8. Pandas DataFrame Manipulation Examples

May 20, 2021

1 Working with Columns

1.1 Adding New Columns to a DataFrame

[1]:
```
import numpy as np
import pandas as pd
df = pd.DataFrame(np.random.rand(7, 5))
df
```

[1]:

	0	1	2	3	4
0	0.341013	0.168498	0.559009	0.997305	0.552174
1	0.283351	0.632512	0.114761	0.238375	0.565838
2	0.511632	0.587121	0.223907	0.580531	0.887183
3	0.480951	0.716959	0.718097	0.960732	0.629211
4	0.633921	0.043808	0.201307	0.383901	0.993152
5	0.692759	0.115581	0.999606	0.496316	0.072766
6	0.678211	0.380589	0.415291	0.963392	0.247482

[2]:
```
dfn = df.copy()
dfn.iloc[0,3]=dfn.iloc[2,3]=dfn.iloc[4,4]=np.nan
dfn
```

[2]:

	0	1	2	3	4
0	0.341013	0.168498	0.559009	NaN	0.552174
1	0.283351	0.632512	0.114761	0.238375	0.565838
2	0.511632	0.587121	0.223907	NaN	0.887183
3	0.480951	0.716959	0.718097	0.960732	0.629211
4	0.633921	0.043808	0.201307	0.383901	NaN
5	0.692759	0.115581	0.999606	0.496316	0.072766
6	0.678211	0.380589	0.415291	0.963392	0.247482

[3]:
```
df[0:2]=pd.DataFrame(np.random.randn(2,5))
df
```

[3]:

	0	1	2	3	4
0	1.639172	1.231785	0.213155	0.093764	-0.368986
1	0.805149	0.785367	0.236845	0.417168	-1.640620

```
2  0.511632  0.587121  0.223907  0.580531  0.887183
3  0.480951  0.716959  0.718097  0.960732  0.629211
4  0.633921  0.043808  0.201307  0.383901  0.993152
5  0.692759  0.115581  0.999606  0.496316  0.072766
6  0.678211  0.380589  0.415291  0.963392  0.247482
```

[4]:
```
df.iloc[:,0]=pd.Series([1,2,3,4,5], index=[0,2,3,4,8])
df
```

[4]:
```
     0         1         2         3         4
0  1.0  1.231785  0.213155  0.093764 -0.368986
1  NaN  0.785367  0.236845  0.417168 -1.640620
2  2.0  0.587121  0.223907  0.580531  0.887183
3  3.0  0.716959  0.718097  0.960732  0.629211
4  4.0  0.043808  0.201307  0.383901  0.993152
5  NaN  0.115581  0.999606  0.496316  0.072766
6  NaN  0.380589  0.415291  0.963392  0.247482
```

[5]:
```
df[[0,1]]=np.arange(7)[:,np.newaxis]
df
```

[5]:
```
     0    1         2         3         4
0  0.0  0.0  0.213155  0.093764 -0.368986
1  1.0  1.0  0.236845  0.417168 -1.640620
2  2.0  2.0  0.223907  0.580531  0.887183
3  3.0  3.0  0.718097  0.960732  0.629211
4  4.0  4.0  0.201307  0.383901  0.993152
5  5.0  5.0  0.999606  0.496316  0.072766
6  6.0  6.0  0.415291  0.963392  0.247482
```

[6]:
```
df[[0,1]]=dfn[[2,3]]
df
```

[6]:
```
          0         1         2         3         4
0  0.559009       NaN  0.213155  0.093764 -0.368986
1  0.114761  0.238375  0.236845  0.417168 -1.640620
2  0.223907       NaN  0.223907  0.580531  0.887183
3  0.718097  0.960732  0.718097  0.960732  0.629211
4  0.201307  0.383901  0.201307  0.383901  0.993152
5  0.999606  0.496316  0.999606  0.496316  0.072766
6  0.415291  0.963392  0.415291  0.963392  0.247482
```

[7]:
```
df[0]=np.random.rand(len(df))
df
```

[7]:
```
          0         1         2         3         4
0  0.893249       NaN  0.213155  0.093764 -0.368986
```

```
1  0.138344  0.238375  0.236845  0.417168 -1.640620
2  0.917339       NaN  0.223907  0.580531  0.887183
3  0.162547  0.960732  0.718097  0.960732  0.629211
4  0.913274  0.383901  0.201307  0.383901  0.993152
5  0.393203  0.496316  0.999606  0.496316  0.072766
6  0.519876  0.963392  0.415291  0.963392  0.247482
```

