'])
ser6
```

```
aa    a1
bb    b2
cc    c3
dd    d4
dtype: object
```

```
ser1
```

```
0         and
1          as
2      assert
```

```
3       break
4       class
5    continue
6         def
7         del
8        elif
dtype: object
```

```
ser1.index = [11,22,33,44,55,66,77,88,99]
ser1
```

```
11         and
22          as
33      assert
44       break
55       class
66    continue
77         def
88         del
99        elif
dtype: object
```

```
v2 = np.random.random(10)
ind2 = np.arange(10,0,-1)
ser8 = pd.Series(v2, ind2)
v2, ind2, ser8
```

```
(array([0.33980705, 0.53160786, 0.02067379, 0.74794524, 0.15088653,
        0.73811167, 0.41810734, 0.95452261, 0.54047065, 0.3751416 ]),
 array([10,  9,  8,  7,  6,  5,  4,  3,  2,  1]),
 10    0.339807
 9     0.531608
 8     0.020674
 7     0.747945
 6     0.150887
 5     0.738112
 4     0.418107
 3     0.954523
 2     0.540471
 1     0.375142
 dtype: float64)
```

```
v2 = np.random.random(10)
ind2 = np.arange(0,10)
ser8 = pd.Series(v2, ind2)
v2, ind2, ser8
```

```
(array([0.92943562, 0.42993222, 0.06646892, 0.34783141, 0.08511081,
        0.45962711, 0.09851398, 0.91672092, 0.24928036, 0.09326218]),
 array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]),
 0    0.929436
```

```
 1    0.429932
 2    0.066469
 3    0.347831
 4    0.085111
 5    0.459627
 6    0.098514
 7    0.916721
 8    0.249280
 9    0.093262
 dtype: float64)
```

```
ser9 = pd.Series(123, index=[1,1,1,1,1,1,1,1,1,1])
ser9
```

```
1    123
1    123
1    123
1    123
1    123
1    123
1    123
1    123
1    123
1    123
1    123
dtype: int64
```

```
ser9 = pd.Series(123, index=[9,8,7,6,5,4,3,2,1,0])
ser9
```

```
9    123
8    123
7    123
6    123
5    123
4    123
3    123
2    123
1    123
0    123
dtype: int64
```

Python Pandas Series Slicing

```
ser8
```

```
0    0.929436
1    0.429932
2    0.066469
3    0.347831
4    0.085111
```

```
5    0.459627
6    0.098514
7    0.916721
8    0.249280
9    0.093262
dtype: float64
```

```
ser8[:]
```

```
0    0.929436
1    0.429932
2    0.066469
3    0.347831
4    0.085111
5    0.459627
6    0.098514
7    0.916721
8    0.249280
9    0.093262
dtype: float64
```

```
ser8[0:10]
```

```
0    0.929436
1    0.429932
2    0.066469
3    0.347831
4    0.085111
5    0.459627
6    0.098514
7    0.916721
8    0.249280
9    0.093262
dtype: float64
```

```
ser8[2:9]
```

```
2    0.066469
3    0.347831
4    0.085111
5    0.459627
6    0.098514
7    0.916721
8    0.249280
dtype: float64
```

```
ser8[1:9:1], ser8[2:8:1], ser8[3:7:1], ser8[4:6:1]
```

```
(1    0.429932
 2    0.066469
 3    0.347831
 4    0.085111
```

```
 5    0.459627
 6    0.098514
 7    0.916721
 8    0.249280
 dtype: float64, 2    0.066469
 3    0.347831
 4    0.085111
 5    0.459627
 6    0.098514
 7    0.916721
 dtype: float64, 3    0.347831
 4    0.085111
 5    0.459627
 6    0.098514
 dtype: float64, 4    0.085111
 5    0.459627
 dtype: float64)
```

