Pandas Tutorial

June 4, 2021

[1]:
```
import pandas as pd
```

[2]:
```
#check pandas version
print (pd.__version__)
```

```
1.1.3
```

#series create, manipulatem querry, delete

[4]:
```
#creating of list
arr = [0, 1, 2, 3, 4]
s1 = pd.Series(arr)
s1
```

[4]:
```
0    0
1    1
2    2
3    3
4    4
dtype: int64
```

[6]:
```
order = [1, 2, 3, 4, 5]
s2 = pd.Series(arr, index=order)
s2
```

[6]:
```
1    0
2    1
3    2
4    3
5    4
dtype: int64
```

[10]:
```
import numpy as np
n = np.random.randn(5) #create a random Narray
index = ['a', 'b', 'c', 'd', 'e']
s2 = pd.Series(n, index=index)
s2
```

[10]:
```
a   -0.112661
b   -1.234770
c   -0.997487
d   -0.321563
e   -2.025659
dtype: float64
```

[11]:
```
#create series from dictionary
d = {'a': 1, 'b': 2, 'c': 3, 'd': 4, 'e': 5}
s3 = pd.Series(d)
s3
```

[11]:
```
a    1
b    2
c    3
d    4
e    5
dtype: int64
```

[12]:
```
# how to modify indexes
print (s1)
s1.index = ['A', 'B', 'C', 'D', 'E']
s1
```

```
0    0
1    1
2    2
3    3
4    4
dtype: int64
```

[12]:
```
A    0
B    1
C    2
D    3
E    4
dtype: int64
```

[13]:
```
# slicing
s1[:3]
```

[13]:
```
A    0
B    1
C    2
dtype: int64
```

[14]:
```
s4 = s1.append(s3)
s4
```

[14]:
```
A    0
B    1
C    2
D    3
E    4
a    1
b    2
c    3
d    4
e    5
dtype: int64
```

[15]:
```
s4.drop('e')
```

[15]:
```
A    0
B    1
C    2
D    3
E    4
a    1
b    2
c    3
d    4
dtype: int64
```

1 Series operations

[16]:
```
arr1 = [0, 1, 2, 3, 4, 5, 7]
arr2 = [6, 7, 8, 9, 5]
```

[17]:
```
s5 = pd.Series(arr2)
s5
```

[17]:
```
0    6
1    7
2    8
3    9
4    5
dtype: int64
```

[19]:
```
s6 = pd.Series(arr1)
s6
```

[19]:
```
0    0
1    1
2    2
3    3
```

```
4    4
5    5
6    7
dtype: int64
```

[20]:
```
s5.add(s6) # addition elements of s5 to s6
```

[20]:
```
0     6.0
1     8.0
2    10.0
3    12.0
4     9.0
5     NaN
6     NaN
dtype: float64
```

[21]:
```
s5.sub(s6) # substraction of s6 from s5
```

[21]:
```
0    6.0
1    6.0
2    6.0
3    6.0
4    1.0
5    NaN
6    NaN
dtype: float64
```

[25]:
```
s7 = s5.mul(s6) #multiplication
s7
```

[25]:
```
0     0.0
1     7.0
2    16.0
3    27.0
4    20.0
5     NaN
6     NaN
dtype: float64
```

[23]:
```
s5.div(s6) # divination
```

[23]:
```
0     inf
1    7.00
2    4.00
3    3.00
4    1.25
5     NaN
6     NaN
```

```
dtype: float64
```

[26]:
```
print ('median', s7.median())
print ('max', s7.max())
print ('min', s7.min())
```

```
median 16.0
max 27.0
min 0.0
```

2 create Dataframe

[27]:
```
dates = pd.date_range('today', periods=6) # Define time sequence as index
dates
```

[27]:
```
DatetimeIndex(['2021-06-04 11:08:09.645088', '2021-06-05 11:08:09.645088',
               '2021-06-06 11:08:09.645088', '2021-06-07 11:08:09.645088',
               '2021-06-08 11:08:09.645088', '2021-06-09 11:08:09.645088'],
              dtype='datetime64[ns]', freq='D')
```

[30]:
```
num_arr = np.random.randn(6, 4) # Import numpy random array
num_arr  # (raws, columns)
```

[30]:
```
array([[ 1.16641881,  1.23424124, -1.36936252,  0.14678799],
       [-2.38097325,  1.6575146 ,  1.03401598,  1.17192031],
       [ 1.13738137,  0.63143114, -0.90741854,  0.31007123],
       [ 0.10158433, -1.89138978,  0.93372665,  2.30973981],
       [-0.49296046, -0.00291236, -0.22033181, -0.35933172],
       [ 1.44744653, -1.24258304, -1.69196529,  1.33171116]])
```