[8]:
```
df[3]=df[0]-df[2]
df
```

[8]:
```
          0         1         2         3         4
0  0.893249       NaN  0.213155  0.680094 -0.368986
1  0.138344  0.238375  0.236845 -0.098502 -1.640620
2  0.917339       NaN  0.223907  0.693432  0.887183
3  0.162547  0.960732  0.718097 -0.555550  0.629211
4  0.913274  0.383901  0.201307  0.711967  0.993152
5  0.393203  0.496316  0.999606 -0.606402  0.072766
6  0.519876  0.963392  0.415291  0.104584  0.247482
```

[9]:
```
df[3]=df[3]*100
df
```

[9]:
```
          0         1         2          3         4
0  0.893249       NaN  0.213155  68.009409 -0.368986
1  0.138344  0.238375  0.236845  -9.850175 -1.640620
2  0.917339       NaN  0.223907  69.343161  0.887183
3  0.162547  0.960732  0.718097 -55.554959  0.629211
4  0.913274  0.383901  0.201307  71.196678  0.993152
5  0.393203  0.496316  0.999606 -60.640230  0.072766
6  0.519876  0.963392  0.415291  10.458439  0.247482
```

[10]:
```
df[2]=np.log(df[2])
df
```

[10]:
```
          0         1         2          3         4
0  0.893249       NaN -1.545735  68.009409 -0.368986
1  0.138344  0.238375 -1.440347  -9.850175 -1.640620
2  0.917339       NaN -1.496523  69.343161  0.887183
3  0.162547  0.960732 -0.331151 -55.554959  0.629211
4  0.913274  0.383901 -1.602924  71.196678  0.993152
5  0.393203  0.496316 -0.000394 -60.640230  0.072766
6  0.519876  0.963392 -0.878775  10.458439  0.247482
```

1.2 Swap Column Contests

[11]:
```
df[[1,2]]=df[[2,1]]
df
```

[11]:

	0	1	2	3	4
0	0.893249	-1.545735	NaN	68.009409	-0.368986
1	0.138344	-1.440347	0.238375	-9.850175	-1.640620
2	0.917339	-1.496523	NaN	69.343161	0.887183
3	0.162547	-0.331151	0.960732	-55.554959	0.629211
4	0.913274	-1.602924	0.383901	71.196678	0.993152
5	0.393203	-0.000394	0.496316	-60.640230	0.072766
6	0.519876	-0.878775	0.963392	10.458439	0.247482

1.3 Deleting Columns

[12]:
```
df.drop(0, axis=1)
```

[12]:

	1	2	3	4
0	-1.545735	NaN	68.009409	-0.368986
1	-1.440347	0.238375	-9.850175	-1.640620
2	-1.496523	NaN	69.343161	0.887183
3	-0.331151	0.960732	-55.554959	0.629211
4	-1.602924	0.383901	71.196678	0.993152
5	-0.000394	0.496316	-60.640230	0.072766
6	-0.878775	0.963392	10.458439	0.247482

[13]:
```
df.drop([0,3], inplace=True)
df
```

[13]:

	0	1	2	3	4
1	0.138344	-1.440347	0.238375	-9.850175	-1.640620
2	0.917339	-1.496523	NaN	69.343161	0.887183
4	0.913274	-1.602924	0.383901	71.196678	0.993152
5	0.393203	-0.000394	0.496316	-60.640230	0.072766
6	0.519876	-0.878775	0.963392	10.458439	0.247482

[14]:
```
df.columns[-1]
```

[14]: 4

[15]:
```
df.drop(df.columns[-1], axis=1)
```

[15]:

	0	1	2	3
1	0.138344	-1.440347	0.238375	-9.850175
2	0.917339	-1.496523	NaN	69.343161
4	0.913274	-1.602924	0.383901	71.196678

```
5  0.393203 -0.000394  0.496316 -60.640230
6  0.519876 -0.878775  0.963392  10.458439
```