```
ser8[0:10:2], ser8[0:10:3], ser8[0:10:4], ser8[0:10:5], ser8[0:10:6]
```

```
(0    0.929436
 2    0.066469
 4    0.085111
 6    0.098514
 8    0.249280
 dtype: float64, 0    0.929436
 3    0.347831
 6    0.098514
 9    0.093262
 dtype: float64, 0    0.929436
 4    0.085111
 8    0.249280
 dtype: float64, 0    0.929436
 5    0.459627
 dtype: float64, 0    0.929436
 6    0.098514
 dtype: float64)
```

```
ser8[::-1], ser8[::-2], ser8[::-3], ser8[::-4], ser8[::-5]
```

```
(9    0.093262
 8    0.249280
 7    0.916721
 6    0.098514
 5    0.459627
 4    0.085111
 3    0.347831
 2    0.066469
 1    0.429932
 0    0.929436
 dtype: float64, 9    0.093262
```

```
 7    0.916721
 5    0.459627
 3    0.347831
 1    0.429932
 dtype: float64, 9    0.093262
 6    0.098514
 3    0.347831
 0    0.929436
 dtype: float64, 9    0.093262
 5    0.459627
 1    0.429932
 dtype: float64, 9    0.093262
 4    0.085111
 dtype: float64)
```

```
ser8[-1:-11:-1], ser8[-2:-10:-1], ser8[-4:-8:-1], ser8[-6:-7:-1],
```

```
(9    0.093262
 8    0.249280
 7    0.916721
 6    0.098514
 5    0.459627
 4    0.085111
 3    0.347831
 2    0.066469
 1    0.429932
 0    0.929436
 dtype: float64, 8    0.249280
 7    0.916721
 6    0.098514
 5    0.459627
 4    0.085111
 3    0.347831
 2    0.066469
 1    0.429932
 dtype: float64, 6    0.098514
 5    0.459627
 4    0.085111
 3    0.347831
 dtype: float64, 4    0.085111
 dtype: float64)
```

```
ser8[-1:], ser8[-2:], ser8[-3:], ser8[-4:], ser8[-5:], ser8[-6:],
ser8[-7:]
```

```
(9    0.093262
 dtype: float64, 8    0.249280
 9    0.093262
 dtype: float64, 7    0.916721
 8    0.249280
 9    0.093262
```

```
 dtype: float64, 6    0.098514
 7    0.916721
 8    0.249280
 9    0.093262
 dtype: float64, 5    0.459627
 6    0.098514
 7    0.916721
 8    0.249280
 9    0.093262
 dtype: float64, 4    0.085111
 5    0.459627
 6    0.098514
 7    0.916721
 8    0.249280
 9    0.093262
 dtype: float64, 3    0.347831
 4    0.085111
 5    0.459627
 6    0.098514
 7    0.916721
 8    0.249280
 9    0.093262
 dtype: float64)
```

```
ser8[:9], ser8[:8], ser8[:7], ser8[:6], ser8[:5], ser8[:4], ser8[:3]
```

```
(0    0.929436
 1    0.429932
 2    0.066469
 3    0.347831
 4    0.085111
 5    0.459627
 6    0.098514
 7    0.916721
 8    0.249280
 dtype: float64, 0    0.929436
 1    0.429932
 2    0.066469
 3    0.347831
 4    0.085111
 5    0.459627
 6    0.098514
 7    0.916721
 dtype: float64, 0    0.929436
 1    0.429932
 2    0.066469
 3    0.347831
 4    0.085111
 5    0.459627
 6    0.098514
 dtype: float64, 0    0.929436
```

```
 1    0.429932
 2    0.066469
 3    0.347831
 4    0.085111
 5    0.459627
 dtype: float64, 0    0.929436
 1    0.429932
 2    0.066469
 3    0.347831
 4    0.085111
 dtype: float64, 0    0.929436
 1    0.429932
 2    0.066469
 3    0.347831
 dtype: float64, 0    0.929436
 1    0.429932
 2    0.066469
 dtype: float64)
```

```
ser4
```

```
0    91
1    82
2    73
3    64
4    55
5    46
6    37
7    28
8    19
9     0
dtype: int64
```

#Python Pandas Series Append