[31]:
```
columns = ['A', 'B', 'C', 'D'] # Use the table as the column name
columns
```

[31]:
```
['A', 'B', 'C', 'D']
```

[32]:
```
df1 = pd.DataFrame(num_arr, index=dates, columns=columns)
df1
```

[32]:

	A	B	C	D
2021-06-04 11:08:09.645088	1.166419	1.234241	-1.369363	0.146788
2021-06-05 11:08:09.645088	-2.380973	1.657515	1.034016	1.171920
2021-06-06 11:08:09.645088	1.137381	0.631431	-0.907419	0.310071
2021-06-07 11:08:09.645088	0.101584	-1.891390	0.933727	2.309740
2021-06-08 11:08:09.645088	-0.492960	-0.002912	-0.220332	-0.359332
2021-06-09 11:08:09.645088	1.447447	-1.242583	-1.691965	1.331711

[36]:
```
# create dataferame with dictionary array

data = {'animal': ['cat', 'cat', 'snake', 'dog', 'dog', 'cat', 'snake', 'cat',␣
 ↪'dog', 'dog'],
       'age': [2.5,3,  0.5, np.nan, 5, 2, 4.5, np.nan, 7, 3],
        'visits': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
        'priority': ['yes', 'yes', 'no', 'yes', 'no', 'no', 'no', 'yes', 'no',␣
 ↪'no']
       }
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i','j']

df2 = pd.DataFrame(data, index=labels)
df2
```

[36]:

	animal	age	visits	priority
a	cat	2.5	1	yes
b	cat	3.0	3	yes
c	snake	0.5	2	no
d	dog	NaN	3	yes
e	dog	5.0	2	no
f	cat	2.0	3	no
g	snake	4.5	1	no
h	cat	NaN	1	yes
i	dog	7.0	2	no
j	dog	3.0	1	no

[37]:
```
# see datatypes of array
df2.dtypes
```

[37]:
```
animal       object
age         float64
visits        int64
priority     object
dtype: object
```

[40]:
```
df3 = df2.head(6)
df3
```

[40]:

	animal	age	visits	priority
a	cat	2.5	1	yes
b	cat	3.0	3	yes
c	snake	0.5	2	no
d	dog	NaN	3	yes
e	dog	5.0	2	no
f	cat	2.0	3	no

[42]:
```
df2.tail(3)
```

[42]:
```
  animal  age  visits priority
h    cat  NaN       1      yes
i    dog  7.0       2       no
j    dog  3.0       1       no
```

[44]:
```
print (df2.index)
df2.columns
```

[44]:
```
Index(['animal', 'age', 'visits', 'priority'], dtype='object')
```

[45]:
```
df2.values
```

[45]:
```
array([['cat', 2.5, 1, 'yes'],
       ['cat', 3.0, 3, 'yes'],
       ['snake', 0.5, 2, 'no'],
       ['dog', nan, 3, 'yes'],
       ['dog', 5.0, 2, 'no'],
       ['cat', 2.0, 3, 'no'],
       ['snake', 4.5, 1, 'no'],
       ['cat', nan, 1, 'yes'],
       ['dog', 7.0, 2, 'no'],
       ['dog', 3.0, 1, 'no']], dtype=object)
```

[46]:
```
df2.describe()
```

[46]:
```
             age     visits
count  8.000000  10.000000
mean   3.437500   1.900000
std    2.007797   0.875595
min    0.500000   1.000000
25%    2.375000   1.000000
50%    3.000000   2.000000
75%    4.625000   2.750000
max    7.000000   3.000000
```

[47]:
```
df2.T
```

[47]:
```
            a    b      c    d    e    f      g    h    i    j
animal    cat  cat  snake  dog  dog  cat  snake  cat  dog  dog
age       2.5    3    0.5  NaN    5    2    4.5  NaN    7    3
visits      1    3      2    3    2    3      1    1    2    1
priority  yes  yes     no  yes   no   no     no  yes   no   no
```

[49]:
```
df2.sort_values(by='animal')
```

[49]:
```
  animal  age  visits priority
a    cat  2.5       1      yes
b    cat  3.0       3      yes
```

```
f    cat  2.0       3       no
h    cat  NaN       1      yes
d    dog  NaN       3      yes
e    dog  5.0       2       no
i    dog  7.0       2       no
j    dog  3.0       1       no
c  snake  0.5       2       no
g  snake  4.5       1       no
```

[53]:
```
# slicing dataframe
df2.sort_values(by='age')[1:3]
```

[53]:
```
  animal  age  visits priority
f    cat  2.0       3       no
a    cat  2.5       1      yes
```