[16]:
```
del df[0]
df
```

[16]:
```
          1         2          3         4
1 -1.440347  0.238375  -9.850175 -1.640620
2 -1.496523       NaN  69.343161  0.887183
4 -1.602924  0.383901  71.196678  0.993152
5 -0.000394  0.496316 -60.640230  0.072766
6 -0.878775  0.963392  10.458439  0.247482
```

1.4 Find Duplicates

[17]:
```
df.iloc[2,1]= df.iloc[0,1]
df
```

[17]:
```
          1         2          3         4
1 -1.440347  0.238375  -9.850175 -1.640620
2 -1.496523       NaN  69.343161  0.887183
4 -1.602924  0.238375  71.196678  0.993152
5 -0.000394  0.496316 -60.640230  0.072766
6 -0.878775  0.963392  10.458439  0.247482
```

[18]:
```
df[2].duplicated()
```

[18]:
```
1    False
2    False
4     True
5    False
6    False
Name: 2, dtype: bool
```

[19]:
```
df[~(df[2].duplicated())]
```

[19]:
```
          1         2          3         4
1 -1.440347  0.238375  -9.850175 -1.640620
2 -1.496523       NaN  69.343161  0.887183
5 -0.000394  0.496316 -60.640230  0.072766
6 -0.878775  0.963392  10.458439  0.247482
```

1.5 Common column-wise methods/attributes

[20]:
```
df
```

[20]:
```
          1         2          3         4
1 -1.440347  0.238375  -9.850175 -1.640620
2 -1.496523       NaN  69.343161  0.887183
4 -1.602924  0.238375  71.196678  0.993152
5 -0.000394  0.496316 -60.640230  0.072766
6 -0.878775  0.963392  10.458439  0.247482
```

[21]:
```
df[1].dtype
```

[21]:
```
dtype('float64')
```

[22]:
```
df[1].size
```

[22]:
```
5
```

[23]:
```
df[2].count()
```

[23]:
```
4
```

[24]:
```
df[1].sum()
```

[24]:
```
-5.418962568645374
```

[25]:
```
df[2].max(skipna=False)
```

[25]:
```
nan
```

[26]:
```
df[3].cov(df[2])
```

[26]:
```
-3.6008972304471585
```

[27]:
```
df[3].corr(df[2])
```

[27]:
```
-0.19314727310180055
```

[28]:
```
df.cov()
```

[28]:
```
           1         2            3          4
1   0.445690  0.108890   -31.405407  -0.041800
2   0.108890  0.116877    -3.600897   0.092870
3 -31.405407 -3.600897  3116.194948  34.387256
4  -0.041800  0.092870    34.387256   1.117227
```

[29]:
```
df.corr()
```

[29]:
```
          1         2         3         4
1  1.000000  0.440310 -0.842706 -0.059236
```

```
2  0.440310  1.000000 -0.193147  0.244022
3 -0.842706 -0.193147  1.000000  0.582793
4 -0.059236  0.244022  0.582793  1.000000
```

[30]:
```
dfn[3]
```

[30]:
```
0         NaN
1    0.238375
2         NaN
3    0.960732
4    0.383901
5    0.496316
6    0.963392
Name: 3, dtype: float64
```

[31]:
```
dfn[3].describe()
```

[31]:
```
count    5.000000
mean     0.608543
std      0.335425
min      0.238375
25%      0.383901
50%      0.496316
75%      0.960732
max      0.963392
Name: 3, dtype: float64
```

[32]:
```
dfn[3].unique()
```

[32]:
```
array([       nan, 0.23837462, 0.96073238, 0.38390058, 0.49631606,
       0.96339199])
```

[33]:
```
dfn[3].value_counts()
```

[33]:
```
0.238375    1
0.496316    1
0.960732    1
0.383901    1
0.963392    1
Name: 3, dtype: int64
```

[34]:
```
dfn[3].sample(10, replace=True)
```

[34]:
```
3    0.960732
2         NaN
1    0.238375
0         NaN
```

```
4    0.383901
2         NaN
0         NaN
5    0.496316
0         NaN
0         NaN
Name: 3, dtype: float64
```

1.6 Common column element-wise methods/attributes

[35]:
```
dfn[3].isnull()
```

[35]:
```
0     True
1    False
2     True
3    False
4    False
5    False
6    False
Name: 3, dtype: bool
```