```
cp_ser4 = ser4.copy()
ser4, cp_ser4
```

```
(0    91
 1    82
 2    73
 3    64
 4    55
 5    46
 6    37
 7    28
 8    19
 9     0
 dtype: int64, 0    91
 1    82
 2    73
```

```
 3    64
 4    55
 5    46
 6    37
 7    28
 8    19
 9     0
 dtype: int64)
```

```
print(ser1, ser2)
ser10 = ser1.append(ser2)
ser1, ser2, ser10
```

```
11         and
22          as
33      assert
44       break
55       class
66    continue
77         def
88         del
99        elif
dtype: object 0        else
1      except
2       False
3     finally
4         for
5        from
6      global
7          if
8      import
9          in
10         is
11     lambda
dtype: object
```

```
(11         and
 22          as
 33      assert
 44       break
 55       class
 66    continue
 77         def
 88         del
 99        elif
 dtype: object, 0        else
 1      except
 2       False
 3     finally
 4         for
 5        from
```

```
 6      global
 7          if
 8      import
 9          in
 10         is
 11     lambda
 dtype: object, 11         and
 22           as
 33       assert
 44        break
 55        class
 66    continue
 77         def
 88         del
 99        elif
 0         else
 1        except
 2         False
 3       finally
 4           for
 5          from
 6        global
 7            if
 8        import
 9            in
 10           is
 11       lambda
 dtype: object)
```

```
print(ser3, ser4)
ser11 = ser3.append(ser4)
ser11
```

```
None      nonlocal
not             or
pass         raise
return        True
try          while
with         yield
dtype: object 0    91
1    82
2    73
3    64
4    55
5    46
6    37
7    28
8    19
9     0
dtype: int64
```

```
None      nonlocal
not             or
pass         raise
return        True
try          while
with         yield
0               91
1               82
2               73
3               64
4               55
5               46
6               37
7               28
8               19
9                0
dtype: object
```

```
ser5
```

```
0           Canada
1            China
2        Indonesia
3            India
4            Japan
5           Mexico
6           Taiwan
7    United States
dtype: object
```

```
ser5.drop(1, inplace=True)
```

```
ser5.drop(2, inplace=False)
ser5
```

```
0           Canada
2        Indonesia
3            India
4            Japan
5           Mexico
6           Taiwan
7    United States
dtype: object
```

```
ser5.drop(2, inplace=True)
ser5
```

```
0           Canada
3            India
4            Japan
5           Mexico
6           Taiwan
```

```
7    United States
dtype: object
```

```
ser5 = ser5.append(pd.Series({7:'US', 8:'UK', 9:'UAE'}))
ser5
```

```
0           Canada
3            India
4            Japan
5           Mexico
6           Taiwan
7    United States
7               US
8               UK
9              UAE
dtype: object
```

#Python Pandas Series Operators

```
dict1 = {1:2,3:4,5:6,7:8,9:10}
ser1 = pd.Series(dict1)
dict2 = {1:22,3:44,5:66,7:88,9:110}
ser2 = pd.Series(dict2)
ser1, ser2
```

```
(1     2
 3     4
 5     6
 7     8
 9    10
 dtype: int64, 1     22
 3     44
 5     66
 7     88
 9    110
 dtype: int64)
```

```
ser1.add(ser2)
```

```
1     24
3     48
5     72
7     96
9    120
dtype: int64
```

```
ser1.sub(ser2)
```

```
1    -20
3    -40
5    -60
7    -80
```

```
9   -100
dtype: int64
```

```
ser2.div(ser1)
```

```
1    11.0
3    11.0
5    11.0
7    11.0
9    11.0
dtype: float64
```

```
ser2.mul(ser1)
```

```
1      44
3     176
5     396
7     704
9    1100
dtype: int64
```

#**Python Pandas DataFrames**

Python Pandas DataFrame Import & Export Operation

```
df_euro2012team_csv = pd.read_csv("Euro2012TEAM.csv")
df_euro2012team_csv
```

