Data Visualization in Python

July 30, 2023

1 Pandas Tutorial

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

[2]:
```
print(pd.__version__)
```

```
1.5.3
```

2 Series create,manipulate,querry,delete

[3]:
```
# create a series from a list
arr = [0,1,2,3,4]
s1 = pd.Series(arr)
s1
```

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

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

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

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

[5]:
```
a   -0.748737
b    0.435630
c   -0.205865
d    0.219608
e    1.490356
dtype: float64
```

[6]:
```
#create series from dictionary
d = {'a':1,'b':2,'c':3,'d':4,'e':5}
s3 = pd.Series(d)
s3
```

[6]:
```
a    1
b    2
c    3
d    4
e    5
dtype: int64
```

[7]:
```
# you can modify the index of series
print(s1)
s1.index = ['A','B','C','D','E']
s1
```

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

[7]:
```
A    0
B    1
C    2
D    3
E    4
dtype: int64
```

[8]:
```
# slicing
a = s1[:3]
a
```

[8]:
```
A    0
B    1
C    2
dtype: int64
```

[9]:
```
s1[:-1]
```

```
[9]: A    0
     B    1
     C    2
     D    3
     dtype: int64
```

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

```
C:\Users\23031\AppData\Local\Temp\ipykernel_7596\3585235679.py:1: FutureWarning:
The series.append method is deprecated and will be removed from pandas in a
future version. Use pandas.concat instead.
  s4 = s1.append(s3)
```

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

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

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

```
[12]: s4
```

```
[12]: A    0
      B    1
      C    2
      D    3
      E    4
      a    1
      b    2
```

```
c    3
d    4
e    5
dtype: int64
```

3 Series Operations

[13]:
```
arr1=[0,1,2,3,4,5,7]
arr2=[6,7,8,9,5]
```

[14]:
```
s5 = pd.Series (arr2)
s5
```

[14]:
```
0    6
1    7
2    8
3    9
4    5
dtype: int64
```

[15]:
```
s6 = pd.Series(arr1)
s6
```

[15]:
```
0    0
1    1
2    2
3    3
4    4
5    5
6    7
dtype: int64
```

[16]:
```
s5.add(s6)
```

[16]:
```
0     6.0
1     8.0
2    10.0
3    12.0
4     9.0
5     NaN
6     NaN
dtype: float64
```

[17]:
```
s5.sub(s6)
```

[17]:
```
0    6.0
1    6.0
```

```
2    6.0
3    6.0
4    1.0
5    NaN
6    NaN
dtype: float64
```

[18]:
```
s5.mul(s6)
```

[18]:
```
0     0.0
1     7.0
2    16.0
3    27.0
4    20.0
5     NaN
6     NaN
dtype: float64
```

[19]:
```
s5.div(s6)
```

[19]:
```
0     inf
1    7.00
2    4.00
3    3.00
4    1.25
5     NaN
6     NaN
dtype: float64
```

[21]:
```
print('median',s6.median())
print('max',s6.max())
print('min',s6.min())
```

```
median 3.0
max 7
min 0
```

# Create Dataframe

[22]:
```
dates = pd.date_range('today',periods=6)
dates
```

[22]:
```
DatetimeIndex(['2023-07-29 12:49:48.724210', '2023-07-30 12:49:48.724210',
               '2023-07-31 12:49:48.724210', '2023-08-01 12:49:48.724210',
               '2023-08-02 12:49:48.724210', '2023-08-03 12:49:48.724210'],
              dtype='datetime64[ns]', freq='D')
```

[23]:
```
num_arr = np.random.randn(6,4)
num_arr
```

```
[23]: array([[ 1.30412668, -0.04158467,  1.19830198,  0.80064344],
             [ 0.90537077, -0.05361948, -2.43870973, -1.70687568],
             [ 1.15557549, -0.59952381, -0.02044303, -1.34126715],
             [-0.2653217 , -1.18509535, -0.64807288,  0.39102997],
             [-0.15061874, -0.44451598,  1.00121652,  1.56433946],
             [ 0.25923383,  0.67575659, -2.32349339, -1.57883324]])
```

```
[24]: columns = ['A','B','C','D']

      df1 = pd.DataFrame(num_arr,index = dates,columns=columns)
      df1
```

```
[24]:                                    A         B         C         D
      2023-07-29 12:49:48.724210  1.304127 -0.041585  1.198302  0.800643
      2023-07-30 12:49:48.724210  0.905371 -0.053619 -2.438710 -1.706876
      2023-07-31 12:49:48.724210  1.155575 -0.599524 -0.020443 -1.341267
      2023-08-01 12:49:48.724210 -0.265322 -1.185095 -0.648073  0.391030
      2023-08-02 12:49:48.724210 -0.150619 -0.444516  1.001217  1.564339
      2023-08-03 12:49:48.724210  0.259234  0.675757 -2.323493 -1.578833
```

```
[27]: # create dataframe 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
```

```
[27]:   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
```