[36]:
```
df[3].round(decimals=2)
```

[36]:
```
1    -9.85
2    69.34
4    71.20
5   -60.64
6    10.46
Name: 3, dtype: float64
```

[37]:
```
df[3].abs()
```

[37]:
```
1     9.850175
2    69.343161
4    71.196678
5    60.640230
6    10.458439
Name: 3, dtype: float64
```

[38]:
```
dfn[3].astype(np.complex64)
```

[38]:
```
0                   NaN
1    0.238375+0.000000j
2                   NaN
3    0.960732+0.000000j
4    0.383901+0.000000j
5    0.496316+0.000000j
```

```
6    0.963392+0.000000j
Name: 3, dtype: complex64
```

[39]:
```
df[1]
```

[39]:
```
1   -1.440347
2   -1.496523
4   -1.602924
5   -0.000394
6   -0.878775
Name: 1, dtype: float64
```

[40]:
```
df[1].diff()
```

[40]:
```
1         NaN
2   -0.056175
4   -0.106401
5    1.602529
6   -0.878380
Name: 1, dtype: float64
```

[41]:
```
df[1].diff(periods=2)
```

[41]:
```
1         NaN
2         NaN
4   -0.162576
5    1.496128
6    0.724149
Name: 1, dtype: float64
```

[42]:
```
df[1].shift()
```

[42]:
```
1         NaN
2   -1.440347
4   -1.496523
5   -1.602924
6   -0.000394
Name: 1, dtype: float64
```

[43]:
```
dfn[3]
```

[43]:
```
0         NaN
1    0.238375
2         NaN
3    0.960732
4    0.383901
5    0.496316
```

```
6    0.963392
Name: 3, dtype: float64
```

[44]:
```
dfn[3].fillna(value=999)
```

[44]:
```
0    999.000000
1      0.238375
2    999.000000
3      0.960732
4      0.383901
5      0.496316
6      0.963392
Name: 3, dtype: float64
```

[45]:
```
dfn[3].replace(np.NaN, 0)
```

[45]:
```
0    0.000000
1    0.238375
2    0.000000
3    0.960732
4    0.383901
5    0.496316
6    0.963392
Name: 3, dtype: float64
```

[46]:
```
dfn[3].pct_change()
```

[46]:
```
0         NaN
1         NaN
2    0.000000
3    3.030347
4   -0.600408
5    0.292824
6    0.941086
Name: 3, dtype: float64
```

[47]:
```
dfcopy=df.copy()
dfcopy['A']=['x1','x2','y1','y2','y3']
dfcopy
```

[47]:

	1	2	3	4	A
1	-1.440347	0.238375	-9.850175	-1.640620	x1
2	-1.496523	NaN	69.343161	0.887183	x2
4	-1.602924	0.238375	71.196678	0.993152	y1
5	-0.000394	0.496316	-60.640230	0.072766	y2
6	-0.878775	0.963392	10.458439	0.247482	y3

[48]:
```
dfcopy.A.str.startswith('x')
```

[48]:
```
1     True
2     True
4    False
5    False
6    False
Name: A, dtype: bool
```

[49]:
```
dfcopy.A.str.upper()
```

[49]:
```
1    X1
2    X2
4    Y1
5    Y2
6    Y3
Name: A, dtype: object
```

2 Working with Rows

2.1 Deleting Rows

[50]:
```
dfx0 = df.append(dfn)
dfx0
```

[50]:

	0	1	2	3	4
1	NaN	-1.440347	0.238375	-9.850175	-1.640620
2	NaN	-1.496523	NaN	69.343161	0.887183
4	NaN	-1.602924	0.238375	71.196678	0.993152
5	NaN	-0.000394	0.496316	-60.640230	0.072766
6	NaN	-0.878775	0.963392	10.458439	0.247482
0	0.341013	0.168498	0.559009	NaN	0.552174
1	0.283351	0.632512	0.114761	0.238375	0.565838
2	0.511632	0.587121	0.223907	NaN	0.887183
3	0.480951	0.716959	0.718097	0.960732	0.629211
4	0.633921	0.043808	0.201307	0.383901	NaN
5	0.692759	0.115581	0.999606	0.496316	0.072766
6	0.678211	0.380589	0.415291	0.963392	0.247482