```
                   Team  Goals  ...  Subs off  Players Used
0               Croatia      4  ...         9            16
1        Czech Republic      4  ...        11            19
2               Denmark      4  ...         7            15
3               England      5  ...        11            16
4                France      3  ...        11            19
5               Germany     10  ...        15            17
6                Greece      5  ...        12            20
7                 Italy      6  ...        18            19
8           Netherlands      2  ...         7            15
9                Poland      2  ...         7            17
10             Portugal      6  ...        14            16
11  Republic of Ireland      1  ...        10            17
12               Russia      5  ...         7            16
13                Spain     12  ...        17            18
14               Sweden      5  ...         9            18
15              Ukraine      2  ...         9            18

[16 rows x 35 columns]
```

```
df_euro2012team_html = pd.read_html("Euro2012TeamHTML.html")
df_euro2012team_html
```

```
[    Unnamed: 0                Team  Goals  ...  Subs on  Subs off
Players Used
 0            0              Croatia      4  ...        9         9
16
 1            1       Czech Republic      4  ...       11        11
19
 2            2              Denmark      4  ...        7         7
15
 3            3              England      5  ...       11        11
16
 4            4               France      3  ...       11        11
19
 5            5              Germany     10  ...       15        15
17
 6            6               Greece      5  ...       12        12
20
 7            7                Italy      6  ...       18        18
19
 8            8          Netherlands      2  ...        7         7
15
 9            9               Poland      2  ...        7         7
17
 10          10             Portugal      6  ...       14        14
16
 11          11  Republic of Ireland      1  ...       10        10
17
 12          12               Russia      5  ...        7         7
16
 13          13                Spain     12  ...       17        17
18
 14          14               Sweden      5  ...        9         9
18
 15          15              Ukraine      2  ...        9         9
18

 [16 rows x 36 columns]]
```

```
df_euro2012team_json = pd.read_json("Euro2012TeamJSON.json")
df_euro2012team_json
```

```
                  Team  Goals  ...  Subs off  Players Used
0              Croatia      4  ...         9            16
1       Czech Republic      4  ...        11            19
2              Denmark      4  ...         7            15
3              England      5  ...        11            16
4               France      3  ...        11            19
5              Germany     10  ...        15            17
6               Greece      5  ...        12            20
7                Italy      6  ...        18            19
8           Netherlands      2  ...         7            15
9                Poland      2  ...         7            17
```

```
10             Portugal      6  ...        14            16
11  Republic of Ireland      1  ...        10            17
12               Russia      5  ...         7            16
13                Spain     12  ...        17            18
14               Sweden      5  ...         9            18
15              Ukraine      2  ...         9            18

[16 rows x 35 columns]
```

```
df_euro2012team_excel = pd.read_excel("Euro2012TeamXLSX.xlsx")
df_euro2012team_excel
```

```
    Unnamed: 0                 Team  Goals  ...  Subs on  Subs off
Players Used
0            0              Croatia      4  ...        9         9
16
1            1       Czech Republic      4  ...       11        11
19
2            2              Denmark      4  ...        7         7
15
3            3              England      5  ...       11        11
16
4            4               France      3  ...       11        11
19
5            5              Germany     10  ...       15        15
17
6            6               Greece      5  ...       12        12
20
7            7                Italy      6  ...       18        18
19
8            8          Netherlands      2  ...        7         7
15
9            9               Poland      2  ...        7         7
17
10          10             Portugal      6  ...       14        14
16
11          11  Republic of Ireland      1  ...       10        10
17
12          12               Russia      5  ...        7         7
16
13          13                Spain     12  ...       17        17
18
14          14               Sweden      5  ...        9         9
18
15          15              Ukraine      2  ...        9         9
18

[16 rows x 36 columns]
```

```
df_euro2012team_excel.to_csv("csv_df_euro2012team_excel.csv")
```

```
df_euro2012team_excel.to_html("html_df_euro2012team_excel.html")

df_euro2012team_excel.to_json("json_df_euro2012team_excel.json")
```

Python Pandas DataFrame General Operation