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

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

```
[30]: df2 .head(2)
```

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

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

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

```
[33]: df2.index
```

```
[33]: Index(['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j'], dtype='object')
```

```
[34]: df2.columns
```

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

```
[35]: df2.values
```

```
[35]: 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)
```

```
[37]: df2.describe() #see statitical data of dataframe
```

```
[37]:             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
```

[38]:
```
df2.T
```

[38]:
```
            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    0.5  NaN  5.0  2.0    4.5  NaN  7.0  3.0
visits      1    3      2    3    2    3      1    1    2    1
priority  yes  yes     no  yes   no   no     no  yes   no   no
```

[39]:
```
df2.sort_values(by='age')
```

[39]:
```
  animal  age  visits priority
c  snake  0.5       2       no
f    cat  2.0       3       no
a    cat  2.5       1      yes
b    cat  3.0       3      yes
j    dog  3.0       1       no
g  snake  4.5       1       no
e    dog  5.0       2       no
i    dog  7.0       2       no
d    dog  NaN       3      yes
h    cat  NaN       1      yes
```

[41]:
```
#Slicing dataframe
df2.sort_values(by='age')[1:3]
```

[41]:
```
  animal  age  visits priority
f    cat  2.0       3       no
a    cat  2.5       1      yes
```

[42]:
```
#query dataframe by tag
df2[['age','visits']]
```

[42]:
```
   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
```

[43]:
```
df2.iloc[1:3] #Query rows 2,3
```

[43]:
```
  animal  age  visits priority
b    cat  3.0       3      yes
c  snake  0.5       2       no
```

[44]:
```
df3 = df2.copy()
df3
```

[44]:
```
  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
```

[45]:
```
df3.isnull()
```

[45]:
```
   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
```

[46]:
```
df3.loc['f','age'] = 1.5
df3
```

[46]:
```
  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
```

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

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

```
[49]: df3['visits'].sum()
```

```
[49]: 19
```

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

```
[51]: 0       A
      1       C
      2       D
      3     AAA
      4    BACA
      5     NaN
      6     CBA
      7     COW
      8     OWL
      dtype: object
```

4 Operations for DataFrame missing values

```
[54]: df4 = df3.copy()
      meanAge = df4['age'].mean()
      df4.fillna(4)
```

```
[54]:   animal  age  visits priority
      a    cat  2.5       1      yes
      b    cat  3.0       3      yes
      c  snake  0.5       2       no
      d    dog  4.0       3      yes
      e    dog  5.0       2       no
      f    cat  1.5       3       no
      g  snake  4.5       1       no
      h    cat  4.0       1      yes
      i    dog  7.0       2       no
      j    dog  3.0       1       no
```

```
[55]: df5 = df3.copy()
      df5.dropna(how='any')
```

[55]:
```
  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
```

5 Dataframe file operations

[56]:
```
df3.to_csv('animal.csv')
```

[59]:
```
df_animal = pd.read_csv('animal.csv')
df_animal.head(3)
```

[59]:
```
  Unnamed: 0 animal  age  visits priority
0          a    cat  2.5       1      yes
1          b    cat  3.0       3      yes
2          c  snake  0.5       2       no
```

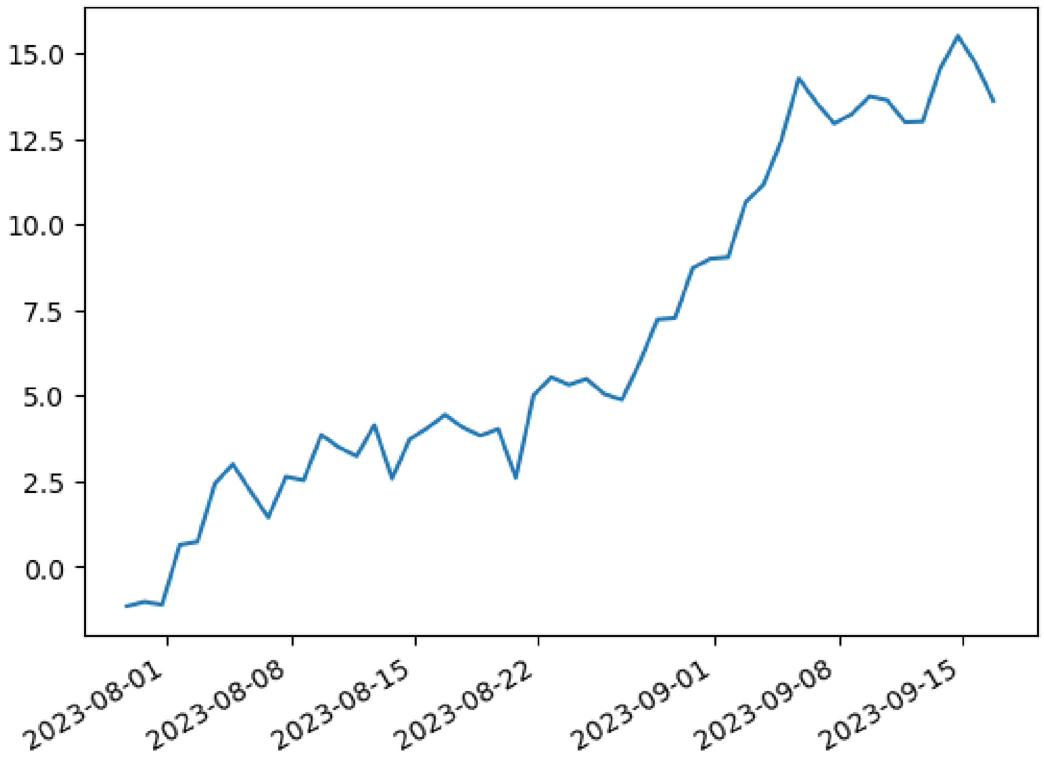
[64]:
```
df3.to_excel('animal.xlsx',sheet_name='Sheet1')
df_animal2 = pd.read_excel('animal.
 ↪xlsx','Sheet1',index_col=None,na_values=['NA'])
df_animal2
```