[55]:
```
# quary dataframe by tag
df2[['age', 'visits']]
```

[55]:
```
   age  visits
a  2.5       1
b  3.0       3
c  0.5       2
d  NaN       3
e  5.0       2
f  2.0       3
g  4.5       1
h  NaN       1
i  7.0       2
j  3.0       1
```

[56]:
```
df2.iloc[1:3]
```

[56]:
```
  animal  age  visits priority
b    cat  3.0       3      yes
c  snake  0.5       2       no
```

[58]:
```
df3 = df2.copy()
df3
```

[58]:
```
  animal  age  visits priority
a    cat  2.5       1      yes
b    cat  3.0       3      yes
c  snake  0.5       2       no
d    dog  NaN       3      yes
e    dog  5.0       2       no
f    cat  2.0       3       no
g  snake  4.5       1       no
```

```
h    cat  NaN       1      yes
i    dog  7.0       2       no
j    dog  3.0       1       no
```

[59]:
```
df3.isnull()
```

[59]:
```
    animal    age  visits  priority
a   False  False   False     False
b   False  False   False     False
c   False  False   False     False
d   False   True   False     False
e   False  False   False     False
f   False  False   False     False
g   False  False   False     False
h   False   True   False     False
i   False  False   False     False
j   False  False   False     False
```

[61]:
```
df3.loc['f', 'age'] = 1.5
df3
```

[61]:
```
  animal  age  visits priority
a    cat  2.5       1      yes
b    cat  3.0       3      yes
c  snake  0.5       2       no
d    dog  NaN       3      yes
e    dog  5.0       2       no
f    cat  1.5       3       no
g  snake  4.5       1       no
h    cat  NaN       1      yes
i    dog  7.0       2       no
j    dog  3.0       1       no
```

[63]:
```
df3[['age']].mean()
```

[63]:
```
age    3.375
dtype: float64
```

[65]:
```
df3['visits'].mean() # min(), max(), sum()
```

[65]:
```
1.9
```

[66]:
```
df3.sum()
```

[66]:
```
animal      catcatsnakedogdogcatsnakecatdogdog
age                                         27
visits                                      19
priority              yesyesnoyesnononoyesnono
```

```
dtype: object
```

[68]:
```
string = pd.Series(['A', 'C', 'D', 'Aaa', 'BaCa', np.nan, 'CBA', 'cow', 'owl' ])
string.str.lower()  #upper()
```

[68]:
```
0       a
1       c
2       d
3     aaa
4    baca
5     NaN
6     cba
7     cow
8     owl
dtype: object
```

3 Operations for DataFrame missing values

[71]:
```
df4 = df3.copy()
meanAge = df4['age'].mean()
df4['age'].fillna(meanAge)  # to fill all positions Nan with meanAge
```

[71]:
```
a    2.500
b    3.000
c    0.500
d    3.375
e    5.000
f    1.500
g    4.500
h    3.375
i    7.000
j    3.000
Name: age, dtype: float64
```

[73]:
```
df5 = df3.copy()
df5
```

[73]:

	animal	age	visits	priority
a	cat	2.5	1	yes
b	cat	3.0	3	yes
c	snake	0.5	2	no
d	dog	NaN	3	yes
e	dog	5.0	2	no
f	cat	1.5	3	no
g	snake	4.5	1	no
h	cat	NaN	1	yes

```
i    dog  7.0       2       no
j    dog  3.0       1       no
```

[74]:
```
df5.dropna(how='any')
```

[74]:
```
   animal  age  visits priority
a     cat  2.5       1      yes
b     cat  3.0       3      yes
c   snake  0.5       2       no
e     dog  5.0       2       no
f     cat  1.5       3       no
g   snake  4.5       1       no
i     dog  7.0       2       no
j     dog  3.0       1       no
```