```
proglang = ['Python','Java','CSS','SQL']
df = pd.DataFrame(proglang)
df
```

```
        0
0  Python
1    Java
2     CSS
3     SQL
```

```
rating = [1,2,3,4]
df[1] = rating
df
```

```
        0  1
0  Python  1
1    Java  2
2     CSS  3
3     SQL  4
```

```
df.columns = ['Programming Language', 'Rating']
df
```

```
  Programming Language  Rating
0               Python       1
1                 Java       2
2                  CSS       3
3                  SQL       4
```

```
# DataFrame Using Dictionary

data = [{'a':'apple','b':'ball','c':'cat'},{'a':'doll','b':'egg'}]
data

df2 = pd.DataFrame(data)
df2

df3 = pd.DataFrame(data, index=['row1','row2'], columns=['a','b'])
df3

df4 = pd.DataFrame(data, index=['row1','row2'], columns=['a','b','c'])
df4

df5 = pd.DataFrame(data, index=['row1','row2'],
```

```
columns=['a','b','c','d'])
df5
```

```
          a     b    c   d
row1  apple  ball  cat NaN
row2   doll   egg  NaN NaN
```

```
df0 = pd.DataFrame({'ID':[1,2,3,4,5],'Name':
['Prajakta','Priyanka','Sayali','Spruha','Sanika']})
df0
```

```
   ID      Name
0   1  Prajakta
1   2  Priyanka
2   3    Sayali
3   4    Spruha
4   5    Sanika
```

```
# Create a DataFrame from Dictionary of Series

dict = {
    'A': pd.Series([1,2,3], index=['a','b','c']),
    'B': pd.Series([1,2,3,4,5], index=['a','b','c','d','e'])
}

df1 = pd.DataFrame(dict)
df1
```

```
     A  B
a  1.0  1
b  2.0  2
c  3.0  3
d  NaN  4
e  NaN  5
```

```
df_dates = pd.date_range(start='2020-01-01', end='2022-02-12')
df_dates
```

```
DatetimeIndex(['2020-01-01', '2020-01-02', '2020-01-03', '2020-01-04',
               '2020-01-05', '2020-01-06', '2020-01-07', '2020-01-08',
               '2020-01-09', '2020-01-10',
               ...
               '2022-02-03', '2022-02-04', '2022-02-05', '2022-02-06',
               '2022-02-07', '2022-02-08', '2022-02-09', '2022-02-10',
               '2022-02-11', '2022-02-12'],
              dtype='datetime64[ns]', length=774, freq='D')
```

```
df_dates = pd.date_range('today', periods=7)
df_dates
```

```
DatetimeIndex(['2022-02-12 13:15:58.194297', '2022-02-13
13:15:58.194297',
```

```
               '2022-02-14 13:15:58.194297', '2022-02-15
13:15:58.194297',
               '2022-02-16 13:15:58.194297', '2022-02-17
13:15:58.194297',
               '2022-02-18 13:15:58.194297'],
              dtype='datetime64[ns]', freq='D')
```

```
df_dates = pd.date_range(start='2021-02-12', periods=7)
df_dates
```

```
DatetimeIndex(['2021-02-12', '2021-02-13', '2021-02-14', '2021-02-15',
               '2021-02-16', '2021-02-17', '2021-02-18'],
              dtype='datetime64[ns]', freq='D')
```

```
m = np.random.random((7,7))
m
```

```
array([[0.74383619, 0.46774943, 0.84353408, 0.14660613, 0.68789805,
        0.39028585, 0.39319436],
       [0.03562929, 0.60871107, 0.04942489, 0.57946828, 0.52207921,
        0.23379314, 0.99442884],
       [0.78496427, 0.14700962, 0.76990515, 0.39437791, 0.82931286,
        0.37766561, 0.78594384],
       [0.34600956, 0.90996251, 0.91489843, 0.63863129, 0.08315084,
        0.41685647, 0.98321884],
       [0.94735581, 0.02124835, 0.19354527, 0.34423463, 0.97649175,
        0.72032213, 0.58607387],
       [0.43334052, 0.32138187, 0.07562768, 0.72217958, 0.23102246,
        0.10083015, 0.73021346],
       [0.70536653, 0.80725288, 0.72772512, 0.47305533, 0.26118419,
        0.55690824, 0.19388046]])
```