[51]:
```
dfx = df.append(dfn, ignore_index=True)
dfx
```

[51]:

	0	1	2	3	4
0	NaN	-1.440347	0.238375	-9.850175	-1.640620
1	NaN	-1.496523	NaN	69.343161	0.887183
2	NaN	-1.602924	0.238375	71.196678	0.993152

```
3        NaN -0.000394  0.496316 -60.640230  0.072766
4        NaN -0.878775  0.963392  10.458439  0.247482
5   0.341013  0.168498  0.559009        NaN  0.552174
6   0.283351  0.632512  0.114761   0.238375  0.565838
7   0.511632  0.587121  0.223907        NaN  0.887183
8   0.480951  0.716959  0.718097   0.960732  0.629211
9   0.633921  0.043808  0.201307   0.383901       NaN
10  0.692759  0.115581  0.999606   0.496316  0.072766
11  0.678211  0.380589  0.415291   0.963392  0.247482
```

[52]:
```
dfx.loc[15]=[0.999]*5
dfx
```

[52]:
```
            0         1         2          3         4
0         NaN -1.440347  0.238375  -9.850175 -1.640620
1         NaN -1.496523       NaN  69.343161  0.887183
2         NaN -1.602924  0.238375  71.196678  0.993152
3         NaN -0.000394  0.496316 -60.640230  0.072766
4         NaN -0.878775  0.963392  10.458439  0.247482
5    0.341013  0.168498  0.559009        NaN  0.552174
6    0.283351  0.632512  0.114761   0.238375  0.565838
7    0.511632  0.587121  0.223907        NaN  0.887183
8    0.480951  0.716959  0.718097   0.960732  0.629211
9    0.633921  0.043808  0.201307   0.383901       NaN
10   0.692759  0.115581  0.999606   0.496316  0.072766
11   0.678211  0.380589  0.415291   0.963392  0.247482
15   0.999000  0.999000  0.999000   0.999000  0.999000
```

[53]:
```
dfn.drop(3, axis=0)
```

[53]:
```
           0         1         2         3         4
0   0.341013  0.168498  0.559009       NaN  0.552174
1   0.283351  0.632512  0.114761  0.238375  0.565838
2   0.511632  0.587121  0.223907       NaN  0.887183
4   0.633921  0.043808  0.201307  0.383901       NaN
5   0.692759  0.115581  0.999606  0.496316  0.072766
6   0.678211  0.380589  0.415291  0.963392  0.247482
```

2.2 Common Row Operation

[54]:
```
df[4]
```

[54]:
```
1   -1.640620
2    0.887183
4    0.993152
5    0.072766
```

```
6    0.247482
Name: 4, dtype: float64
```

[55]:
```
np.where( df[4]>=0.5 )
```

[55]:
```
(array([1, 2], dtype=int64),)
```

[56]:
```
np.where( df[4]>=0.5 , 999, 0)
```

[56]:
```
array([  0, 999, 999,   0,   0])
```

[57]:
```
dfx=df.sample(10, replace=True, axis=0)
dfx
```

[57]:

	1	2	3	4
4	-1.602924	0.238375	71.196678	0.993152
4	-1.602924	0.238375	71.196678	0.993152
2	-1.496523	NaN	69.343161	0.887183
4	-1.602924	0.238375	71.196678	0.993152
2	-1.496523	NaN	69.343161	0.887183
6	-0.878775	0.963392	10.458439	0.247482
1	-1.440347	0.238375	-9.850175	-1.640620
2	-1.496523	NaN	69.343161	0.887183
6	-0.878775	0.963392	10.458439	0.247482
5	-0.000394	0.496316	-60.640230	0.072766

[58]:
```
dfx.index.duplicated()
```

[58]:
```
array([False,  True, False,  True,  True, False, False,  True,  True,
       False])
```