4 DataFrame file operations

[75]:
```
df3.to_csv('Animals.csv')
#df_animal1 = pd.read_csv('Animal.csv')
```

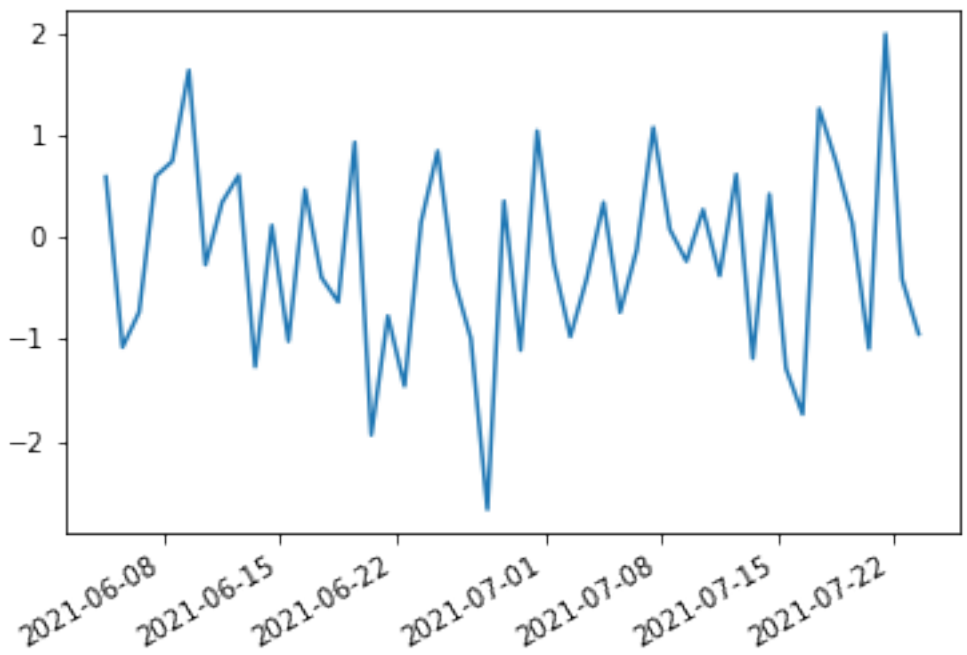
[76]:
```
#df3.to_excel('animal.xlsx', sheet_name='Sheet1')
#df_animal2.read_excel('animal.xlsx', 'Sheet1', index_col=None, na_values=['NA'])
```

5 Visualisation in Pandas

[80]:
```
# series and dataframe line chart
import numpy as np
%matplotlib inline

ts = pd.Series(np.random.randn(50), index=pd.date_range('today', periods=50))
ts.cumsum()
ts.plot()
```

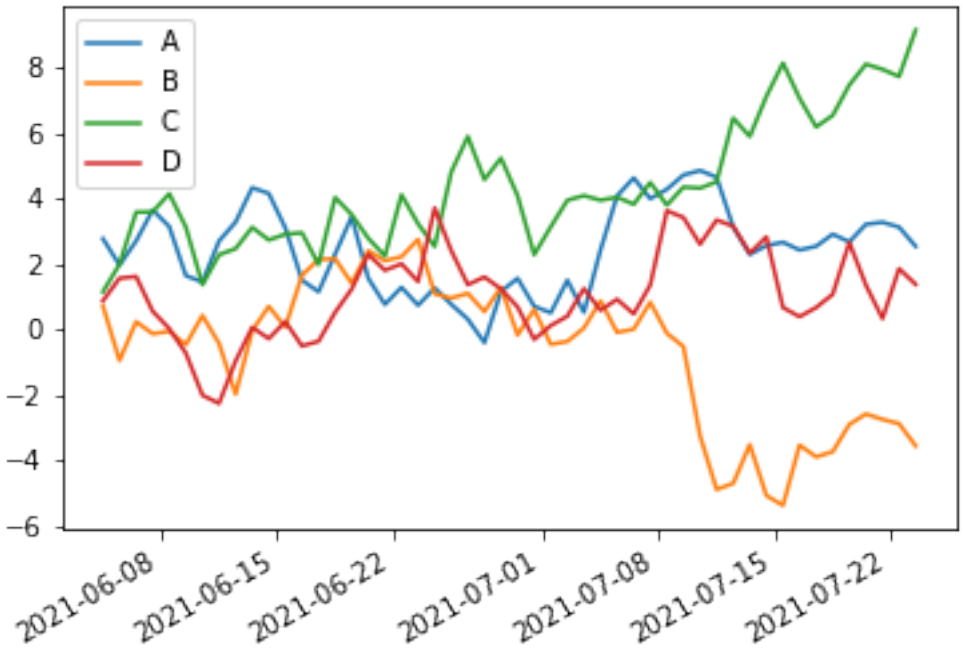
[80]:
```
<AxesSubplot:>
```



[81]:
```
# with dataframe

df = pd.DataFrame(np.random.randn(50, 4), index=ts.index, columns=['A', 'B',␣
↪'C', 'D'])
df = df.cumsum()
df.plot()
```

[81]: <AxesSubplot:>



6 Rmove repeated data using pandas

[83]:
```
df = pd.DataFrame({'A': [1, 2, 2, 2, 4, 4, 5, 5, 6, 6, 7 ,8, 8]})
df.loc[  df['A'].shift() != df['A']  ]
```

[83]:

	A
0	1
1	2
4	4
6	5
8	6
10	7
11	8

[ ]:
```

```