```
dframe = pd.DataFrame(m , index=df_dates)
dframe
```

```
                  0         1         2  ...         4         5
6
2021-02-12  0.743836  0.467749  0.843534  ...  0.687898  0.390286
0.393194
2021-02-13  0.035629  0.608711  0.049425  ...  0.522079  0.233793
0.994429
2021-02-14  0.784964  0.147010  0.769905  ...  0.829313  0.377666
0.785944
2021-02-15  0.346010  0.909963  0.914898  ...  0.083151  0.416856
0.983219
2021-02-16  0.947356  0.021248  0.193545  ...  0.976492  0.720322
0.586074
2021-02-17  0.433341  0.321382  0.075628  ...  0.231022  0.100830
0.730213
2021-02-18  0.705367  0.807253  0.727725  ...  0.261184  0.556908
0.193880
```

```
[7 rows x 7 columns]
```

```
dframe.columns = ['C1','C2','C3','C4','C5','C6','C7']
dframe
```

```
                  C1        C2        C3  ...        C5        C6
C7
2021-02-12  0.743836  0.467749  0.843534  ...  0.687898  0.390286
0.393194
2021-02-13  0.035629  0.608711  0.049425  ...  0.522079  0.233793
0.994429
2021-02-14  0.784964  0.147010  0.769905  ...  0.829313  0.377666
0.785944
2021-02-15  0.346010  0.909963  0.914898  ...  0.083151  0.416856
0.983219
2021-02-16  0.947356  0.021248  0.193545  ...  0.976492  0.720322
0.586074
2021-02-17  0.433341  0.321382  0.075628  ...  0.231022  0.100830
0.730213
2021-02-18  0.705367  0.807253  0.727725  ...  0.261184  0.556908
0.193880

[7 rows x 7 columns]
```

```
dframe.index
```

```
DatetimeIndex(['2021-02-12', '2021-02-13', '2021-02-14', '2021-02-15',
               '2021-02-16', '2021-02-17', '2021-02-18'],
              dtype='datetime64[ns]', freq='D')
```

```
dframe.columns
```

```
Index(['C1', 'C2', 'C3', 'C4', 'C5', 'C6', 'C7'], dtype='object')
```

```
dframe.dtypes
```

```
C1    float64
C2    float64
C3    float64
C4    float64
C5    float64
C6    float64
C7    float64
dtype: object
```

```
dframe.sort_values(by='C1')
```

```
                  C1        C2        C3  ...        C5        C6
C7
2021-02-13  0.035629  0.608711  0.049425  ...  0.522079  0.233793
0.994429
```

```
2021-02-15  0.346010  0.909963  0.914898  ...  0.083151  0.416856
0.983219
2021-02-17  0.433341  0.321382  0.075628  ...  0.231022  0.100830
0.730213
2021-02-18  0.705367  0.807253  0.727725  ...  0.261184  0.556908
0.193880
2021-02-12  0.743836  0.467749  0.843534  ...  0.687898  0.390286
0.393194
2021-02-14  0.784964  0.147010  0.769905  ...  0.829313  0.377666
0.785944
2021-02-16  0.947356  0.021248  0.193545  ...  0.976492  0.720322
0.586074

[7 rows x 7 columns]
```

```
dframe.sort_values(by='C7')
```

```
                 C1        C2        C3  ...        C5        C6
C7
2021-02-18  0.705367  0.807253  0.727725  ...  0.261184  0.556908
0.193880
2021-02-12  0.743836  0.467749  0.843534  ...  0.687898  0.390286
0.393194
2021-02-16  0.947356  0.021248  0.193545  ...  0.976492  0.720322
0.586074
2021-02-17  0.433341  0.321382  0.075628  ...  0.231022  0.100830
0.730213
2021-02-14  0.784964  0.147010  0.769905  ...  0.829313  0.377666
0.785944
2021-02-15  0.346010  0.909963  0.914898  ...  0.083151  0.416856
0.983219
2021-02-13  0.035629  0.608711  0.049425  ...  0.522079  0.233793
0.994429