[64]:
```
  Unnamed: 0 animal  age  visits priority
0          a    cat  2.5       1      yes
1          b    cat  3.0       3      yes
2          c  snake  0.5       2       no
3          d    dog  NaN       3      yes
4          e    dog  5.0       2       no
5          f    cat  1.5       3       no
6          g  snake  4.5       1       no
7          h    cat  NaN       1      yes
8          i    dog  7.0       2       no
9          j    dog  3.0       1       no
```

6 Visualization in Pandas

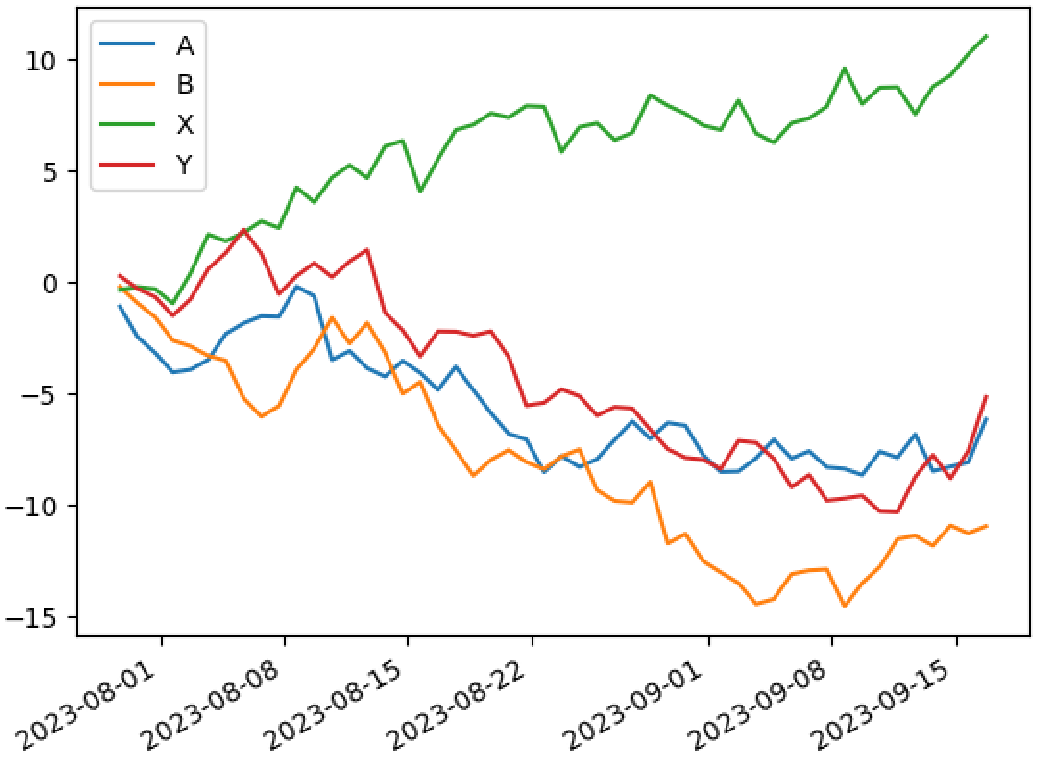
[66]:
```
# Series and dataframe line chart
import numpy as np
%matplotlib inline

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

[66]: <Axes: >



[67]:
```
df = pd.DataFrame(np.random.randn(50,4),index=ts.
↪index,columns=['A','B','X','Y'])
df = df.cumsum()
df.plot()
```

[67]: <Axes: >



7 Pratice example

8 Remove repeated data using pandas

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

```
[69]:     A
      0   1
      1   2
      4   4
      6   5
      8   6
      10  7
      11  8
```

```
[ ]:
```