3 Working with Whole DataFrame

3.1 Peek at the DataFrame contests/structure

[59]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 5 entries, 1 to 6
Data columns (total 4 columns):
1    5 non-null float64
2    4 non-null float64
3    5 non-null float64
4    5 non-null float64
dtypes: float64(4)
memory usage: 360.0 bytes
```

[60]:
```
df.head(2)
```

[60]:
```
          1         2          3         4
1 -1.440347  0.238375  -9.850175 -1.640620
2 -1.496523       NaN  69.343161  0.887183
```

[61]:
```
dfx.describe()
```

[61]:
```
               1         2          3          4
count  10.000000  7.000000  10.000000  10.000000
mean   -1.249663  0.482371  37.204599   0.456811
std     0.519545  0.341832  46.884636   0.820094
min    -1.602924  0.238375 -60.640230  -1.640620
25%    -1.576323  0.238375  10.458439   0.247482
50%    -1.496523  0.238375  69.343161   0.887183
75%    -1.019168  0.729854  70.733299   0.966660
max    -0.000394  0.963392  71.196678   0.993152
```

3.2 DataFrame utility methods

[62]:
```
dfx.rank(axis=0)
```

[62]:
```
      1    2    3    4
4   2.0  2.5  9.0  9.0
4   2.0  2.5  9.0  9.0
2   5.0  NaN  6.0  6.0
4   2.0  2.5  9.0  9.0
2   5.0  NaN  6.0  6.0
6   8.5  6.5  3.5  3.5
1   7.0  2.5  2.0  1.0
2   5.0  NaN  6.0  6.0
6   8.5  6.5  3.5  3.5
5  10.0  5.0  1.0  2.0
```

[63]:
```
df.sort_values(by=1)
```

[63]:
```
          1         2          3         4
4 -1.602924  0.238375  71.196678  0.993152
2 -1.496523       NaN  69.343161  0.887183
1 -1.440347  0.238375  -9.850175 -1.640620
6 -0.878775  0.963392  10.458439  0.247482
5 -0.000394  0.496316 -60.640230  0.072766
```

[64]:
```
df.sort_values(by=6, axis=1, ascending=False)
```

[64]:
```
           3         2         4         1
1  -9.850175  0.238375 -1.640620 -1.440347
2  69.343161       NaN  0.887183 -1.496523
4  71.196678  0.238375  0.993152 -1.602924
5 -60.640230  0.496316  0.072766 -0.000394
6  10.458439  0.963392  0.247482 -0.878775
```

[65]:
```
dfx.sort_index(axis=0)
```

[65]:
```
           1         2          3         4
1 -1.440347  0.238375  -9.850175 -1.640620
2 -1.496523       NaN  69.343161  0.887183
2 -1.496523       NaN  69.343161  0.887183
2 -1.496523       NaN  69.343161  0.887183
4 -1.602924  0.238375  71.196678  0.993152
4 -1.602924  0.238375  71.196678  0.993152
4 -1.602924  0.238375  71.196678  0.993152
5 -0.000394  0.496316 -60.640230  0.072766
6 -0.878775  0.963392  10.458439  0.247482
6 -0.878775  0.963392  10.458439  0.247482
```

[66]:
```
dfx=dfx.fillna(value=999)
dfx.astype(np.int64)
```

[66]:
```
    1    2   3  4
4 -1    0  71  0
4 -1    0  71  0
2 -1  999  69  0
4 -1    0  71  0
2 -1  999  69  0
6  0    0  10  0
1 -1    0  -9 -1
2 -1  999  69  0
6  0    0  10  0
5  0    0 -60  0
```

3.3 Maths on the whole DataFrame

[67]:
```
df-df.mean()
```

[67]:
```
           1         2          3         4
1 -0.356555 -0.245740 -25.951749 -1.752612
2 -0.412730       NaN  53.241586  0.775190
4 -0.519131 -0.245740  55.095103  0.881159
5  1.083398  0.012202 -76.741805 -0.039227
6  0.205018  0.479278  -5.643135  0.135490
```

[68]:
```
dfn-dfn.shift()
```

[68]:

	0	1	2	3	4
0	NaN	NaN	NaN	NaN	NaN
1	-0.057662	0.464014	-0.444249	NaN	0.013665
2	0.228281	-0.045391	0.109147	NaN	0.321344
3	-0.030682	0.129838	0.494189	NaN	-0.257971
4	0.152970	-0.673150	-0.516789	-0.576832	NaN
5	0.058838	0.071773	0.798299	0.112415	NaN
6	-0.014547	0.265008	-0.584314	0.467076	0.174717