[7 rows x 7 columns]
```

```
dframe.sort_values(by='C1', ascending=True)
```

```
                 C1        C2        C3  ...        C5        C6
C7
2021-02-13  0.035629  0.608711  0.049425  ...  0.522079  0.233793
0.994429
2021-02-15  0.346010  0.909963  0.914898  ...  0.083151  0.416856
0.983219
2021-02-17  0.433341  0.321382  0.075628  ...  0.231022  0.100830
0.730213
2021-02-18  0.705367  0.807253  0.727725  ...  0.261184  0.556908
0.193880
2021-02-12  0.743836  0.467749  0.843534  ...  0.687898  0.390286
0.393194
2021-02-14  0.784964  0.147010  0.769905  ...  0.829313  0.377666
```

```
0.785944
2021-02-16  0.947356  0.021248  0.193545  ...  0.976492  0.720322
0.586074

[7 rows x 7 columns]
```

```
dframe.sort_values(by='C1', ascending=False)
```

```
                 C1        C2        C3  ...        C5        C6
C7
2021-02-16  0.947356  0.021248  0.193545  ...  0.976492  0.720322
0.586074
2021-02-14  0.784964  0.147010  0.769905  ...  0.829313  0.377666
0.785944
2021-02-12  0.743836  0.467749  0.843534  ...  0.687898  0.390286
0.393194
2021-02-18  0.705367  0.807253  0.727725  ...  0.261184  0.556908
0.193880
2021-02-17  0.433341  0.321382  0.075628  ...  0.231022  0.100830
0.730213
2021-02-15  0.346010  0.909963  0.914898  ...  0.083151  0.416856
0.983219
2021-02-13  0.035629  0.608711  0.049425  ...  0.522079  0.233793
0.994429

[7 rows x 7 columns]
```

```
df5
```

```
          a     b    c   d
row1  apple  ball  cat NaN
row2   doll   egg  NaN NaN
```

```
df5.pop('c')
df5
```

```
          a     b   d
row1  apple  ball NaN
row2   doll   egg NaN
```

```
df5.drop(columns='d', inplace=True)
df5
```

```
          a     b
row1  apple  ball
row2   doll   egg
```

```
df
```

```
  Programming Language  Rating
0               Python       1
1                 Java       2
```

```
2                  CSS       3
3                  SQL       4
```

```
df.loc[0], df.loc[1], df.loc[2], df.loc[3],
```

```
(Programming Language    Python
 Rating                       1
 Name: 0, dtype: object, Programming Language    Java
 Rating                       2
 Name: 1, dtype: object, Programming Language    CSS
 Rating                       3
 Name: 2, dtype: object, Programming Language    SQL
 Rating                       4
 Name: 3, dtype: object)
```

```
df.loc[:]
```

```
  Programming Language  Rating
0               Python       1
1                 Java       2
2                  CSS       3
3                  SQL       4
```

```
df.loc[0:1]
```

```
  Programming Language  Rating
0               Python       1
1                 Java       2
```

```
df.loc[1:2]
```

```
  Programming Language  Rating
1                 Java       2
2                  CSS       3
```

```
df.loc[2:3]
```

```
  Programming Language  Rating
2                  CSS       3
3                  SQL       4
```

```
df.iloc[:]
```

```
  Programming Language  Rating
0               Python       1
1                 Java       2
2                  CSS       3
3                  SQL       4
```

```
df.iloc[1]
```

```
Programming Language    Java
Rating                     2
Name: 1, dtype: object
```

```
df.iloc[2]
```

```
Programming Language    CSS
Rating                    3
Name: 2, dtype: object
```

```
df.iloc[:]
```

```
  Programming Language  Rating
0               Python       1
1                 Java       2
2                  CSS       3
3                  SQL       4
```

```
df.iloc[0:3]
```

```
  Programming Language  Rating
0               Python       1
1                 Java       2
2                  CSS       3
```

```
df.iloc[0:4]
```

```
  Programming Language  Rating
0               Python       1
1                 Java       2
2                  CSS       3
3                  SQL       4
```

```
df.iloc[1:2]
```

```
  Programming Language  Rating
1                 Java       2
```

```
df.loc[df.Rating > 2]
```

```
  Programming Language  Rating
2                  CSS       3
3                  SQL       4
```

```
df
```

```
  Programming Language  Rating
0               Python       1
1                 Java       2
2                  CSS       3
3                  SQL       4
```

```
df.columns = ['Lang', 'Rating']
df
```

```
     Lang  Rating
0  Python       1
1    Java       2
2     CSS       3
3     SQL       4
```

```
df.loc[df.Lang=='Java']
```

```
   Lang  Rating
1  Java       2
```

```
dframe
```

```
                  C1        C2        C3  ...        C5        C6
C7
2021-02-12  0.743836  0.467749  0.843534  ...  0.687898  0.390286
0.393194
2021-02-13  0.035629  0.608711  0.049425  ...  0.522079  0.233793
0.994429
2021-02-14  0.784964  0.147010  0.769905  ...  0.829313  0.377666
0.785944
2021-02-15  0.346010  0.909963  0.914898  ...  0.083151  0.416856
0.983219
2021-02-16  0.947356  0.021248  0.193545  ...  0.976492  0.720322
0.586074
2021-02-17  0.433341  0.321382  0.075628  ...  0.231022  0.100830
0.730213
2021-02-18  0.705367  0.807253  0.727725  ...  0.261184  0.556908
0.193880

[7 rows x 7 columns]
```

```
dframe.loc[:, :]
```

```
                  C1        C2        C3  ...        C5        C6
C7
2021-02-12  0.743836  0.467749  0.843534  ...  0.687898  0.390286
0.393194
2021-02-13  0.035629  0.608711  0.049425  ...  0.522079  0.233793
0.994429
2021-02-14  0.784964  0.147010  0.769905  ...  0.829313  0.377666
0.785944
2021-02-15  0.346010  0.909963  0.914898  ...  0.083151  0.416856
0.983219
2021-02-16  0.947356  0.021248  0.193545  ...  0.976492  0.720322
0.586074
2021-02-17  0.433341  0.321382  0.075628  ...  0.231022  0.100830
0.730213
2021-02-18  0.705367  0.807253  0.727725  ...  0.261184  0.556908
```

```
0.193880

[7 rows x 7 columns]
```

```
dframe.loc[:,['C1','C5']]
```

```
                  C1        C5
2021-02-12  0.743836  0.687898
2021-02-13  0.035629  0.522079
2021-02-14  0.784964  0.829313
2021-02-15  0.346010  0.083151
2021-02-16  0.947356  0.976492
2021-02-17  0.433341  0.231022
2021-02-18  0.705367  0.261184
```

```
dframe.loc[['2021-02-12','2021-02-15'],['C1','C5']]
```

```
                  C1        C5
2021-02-12  0.743836  0.687898
2021-02-15  0.346010  0.083151
```

```
dframe.iloc[:,0:3]
```

```
                  C1        C2        C3
2021-02-12  0.743836  0.467749  0.843534
2021-02-13  0.035629  0.608711  0.049425
2021-02-14  0.784964  0.147010  0.769905
2021-02-15  0.346010  0.909963  0.914898
2021-02-16  0.947356  0.021248  0.193545
2021-02-17  0.433341  0.321382  0.075628
2021-02-18  0.705367  0.807253  0.727725
```

```
dframe.iloc[1:2,0:3]
```

```
                  C1        C2        C3
2021-02-13  0.035629  0.608711  0.049425
```